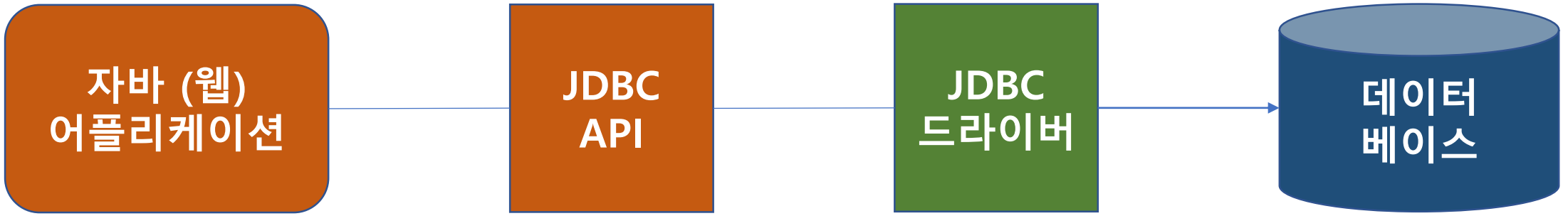
JDBC 란

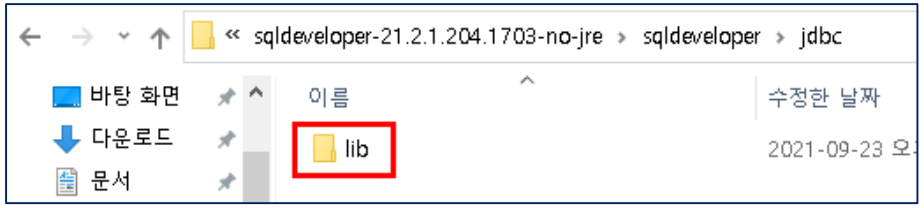
JDBC(Java DataBase Connectivity)는 데이터베이스에 연결 및 작업을 하기 위한 자바 표준 인터페이스이다. 자바는 DBMS(Oracle, MySQL, MongoDB 등)의 종류에 상관 없이 하나의 JDBC API를 이용해서 데이터베이스 작업을 처리한다.

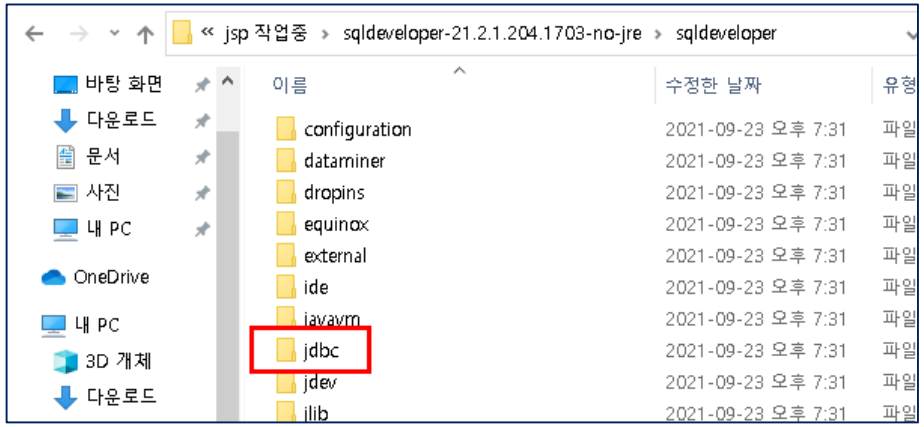
JDBC는 DB에 접근해서 CRUD를 효율적으로 할 수 있게 하는 메소드를 제공한다.

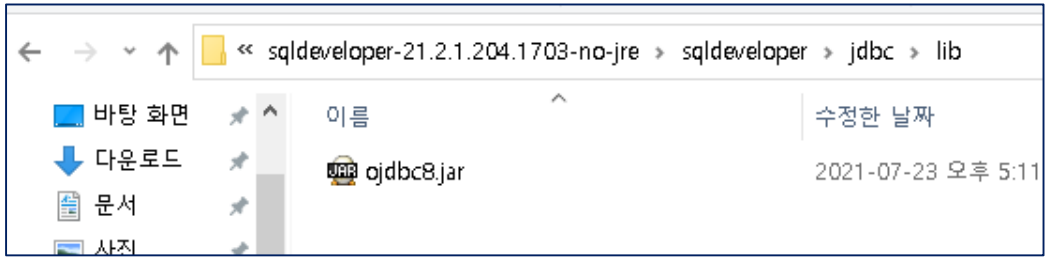


Oracle 드라이버 준비

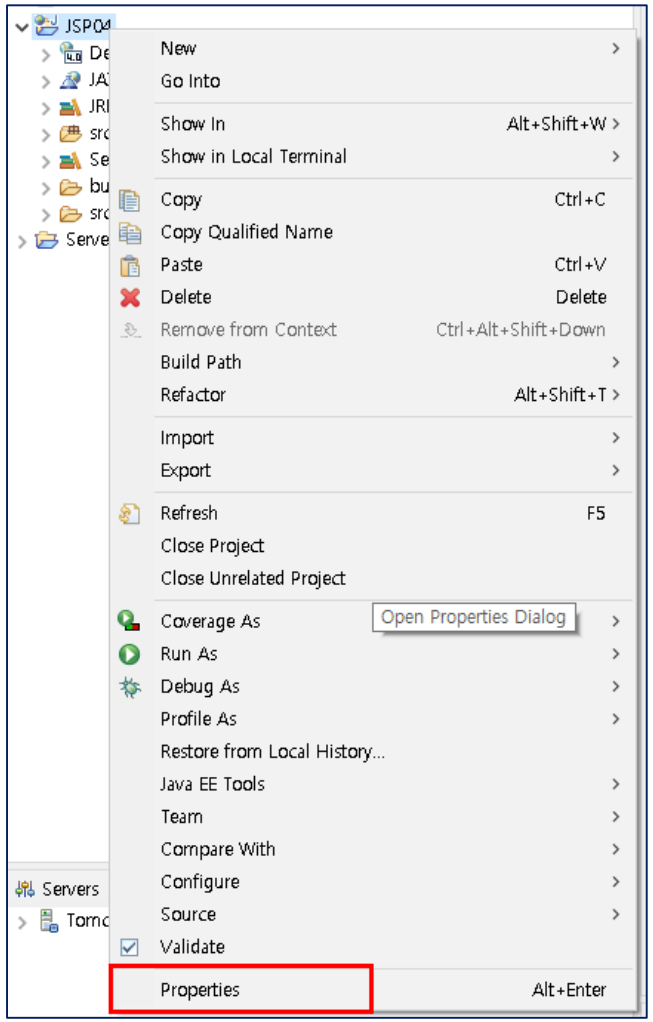
경로 : sqldeveloper-...jre → sqldeveloper → jdbc → lib

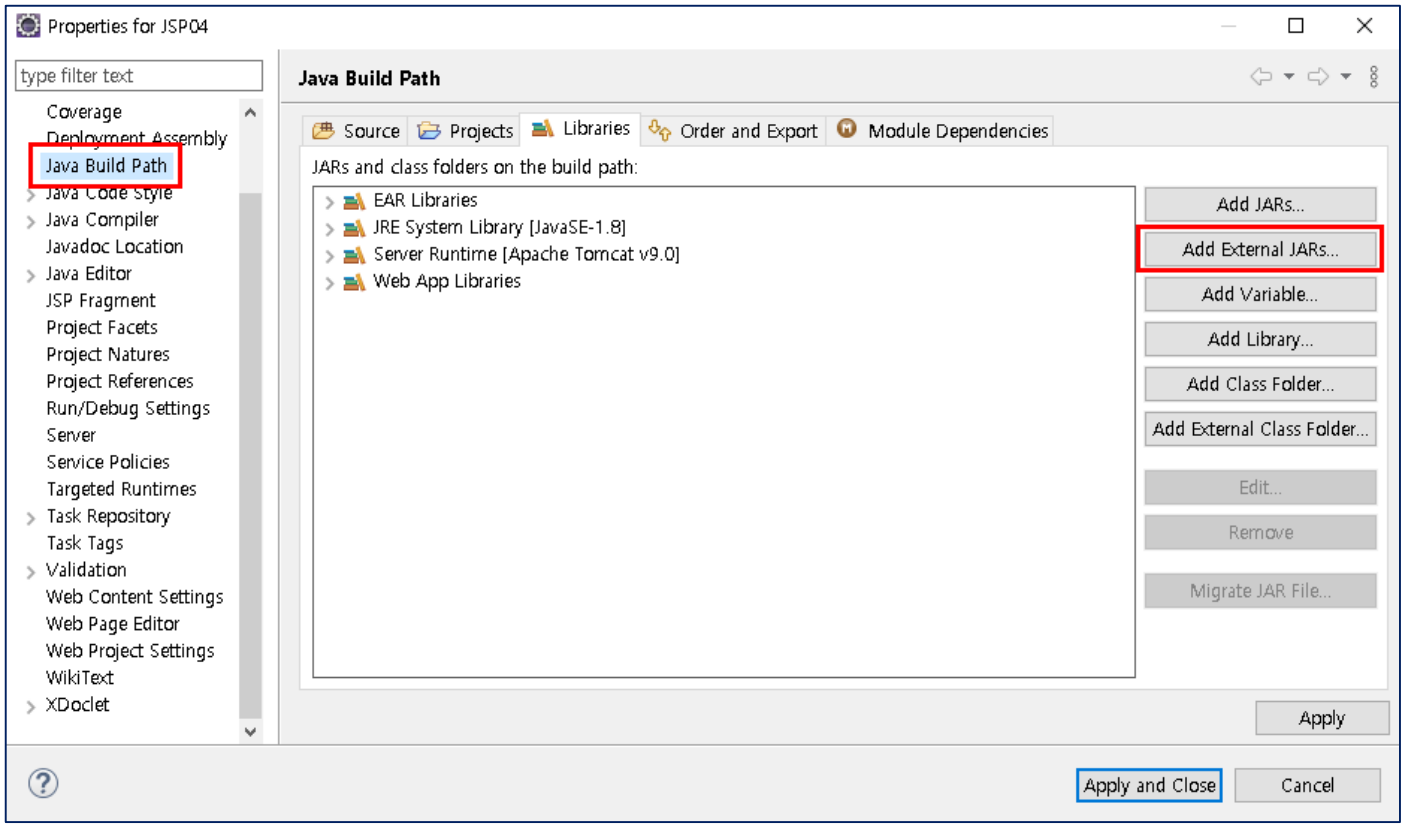




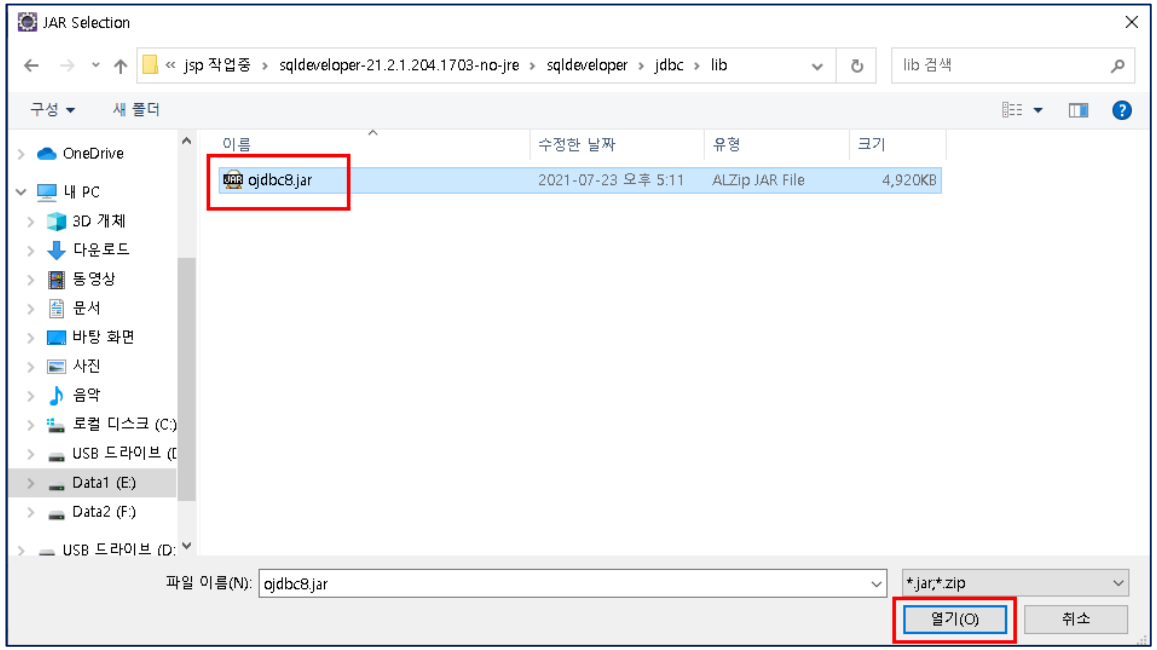


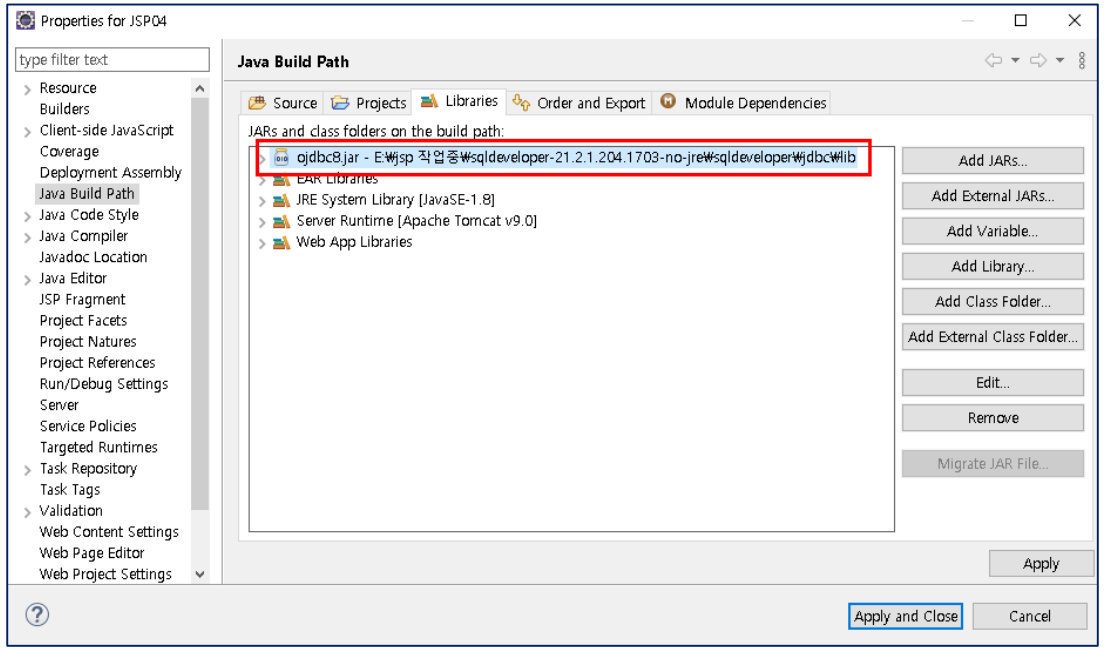
이클립스에 드라이버 추가



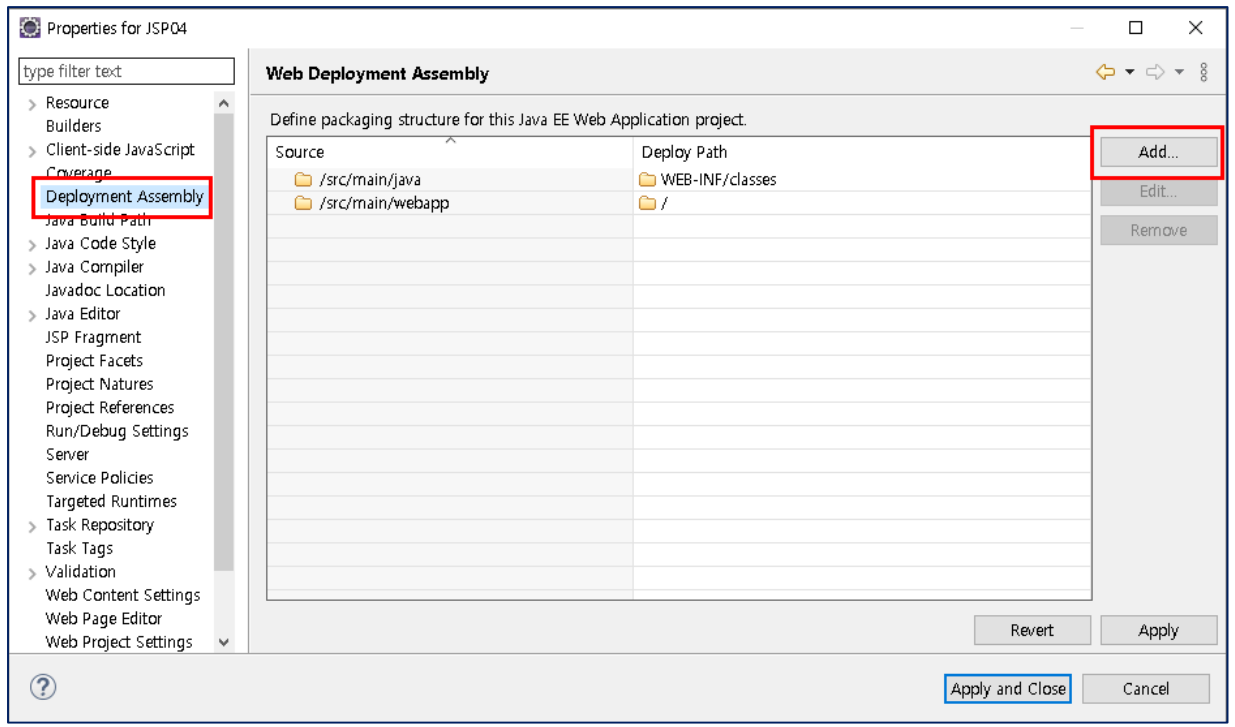


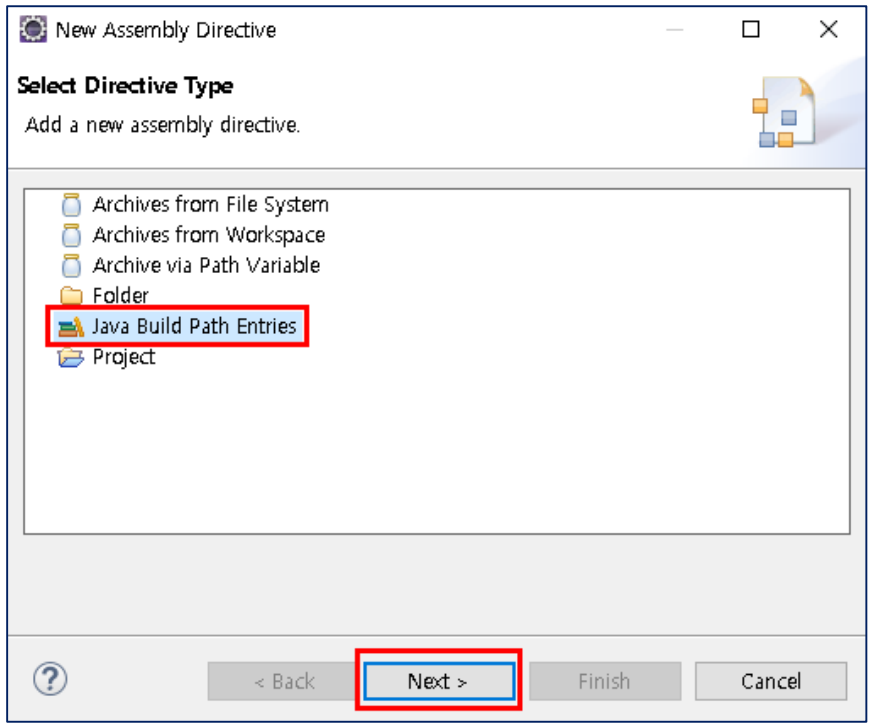
이클립스에 드라이버 추가



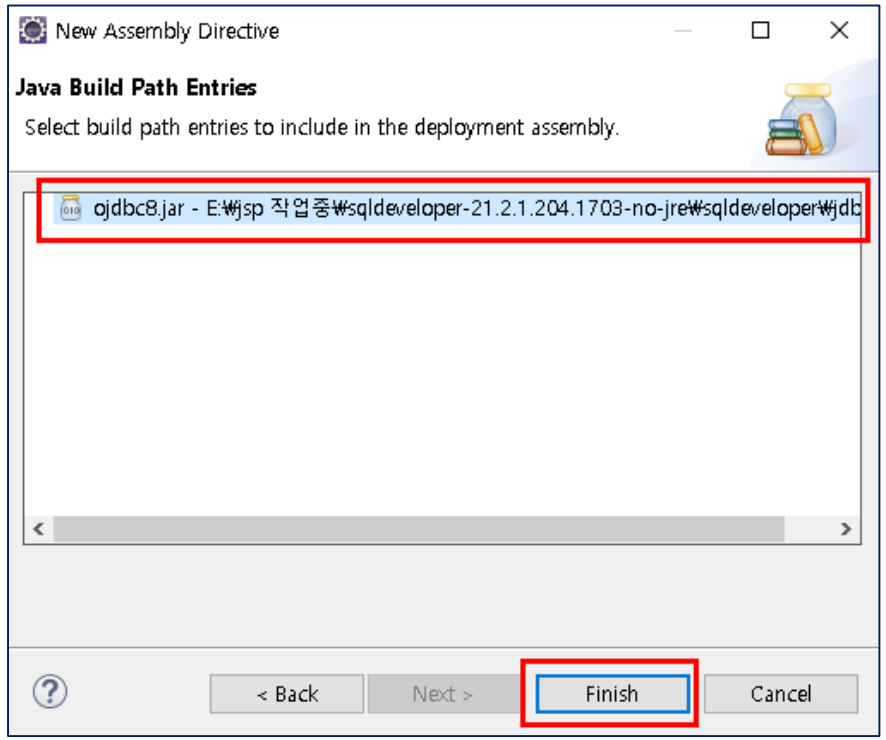


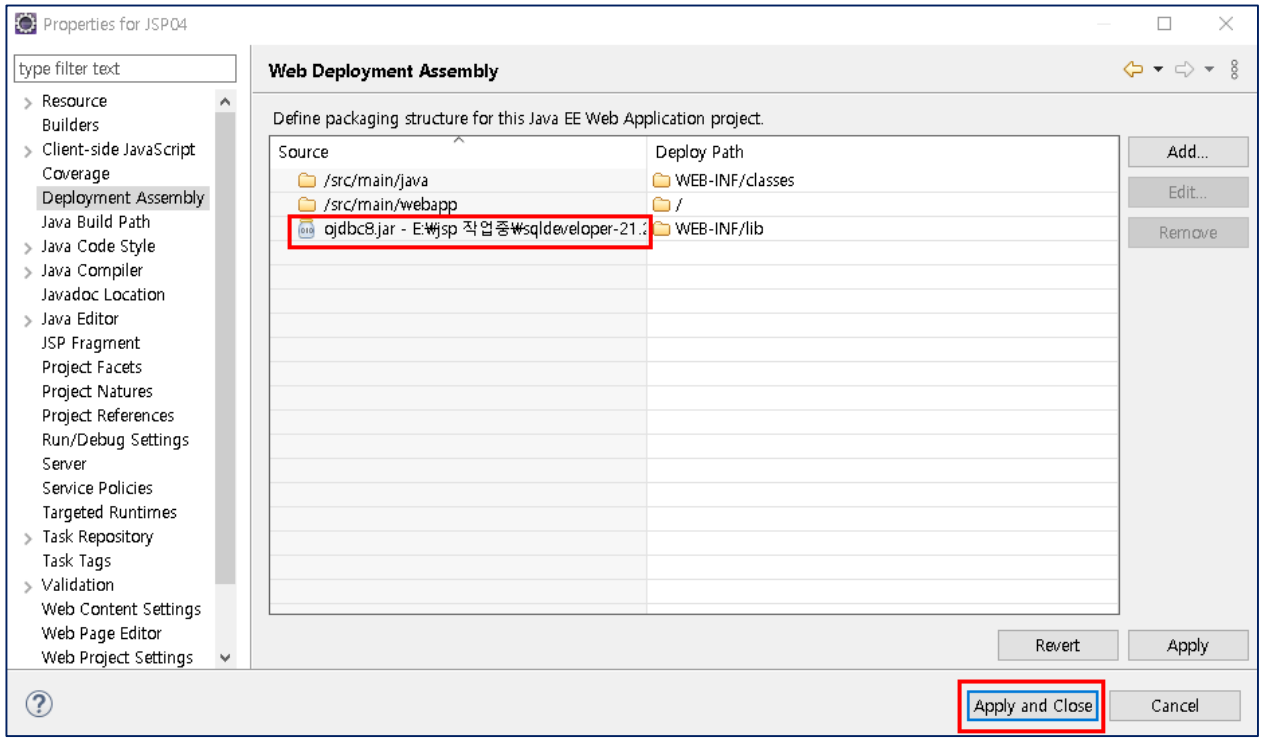
이클립스에 드라이버 추가



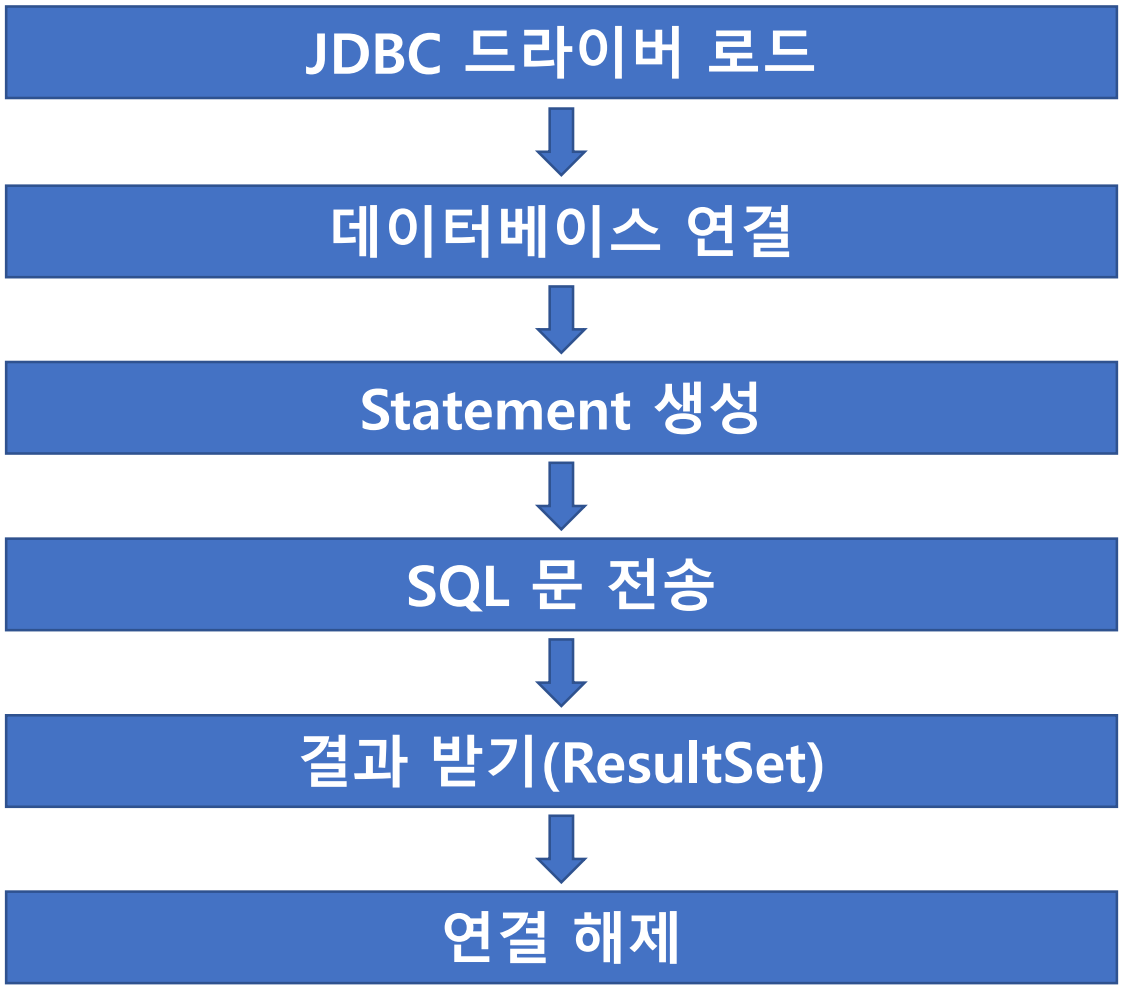


이클립스에 드라이버 추가





JDBC 연결 순서



Class.forName("oracle.jdbc.driver.OracleDriver");

DriverManager.getConnection(URL, ID, PW);

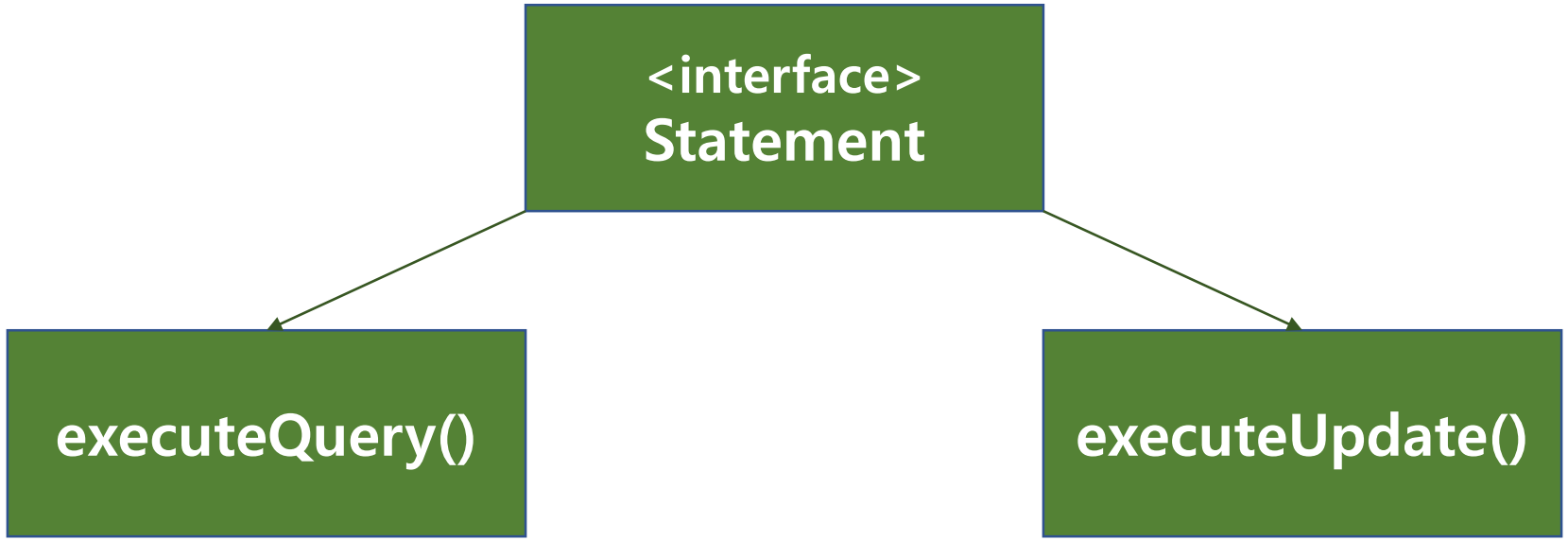
connection.createStatement();

statement.executeQuery()
statement.executeUpdate()

executeQuery() : 결과 값은 ResultSet 객체로 받는다.

close() 메서드를 이용해 닫아 주도록 한다.
사용이 끝난 연결은 반드시 해제해 주어야 한다.

Statement 객체



SQL 문 실행 후 여러 개의 결과값이 생기는 경우 사용

EX) SELECT

SQL 문 실행 후 테이블의 내용만 변경되는 경우 사용

EX) UPDATE, DELETE

excuteQuery( ) / ResultSet

excuteQuery() 메서드는 데이터베이스 검색 전용 메서드인 SELECT 문을 실행 시킬 때 사용하며 결과값을 ResultSet 타입으로 반환한다. SELECT 문을 수행하고 반환된 값을 레코드 셋이라고 부르며 레코드 셋은 여러행으로 구성되어 있다.

여러 행을 한꺼번에 처리할 수는 없고 한 개의 행 단위로 처리할 수 있다.

메소드	설명
next()	현재 행에서 한행 다음으로이동
getter 메소드 (getString, getInt 등)	각 행에서 원하는 컬럼 값을 접근해서 얻으려면 getter 메소드를 사용 (컬럼 이름으로 값을 얻을 수 있다.)

excuteUpdate( ) 메소드

데이터베이스의 테이블의 내용을 변경해 주는 SQL문 (INSERT, UPDATE, DELETE 등)일 경우에 사용한다.

DML(Data Manipulation Language 데이터 조작어)인 INSERT와 UPDATE, DELETE 문인 경우에는 관련된(추가, 변경, 삭제) 레코드의 수를 반환한다.

PreparedStatement

Statement를 상속받은 인터페이스로 SQL구문을 실행하는 기능의 객체이다.

PreparedStatement 는 객체 생성시에 지정된 SQL 명령어만을 실행할 수 있다.
(컴파일시에 실행할 SQL구문이 지정되므로 실행시에 SQL 구문이 변경될 수 없다. - 이는 보안상 유리하다 -)

SQL문에서 변수(데이터)가 들어갈 자리는 ' ? ' 로 표시한 후 setter 메소드를 사용하여 ? 로 표시한 곳에 데이터를 넣어준다.

JSP

DAO 와 DTO

DAO란 Data Access Object의 약자로 데이터베이스의 data에 접근하기 위한 객체.
데이터베이스에 접속해서 데이터 추가, 삭제, 수정 등의 작업을 하는 클래스이다.
데이터베이스 접근을 하기 위한 로직과 비지니스 로직을 분리하기 위해 사용한다.

DTO(Data Transfer Object) 는 계층간 데이터 교환을 위한 자바빈즈.
DAO 클래스를 이용하여 데이터베이스에서 데이터를 관리할 때 일반적인 변수에 할당하여 작업 할수도 있지만, 해당 데이터의 클래스를 만들어 사용한다.

커넥션 풀 (DBCP)

DB를 다룰 때, 사용한 Connection 객체, Statement 객체 같은 자원들을 효율적으로 사용하기 위한 방법이다.

웹 서버가 DB에 접속해야할 때, 너무 많은 요청이 있을 경우 과부하가 있을 수 있다. 그래서 매번 해야하는 작업인 Connection을 미리 만들어놓고 필요할 때마다 가져다 쓰는 것이다.

Connection Pool은 톰캣 컨테이너에 미리 Connection 객체를 만들어 놓고, 필요할 때마다 빌려서 사용하고 반환해주는 것이다.

커넥션 풀(DBCP) 사용 이유

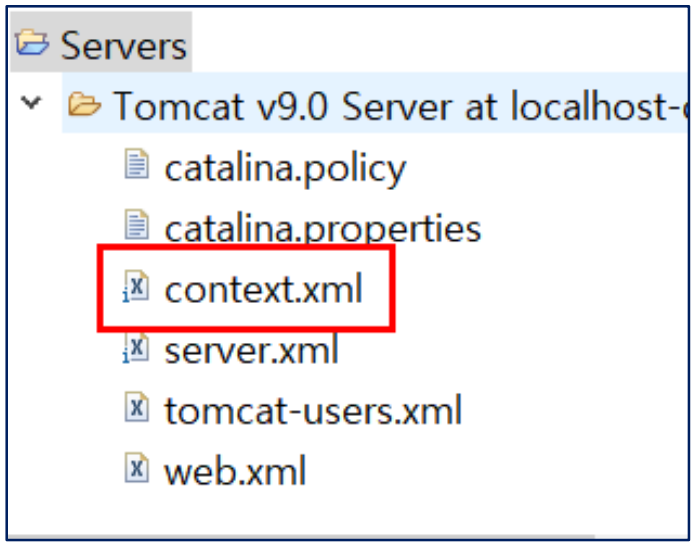
한명의 접속자가 해당 웹 사이트에서 게시판을 확인하고 자신이 쓴 게시물을 수정하고 또 새로운 게시글을 등록한다고 했을 때, DB접속은 아래와 같이 발생한다.

1) 데이터 취득 - 2) 검색 후 데이터 취득 - 3) 데이터 갱신 - 4) 데이터 새등록

한명의 접속자로 단 시간에 4번의 DB 접속이 일어난다. 만약 접속자가 한명이 아니라 1000명 이라면? 서버에 큰 부하가 걸리게 된다. 커넥션 풀이란 미리 커넥션 객체를 생성하고 해당 커넥션 객체를 관리하는것을 의미한다.

커넥션 풀 (DBCP)

Servers에 context.xml



context.xml 에 다음 내용을 작성

```
<Resource auth="Container"
        name="jdbc/oracle"
        driverClassName="oracle.jdbc.driver.OracleDriver"
        type="javax.sql.DataSource"
        url="jdbc:oracle:thin:@localhost:1521:xe"
        username="C##JSPUSER"
        password="jsp123"
        loginTimeout="10"
        maxActive="50"
        maxIdle="20"
        maxWait="2000"
        testOnBorrow="true" />
```