

# Spring Framework

## 12. Spring JDBC 환경설정

# CONTENTS

1

DB 설정 및 JDBC Driver 설치

2

Spring JDBC 설치 및 DataSource 설정

3

사용자관리 프로젝트 테스트

# 학습 목표

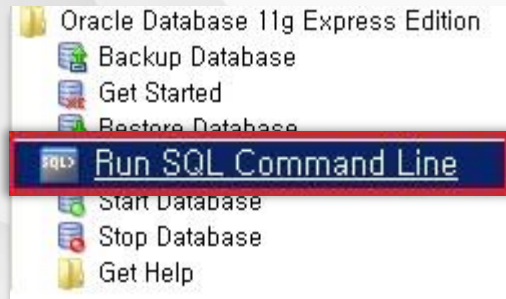
- DB 설정 및 JDBC Driver 설치에 대하여 이해할 수 있습니다.
- Spring JDBC 설치 및 DataSource 설정에 대하여 이해할 수 있습니다.
- 사용자관리 프로젝트 테스트에 대하여 이해할 수 있습니다.

A person's hands are shown holding a smartphone, with the screen glowing. The background is dark with out-of-focus, colorful bokeh lights in shades of yellow, orange, and blue. A semi-transparent dark banner is at the bottom, containing a yellow decorative element and the title text.

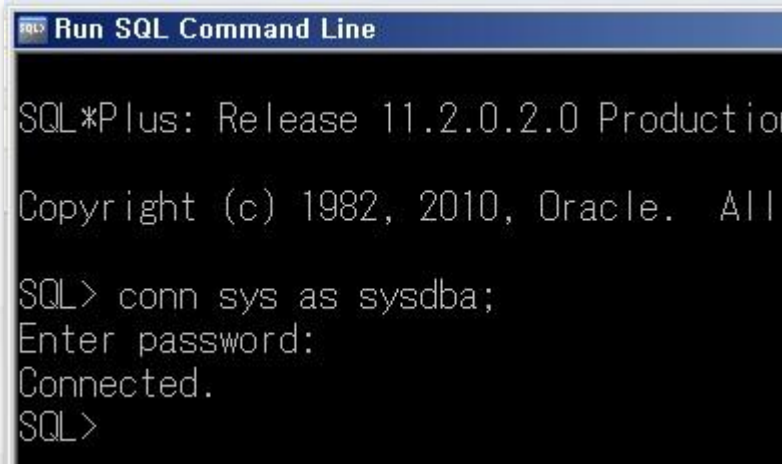
## 1. DB 설정 및 JDBC Driver 설치

## ■ DBA 권한으로 접속

### SQL Command 실행



### DBA권한으로 접속



Password는 설치할 때 지정한 oracle11

## ■ DB scott 계정 생성

### scott 계정 생성

```
SQL> create user scott identified by tiger  
2  default tablespace users  
3  temporary tablespace temp;  
  
User created.
```

### scott 계정에 권한 주기

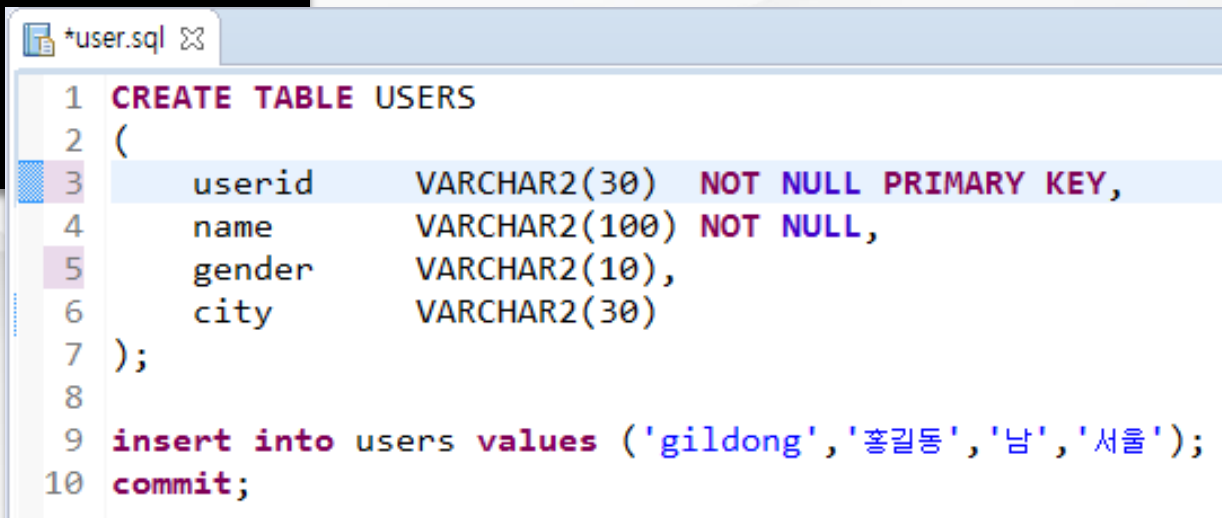
```
SQL> grant connect, resource to scott;  
  
Grant succeeded.
```

## ■ DB users 테이블 생성

scott 계정으로 접속 후 users 테이블 생성

```
SQL> conn scott/tiger
Connected.
SQL> start c:\springframework\user.sql;

Table created.
1 row created.
Commit complete.
```



```
*user.sql x
1 CREATE TABLE USERS
2 (
3     userid      VARCHAR2(30) NOT NULL PRIMARY KEY,
4     name        VARCHAR2(100) NOT NULL,
5     gender      VARCHAR2(10),
6     city        VARCHAR2(30)
7 );
8
9 insert into users values ('gildong', '홍길동', '남', '서울');
10 commit;
```

## ■ Oracle Jdbc Driver 라이브러리 검색 및 설치

<http://mvnrepository.com>에 접근한다.



oracle ojdbc6 로 검색한다.



oracle jdbc 12.1.0.1 버전을 pom.xml에 추가한다.



## ■ Oracle Jdbc Driver 라이브러리 검색 및 설치



The screenshot shows the Maven Repository search interface. The search bar contains 'oracle ojdbc6' and the 'Search' button is visible. Below the search bar, it says 'Found 195 results'. The first result is highlighted with a red border. It shows a package icon with '{h}' inside, followed by the text '2. Oracle JDBC Driver For Java Oracle.jdbc.driver.OracleDriver Ojdbc6' and 'com.hynnet » oracle-driver-ojdbc6'. Below this, it says 'Oracle JDBC Driver For Java Oracle.jdbc.driver.OracleDriver Ojdbc6'.

```
<!-- https://mvnrepository.com/artifact/com.hynnet/oracle-driver-ojdbc6 -->
<dependency>
  <groupId>com.hynnet</groupId>
  <artifactId>oracle-driver-ojdbc6</artifactId>
  <version>12.1.0.1</version>
</dependency>
```

A person's hands are holding a smartphone, with the screen glowing. The background is dark with out-of-focus, colorful bokeh lights in shades of yellow, orange, and blue. A semi-transparent dark banner with a yellow decorative element on the left is positioned at the bottom.

## 2. Spring JDBC 설치 및 DataSource 설정

### ■ Spring JDBC 설치

<http://mvnrepository.com>에 접근한다.

`spring jdbc` 로 검색한다.

`spring jdbc 3.2.17` 버전을 `pom.xml`에 추가한다.


```
<!-- https://mvnrepository.com/artifact/org.springframework/spring-jdbc -->
<dependency>
  <groupId>org.springframework</groupId>
  <artifactId>spring-jdbc</artifactId>
  <version>3.2.17.RELEASE</version>
</dependency>
```

### DataSource 설정

- DataSource를 Spring Bean으로 등록하여 사용할 수 있다.

```
*beans.xml
1
2
3
4
5
6
7
8 <context:property-placeholder
9     location="classpath:config/value.properties" />
10
11 <bean id="dataSource"
12     class="org.springframework.jdbc.datasource.SimpleDriverDataSource">
13     <property name="driverClass" value="${db.driverClass}" />
14     <property name="url" value="${db.url}" />
15     <property name="username" value="${db.username}" />
16     <property name="password" value="${db.password}" />
17 </bean>
```

```
value.properties
1 db.driverClass=oracle.jdbc.OracleDriver
2 db.url=jdbc:oracle:thin:@127.0.0.1:1521:orcl
3 db.username=scott
4 db.password=tiger
```

A person's hands are shown holding a smartphone, with the screen glowing. The background is dark with out-of-focus, colorful bokeh lights in shades of yellow, orange, and blue. A semi-transparent dark banner is at the bottom, containing a yellow decorative bar and the title text.

### 3. 사용자관리 프로젝트 실행

## ■ 사용자관리 프로젝트의 Bean 등록 및 의존관계 설정

### ❖ <context:component-scan> 태그 사용

- ◉ @Service, @Repository 어노테이션을 선언한 클래스들과 @Autowired 어노테이션을 선언하여 의존관계를 설정한 클래스들이 위치한 패키지를 Scan하기 위한 설정을 XML에 해주어야 한다.

\*beans.xml

10

11 <context:component-scan base-package="myspring.user" />

12

## DataSource 설정 테스트

```
beans.xml
26 <bean id="dataSource"
27     class="org.springframework.jdbc.datasource.SimpleDriverDataSource">
28     <property name="driverClass" value="${db.driverClass}" />
29     <property name="url" value="${db.url}" />
30     <property name="username" value="${db.username}" />
31     <property name="password" value="${db.password}" />
32 </bean>
```

```
public class UserClient {
    @Test
    public void dataSourceTest() {
        DataSource ds = (DataSource) context.getBean("dataSource");
        try {
            System.out.println(ds.getConnection());
        } catch (SQLException e) {
            e.printStackTrace();
        }
    }
}
```

실행 결과

oracle.jdbc.driver.T4CConnection@19d31f3

#### ■ 사용자 조회 테스트

```
UserClient.java
35
36 @Autowired
37 UserService service;
38
39 @Test
40 public void getUserTest() {
41     UserVO user = service.getUser("gildong");
42     System.out.println(user);
43     assertEquals("홍길동", user.getName());
44 }
45
```

#### 실행 결과

```
Markers Console
<terminated> UserClient (2) [JUnit] C:\Program Files (x86)\Java\jre1.8.0_91\bin\javaw.e
User [userId=gildong, name=홍길동, gender=남, city=서울]
```



#### ■ 사용자 등록 및 목록조회 테스트

```
UserClient.java
35
36 @Autowired
37 UserService service;
38
39 @Test
40 public void insertUserTest() {
41     service.insertUser(new UserVO("dooly", "둘리", "남", "경기"));
42
43     for (UserVO user : service.getUserList()) {
44         System.out.println(user);
45     }
46 }
```

#### 실행 결과

```
Markers Console Maven Repositories JUnit Properties Data
<terminated> UserClient (2) [JUnit] C:\Program Files (x86)\Java\jre1.8.0_91\bin\javaw.
등록된 Record UserId=dooly Name=둘리
User [userId=dooly, name=둘리, gender=남, city=경기]
User [userId=gildong, name=홍길동, gender=남, city=서울]
```

#### ■ 사용자 정보수정 테스트

```
UserClient.java
36 @Autowired
37 UserService service;
38
39 @Test
40 public void updateUserTest() {
41     service.updateUser(new UserVO("dooly", "김둘리", "여", "부산"));
42
43     UserVO user = service.getUser("dooly");
44     System.out.println(user);
45 }
```

#### 실행 결과

```
Markers Console Maven Repositories JUnit Properties
<terminated> UserClient (2) [JUnit] C:\Program Files (x86)\Java\jre1.8.0_91\bin\j
갱신된 Record with ID = dooly
User [userId=dooly, name=김둘리, gender=여, city=부산]
```

#### ■ 사용자 정보삭제 테스트

```
UserClient.java
--
36 @Autowired
37 UserService service;
38
39 @Test
40 public void deleteUserTest() {
41     service.deleteUser("dooly");
42
43     for (UserVO user : service.getUserList()) {
44         System.out.println(user);
45     }
46 }
```

#### 실행 결과

```
Markers Console Maven Repositories JUnit Properties Da
<terminated> UserClient (2) [JUnit] C:\Program Files (x86)\Java\jre1.8.0_91\bin\java
삭제된 Record with ID = dooly
User [userId=gildong, name=홍길동, gender=남, city=서울]
```



학습정리

지금까지 **[Spring JDBC 환경설정]**에 대해서 살펴보았습니다.

## DB 설정 및 JDBC Driver 설치

- ◉ DB 접속 계정 생성 및 테이블 생성
- ◉ Oracle JDBC Driver 설치

## Spring JDBC 설치 및 DataSource 설정

- ◉ Spring JDBC 라이브러리 설치
- ◉ DataSource 설정

## 사용자관리 프로젝트 테스트

- ◉ 사용자 관리 프로젝트 실행
- ◉ 사용자 조회, 등록, 목록조회, 수정, 삭제 테스트