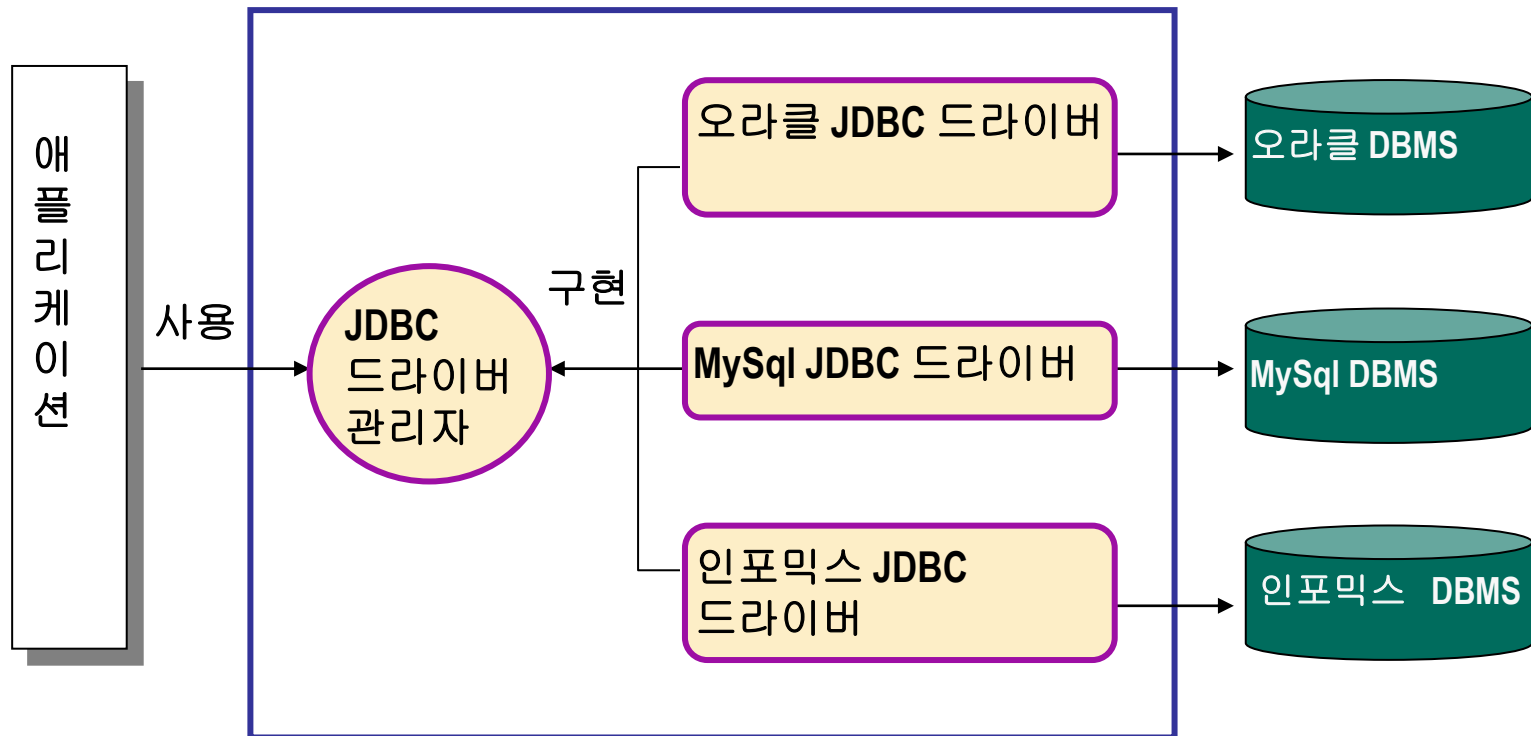




# JDBC

# JDBC 구조

- **JDBC(Java Database Connectivity)**
- 자바에서 데이터베이스를 표준화 된 방법으로 접속할





# JDBC 드라이버 설치

- **JDBC** 드라이버는 사용하고자 하는 데이터베이스 벤더 별로 제공 됨
- 오라클 **JDBC**드라이버

오라클홈 \ app \ oracle \ product \ 10.2.0 \ server \ jdbc \ lib \ ojdbc14.jar

- 생성된 프로젝트에서 사용

**Web1-Mission \ WebContent \ WEB-INF \ lib** 폴더에 복사



# JDBC API

- **import java.sql.\*;**  
JDBC클래스는 **java.sql**패키지, **javax.sql** 패키지 안에 포함되어 있음
- **DriverManager**  
JDBC드라이버를 선택하고 연결객체를 얻어오는 역할을 한다.
- **Connection**  
실제의 데이터베이스와 연결하는 동작을 한다.
- **Statement**  
SQL문을 실행시키는 역할을 한다.
- **ResultSet(select)**  
SQL문의 실행결과인 레코드셋을 담는 클래스이다.

# JDBC 프로그램 순서

**Class.forName(..)**

1단계: **JDBC**드라이버를 로드한다.

**DriverManager  
Connection**

2단계: **SQL**데이터베이스와 연결한다.

**Statement**

3단계: **SQL**쿼리를 실행한다.

**ResultSet**

4단계: 결과를 얻어낸다.

**close**

5단계: 닫는다.

# JDBC 프로그램 단계

- 데이터베이스 드라이버 로드

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

- 데이터베이스 연결

```
Connection conn = DriverManager.getConnection(  
                                JDBC_url,"아이디","비밀번호");
```

**JDBC\_URL 구성 = jdbc:oracle:thin:@IP주소:포트:SID**

**IP 주소** : 오라클이 설치된 컴퓨터의 **IP** 주소 혹은 도메인 이름

**포트** : 리스너의 사용 포트 기본값은 **1521**

**SID** : 오라클 인스턴스 이름으로 **MySQL**에서는 **DB** 이름으로도 불림

# JDBC 프로그램 단계

- **SQL 실행 객체 얻기 및 실행**

**Statement ← PreparedStatement ← CallableStatement**

객체 생성 시 **SQL** 문장을 미리 생성하고 변수 부는 별도의 메서드로 대입하는 방식으로 성능과 관리 면에서 모두 권장 되는 방식임

```
PreparedStatement pstmt = conn.prepareStatement(  
    “insert into test(name, title) values(?,?)”);
```

```
pstmt.setString(1, “길동”);
```

```
pstmt..setString(2, “테스트”);
```

```
int cnt = pstmt.executeUpdate();
```

# JDBC 프로그램 단계

- 결과 받기

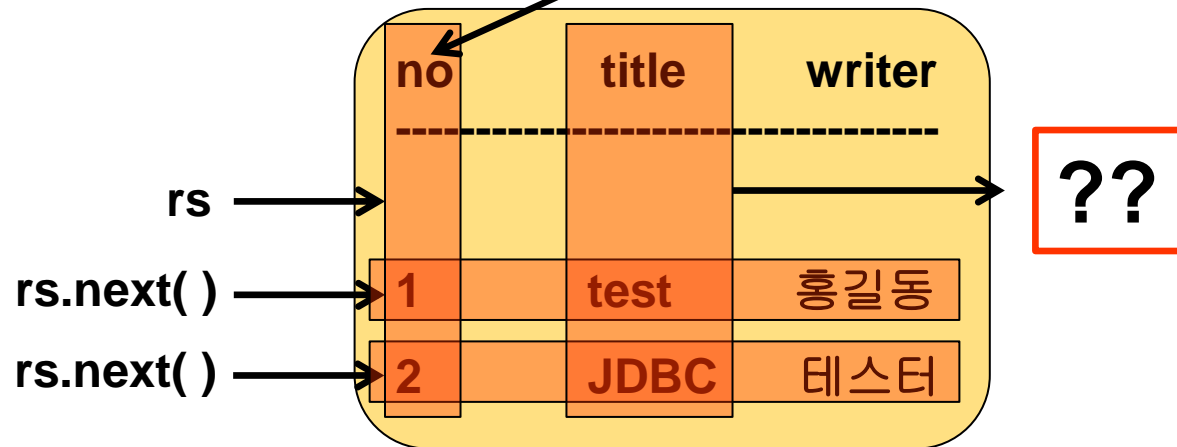
```
ResultSet rs = pstmt.executeQuery( );
```

ResultSet은 커서 개념의 연결 포인터

기본적으로 `next( )` 메서드를 통해 로우 이동

```
select no, title, writer form t_board
```

```
rs.getInt("no")  
rs.getInt(1)
```







# JDBC 프로그램 단계

- 연결해제

**Connection** 을 **close()** 해주지 않으면 사용하지 않는 연결이 유지됨.

**DB** 자원을 낭비하게 됨.

```
conn.close();
```