

# 15장. JDBC Connection

오라클 DB 연동



# 데이터와 정보

- 데이터베이스는 데이터(data)와 베이스(base)의 합성어이다.
  - 데이터는 어떤 필요에 의해 수집했지만 아직 특정 목적을 위해 정제되지 않은 값, 사실 또는 자료를 의미한다.
  - 정보는 수집된 데이터를 어떤 목적을 위해 분석, 가공하여 가치를 추가하거나 새로운 의미를 부여한 결과이다.

A카드사는 최근 몇 년간 급증한 커피 소비 동향을 파악하기 위해 A카드사에서 발급한 카드를 사용한 커피 전문점 결제 내역을 성별과 나이 대 별로 분류하였다.

이 분류 작업과 관련하여 A카드사는 커피 전문점 결제 분포에서 20대 또는 30대 여성이 압도적으로 우위에 있을 것이라 예상했다.

커피 전문점 결제 분포의 최상위 순위를 30~40대 남성이 차지하고 있었던 것이다. 20~30대 여성의 결제 비율을 가볍게 넘어설 정도의 차이가 벌어진 것은 아니지만 예상을 뒤엎는 결과였다.



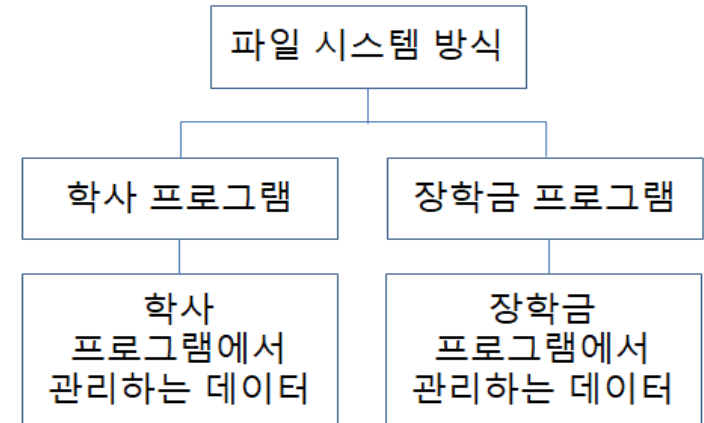
# 파일 시스템과 DBMS

## 학사 프로그램

학번	이름	학과	상태
2019-0001	홍길동	컴퓨터공학과	군휴학
2019-0002	이순신	경영학과	졸업
2019-0003	유관순	철학과	재학

## 장학금 신청 프로그램

장학금	이름	상태	가능여부
국가	홍길동	군휴학	신청불가
성적	이순신	재학	신청가능
근로	유관순	재학	신청가능



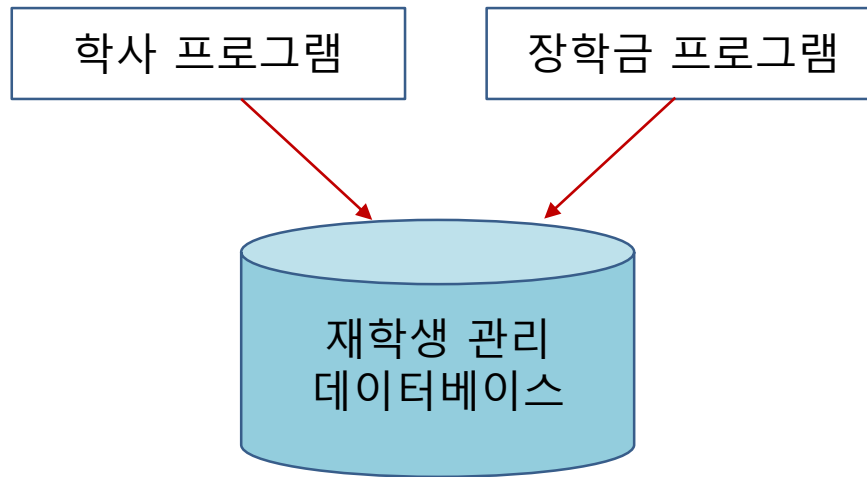
파일 시스템은 서로 다른 여러 응용 프로그래밍 제공하는 기능에 맞게 필요한 데이터를 각각 저장하고 관리한다. 따라서 각 파일에 저장한 데이터는 서로 연관이 없고 중복 또는 누락이 발생할 수 있다.

예) 엑셀, 워드 프로그램



# 파일 시스템과 DBMS

- DBMS(데이터베이스 관리 시스템)
  - 데이터베이스의 조작과 관리를 극대화한 시스템 소프트웨어이다.
  - 데이터베이스 관리 시스템은 보통 '디비엠스(DBMS:DataBase Management System)'라고 부르며, '디비' 또는 '데이터베이스'라고도 부른다.
  - 대표적으로 오라클 , MySQL, MSSQL, 엑세스 등이 있다.



학생과 관련된 일련의 데이터를 한곳에 모아 관리하면 데이터의 오류, 누락, 중복 등의 문제를 해결할 수 있다.

예) ERP(전사적 자원관리)  
인사, 회계, 생산관리 프로그램



# 데이터 모델

- 데이터 모델

- 컴퓨터에 데이터를 저장하는 방식을 정의해 놓은 개념 모형이다.
- 계층형 데이터 모델, 네트워크형 데이터 모델, 관계형 데이터 모델, 객체 지향형

- 데이터 모델링(Data Modeling)

- 데이터 베이스의 설계시 클라이언트의 요구를 분석하여 논리모델을 구성하고 물리모델을 사용해 데이터 베이스에 반영하는 작업
- 기본 요소

구분	개념	실제 예
엔티티(Entity)	물리적 개념에서는 테이블로 표현	고객, 상품, 주문
속성(Attribute)	물리적 개념에서는 칼럼(Column)으로 표현	고객아이디, 고객명, 주소
관계(Releationship)	기본키와 참조키로 정의 됨(일대일, 일대다)	고객과 주문과의 관계



# 관계형 데이터베이스

- 관계형 데이터 모델

- 데이터간의 관계에 초점을 둔 모델로 현재 가장 많이 사용하는 모델이다.
- 예) 회사의 직원정보, 소속된 부서정보 데이터 관리
- 직원 정보와 부서 정보를 하나의 묶음으로 관리하면 데이터 구조가 간단해진다. 하지만 같은 부서 직원들은 부서 정보가 중복되므로 효율적인 관리가 어려워진다. 왜냐하면 부서 이름이 바뀌면 직원들의 부서 정보를 일일이 찾아서 수정해주어야 한다.

직원 정보	직원 번호	직원 이름	직원 직급	부서이름	위치
직원 번호	0001	홍길동	과장	회계팀	서울
직원 이름	0002	성춘향	대리	연구소	수원
직원 직급	0003	이몽룡	사원	영업팀	분당
부서이름	0004	심청이	사원	회계팀	서울
위치					

데이터 중복발생

※ 정규화 전의 형태



# 관계형 데이터베이스



※ 정규화 후의 형태 -> 1대 多의 구조로 변경된다.

새로운 엔티티(테이블)로 중복 그룹을 분리해 주어야 한다.



# 관계형 데이터베이스

- 관계형 데이터베이스의 구성 요소

- 테이블(Table)

표 형태의 데이터 저장 공간을 테이블이라고 한다. 2차원 형태로 행과 열로 구성

행(ROW)

저장하려는 하나의 개체를 구성하는 여러 값을 가로로 늘어뜨린 형태다.

열(COLUMN)

저장하려는 데이터를 대표하는 이름과 공통 특성을 정의

- 특별한 의미를 지닌 열 - 키

기본키(Primary Key)

- 테이블의 지정된 행을 식별할 수 있는 유일한 값이어야 한다.
- 값의 중복이 없어야 한다.
- NULL값을 가질 수 없다.

보조키

- 대체키 또는 후보키라 하며 후보키 중에서 기본키로 지정되지 않은 열이다.





# 관계형 데이터베이스

- 특별한 의미를 지닌 열 - 키

외래키(FK : Foreign Key)

- 특정 테이블에 포함되어 있으면서 다른 테이블의 기본키로 지정된 키



# SQL이란?

- SQL(Structured Query Language)

- '에스큐엘', 또는 '시퀄'이라 부른다.
- 사용자와 데이터베이스 시스템 간에 의사 소통을 하기 위한 언어이다.
- 사용자가 SQL을 이용하여 DB 시스템에 데이터의 검색, 조작, 정의등을 요구하면 DB 시스템이 필요한 데이터를 가져와서 결과를 알려준다.

구분	개념
DDL(Data Definition Language) - 데이터 정의어	테이블을 포함한 여러 객체를 생성, 수정, 삭제하는 명령어
DML(Data Manipulation Language) - 데이터 조작어	데이터를 저장, 검색, 수정, 삭제하는 명령어
DCL(Data Control Language) 데이터 제어어	데이터 사용 권한과 관련된 명령어



# 오라클 데이터 베이스

## ❖ 오라클 데이터베이스와 버전

### ▷ Oracle 데이터베이스

- 오라클사가 만든 DBMS 제품이다.
- 최신 버전은 2021년에 출시한 21c 버전이다.

현재 18c XE(Express Edition)를 무료로 제공하고 있다.

### ▷ Oracle 데이터베이스 설치

1. 계정 생성하고 로그인 하기
2. 다운로드 하기
3. 파일 압축 풀기
4. 설치 프로그램 실행하기



# 오라클 데이터 베이스

❖ 오라클 데이터베이스와 설치 -> 검색 : 오라클 DB 다운로드

데이터베이스

[Database 21c Client](#)

[Database 19c Enterprise/Standard Editions](#)

[Database 18c Express Edition](#)

[Audit Vault and Database Firewall](#)

[Berkeley DB](#)

[Big Data Connectors](#)

## Oracle Database Express Edition (XE) Release 18.4.0.0.0 (18c)

Download

Description



[Oracle Database 18c Express Edition for Linux x64](#)

(2,521,766,408 bytes -  
[Sha256sum:  
4df0318d72a0b97f54c



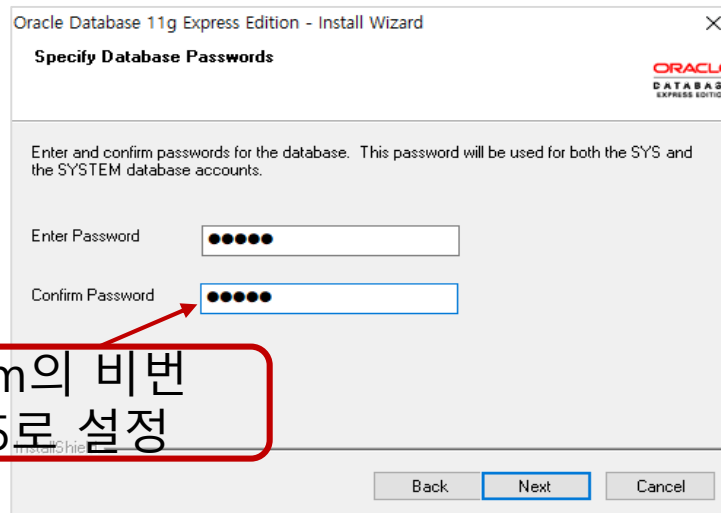
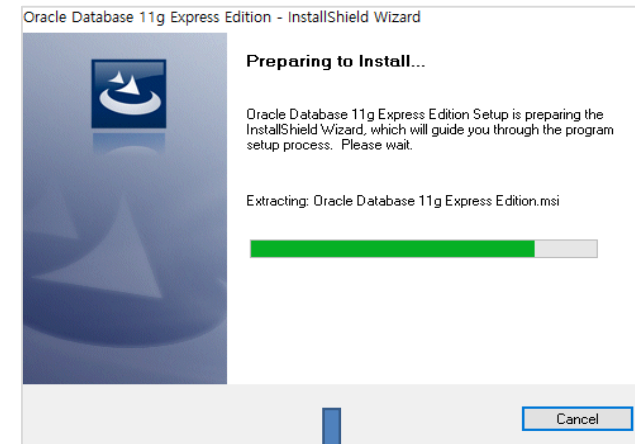
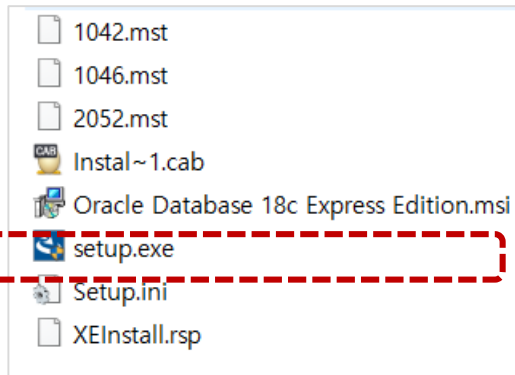
[Oracle Database 18c Express Edition for Windows x64](#)

(2,054,264,100 bytes -  
[Sha256sum:  
7C1A85D05F6FCC5BE

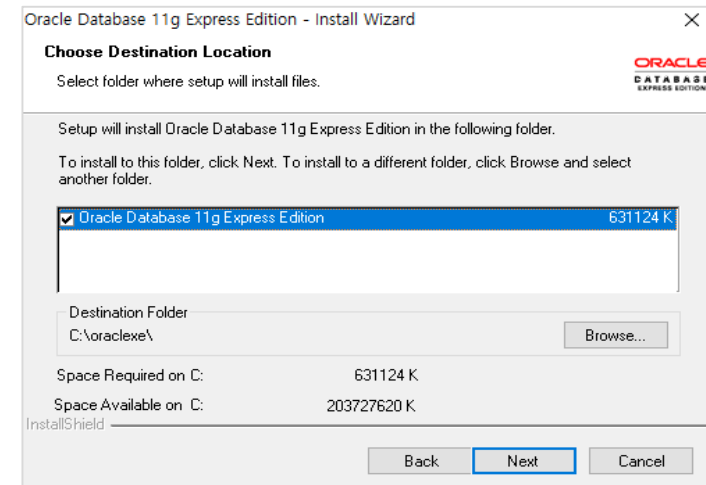


# 오라클 데이터 베이스

## ❖ 오라클 데이터베이스와 설치

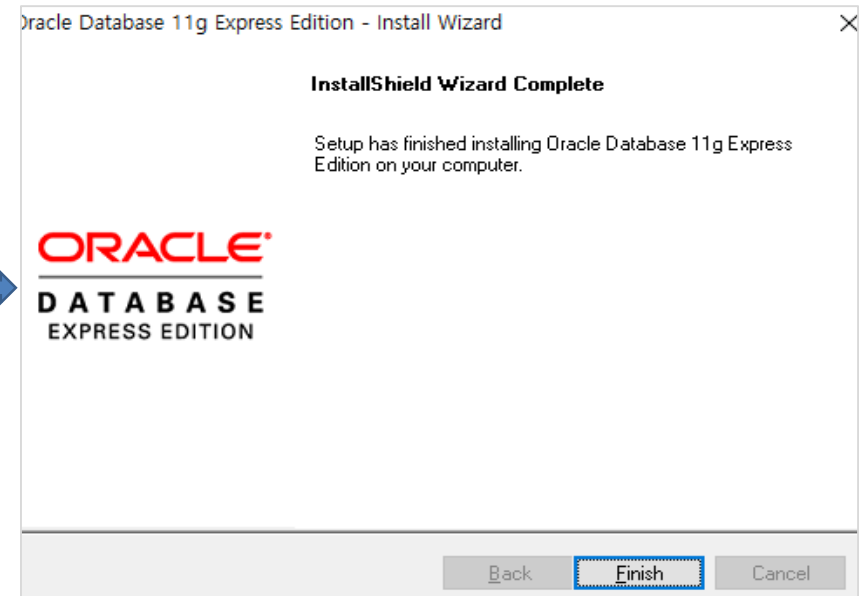
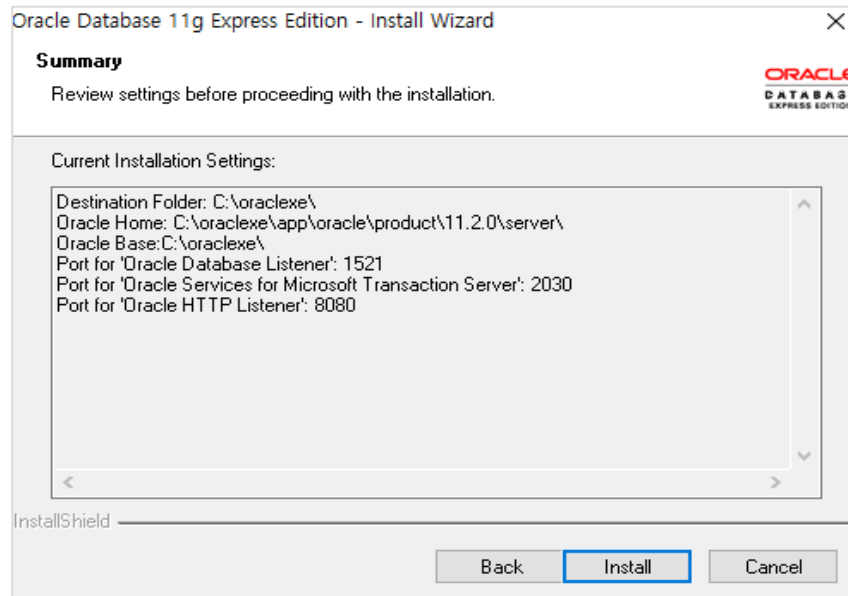


system의 비번  
12345로 설정



# 오라클 데이터 베이스




## ❖ 오라클 데이터베이스와 설치



# 오라클 데이터 베이스

## ❖ 오라클 데이터베이스와 설치 후 확인

내컴퓨터 – 우측마우스 – 관리 – 서비스 및 응용프로그램 - 서비스

 OracleServiceXE	실행 ...	자동
 OracleXEClrAgent		수동
 OracleXETNSListener	실행 ...	자동



# 오라클 데이터 베이스

## ❖ Oracle DB 시스템에 접속

- ① 명령프롬프트(cmd) 열기
- ② **sqlplus**(DBMS 소프트웨어) 입력
- ③ 사용자명(user) : **system**
- ④ 비밀번호 : **12345**

```
명령 프롬프트 - sqlplus
Microsoft Windows [Version 10.0.18363.1379]
(c) 2019 Microsoft Corporation. All rights reserved.

C:\Users\김기용>sqlplus

SQL*Plus: Release 11.2.0.2.0 Production on 수 2월 24 06:01:25 2021

Copyright (c) 1982, 2014, Oracle. All rights reserved.

Enter user-name: system
Enter password:

Connected to:
Oracle Database 11g Express Edition Release 11.2.0.2.0 - 64bit Production

SQL> SHOW USER;
USER is "SYSTEM"
SQL>
```







# 오라클 SQL 디벨로퍼

## ◆ 개발도구 - 오라클 SQL 디벨로퍼 : (sqldeveloper 다운로드 검색)

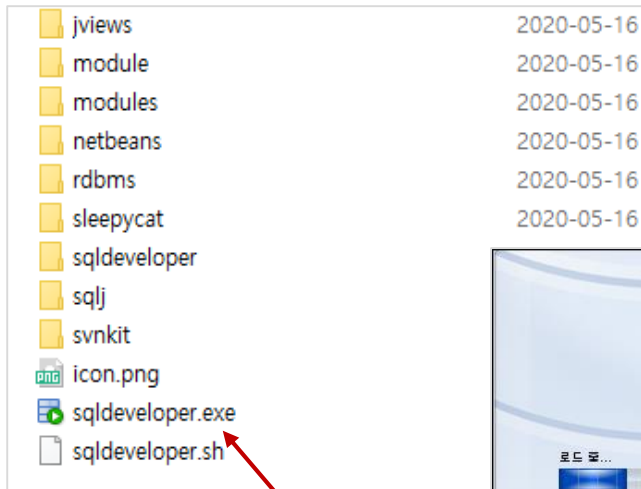
Oracle SQL developer는 오라클 데이터베이스에서 SQL 작업을 수행하는 통합개발환경(IDE)이다. SQL과 PL/SQL 코드 작성을 위해 다양한 기능을 제공하며 프리웨어이다.

Platform	Download	Notes
Windows 64-bit with JDK 8 included	 <a href="#">Download (490 MB)</a>	<ul style="list-style-type: none"><li>• MD5: 8ddbc6663eb774e179b33f702ecff101</li><li>• SHA1: b1b08c57eb0ba95713a0e42f9ab58d9a6446442f</li><li>• <a href="#">Installation Notes</a></li></ul>
Windows 32-bit/64-bit	 <a href="#">Download (410 MB)</a>	<ul style="list-style-type: none"><li>• MD5: ec986f454d747b742830284e6cd46fb0</li><li>• SHA1: f250ec93895f7b3fb4ae240ef32705cc5392e1b1</li><li>• <a href="#">Installation Notes</a></li><li>• <a href="#">JDK 8 or 11 required</a></li></ul>

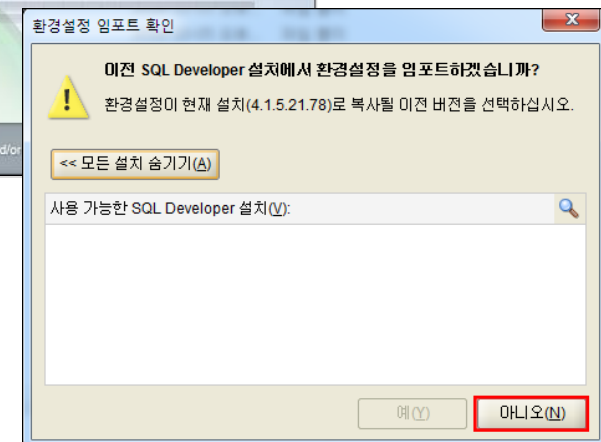


# 오라클 SQL 디벨로퍼

## ◆ Sqldeveloper 설치



실행



# 오라클 SQL 디벨로퍼

## ◆ 데이터베이스 만들기 및 접속

### 1. 관리자 (system) 계정만들기

사용자이름(USER) – system, 비밀번호 - oracle

①

Oracle SQL Developer

파일(F) 편집(E) 보기(V) 이동(N) 실행(R) 팀(M)

접속

새 데이터베이스 접속...  
새 Oracle NoSQL 접속...  
새 클라우드 접속...

인덱스  
패키지  
프로시저  
연산자  
함수  
대기열  
대기열 테이블  
트리거

새로 만들기/데이터베이스 접속 선택

접속 이름	접속 세부정보
HRdb	HR@//localho...
SYS	system@//loc...

Name: SYS

데이터베이스 유형: Oracle

사용자 정보: 프록시 사용자

인증 유형: 기본값

사용자 이름(U): system

비밀번호(P): .....

접속 유형(Y): 기본

세부정보: 고급

호스트 이름(A): localhost

포트(B): 1521

☒ SID(I): XE

☐ 서비스 이름(E):

테스트후 성공하면 접속

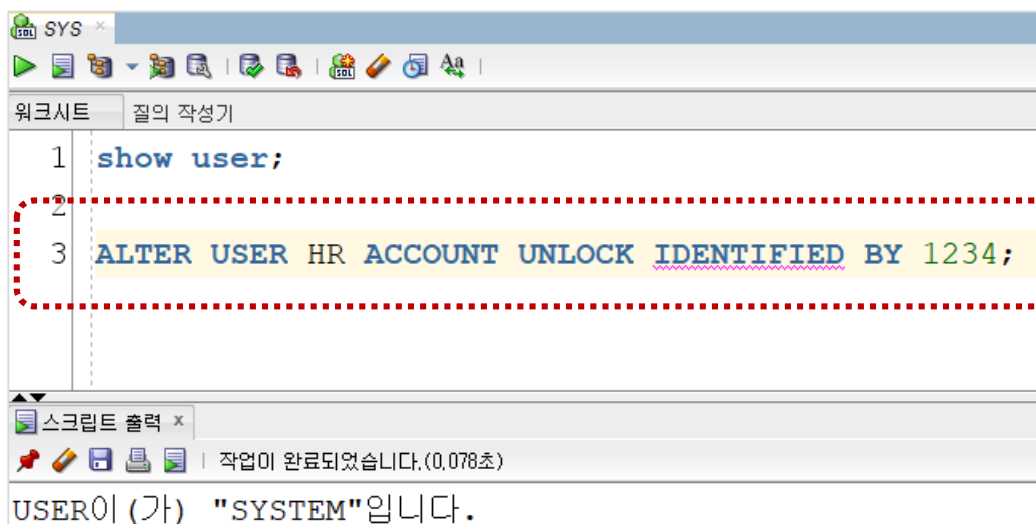
상태: 도움말(H) 저장(S) 지우기(C) 테스트(T) 접속(O) 취소



# 오라클 SQL 디벨로퍼

## ◆ 샘플 스키마(데이터베이스) 사용하기

### 2. HR 스키마 사용을 위해 계정 ROCK 풀기



```
1 show user;
2
3 ALTER USER HR ACCOUNT UNLOCK IDENTIFIED BY 1234;
```

스크립트 출력 x

작업이 완료되었습니다.(0.078초)

USER01 (가) "SYSTEM"입니다.

▣ 스키마란 사용자와 데이터베이스를 구성하는 객체들, 데이터를 포괄하는 개념



# 오라클 SQL 디벨로퍼

## ◆ HR 스키마

### 3. HR 스키마

새로 만들기/데이터베이스 접속 선택

접속 이름	접속 세부정보
JAVA 연동	SCOTT@//loc...
scott	scott@//local...
system	system@//loc...

Name 실습용HR Color

데이터베이스 유형 Oracle

**사용자 정보** 프록시 사용자

인증 유형 기본값

사용자 이름(U) HR 롤(L) 기본값

비밀번호(P) .... ☐ 비밀번호 저장(Y)

접속 유형(Y) 기본

**세부정보** 고급

호스트 이름(A) localhost

포트(R) 1521

☒ SID(I) XE

☐ 서비스 이름(E)

상태: 성공

도움말(H) 저장(S) 지우기(C) 테스트(T) 접속(O) 취소

포트 번호

오라클 DB

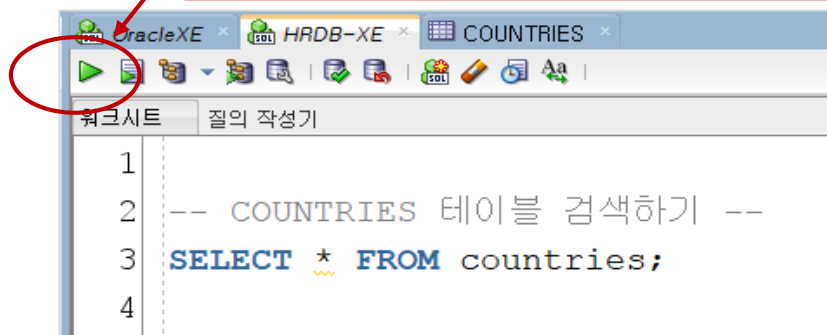


# SELECT – 자료 검색

## HR 스키마의 테이블 검색하기



실행시 실행 단추 클릭 OR (ctrl+Enter)



주석은 -- (하이픈2개)

```
-- COUNTRIES 테이블 검색하기 --  
SELECT * FROM countries;  
  
-- 특정 열 검색 --  
SELECT COUNTRY_ID, COUNTRY_NAME FROM countries;
```

A screenshot of the '결과' (Results) window in SQL Developer. It displays the output of the SQL query. The window title is '스크립트 출력 x 결과 x'. The status bar indicates '인출된 모든 행: 26(0.004초)'. The results are shown in a table with two columns: COUNTRY\_ID and COUNTRY\_NAME.

	COUNTRY_ID	COUNTRY_NAME
1	AR	Argentina
2	AU	Australia
3	BE	Belgium
4	BR	Brazil
5	CA	Canada
6	CH	Switzerland
7	CN	China
8	DE	Germany
9	DK	Denmark
10	EG	Egypt



# 오라클 SQL 디벨로퍼

## ◆ 새로운 테이블 만들기(생성)

1

Oracle SQL Developer

파일(F) 편집(E) 보기(V) 이동(N) 실행(R) 팀(M) 도구(D) 창(W) 도움말(H)

접속

Oracle 접속

JAVA 연동

테이블(필터링됨)

뷰

인덱스

프로시저

연산자

함수

대기열

대기열 테이블

트리거

유형

시퀀스

구체화된 뷰

구체화된 뷰 로그

새 갤러리

범주(C):

General

Deployment Profiles

XML

접속

데이터베이스 계층

데이터베이스 객체

데이터베이스 파일

모든 항목

항목(I):

SQL 파일

SQL(.sql) 파일의 이름을 지정할 수 있습니다. 편집을 위해 새 파일이 열립니다.

SQL 파일 생성

새 SQL 스크립트를 생성한 다음 편집 및 실행을 위해 엽니다.

파일 이름(F):

product.sql

디렉토리(D):

C:\wdbworks\java\_db

찾아보기(B)...

도움말(H)

확인

취소

디렉토리 선택

위치(L):

C:\wdbworks\java\_db

java\_db

user

sql

폴

데스크톱

아파치-톰캣-9.0.16

BankApp.jar

Boot

C

C++

C++2020

C++programming

C2020

Config.Msi

data\_analysis

DB

dbworks

java\_db

sql

demo

Documents and Settings

doit

Drivers

디렉토리 이름(D):

도움말(H)

선택

취소



# 오라클 SQL 디벨로퍼

## ◆ 테이블 만들기(생성)

```
create table 테이블이름(  
    열이름 데이터 타입  
);
```

```
-- 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)  
);
```

테이블 생성 SQL문 코드

자료형	설명
varchar2	가변길이 문자열 데이터
number	숫자를 저장
DATE	날짜 형식 저장





# 오라클 SQL 디벨로퍼

## ◆ 자료 검색 / 삽입 / 삭제

- 자료 검색 : select 열이름 from 테이블이름
- 자료 삽입 : insert into 테이블이름 values (데이터 값1, 데이터 값2.... )
- 자료 삭제 : delete from 테이블이름 where 조건식
- 자료 수정 : update 테이블이름 set 열이름=데이터값 where 조건식

```
-- Engineer 정보 테이블 --  
CREATE TABLE Engineer(  
    companyId    number(4) primary key,  
    name         varchar2(20)  
);  
  
-- 테이블 구조 --  
DESC ENGINEER;  
  
-- 사원 추가 --  
insert into Engineer values(1001, '추신수');  
  
-- 사원 검색 --  
select * from engineer;  
  
insert into Engineer values(1002, '박인비');  
insert into Engineer values(1003, '손흥민');
```

```
-- 사원 삭제 --  
delete from Engineer where companyId = 1003;  
insert into Engineer values(1004, '김연아');  
  
-- 사번 정렬 --  
select * from engineer order by companyId;  
  
commit;
```



# 오라클 SQL 디벨로퍼

## ◆ Commit과 Rollback 명령

- Commit(커밋)

자료의 삽입, 삭제, 변경이 일어난 경우 commit 명령으로 완료함

- RollBack(롤백)

롤백 명령으로 복구할 수 있다.

단, commit 명령을 사용한 후에는 복구되지 않음

```
--내림 차순 정렬--  
SELECT * FROM Member ORDER BY id DESC;  
  
-- 이름이 '추신수'인 회원 삭제 --  
DELETE FROM member WHERE name = '추신수';  
ROLLBACK;  
  
-- ROLLBACK 이후 삭제 전으로 복구됨 --  
DELETE FROM member WHERE name = '추신수';  
COMMIT;  
  
-- COMMIT : 커밋을 하면 ROLLBACK 명령이 적용되지 않음--  
ROLLBACK;
```



# 오라클 SQL 디벨로퍼

## ◆ 칼럼(열) 추가, 삭제, 이름 변경

- 칼럼 추가

ALTER TABLE 테이블명 ADD(컬럼명 데이터타입(사이즈))

- 칼럼 삭제

ALTER TABLE 테이블명 DROP(컬럼명 데이터타입(사이즈))

- 칼럼 수정

ALTER TABLE 테이블명 MODIFY(컬럼명 데이터타입(사이즈))

- 칼럼 이름 변경

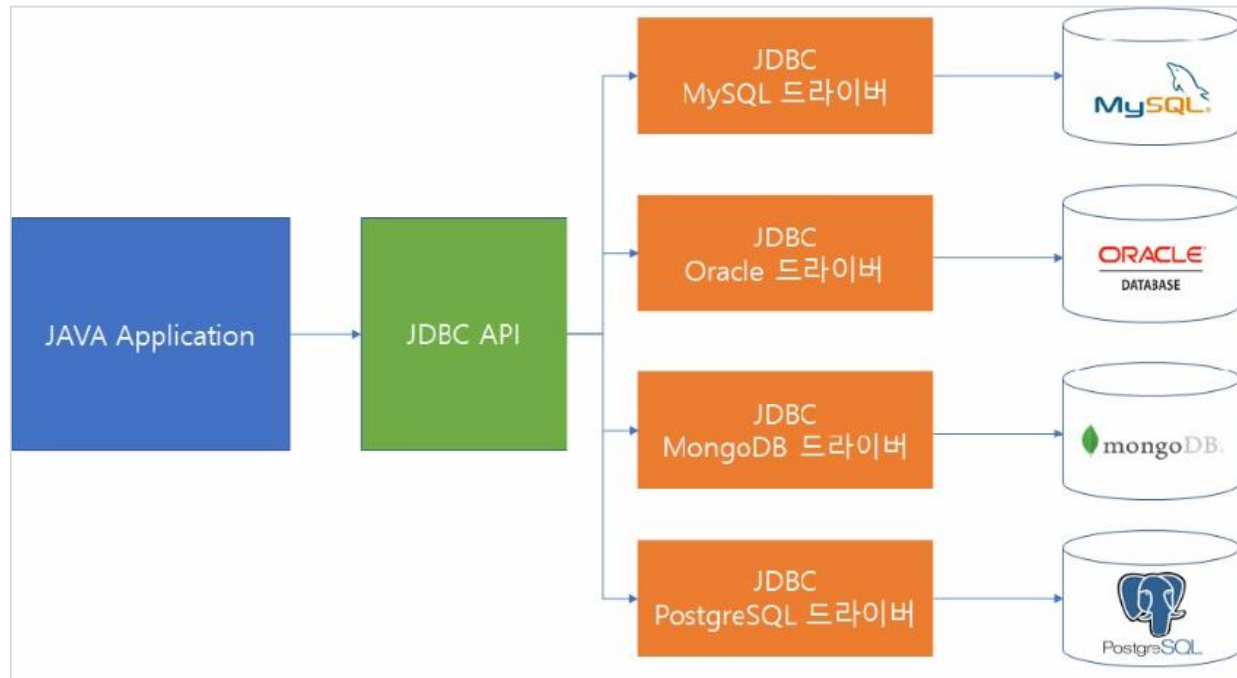
ALTER TABLE 테이블명 RENAME COLUMN 원래컬럼명 TO 바꿀 컬럼명;



# 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

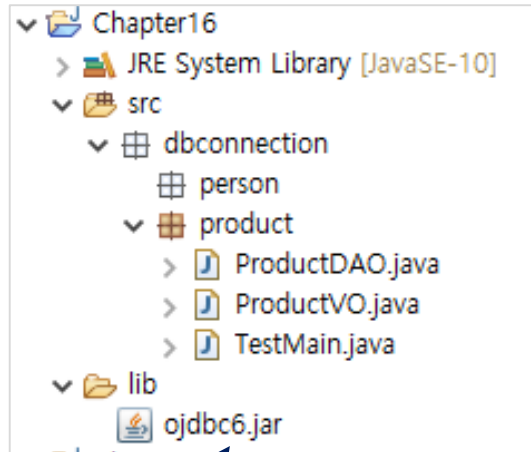
로컬 디스크 (C:) > app > suguin > product > 11.2.0 > dbhome_1 > jdbc > lib		
이름	수정한 날짜	유형
 ojdbc5.jar	2010-03-04 오전 4:37	ALZip JAR File
 ojdbc5_g.jar	2010-03-04 오전 4:37	ALZip JAR File
 ojdbc5dms.jar	2010-03-04 오전 4:37	ALZip JAR File
 ojdbc5dms_g.jar	2010-03-04 오전 4:37	ALZip JAR File
 ojdbc6.jar	2010-03-04 오전 4:37	ALZip JAR File
 ojdbc6_g.jar	2010-03-04 오전 4:37	ALZip JAR File

로컬 디스크 (C:) > sqldeveloper > jdbc > lib	
이름	
 ojdbc8.jar	

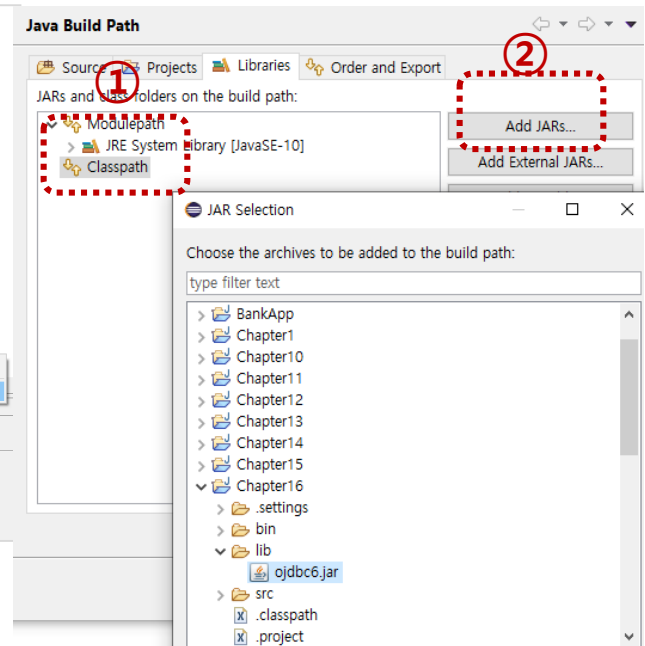
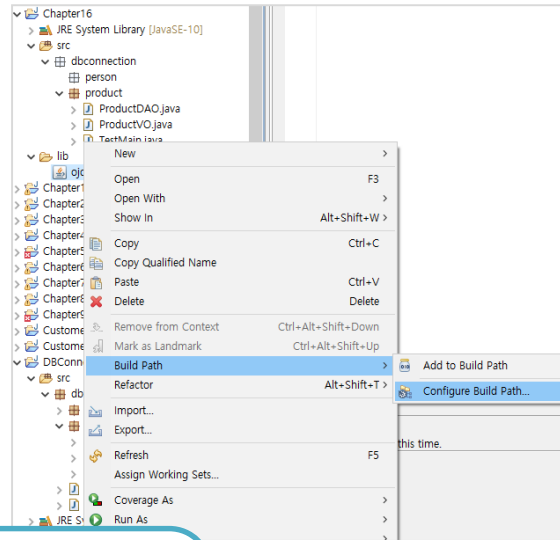


# JDBC(Java Database Connectivity)

## ◆ JDBC를 이용한 데이터베이스 연동하기 ojdbc 이클립스 프로젝트에 복사하기



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

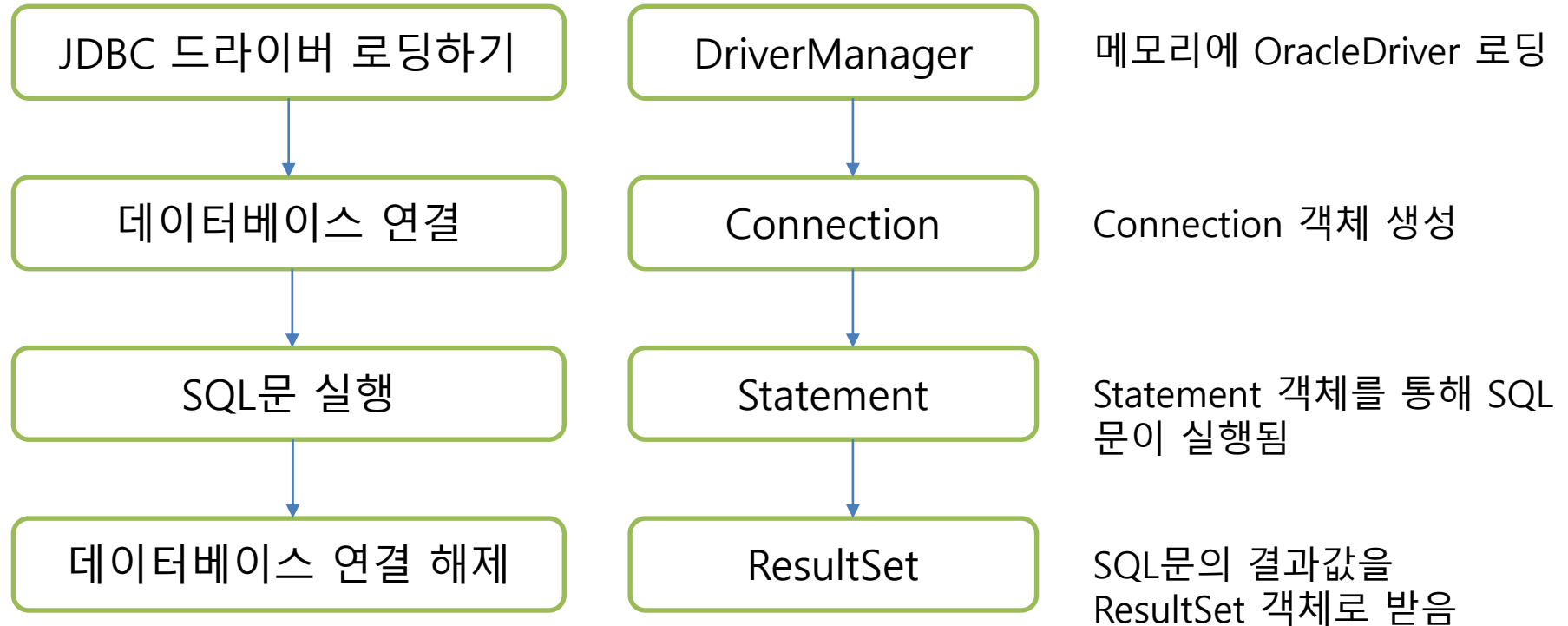


프로젝트 > 우측마우스 > Build Path > Cofigure Build Path > Libraries(Classpath) > Add JARs



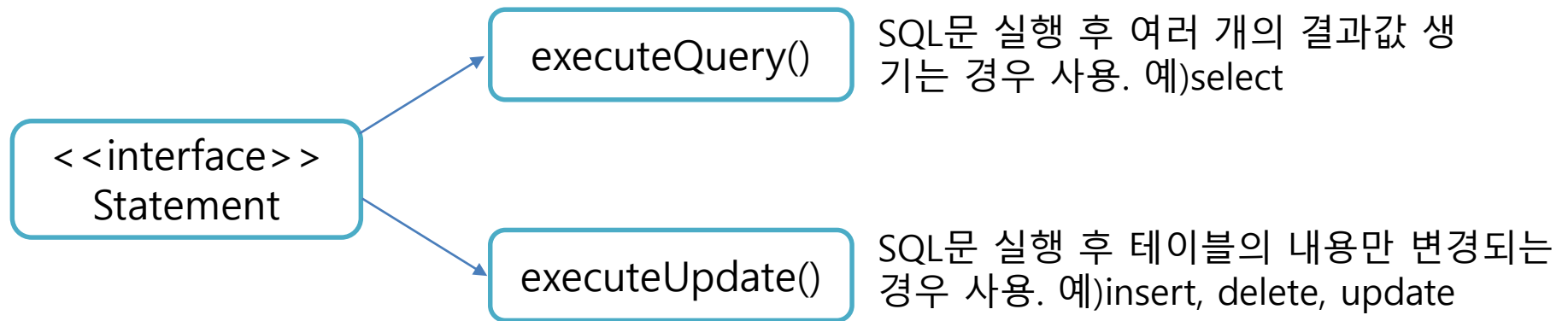
# JDBC(Java Database Connectivity)

## ➤ 데이터베이스 연결 순서

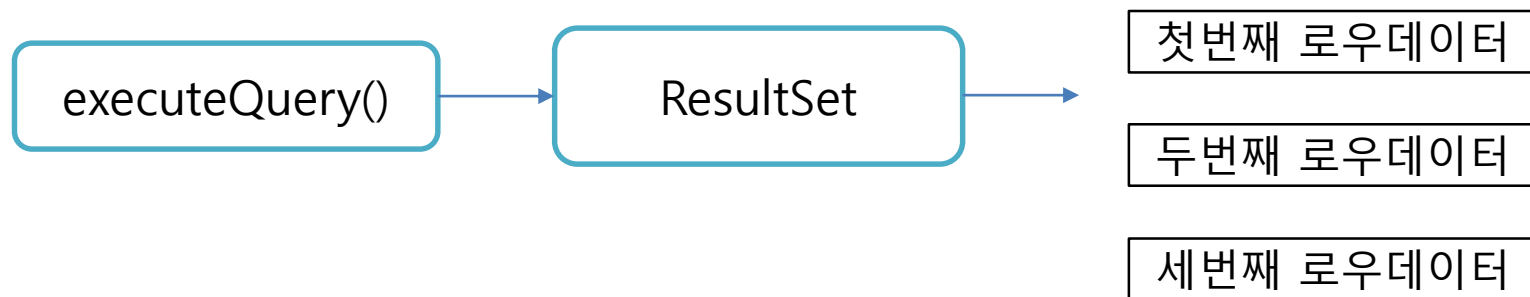


# JDBC(Java Database Connectivity)

## ➤ Statement 객체 살펴보기



### executeQuery() 실행 후 반환 되는 레코드셋



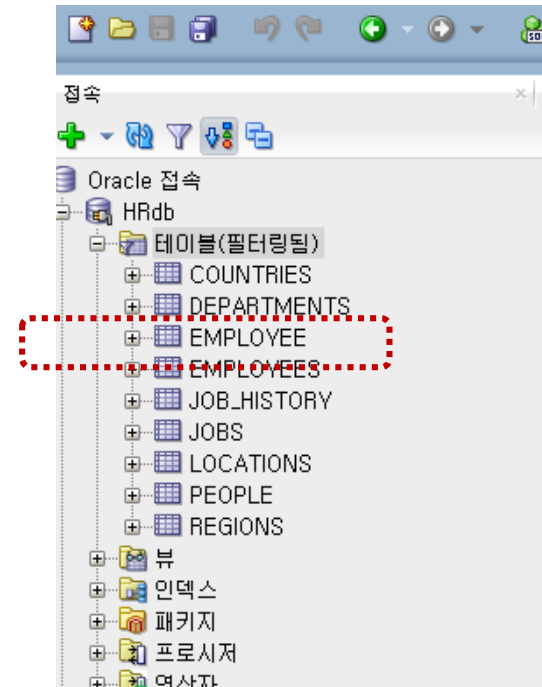


# 연결 테스트(Connection Test)

## ◆ SYSTEM에서 person 테이블 생성

```
-- employee 테이블 생성
CREATE TABLE employee (
  companyId NUMBER(4) PRIMARY KEY,
  name      VARCHAR2(20)
);
-- 테이블 설명 (구조)
DESC employee;
```

이름	널?	유형
-----		
COMPANYID	NOT NULL	NUMBER(4)
NAME		VARCHAR2(20)



# 연결 테스트(Connection Test)

## ◆ DB에 연결하기

```
public class JdbcTest {  
  
    private static String driverClass = "oracle.jdbc.OracleDriver";  
    private static String url = "jdbc:oracle:thin:@localhost:1522: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();  
            }  
        }  
    }  
}
```



# DAO와 VO 정의와 사용법

## 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.

## 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.



# DAO와 VO 정의와 사용법

## PersonVO.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

## PersonDAO.java

```
public class PersonDAO {  
    private String driverClass = "oracle.jdbc.OracleDriver";  
    private String url = "jdbc:oracle:thin:@localhost:1522: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

## PersonDAO.java 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

## PersonDAO.java – 자료 추가(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

## PersonDAO.java

### - 자료조회 메서드

```
// 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

## PersonDAO.java

### - 자료조회

```
//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

## PersonDAO.java

### - 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

## 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");*/
```

