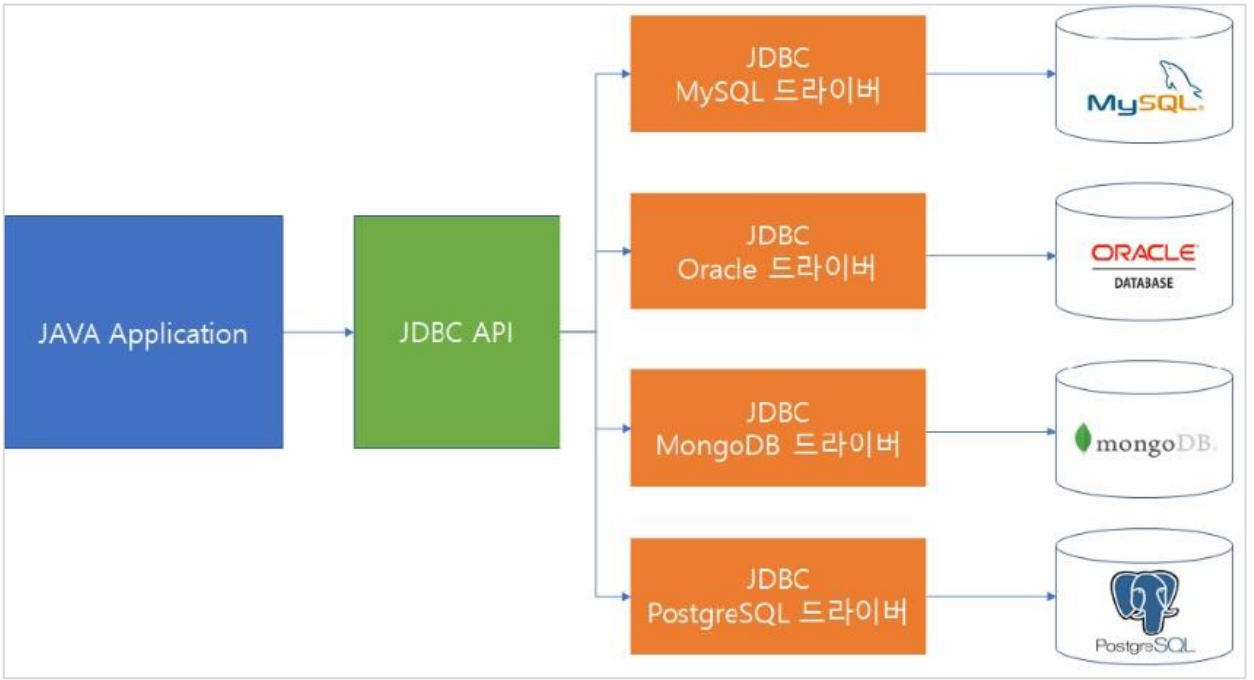
13장. JDBC Connection

오라클 DB 연동

JDBC(Java Database Connectivity)

◆ JDBC(Java Database Connectivity) 정의와 사용

- 자바 애플리케이션에서 DBMS에 연결해주는 기능을 갖는 API이다.
- 오라클, MySQL, MS-SQL 개발사가 구현한 '드라이버'가 존재함

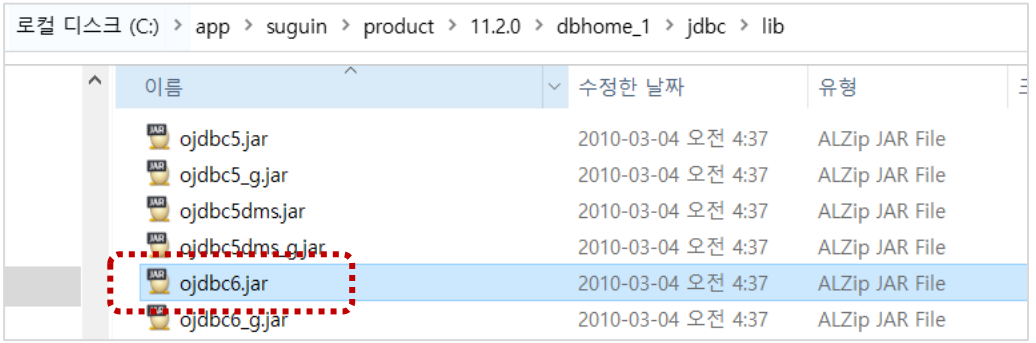


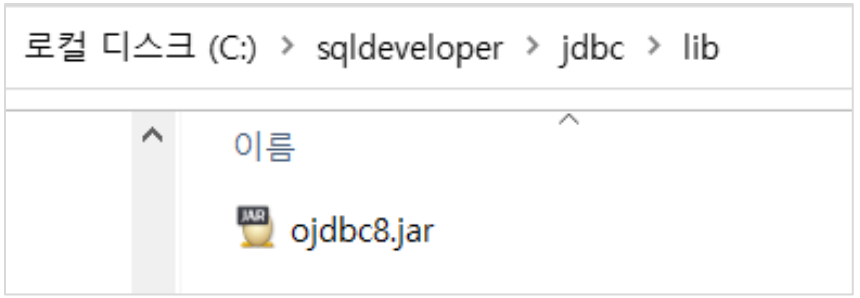
JDBC(Java Database Connectivity)

◆ JDBC를 이용한 데이터베이스 연동하기

ojdbc 드라이버 구하기

1. 오라클 설치 경로 2. sql 디벨로퍼 설치 경로

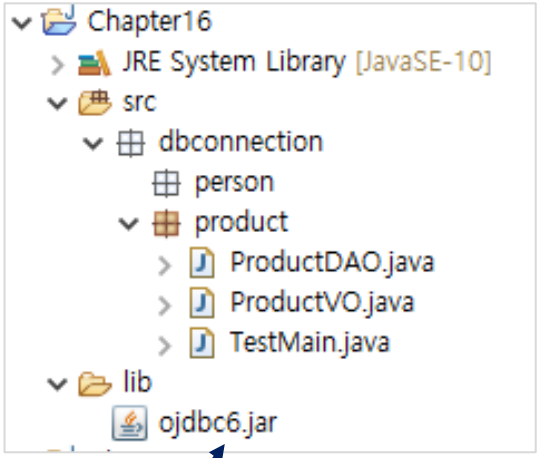


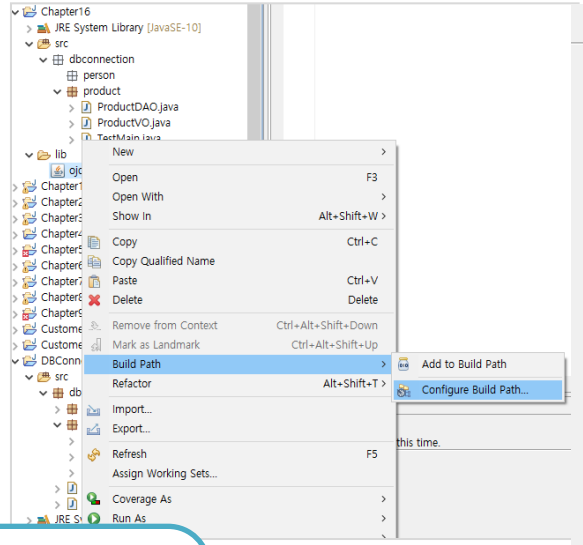


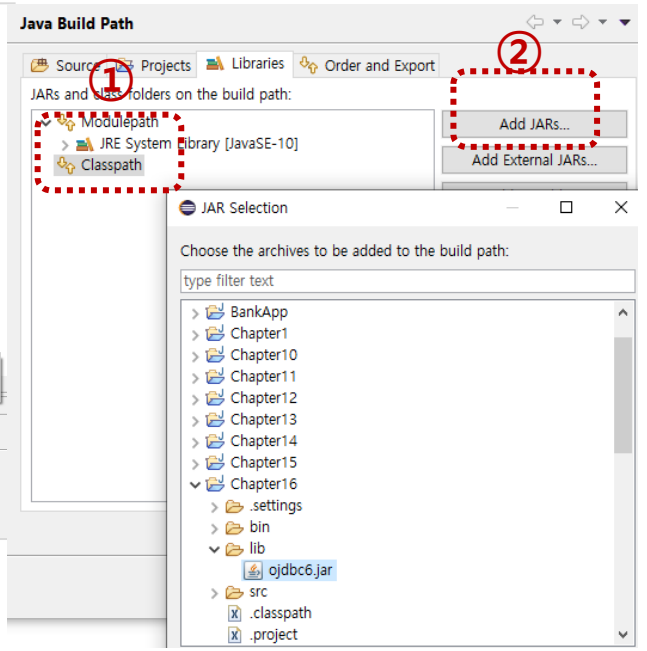
JDBC(Java Database Connectivity)

◆ JDBC를 이용한 데이터베이스 연동하기

ojdbc 이클립스 프로젝트에 복사하기





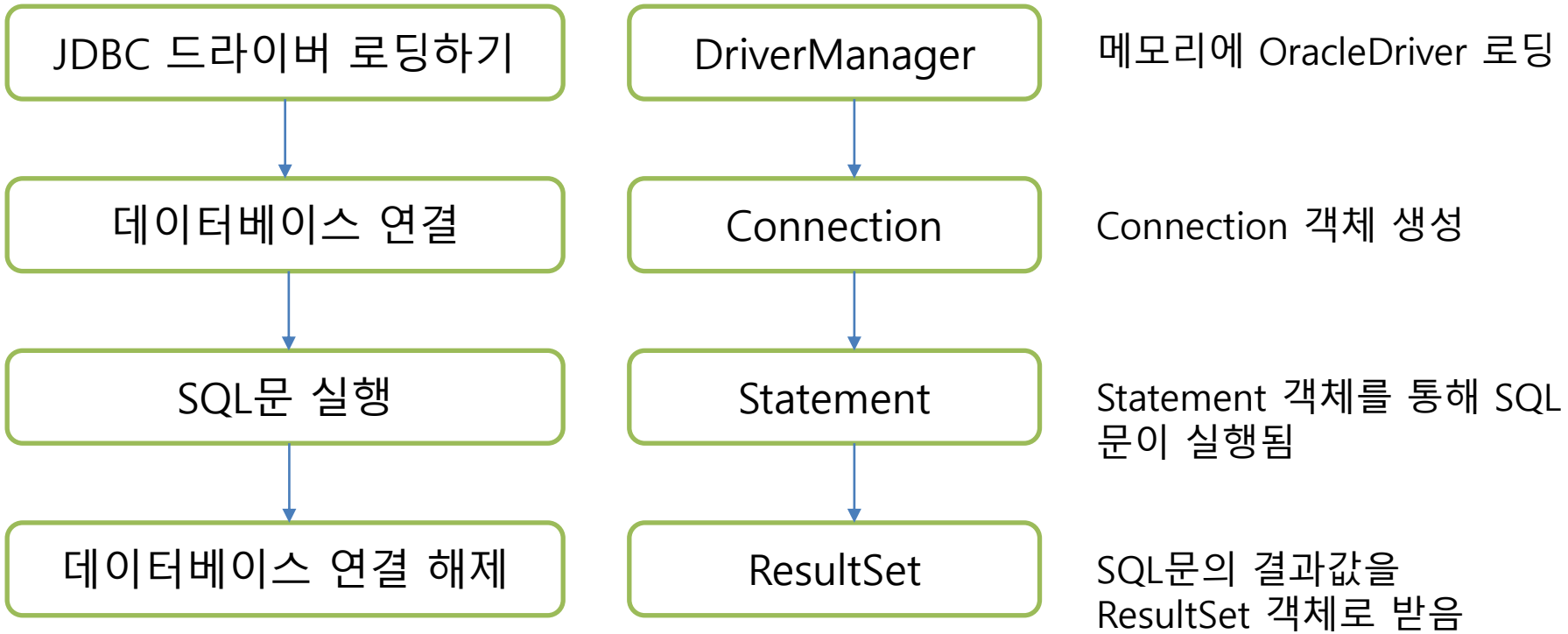


1. 프로젝트 만든후, lib폴더 만들기
2. 오라클 드라이버 .jar파일 복사
3. 클래스 패스 설정

프로젝트 > 우측마우스 > Build Path > Cofigure Build Path > Libraries(Classpath) > Add JARs

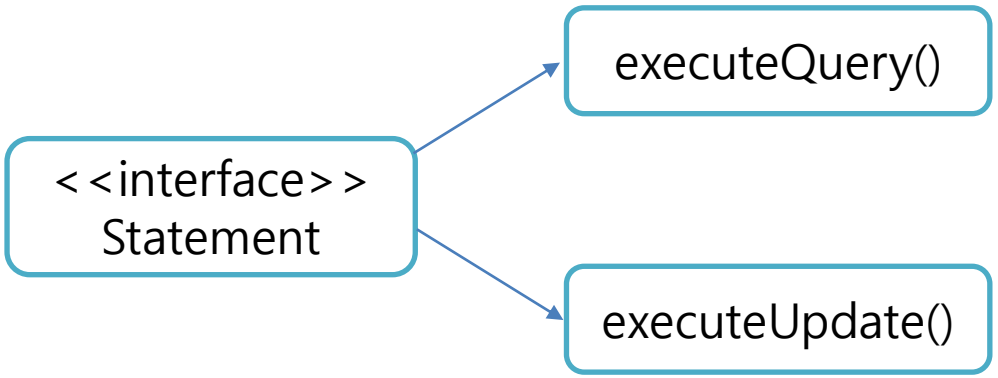
JDBC(Java Database Connectivity)

➢ 데이터베이스 연결 순서



JDBC(Java Database Connectivity)

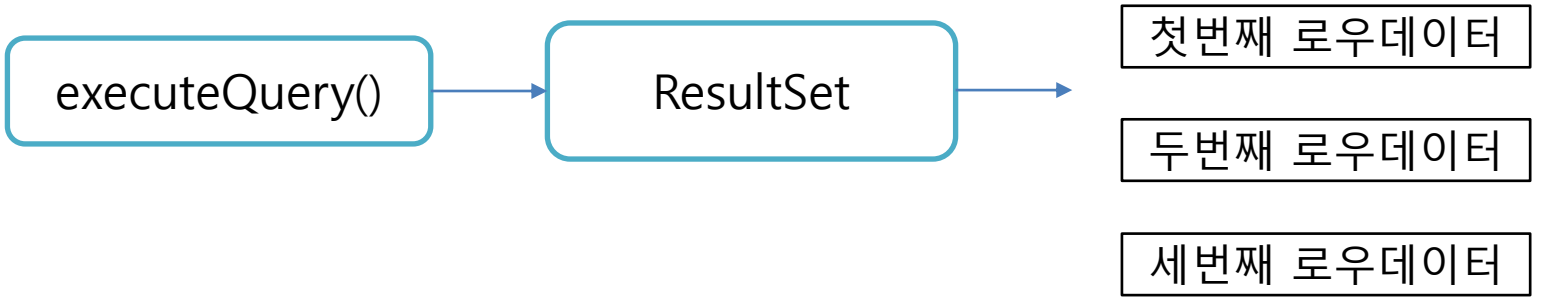
➢ Statement 객체 살펴보기



SQL문 실행 후 여러 개의 결과값 생기는 경우 사용. 예)select

SQL문 실행 후 테이블의 내용만 변경되는 경우 사용. 예)insert, delete, update

executeQuery() 실행 후 반환 되는 레코드셋



연결 테스트(Connection Test)

◆ DB에 연결하기

```java
public class JdbcTest {

    private static String driverClass = "oracle.jdbc.OracleDriver";
    private static String url = "jdbc:oracle:thin:@localhost:1521:xe";
    private static String username = "system";
    private static String password = "12345";

    public static void main(String[] args) {

        Connection conn = null;

        try {
            Class.forName(driverClass);
            conn = DriverManager.getConnection(url, username, password);
            System.out.println("DB 연결 성공!!");
        } catch (ClassNotFoundException e) {
            e.printStackTrace();
        } catch (SQLException e) {
            e.printStackTrace();
        } finally {
            try {
                conn.close();
            } catch (SQLException e) {
                e.printStackTrace();
            }
        }
    }
}
```

오라클 SQL 디벨로퍼

◆ 테이블 만들기(생성)

```
-- Person 테이블 생성 --
CREATE TABLE person(
    userId  VARCHAR2(10),
    userPw VARCHAR2(8) NOT NULL,
    name  VARCHAR2(20) NOT NULL,
    age   NUMBER(3),
    PRIMARY KEY(userId)
);
```

DAO와 VO 정의와 사용법

VO(Value Object)의 정의와 사용법

- 여러 다른 타입의 데이터를 다른 클래스로 전달할 때 사용
- 만드는 방법

> DB 테이블의 필드명을 속성으로 선언한다.
> 생성자를 구현한다.
> 각 속성에 대한 getter/setter 메서드를 구현한다.

DAO(Data Access Object)의 정의와 사용법

- 자바 프로그램에서 데이터베이스 작업만 수행하는 코드
- 하나의 클래스 안에 코드가 많아져서 개발이나 유지관리가 힘들어진다.
- 화면기능, 데이터베이스 연동 기능 등을 각각 담당하는 클래스로 나누어 프로그램을 구현한다.

DAO와 VO 정의와 사용법

PersonVO.java

```java
package dbconnection;

public class Person {
    private String userId;
    private String userPw;
    private String name;
    private int age;

    public String getUserId() {
        return userId;
    }
    public void setUserId(String userId) {
        this.userId = userId;
    }
    public String getUserPw() {
        return userPw;
    }
    public void setUserPw(String userPw) {
        this.userPw = userPw;
    }
    public String getName() {
        return name;
    }
    public void setName(String name) {
        this.name = name;
    }
    public int getAge() {
        return age;
    }
    public void setAge(int age) {
        this.age = age;
    }
```

DAO와 VO 정의와 사용법

```java
public class PersonDAO {
    private String driverClass = "oracle.jdbc.OracleDriver";
    private String url = "jdbc:oracle:thin:@localhost:1521:xe";
    private String username = "system";
    private String password = "12345";

    private Connection conn = null;
    private PreparedStatement pstmt = null;
    private ResultSet rs = null;

    public void connDB() {
        try {
            Class.forName(driverClass);
            conn = DriverManager.getConnection(url, username, password);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
```

DAO와 VO 정의와 사용법

종료 메서드

```
//연결 종료 메서드
private void disconnect() {
    if(pstmt != null) {
        try {
            pstmt.close();
        } catch (SQLException e) {
            e.printStackTrace();
        }
    }
    if(conn != null) {
        try {
            conn.close();
        } catch (SQLException e) {
            e.printStackTrace();
        }
    }
}
```

```
//연결 종료 메서드
private void disconnectRS() {
    if(rs != null) {
        try {
            rs.close();
        } catch (SQLException e) {
            e.printStackTrace();
        }
    }
    if(pstmt != null) {
        try {
            pstmt.close();
        } catch (SQLException e) {
            e.printStackTrace();
        }
    }
    if(conn != null) {
        try {
            conn.close();
        } catch (SQLException e) {
            e.printStackTrace();
        }
    }
}
```

DAO와 VO 정의와 사용법

자료 삽입 메서드 정의

```
// 사람 추가
public void create(Person person) {
    String userId = person.getUserId();
    String userPw = person.getUserPw();
    String name = person.getName();
    int age = person.getAge();

    connDB();
    String sql = "INSERT INTO person VALUES (?, ?, ?, ?)";
    try {
        pstmt = conn.prepareStatement(sql);
        pstmt.setString(1, userId);
        pstmt.setString(2, userPw);
        pstmt.setString(3, name);
        pstmt.setInt(4, age);
        pstmt.executeUpdate();
    } catch (SQLException e) {
        e.printStackTrace();
    } finally {
        disconnect();
    }
}
```

DAO와 VO 정의와 사용법

- 전체 자료 조회

```java
// 전체 목록 조회
public ArrayList<Person> getListAll(){
    ArrayList<Person> list = new ArrayList<>();
    connDB();
    String sql = "SELECT * FROM person";
    try {
        pstmt = conn.prepareStatement(sql);
        rs = pstmt.executeQuery();
        while(rs.next()) {
            String id = rs.getString("userId");
            String pw = rs.getString("userPw");
            String name = rs.getString("name");
            int age = rs.getInt("age");

            Person person = new Person();
            person.setUserId(id);
            person.setUserPw(pw);
            person.setName(name);
            person.setAge(age);

            list.add(person);
        }
    } catch (SQLException e) {
        e.printStackTrace();
    } finally {
        disconnectRS();
    }
    return list;
}
```

DAO와 VO 정의와 사용법

특정한 자료 조회 : 상세 보기

```
//1명 상세 보기
public Person getOne(String userId) {
    Person person = new Person();
    connDB();
    String sql = "SELECT * FROM person WHERE userId = ?";
    try {
        pstmt = conn.prepareStatement(sql);
        pstmt.setString(1, userId);
        rs = pstmt.executeQuery();
        if(rs.next()) {
            person.setUserId(rs.getString("userId"));
            person.setUserPw(rs.getString("userPw"));
            person.setName(rs.getString("name"));
            person.setAge(rs.getInt("age"));
        }

    } catch (SQLException e) {
        e.printStackTrace();
    }
    return person;
}
```

DAO와 VO 정의와 사용법

- 자료삭제

```
//삭제
public boolean delete(String userId) {
    connDB();
    String sql = "DELETE FROM person WHERE userId = ?";
    try {
        pstmt = conn.prepareStatement(sql);
        pstmt.setString(1, userId);
        pstmt.executeUpdate();
    } catch (SQLException e) {
        e.printStackTrace();
        return false;
    }
    return true;
}
```

DAO와 VO 정의와 사용법

LoginTest.java

- 로그인

```
Scanner scan = new Scanner(System.in);
PersonDAO dao = new PersonDAO();

System.out.print("아이디 입력 : ");
String uid = scan.nextLine();

System.out.print("비밀번호 입력 : ");
String pwd = scan.nextLine();

ArrayList<Person> list = dao.getListAll();

boolean access = false;  //로그인 성공 여부
for(int i = 0; i < list.size(); i++) {
    Person p = list.get(i);
    String dbId = p.getUserId();
    String dbPw = p.getUserPw();

    if(dbId.equals(uid) && dbPw.equals(pwd)){
        System.out.println("로그인에 성공했습니다.");
        access = true;
        break;
    }
}
if(access==false) {
    System.out.println("아이디나 비밀번호가 일치하지 않습니다.");
}
scan.close();
```

DAO와 VO 정의와 사용법

PersonMain.java

```
public class Main {

    public static void main(String[] args) {
        PersonDAO dao = new PersonDAO();
        Person p1 = new Person();

        //생성
        p1.setUserId("kim");
        p1.setUserPw("k1234567");
        p1.setName("김산");
        p1.setAge(29);

        dao.create(p1);
```

DAO와 VO 정의와 사용법

PersonMain.java

```
// 목록 조회
ArrayList<Person> list = dao.getListAll();

for(int i = 0; i < list.size(); i++) {
    Person p = list.get(i);
    String id = p.getUserId();
    String pw = p.getUserPw();
    String name = p.getName();
    int age = p.getAge();

    System.out.println(id + ", " + pw + ", " + name + ", " + age);
}

/*// 1명 보기
Person p = dao.getOne("kim");
System.out.println("아이디 : " + p.getUserId());
System.out.println("비밀번호 : " + p.getUserPw());
System.out.println("이름 : " + p.getName());
System.out.println("나이 : " + p.getAge());

//삭제
//dao.delete("park");*/
```