

# Oracle Database

Introduction

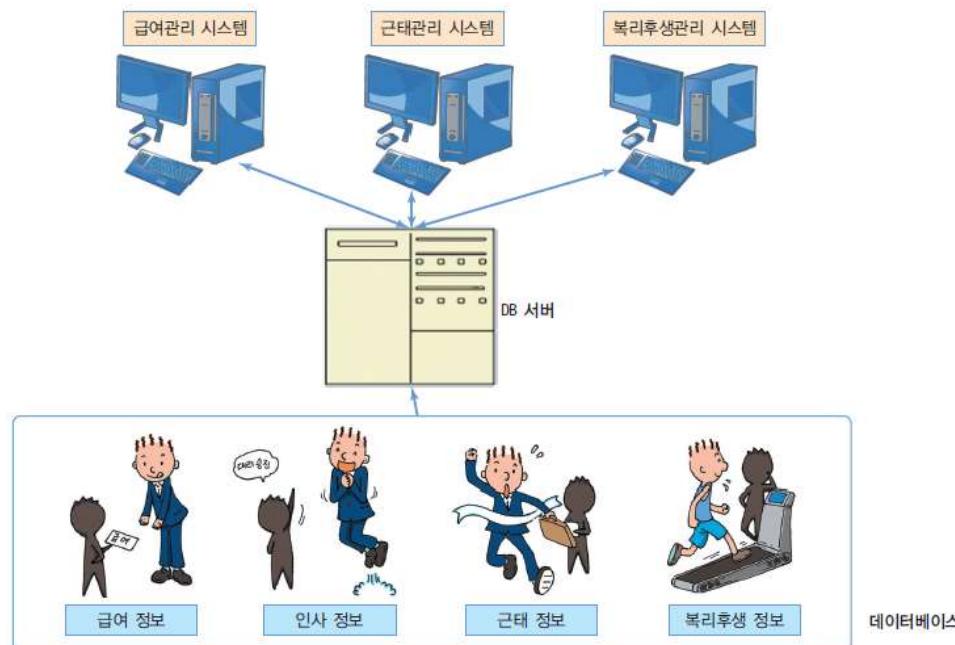
# Introduction

데이터베이스와 데이터베이스 관리 시스템

# 데이터베이스와 데이터베이스 관리 시스템

## ▶ Database

- ▶ 데이터의 집합 (a Set of Data)
- ▶ 여러 응용 시스템(프로그램)들의 통합된 정보들을 저장하여 운영할 수 있는 공용(Shared) 데이터의 집합
- ▶ 효율적으로 저장, 검색, 간접화할 수 있도록 데이터 집합들끼리 연관시키고 조직화되어야 한다



# 데이터베이스와 데이터베이스 관리 시스템

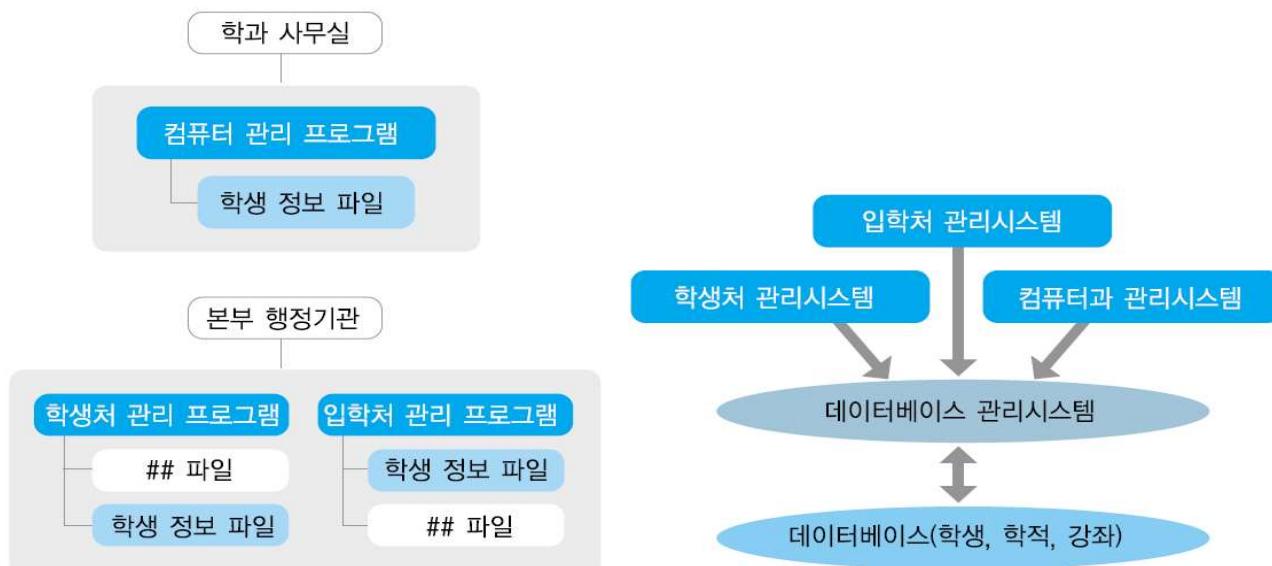
- ▶ 데이터베이스의 특성
  - ▶ 실시간 접근성 (**Real-time Accessibility**)  
사용자의 요구를 즉시 처리할 수 있다
  - ▶ 계속적인 변화 (**Continuous Evolution**)  
정확한 값을 유지하기 위해 삽입, 삭제, 수정 작업 등을 이용하여 데이터를 지속적으로 갱신할 수 있다
  - ▶ 동시 공유성 (**Concurrent Sharing**)  
사용자마다 서로 다른 목적으로 사용하므로 동시에 여러 사람이 동일한 데이터에 접근하고 이용할 수 있다
  - ▶ 내용 참조 (**Contents Reference**)  
저장한 데이터 레코드의 위치나 주소가 아닌 사용자가 요구하는 데이터의 내용, 즉 데이터 값에 따라 참조 할 수 있어야 한다

# 데이터베이스와 데이터베이스 관리 시스템

- ▶ 데이터베이스 관리 시스템 (Database Management System = DBMS)
  - ▶ 데이터베이스를 관리하는 소프트웨어
  - ▶ 여러 응용 소프트웨어(프로그램) 또는 시스템이 동시에 데이터베이스에 접근하여 사용할 수 있게 한다
  - ▶ 필수 3기능
    - ▶ 정의 기능 : 데이터베이스의 논리적, 물리적 구조를 정의
    - ▶ 조작 기능 : 데이터를 검색, 삽입, 갱신, 삭제하는 기능
    - ▶ 제어 기능 : 데이터베이스의 내용 정확성과 안전성을 유지하도록 제어하는 기능
  - ▶ Oracle, Microsoft SQL Server, MySQL, DB2 등의 상용 또는 공개 DBMS가 있다

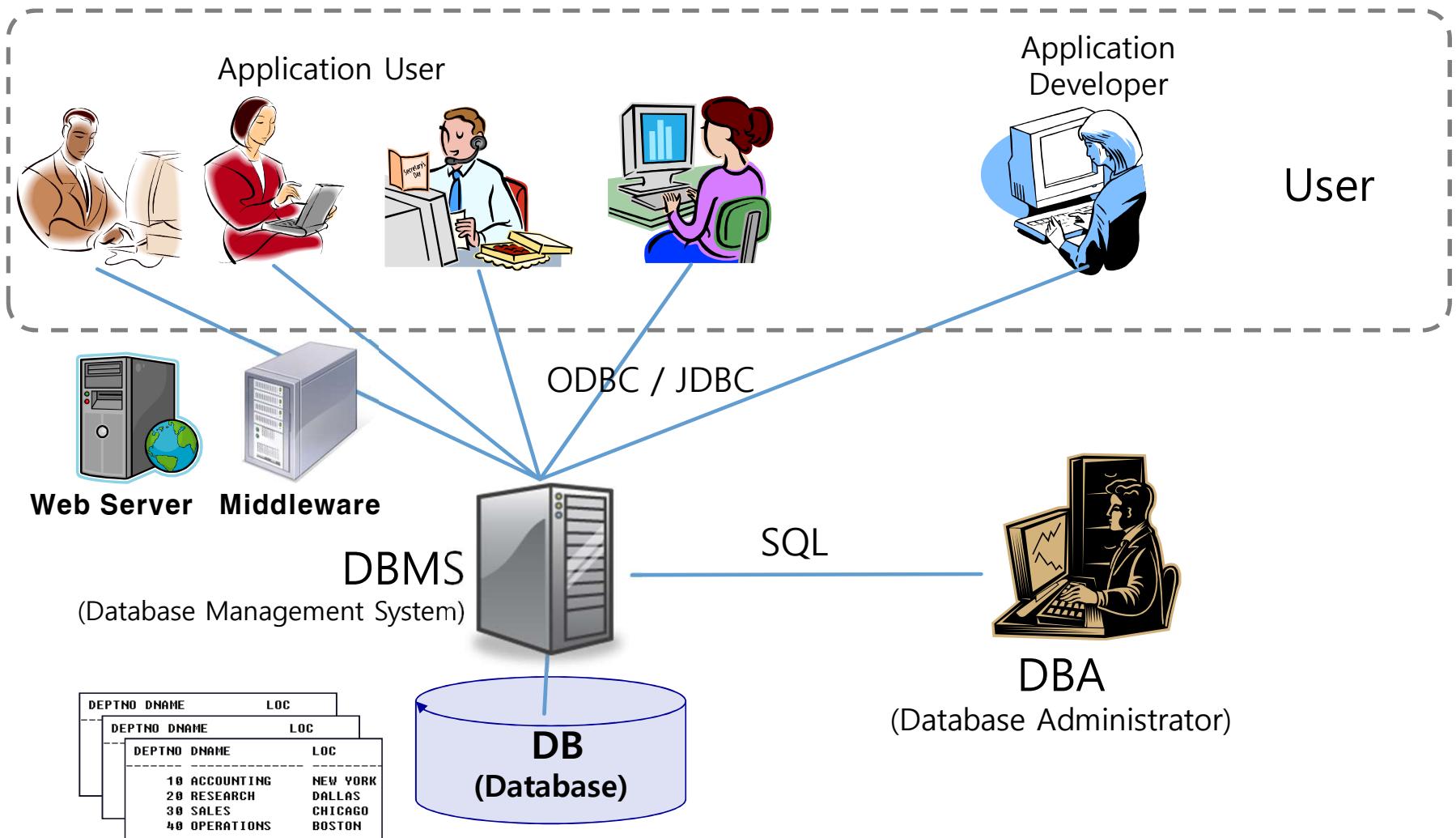
# 데이터베이스와 데이터베이스 관리 시스템

- ▶ 기존 파일 시스템의 문제
  - ▶ 데이터 종속성으로 인한 문제
  - ▶ 데이터 중복성으로 인한 문제
- ▶ 데이터베이스의 도입
  - ▶ 데이터의 종속성 보완
  - ▶ 중복성 제거



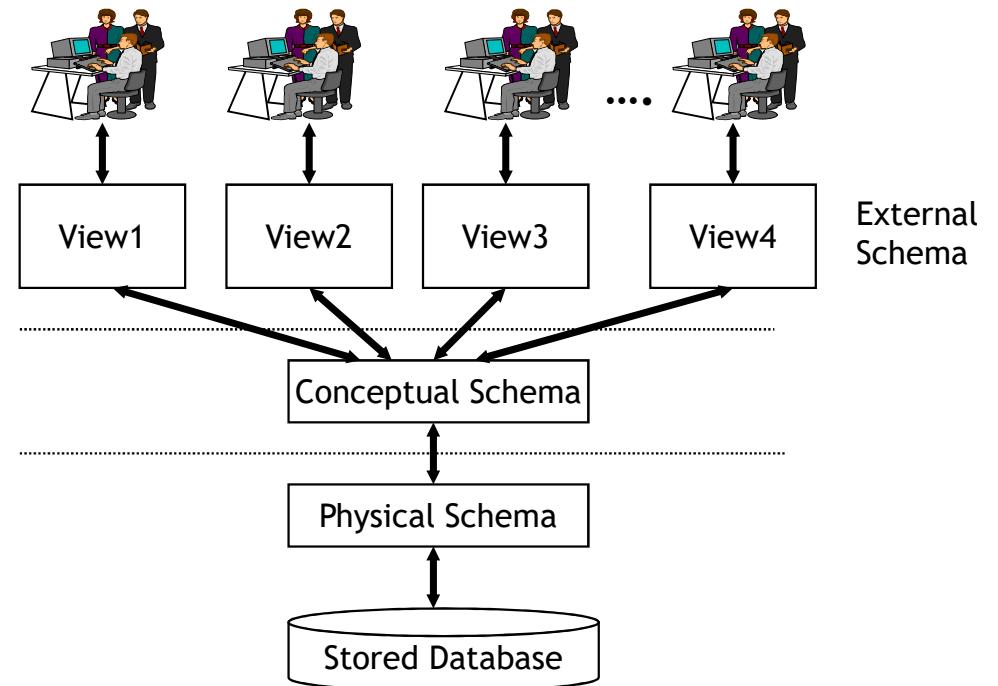
[그림 7-3] 파일관리시스템(좌)과 데이터베이스 관리시스템(우)

# 데이터베이스와 데이터베이스 관리 시스템



# Database Schama

- ▶ Schema
  - ▶ 데이터베이스의 논리적 정의
- ▶ 3단계 Schema
  - ▶ External Schema (외부 스키마)
    - ▶ 각 사용자 입장에서 본 Database 구조
    - ▶ 사용자마다 서로 다른 Schema를 가짐
    - ▶ 개념 Schema에 대한 서브 Schema
  - ▶ Conceptual Schema (개념적 스키마)
    - ▶ 조직 전체의 입장에서 본 Database 구조
    - ▶ 한 개의 Schema만 존재하며, 서로 다른 사용자가 공유
    - ▶ Data 객체(개체, 관계), 제약 조건에 대한 명세를 유지
  - ▶ Physical Schema (물리적 스키마)
    - ▶ 저장 장치의 입장에서 본 Database 구조
    - ▶ 각 data 객체의 저장 구조를 표현
    - ▶ 내부 레코드의 형식
    - ▶ 저장 data 항목의 표현 방법



# 데이터베이스와 데이터베이스 관리 시스템

- ▶ 데이터베이스 관리 시스템의 장점
  - ▶ 데이터 독립성 및 중복이 최소화
  - ▶ 데이터의 일관성 및 무결성 유지
  - ▶ 데이터 보안 보장
  - ▶ 표준화되고 일관된 데이터 관리 가능
  - ▶ 응용프로그램 개발 시간의 단축
  - ▶ 데이터 동시 사용 가능
  - ▶ 데이터 회복 가능
- ▶ 데이터베이스 관리 시스템의 단점
  - ▶ 시스템 자원 요구로 운영비 증대
  - ▶ 고급 프로그래밍 필요로 자료 처리의 복잡화
  - ▶ 부분적 데이터베이스 손실이 전체 시스템을 정지
  - ▶ 장애 발생 대비를 위한 복잡한 Back Up과 Recovery 작업 필요

# 데이터베이스와 데이터베이스 관리 시스템

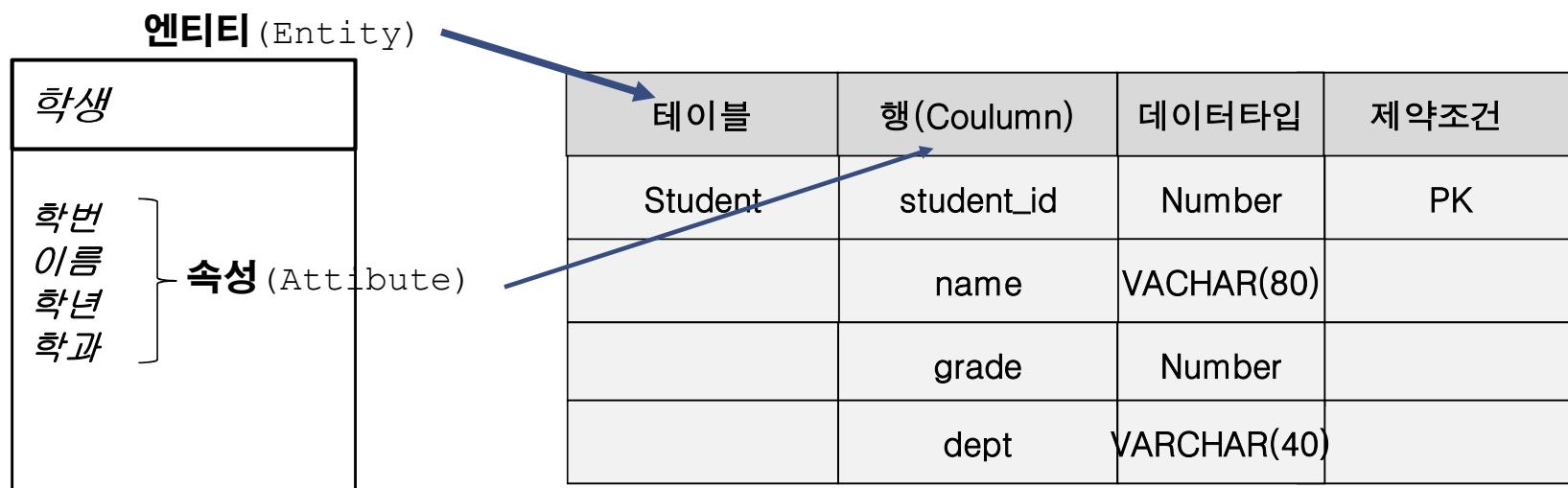
- ▶ 데이터베이스의 종류
  - ▶ 관계형 데이터베이스 (Relational Database = RDB)
    - ▶ 1970년 IBM E. F. Codd에 의해 제안되어 수십년 동안 주류 데이터베이스로 성장 확대
    - ▶ 키와 값들의 간단한 관계를 테이블화 시킨 매우 간단한 원칙의 개념을 가진 데이터베이스
    - ▶ 일련의 정형화된 테이블로 구성된 데이터 항목들의 집합이며 각 테이블은 데이터의 성격에 따라 여러 개의 컬럼(키)이 포함된다
    - ▶ 사용자는 SQL이라는 표준 질의어를 통해 데이터를 조작 또는 조회할 수 있다
      - ▶ SQL (Structured Query Language)
  - ▶ 객체지향 데이터베이스 (Object Oriented Database = OODB)
    - ▶ 정보를 객체의 형태로 표현하는 데이터베이스
    - ▶ 객체 모델이 그대로 데이터베이스에도 적용되어 데이터 모델을 그대로 응용프로그램에 적용, 데이터 변환과 질의 작업이 필요하지 않은 장점이 있다

# 데이터베이스와 데이터베이스 관리 시스템

- ▶ 데이터베이스의 종류
  - ▶ 객체관계형 데이터베이스 (Object Relation Database = ORDB)
    - ▶ 관계형 데이터베이스에서 사용하는 데이터를 확장
    - ▶ 관계형 데이터베이스를 객체 지향 모델링과 데이터를 관리하는 기능을 갖도록 확장한 것
  - ▶ NoSQL
    - ▶ 대용량 데이터, 비정형 데이터의 웹 서비스와 SNS, 클라우드 컴퓨팅의 확대 보급과 대중화로 최근 주목받고 있는 데이터베이스 기술
  - ▶ 그 외
    - ▶ Hierarchical Database
    - ▶ Network Database

# 관계형 데이터베이스

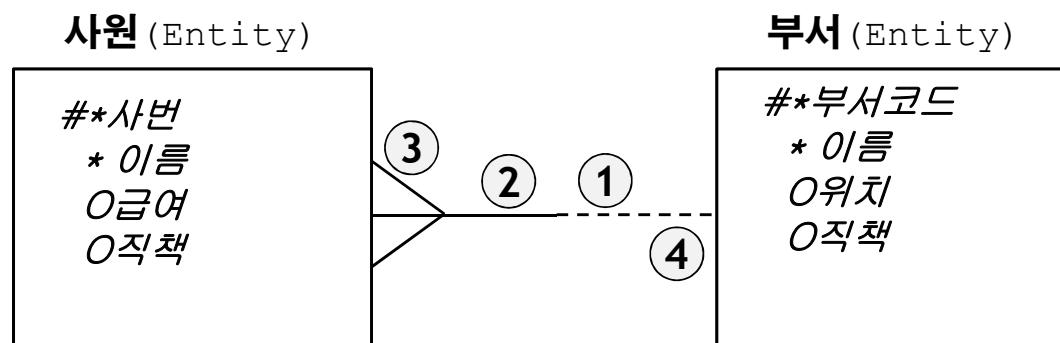
- ▶ 논리적(개념적) 데이터 모델링과 물리적인 데이터베이스



- ▶ RDBMS의 이해
  - ▶ Relation 또는 2차원 Table을 이용하여 정보 저장
    - ▶ Relation = Table 간의 연관
    - ▶ Attribute = Column
    - ▶ Tuple = Row

# 관계형 데이터베이스

## ▶ ERD 예시를 통해 보는 관계



1. 어떤 부서는 사원을 배치 받지 않을 수 있다 (점선)
2. 사원은 특정 부서에 소속되어 있다 (실선)
3. 한 부서에는 여러 명의 사원이 소속되어 있다 (다중선)
4. 한 사원은 하나의 부서에만 속한다 (단일선)

# : 대표값 - PK(Primary Key)  
• : NOT NULL  
○ : NUL 가능

# 관계형 데이터베이스

▶ 테이블

**Table Student**

**column(Attribute)**

student_id	name	grade	dept
1	정성진	1	컴퓨터
2	박현진	2	수학
3	홍길동	4	물리
...	...	...	...

field                              field

**row( record )**

테이블 : RDBMS의 기본적 저장 구조. 한 개 이상의 **column**과 0개 이상의 **row**로 구성

열(Column) : 테이블상에서의 단일 종류의 데이터를 나타냄. 특정 데이터 타입 및 크기를 가지고 있음

행(Row) : Column 값들의 조합. 튜플, 레코드라고 함

기본키(PK)에 의해 구분된다. 기본키는 중복을 허용하지 않으며 없어서는 안된다

Field : Row와 Column의 교차점으로 Field는 데이터를 포함할 수 있고 없을 때는 **NULL** 값을 가지고 있다

# 관계형 데이터베이스

## : 무결성 제약조건(Integrity Constraint)

- ▶ 개체 무결성(Entity Integrity)
  - ▶ Table은 중복된 ROW를 가질 수 없으며 모든 Table은 각각의 ROW를 유일하게 식별할 수 있는 Column의 집합을 가진다. 이러한 Column의 집합 중에서 대표되는 컬럼을 Primary Key(PK)로 정의한다
  - ▶ Primary Key의 값은 항상 유일(Unique)하며 널(NULL)을 허용해서는 안된다
- ▶ 참조 무결성(Referential Integrity)
  - ▶ Table들은 Foreign Key(FK)를 통해 서로 연결되어 있다. Foreign Key는 다른 Table 또는 자신 Table의 PK 값을 참조하기 위해 복사하여 가지고 있는 Column을 말한다
  - ▶ 참조 무결성이 지켜지기 위해서 FK Column의 값은 참조하는 PK 컬럼 값 중의 하나이거나 Null이어야 한다
- ▶ 무결성 제약은 DBMS 시스템이 자동으로 수행한다

# 관계형 데이터베이스

- ▶ Primary Key (PK) : 기본키
  - ▶ 관계(Relation)에서 튜플을 구분하기 위하여 사용하는 기본 키
  - ▶ 하나의 애트리뷰트, 또는 애트리뷰트의 집합(복합키) 가능
  - ▶ 관리자에 의해 릴레이션 생성시 정의된 (자동으로 Index 생성)
  - ▶ 동일한 PK를 지닌 레코드는 존재할 수 없음
- ▶ 기타
  - ▶ Candidate Key (후보키) : 튜플을 식별할 수 있는 최소한의 애트리뷰트 집합
    - ▶ 하나의 릴레이션에는 PK가 될 수 있는 키가 여러 개 있을 수 있음
    - ▶ 유일성과 희소성이 있으면 Candidate Key가 될 수 있음
  - ▶ Alternative Key (대체키) : 후보키 중 기본 키가 아닌 것
  - ▶ Composite Key (복합키) : 둘 이상의 애트리뷰트가 하나의 Key를 이루는 것

# 관계형 데이터베이스

- ▶ 학생

- ▶ 기본키 : 학번
- ▶ 후보키 : 기본키와 동일
- ▶ 주민등록번호 등이 있다면 후보키가 될 수 있음

학생  
(STUDENT)

학번 (Sno)	이름 (Sname)	학년 (Year)	학과 (Dept)
100	나 수 영	4	컴퓨터
200	이 찬 수	3	전기
300	정 기 태	1	컴퓨터
400	송 병 길	4	컴퓨터

- ▶ 등록

- ▶ 기본키 : (학번, 과목번호)
- ▶ 후보키 : 기본키와 동일
- ▶ 학번이나 과목번호만으로는 키가 되지 못함
- ▶ 등록번호와 같이 별도의 단일키를 추가하여 PK로 지정할 수도 있음

등록  
(ENROL)

학번 (Sno)	과목번호 (Cno)	성적 (Grade)
100	C413	A
100	E412	A
200	C123	B
300	C312	A

# 관계형 데이터베이스

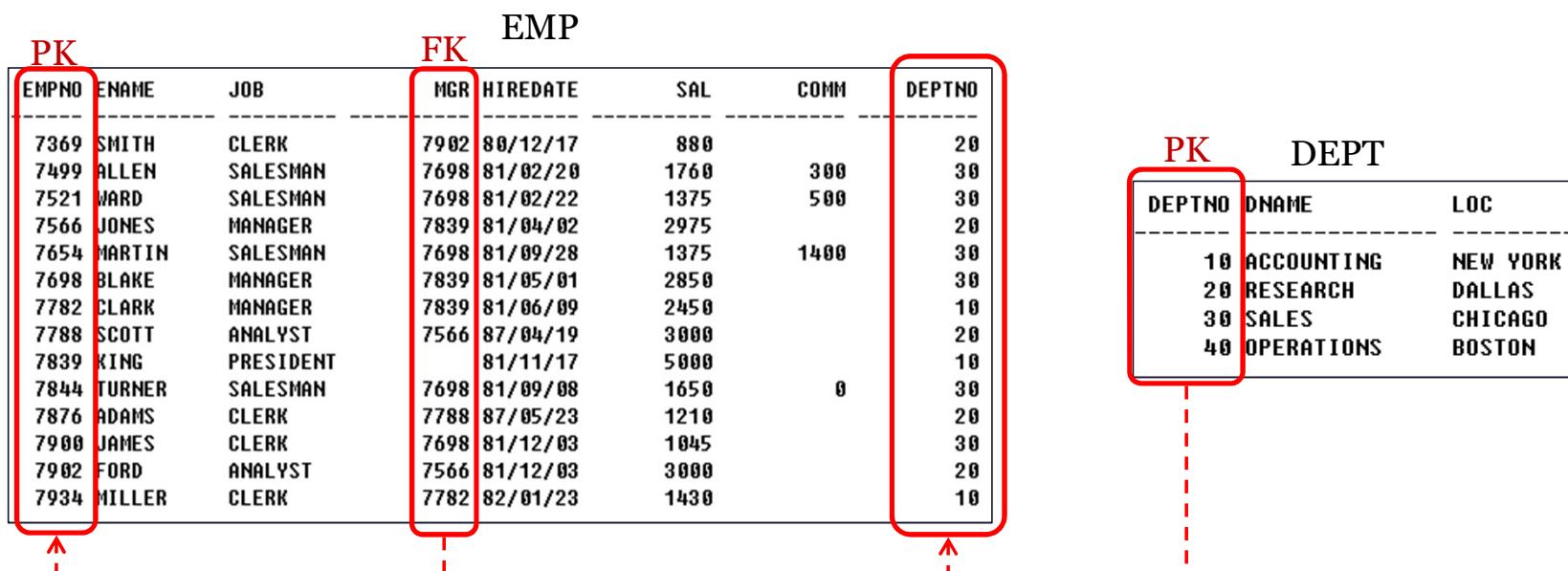
## ▶ Foreign Key (FK) : 외래키

- ▶ 기본키를 참조하는 애트리뷰트
- ▶ 다른 릴레이션의 튜플을 대표
- ▶ 릴레이션 간의 관계를 나타내기 위해 사용
- ▶ NULL 가능 (참조되지 않음을 의미)

EMP						
PK	ENAME	JOB	MGR	HIREDATE	SAL	COMM
7369	SMITH	CLERK	7902	80/12/17	880	
7499	ALLEN	SALESMAN	7698	81/02/20	1760	300
7521	WARD	SALESMAN	7698	81/02/22	1375	500
7566	JONES	MANAGER	7839	81/04/02	2975	
7654	MARTIN	SALESMAN	7698	81/09/28	1375	1400
7698	BLAKE	MANAGER	7839	81/05/01	2850	
7782	CLARK	MANAGER	7839	81/06/09	2450	
7788	SCOTT	ANALYST	7566	87/04/19	3000	
7839	KING	PRESIDENT		81/11/17	5000	
7844	TURNER	SALESMAN	7698	81/09/08	1650	0
7876	ADAMS	CLERK	7788	87/05/23	1210	
7900	JAMES	CLERK	7698	81/12/03	1045	
7902	FORD	ANALYST	7566	81/12/03	3000	
7934	MILLER	CLERK	7782	82/01/23	1430	

DEPT		
PK	DNAME	LOC
10	ACCOUNTING	NEW YORK
20	RESEARCH	DALLAS
30	SALES	CHICAGO
40	OPERATIONS	BOSTON



# SQL(Structured Query Language) 개요

- ▶ 데이터베이스 스키마 생성, 자료의 검색, 수정, 그리고 데이터베이스 객체 접근 관리 등을 위해 고안된 언어
  - ▶ 비절차식 언어
  - ▶ 1970년대 IBM의 SYSTEM R 프로젝트를 통해 개발
- ▶ 다수의 데이터베이스 관련 프로그램의 표준 언어
  - ▶ RDBMS에서 사용하기 위해 ANSI에서 책정한 표준 언어
  - ▶ DBMS 제품별로 SQL에 대한 추가 및 확장
- ▶ PL/SQL
  - ▶ 응용프로그램의 logic을 추가하여 SQL을 확장한 Oracle의 절차적 언어
  - ▶ 3세대 언어로 IF 문장이나 LOOP 문장을 통해 프로그램의 흐름을 제어할 수 있으며 SQL 문장을 이용 data 조작 가능
  - ▶ Client가 PL/SQL 블록 실행을 요청하면 Oracle 서버는 Block 전체를 실행한 후 결과를 리턴하므로 한 번의 서버 호출로 다양한 SQL과 로직을 구사할 수 있다는 장점
- ▶ SQL\*Plus
  - ▶ SQL 및 PL/SQL 문장을 인식하고 실행시켜주는 Oracle의 Tool
  - ▶ SQL, PL/SQL 등을 직접 입력하여 Oracle 서버로 보내 실행한 후 결과를 받아본다

# SQL(Structured Query Language) 개요

## : SQL 명령어의 종류

- ▶ **DML(Data Manipulation Language)**
  - ▶ 데이터 처리를 위해 응용프로그램과 데이터베이스 관리 시스템간의 인터페이스를 위한 언어
  - ▶ 데이터 처리를 위한 연산의 집합으로 데이터의 검색, 삽입, 수정 및 삭제하기 위한 수단을 제공
  - ▶ SELECT, INSERT, UPDATE, DELETE, MERGE
- ▶ **DDL(Data Definition Language)**
  - ▶ 데이터베이스 구조, 데이터 형식, 접근 방식 등 데이터베이스를 구축, 변경할 목적으로 사용하는 언어
  - ▶ DDL 컴파일러가 컴파일한 후 데이터 사전에 저장
  - ▶ 데이터의 논리적, 물리적 구조를 생성, 변경, 삭제하는 기능을 제공
  - ▶ CREATE, ALTER, DROP, RENAME
- ▶ **DCL(Data Control Language)**
  - ▶ 보안 및 권한 제어, 무결성, 회복, 병행 제어를 위한 언어
  - ▶ : 데이터에 대한 권한 관리 및 트랜잭션 제어
  - ▶ GRANT, REVOKE 등

# SQL\*Plus 기초

## : Login

- ▶ Windows 환경에서의 로그인
  - ▶ Windows 메뉴 > Run SQL Command Line
    - ▶ 프롬프트에서 `CONN 계정명/[PASSWORD]`
  - ▶ Windows 커맨드 프롬프트에서
    - ▶ `SQLPLUS 계정명/[PASSWORD]`
- ▶ 접속을 끊고 SQL\*Plus를 종료하려면 `EXIT` 명령을 실행

# SQL\*Plus 기초

- ▶ SQL\*Plus에서 SQL 문장 작성시 유의사항
  - ▶ 모든 SQL 문장은 세미콜론(;)으로 끝난다
  - ▶ SQL 문장은 한 줄로 입력하거나 여러 줄로 보기 좋게 나누어 입력한다
  - ▶ SQL 문장은 대소문자를 구분하지 않는다 (Case Insensitive)
  - ▶ data 값은 대소문자를 가린다 (Case Sensitive)
- ▶ SQL\*Plus의 결과값 출력
  - ▶ Default Column Heading
    - ▶ Column 명이 대문자로 출력된다
  - ▶ Default Data Justification
    - ▶ Number 값: right-justified
    - ▶ Character, Date 값: left-justified

# Installation

OracleXE 11g와 SQL Developer 설치

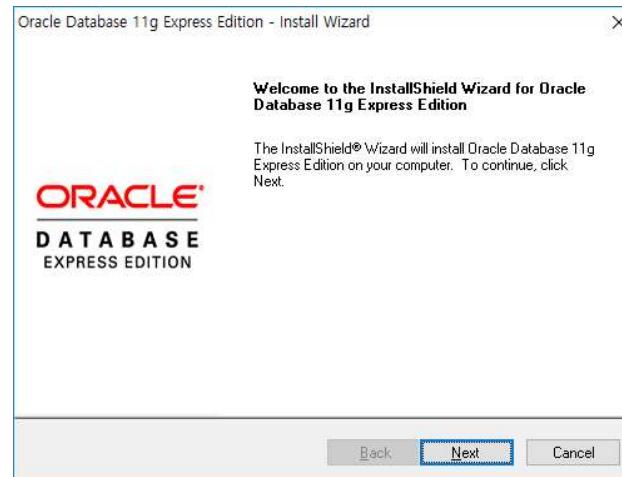
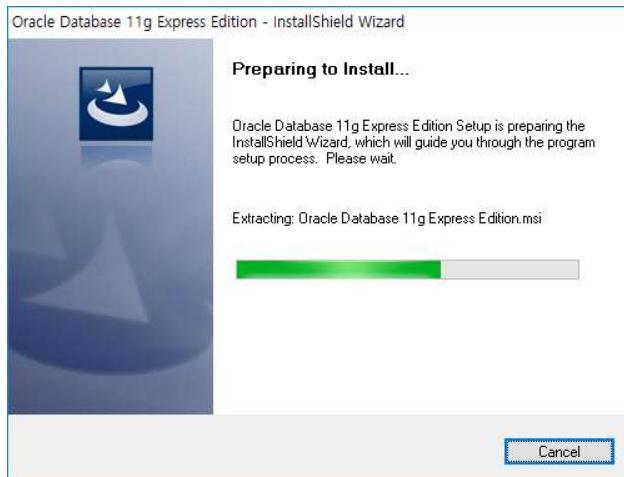
# Installation

## : OracleXE 11g

- ▶ Oracle 홈페이지에서 오라클 XE 11g 버전을 다운로드  
(Oracle Database Express Edition 11g Release 2)
- ▶ 직접 다운로드 경로  
<http://www.oracle.com/technetwork/database/database-technologies/express-edition/downloads/index.html>
- ▶ 압축을 풀고 setup 파일을 실행

# Installation : OracleXE 11g

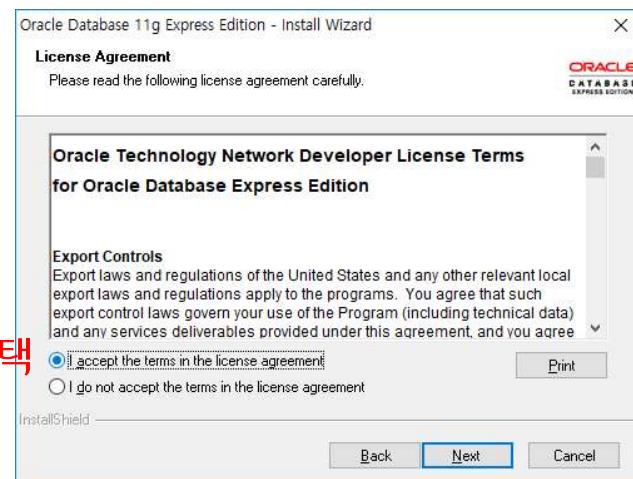
- ▶ setup.exe 설치 파일 실행



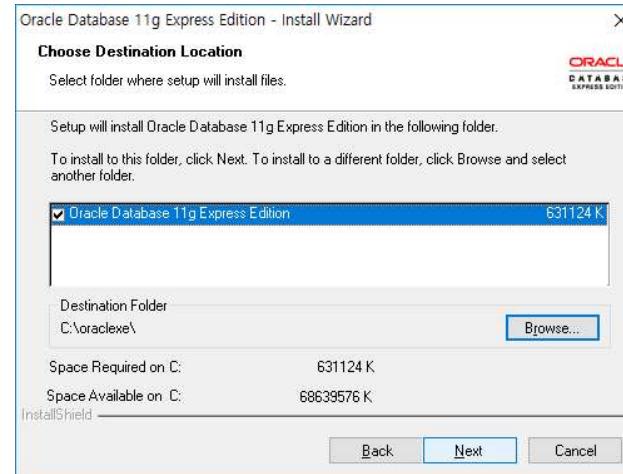
Next 클릭

# Installation : OracleXE 11g

- ▶ 라이선스 동의 및 설치 폴더 지정



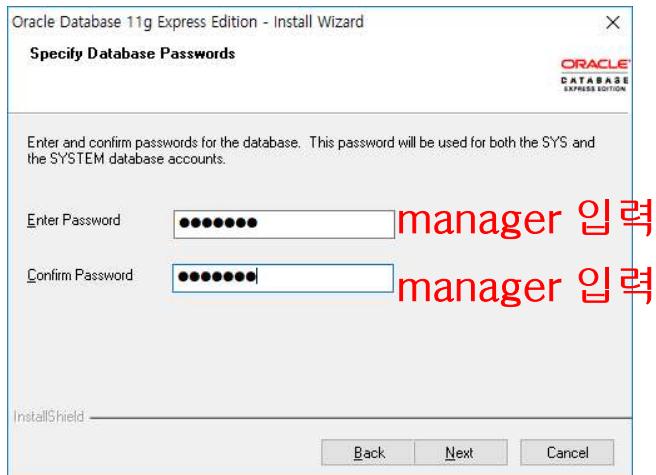
Next 클릭



Next 클릭

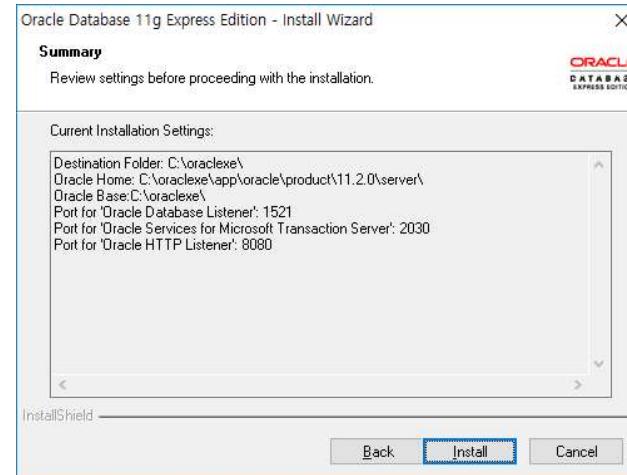
# Installation : OracleXE 11g

- ▶ sys, system 계정 비밀번호 설정



Next 클릭

수업진행을 위해 비밀번호는 반드시

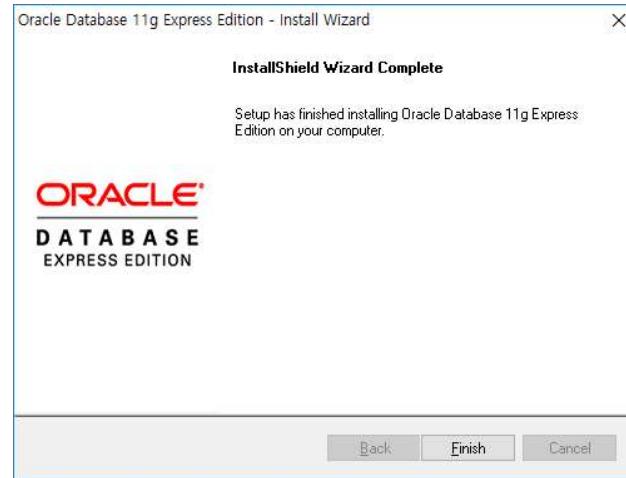
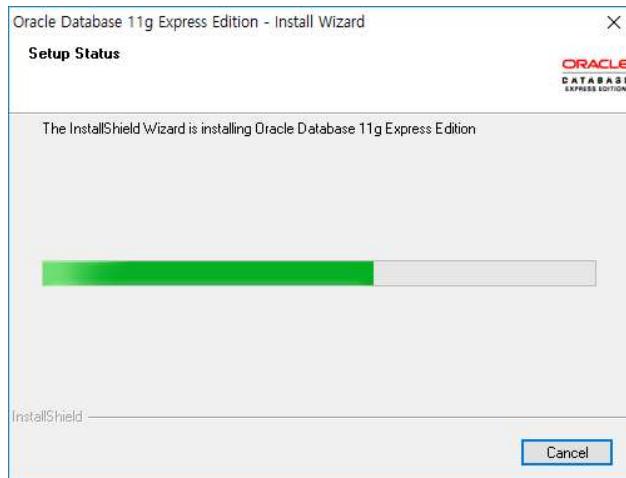


Next 클릭

manager 로 설정

# Installation : OracleXE 11g

## ▶ 설치 완료



# Install

## : Change APEX(Oracle Web Admin) PORT

- ▶ Oracle 11g XE는 기본적으로 APEX라 불리는 Web Admin을 제공
- ▶ 하지만, 기본 포트가 8080으로 되어 있어 Apache Tomcat의 기본 포트와 충돌 발생
- ▶ Apache Tomcat 을 함께 사용하기 위한 환경이라면 이 기능을 사용하지 않거나 포트를 변경해 주어야 함

```
SQL> connect system/manager
Connected.

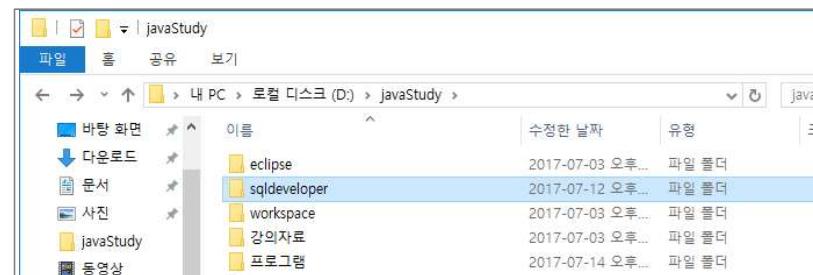
SQL> EXEC DBMS_XDB.SETHTTPPORT(8088);

PL/SQL procedure successfully completed.

SQL>
```

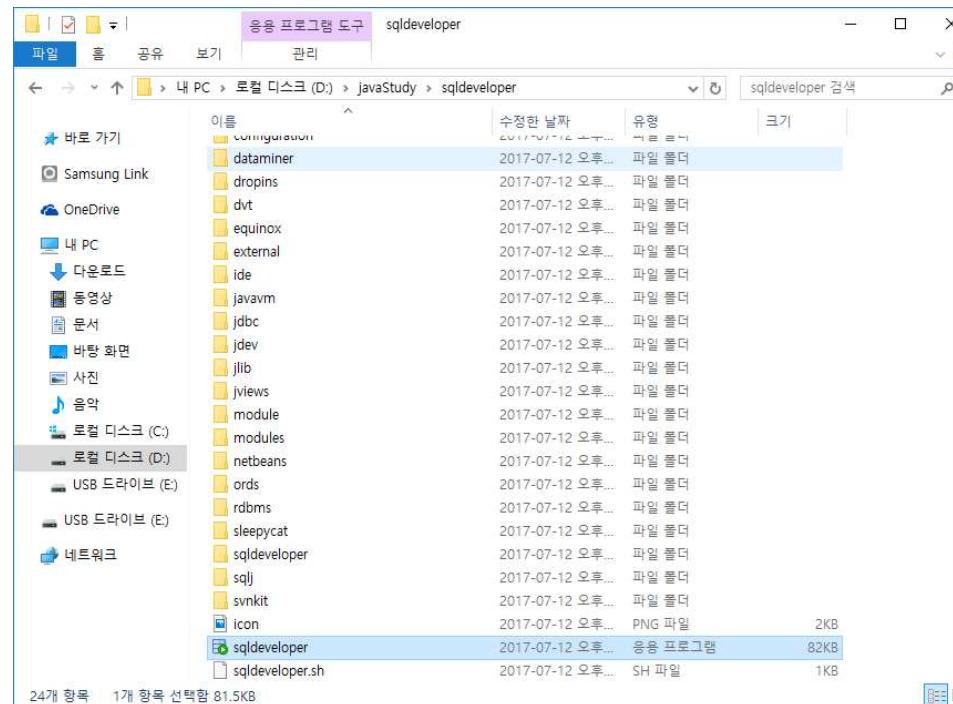
# Installation : SQL Developer

- ▶ 오라클 웹사이트에서 SQL Developer 프로그램을 다운  
“JDK가 포함되지 않은 버전”
  - ▶ 직접 다운로드 경로  
<http://www.oracle.com/technetwork/developer-tools/sql-developer/downloads/index.html>
- ▶ 사용  
설치 버전이 아니므로  
D:\javaStudy 에 풀더 복사 후 이용



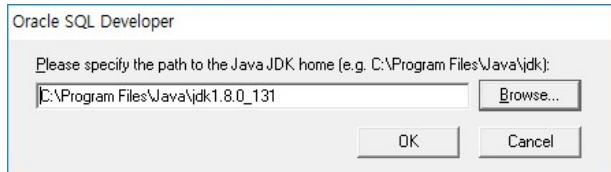
# Installation : SQL Developer

- ▶ D:\javaStudy\sqldeveloper\sqldeveloper.exe 파일 실행



# Installation : SQL Developer

## ▶ JDK 연결



자신의 jdk디렉토리 선택 후  
OK 클릭

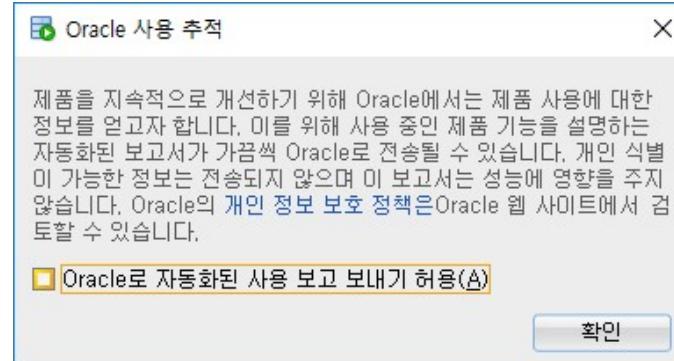


# Installation : SQL Developer

- ▶ 이전 버전 확인 및 사용 추적 옵션



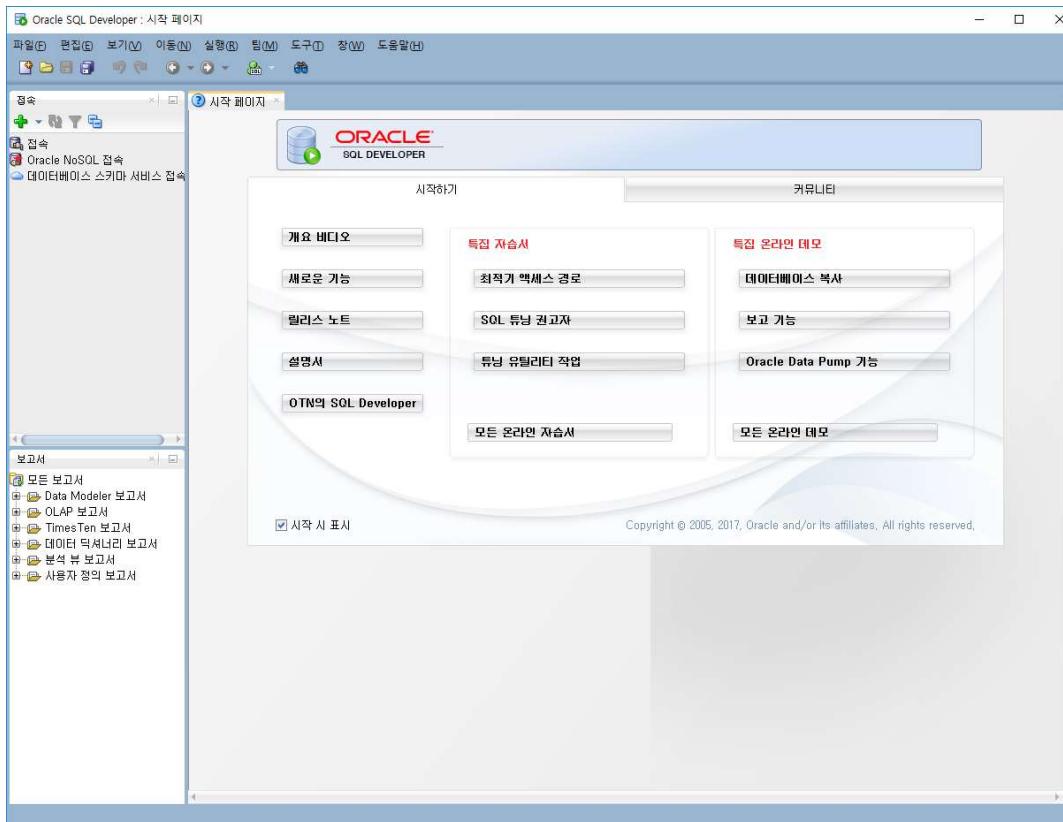
아니오 클릭



확인 클릭

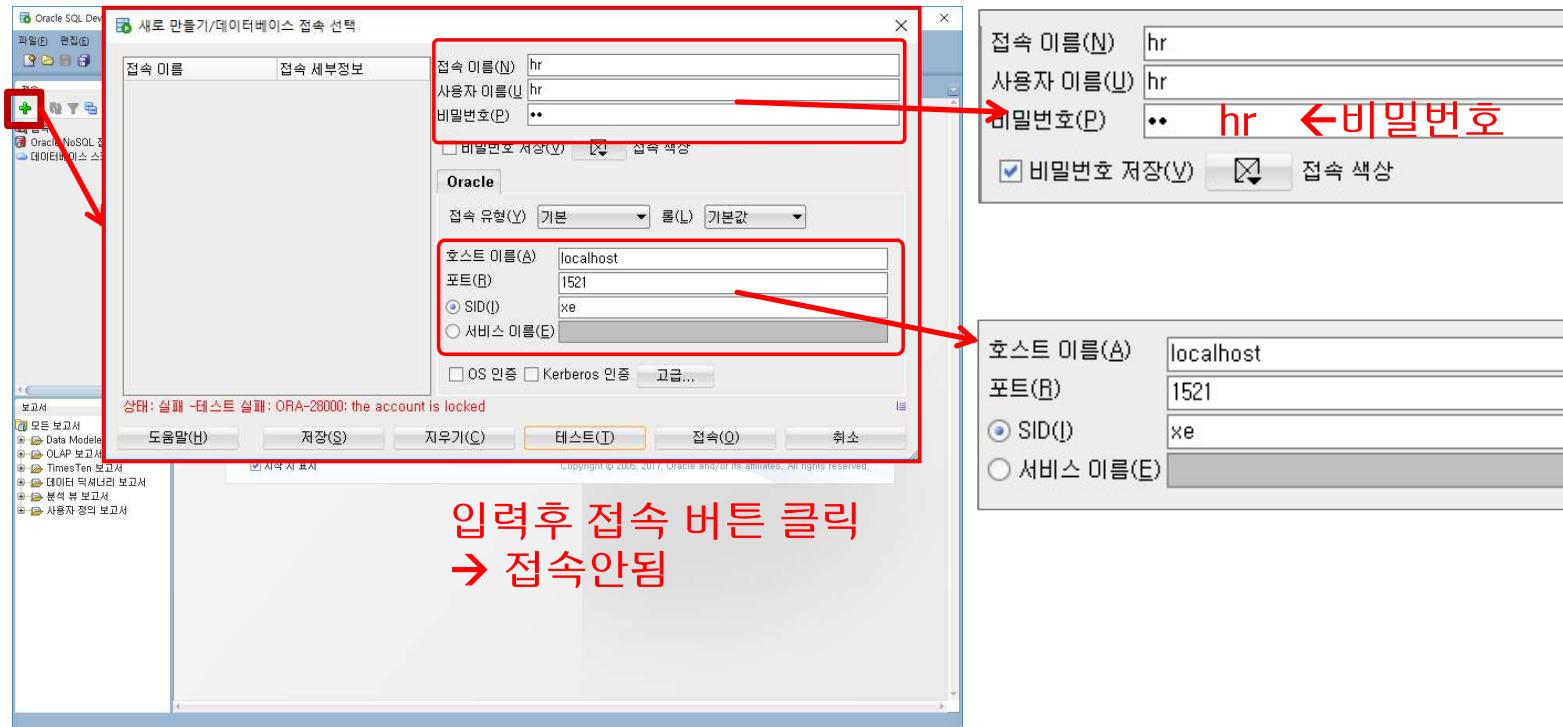
# Installation : SQL Developer

## ▶ 실행 화면 예시



# Installation : 데이터베이스 접속 시도

## ▶ 실행 화면 예시



# Installation : 데이터베이스 접속 시도

- ▶ hr 계정 풀기

```
Microsoft Windows [Version 10.0.15063]
(c) 2017 Microsoft Corporation. All rights reserved.

C:\Users\BIT>sqlplus sys/oracle as sysdba

SQL*Plus: Release 11.2.0.2.0 Production on 수 7월 12 17:45:07 2017

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

Connected to:
Oracle Database 11g Express Edition Release 11.2.0.2.0 - 64bit Production
SQL> ALTER USER hr IDENTIFIED BY hr ACCOUNT unlock ;
User altered.

SQL>
```

콘솔창에서  
빨간색 텍스트 입력 후  
실행

```
Microsoft Windows [Version 10.0.15063]
(c) 2017 Microsoft Corporation. All rights reserved.

C:\Users\BIT>sqlplus sys/oracle as sysdba

SQL*Plus: Release 11.2.0.2.0 Production on 수 7월 12
17:45:07 2017

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

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

SQL>ALTER USER hr IDENTIFIED BY hr ACCOUNT unlock ;

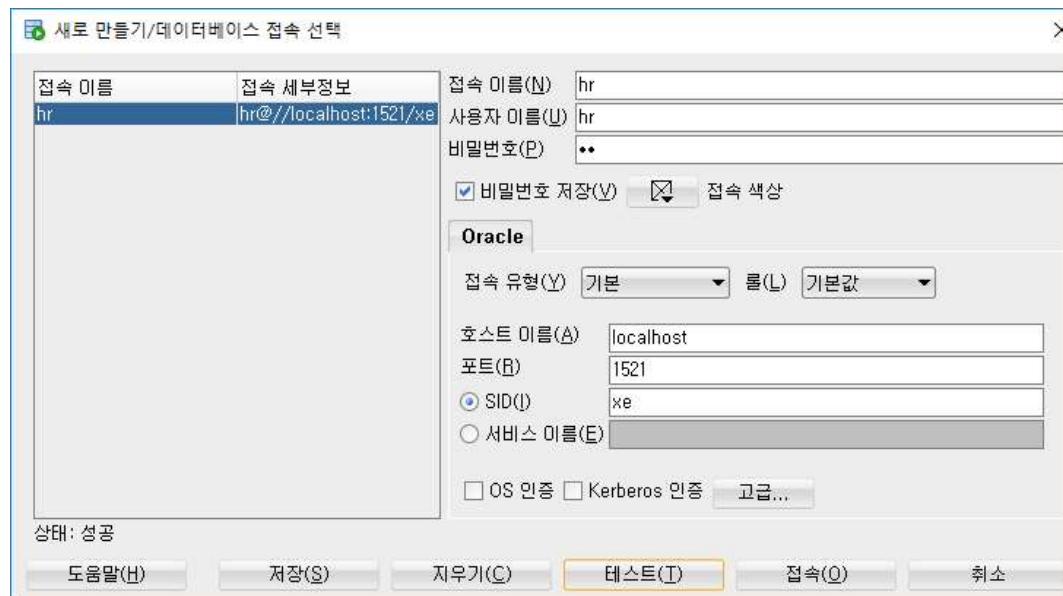
User altered.

SQL>
```

# Installation

## : 데이터베이스 접속 시도

- ▶ hr 계정 접속



접속 클릭

# Installation

## : SCOTT 계정 설정

- ▶ 11g XE에는 11g 정식버전에 있는 테스트 계정 SCOTT이 없음
- ▶ C:\를 선택해 XE를 설치했다면 SCOTT 계정 생성 파일 경로는  
*C:\oraclexe\app\oracle\product\11.2.0\server\rdbsms\admin\scott.sql*  
(반드시 탐색기에서 확인)
- ▶ sysdba 권한으로 SQL\*Plus에 접속  
SQL\*Plus 접속 상태에서 위 파일을 실행  
파일 실행시에는 전체 경로 앞에 @을 붙인다

```
C:\Users\BIT>sqlplus sys/oracle as sysdba
.
.
.
SQL>@C:\oraclexe\app\oracle\product\11.2.0\server\rd
bms\admin\scott.sql;
SQL>
```

# Installation

## : SCOTT 계정 설정

- ▶ 스크립트 실행 후 성공하면 에러 없이 커서가 바로 떨어짐
- ▶ 이 상태에서 SQL\*Plus에 접속된 사용자를 확인

```
SQL> show user
USER is 'SCOTT'
SQL>
```

- ▶ SCOTT 계정의 비밀번호를 tiger로 변경한다

```
SQL> alter user scott identified by tiger;
USER is 'SCOTT'
SQL>
```

- ▶ SQL\*Plus에서 빠져나오려면 exit를 사용

```
SQL> exit
Disconnected from Oracle Database 11g Express Edition Release
11.2.0.2.0 - 64bit Production
C:\Users\BIT>
```

- ▶ 이 과정으로 SCOTT 계정이 만들어지고 SCOTT 계정 내의 테스트용 테이블들이 만들어짐

# Installation

## : SCOTT 계정 설정

- ▶ [실습] SQL\*Plus를 이용, SCOTT 계정으로 접속하여 SCOTT 계정 테이블을 확인해 봅시다
  - ▶ 계정 내 테이블 확인:
    - ▶ SELECT \* from tab;

```
C:\Users\BIT> sqlplus scott/tiger

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

SQL> select * from tab;

TNAME                      TABTYPE
-----
BONUS                       TABLE
DEPT                        TABLE
EMP                         TABLE
SALGRADE                     TABLE

SQL>
```

# Installation

## : SCOTT 계정 설정

- ▶ [실습] SCOTT 계정 내 DEPT 테이블 구조를 확인해 봅시다

```
SQL> DESC DEPT;
      Name          Null?    Type
-----  -----
DEPTNO           NOT NULL NUMBER(2)
DNAME            VARCHAR2(14)
LOC              VARCHAR2(13)

SQL>
```

- ▶ 테이블 구조 확인
  - ▶ DESC <테이블명>

# Oracle Database

Oracle SQL

# Oracle SQL

Basic Query - SELECT 문의 기초

# SELECT

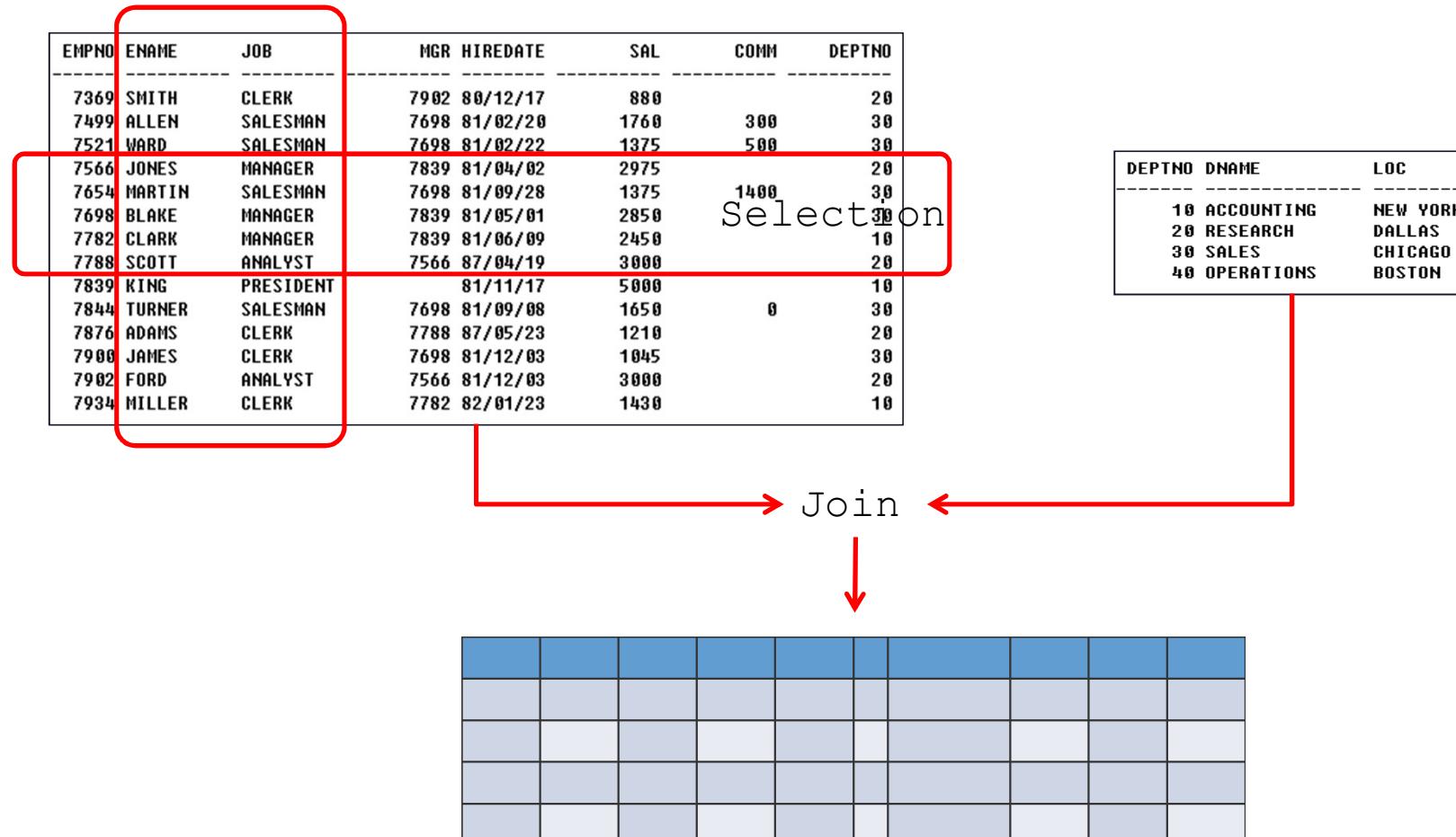
- ▶ 데이터베이스에서 원하는 데이터를 검색, 추출
- ▶ Syntax

```
SELECT [ALL|DISTINCT] 열_리스트  
FROM 테이블_리스트  
[WHERE 조건]  
[GROUP BY 열_리스트 [HAVING 그룹 조건]]  
[ORDER BY 열_리스트 [ASC | DESC]];
```

- ▶ 기능
  - ▶ Projection : 원하는 컬럼 선택
  - ▶ Selection : 원하는 튜플 선택
  - ▶ Join : 두 개의 테이블 결합
  - ▶ 기타 : 각종 계산, 정렬, 요약(Aggregation)

# SELECT의 기능

## Projection



# 기본 SELECT 문

- ▶ 형식

```
SELECT * | {[DISTINCT] column|expression [alias], ...}
FROM table
```

- ▶ 내용 설명

- ▶ \* : 모든 컬럼 반환
- ▶ DISTINCT : 중복된 결과 제거
- ▶ SELECT 컬럼명 : Projection
- ▶ FROM 대상 테이블
- ▶ ALIAS : 컬럼 이름 변경(표시용)
- ▶ Expression : 기본적인 연산 및 함수 사용 가능

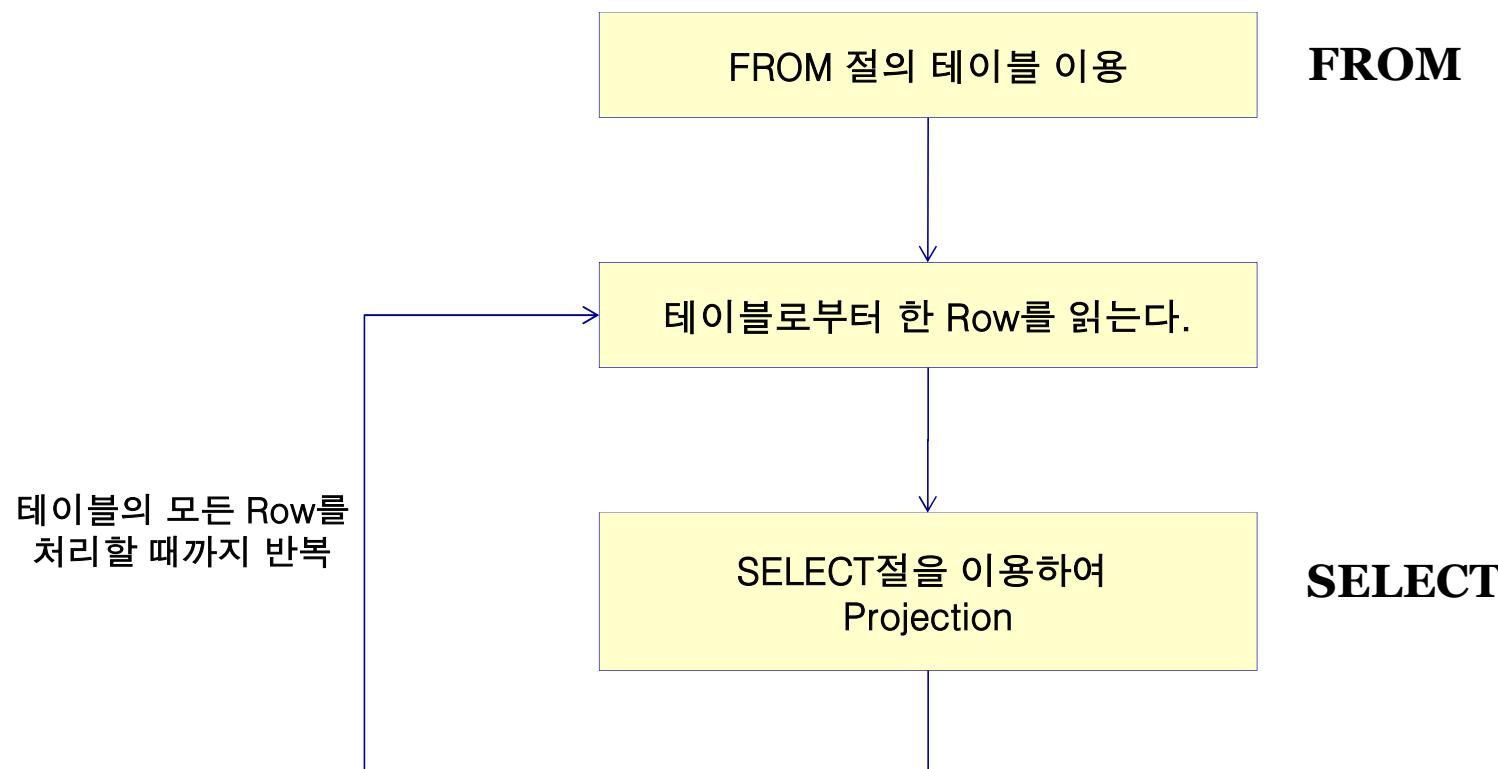
# 기본 SELECT 문의 예

- ▶ SELECT \* FROM emp;
- ▶ SELECT ename FROM emp;
- ▶ SELECT ename, job FROM emp;
- ▶ SELECT ename 이<sup>를</sup> FROM emp;

SQL> select * from emp;							
EMPNO	ENAME	JOB	MGR	HIREDATE	SAL	COMM	DEPTNO
7369	SMITH	CLERK	7902	80/12/17	800		20
7499	ALLEN	SALESMAN	7698	81/02/20	1600	300	30
7521	WARD	SALESMAN	7698	81/02/22	1250	500	30
7566	JONES	MANAGER	7839	81/04/02	2975		20
7654	MARTIN	SALESMAN	7698	81/09/28	1250	1400	30
7698	BLAKE	MANAGER	7839	81/05/01	2850		30
7782	CLARK	MANAGER	7839	81/06/09	2450		10
7788	SCOTT	ANALYST	7566	87/04/19	3000		20
7839	KING	PRESIDENT		81/11/17	5000		10
7844	TURNER	SALESMAN	7698	81/09/08	1500	0	30
7876	ADAMS	CLERK	7788	87/05/23	1100		20
7900	JAMES	CLERK	7698	81/12/03	950		30
7902	FORD	ANALYST	7566	81/12/03	3000		20
7934	MILLER	CLERK	7782	82/01/23	1300		10

# SELECT ... FROM 절의 처리

: Flow



# SELECT / FROM 절

- ▶ 모든 컬럼의 조회
  - ▶ 컬럼 목록에 \*를 표시하면 테이블 내 모든 컬럼을 Projection 한다

```
SELECT * FROM {TABLE명};
```

- ▶ SQL Recap:
  - ▶ 마지막은 세미콜론(;)
  - ▶ 대소문자 구분하지 않음
- ▶ 테이블 employees의 모든 튜플을 불러와 모든 컬럼을 Projection 한다.

```
SELECT * FROM employees;
```

- ▶ 테이블 departments의 모든 튜플을 불러와 모든 컬럼을 Projection 한다

```
SELECT * FROM departments;
```

# SELECT / FROM 절

- ▶ 원하는 컬럼의 조회 (Projection을 원하는 컬럼명을 지정)
  - ▶ 컬럼 목록에 \*를 표시하면 테이블 내 모든 컬럼을 Projection 한다

```
SELECT {컬럼명1}, {컬럼명2}, ...
FROM {TABLE명};
```

- ▶ 예: 다음 쿼리문을 보고 출력 결과를 예측해 봅시다
- ```
SELECT employee_id, first_name, last_name
FROM employees;
```
- ▶ [연습] hr.employees
    - ▶ 사원의 이름(first\_name)과 전화번호, 입사일, 급여를 출력해 봅시다
    - ▶ 사원의 이름(first\_name)과 성(last\_name), 급여, 전화번호, 입사일을 출력해 봅시다

# 산술연산(Arithmetic Operation)

## ▶ 기본적인 산술연산 사용 가능

- ▶ +, -, \*, /, 부호, 괄호 등
- ▶ 우선순위 : 부호 -> 괄호 -> \*, / -> +, -
- ▶ 컬럼명, 숫자
- ▶ 예
  - ▶ SELECT ename, (sal + 200) \* 12 FROM emp;

```
SQL> SELECT ename, (sal+200) * 12 FROM emp;  
  
ENAME      (SAL+200)*12  
-----  
SMITH      12000  
ALLEN      21600  
WARD       17400  
JONES      38100  
MARTIN     17400  
BLAKE      36600
```

# SELECT / FROM 절

: 산술연산 연습

- ▶ 단순 수식 계산하기

```
SELECT {산술식} FROM dual;
```

- ▶ dual : Oracle의 Pseudo Table. 특정 테이블이 아닌 오라클 시스템으로부터 값을 가져 올 때 사용
- ▶ 컬럼명 대신 산술식을 부여한다
- ▶ 필드 값의 산술연산

```
SELECT first_name, salary  
FROM employees;  
  
SELECT first_name, salary, salary*12  
FROM employees;  
  
SELECT first_name, salary, salary*12, (salary+300)*12  
FROM employees;
```

- ▶ [예제] 다음을 실행해보고  
오류의 원인이 무엇인지 알아봅시다

```
SELECT job_id*12  
FROM employees;
```

# NULL

- ▶ 아무런 값도 정해지지 않았음을 의미
- ▶ 어떠한 데이터타입에도 사용 가능
- ▶ NULL은 0이나 space 등과는 다르다
- ▶ NOT NULL 컬럼이나 Primary Key 속성의 컬럼에는 사용할 수 없음
- ▶ NULL을 포함한 산술식은 NULL
  - ▶ `SELECT sal, comm, (sal + comm) * 12 FROM emp;`
- ▶ NVL(expr1, expr2)
  - ▶ `expr1`이 NULL이면 `expr2`를 출력
  - ▶ 데이터타입이 호환 가능해야 함
  - ▶ `SELECT sal, comm, (sal + NVL(comm, 0)) * 12 FROM emp`

# Column Alias

- ▶ 컬럼의 출력 제목을 변경
- ▶ alias 내에 공백이다 특수문자를 포함하고자 한다면 큰따옴표(" ")를 사용
- ▶ 형태
  - ▶ SELECT ename name FROM emp;
  - ▶ SELECT ename as name FROM emp;
  - ▶ SELECT ename "name" FROM emp;
  - ▶ SELECT (sal + comm) "Annual Salary" From emp;

```
SQL> select empno no, ename as name, job "to do" from emp;
```

| NO   | NAME   | TO DO     |
|------|--------|-----------|
| 7369 | SMITH  | CLERK     |
| 7499 | ALLEN  | SALESMAN  |
| 7521 | WARD   | SALESMAN  |
| 7566 | JONES  | MANAGER   |
| 7654 | MARTIN | SALESMAN  |
| 7698 | BLAKE  | MANAGER   |
| 7782 | CLARK  | MANAGER   |
| 7788 | SCOTT  | ANALYST   |
| 7839 | KING   | PRESIDENT |
| 7844 | TURNER | SALESMAN  |
| 7876 | ADAMS  | CLERK     |

# Literal

- ▶ SELECT 절에 사용되는 문자, 숫자, Date 타입 등의 상수
- ▶ Date 타입이나 문자열은 작은따옴표(' ')로 둘러싸야 함
- ▶ 문자열 결합(Concatenation) 연산자 : ||
- ▶ 예
  - ▶ SELECT ename, 1000, SYSDATE FROM emp;
  - ▶ SELECT 'Name is ' || ename || 'and no is ' || empno FROM emp;

```
SQL> SELECT 'Name is ' || ename || ' and no is ' || empno FROM emp;  
  
'NAMEIS'||ENAME||'ANDNOIS'||EMPNO  
-----  
Name is SMITH and no is 7369  
Name is ALLEN and no is 7499  
Name is WARD and no is 7521  
Name is JONES and no is 7566  
Name is MARTIN and no is 7654  
Name is BLAKE and no is 7698  
Name is CLARK and no is 7782  
Name is SCOTT and no is 7788
```

# SELECT / FROM 절 : Alias 연습

- ▶ Projection 컬럼에 Alias 사용 예

```
SELECT employee_id as empNO, first_name "E-name", salary "급여"  
FROM employees;
```

- ▶ [예제] hr.employees 전체 튜플에 다음과 같이 Column Alias를 붙여 출력해 봅니다
  - ▶ 이름 : first\_name last\_name
  - ▶ 입사일: hire\_date
  - ▶ 전화번호 : phone\_number
  - ▶ 급여 : salary
  - ▶ 연봉 : salary \* 12

## SELECT / FROM 절 : NULL과 NVL 연습

- ▶ 다음 쿼리를 살펴봅시다

```
SELECT first_name, salary, commission_pct,  
       salary + salary * commission_pct  
  FROM employees;
```

- ▶ SQL Recap:
  - ▶ NULL이 포함된 산술계산은 NULL  
-> commission\_pct가 NULL인 사원은 최종 급여가 계산되지 않고 NULL로 출력
- ▶ NVL을 이용, 수정해 봅시다

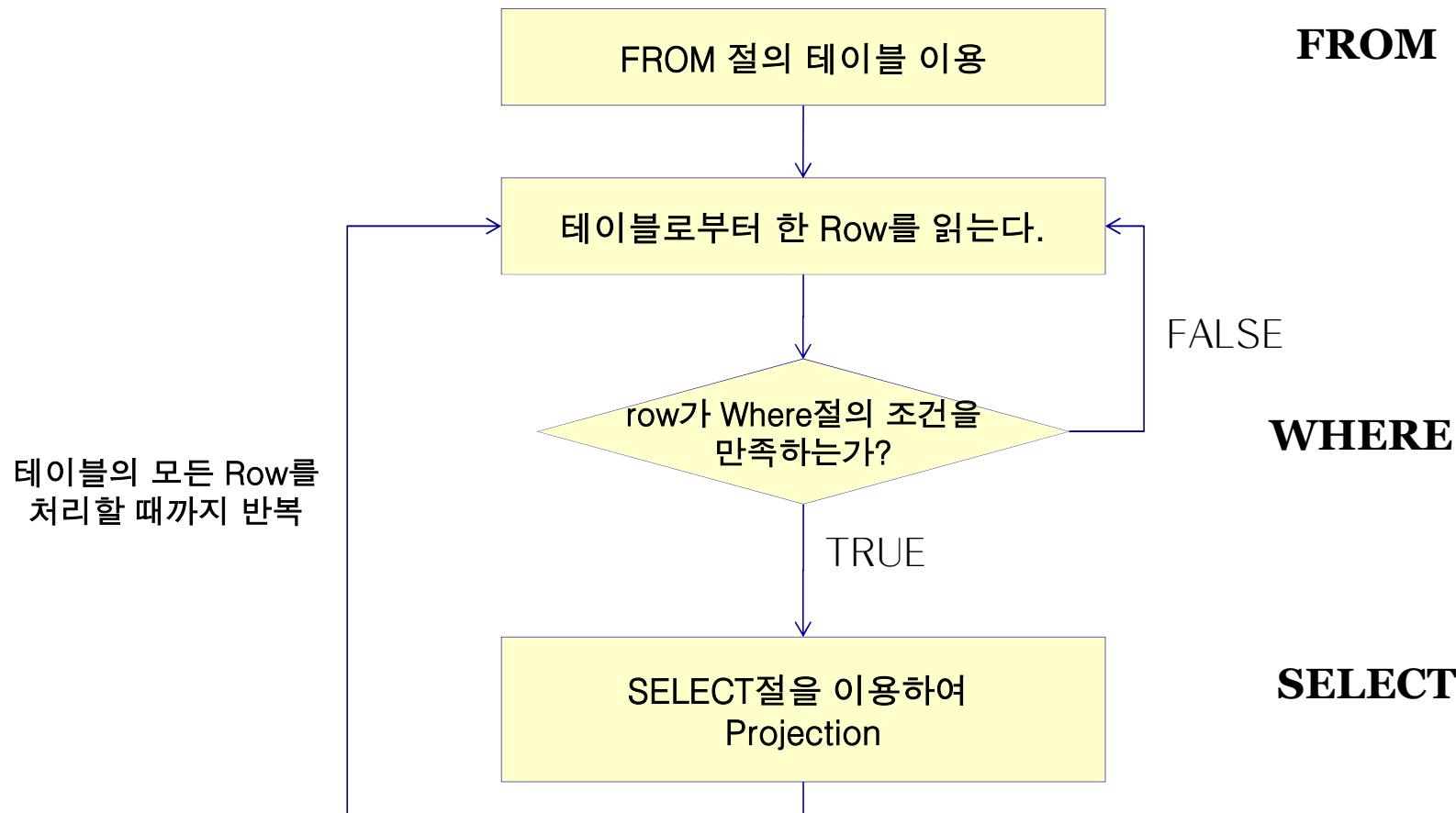
```
SELECT first_name, salary, nvl(commission_pct, 0),  
       salary + salary * nvl(commission_pct, 0)  
  FROM employees;
```

# WHERE

- ▶ 조건을 부여하여 만족하는 ROW를 선택(Selection)
- ▶ 연산자
  - ▶ =, !=, >, <, <=, >=
  - ▶ IN : 집합에 포함되는가?
  - ▶ BETWEEN a AND b : a와 b 사이인가?
  - ▶ IS NULL, IS NOT NULL : NULL 여부 검사
  - ▶ AND, OR : 둘 다 만족? 둘 중 하나만 만족?
  - ▶ NOT : 만족하지 않음?
  - ▶ ANY, ALL : 집합 중 어느 한 열, 집합 중 모든 열 (다른 비교 연산자와 함께 사용)
  - ▶ EXIST : 결과 Row가 한 개 이상 있는가? (Subquery에서 사용)

# WHERE 절의 처리

: Flow



# 비교 연산자

```
SELECT dname  
      FROM dept  
     WHERE deptno = 30;
```

```
SELECT ename, sal  
      FROM emp  
     WHERE sal >= 1500;
```

| Operator | Purpose                  |
|----------|--------------------------|
| =        | Equal to                 |
| <>       | Not equal to             |
| >        | Greater than             |
| >=       | Greater than or equal to |
| <        | Less than                |
| <=       | Less than or equal to    |

# 비교 연산자

```
SELECT dname  
      FROM dept  
     WHERE deptno IN (10, 20);
```

```
SELECT ename, sal  
      FROM emp  
     WHERE job = ANY('ANALYST', 'CLERK');
```

```
SELECT ename, sal  
      FROM emp  
     WHERE sal > ALL(2000, 3000);
```

| Operator                   | Purpose                                                                                     |
|----------------------------|---------------------------------------------------------------------------------------------|
| IN ( <i>list</i> )         | Equal to any member of <i>list</i> .                                                        |
| ANY ( <i>list</i> )        | Compares a value to each value in a <i>list</i> . Must be preceded by =, !=, >, <, <=, >=.  |
| ALL ( <i>list</i> )        | Compares a value to every value in a <i>list</i> . Must be preceded by =, !=, >, <, <=, >=. |
| EXISTS ( <i>subquery</i> ) | TRUE if a <i>subquery</i> returns at least one row.                                         |

- ANY, ALL 연산자의 앞에는 비교연산자가 와야 함
- IN 연산자는 = ANY 연산자와 같은 결과
- NOT IN 연산자는 <> ALL 연산자와 같은 결과

# 비교 연산자

```
SELECT ename  
      FROM emp  
     WHERE hiredate BETWEEN  
           '82/01/01' AND  
           '82/12/31';
```

```
SELECT dname  
      FROM dept  
     WHERE dname LIKE 'A%';
```

```
SELECT ename  
      FROM emp  
     WHERE comm IS NULL;
```

| Operator                         | Purpose                                                                                                                                                                               |
|----------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| BETWEEN $a$<br>AND $b$           | greater than or equal to $a$ and less than or equal to $b$ .                                                                                                                          |
| $a$ LIKE $b$<br>[ESCAPE ' $c$ '] | TRUE if $a$ does match the pattern $b$ .<br>‘%’ and ‘_’ are wildcard characters within $b$ . A wildcard character is treated as a literal if preceded by the escape character ‘ $c$ ’ |
| IS NULL                          | Null test                                                                                                                                                                             |

## WHERE 절

: 비교 연산자 연습

- ▶ [예제] hr.employees

- ▶ 급여가 15000 이상인 사원들의 이름과 연봉을 출력하십시오.
- ▶ 07/01/01일 이후 입사자들의 이름과 입사일을 출력하십시오.
- ▶ 이름이 'Lex'인 사원의 연봉과 입사일, 부서 ID를 출력하십시오.
- ▶ 부서 ID가 10인 사원의 명단이 필요합니다.

# LIKE 연산

- ▶ Wildcard를 이용한 문자열 부분 매칭
- ▶ Wildcard
  - ▶ % : 임의의 길이의 문자열(공백 문자 포함)
  - ▶ \_ : 한 글자
- ▶ Escape
  - ▶ ESCAPE 뒤의 문자열로 시작하는 문자는 Wildcard 가 아닌 것으로 해석
- ▶ 예
  - ▶ ename LIKE 'KOR%' : KOR로 시작하는 모든 문자열 (KOR 포함)
  - ▶ ename LIKE 'KOR\_-' : KOR 뒤에 하나의 문자가 오는 모든 문자열
  - ▶ ename LIKE 'KOR/%%' ESCAPE '/' : 'KOR%'로 시작하는 모든 문자열

## 논리 연산자

- ▶ 논리 연산자를 이용하여 복잡한 질의 조건을 줄 수 있으며, 전체 조건식의 연산 결과가 TRUE인 Row만 선택된다

|   |                      |
|---|----------------------|
| 1 | Arithmetic operators |
| 2 | Comparison operators |
| 3 | NOT                  |
| 4 | AND                  |
| 5 | OR                   |

# 논리 연산자의 결과값

## ▶ NULL을 주의

- ▶ NULL이 있으면 기본적으로 NULL, 확실히 답이 나오는 경우만 계산 가능

| NOT | TRUE  | FALSE | NULL |
|-----|-------|-------|------|
|     | FALSE | TRUE  | NULL |

| AND   | TRUE  | FALSE | NULL  |
|-------|-------|-------|-------|
| TRUE  | TRUE  | FALSE | NULL  |
| FALSE | FALSE | FALSE | FALSE |
| NULL  | NULL  | FALSE | NULL  |

| OR    | TRUE | FALSE | NULL |
|-------|------|-------|------|
| TRUE  | TRUE | TRUE  | TRUE |
| FALSE | TRUE | FALSE | NULL |
| NULL  | TRUE | NULL  | NULL |

## WHERE 절

: 논리 연산자 연습

- ▶ 조건이 두 개 이상일 때 한꺼번에 조회하기

```
select first_name, salary  
from employees  
where salary >= 14000  
and salary <= 17000;
```

• 급여가 14000 이상 17000이하인 사원의  
이름과 급여를 출력

- ▶ [예제] hr.employees
  - ▶ 급여가 14000 이하이거나 17000 이상인 사원의 이름과 급여를 출력하십시오.
  - ▶ 부서 ID가 90인 사원 중, 급여가 20000 이상인 사원은 누구입니까?

# WHERE 절

## : BETWEEN 연습

- ▶ BETWEEN 연산자를 이용한 특정 구간의 값 출력

```
select first_name, salary
from employees
where salary between 14000 and 17000;
```

• 급여가 14000 이상 17000이하인 사원의 이름과 급여를 출력하시오

- ▶ 앞서 비교 연산자를 이용한 쿼리문과 비교해 봅시다
  - ▶ 작은 값을 앞쪽에, 큰 값을 뒤쪽에 부여
  - ▶ 유용하지만 느린 연산자에 속함
- 
- ▶ [예제] hr.employees
    - ▶ 입사일이 07/01/01 ~ 07/12/31 구간에 있는 사원의 목록을 출력해 봅시다

# WHERE 절

## : IN 연습

- ▶ 여러 조건을 검사하기 위해 IN 연산자를 활용합니다

```
SELECT first_name, last_name, salary  
FROM employees  
WHERE first_name IN ('Neena', 'Lex', 'John');
```

- ▶ [예제]
  - ▶ 부서 ID가 10, 20, 40인 사원의 명단을 출력해 봅시다
  - ▶ MANAGER ID가 100, 120, 147 인 사원의 명단을 출력해 봅시다
    - ▶ 비교연산자+논리연산자를 이용하여 출력해 보고
    - ▶ IN 연산자를 이용해 출력해 봅시다
    - ▶ 두 쿼리를 비교해 보고 IN 연산자의 유용한 점을 생각해 봅시다

# WHERE 절

## : Like 연습

- ▶ SQL Recap:
  - ▶ % : 임의의 길이의 문자열(공백 문자 가능)
  - ▶ \_ : 임의의 한 글자
- ▶ [예제] hr.employees
  - ▶ 이름에 **am**을 포함한 사원의 이름과 급여를 출력해 봅시다
  - ▶ 이름의 두 번째 글자가 **a**인 사람의 이름과 급여를 출력해 봅시다
  - ▶ 이름의 네 번째 글자가 **a**인 사원의 이름을 출력해 봅시다
  - ▶ 이름이 **4**글자인 사원 중 끝에서 두 번째 글자가 **a**인 사원의 이름을 출력해 봅시다

# 연산자 우선 순위

1. Arithmetic Operators
2. Concatenation Operators
3. Comparison Conditions
4. IS [NOT] NULL, LIKE, [NOT] IN
5. [NOT] BETWEEN
6. NOT Logical condition
7. AND Logical condition
8. OR logical condition

# ORDER BY

- ▶ 주어진 컬럼 리스트의 순서로 결과를 정렬
- ▶ 결과 정렬 방법
  - ▶ ASC : 오름차순 (작은값 -> 큰값) - default
  - ▶ DESC : 내림차순 (큰 값 -> 작은 값)
- ▶ 정렬 기준은 여러 컬럼에 지정 가능
- ▶ 컬럼 이름 대신 Alias, expr, SELECT 절 상의 순서(1, 2, 3...)도 사용 가능
  - ▶ 예) `SELECT * FROM emp ORDER BY deptno, sal DESC`  
: 의미 해석 = 부서 번호 순으로 정렬한 후, `sal`이 높은 사람부터 출력

# ORDER BY

- ▶ ORDER BY 절을 이용하여 데이터를 사용하기 편리하게 정렬하기

```
SELECT first_name, salary  
FROM employees  
ORDER BY salary DESC;
```

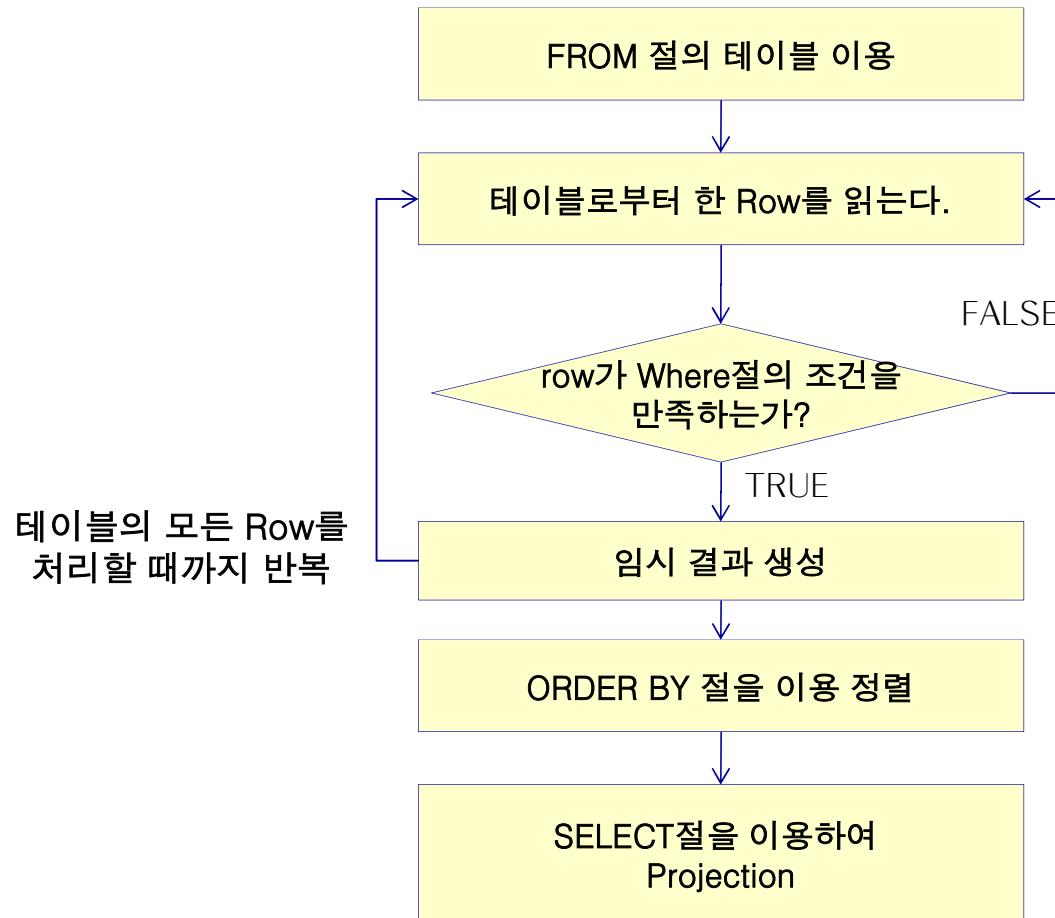
```
SELECT first_name, salary  
FROM employees  
WHERE salary >= 9000  
ORDER BY salary DESC;
```

- ▶ 기본값: 오름차순 (ASC: 작은 값 -> 큰 값 순) <-> 내림차순(DESC)은 반대
  - ▶ 한글: 가, 나, 다, 라 ...
  - ▶ 영어: A, B, C, D
  - ▶ 숫자: 1, 2, 3, 4
  - ▶ 날짜 : 오래된 날짜 -> 최근 날짜 순
- ▶ 정렬 조건이 복수일 경우는 ,로 구분하여 명시

## ORDER BY

- ▶ [연습] hr.employees
  - ▶ 부서 번호를 오름차순으로 정렬하고 부서번호, 급여, 이름을 출력하십시오
  - ▶ 급여가 10000 이상인 직원의 이름을 급여 내림차순(높은 급여 -> 낮은 급여)으로 출력하십시오
  - ▶ 부서 번호, 급여, 이름 순으로 출력하되 부서번호 오름차순, 급여 내림차순으로 출력하십시오.

# ORDER BY 절의 처리



# Oracle SQL

단일행 함수 (Single Row Function)

# SQL Functions

- ▶ Single-Row Function : 하나의 Row를 입력으로 받는 함수
  - ▶ 숫자 함수
  - ▶ 문자 함수
  - ▶ 날짜 함수
  - ▶ 변환 함수
  - ▶ 기타 함수
- ▶ Aggregation Function : 집합 함수
- ▶ Analytic Function : 분석 함수
- ▶ Regular Expression : 정규표현식 (Oracle 10g 이상)

# 단일행 함수

## : 문자열 함수

| Function             | 설명                                       |
|----------------------|------------------------------------------|
| CONCAT(s1,s2)        | 문자열 결합                                   |
| INITCAP(s)           | 첫글자만 대문자로 변경                             |
| LOWER(s)             | 소문자로 변경                                  |
| UPPER(s)             | 대문자로 변경                                  |
| LPAD(s1,n,s2)        | 문자열의 왼쪽 채움 (길이:n, 채움문자 s2)               |
| RPAD(s1,n,s2)        | 문자열 오른쪽 채움 (길이:n, 채움문자 s2)               |
| LTRIM(s,c)           | 문자열 왼쪽 c문자열 제거                           |
| RTRIM(s,c)           | 문자열 오른쪽 c문자열 제거                          |
| CHR(n)               | ASCII값이 n인 문자 반환                         |
| REPLACE(s,p,r)       | 문자열 치환, S속의 p문자열을 r로 치환                  |
| SUBSTR(s,m,n)        | 부분 문자열, m번째부터 길이 n인 문자열 반환               |
| TRANSLATE(s,from,to) | s에서 from 문자열의 각 문자를 to문자열의 각 문자로 변환      |
| ASCII(s)             | ASCII값 반환                                |
| INSTR(s1,s2,m,n)     | 문자열 검색, s1의 m번째부터 s2 문자열이 나타나는 n번째 위치 반환 |
| LENGTH(s)            | 문자열 길이 반환                                |

# 단일행 함수

## : 문자열 함수 사용 예

### ▶ 대소문자 변환

| Function                   | Result          |
|----------------------------|-----------------|
| LOWER('Database system')   | database system |
| UPPER('Database system')   | DATABASE SYSTEM |
| INITCAP('Database system') | Database System |

### ▶ 문자열 조작

| 함수                            | 결과         |
|-------------------------------|------------|
| CONCAT('Data', 'Base')        | DataBase   |
| SUBSTR('Database',2,4)        | atab       |
| LENGTH('database')            | 8          |
| INSTR('Database', 'b')        | 5          |
| LPAD(salary,10,'*')           | *****24000 |
| RPAD(salary, 10, '*')         | 24000***** |
| TRIM('#' FROM '##Database##') | Database   |

# 단일행 함수

## : 주요 문자열 함수

### ▶ INITCAP(컬럼명)

- ▶ 영어의 첫 글자만 대문자로 출력하고 나머지는 소문자로 출력하는 함수

```
SELECT email, INITCAP(email), department_id  
      FROM employees  
     WHERE department_id = 100;
```

### ▶ LOWER(컬럼명) / UPPER(컬럼명)

- ▶ 문자열 값을 전부 소문자/대문자로 변경하는 함수

```
SELECT first_name, LOWER(first_name), UPPER(first_name)  
      FROM employees  
     WHERE department_id = 100;
```

- ▶ 대소문자 구분 없이 문자열을 검색하고자 할 때 유용하게 사용

# 단일행 함수

## : 주요 문자열 함수

- ▶ SUBSTR(컬럼명, 시작위치, 글자수)
  - ▶ 주어진 문자열에서 특정 길이의 문자열을 구하는 함수

```
SELECT first_name, SUBSTR(first_name,1,3), SUBSTR(first_name,-3,2)
  FROM employees
 WHERE department_id = 100;
```

- ▶ 시작 위치가 양수인 경우 왼쪽부터 검색하여 글자수만큼 추출
- ▶ 시작 위치가 음수인 경우 오른쪽 -> 왼쪽 검색을 한 후 글자수만큼 추출

# 단일행 함수

## : 주요 문자열 함수

- ▶ LPAD(컬럼명, 자리수, '채울문자') / RPAD(컬럼명, 자리수, '채울문자')
  - ▶ LPAD() : 왼쪽 공백에 특별한 문자로 채우기
  - ▶ RPAD() : 오른쪽 공백에 특별한 문자로 채우기

```
SELECT first_name,
       LPAD(first_name,10,'*'),
       RPAD(first_name,10,'*')
FROM employees;
```

# 단일행 함수

## : 주요 문자열 함수

- ▶ REPLACE(컬럼명, 문자열1, 문자열2)
  - ▶ 컬럼명에서 문자열1을 문자열2로 바꾸는 함수

```
SELECT first_name,  
       REPLACE(first_name, 'a', '*')  
FROM employees  
WHERE department_id =100;
```

```
SELECT first_name,  
       REPLACE(first_name, 'a', '*'),  
       REPLACE(first_name, SUBSTR(first_name, 2, 3), '***')  
FROM employees  
WHERE department_id =100;
```

# 단일행 함수

## : 숫자 함수

| Function   | 설명                | Example        | Result |
|------------|-------------------|----------------|--------|
| ABS(n)     | 절대값               | ABS(-5)        | 5      |
| CEIL(n)    | n 보다 크거나 같은 최소 정수 | CEIL(-2.4)     | -2     |
| FLOOR(n)   | n 보다 작거나 같은 최대 정수 | FLOOR(-2.4)    | -3     |
| MOD(m,n)   | 나머지               | MOD(13,2)      | 1      |
| POWER(m,n) | m의 n승             | POWER(2,3)     | 8      |
| ROUND(m,n) | 소수점아래 n자리까지 반올림   | ROUND(4.567,2) | 4.57   |
| TRUNC(m,n) | 소수점아래 n자리미만 버림    | TRUNC(4.567,2) | 4.56   |
| SIGN(n)    | 부호 (1, 0, -1)     | SIGN(-10)      | -1     |

## 단일행 함수

### : 주요 숫자 함수

- ▶ ROUND(숫자, 출력을 원하는 자리수)

- ▶ 주어진 숫자의 반올림을 하는 함수

```
SELECT ROUND(123.346, 2) "r2",
       ROUND(123.456, 0) "r0",
       ROUND(123.456, -1) "r-1"
FROM dual;
```

- ▶ TRUNC(숫자, 출력을 원하는 자리수)

- ▶ 주어진 숫자의 버림을 하는 함수

```
SELECT TRUNC(123.346, 2) "r2",
       TRUNC(123.456, 0) "r0",
       TRUNC(123.456, -1) "r-1"
FROM dual;
```

# Date 타입

## ▶ Oracle의 Date Type

- ▶ century, year, month, day, hours, minutes, seconds 등을 포함한 내부 표현 (7bytes)
- ▶ Date Format에 따라 입/출력됨
- ▶ 기본 Date Format : 'RR/MM/DD' or 'DD-MON-RR'
  - ▶ RR은 Y2K를 고려, 2자리 년도 표기 (00 ~ 49: 2000년대 / 50~99: 1900년대)
- ▶ 포맷 확인
  - ▶ SELECT value FROM nls\_session\_parameters WHERE parameter = 'NLS\_DATE\_FORMAT';

Oracle RR Format

| 현재 연도       | 처리 연도   | YY          | RR          |
|-------------|---------|-------------|-------------|
| 1950 – 1999 | 00 – 49 | 1900 – 1949 | 2000 – 2049 |
|             | 50 – 99 | 1950 – 1999 | 1950 – 1999 |
| 2000 – 2049 | 00 – 49 | 2000 – 2049 | 2000 – 2049 |
|             | 50 – 99 | 2050 – 2099 | 1950 – 1999 |

# 단일행 함수

## : Date 함수

| Function              | Purpose                 |
|-----------------------|-------------------------|
| ADD_MONTHS(d,n)       | d날짜에 n달 더함              |
| LAST_DAY(d)           | d의 달의 마지막 날             |
| MONTHS_BETWEEN(d1,d2) | d1, d2사이의 달 수           |
| NEW_TIME(d,z1,z2)     | z1타임존의 d에서 z2타임존의 날짜 생성 |
| NEXT_DAY(d,day)       | d날 후의 첫 day요일의 날짜       |
| ROUND(d,fmt)          | fmt에 따른 날짜 반올림          |
| TRUNC(d,fmt)          | fmt에 따른 날짜 버림           |
| SYSDATE               | 현재 날짜 시간 반환             |

# 단일행 함수

: Date 함수 사용 예

| FUNCTION                                 | RESULT      |
|------------------------------------------|-------------|
| MONTHS_BETWEEN ('01-SEP-95','11-JAN-94') | 19.677419   |
| ADD_MONTHS ('11-JAN-94',6)               | '11-JUL-94' |
| NEXT_DAY ('01-SEP-95','FRIDAY')          | '08-SEP-95' |
| LAST_DAY('01-FEB-95')                    | '28-FEB-95' |

| 현재날짜를 '25-JUL-95'가정      |           |
|--------------------------|-----------|
| ROUND(SYSDATE,'MONTH')   | 01-AUG-95 |
| ROUND(SYSDATE , 'YEAR')  | 01-Jan-96 |
| TRUNC(SYSDATE , 'MONTH') | 01-JUL-95 |
| TRUNC(SYSDATE , 'YEAR')  | 01-JAN-95 |

## 단일행 함수

: 주요 Date 함수

### ▶ SYSDATE

- ▶ 현재 날짜와 시간을 출력해주는 함수

```
SELECT sysdate  
FROM dual;
```

```
SELECT sysdate  
FROM employees;
```

### ▶ MONTH\_BETWEEN(d1, d2)

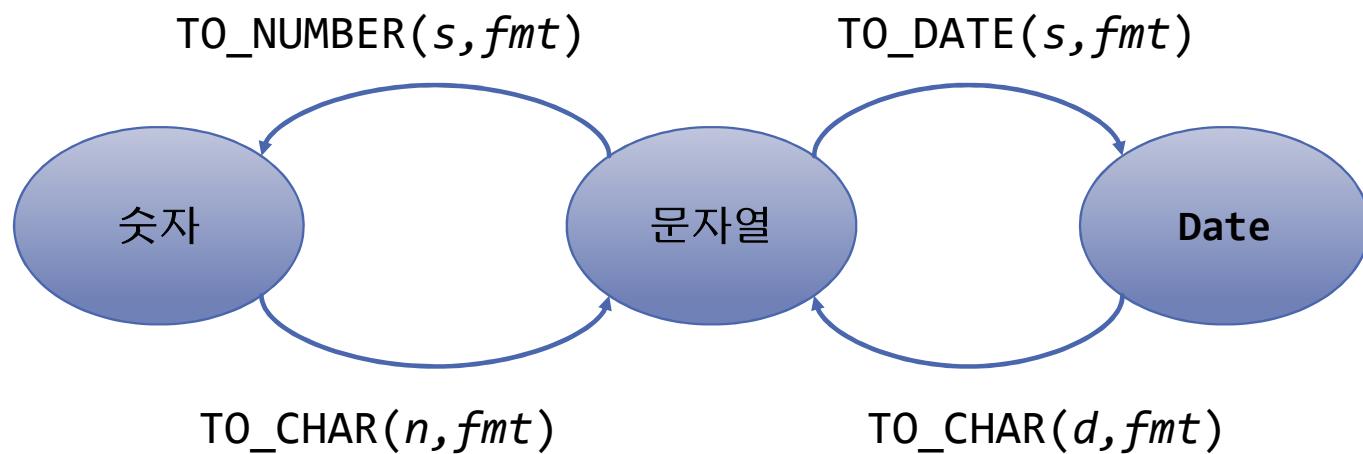
- ▶ d1 날짜와 d2 날짜 사이의 개월수를 출력하는 함수

```
SELECT MONTH_BETWEEN(sysdate, hire_date)  
FROM employees  
WHERE department_id = 110;
```

# 단일행 함수

## : 변환 함수

- ▶ 목시적 변환 : 변환 함수 없어도 어느 정도 자동으로 변환됨
- ▶ 자동으로 변환되지 않을 때는 명시적 변환 함수를 사용



# 단일행 함수

## : Date 변환 포맷

- ▶ 예시는 American 포맷인 경우

| fmt         | Description              | Example           | fmt            | Description                                   | Example            |
|-------------|--------------------------|-------------------|----------------|-----------------------------------------------|--------------------|
| SS          | Second.                  | 0 – 59            | W              | Week of month.                                | 1 – 5              |
| SSSS        | Seconds past midnight.   | 0 – 86399         | WW             | Week of year.                                 | 1 – 53             |
| MI          | Minute.                  | 0 – 59            | MM             | Two-digit numeric abbreviation of month.      | 1 – 12             |
| HH,<br>HH24 | Hour of day.             | 0 – 12,23         | MON            | Abbreviated name of month.                    | JAN – DEC          |
| AM,PM       | Meridian indicator.      | AM,PM             | MONTH          | Name of month.                                | JANUARY – DECEMBER |
| DD          | Day of month.            | 1 – 31            | Q              | Quarter of year.                              | 1 – 4              |
| DAY         | Name of day.             | SUNDAY – SATURDAY | RM             | Roman numeral month.                          | I – XII            |
| DY          | Abbreviated name of day. | SUN – SAT         | AD,BC          | AD, BC Indicator.                             | AD, BC             |
| D           | Day of week.             | 1 – 7             | Y,YY,YY<br>Y   | 1,2,3-digit year.                             |                    |
| DDD         | Day of year.             | 1 – 366           | YYYY,<br>SYYYY | 4-digit year. "S" prefixes BC dates with "-". |                    |

# 단일행 함수

## : Date 변환 포맷 (계속)

- ▶ 예시는 American 포맷인 경우

| fmt         | Description                                                                                                                                                                                                                                                                          | Example |
|-------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|---------|
| YEAR, SYEAR | Year, spelled out. "S" prefixes BC dates with "-".                                                                                                                                                                                                                                   |         |
| RR          | Given a year with 2 digits.<br><i>Returns a year in the next century if the year is &lt;50 and the last 2 digits of the current year are &gt;=50.</i><br><i>Returns a year in the preceding century if the year is &gt;=50 and the last 2 digits of the current year are &lt;50.</i> |         |
| RRRR        | Round year.                                                                                                                                                                                                                                                                          |         |
| CC, SCC     | One greater than the first two digits of a four-digit year;<br>"S" prefixes BC dates with "-".                                                                                                                                                                                       |         |
| J           | Julian day. the number of days since January 1, 4712 BC.                                                                                                                                                                                                                             |         |
| SP          | Spelled number.                                                                                                                                                                                                                                                                      |         |
| TH          | Ordinal number.                                                                                                                                                                                                                                                                      |         |

## 단일행 함수

: 변환 연습

- ▶ TO\_CHAR(날짜, '출력형식')

```
SELECT sysdate,  
       TO_CHAR(sysdate, 'YYYY-MM-DD HH24:MI:SS')  
FROM dual;
```

# 단일행 함수

## : 숫자 변환 포맷

| fmt | Description        | Example |
|-----|--------------------|---------|
| 9   | 숫자                 | 99999   |
| 0   | 강제로 0 출력           | 09999   |
| ,   | 지정된 위치에 ,          | 99,999  |
| .   | 소수점                | 999.99  |
| \$  | \$ 마크              | \$99999 |
| FM  | 앞부분의 채움 문자(공백) 없앰. | FM90.9  |
| L   | Local 화폐단위         | L99,999 |
| MI  | 음수에 – 부호           | 99999MI |
| PR  | 음수에 괄호             | 99999PR |
| RN  | 로마자 (대소문자 따라 다름)   | RN rn   |
| S   | 부호 기호              | S99,999 |
| X   | 16진수               | XXX xxx |

## 단일행 함수

: 숫자 변환 포맷 연습

- ▶ TO\_CHAR(숫자, '출력형식')
  - ▶ 숫자형 -> 문자형으로 변환하기

```
SELECT first_name, to_char(salary*12, '$999,999.99') "SAL"  
FROM employees  
WHERE department_id = 110;
```

# 단일행 함수

## : 기타 함수

- ▶ NULL 관련
  - ▶ NVL(expr1, expr2) : expr1이 NULL이면 expr2, 아니면 expr1
  - ▶ NVL2(expr1, expr2, expr3) : expr1이 NULL이면 expr3, 아니면 expr2
  - ▶ NULLIF(expr1, expr2) : 두 식이 같으면 NULL, 아니면 expr1
  - ▶ COALESCE(expr1, expr2, ... exprN) : 첫 NOT NULL인 식 반환, 없으면 exprN
- ▶ 데이터 타입에 유의한다
- ▶ 예
  - ▶ 

```
SELECT ename, NVL(TO_CHAR(mgr), 'No Manager')
  FROM emp;
```

# Conditional Expression

## ▶ CASE

```
CASE {expr1} WHEN {expr2} THEN {expr3}
          [WHEN {expr4} THEN {expr5}]
          ...
          ELSE {expr6}]
```

- ▶ IF - THEN - ELSE와 비슷한 로직을 제공
- ▶ expr1이 expr2와 일치할 때 -> expr3
- ▶ expr1이 expr4와 일치할 때 -> expr5
- ▶ 만족하는 조건이 없으면 -> expr6

## ▶ DECODE

```
DECODE( {value}, {if1}, {then1}
        [, {if2}, {then2}]
        , {else} )
```

# Conditional Expression

## : Example

- ▶ CASE

```
▶ SELECT ename, job, sal, CASE job WHEN 'CLERK' THEN 1.10 * sal  
      WHEN 'MANAGER' THEN 1.15 * sal  
      WHEN 'PRESIDENT' THEN 1.20 * sal  
      ELSE sal END REVISED_SALARY  
FROM emp;
```

- ▶ DECODE

```
▶ SELECT ename, job, sal, DECODE(job, 'CLERK', 1.10 * sal,  
      'MANAGER', 1.15 * sal,  
      'PRESIDENT', 1.20 * sal,  
      sal) REVISED_SALARY  
FROM emp;
```

# Conditional Expression

## : Example

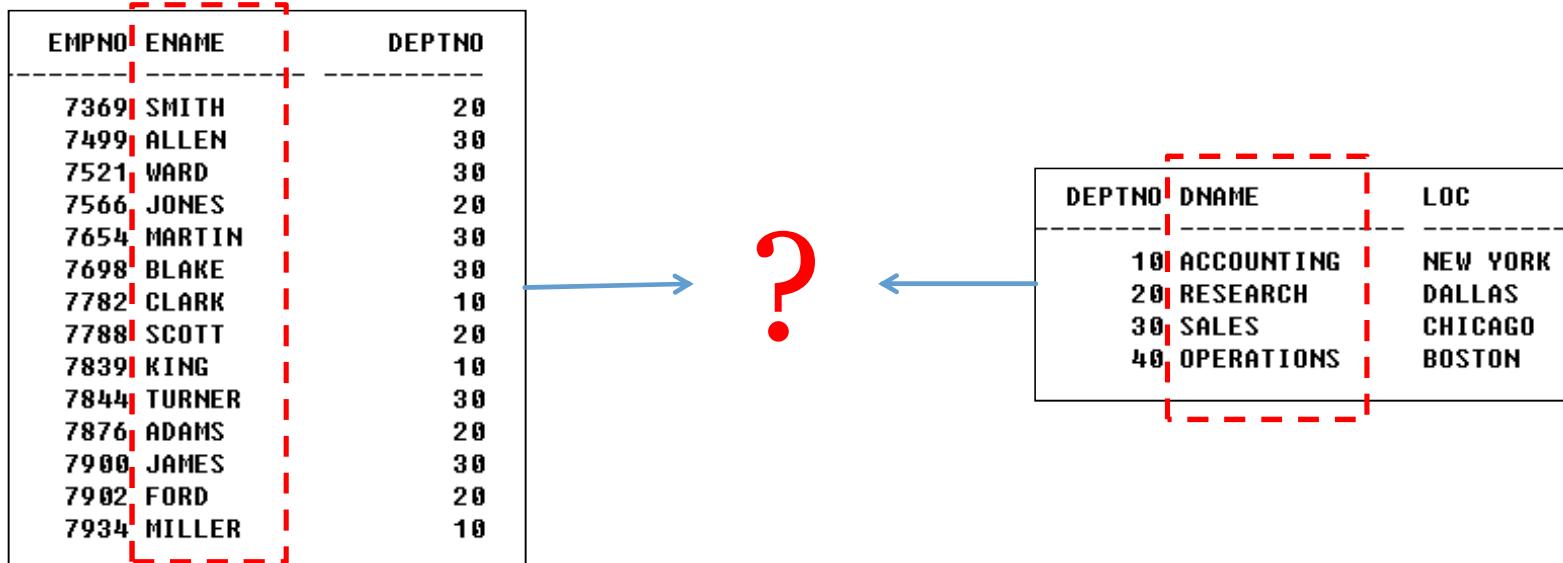
- ▶ [연습] hr.employees
  - ▶ 직원의 이름, 부서, 팀을 출력하십시오
  - ▶ 팀은 코드로 결정하며 다음과 같이 그룹 이름을 출력합니다
    - ▶ 부서 코드가 10 ~ 30이면: 'A-GROUP'
    - ▶ 부서 코드가 40 ~ 50이면: 'B-GROUP'
    - ▶ 부서 코드가 60 ~ 100이면 : 'C-GROUP'
    - ▶ 나머지 부서는 : 'REMAINDER'

# Oracle SQL

JOIN

# JOIN

- ▶ 둘 이상의 테이블을 합쳐 하나의 큰 테이블로 만드는 방법
- ▶ 필요성
  - ▶ 관계형 모델에서는 데이터의 일관성이나 효율을 위하여 데이터의 중복을 최소화 (정규화)
  - ▶ Foreign Key를 이용하여 참조
  - ▶ 정규화 된 테이블로부터 결합된 형태의 정보를 추출할 필요가 있음
  - ▶ 예) 직원의 이름과 직원이 속한 부서명을 함께 보고 싶다면?



# 카티션 프로덕트

- ▶ 두 테이블에서 그냥 결과를 선택한다면
  - ▶ SELECT ename, dname from emp, dept
  - ▶ 결과 : 두 테이블 행들의 가능한 모든 쌍이 추출
  - ▶ 일반적으로 사용자가 원하는 결과가 아님
- ▶ Cartesian Product
$$X \times Y = \{(x, y) | x \in X \text{ and } y \in Y\}$$
- ▶ Cartesian Product를 막기 위해서는 올바른 JOIN 조건을 WHERE 절에 부여해야 함
- ▶ 양쪽 테이블로부터 조합 가능한 모든 쌍이 선택되기 때문에 Cross Join이라 불리기도 함



| ENAME  | DNAME      |
|--------|------------|
| SMITH  | ACCOUNTING |
| ALLEN  | ACCOUNTING |
| WARD   | ACCOUNTING |
| JONES  | ACCOUNTING |
| MARTIN | ACCOUNTING |
| BLAKE  | ACCOUNTING |
| CLARK  | ACCOUNTING |
| SCOTT  | ACCOUNTING |
| ...    |            |
| ALLEN  | OPERATIONS |
| WARD   | OPERATIONS |
| JONES  | OPERATIONS |
| MARTIN | OPERATIONS |
| BLAKE  | OPERATIONS |
| CLARK  | OPERATIONS |
| SCOTT  | OPERATIONS |
| KING   | OPERATIONS |
| TURNER | OPERATIONS |
| ADAMS  | OPERATIONS |
| JAMES  | OPERATIONS |
| FORD   | OPERATIONS |
| MILLER | OPERATIONS |

56 개의 행이 선택되었습니다.

# Simple Join

## ▶ Syntax

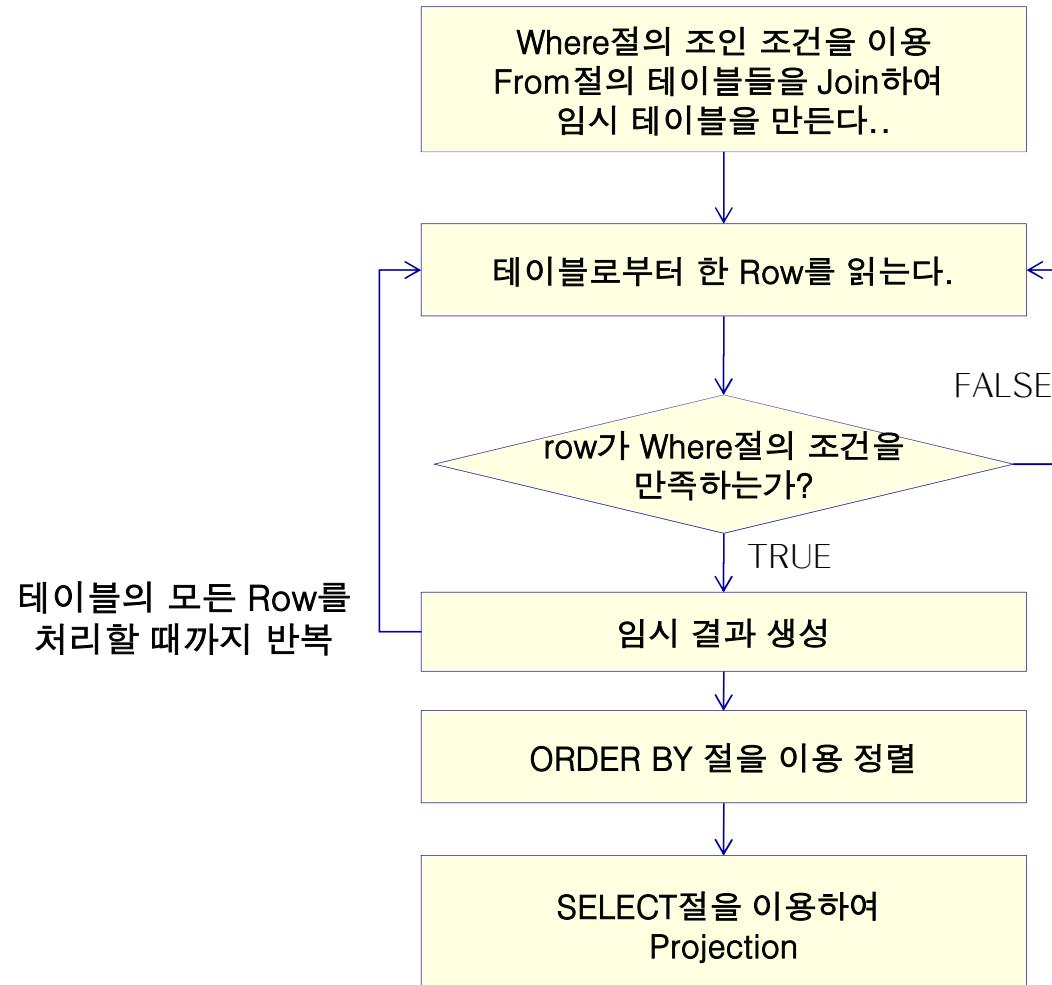
```
SELECT t1.col1, t1.col2, t2.col1 ...
  FROM Table1 t1, Table2 t2
 WHERE t1.col3 = t2.col3
```

## ▶ 설명

- ▶ **FROM** 절에 필요한 테이블을 모두 적는다
- ▶ 컬럼 이름의 모호성 (어느 테이블에 속하는지 불명확) 을 피하기 위해 **Table** 명에 alias를 사용 (테이블 이름으로 직접 지칭도 가능)
- ▶ 적절한 **Join** 조건을 **WHERE** 절에 부여 (일반적으로 테이블 개수 -1 개의 조인 조건이 필요)
- ▶ 일반적으로 **PK**와 **FK**간의 = 조건이 붙는 경우가 많음

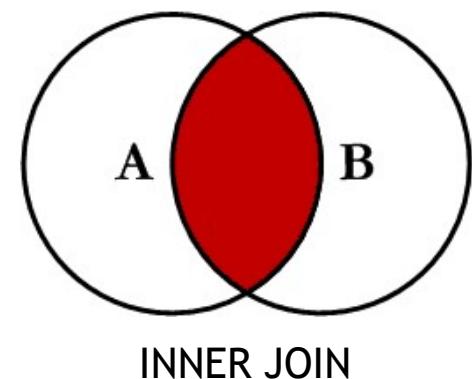
# Join의 처리

## : Flow



# Join의 종류

- ▶ 용어
  - ▶ Cross Join (Cartesian Product) : 모든 가능한 쌍이 나타남
  - ▶ Inner Join : Join 조건을 만족하는 튜플만 나타남
    - ▶ Theta Join : 조건(Theta)에 의한 조인
    - ▶ Equi-Join : Theta Join & 조건 0| Equal (=)
    - ▶ Natural Join : Equi-Join & 동일한 컬럼명 합쳐짐
  - ▶ Outer Join : 조건을 만족하지 않는 튜플(짝이 없는 튜플)도 null과 함께 나타남
  - ▶ Self Join : 자기 자신과 조인



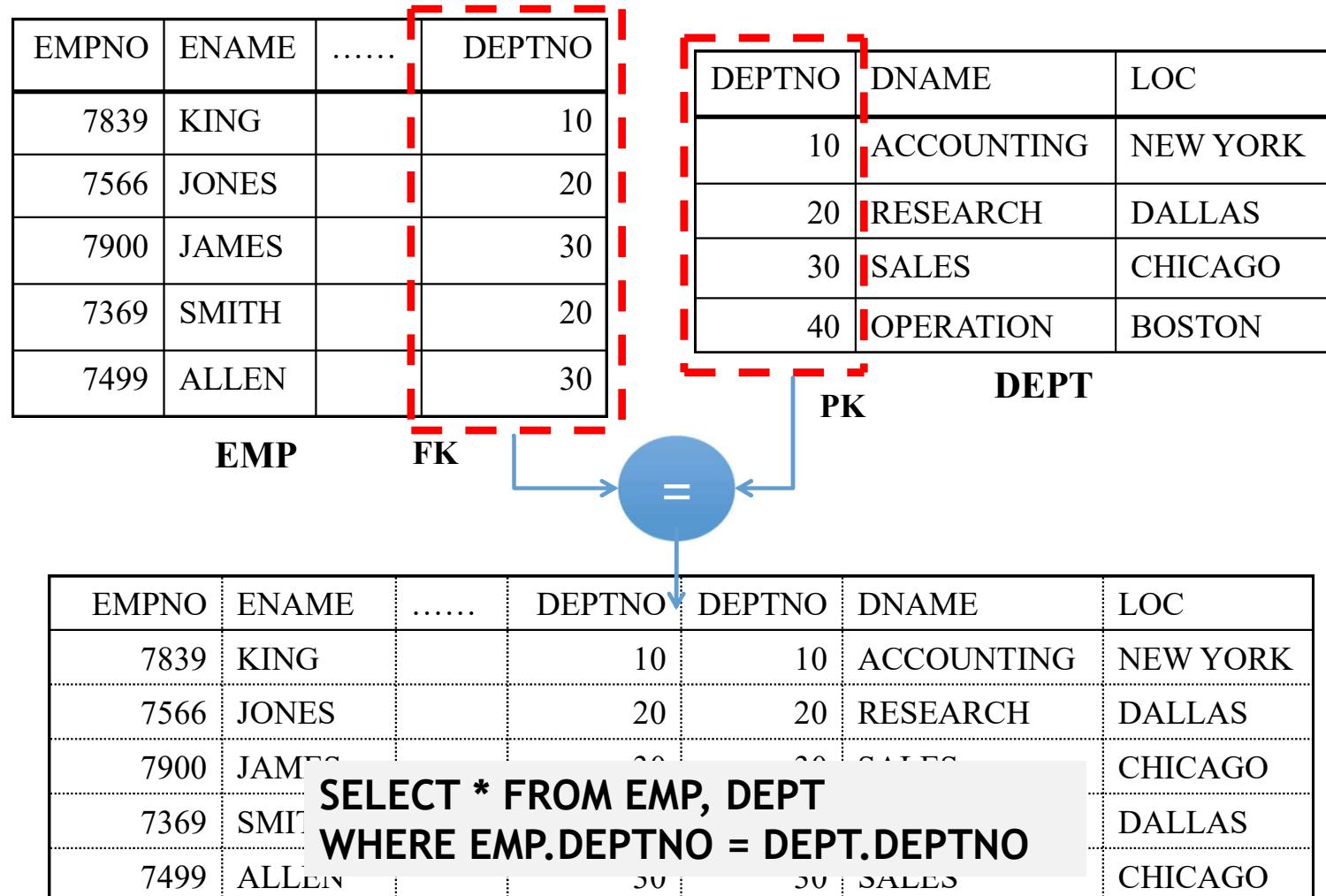
# SQL:1999 Syntax (Oracle 9i)

- ▶ FROM 절에서 바로 Join을 명시적으로 정의

```
SELECT table1.column, table2.column
  FROM table1
    [CROSS JOIN table2] |
    [NATURAL JOIN table2] |
    [JOIN table2 USING (column_name)] |
    [JOIN table2
      ON(table1.column_name = table2.column_name)] |
    [LEFT|RIGHT|FULL OUTER JOIN table2
      ON (table1.column_name = table2.column_name)];
```

- ▶ 예
  - ▶ SELECT \* FROM emp **JOIN** dept **USING** (deptno);
  - ▶ SELECT \* FROM emp **JOIN** dept **ON** emp.deptno = dept.deptno
  - ▶ SELECT \* FROM emp **RIGHT OUTER JOIN** dept **ON** (emp.deptno = dept.deptno)

# Equi-Join



# Equi-Join

- ▶ [연습] hr.employees and hr.departments
  - ▶ employees와 departments를 department\_id를 기준으로 Join 하여 first\_name, department\_id, department\_name을 출력해 봅시다

```
SELECT first_name, em.department_id,
       de.department_name
  FROM employees em, departments de
 WHERE em.department_id = de.department_id;
```

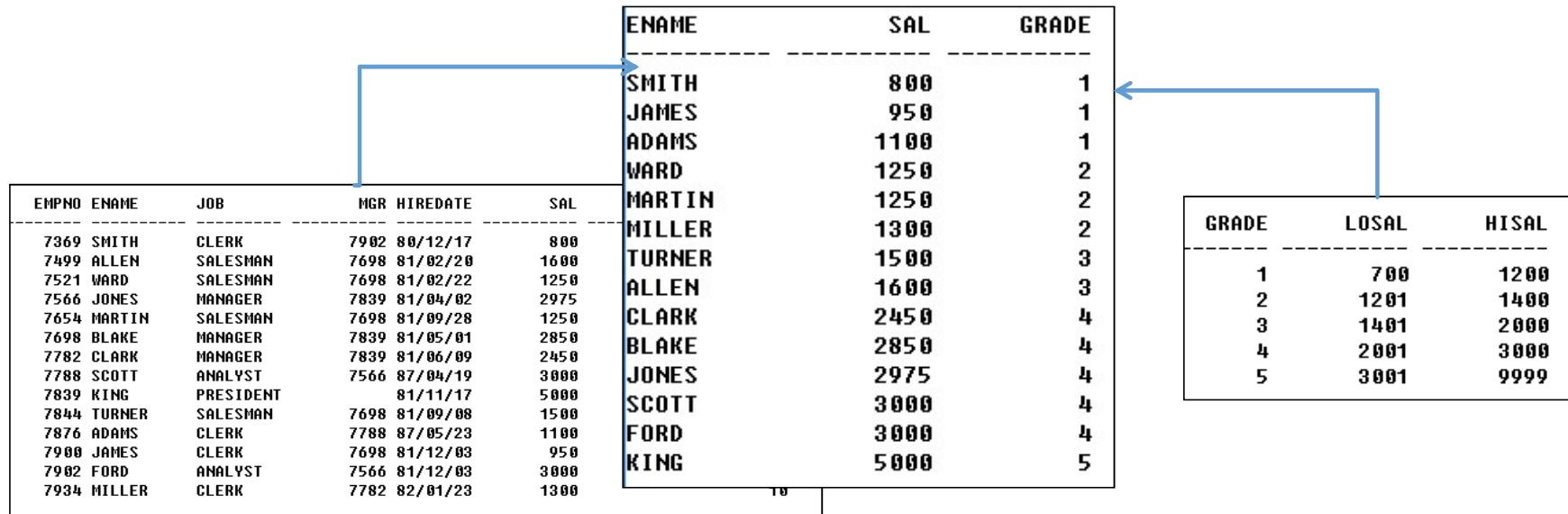
- ▶ 총 몇 건의 ROW가 검색되는지 확인해 봅시다
  - ▶ null은 조인되지 않음을 확인합니다.
  - ▶ 부서를 배정받지 못한 사원(department\_id 가 NULL) 은 누구인지 확인해 봅시다

# Theta Join

## ▶ 정의

- ▶ 임의의 조건을 Join 조건으로 사용
- ▶ Non-Equi Join이라고도 함
- ▶ Equal(=) 이외의 연산자를 사용하여 Join Condition을 작성한 경우를 일컬음

```
SELECT e.ename, e.sal, s.grade  
FROM emp e, salgrade s  
WHERE e.sal BETWEEN s.losal AND s.hisal
```



# Outer Join

- ▶

## 정의

- ▶ Join 조건을 만족하지 않는(짝이 없는) 튜플의 경우 Null 을 포함하여 결과를 생성
- ▶ 모든 행이 결과 테이블에 참여

- ▶

## 종류

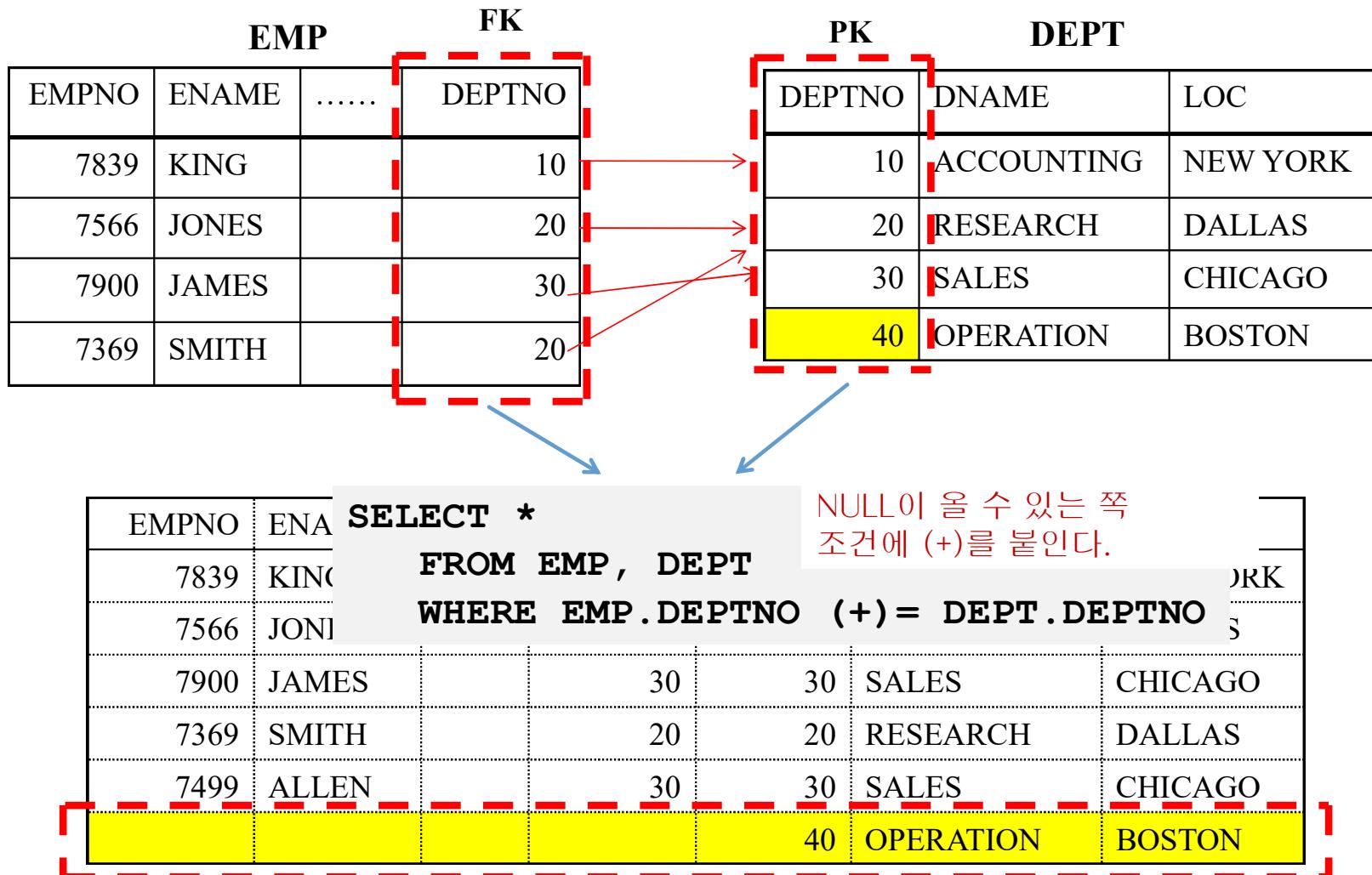
- ▶ Left Outer Join : 왼쪽의 모든 튜플은 결과 테이블에 나타남
- ▶ Right Outer Join : 오른쪽의 모든 튜플은 결과 테이블에 나타남
- ▶ Full Outer Join : 양쪽 모두 결과 테이블에 참여

- ▶

## 표현 방법

- ▶ NULL이 올 수 있는 쪽 조건에 (+)를 붙인다
  - ▶ 어느 쪽에 붙이느냐에 따라 의미가 변하므로 올바른 위치에 붙여야 한다

# Outer Join



# Outer Join

## : Left Outer Join

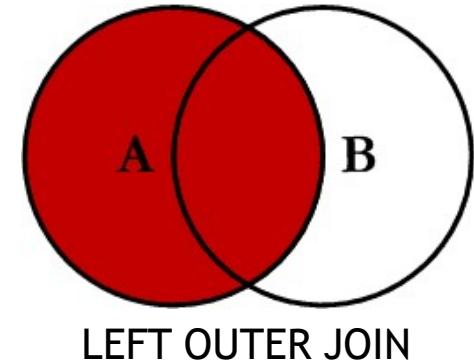
- ▶ 왼쪽 테이블의 모든 row를 결과 테이블에 나타냄

- ▶ ANSI SQL의 예

```
SELECT e.department_id, e.first_name, d.department_name
  FROM employees e LEFT OUTER JOIN departments d
    ON e.department_id = d.department_id ;
```

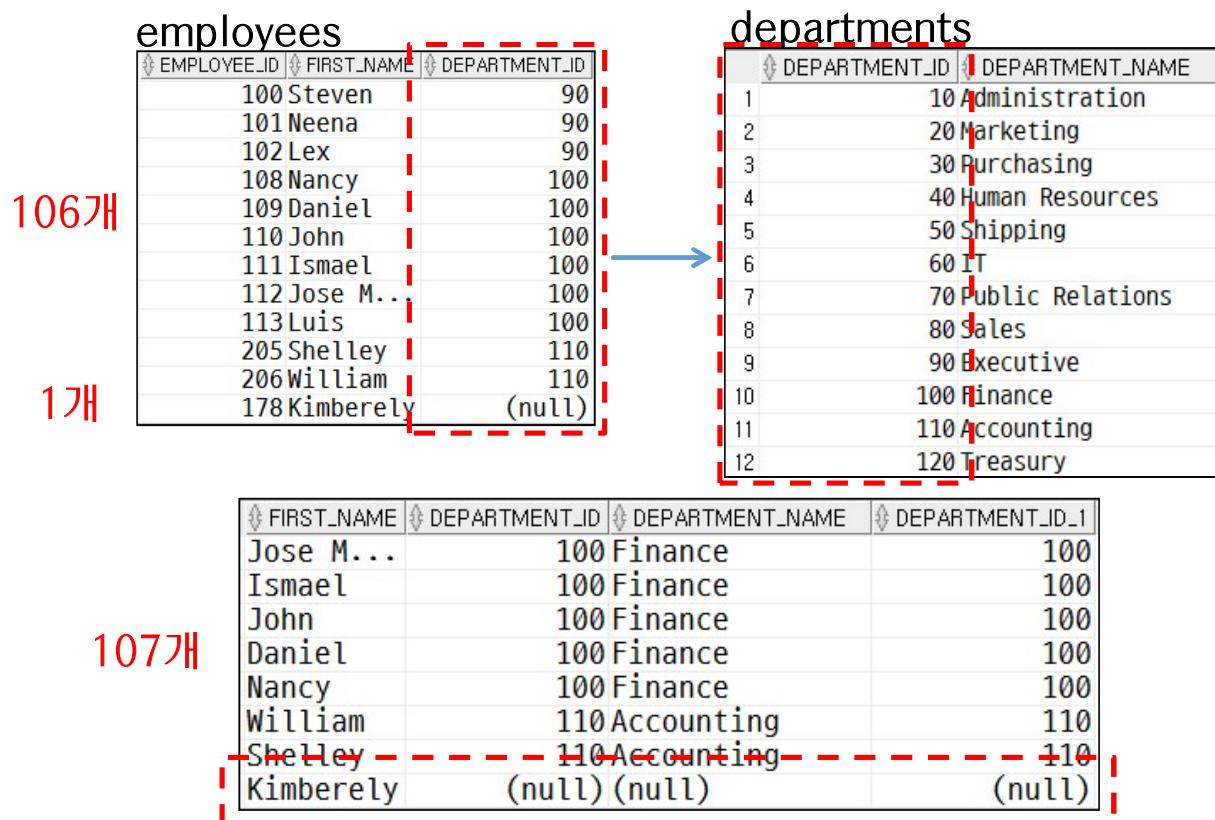
- ▶ Oracle SQL의 예

```
SELECT e.department_id, e.first_name, d.department_name
  FROM employees e, departments d
 WHERE e.department_id = d.department_id(+);
```



# Outer Join

## : Left Outer Join



# Outer Join

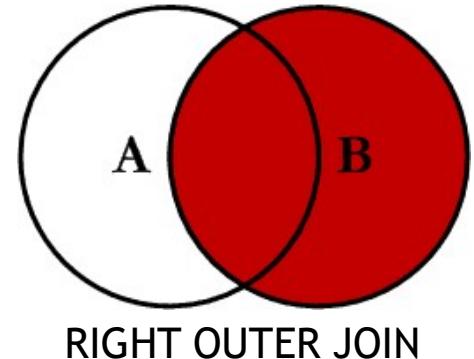
## : Right Outer Join

- ▶ 오른쪽 테이블의 모든 row를 결과 테이블에 나타냄
  - ▶ ANSI SQL의 예

```
SELECT e.department_id, e.first_name, d.department_name
  FROM employees e RIGHT OUTER JOIN departments d
    ON e.department_id = d.department_id ;
```

- ▶ Oracle SQL의 예

```
SELECT e.department_id, e.first_name, d.department_name
  FROM employees e, departments d
 WHERE e.department_id (+) = d.department_id ;
```



# Outer Join

## : Right Outer Join

The diagram illustrates a Right Outer Join between two tables: 'departments' and 'employees'. A blue arrow points from the 'departments' table to the 'employees' table, indicating the direction of the join.

**departments** (Left Table):

| DEPARTMENT_ID | DEPARTMENT_NAME  |
|---------------|------------------|
| 1             | Administration   |
| 2             | Marketing        |
| 3             | Purchasing       |
| 4             | Human Resources  |
| 5             | Shipping         |
| 6             | IT               |
| 7             | Public Relations |
| 8             | Sales            |
| 9             | Executive        |
| 10            | Finance          |
| 11            | Accounting       |
| 12            | Treasury         |

**employees** (Right Table):

| EMPLOYEE_ID | FIRST_NAME | DEPARTMENT_ID |
|-------------|------------|---------------|
| 100         | Steven     | 90            |
| 101         | Neena      | 90            |
| 102         | Lex        | 90            |
| 108         | Nancy      | 100           |
| 109         | Daniel     | 100           |
| 110         | John       | 100           |
| 111         | Ismael     | 100           |
| 112         | Jose M...  | 100           |
| 113         | Luis       | 100           |
| 205         | Shelley    | 110           |
| 206         | William    | 110           |
| 178         | Kimberely  | (null)        |

**Result Table:**

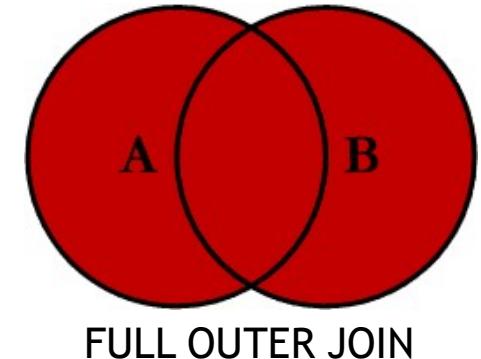
|     |               |                   |
|-----|---------------|-------------------|
| 103 | 100 Ismael    | Finance           |
| 104 | 100 Nancy     | Finance           |
| 105 | 110 William   | Accounting        |
| 106 | 110 Shelley   | Accounting        |
| 107 | (null) (null) | Treasury          |
| 108 | (null) (null) | Corporate Tax     |
| 109 | (null) (null) | Control And Cr... |
| 110 | (null) (null) | Shareholder Se... |
| 111 | (null) (null) | Benefits          |
| 112 | (null) (null) | Manufacturing     |
| 113 | (null) (null) | Construction      |
| 114 | (null) (null) | Contracting       |
| 115 | (null) (null) | Operations        |
| 116 | (null) (null) | IT Support        |
| 117 | (null) (null) | NOC               |
| 118 | (null) (null) | IT Helpdesk       |
| 119 | (null) (null) | Government Sales  |
| 120 | (null) (null) | Retail Sales      |
| 121 | (null) (null) | Recruiting        |
| 122 | (null) (null) | Payroll           |

**Count of Rows:**

- departments:** 11개 (사용o)
- employees:** 106개
- Result Table:** 122개
- employees (excluding nulls):** 16개
- Total Non-null Employees:** 106개 + 16개 = 122개

# Outer Join

## : Full Outer Join



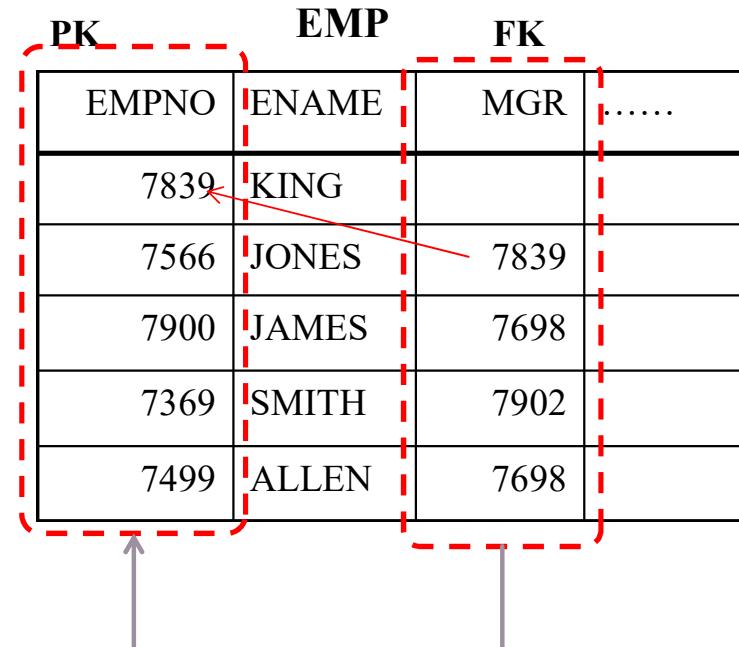
```
SELECT e.department_id, e.first_name, d.department_name  
FROM employees e FULL OUTER JOIN departments d  
ON e.department_id = d.department_id ;
```

| 76 | 80 Alyssa        | Sales    | 105 | 700 |
|----|------------------|----------|-----|-----|
| 77 | 80 Jonathon      | Sales    | 106 | 701 |
| 78 | 80 Jack          | Sales    | 107 | 702 |
| 79 | (null) Kimberely | (null)   | 108 | 703 |
| 80 | 80 Charles       | Sales    | 109 | 704 |
| 81 | 50 Winston       | Shipping | 110 | 705 |
| 82 | 50 Jean          | Shipping | 111 | 706 |
| 83 | 50 Martha        | Shipping | 112 | 707 |
| 84 | 50 Girard        | Shipping | 113 | 708 |
| 85 | 50 Nandita       | Shipping | 114 | 709 |
| 86 | 50 Alexis        | Shipping | 115 | 710 |
| 87 | 50 Julia         | Shipping | 116 | 711 |
|    |                  |          | 117 | 712 |
|    |                  |          | 118 | 713 |
|    |                  |          | 119 | 714 |
|    |                  |          | 120 | 715 |
|    |                  |          | 121 | 716 |
|    |                  |          | 122 | 717 |
|    |                  |          | 123 | 718 |

# Self Join

- ▶ 정의
  - ▶ 자기 자신과 Join
  - ▶ 동일한 테이블 명이 2번 이상 사용되므로 Alias를 사용할 수밖에 없음

```
SELECT *
  FROM EMP E1, EMP E2
 WHERE E1.EMPNO = E2.MGR
```



| EMPNO | ENAME | MGR  | ..... | EMPNO | ENAME |
|-------|-------|------|-------|-------|-------|
| 7566  | JONES | 7839 |       | 7839  | KING  |
| 7900  | JAMES | 7698 |       | 7698  | BLAKE |
| 7369  | SMITH | 7902 |       | 7902  | FORD  |
| 7499  | ALLEN | 7698 |       | 7698  | BLAKE |

# Self Join

## [연습] hr.employees

- ▶ SELF JOIN을 이용하여 다음의 값을 출력하시오

- ▶ EMPLOYEE\_ID
- ▶ FIRST\_NAME
- ▶ MANAGER의 EMPLOYEE ID
- ▶ MANAGER의 FIRST\_NAME

| employees emp |            |            | employees man |            |            |
|---------------|------------|------------|---------------|------------|------------|
| EMPLOYEE_ID   | FIRST_NAME | MANAGER_ID | EMPLOYEE_ID   | FIRST_NAME | MANAGER_ID |
| 102 Lex       |            | 100        | 102 Lex       |            | 100        |
| 103 Alexander |            | 102        | 103 Alexander |            | 102        |
| 104 Bruce     |            | 103        | 104 Bruce     |            | 103        |
| 105 David     |            | 103        | 105 David     |            | 103        |
| 106 Valli     |            | 103        | 106 Valli     |            | 103        |
| 107 Diana     |            |            | 107 Diana     |            |            |
| 108 Nancy     |            |            | 102 Lex       | Steven     | 101        |
| 109 Daniel    |            |            | 103 Alexander | Lex        | 108        |
| 110 John      |            |            | 104 Bruce     | Alexander  | 108        |
| 111 Ismael    |            |            | 105 David     | Alexander  | 108        |
|               |            |            | 106 Valli     | Alexander  |            |
|               |            |            | 107 Diana     | Alexander  |            |

```
SELECT emp.employee_id, emp.first_name,
       emp.manager_id, man.first_name manager
  FROM employees emp, employees man
 WHERE emp.manager_id = man.employee_id
```

# Oracle SQL

Group & Aggregation

# Aggregation Function (집계함수)

- ▶ 여러 행으로부터 하나의 결과값을 반환
- ▶ 종류
  - ▶ AVG
  - ▶ COUNT
    - ▶ COUNT(\*) : 테이블 내의 행 수 (NULL도 카운트됨)
    - ▶ COUNT(expr) : 테이블 내의 행 수 (NULL 제외)
  - ▶ MAX
  - ▶ MIN
  - ▶ SUM
  - ▶ STDDEV
  - ▶ VARIANCE

# Aggregation Function

```
SELECT sal FROM emp;
```

| SAL  |
|------|
| 800  |
| 1600 |
| 1250 |
| 2975 |
| 1250 |
| 2850 |
| 2450 |
| 3000 |
| 5000 |
| 1500 |
| 1100 |
| 950  |
| 3000 |
| 1300 |

```
SELECT AVG(sal) FROM emp;
```

AVG(SAL)  
-----  
2073.21429

# Aggregation Function

## ▶ count()

- ▶ 함수에 입력되는 데이터의 총 건수를 구하는 함수
- ▶ \* 를 사용하면 null을 포함한 총 Row의 개수를 구하며, 필드를 명시할 경우 null 값을 제외한다

```
SELECT COUNT(*), COUNT(commission_pct)
FROM employees;
```

null 포함

null 제외

```
SELECT COUNT(*)
FROM employees
WHERE salary > 16000;
```

# Aggregation Function

- ▶ `sum()`
  - ▶ 입력된 데이터들의 합계 값을 구하는 함수

```
SELECT COUNT(*), SUM(salary)  
FROM employees;
```

# Aggregation Function

## ▶ avg()

- ▶ 입력된 데이터들의 평균 값을 구하는 함수
- ▶ 주의: null 값이 있는 경우 빼고 계산해야 함 = nvl 함수와의 조합

```
SELECT COUNT(*), SUM(salary), AVG(salary)
FROM employees;
```

| name | point    |
|------|----------|
| 홍길동  | 70       |
| 일지매  | null → 0 |
| 유관순  | 50       |

- $120 / 3 = 40$
- $120 / 2 = 60 \checkmark$

```
SELECT COUNT(*), SUM(salary), AVG(VNL(salary,0))
FROM employees;
```

- ▶ NULL 값을 포함시킬 것인지, 뺄 것인지에 따라 통계 결과가 달라진다  
어떤 값을 대상으로 통계 값을 잡을 것인지는 정책으로 결정

# Aggregation Function

- ▶ `min()` / `max()`
  - ▶ 입력된 값 중 가장 작은 값/큰 값을 구하는 함수
  - ▶ 여러 건의 데이터를 순서대로 정렬 후 값을 구하기 때문에 데이터가 많을 때는 느리다 (사용에 유의)

```
SELECT COUNT(*), MAX(salary), MIN(salary)
FROM employees;
```

# 일반적인 오류

- ▶ 부서의 평균 연봉을 구하고자 다음과 같은 Query를 실행

```
SELECT deptno, AVG(sal) FROM emp;
```



- ▶ 주의
  - ▶ 집계함수의 결과는 하나의 ROW
  - ▶ deptno는 하나의 ROW에 표현할 수 없음
  - ▶ 부서별과 같은 내용이 필요할 때는 GROUP BY 절 사용

# GROUP BY

```
SELECT deptno, sal  
FROM emp  
ORDER BY deptno;
```

| DEPTNO | SAL  |
|--------|------|
| 10     | 2450 |
|        | 5000 |
|        | 1300 |
| 20     | 2975 |
|        | 3000 |
|        | 1100 |
| 20     | 800  |
|        | 3000 |
|        | 1250 |
| 30     | 1500 |
|        | 1600 |
|        | 950  |
|        | 2850 |
| 30     | 1250 |
|        |      |

```
SELECT deptno, AVG(sal)  
FROM emp  
GROUP BY deptno  
ORDER BY deptno;
```

| DEPTNO | AVG(SAL)   |
|--------|------------|
| 10     | 2916.66667 |
| 20     | 2175       |
| 30     | 1566.66667 |

# 일반적 오류

- ▶ 부서별 급여에 부서명도 함께 출력?

```
SELECT deptno, dname, AVG(sal)
      FROM emp
 GROUP BY deptno
 ORDER BY deptno;
```

- ▶ 비록 부서 번호에 따라 부서명은 하나로 결정될 수 있지만, dname은 groupin에 참여하지 않았으므로 하나의 row로 aggregate 되었다고 볼 수 없음
- ▶ 주의
  - ▶ SELECT의 Col 목록에는 Group by에 참여한 필드나 aggregate 함수만 올 수 있다
  - ▶ Group by 이후에는 Group by에 참여한 필드나 aggregate 함수만 남아있는 셈
    - ▶ HAVING, ORDER BY도 마찬가지

# HAVING 절

- ▶ Aggregation 결과에 대해 다시 condition을 검사할 때
- ▶ 일반적 오류
  - ▶ 평균 월급이 2000 이상인 부서는?

```
SELECT deptno, AVG(sal)
  FROM emp
 WHERE AVG(sal) > 2000
 GROUP BY deptno;
```



- ▶ 주의
  - ▶ WHERE 절은 Aggregation 이전, HAVING 절은 Aggregation 이후의 필터링
  - ▶ HAVING 절에는 GROUP BY에 참여한 컬럼이나 Aggregation 함수만 사용 가능

# 단일 SQL문의 실행

: Flow



# GROUP BY 절

- ▶ [예제] hr.employees
  - ▶ 급여(salary) 합계가 20000 이상인 부서의 부서 번호와 인원 수, 급여 합계를 출력하기 위해 다음과 같은 쿼리를 작성했다.

```
SELECT department_id, COUNT(*) , SUM(salary)
  FROM employees
 WHERE SUM(salary) > 20000
 GROUP BY department_id;
```

- ▶ 위 쿼리를 살펴보고 문제점이 무엇인지 생각해 봅시다

# GROUP BY 절

- ▶ [SOLUTION] hr.employees
  - ▶ GROUP 함수는 WHERE 절 이후에 처리되므로 SUM(salary)는 WHERE 절에 사용할 수 없음

```
SELECT department_id, COUNT(*) , SUM(salary)
  FROM employees
 WHERE SUM(salary) > 20000
 GROUP BY department_id
 HAVING SUM(salary) > 20000
```

- ▶ Having 절에는 그룹함수와 GROUP BY에 참여한 컬럼만 사용할 수 있음

# 단일 SQL 작성법

1. 최종 출력될 정보에 따라 원하는 컬럼을 **SELECT** 절에 추가
2. 원하는 정보를 가진 테이블들을 **FROM** 절에 추가
3. **WHERE** 절에 알맞은 **JOIN** 조건 추가
4. **WHERE** 절에 알맞은 검색 조건 추가
5. 필요에 따라 **GROUP BY**, **HAVING** 등을 통해 Grouping하고 Aggregate
6. 정렬 조건 **ORDER BY**에 추가

# ROLLUP

```
SELECT deptno, job, sum(sal)
  FROM emp
 GROUP BY ROLLUP(deptno, job)
```

GROUP BY 절과 함께 사용되며 그룹 지어진 결과에 대하여  
좀 더 상세한 정보를 변환하는 기능을 수행

ROLLUP(A, B) :  
GROUP BY (A, B) &  
GROUP BY (A) &  
ALL

*Regular rows*

*Superaggregate rows*

.....

.....

| DEPTNO | JOB       | SUM(SAL) |
|--------|-----------|----------|
| 10     | CLERK     | 1300     |
| 10     | MANAGER   | 2450     |
| 10     | PRESIDENT | 5000     |
| 10     |           | 8750     |
| 20     | ANALYST   | 6000     |
| 20     | CLERK     | 1900     |
| 20     | MANAGER   | 2975     |
| 20     |           | 10875    |
| 30     | CLERK     | 950      |
| 30     | MANAGER   | 2850     |
| 30     | SALESMAN  | 5600     |
| 30     |           | 9400     |
|        |           | 29025    |

# CUBE

```
SELECT deptno, job, sum(sal)
   FROM emp
 GROUP BY CUBE(deptno, job)
```

Cross-Tab에 대한 Summary를 추출하는데 사용  
: ROLLUP에 의해 출력되는 Item Total 값과  
Column Total 값을 나타낼 수 있음

CUBE(A, B) :  
GROUP BY (A, B) &  
GROUP BY (A) &  
GROUP BY (B) &  
ALL

*Regular rows*  
*Superaggregate rows*

| DEPTNO | JOB       | SUM(SAL) |
|--------|-----------|----------|
| 10     | CLERK     | 1300     |
| 10     | MANAGER   | 2450     |
| 10     | PRESIDENT | 5000     |
| 10     |           | 8750     |
| 20     | ANALYST   | 6000     |
| 20     | CLERK     | 1900     |
| 20     | MANAGER   | 2975     |
| 20     |           | 10875    |
| 30     | CLERK     | 950      |
| 30     | MANAGER   | 2850     |
| 30     | SALESMAN  | 5600     |
| 30     |           | 9400     |
|        | ANALYST   | 6000     |
|        | CLERK     | 4150     |
|        | MANAGER   | 8275     |
|        | PRESIDENT | 5000     |
|        | SALESMAN  | 5600     |
|        |           | 29025    |

# Oracle SQL

SUBQUERY, SET Operation

# Subquery

- ▶ 하나의 SQL 질의문 속에 다른 SQL 질의문이 포함되어 있는 형태
- ▶ 예) 'SCOTT' 보다 급여가 많은 사람은?
  - ▶ 급여가 많은 사람의 이름?
    - ▶ SELECT ename FROM emp WHERE sal > ???
  - ▶ 'SCOTT'의 급여는?
    - ▶ SELECT sal FROM emp WHERE ename='SCOTT'

```
SELECT ename
      FROM emp
 WHERE sal > ( SELECT sal
                  FROM emp
                 WHERE ename = 'SCOTT' )
```

# Subquery

- ▶ [연습] hr.employees
  - ▶ 'Den' 보다 급여를 많이 받는 사원의 이름과 급여는?

```
select salary      11000
      from employees
     where first_name='Den'
```

```
select first_name, salary
      from employees
     where salary > ?? 11000
```



```
select employee_id, first_name, salary
      from employees
     where salary > (select salary
                        from employees
                       where first_name='Den');
```

# Single-Row Subquery

- ▶ Subquery의 결과가 한 ROW인 경우
- ▶ Single-Row Operator를 사용해야 함: =, >, >=, <, <=, <>

```
SELECT ename, sal, deptno
  FROM emp
 WHERE ename = (SELECT MIN(ename) FROM emp);
```

```
SELECT ename, sal
  FROM emp
 WHERE sal < (SELECT AVG(sal) FROM emp);
```

```
SELECT ename, deptno
  FROM emp
 WHERE deptno = (SELECT deptno
                   FROM dept
                  WHERE dname = 'SALES');
```

# Single-Row Subquery

- ▶ [연습] hr.employees
  - ▶ 급여를 가장 적게 받는 사람의 이름, 급여, 사원 번호를 출력하시오

```
SELECT first_name, salary, employee_id  
FROM employees  
WHERE salary = (SELECT MIN(salary)  
                 FROM employees);
```

- ▶ 평균 급여보다 적게 받는 사원의 이름, 급여를 출력해 보세요.

# Multi-Row Subquery

- ▶ Subquery의 결과가 둘 이상의 Row
- ▶ Multi-Row에 대한 연산을 사용해야 함: ANY, ALL, IN, EXIST ...

```
SELECT ename, sal, deptno
      FROM emp
 WHERE ename = (SELECT MIN(ename)
                  FROM emp GROUP BY deptno);
```



```
SELECT ename, sal, deptno
      FROM emp
 WHERE ename IN (SELECT MIN(ename)
                  FROM emp GROUP BY deptno);
```



```
SELECT ename, sal, deptno
      FROM emp
 WHERE ename = ANY (SELECT MIN(ename)
                  FROM emp GROUP BY deptno);
```



# Multi-Row Subquery

| 연산자       | 설명                                             |
|-----------|------------------------------------------------|
| IN        | 리턴되는 값 중에서 조건에 해당하는 값이 있으면 참                   |
| ANY, SOME | 서브쿼리에 의해 리턴되는 각각의 값과 조건을 비교하여 하나 이상을 만족하면 참    |
| ALL       | 값을 서브쿼리에 의해 리턴되는 모든 값을 비교하여 모두 만족해야 참          |
| EXISTS    | 메인 쿼리의 비교 조건이 서브쿼리의 결과 중에서 만족하는 값이 하나라도 존재하면 참 |

- ANY는 OR과 비슷
- ALL은 AND와 비슷

# Multi-Row Subquery

- ▶ [연습] hr.employees

- ▶ IN

```
select first_name, salary  
from employees  
  
where salary IN (select salary  
                  from employees  
                  where department_id = 110);
```

|   | FIRST_NAME | SALARY |
|---|------------|--------|
| 1 | Shelley    | 12008  |
| 2 | Nancy      | 12008  |
| 3 | William    | 8300   |

```
where salary = 12008  
or salary = 8300
```

| SALARY |
|--------|
| 12008  |
| 8300   |

# Multi-Row Subquery

- ▶ [연습] hr.employees
  - ▶ ALL (AND)

```
select first_name, salary  
from employees  
  
where salary > ALL (select salary  
                      from employees  
                     where department_id = 110);
```

```
where salary > 12008  
and salary > 8300
```

| SALARY |
|--------|
| 12008  |
| 8300   |

|   | FIRST_NAME | SALARY |
|---|------------|--------|
| 1 | Michael    | 13000  |
| 2 | Karen      | 13500  |
| 3 | John       | 14000  |
| 4 | Lex        | 17000  |
| 5 | Neena      | 17000  |
| 6 | Steven     | 24000  |

# Multi-Row Subquery

- ▶ [연습] hr.employees

- ▶ ANY (OR)

```
select first_name, salary  
from employees
```

```
where salary > ANY (select salary  
                     from employees  
                     where department_id = 110);
```

```
where salary > 12008  
or salary > 8300
```

| SALARY |
|--------|
| 12008  |
| 8300   |

|    | FIRST_NAME | SALARY |
|----|------------|--------|
| 1  | Steven     | 24000  |
| 2  | Neena      | 17000  |
| 3  | Lex        | 17000  |
| 4  | John       | 14000  |
| 5  | Karen      | 13500  |
| 6  | Michael    | 13000  |
| 7  | Nancy      | 12008  |
| 8  | Shelley    | 12008  |
| 9  | Alberto    | 12000  |
| 10 | Lisa       | 11500  |
| 11 | Den        | 11000  |
| 12 | Gerald     | 11000  |
| 21 | Danielle   | 9500   |
| 22 | David      | 9500   |
| 23 | Patrick    | 9500   |
| 24 | Peter      | 9000   |
| 25 | Alexander  | 9000   |
| 26 | Allan      | 9000   |
| 27 | Daniel     | 9000   |
| 28 | Alyssa     | 8800   |
| 29 | Jonathon   | 8600   |
| 30 | Jack       | 8400   |

# Correlated Query

- ▶ Outer Query와 Inner Query가 서로 연관되어 있음
- ▶ 해석 방법
  - ▶ Outer query의 한 Row를 얻는다
  - ▶ 해당 Row를 가지고 Inner Query를 수행한다
  - ▶ 수행 결과를 이용, Outer query의 WHERE 절을 evaluate
  - ▶ 결과가 참이면 해당 Row를 결과에 포함시킨다

```
SELECT ename, sal, deptno
      FROM emp outer
 WHERE sal > (SELECT AVG(sal)
                  FROM emp
                 WHERE deptno = outer.deptno);
```

# Subquery

: 예제

- ▶ 각 부서별로 최고급여를 받는 사원을 출력하시요

```
SELECT deptno, empno, ename, sal
      FROM emp
 WHERE (deptno,sal) IN (SELECT deptno, max(sal)
                           FROM emp
                           GROUP BY deptno);
```

```
SELECT e.deptno, e.empno, e.ename, e.sal
      FROM emp e, (SELECT s.deptno, max(s.sal) msal
                    FROM emp s GROUP BY deptno) m
 WHERE e.deptno = m.deptno AND e.sal = m.msal;
```

```
SELECT deptno, empno, ename, sal
      FROM emp e
 WHERE e.sal = (SELECT max(sal)
                  FROM emp
                  WHERE deptno = e.deptno);
```

# Subquery

: 예제

- ▶ [예제] hr.employees

- ▶ 각 부서별로 최고급여를 받는 사원을 출력하세요 – (조건절에서 비교)

```
SELECT department_id, employee_id, first_name, salary
FROM employees
WHERE (department_id, salary) in (SELECT department_id, MAX(salary)
                                    FROM employees
                                    GROUP BY department_id)
```

employees 테이블

| DEPARTMENT_ID | EMPLOYEE_ID | FIRST_NAME | SALARY |
|---------------|-------------|------------|--------|
| 1             | 10          | Jennifer   | 4400   |
| 2             | 20          | Michael    | 13000  |
| 3             | 20          | Pat        | 6000   |
| 4             | 30          | Den        | 11000  |
| 5             | 30          | Alexander  | 3100   |
| 9             | 30          | Karen      | 2500   |
| 10            | 40          | Susan      | 6500   |
| 11            | 50          | Matthew    | 8000   |
| 12            | 50          | Adam       | 8200   |
| 13            | 50          | Davam      | 7000   |

| DEPARTMENT_ID | MAX(SALARY) |
|---------------|-------------|
| 10            | 4400        |
| 20            | 13000       |
| 30            | 11000       |
| 40            | 6500        |
| 50            | 8200        |
| 60            | 9000        |
| 70            | 10000       |
| 80            | 14000       |
| 90            | 24000       |

# Subquery

: 예제

## ▶ [예제] hr.employees

- ▶ 각 부서별로 최고급여를 받는 사원을 출력하세요 – (테이블에서 조인)

```
SELECT e.department_id, e.employee_id, e.first_name, e.salary  
FROM employees e, (SELECT department_id, max(salary) salary  
                      FROM employees  
                     GROUP BY department_id) s  
  
WHERE e.department_id = s.department_id  
      AND e.salary = s.salary;
```

| employees.e |               |             |            | salary.s |               |        |
|-------------|---------------|-------------|------------|----------|---------------|--------|
|             | DEPARTMENT_ID | EMPLOYEE_ID | FIRST_NAME | SALARY   | DEPARTMENT_ID | SALARY |
| 1           | 10            | 200         | Jennifer   | 4400     | 10            | 4400   |
| 2           | 20            | 201         | Michael    | 13000    | 20            | 13000  |
| 3           | 20            | 202         | Pat        | 6000     | 30            | 11000  |
| 4           | 30            | 114         | Den        | 11000    | 40            | 6500   |
| 5           | 30            | 115         | Alexander  | 3100     | 50            | 8200   |
| 6           | 30            | 116         | Shelli     | 2900     | 60            | 9000   |
| 7           | 30            | 117         | Sigal      | 2800     | 70            | 10000  |
| 8           | 30            | 118         | Guy        | 2600     | 80            | 14000  |
| 9           | 30            | 119         | Karen      | 2500     | 90            | 24000  |
| 10          | 40            | 203         | Susan      | 6500     | 100           | 12008  |
| 11          | 50            | 120         | Matthew    | 8000     | 110           | 12008  |
| 12          | 50            | 121         | Adam       | 8200     | (null)        | 7000   |

# Top-K Query (Oracle)

- ▶ ROWNUM : 질의의 결과에 가상으로 부여되는 Oracle의 Pseudo Column
- ▶ Top-K Query: 조건을 만족하는 상위 k개의 결과를 빨리 얻기
  - ▶ 81년도에 입사한 사람 중 급여가 가장 많은 3명은 누구인가?

```
SELECT rownum, ename, sal
      FROM emp
 WHERE hiredate like '81%' AND rownum < 4
 ORDER BY sal DESC;
```

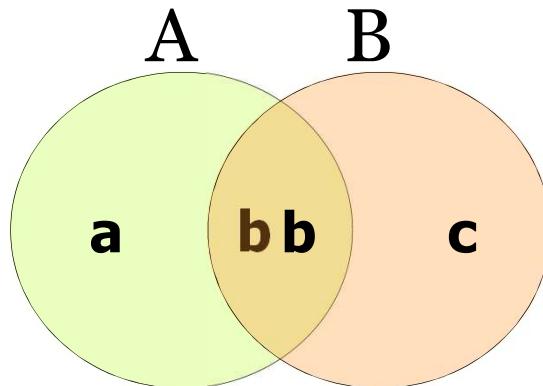


```
SELECT rownum, ename, sal
      FROM (SELECT *
              FROM emp
             WHERE hiredate like '81%'
           ORDER BY sal DESC)
 WHERE rownum < 4;
```



# 집합(SET) Operator

- ▶ 두 집합의 결과를 가지고 집합 연산을 수행
- ▶ UNION, UNION ALL, INTERSECT, MINUS



- A UNION B = {a, b, c}
- A UNION ALL B = {a, b, b, c}
- A INTERSECT B = {b}
- A MINUS B = {a}

```
SELECT ename FROM emp
UNION
SELECT dname FROM dept;
```

# RANK 관련 함수

```
SELECT sal, ename,  
       RANK() OVER (ORDER BY sal DESC) AS rank,  
       DENSE_RANK() OVER (ORDER BY sal DESC) AS dense_rank,  
       ROW_NUMBER() OVER (ORDER BY sal DESC) AS row_number,  
       rownum AS "rownum"  
FROM emp;
```



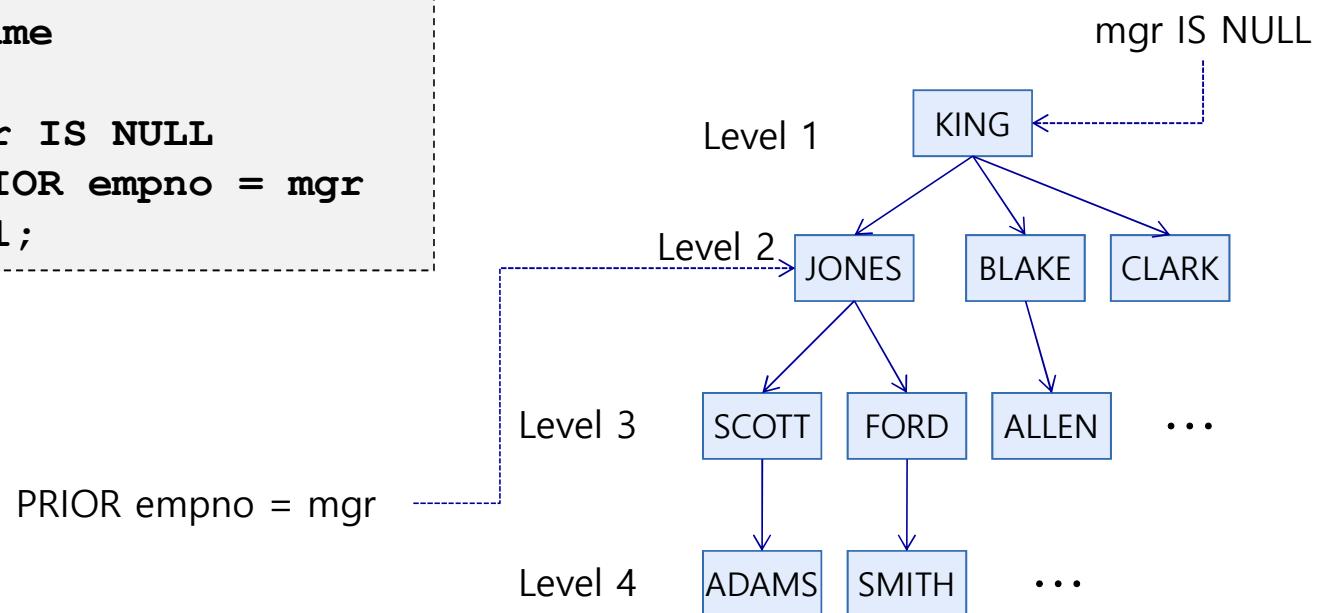
| SAL ENAME   | RANK | DENSE_RANK | ROW_NUMBER | rownum |
|-------------|------|------------|------------|--------|
| 5000 KING   | 1    | 1          | 1          | 9      |
| 3000 FORD   | 2    | 2          | 2          | 13     |
| 3000 SCOTT  | 2    | 2          | 3          | 8      |
| 2975 JONES  | 4    | 3          | 4          | 4      |
| 2850 BLAKE  | 5    | 4          | 5          | 6      |
| 2450 CLARK  | 6    | 5          | 6          | 7      |
| 1600 ALLEN  | 7    | 6          | 7          | 2      |
| 1500 TURNER | 8    | 7          | 8          | 10     |
| 1300 MILLER | 9    | 8          | 9          | 14     |
| 1250 WARD   | 10   | 9          | 10         | 3      |
|             |      |            |            | 11 5   |
|             |      |            |            | 12 11  |
|             |      |            |            | 13 12  |
|             |      |            |            | 14 1   |

- RANK : 중복순위 다음은 해당 개수만큼 건너뛰고 다음 순위 반환
- DENSE\_RANK : 중복순위와 상관 없이 다음 순위를 반환
- ROW\_NUMBER : 중복을 같은 순위로 두지 않고 무조건 순서대로 반환

# Hierarchical Query (Oracle)

- ▶ 트리 형태 구조를 추출하기 위한 질의
- ▶ START WITH (ROOT 조건), CONNECT BY PRIOR (연결조건)
- ▶ LEVEL : 트리의 레벨을 나타내는 Pseudo Column

```
SELECT level, ename
  FROM emp
 START WITH mgr IS NULL
 CONNECT BY PRIOR empno = mgr
 ORDER BY level;
```



# Oracle Database

DCL - Data Control Language

# Database User

- ▶ 데이터베이스 사용자
  - ▶ 데이터베이스에 접속하여 데이터를 이용하기 위해 접근하는 모든 사람
  - ▶ 이용 목적에 따라 데이터베이스 관리자(Database Administrator: DBA), 최종 사용자, 응용프로그래머로 구분
- ▶ Oracle 데이터베이스 생성시 기본적으로 생성되는 계정
  - ▶ SYS
    - ▶ Data Dictionary Table의 소유자
  - ▶ SYSTEM
    - ▶ 각종 Dictionary View의 소유자
- ▶ 각 USER는 동일한 이름의 SCHEMA 소유자로, 해당 SCHEMA의 모든 객체들에 접근할 수 있다

# 사용자 관리

- ▶ Syntax
  - ▶ 사용자 생성 : CREATE USER username IDENTIFIED BY password;
  - ▶ 비밀번호 변경 : ALTER USER username IDENTIFIED BY password;
  - ▶ 사용자 삭제 : DROP USER username [CASCADE];
- ▶ 주의사항
  - ▶ 일반적으로 DBA의 일
  - ▶ 사용자를 생성하려면 CREATE USER 권한 필요
  - ▶ 생성된 사용자가 Login 하려면 CREATE SESSION 권한 필요
  - ▶ 일반적으로 CONNECT, RESOURCE의 ROLE을 부여하면 일반사용자 역할을 수행할 수 있음

# 사용자 관리

- ▶ [연습] bituser 사용자를 생성하고 비밀번호를 변경해 봅니다

```
SQL> conn system/manager
Connected.
SQL> CREATE USER bituser IDENTIFIED BY bituser;
User created.
SQL> conn bituser/bituser
ERROR:
ORA-01045: user BITUSER lacks CREATE SESSION privilege;
logon denied
```

- ▶ USER를 생성해도 권한(Privilege)을 주지 않으면 아무 일도 할 수 없음
  - ▶ 위 예에서는 사용자가 CREATE SESSION 권한을 갖고있지 않아 오류 발생
- ▶ 사용자는 DBA가 초기화한 암호를 갖게 되며, 로그인 후 ALTER USER 명령을 이용, 각자의 암호를 재설정

# 사용자 정보 확인

- ▶ 관련 DICTIONARY
  - ▶ USER\_USERS : 현재 사용자 관련 정보
  - ▶ ALL\_USERS : DB의 모든 사용자 정보
  - ▶ DBA\_USERS : DB의 모든 사용자 상세 정보 (DBA만 사용 가능)

```
SELECT * FROM USER_USERS;
```

- ▶ [연습] USER\_USERS, ALL\_USERS, DBA\_USERS Dictionary로부터 user 목록을 각각 출력해 봅니다

```
SQL> conn scott/tiger
SQL> SELECT username FROM user_users;
SQL> SELECT username FROM all_users;
SQL> SELECT username FROM dba_users;
(ERROR 발생, DBA 권한을 갖고 있지 않음)
SQL> conn system/manager
SQL> /
(SUCCESS, system 계정은 DBA 권한을 갖고 있음)
```

# 권한(Privilege)과 롤(Role)

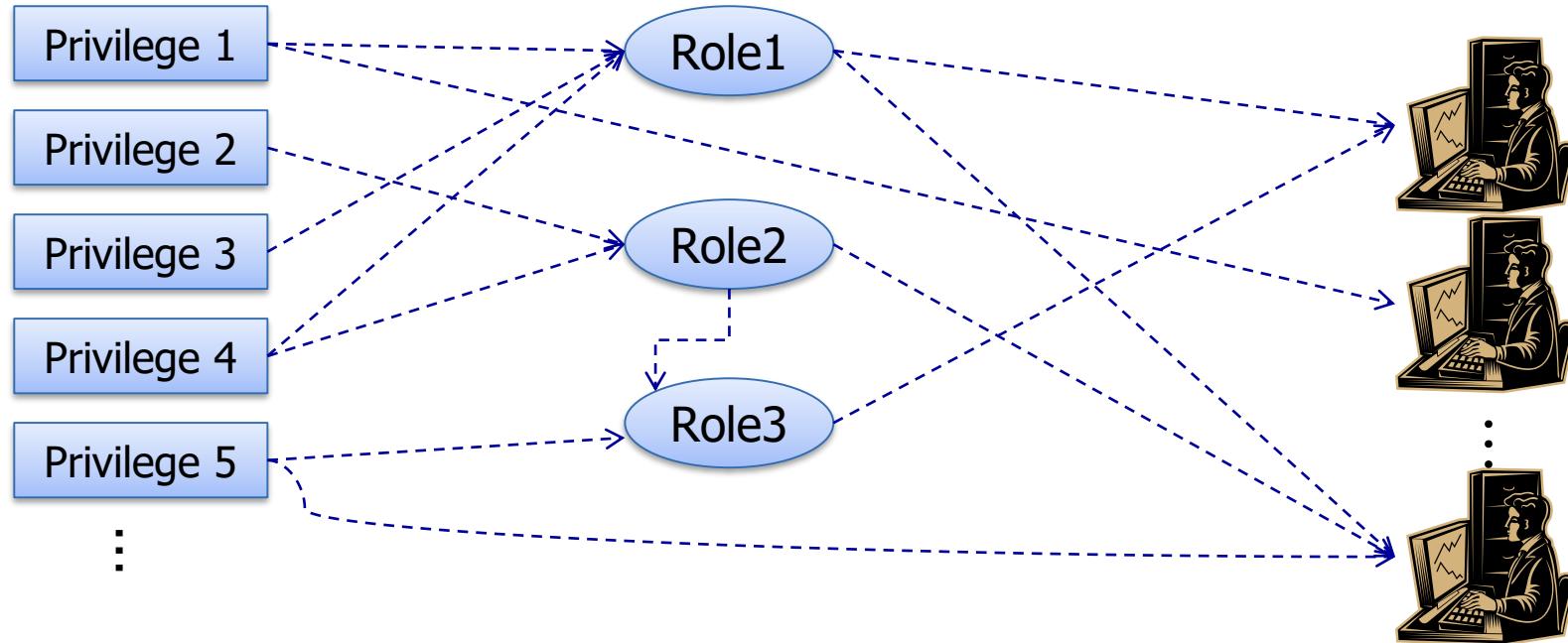
## : Privileges

### ▶ 권한(Privilege)

- ▶ 사용자가 특정 SQL 문을 실행하거나 특정 정보에 접근할 수 있는 권리
- ▶ 사용자는 작업에 요구되는 관련 권한에 대한 허가(GRANT)를 받아야 한다
- ▶ 종류
  - ▶ 시스템 권한(80여개) :
  - ▶ 스키마 객체 권한

# 권한(Privilege)과 롤(Role)

- ▶ 롤(Role)
  - ▶ 권한을 쉽게 관리하기 위하여 특정한 종류별로 묶어놓은 그룹



# GRANT / REVOKE

- ▶ 권한/롤을 부여(GRANT)하거나 회수(REVOKE)
- ▶ Syntax (System Privileges)

```
GRANT [system_priv|role [,system_priv|role ...]  
      TO {user [,user ...]|role|PUBLIC}  
      [WITH ADMIN OPTION];
```

```
REVOKE [system_priv|role [,system_priv|role ...]  
       FROM {user [,user ...]|role|PUBLIC};
```

- ▶ Syntax (Object Privileges)

```
GRANT {[object_priv [,object_priv ...]|ALL}  
       ON object TO {user [,user ...]|role|PUBLIC}  
       [WITH ADMIN OPTION];
```

```
REVOKE {[object_priv [,object_priv ...]|ALL}  
        FROM {user [,user ...]|role|PUBLIC};
```

- ▶ GRANT, REVOKE 문장은 실행 즉시 효력을 발휘한다 (재접속 등 필요 없음)

# GRANT / REVOKE

- ▶ 시스템 권한: 관리자로 수행 (ADMIN OPTION/GANT ANY PRIVILEGES 권한)

```
GRANT create session TO user1;
```

```
REVOKE create session FROM user1;
```

- ▶ 스키마 객체 권한

```
GRANT select ON emp TO user1;
```

```
REVOKE select ON emp FROM user1;
```

- ▶ WITH GRANT OPTION

- ▶ 해당 권한을 받은 사용자가 다시 제3자에게 권한을 부여할 수 있도록 하는 옵션
- ▶ 권한을 REVOKE 하면 해당 사용자가 3자에게 부여한 권한들도 함께 회수됨

```
GRANT select ON emp TO user2  
WITH GRANT OPTION;
```

# GRANT / REVOKE

- ▶ [연습] CREATE SESSION 권한을 부여해 봅니다

```
SQL> conn system/manager
SQL> GRANT create session TO bituser;
SQL> conn bituser/bituser
SQL> CREATE TABLE test (a NUMBER);
(* ERROR 발생. WHY? )
```

- ▶ 사용자에게 CONNECT, RESOURCE ROLE을 부여하면 TABLE 생성 등을 할 수 있다

```
SQL> conn system/manager
SQL> GRANT connect, resource TO bituser;
SQL> conn bituser/bituser
SQL> CREATE TABLE test (a NUMBER);
Table created.
SQL> DESC test
```

# ROLE

- ▶ ROLE을 생성한 후 Role에 Privilege를 Grant하여 Role 관리
  - ▶ 주로 DBA가 하는 작업

```
CREATE ROLE reviewer;  
GRANT select any table TO reviewer;  
GRANT create session, resource TO reviewer;
```

- ▶ 특정 Role을 사용자에게 Grant / Revoke

```
GRANT reviewer TO user3;
```

# 권한의 확인

- ▶ 관련 Dictionary
  - ▶ ROLE\_SYS\_PRIVS: System privileges granted to roles
  - ▶ ROLE\_TAB\_PRIVS: Table privileges granted to roles
  - ▶ USER\_ROLE\_PRIVS: Roles accessible by the user
  - ▶ USER\_TAB\_PRIVS\_MADE: Object privileges granted on the user's object
  - ▶ USER\_TAB\_PRIVS\_REC'D: Object privileges granted to the user
  - ▶ USER\_COL\_PRIVS\_MADE: Object privileges granted on the columns of the user's object
  - ▶ USER\_COL\_PRIVS\_REC'D: Object privileges granted to the user on specific columns
  - ▶ USER\_SYS\_PRIVS: Lists system privileges granted to the user

```
SELECT * FROM USER_ROLE_PRIVS;
```

```
SELECT * FROM ROLE_SYS_PRIVS;
```

# 권한의 확인

- ▶ [연습]
  - ▶ 현재 사용자에게 부여된 Role을 조회해 봅시다

```
SQL> SELECT * FROM user_role_privs;
```

- ▶ 일반적으로 부여되는 CONNECT와 RESOURCE 안에 있는 세부적인 Privilege 들을 확인해 봅시다

```
SQL> DESC role_sys_privs;
SQL> SELECT privilege FROM role_sys_privs WHERE
role='RESOURCE';
SQL> SELECT privilege FROM role_sys_privs WHERE
role='CONNECT';
```

# Oracle Database

DDL - Data Definition Language

# DDL 요약

- ▶ **CREATE TABLE** : 테이블 생성
- ▶ **ALTER TABLE** : 테이블 관련 변경
- ▶ **DROP TABLE** : 테이블 삭제
- ▶ **RENAME** : 이름 변경
- ▶ **TRUNCATE** : 테이블의 모든 데이터 삭제
- ▶ **COMMENT** : 테이블에 설명 추가

# 테이블 생성

- ▶ CREATE TABLE 문 이용
- ▶ Syntax

```
CREATE TABLE [schema.]table_name  
  (column datatype [DEFAULT expr]  
   [column_constraints],  
   ....,  
   [table_constraints]);
```

- ▶ Oracle에서의 Table 관리
  - ▶ Oracle은 Database의 공간을 Tablespace라는 논리적 공간으로 분할하여 관리
  - ▶ Table을 만들기 위해서는 CREATE TABLE 권한과 객체를 생성할 수 있는 저장장소가 있어야 한다
  - ▶ DBA는 user에게 권한을 주는 데이터 조작어(DCL) 명령을 사용한다
  - ▶ 다른 user의 table을 참조하려면 참조되는 테이블은 동일한 데이터베이스 내에 있어야 한다
  - ▶ 참조하는 테이블이 제약조건을 만드는 user의 소유가 아니라면 소유자 이름(Schema)이 제약조건에서 참조되는 table 이름 앞에 붙어야 한다
  - ▶ DEFAULT 옵션을 이용하면 column의 기본값을 지정할 수 있다. 새로운 row 입력시 해당 컬럼에 값이 입력되지 않으면 default 옵션에 설정된 값이 자동으로 부여된다

# 테이블 생성

- ▶ 테이블명, 컬럼명, 데이터 타입 등 정의
- ▶ [연습] 오른쪽 DDL 쿼리를 이용, book 테이블을 만들어 봅시다
- ▶ DESC 명령을 이용, 생성된 테이블이 원하는 구조대로 만들어졌는지 확인해 봅니다

```
CREATE TABLE book (
    book_id NUMBER(5),
    title   VARCHAR2(50),
    author  VARCHAR2(10),
    pub_date DATE DEFAULT SYSDATE
);
```



| book_id | title | author | pub_date   |
|---------|-------|--------|------------|
| 1       | 토지    | 박경리    | 2005-03-12 |
| 2       | 슬램덩크  | 다케이코   | 2006-04-05 |
| ...     | ...   | ...    | ...        |
|         |       |        |            |

# Subquery를 이용한 테이블 생성

- ▶ Subquery의 결과와 동일한 테이블 생성됨
- ▶ 질의 결과 레코드들이 포함됨
- ▶ NOT NULL 제약 조건만 상속됨

```
CREATE TABLE account_employees
AS (
    SELECT *
        FROM employees
    WHERE job_id = 'FI_ACCOUNT'
);
```

# Naming Rules

- ▶ 테이블, 컬럼 등의 이름 명명 규칙
  - ▶ 문자로 시작
  - ▶ 30자 이내
  - ▶ A-Z, a-z, 0-9, \_, \$, #
  - ▶ 같은 사용자가 소유한 다른 객체의 이름과 겹치지 않아야 함  
(다른 사용자 소유의 객체와는 같을 수도 있음)
  - ▶ 오라클 예약어는 사용할 수 없음

# TABLE의 종류

- ▶ 사용자 테이블 (User Tables)
  - ▶ 사용자에 의해 만들어지고 관리되는 테이블의 집합
  - ▶ 사용자 정보를 포함함
- ▶ Data Dictionary
  - ▶ 오라클 서버에 의해 만들어지고 관리되는 테이블의 집합
  - ▶ 데이터베이스 정보를 포함

# 기본 데이터 타입

- ▶ 사용 빈도가 가장 높은 타입들

| Data type      | Description                                                                        |
|----------------|------------------------------------------------------------------------------------|
| VARCHAR2(size) | 가변길이 문자열 (최대 4000byte)                                                             |
| CHAR(size)     | 고정길이 문자열 (최대 2000byte)                                                             |
| NUMBER(p,s)    | 가변길이 숫자. 전체 p자리 중 소수점 이하 s자리<br>(p:38, s:-84~127, 21Byte)<br>자리수 지정 없으면 NUMBER(38) |
| DATE           | 고정길이 날짜+시간, 7Byte                                                                  |

- ▶ 참고

- ▶ VARCHAR2와 CHAR의 차이점을 구분한다
- ▶ INT, FLOAT 등의 ANSI Type도 내부적으로 NUMBER(38)로 변환됨

# 기본 데이터 타입

| Data type              | Description                                                                                         |
|------------------------|-----------------------------------------------------------------------------------------------------|
| NCHAR(size)            | national character set에 따라 결정되는 size 만큼의 고정길이 character data로 최대 2000byte까지 가능. 디플트는 1 character.   |
| NVARCHAR2 (size)       | national character set에 따라 결정되는 size 만큼의 가변길이 character data로 최대 4000 byte까지 가능하며 반드시 길이를 정해 주어야 함. |
| LONG                   | 가변 길이 character data로 최대 2 gigabyte까지 가능.                                                           |
| RAW (size)             | 가변 길이 raw binary data로 최대 2000 까지 가능하며 반드시 길이를 주어야 함.                                               |
| LONG RAW               | 가변길이 raw binary data로 최대 2 gigabyte까지 가능.                                                           |
| BLOB                   | Binary data로 4 gigabyte까지 가능.                                                                       |
| CLOB                   | Single-byte character data로 4 gigabyte까지 가능.                                                        |
| NCLOB                  | national character set까지 포함한 모든 character data로 4 gigabyte까지 가능.                                    |
| BFILE                  | 외부 파일로 저장된 binary data로 4 gigabyte까지 가능.                                                            |
| ROWID                  | Row의 물리적 주소를 나타내는 binary data로 extended rowid 는 10 byte, restricted rowid는 6 byte 길이.               |
| TIMESTAMP              | Date값을 미세한 초 단위까지 저장. NLS_TIMESTAMP_FORMAT 형식으로 처리.                                                 |
| INTERVAL YEAR TO MONTH | 두 datetime 값의 차이에서 YEAR와 MONTH값 만을 저장.                                                              |
| INTERVAL DAY TO SECOND | 두 datetime 값의 차이를 DAY, HOUR, MINUTE, SECOND 까지 저장.                                                  |

# ALTER TABLE

- ▶ 컬럼 추가 (ADD)

- ▶ ALTER TABLE book ADD (pubs VARCHAR2(50));

- ▶ 컬럼 수정 (MODIFY)

- ▶ ALTER TABLE book MODIFY (title VARCHAR2(100));

- ▶ 컬럼 삭제 (DROP)

- ▶ ALTER TABLE book DROP (author);

- ▶ UNUSED 컬럼

- ▶ ALTER TABLE book SET UNUSED (author);

- ▶ ALTER TABLE book DROP UNUSED COLUMNS;

# 기타 테이블 관련 명령

- ▶ 테이블 삭제 (DROP TABLE)
  - ▶ DROP TABLE book;
- ▶ 데이터 삭제 (TRUNCATE TABLE)
  - ▶ TRUNCATE TABLE book;
- ▶ Comment
  - ▶ COMMENT ON TABLE book IS 'this is comment';
  - ▶ SELECT \* FROM user\_tab\_comments;
- ▶ RENAME
  - ▶ RENAME book TO article;
- ▶ ROLLBACK 대상이 아님

# Constraint(제약조건)

- ▶ Database 테이블 레벨에서 특정한 규칙을 설정함
- ▶ 예상치 못한 데이터의 손실이나 일관성을 어기는 데이터의 추가, 변경 등을 예방
- ▶ 종류
  - ▶ NOT NULL
  - ▶ UNIQUE
  - ▶ PRIMARY KEY
  - ▶ FOREIGN KEY
  - ▶ CHECK

# 제약조건 정의

- ▶ Syntax
  - ▶ CREATE TABLE 테이블명 (  
    컬럼명 DataType [DEFAULT 기본값][컬럼 제약 조건],  
    컬럼명 DataType [DEFAULT 기본값][컬럼 제약 조건],  
    ...  
    [테이블 제약조건] ... );
  - ▶ 컬럼 제약 조건 : [CONSTRAINT 이름] constraint\_type
  - ▶ 테이블 제약 조건 : [CONSTRAINT 이름] constraint\_type(column, ...)
- ▶ 주의
  - ▶ 제약조건에 이름을 부여하지 않으면 Oracle이 Sys-Cn의 형태로 자동 부여

# 제약조건

## ▶ NOT NULL

- ▶ NULL 값을 허용하지 않음
- ▶ 컬럼 형태로만 제약조건 정의할 수 있음(테이블 제약 조건 불가)

```
CREATE TABLE book (
    book_id NUMBER(5) NOT NULL
);
```

## ▶ UNIQUE

- ▶ 중복된 값을 허용하지 않음 (NULL은 들어올 수 있음)
- ▶ 복합 컬럼에 대해서도 정의 가능
- ▶ 자동적으로 인덱스 생성

```
CREATE TABLE book (
    book_id NUMBER(5) CONSTRAINT c_nook_u UNIQUE
);
```

# 제약조건

- ▶ PRIMARY KEY
  - ▶ NOT NULL + UNIQUE (인덱스 자동 생성)
  - ▶ 테이블 당 하나만 나올 수 있음
  - ▶ 복합 컬럼에 대하여 정의 가능 (순서 중요)
- ▶ CHECK
  - ▶ 임의의 조건 검사 조건식이 참이어야 변경 가능
  - ▶ 동일 테이블의 컬럼만 이용 가능

```
CREATE TABLE book (
    ...
    PRIMARY KEY (book_id)
);
```

```
CREATE TABLE book (
    rate NUMBER CHECK (rate IN (1,2,3,4,5))
);
```

# 실습

- ▶ [연습] author 테이블 생성
  - ▶ book 테이블의 author 컬럼을 author 테이블로 분리, 관리하고자 합니다.
  - ▶ 다음 조건으로 author 테이블을 생성해 봅시다

## 1. 컬럼 타입

```
author_id : NUMBER(10)
author_name : VARCHAR2(100)
author_desc : VARCHAR2(500)
```

## 2. 제약조건

```
author_id : primary key
author_name : NOT NULL
```

# 실습

- ▶ [연습] book 테이블 변경
  - ▶ book 테이블의 author 컬럼을 삭제해 봅니다
  - ▶ book 테이블에 author\_id 컬럼을 추가합니다
    - ▶ author 테이블의 author\_id 컬럼과 같은 형식(**NUMBER(10)**)으로 지정합니다

# 제약조건

## ▶ FOREIGN KEY

- ▶ 참조 무결성 제약
- ▶ 일반적으로 REFERENCE 테이블의 PK를 참조
- ▶ REFERENCE 테이블에 없는 값은 삽입 불가
- ▶ REFERENCE 테이블의 레코드 삭제시 동작
  - ▶ ON DELETE CASCADE : 해당하는 FK를 가진 참조행도 삭제
  - ▶ ON DELETE SET NULL : 해당하는 FK를 NULL로 바꿈

```
CREATE TABLE book (
    ...
    author_id NUMBER(10),
    CONSTRAINT c_book_fk FOREIGN KEY (author_id)
        REFERENCES author(id)
        ON DELETE SET NULL
);
```

# ADD / DROP CONSTRAINTS

- ▶ 제약조건 추가
  - ▶ ALTER TABLE 테이블명 ADD CONSTRAINT ...
  - ▶ NOT NULL은 추가 못함

```
ALTER TABLE book ADD CONSTRAINT c_book_fk  
FOREIGN KEY (author_id) REFERENCES author(author_id);
```

- ▶ 제약조건 삭제
  - ▶ ALTER TABLE 테이블명 DROP CONSTRAINT 제약조건명
  - ▶ PRIMARY KEY의 경우 FK 조건이 걸린 경우에는 CASCADE로 삭제해야 함

```
ALTER TABLE book DROP CONSTRAINT c_book_fk;  
ALTER TABLE author DROP PRIMARY KEY CASCADE;
```

# ENABLE/DISABLE CONSTRAINTS

- ▶ 제약조건 비활성화
  - ▶ 제약조건 검사를 중지함
  - ▶ CASCADE를 사용하여 의존되어 있는 다른 조건을 함께 중지시킬 수 있음
  - ▶ 대규모 데이터 변경 등의 속도를 빠르게 함
  - ▶ UNIQUE, PRIMARY KEY의 경우 인덱스 제거됨

```
ALTER TABLE book DISABLE CONSTRAINT c_book_fk CASCADE;
```

- ▶ 제약조건 활성화
  - ▶ 중지되어 있던 제약조건 검사를 활성화
  - ▶ UNIQUE, PRIMARY KEY의 경우 인덱스 자동 생성

```
ALTER TABLE book ENABLE CONSTRAINT c_book_fk CASCADE;
```

# CASCADE CONSTRAINT

- ▶ 제약조건이 걸려 있는 테이블이나 컬럼은 삭제시 에러 발생

```
ALTER TABLE book DROP (author_id) CASCADE CONSTRAINT;
```

- ▶ 컬럼이나 테이블을 DROP 할 때 관련 제약조건도 함께 삭제할 때

```
DROP TABLE book CASCADE CONSTRAINT;
```

# Data Dictionary

- ▶ Oracle이 관리하는 모든 정보를 저장하는 카탈로그
- ▶ 내용
  - ▶ 모든 스키마 객체 정보, 스키마 객체의 공간 정보, 컬럼의 기본값, 제약 조건 정보, 오라클 사용자 정보, 권한 및 룰 정보, 기타 데이터베이스 정보 ...
- ▶ Base-Table과 View로 구성됨
  - ▶ VIEW의 Prefix
    - ▶ USER : 로그인한 사용자 레벨
    - ▶ ALL : 모든 사용자 정보
    - ▶ DBA : 관리자
- ▶ SYS schema에 속함
- ▶ 주의
  - ▶ DICTIONARY의 테이블이나 컬럼 이름은 모두 대문자 사용!

# Data Dictionary

## : Example

- ▶ 모든 Dictionary 정보 확인

```
SELECT * FROM DICTIONARY
```

- ▶ 사용자 스키마 객체 확인 (테이블)

```
SELECT object_name  
      FROM user_objects  
     WHERE object_type = 'TABLE';
```

- ▶ 제약조건 확인 (BOOK 테이블의 예)

```
SELECT constraint_name, constraint_type, search_condition  
      FROM user_constraints  
     WHERE table_name = 'BOOK';
```

# Data Dictionary

## : Example 2

- ▶ 제약조건 컬럼 확인

```
SELECT constraint_name, column_name  
      FROM user_cons_columns  
     WHERE table_name = 'EMP';
```

- ▶ 모든 사용자 확인 (dba에서 확인 가능)

```
SELECT username, default_tablespace, temporary_tablespace  
      FROM DBA_USERS;
```

# Oracle SQL

DML - INSERT, UPDATE, DELETE

# Data Manipulation Language

- ▶ 종류
  - ▶ Add New Row(s)
    - ▶ INSERT INTO 테이블명 [(컬럼 리스트)] VALUES (값 리스트);
  - ▶ Modify Existing Row(s)
    - ▶ UPDATE 테이블명 SET 변경내용 [WHERE 조건];
  - ▶ Remove Existing Row(s)
    - ▶ DELETE FROM 테이블명 [WHERE 조건];
- ▶ 트랜잭션의 대상
  - ▶ 트랜잭션은 DML의 집합으로 이루어짐
  - ▶ ALL or NOTHING

# INSERT

- ▶ 테이블에 새로운 튜플을 삽입할 때 사용하는 명령
- ▶ Syntax

```
INSERT INTO table_name [{column [, column ...]}]
{VALUES (value[, value...]) | Subquery};
```

- ▶ 대응하는 column과 value는 개수와 타입이 일치해야 한다
- ▶ 테이블 내 모든 컬럼의 내용을 삽입할 때는 column 명을 생략할 수 있지만,  
이때는 CREATE TABLE 문에 기술된 컬럼 순으로 value 값을 지정해야 한다
- ▶ Subquery를 이용하여 다른 테이블의 검색 결과를 삽입할 수 있다

# INSERT

- ▶ 목시적 방법: 컬럼 이름, 순서 지정하지 않음. 테이블 생성시 정의한 순서에 따라 값 지정

```
INSERT INTO author  
VALUES (1, '박경리', '토지 작가');
```

- ▶ 명시적 방법: 컬럼 이름 명시적 사용. 지정되지 않은 컬럼은 NULL 자동 입력

```
INSERT INTO author( author_id, author_name )  
VALUES (2, '이문열');
```

- ▶ Subquery 이용: 타 테이블로부터 데이터 복사 (테이블은 미리 존재해야 함)

```
INSERT INTO department_usa  
SELECT department_id, department_name  
FROM departments  
WHERE department_name = 'IT';
```

- ▶ 참고로 Subquery 결과를 없는 테이블을 생성하고 데이터를 복사하고자 할 때는 CREATE TABLE AS SELECT 이용

# UPDATE

- ▶ 테이블에 있는 튜플들 중에서 특정 튜플의 내용을 갱신할 때 사용하는 명령

- ▶ Syntax

```
UPDATE table_name  
SET column=value[, column=value ...]  
[WHERE condition];
```

- ▶ WHERE 절이 생략된 UPDATE 문장은 해당 테이블 내의 모든 Row를 변경하므로 주의 해서 사용해야 한다

# UPDATE

- ▶ 조건을 만족하는 레코드를 변경
  - ▶ 10번 부서원의 급여를 100 인상 & 수수료를 0으로 변경

```
UPDATE emp  
SET sal = sal + 100, comm = 0  
WHERE deptno = 10;
```

- ▶ WHERE 절이 생략되면 모든 레코드에 적용
  - ▶ 모든 직원의 급여를 10% 인상

```
UPDATE emp SET sal = sal * 1.1
```

- ▶ Subquery를 이용한 변경
  - ▶ 담당 업무가 'SCOTT'과 같은 사람들의 월급을 부서 최고액으로 변경

```
UPDATE emp SET sal = (SELECT MAX(sal) FROM emp)  
WHERE job =  
(SELECT job FROM emp WHERE ename='SCOTT');
```

# DELETE

- ▶ 테이블에 있는 튜플들 중에서 특정 튜플을 삭제할 때 사용하는 명령

- ▶ Syntax

```
DELETE FROM table_name  
[WHERE condition];
```

- ▶ WHERE 절이 생략된 DELETE 문장은 해당 테이블 내의 모든 Row를 삭제하므로 주의해서 사용해야 한다

# DELETE

- ▶ 조건을 만족하는 레코드 삭제

- ▶ 이름이 'SCOTT'인 사원 삭제

```
DELETE FROM emp
WHERE ename = 'SCOTT';
```

- ▶ 조건이 없으면 모든 레코드 삭제 (주의!)

- ▶ 모든 직원 정보 삭제

```
DELETE FROM emp;
```

- ▶ Subquery를 이용한 DELETE

- ▶ 'SALES' 부서의 직원 모두 삭제

```
DELETE FROM emp
WHERE deptno = (SELECT deptno FROM dept
WHERE dname = 'SALES');
```

# 참고사항

- ▶ 데이터 입력, 수정시 자주 사용되는 Pseudo 컬럼

- ▶ USER : 현재 사용자명
- ▶ SYSDATE : 현재 날짜와 시간
- ▶ ROWID : 열의 위치 정보

```
INSERT INTO emp(eno, hiredate) VALUES (200, SYSDATE);
```

- ▶ DEFAULT : default 값이 정의된 컬럼에 기본 값을 입력할 경우 사용

```
INSERT INTO book VALUES (200, 'Gems', DEFAULT);
```

- ▶ DELETE와 TRUNCATE의 차이

- ▶ Delete는 Rollback 가능. 그러나 대량의 log 등을 유발하므로 Truncate보다 느림

- ▶ 모든 DML 문은 Integrity Constraint를 어길 경우 에러 발생

# Oracle SQL

Transaction

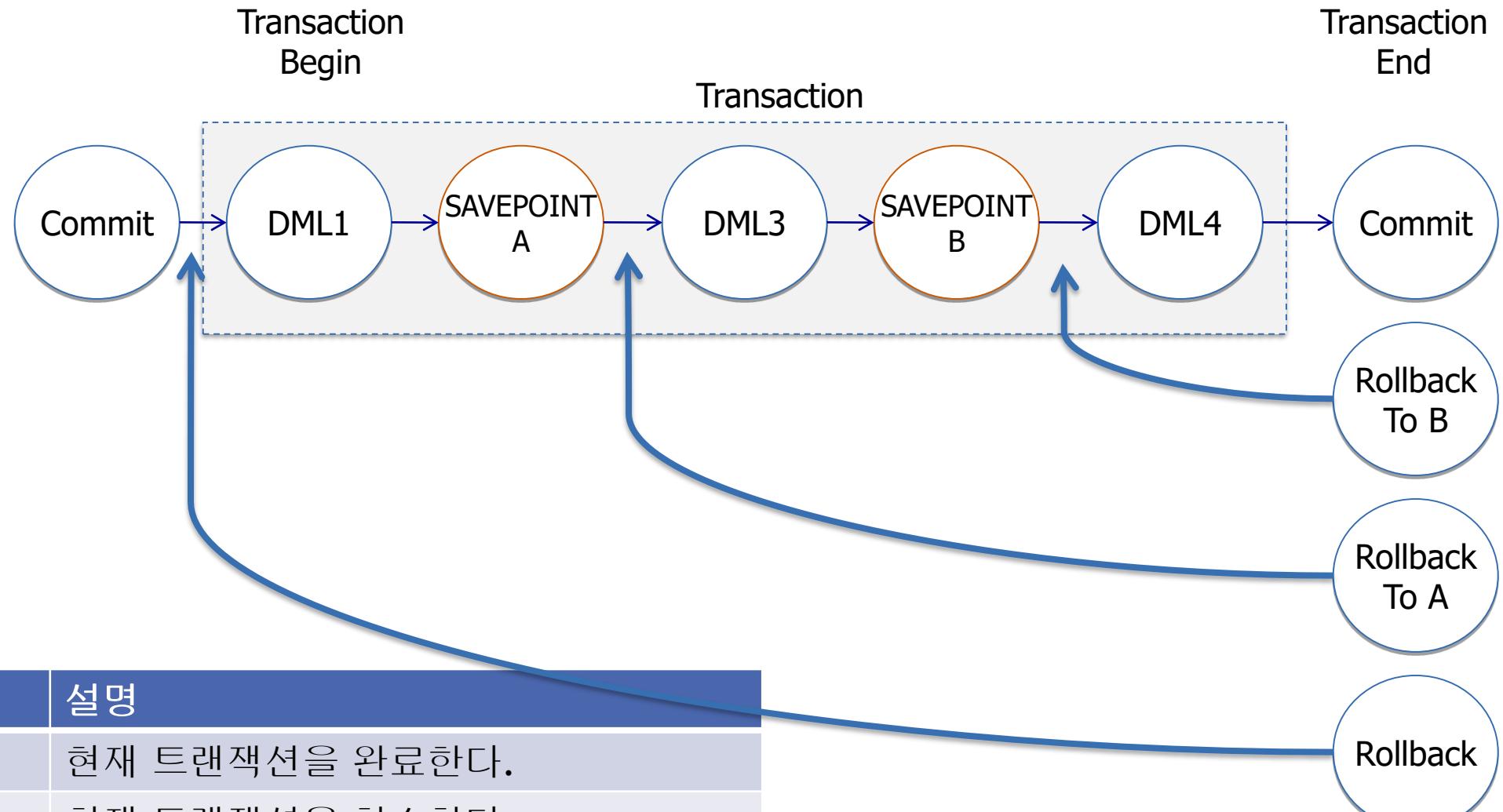
# Transaction

- ▶ 정의
  - ▶ DB에서 하나의 작업으로 처리되는 논리적 작업 단위
  - ▶ DBMS의 Concurrency Control과 Recovery에서 중요한 역할을 수행
- ▶ ACID Property
  - ▶ 원자성(Atomicity): All or Nothing. 하나의 단위로 처리되어야 함 (중간까지만 처리됨은 불가)
  - ▶ 일관성(Consistency) : 데이터베이스의 일관성(무결성)을 깨지 않아야 함
  - ▶ 격리성(Isolation): 다른 Transaction과 동시에 수행되더라도 독립적으로 영향을 받지 않아야 함
  - ▶ 영속성(Durability) : 한번 수행 완료(commit) 되면 영원히 반영되어 있어야 함(시스템 Crash 상황에서라도)

# Transaction in Oracle

- ▶ 구성
  - ▶ DML (INSERT, UPDATE, DELETE)의 집합
  - ▶ DDL이나 DCL은 한 문장이 트랜잭션으로 처리
- ▶ 트랜잭션의 정의
  - ▶ 시작
    - ▶ 명시적인 트랜잭션 시작 명령 없음(타 DBMS와의 차이점)
    - ▶ 첫 DML이 시작되면 트랜잭션 시작
  - ▶ 명시적 종료:
    - ▶ COMMIT
    - ▶ ROLLBACK
  - ▶ 묵시적 종료
    - ▶ DDL, DCL 등이 수행될 때 (atomic commit)
    - ▶ SQL\*PLUS 등에서의 정상적 종료 (atomic commit)
    - ▶ 시스템 오류 (atomic rollback)

# Transaction Control



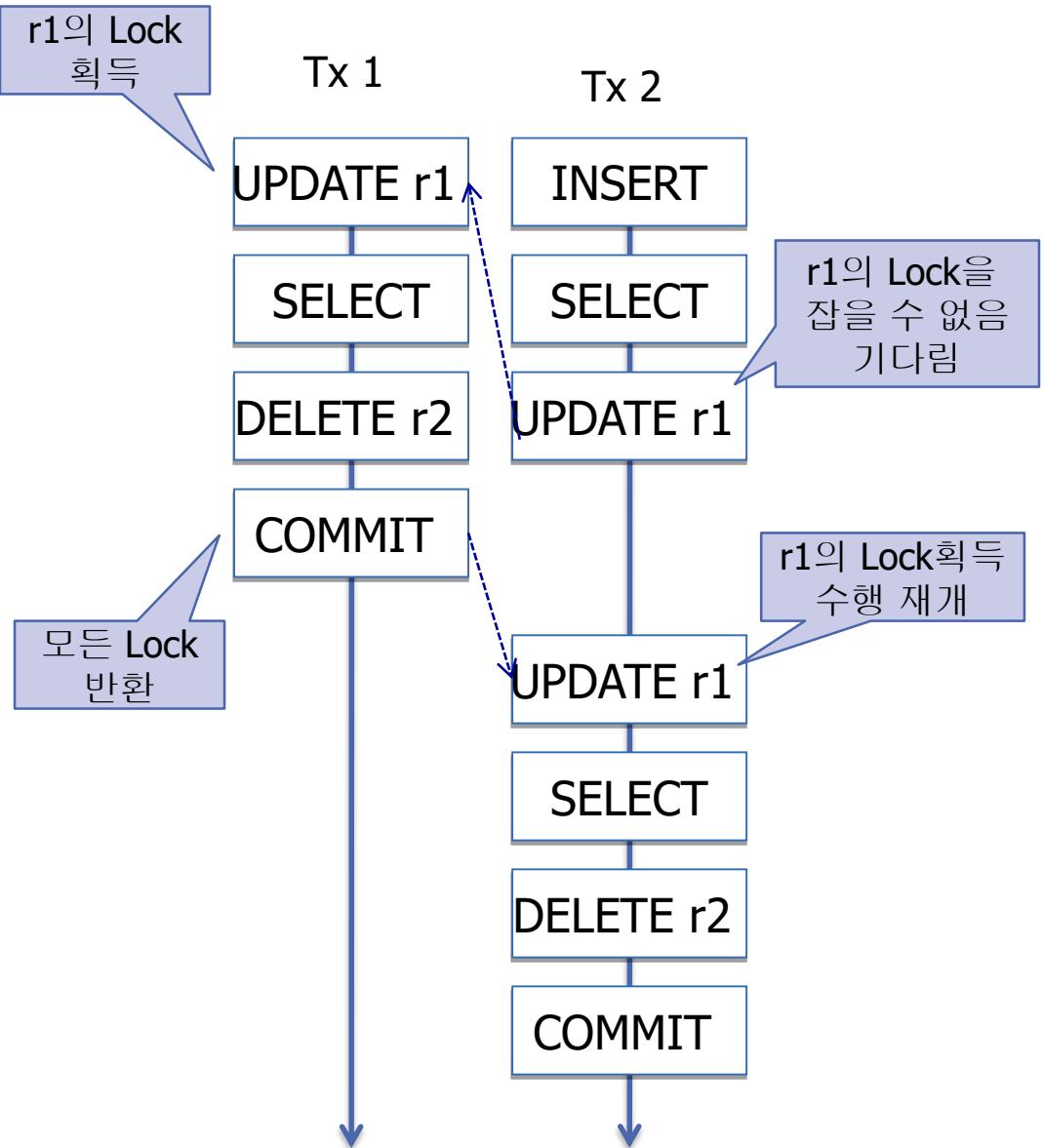
| Command       | 설명                   |
|---------------|----------------------|
| COMMIT        | 현재 트랜잭션을 완료한다.       |
| ROLLBACK      | 현재 트랜잭션을 취소한다        |
| SAVEPOINT x   | 트랜잭션의 중간에 롤백 지점을 만든다 |
| ROLLBACK TO x | 지정한 롤백 지점으로 돌아간다     |

# State of Data

- ▶ Before Commit/Rollback
  - ▶ 현재 사용자는 DML의 결과를 볼 수 있다
  - ▶ 다른 사용자는 현재 DML 결과를 볼 수 없다 (변경 이전 버전이 보임. Read Consistency)
  - ▶ DML에 의해 변경된 모든 Row는 Lock이 걸린다. (다른 트랜잭션에서 수정 불가)
- ▶ After Commit
  - ▶ 변경이 영속적으로 DB에 반영됨. 이전 상태는 사라짐 (더 이상 Rollback 불가)
  - ▶ 변경 결과를 모든 사용자가 볼 수 있음
  - ▶ 모든 Lock이 풀림. 모든 Savepoint 사라짐
- ▶ After Rollback
  - ▶ 모든 DML의 변경이 취소됨
  - ▶ 모든 Lock이 풀림. 모든 Savepoint 사라짐

# Lock in Oracle

- ▶ Lock
  - ▶ Concurrent Transaction의 올바른 실행을 보장
  - ▶ Oracle이 자동으로 Lock을 관리
- ▶ Lock의 종류
  - ▶ DML: Table Share, Row Exclusive
    - ▶ 오라클이 자동으로 Lock 획득 시도
  - ▶ SELECT: No Locks required
  - ▶ DDL : Protects object definitions
- ▶ Locked 된 Row는 COMMIT이나 ROLLBACK이 수행될 때까지 유지됨

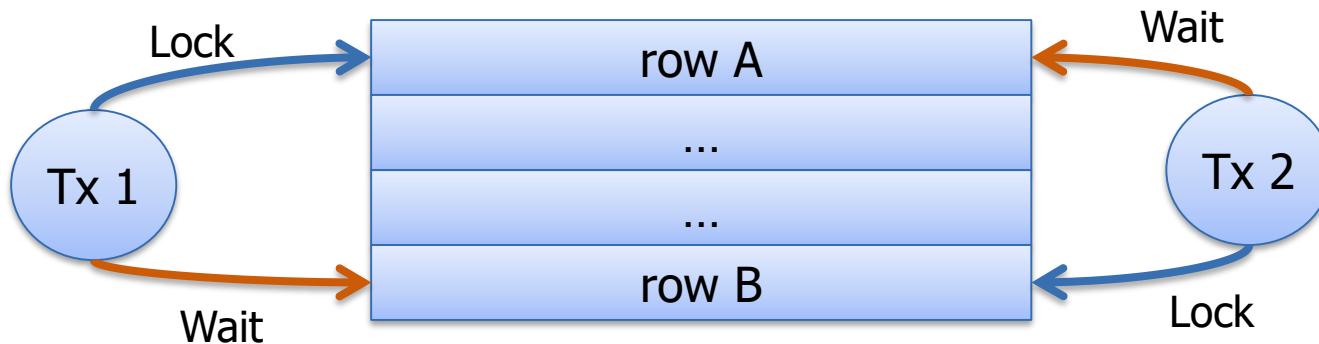


# Set Transaction Property

- ▶ Transaction Type
  - ▶ SET TRANSACTION READ ONLY;
    - ▶ 읽기 전용 (Lock 필요 없음)
    - ▶ Transaction-Level Read Consistency (트랜잭션 시작할 때 봤던 값이 끝까지 나옴)
  - ▶ SET TRANSACTION READ WRITE; (default)
    - ▶ Statement-Level Read Consistency (Statement가 시작할 때 값을 읽게 됨), WRITE 가능
- ▶ Isolation Level
  - ▶ SET TRANSACTION ISOLATION LEVEL SERIALIZABLE;
    - ▶ 나보다 늦게 시작한 트랜잭션이 먼저 수정 후 COMMIT한 row를 수정하려면 ERROR 발생
    - ▶ 모든 충돌연산의 순서가 트랜잭션 순서와 동일해짐
  - ▶ SET TRANSACTION ISOLATION LEVEL READ COMMITTED; (default)
    - ▶ 기본적인 Lock 기반 변경. Commit 된 데이터 변경 가능

# DEADLOCK

- ▶ 둘 이상의 트랜잭션이 서로 상대방의 Lock을 순환 대기하여 어떤 트랜잭션도 더 이상 진행할 수 없는 상태
- ▶ Oracle이 주기적으로 자동 detect 하여 에러를 돌려준다



# Oracle Database

DB Objects

# Oracle Main DB Objects

| Object   | 용도                                                                    |
|----------|-----------------------------------------------------------------------|
| VIEW     | 하나 혹은 복수개의 테이블 혹은 뷰를 기반으로 한 논리적 테이블                                   |
| INDEX    | 테이블 혹은 클러스터의 색인된 컬럼에 나타나는 각 값에 대한 항목을 포함하고 열에 대한 직접적이고 빠른 접근을 제공하는 객체 |
| SEQUENCE | 유일한 순차 값을 생성하는 스키마 객체                                                 |
| SYNONYM  | 테이블, 뷰, 시퀀스 혹은 프로그램 유닛의 별칭(alias)                                     |

# View

## : Create View

- ▶ Syntax

```
CREATE [OR REPLACE] [FORCE|NO FORCE] VIEW view_name  
[(alias[,alias]...)]  
AS Subquery  
[WITH READ ONLY]  
[WITH CHECK OPTION [CONSTRAINT constraint]];
```

- ▶ View의 종류

- ▶ Simple View

- ▶ 단일 테이블을 기반으로 하며 함수나 expression으로 정의된 컬럼을 포함하지 않은 뷰
    - ▶ 몇 가지 제약 조건만 피하면 Update, Insert, Delete 등 DML 작업 수행이 가능

- ▶ Complex View

- ▶ 다수의 테이블을 기반으로 하며 함수나 데이터 그룹 등을 포함하고 있는 뷰
    - ▶ Update, Insert, Delete 등 DML 작업 수행 불가

- ▶ Updatable Join View

# View

## : Create View

- ▶ Simple View 작성 예

```
CREATE OR REPLACE VIEW emp_80
AS SELECT employee_id, first_name, last_name, manager_id, salary
        FROM employees
      WHERE department_id = 80
    WITH READ ONLY;
```

- ▶ 이렇게 작성된 뷰는 일반 테이블처럼 SELECT 할 수 있다

```
SELECT first_name || ' ' || last_name as name,
       salary
  FROM emp_80;

DESC emp_80; # view 구조의 확인
```

# View

## : Create View / Drop View

- ▶ Complex View 작성 예

```
CREATE OR REPLACE VIEW emp_detail
  (employee_id, employee_name, manager_name, department_name)
AS SELECT
    emp.employee_id,
    emp.first_name || ' ' || emp.last_name,
    man.first_name || ' ' || man.last_name,
    department_name
  FROM employees emp, employees man, departments dept
 WHERE emp.department_id = dept.department_id
   AND emp.manager_id = man.employee_id;
```

- ▶ View 를 삭제하고자 하면 DROP VIEW 문을 이용한다

- ▶ VIEW에 대한 정의만 삭제되며 기반이 되는 테이블은 삭제되지 않는다

```
DROP VIEW emp_detail;
```

# View

## : Dictionary for Views

- ▶ USER\_VIEWS
- ▶ USER\_OBJECTS

```
SELECT view_name, text  
      FROM USER_VIEWS;
```

```
SELECT object_name, created, status  
      FROM user_objects  
     WHERE object_type='VIEW' ;
```

- ▶ USER\_OBJECTS의 status는 Base Table의 상태에 따라 변화한다  
Base Table에 변화가 있으면 그 테이블에 의존하는 View는 INVALID 상태로 바뀌고  
View에 대한 질의가 수행되는 순간 Oracle은 View를 다시 컴파일하고 VALID한 경우  
status를 변경한 후 질의를 수행한다

# Index

## : Create Index

- ▶ 인덱스는 데이터에 대한 조회 속도를 높이기 위해 만든다
  - ▶ 데이터에 대한 처리 요청시 Oracle은 효율적인 처리를 위해 필요한 인덱스들을 찾아 이용
  - ▶ 인덱스는 특정 row를 찾거나 일정한 range의 row를 찾을 때 유용
  - ▶ 인덱스는 만들고 나면 Oracle 서버가 자동으로 사용하고 유지보수한다
    - ▶ 데이터에 대한 Update, Insert, Delete를 요청하면 연관 있는 인덱스에 자동으로 반영해준다
  - ▶ 인덱스는 테이블과 독립적이므로 인덱스를 Drop 해도 테이블의 데이터에는 영향을 주지 않는다. 다만 해당 테이블의 DML 처리 속도가 달라질 수 있다
- 
- ▶ Syntax

```
CREATE [UNIQUE] INDEX index_name  
    ON table_name (Column|Expr[, Column|Expr]...) ;
```

# Index

## : Create Index / Drop Index

- ▶ hr.employees 테이블을 복사하여 새 테이블 s\_emp를 만들고 employee\_id 컬럼에 UNIQUE INDEX를 만들어 봅니다

```
CREATE TABLE s_emp  
AS SELECT * FROM employees;
```

```
CREATE UNIQUE INDEX s_emp_id_pk  
ON s_emp (employee_id);
```

- ▶ 인덱스를 삭제하고자 하면 DROP INDEX 문을 실행

- ▶ Syntax

```
DROP INDEX schema.index_name;
```

- ▶ Example

```
DROP INDEX s_emp_id_pk;
```

# Index

## : Dictionary for Indexes

- ▶ USER\_INDEXES
- ▶ USER\_IND\_COLUMNS
- ▶ Example

```
SELECT t.index_name, t.uniqueness, c.column_name, c.column_position
  FROM user_indexes t, user_ind_columns c
 WHERE c.index_name = t.index_name
   AND t.table_name = 'EMP';
```

- ▶ Index가 필요한 경우
  - ▶ WHERE 절이나 JOIN 조건에 빈번하게 사용되는 컬럼
  - ▶ 많은 데이터를 담고 있으나 적은 수의 row들만 받아오는 쿼리 수행
  - ▶ 넓은 범위의 값을 포함하거나 null 값이 많은 컬럼들
  - ▶ 자주 업데이트 되지 않는 테이블

# Sequence

## : Create Sequence

- ▶ 시퀀스는 유일한 정수값을 발생시키는 객체
- ▶ 주로 Primary Key 값을 자동으로 발생시키고자 할 때 생성한다
- ▶ Syntax

```
CREATE SEQUENCE sequence_name
    [INCREMENT BY n]      # 증가값
    [START WITH n]        # 시작값
    [{MAXVALUE n|NOMAXVALUE}] # 최대값
    [{MINVALUE n|NOMINVALUE}] # 최소값
    [{CYCLE|NOCYCLE}]
    [{CACHE n|NOCACHE}]    # Oracle Server가 미리 캐시해 놓을 개수
```

- ▶ NOMAXVALUE가 default : 10의 27승
- ▶ NOMINVALUE가 default : 1
- ▶ CACHE n : 기본값은 20

# Sequence

## : Create Sequence

- ▶ author 테이블을 만들고, PK에 사용할 Sequence를 만들어 봅니다

```
CREATE TABLE author (
    id      NUMBER(10),
    name    VARCHAR2(50) NOT NULL,
    bio     VARCHAR2(500),
    PRIMARY KEY(id)
);

CREATE SEQUENCE seq_author_id
    START WITH 1
    INCREMENT BY 1
    MAXVALUE 1000000;
```

# Sequence

## : Using Sequence

- ▶ Sequence 값을 SQL 문장에서 사용하기 위해서는 아래 pseudo 컬럼들을 이용
  - ▶ CURRVAL : 지정 시퀀스의 현재 값을 알아낸다
  - ▶ NEXTVAL : 지정 시퀀스의 값을 증가시키고 해당 값을 반환한다
- ▶ Example

```
INSERT INTO author (id, name)
    VALUES (seq_author_id.NEXTVAL, 'Steven King');

SELECT seq_author_id.CURRVAL FROM dual;
```

# Sequence

: Alter Sequence / Drop Sequence

- ▶ Sequence의 증가값, 최소값, 최대값, 캐시 수 등을 변경하려면 ALTER SEQUENCE 문을 사용한다
- ▶ Syntax

```
ALTER SEQUENCE sequence_name
  [INCREMENT BY n]    # 증가값
  [ {MAXVALUE n|NOMAXVALUE} ] # 최대값
  [ {MINVALUE n|NOMINVALUE} ] # 최소값
  [ {CYCLE|NOCYCLE} ]
  [ {CACHE n|NOCACHE} ]      # Oracle Server가 미리 캐시해 놓을 개수
```

- ▶ 시퀀스를 삭제하려면 DROP SEQUENCE 문을 이용한다

```
DROP SEQUENCE schema.index_name;
```

# Sequence

## : Dictionary for Sequence

- ▶ USER\_SEQUENCES
- ▶ USER\_OBJECTS
- ▶ Example

```
SELECT sequence_name, min_value, max_value, increment_by, last_number  
      FROM user_sequences;
```

```
SELECT object_name  
      FROM user_objects  
     WHERE object_type='SEQUENCE' ;
```

# Oracle Database

JDBC Programming

# JDBC Programming

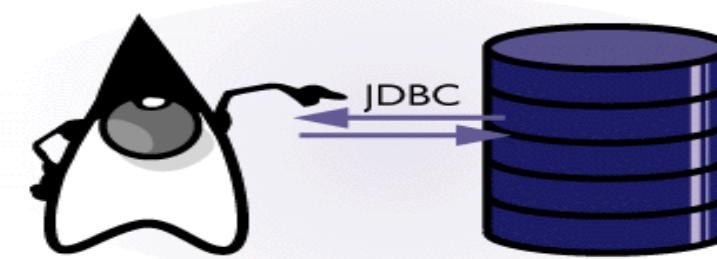
JDBC 기본 API

# JDBC 개요

## : 정의

- ▶ JDBC (Java Database Connectivity)

- ▶ 자바를 이용한 데이터베이스 접속과 SQL 문장의 실행, 그리고 실행 결과로 얻어진 데이터의 핸들링을 제공하는 방법과 절차에 관한 규약
- ▶ 자바 프로그램 내에서 SQL 문장을 실행하기 위한 자바 API
- ▶ SQL과 프로그래밍 언어의 통합 접근 중 한 형태
- ▶ 개발자를 위한 표준 인터페이스인 JDBC API와 데이터베이스 벤더, 또는 기타 서드파티에서 제공하는 드라이브(driver)



# JDBC 개요

## : 환경 구성

- ▶ JDK 설치
  - ▶ <http://java.sun.com>
- ▶ JDBC 드라이버 설치
  - ▶ 오라클 JDBC 드라이버를 다운로드 받는다 (ojdbc6.jar)
    - ▶ <http://www.oracle.com/technetwork/apps-tech/jdbc-112010-090769.html>
    - ▶ 다운로드 받은 JDBC 드라이버를 %JAVA\_HOME%\jre\lib\ext에 복사
    - ▶ 오라클을 설치한 경우의 위치:  
C:\oraclexe\app\oracle\product\11.2.0\server\jdbc\lib
  - ▶ 또는 CLASSPATH에 경로 추가
  - ▶ 이클립스의 경우
    - ▶ 프로젝트 우클릭 - Properties - Java Build Path - Libraries(탭) - Add External JARs

# JDBC 개요

: 참고

- ▶ Java API Reference
  - ▶ <https://docs.oracle.com/javase/8/docs/api/>
- ▶ JDBC Tutorial
  - ▶ <https://docs.oracle.com/javase/tutorial/jdbc/index.html>

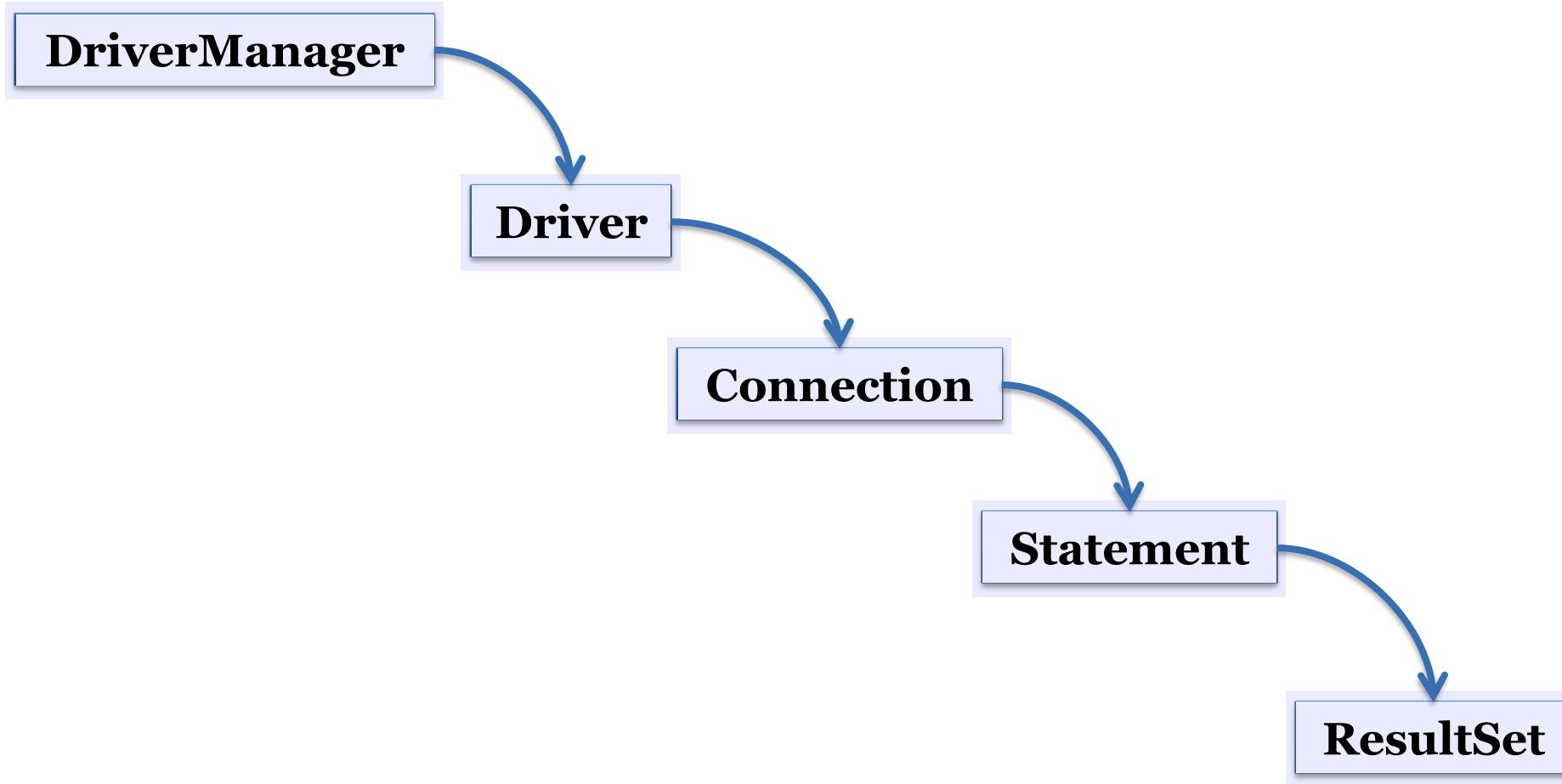
# JDBC 사용

## : 단계별 정리

- ▶ JDBC를 이용한 데이터베이스 연결 방법
  - ▶ 1단계 : `import java.sql.*;`
  - ▶ 2단계 : 드라이버 로드
  - ▶ 3단계 : `Connection` 객체 생성
  - ▶ 4단계 : `Statement` 객체 생성 및 질의 수행
  - ▶ 5단계 : SQL 결과물이 있다면 `ResultSet` 객체를 생성
  - ▶ 6단계 : 모든 객체를 닫는다

# JDBC 사용

: 사용 Class



# JDBC 사용

## : 드라이버 로드와 Connection 얻기

- ▶ IMPORT

```
import java.sql.*;
```

- ▶ 드라이버 로드

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

- ▶ Connection 얻기

```
String dburl = "jdbc:oracle:thin:@localhost:1521:xe";
Connection conn = DriverManager.getConnection(dburl, ID, PWD);
```

# JDBC 사용

## : SQL 실행 및 결과 얻기

- ▶ Statement 생성

```
Statement stmt = conn.createStatement();
```

- ▶ 질의 수행 및 결과 얻기

```
ResultSet rs = stmt.executeQuery("SELECT no FROM user");
```

- ▶ 질의 수행을 위한 stmt의 메서드

- ▶ stmt.execute(QUERY\_STRING);

any SQL

- ▶ stmt.executeQuery(QUERY\_STRING);

SELECT

- ▶ stmt.executeUpdate(QUERY\_STRING);

INSERT, UPDATE,  
DELETE

# JDBC 사용

## : 질의 수행 메서드 예제

- ▶ executeQuery(String sql) - SELECT

```
Statement stmt = con.createStatement();
ResultSet rs = stmt.executeQuery("SELECT * FROM emp");
```

- ▶ executeUpdate(String sql) - INSERT, UPDATE, DELETE

```
Statement stmt = con.createStatement();
int updateCount = stmt.executeUpdate("INSERT INTO test VALUES (1)");
```

- ▶ execute(String sql) - 기타 SQL 문

```
Statement stmt = con.createStatement();
if (stmt.execute(sql)) {
    ResultSet rs = stmt.getResultSet();
    ...
} else {
    int rowCount = stmt.getUpdateCount();
    ...
}
```

# JDBC 사용

## : 결과 활용 및 객체 닫기

- ▶ ResultSet으로 결과 받기

```
ResultSet rs = stmt.executeQuery( "select no from user" );
while ( rs.next() )
    System.out.println( rs.getInt( "no" ) );
```

- ▶ Close

```
rs.close();
stmt.close();
conn.close();
```

# JDBC 사용

## : ResultSet

- ▶ SQL 문에 대한 결과를 얻는다
  - ▶ 이동 함수 : next, previous, first, last, beforeFirst, afterLast, relative, absolute
    - ▶ 기본 : TYPE\_FORWARD\_ONLY & CURSOR\_READ\_ONLY  
(= next로만 이동 가능 & Read Only)
  - ▶ 값 추출 함수 : getXXX(숫자/이름) - 데이터 타입에 따라 알맞게

|        | DEPTNO | DNAME      | LOC      |
|--------|--------|------------|----------|
| Cursor | 10     | ACCOUNTING | NEW YORK |
|        | 20     | RESEARCH   | DALLAS   |
|        | 30     | SALES      | CHICAGO  |
|        | 40     | OPERATIONS | BOSTON   |

next()  
↓

getInt(1)    getString(2)    getString("LOC")

# JDBC 사용

- ▶ [실습] ConnectionTest.java
  - ▶ hr/hr 계정에 JDBC를 이용하여 Connection을 연결
  - ▶ 결과 메시지
    - ▶ 접속에 성공했을 때 : "연결 성공!!"
    - ▶ JDBC 드라이버를 로드하지 못했을 때 : "JDBC Driver를 찾지 못했습니다."
    - ▶ 그외 SQL 에러가 발생했을 경우 : "SQLError!!"
- ▶ [실습] SelectTest.java
  - ▶ hr/hr 계정에 JDBC를 이용하여 Connection을 연결, departments 테이블로부터 데이터를 불러와서 {department\_id}:{department\_name} 형태로 출력
  - ▶ 결과 메시지
    - ▶ {department\_id}:{department\_name}의 리스트 형태로 출력
    - ▶ JDBC 드라이버를 로드하지 못했을 때 : "JDBC Driver를 찾지 못했습니다."
    - ▶ 그외 SQL 에러가 발생했을 경우 : "SQLError!!"

```
Problems @ Javadoc Declaration Console
<terminated> SelectTest [Java Application] C:\WProgram
10:Administration
20:Marketing
30:Purchasing
40:Human Resources
50:Shipping
60:IT
70:Public Relations
```

# JDBC 사용

## : 주의

- ▶ 에러시 해결 방법
  - ▶ 에러 메시지를 확인
  - ▶ 대소문자가 틀렸나 확인 (클래스명, 파일명 등)
  - ▶ JDBC는 제대로 로드 되었나?
  - ▶ CLASSPATH나 PATH는 잘 설정되어 있는가?
  - ▶ DBMS는 제대로 실행되고 있는가?
  - ▶ DBMS로의 접근이 가능한가?
  - ▶ DB Name이나 USERNAME, Password는 올바른가?

# JDBC 사용

## : 실습 문제

- ▶ [실습 1] hr/hr 계정의 사원 이름과 매니저 이름을 함께 출력해 봅시다
  - ▶ 사원 이름은 이름 성 순으로 표기합니다
  - ▶ 정렬은 사원 이름 내림차순입니다
  - ▶ HREmpList 프로젝트를 만들고 HREmpList 클래스에서 실행되어야 합니다
- ▶ [실습 2] 사원 검색 프로그램
  - : 클래스 Scanner를 사용하여 사원 이름을 입력 받아 사원 정보를 검색하는 프로그램을 작성해 봅시다
  - ▶ 부분 이름 검색이 가능해야 합니다
  - ▶ 성, 이름 컬럼에 대해 OR 검색이 되어야 합니다
  - ▶ 출력 사원 정보는 이름 성, Email, 전화번호, 입사일입니다
  - ▶ HRSearchEmployees 프로젝트를 만들고 HRSearchEmployee 클래스에서 실행되어야 합니다

# JDBC 사용

## : 실습 문제

- ▶ [실습 3] 급여 검색 프로그램 작성

: 사용자로부터 최소 급여와 최대 급여를 입력 받아 급여가 이 범위 내에 속하는 사원의 정보를 출력하는 프로그램을 작성해 봅시다

- ▶ 정렬은 salary 오름차순입니다
- ▶ HRSalary 프로젝트의 HRSalary 클래스에서 실행되어야 합니다
- ▶ 결과 예시

2000 10000

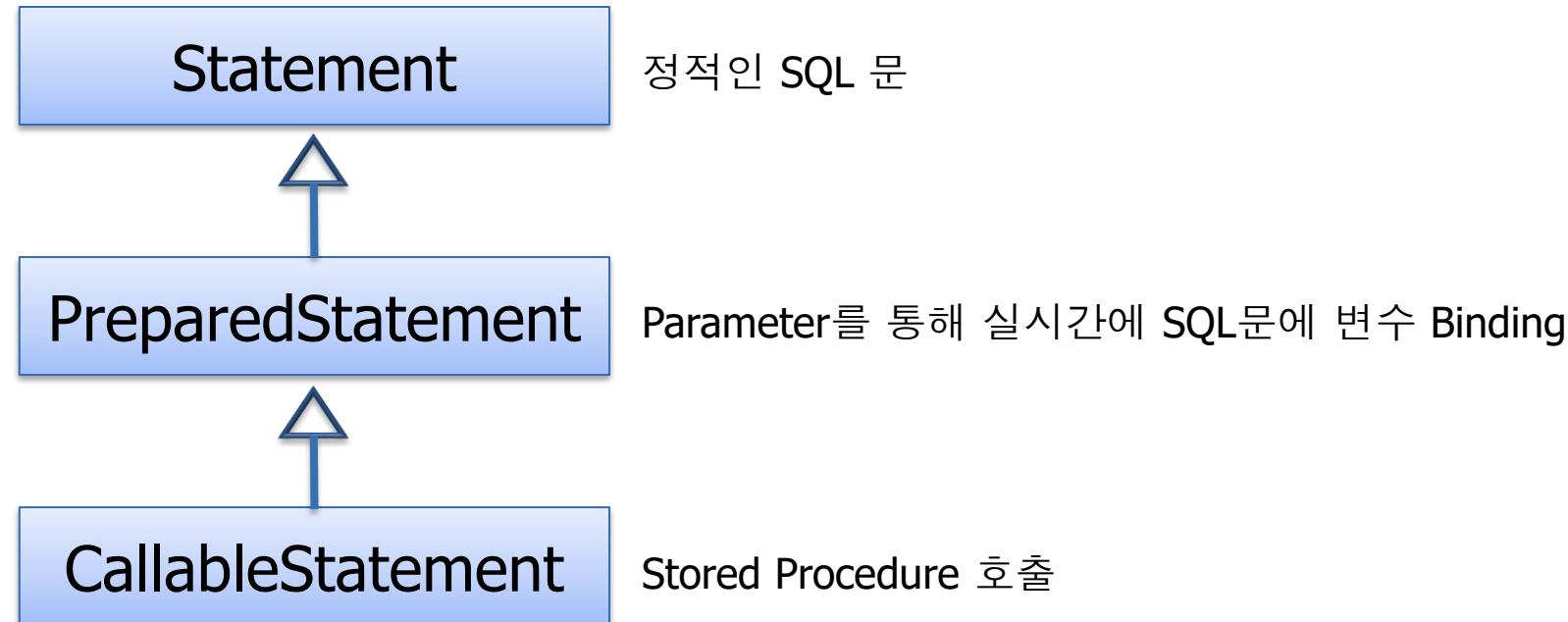
=====

|                 |       |
|-----------------|-------|
| Kevin Freeney   | 3000  |
| Donald OConnell | 4000  |
| Pat Fay         | 90000 |

# Statement 활용

: 상속 관계

- ▶ Statement를 상속받아 각 클래스가 정의되어 있음
  - ▶ 자식 클래스는 부모 클래스의 기능 + alpha



# Statement 활용

## : PreparedStatement

- ▶ 수행방법
  - ▶ SQL 미리 준비 : 파싱, 실행 계획 저장
  - ▶ 실시간에 Parameter 바인딩
- ▶ 바인딩
  - ▶ 바인딩 변수 : ?
  - ▶ setXXX 메서드 : 숫자 1부터
- ▶ 장점
  - ▶ 효율성
  - ▶ 보안 (SQL Injection 회피)

# Statement 활용

## : PreparedStatement

```
...
String sql = "SELECT no FROM user WHERE no = ?";
PreparedStatement pstmt = conn.prepareStatement(sql);
pstmt.setInt(1, 10);
ResultSet rs = pstmt.executeQuery();
...
```

- ▶ [실습 4] 실습문제 2, 3을 PreparedStatement를 사용해 다시 작성해 봅시다

# JDBC Programming

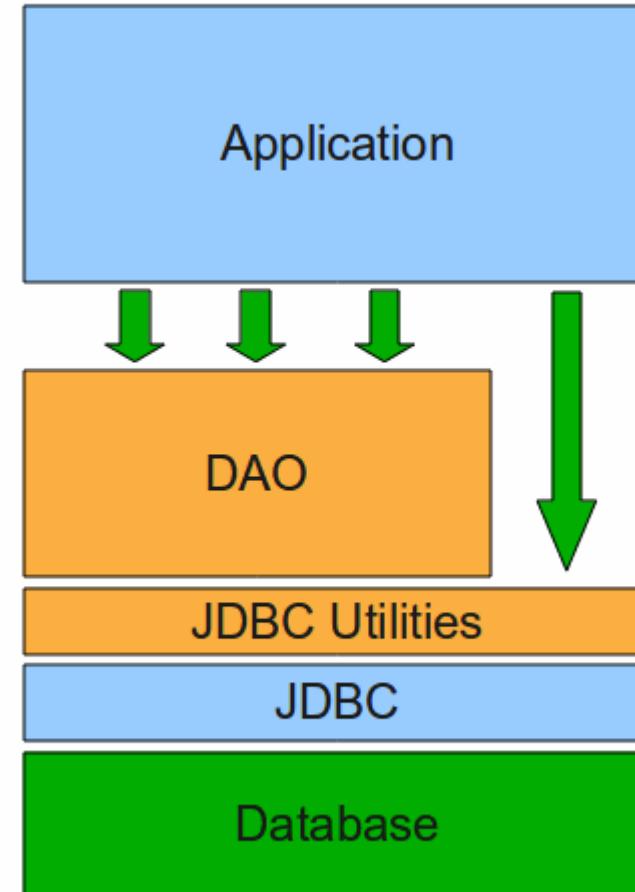
DAO (Data Access Object)

# DAO (Data Access Object)

## : 개념

### ▶ DAO (Data Access Object)

- ▶ DB를 사용하여 데이터를 조회하거나 조작하는 기능을 전담하도록 만든 오브젝트(트랜잭션 객체)
- ▶ 주요 로직에서 Database를 접속, 질의를 수행하고 결과를 반환하는 로직을 DAO에 위임하도록 구현
- ▶ Domain Logic으로부터 Persistence Mechanism을 숨기기 위해 사용
  - ▶ 적절히 디자인을 하면 모든 Domain Logic을 바꾸는 대신 DAO를 바꾸기만 하면 된다
  - ▶ 사용자는 자신이 필요한 Interface를 DAO에 던지고 DAO는 이 인터페이스를 구현한 객체를 사용자에게 반환



# DAO (Data Access Object)

## : 개념

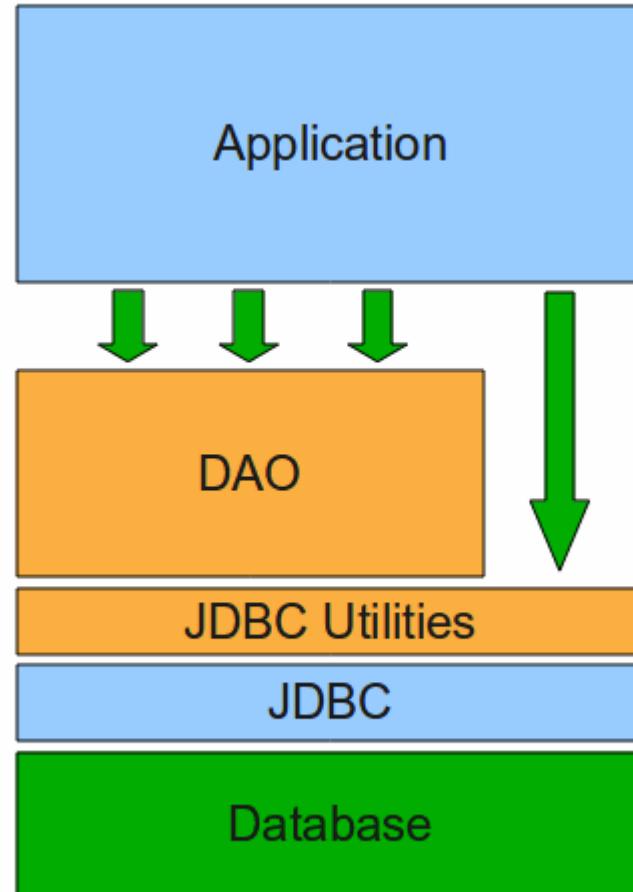
### ▶ DTO(Data Transfer Object)

- ▶ 계층간 데이터 교환을 위한 자바빈즈
- ▶ 컨트롤러, 뷰, 비즈니스 계층, 퍼시스턴스 계층간 데이터 교환

- ▶ VO(Value Object) 패턴이라고도 함

- ▶ 일반적으로 **DTO**는 로직을 갖고 있지 않은 순수 데이터 객체이며 속성과 그 속성에 접근하기 위한 **getter**, **setter** 메서드만 가진다.

- ▶ 추가적으로 **toString()**, **equals()** 등의 **Object** 클래스 메서드를 작성하기도 한다

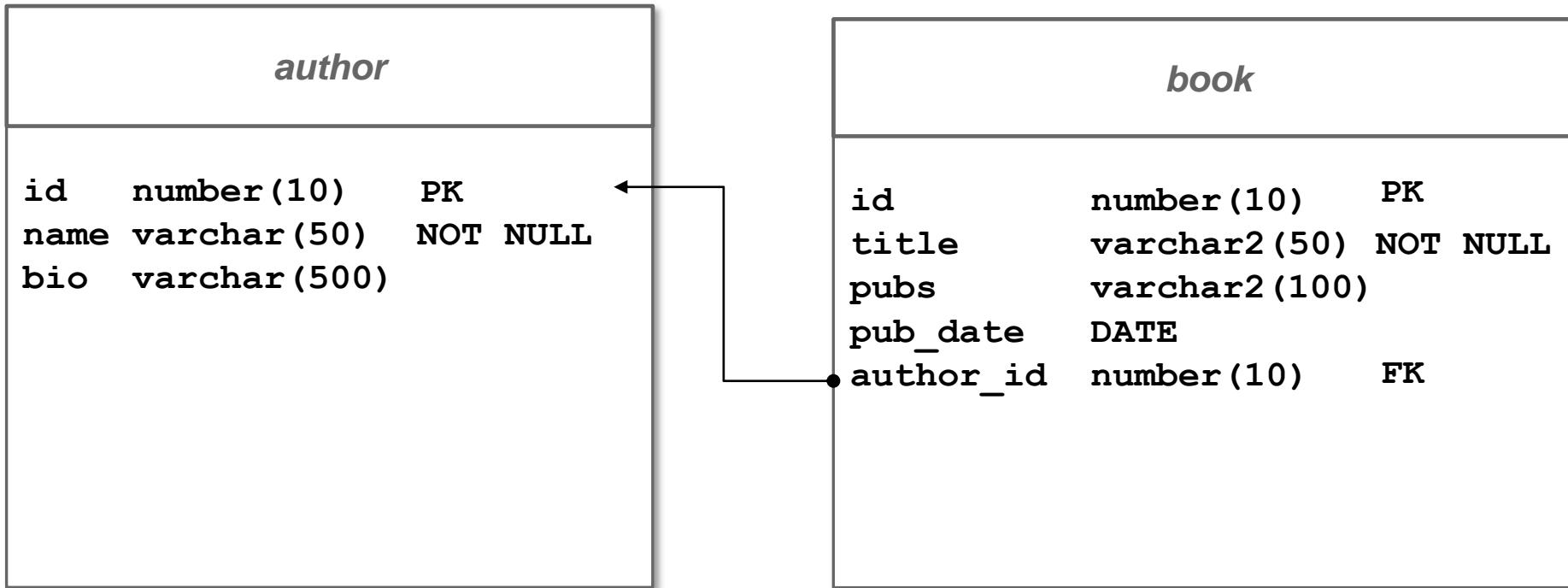


# DAO

## : 예제

- ▶ 다음의 book, author 테이블에서 DAO 패턴을 연습하는 예제입니다

- ▶ Scheme



# DAO

## : 예제

- ▶ 실습을 위해 다음 DDL을 검토하고 데이터베이스에 반영합니다

```
DROP TABLE book CASCADE CONSTRAINTS;
DROP TABLE author CASCADE CONSTRAINTS;
DROP SEQUENCE seq_author;
DROP SEQUENCE seq_book;
```

```
CREATE TABLE author (
    id      NUMBER(10),
    name    VARCHAR2(50) NOT NULL,
    bio     VARCHAR2(500),
    PRIMARY KEY(id)
);
```

```
CREATE TABLE book (
    id      NUMBER(10),
    title   VARCHAR2(50) NOT NULL,
    pubs    VARCHAR2(100),
    pub_date DATE,
    author_id NUMBER(10),
    PRIMARY KEY(id),
    CONSTRAINT c_book_fk FOREIGN KEY (author_id)
    REFERENCES author(id)
    ON DELETE CASCADE
);
```

# DAO

## : 예제

- ▶ Primary Key 생성을 위해 Sequence를 작성하고 데이터베이스에 반영합니다

```
CREATE SEQUENCE seq_book
    START WITH 1
    INCREMENT BY 1
    MAXVALUE 1000000;

CREATE SEQUENCE seq_author
    START WITH 1
    INCREMENT BY 1
    MAXVALUE 1000000;
```

1. `select seq_book.nextval from dual`  
`select seq_author.nextval from dual`
2. `insert` 쿼리를 연습합니다.
3. 연습이 끝나면, 2) DDL를 실행해서 모든 테이블과 시퀀스를 다시 생성합니다.

# DAO

## : 예제

- ▶ AuthorVO의 작성

- ▶ 데이터베이스 테이블의 필드와 매칭되는 필드들을 가진 VO 객체를 생서

```
public class AuthorVO {  
    private Long id;  
    private String name;  
    private String bio;  
  
    public Long getId() { return id; }  
    public void setId(Long id) {  
        this.id = id;  
    }  
  
    // ...  
}
```

# DAO

## : 예제

- ▶ AuthorDAO의 작성

- ▶ AuthorDAO 클래스를 생성하고 공통 부분이 커넥션 부분을 메서드로 분리

```
public class AuthorDAO {  
    private Connection getConnection() throws SQLException {  
        Connection conn = null;  
        try {  
            Class.forName("oracle.jdbc.driver.OracleDriver");  
            String dburl = "jdbc:oracle:thin:@localhost:1521:xe";  
            conn = DriverManager.getConnection(dburl, ID, PWD);  
        } catch( ClassNotFoundException e ) {  
            System.out.println( "JDBC Driver 를 찾을 수 없습니다." );  
        }  
        return conn;  
    }  
}
```

# DAO

## : 예제

- ▶ AuthorDAO의 작성

- ▶ 데이터베이스 -> DTO로 변환하는 메서드를 작성
- ▶ Connection, Statement, ResultSet 객체를 선언
- ▶ 결과를 반환할 변수 선언

```
public List<AuthorVO> getList() {  
    List<AuthorVO> list = new ArrayList<AuthorVO>();  
  
    Connection conn = null;  
    Statement stmt = null;  
    ResultSet rs = null;  
  
    }  
}
```

# DAO

## : 예제

- ▶ 쿼리의 실행과 데이터 바인딩  
(in AuthorDAO)

```
public List<AuthorVO> getList() {  
    // ... Continued  
    try {  
        conn = getConnection();  
        stmt = conn.createStatement();  
  
        String sql = "select id, name, bio from author";  
        rs = stmt.executeQuery( sql );  
  
        while( rs.next() ){  
            Long id = rs.getLong( 1 );  
            String name = rs.getString( 2 );  
            String bio = rs.getString( 3 );  
  
            AuthorVO authorVO = new AuthorVO();  
            authorVO.setId(id);  
            authorVO.setName(name);  
            authorVO.setBio(bio);  
  
            list.add( authorVO );  
        }  
    }  
}
```

# DAO

## : 예제

- ▶ 예외 처리와 자원 반환
- ▶ 결과값 리턴  
(in AuthorDAO)

```
public List<AuthorVo> getList() {  
    // ... Continued  
    } catch ( SQLException e ) {  
        System.out.println( "error:" + e );  
    } finally {  
        try {  
            if( rs != null ) {  
                rs.close();  
            }  
            if( stmt != null ) {  
                stmt.close();  
            }  
            if( conn != null ) {  
                conn.close();  
            }  
        } catch (SQLException e) {  
            e.printStackTrace();  
        }  
    }  
    return list;  
}
```

# DAO

## : 실습

- ▶ [실습] AuthorDAO를 완성해 봅시다
- ▶ 인터페이스는 다음과 같이 구성합니다.

| Interface                                 | 설명                                                          |
|-------------------------------------------|-------------------------------------------------------------|
| public List<AuthorVO> getList ()          | 전체 Author의 List를 반환                                         |
| public AuthorVO get (Long id)             | 개별 Author 객체를 반환                                            |
| public boolean insert (AuthorVO authorVO) | AuthorVO 객체를 받아 author 테이블에 INSERT 성공하면 true, 실패하면 false 반환 |
| public boolean delete (Long id)           | 개별 Author를 테이블에서 삭제<br>DELETE 성공하면 true, 실패하면 false 반환      |
| public boolean update (AuthorVO authorVO) | AuthorVO를 받아서 UPDATE<br>성공하면 true, 실패하면 false 반환            |

# DAO

## : 실습

- ▶ [실습] 사원 정보 검색 프로그램 (DAO 버전)

: 클래스 Scanner를 사용하여 사원 이름을 입력 받아 사원 정보를 검색하는 프로그램을 작성해 봅시다

- ▶ 부분 이름 검색이 가능해야 합니다
- ▶ 성, 이름 컬럼에 대해 OR 검색이 되어야 합니다
- ▶ 출력 사원 정보는 이름 성, Email, 전화번호, 입사일입니다
- ▶ HRApp 프로젝트에서 HRMain Class에 main 메서드가 있어야 합니다

# DAO

## : 실습

- ▶ [실습 2] 급여 검색 프로그램 (DAO 버전)

: 사용자로부터 최소 급여와 최대 급여를 입력 받아 급여가 이 범위 내에 속하는 사원의 정보를 출력하는 프로그램을 작성해 봅시다

- ▶ 정렬은 salary 오름차순입니다
- ▶ HRApp 프로젝트에서 HRSalary 클래스에 main 메서드가 있어야 합니다
- ▶ 결과 예시

2000 10000

=====

|                 |       |
|-----------------|-------|
| Kevin Freeney   | 3000  |
| Donald OConnell | 4000  |
| Pat Fay         | 90000 |

# How to Install Oracle on a Mac (2020)

May 10, 2020 | 59 comments

In this post, you're going to learn the exact steps to take to install Oracle on your Mac computer.

This guide includes:

- step-by-step instructions on downloading and setting up the required software
- screenshots at each stage
- explanations of the configurations you need to set (such as connecting to the database)

So if you want to set up Oracle on your Mac computer, you'll love this guide.

Let's get right into it.

## Overall Process

---

Installing an Oracle database on a Mac computer is a bit different to installing it on a Windows or Linux computer.

Oracle doesn't support running an Oracle database directly on a Mac computer. You can't install Oracle Express, for example, in the same way that you can on a Windows computer.

The good news is that you can use a **Virtual Machine**. To get set up with Oracle on your Mac, the general process is shown below.

## Table of Contents

---

1. [Get VirtualBox](#)
2. [Download the Oracle Developer VM](#)
3. [Set Up Oracle VM](#)
4. [Run the Oracle VM](#)
5. [Test Using SQL Developer](#)

Let's take a look at the requirements, and then get into the steps.

Note: there are other ways to access an Oracle database on a Mac (using a Docker container, using a cloud-hosted database such as Oracle Cloud or AWS). This guide doesn't cover those methods, but I'll outline those in another post.

## Requirements for Oracle VM on a Mac

---

So, what do you need to be able to run Oracle using a VM on a Mac?

The requirements for running a virtual machine are listed on [Oracle's page here](#) (last updated 20 Jun 2019), and are shown below:

- At least **2GB RAM**. Default VM is 1G RAM, for better performance increase.
- At least **15GB of free space**
- **2GHz processor** (a lesser processor will be acceptable but slower)
- Mozilla Firefox 2.0 or higher, Internet Explorer 7 or higher, Safari 3.0 and higher or Google Chrome 1.0 or higher
- Admin privileges on your computer

My current Mac is a **2018 MacBook Pro** with **8GB of RAM** and a **256GB hard drive**, which runs the VM pretty well.

On my previous MacBook Air (2015, 4GB RAM), it actually ran pretty slowly, because the 4GB RAM is split with 2GB for the VM and 2 GB for the Mac OS, bringing both systems to a crawl. MacBook Airs are not designed to be able to run intensive programs such as VMs.

If you really want to use Oracle on a Mac, but don't quite meet the requirements, you can use [Oracle Live SQL](#), which is a web-based Oracle database.

Assuming you want to install Oracle on your Mac, let's look at the steps.



## Download This Guide in a PDF

Save this guide as a PDF so you can refer to it later. All steps and screenshots included.

Email



DOWNLOAD

## Step 1: Get VirtualBox

The first step to getting Oracle on your Mac is to download a program called VirtualBox. This program allows you to run virtual machines, which are self-contained operating systems. It will let you run a Windows or Linux operating system inside your Mac.

I suggest VirtualBox because it's free, and Oracle provides a ready-made file that you can use with it. One alternative is Parallels, which I explain at the end of this guide.

To download VirtualBox:

**Step 1A:** Go to the VirtualBox website: <https://www.virtualbox.org/wiki/Downloads> (opens in new tab).

The current version is shown at the top of the page (currently it is 6.2.6).



# VirtualBox

## Download VirtualBox

Here you will find links to VirtualBox binaries and its source code.

[About](#)  
[Screenshots](#)  
[Downloads](#)  
[Documentation](#)  
    [End-user docs](#)  
    [Technical docs](#)  
[Contribute](#)  
[Community](#)

### VirtualBox binaries

By downloading, you agree to the terms and conditions of the respective license.

If you're looking for the latest VirtualBox 6.0 packages, see [VirtualBox 6.0 builds](#). Please also use version 6.0 if you're using OS X 10.11 El Capitan or later.

If you're looking for the latest VirtualBox 5.2 packages, see [VirtualBox 5.2 builds](#). Please also use version 5.2 if you're using OS X 10.10 Yosemite or later.

### VirtualBox 6.1.6 platform packages

- [Windows hosts](#)
- [OS X hosts](#)
- [Linux distributions](#)
- [Solaris hosts](#)

The binaries are released under the terms of the GPL version 2.

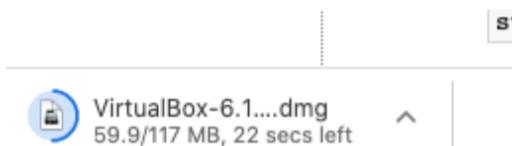
See the [changelog](#) for what has changed.

You might want to compare the checksums to verify the integrity of downloaded packages. The [SHA256 checksums](#) and [MD5 checksums](#) are available.

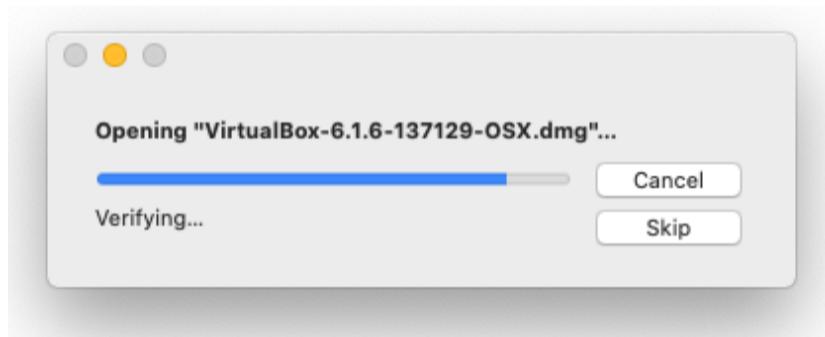
- [SHA256 checksums](#), [MD5 checksums](#)

**Step 1B:** Click on "OS X Hosts". This is a direct link to a .dmg file, which is the installer file for Mac.

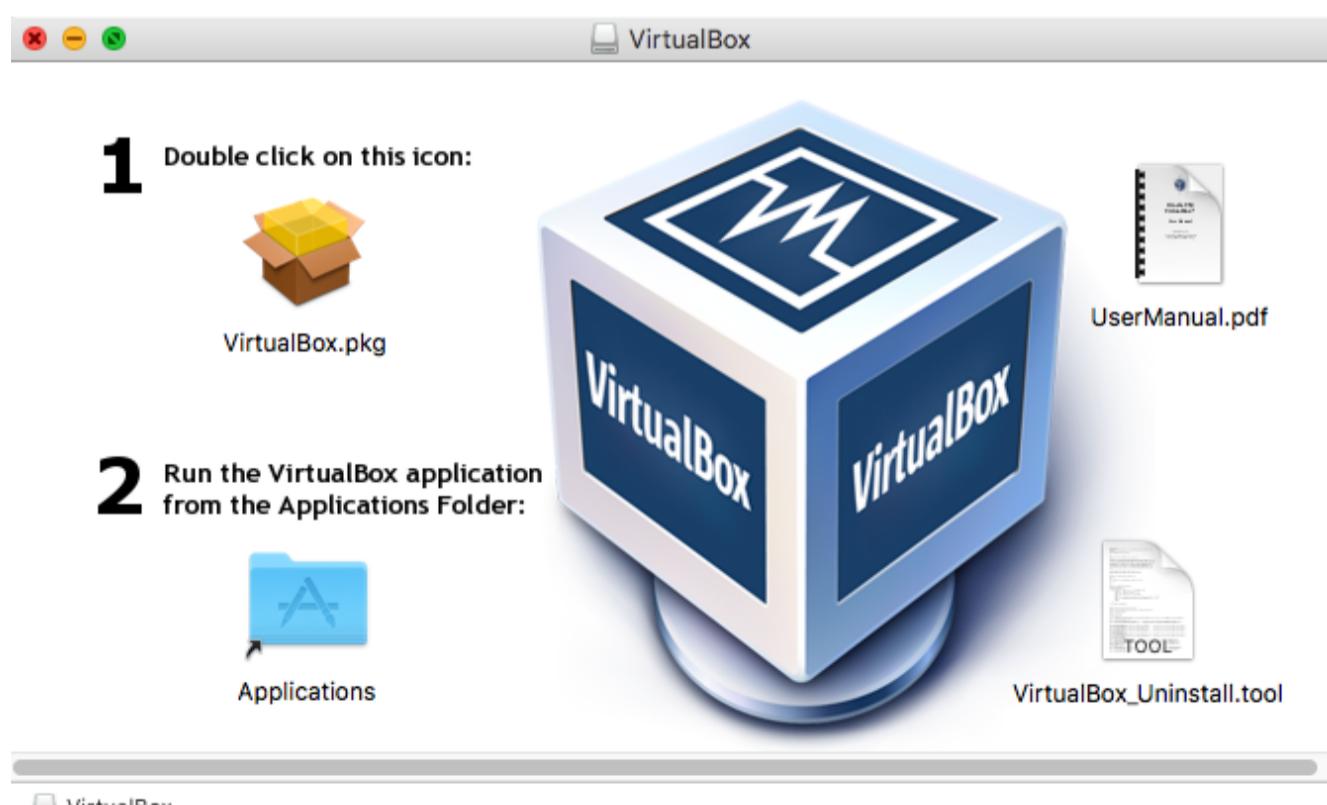
The file will start downloading. It's approximately 122 MB.



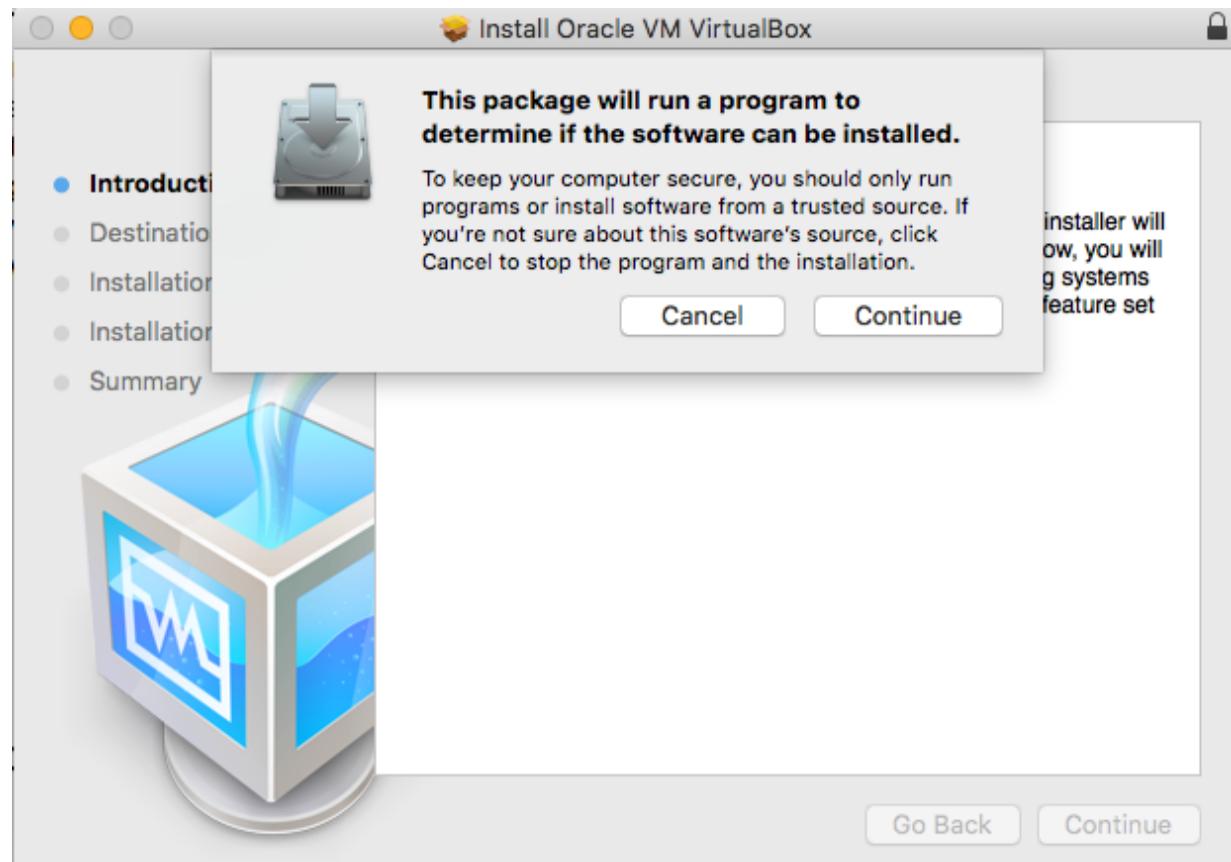
**Step 1C:** Once the file is downloaded, open it.



**Step 1D:** Double click the VirtualBox.pkg icon in the window that appears.



**Step 1E:** Click Continue if a message appears about installing the package.

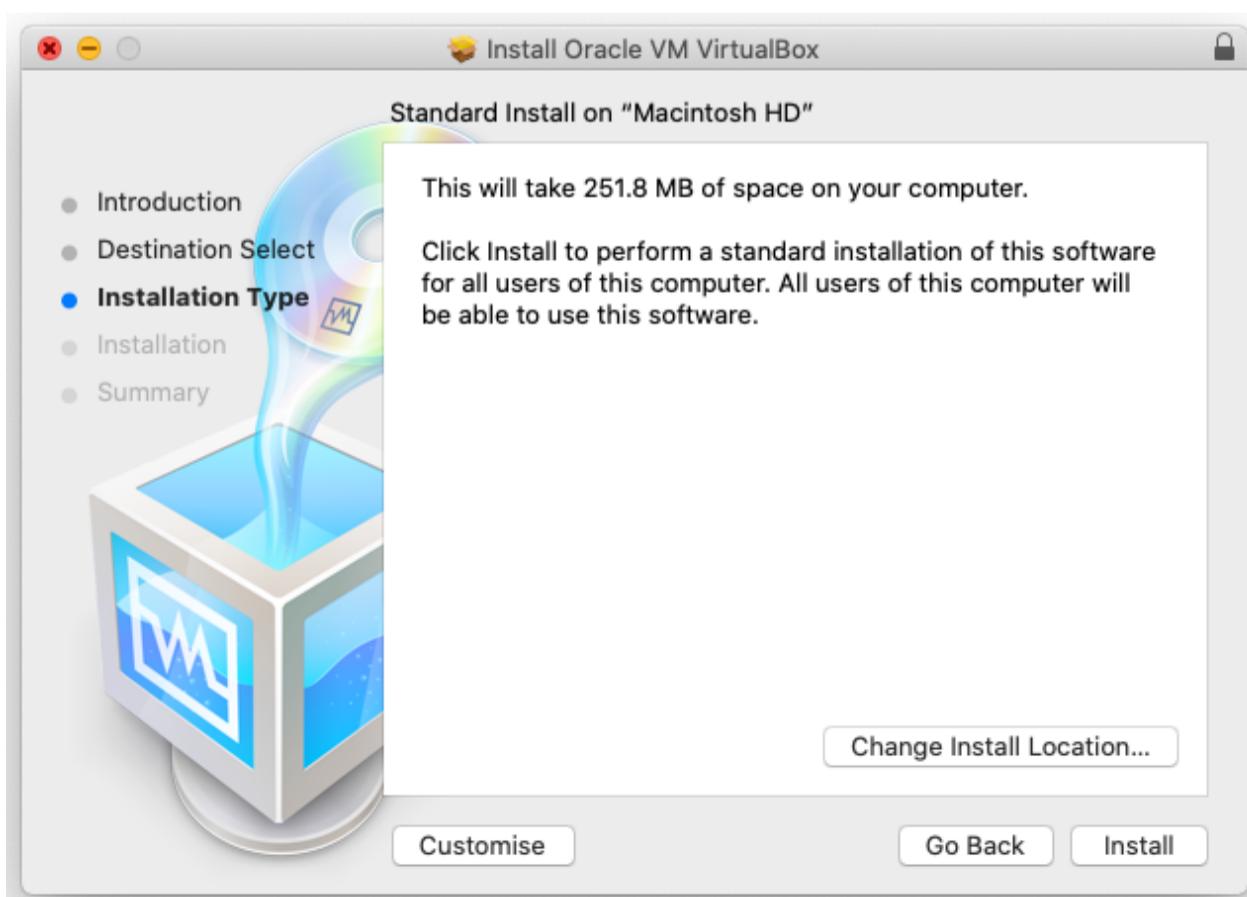


**Step 1F:** Click Continue on the welcome screen.

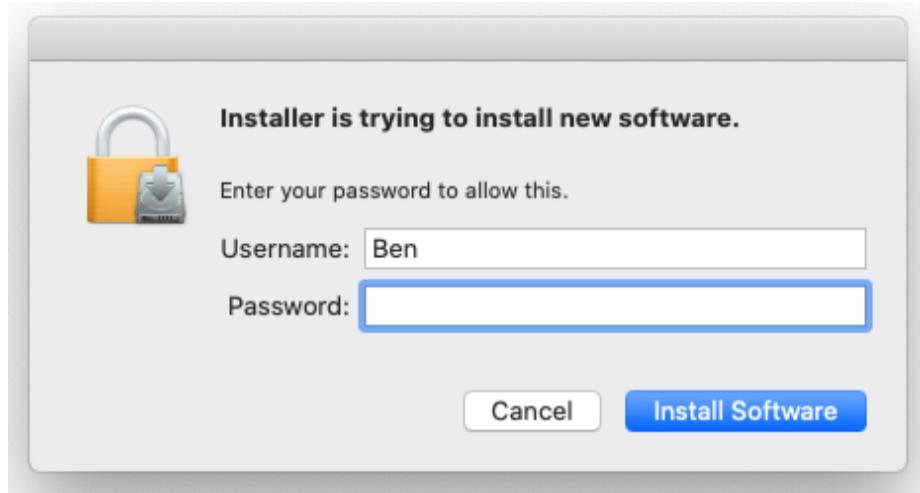


The installer will tell you how much space VirtualBox will take up (approx 251.8 MB).

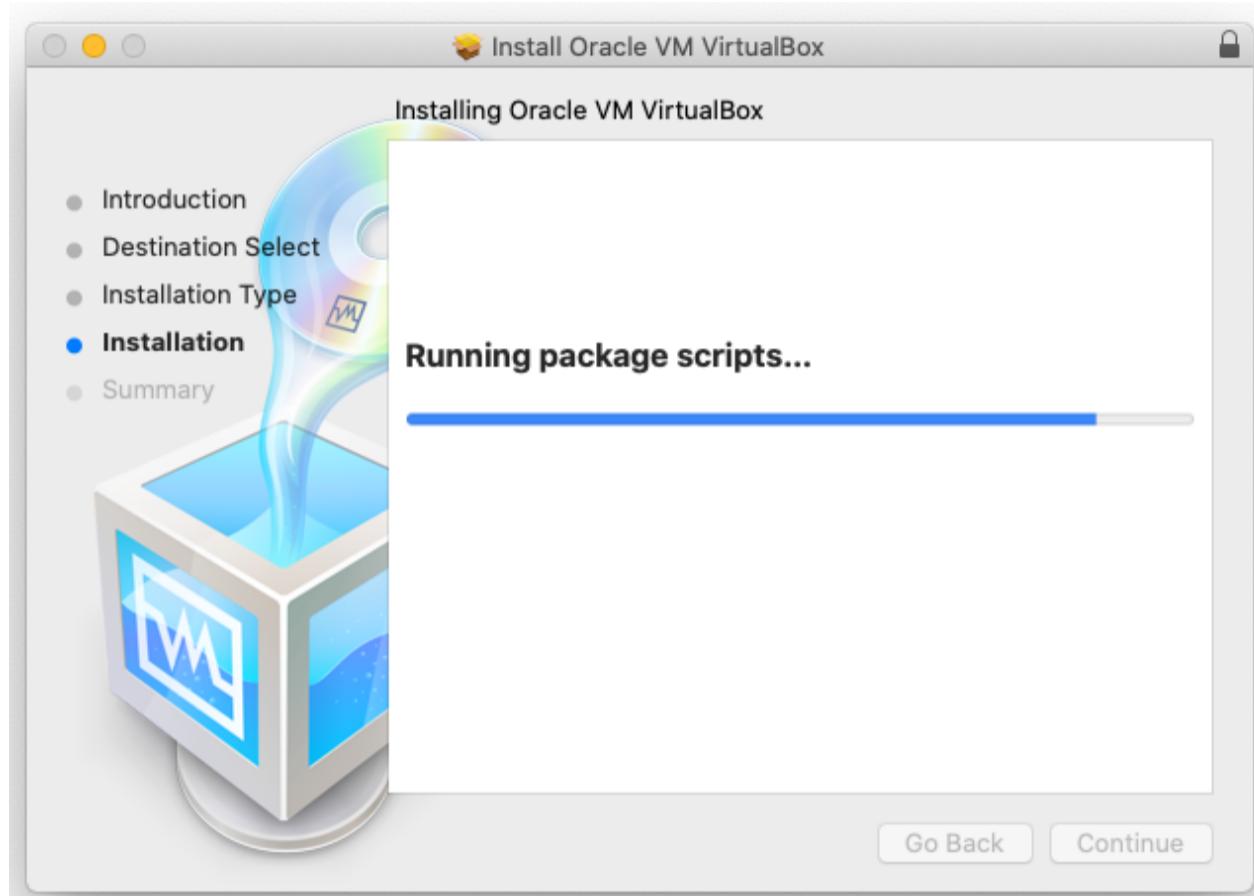
**Step 1G:** Click Install.



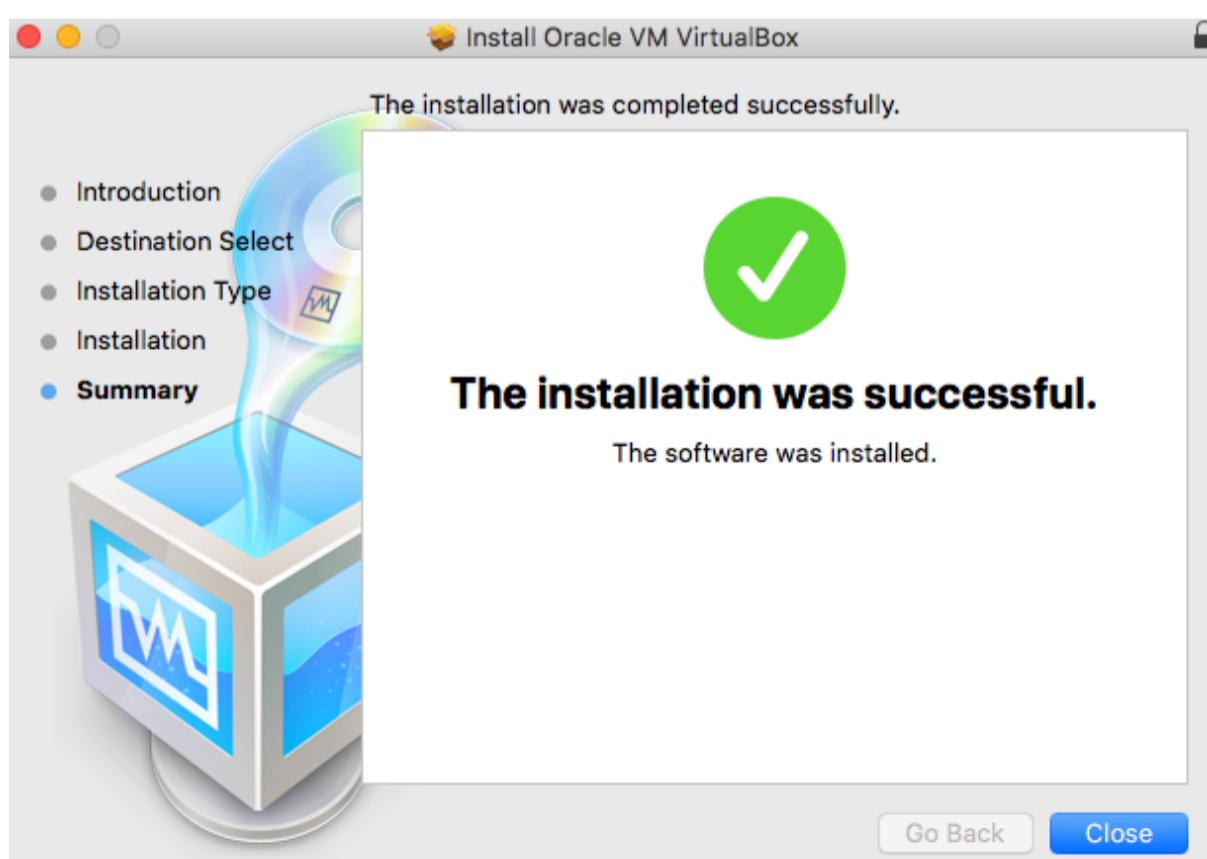
**Step 1H:** Enter your password if prompted.



The program will now be installed.



**Step 1I:** Once completed, it will inform you the installation was successful.



**Step 1J:** Click Close, and move the installer to the trash.

### Installation Failed?

Did you get an error saying the installation had failed for VirtualBox VM?



***The installation failed.***

*The Installer encountered an error that caused the installation to fail. Contact the software manufacturer for assistance.*

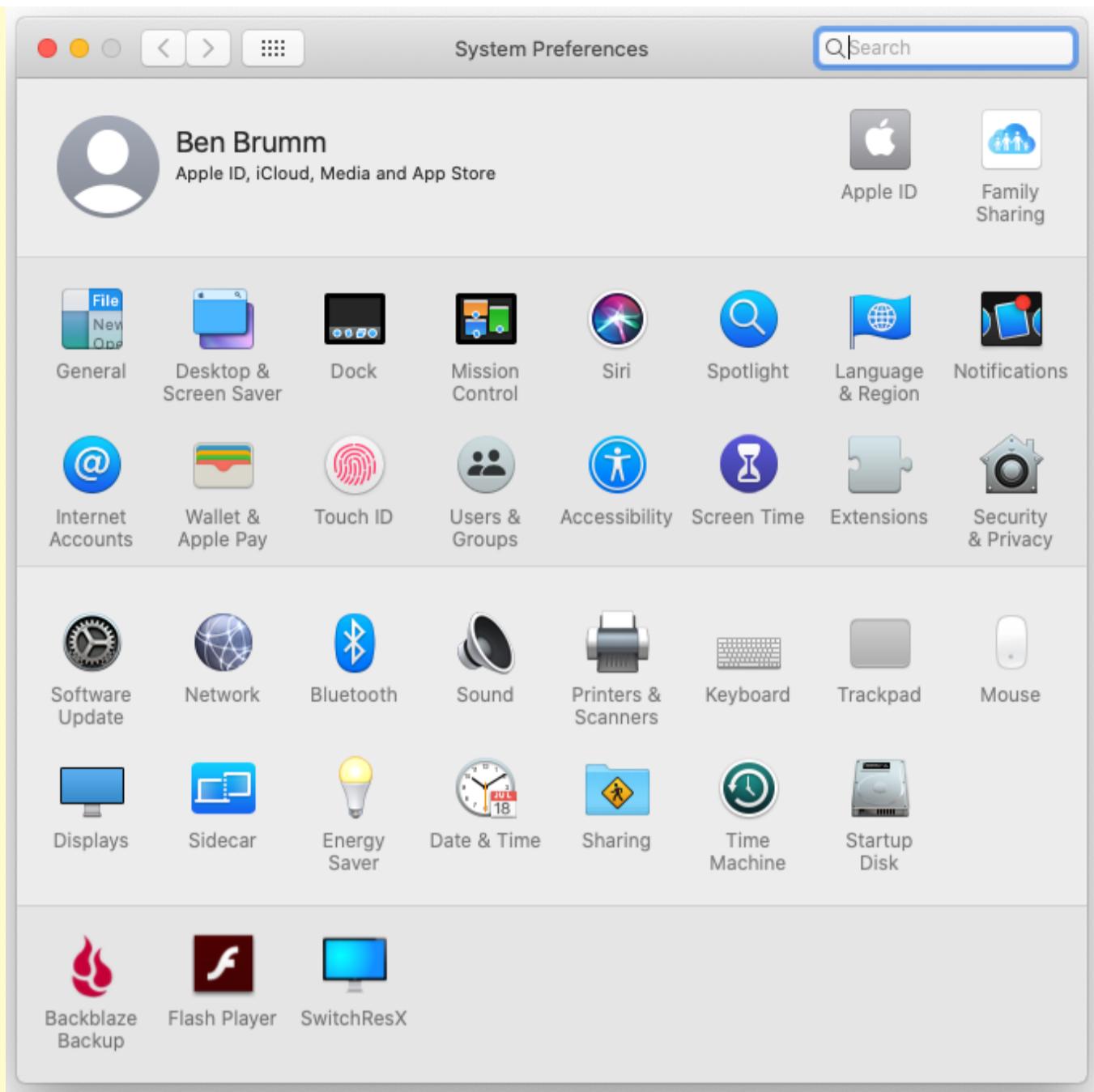
I got this error when attempting to install VirtualBox on 10 May 2020, running macOS Catalina 10.15.4.

How can you resolve this?

These are the steps I followed (thanks to Daniel Meechan's [article here](#)):

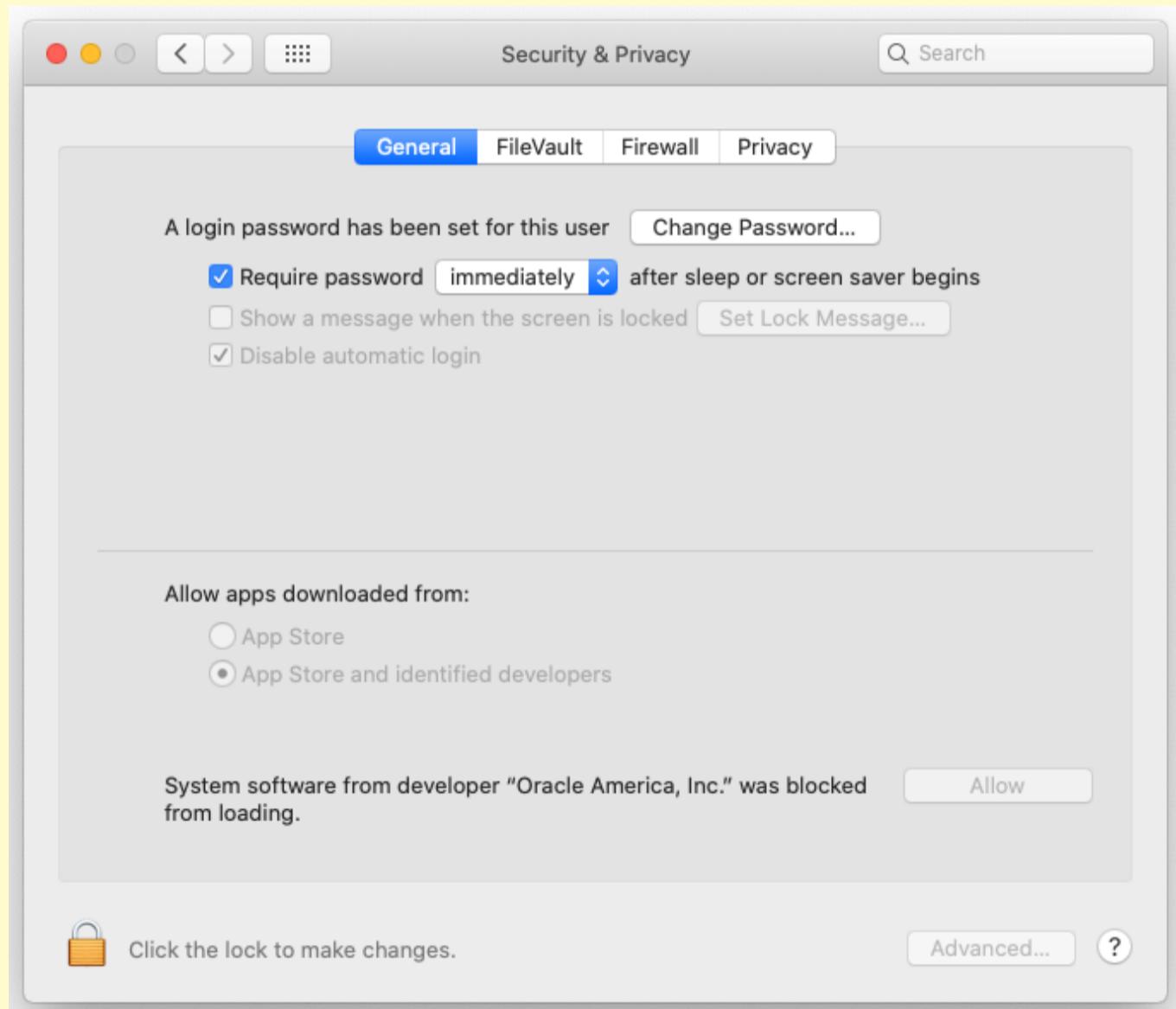
Follow the **steps from 1D to 1G above**, stopping when you get to the screen with the “Change Install Location” and “Install” buttons. Don’t click Install.

Open **System Preferences** from the Apple menu.

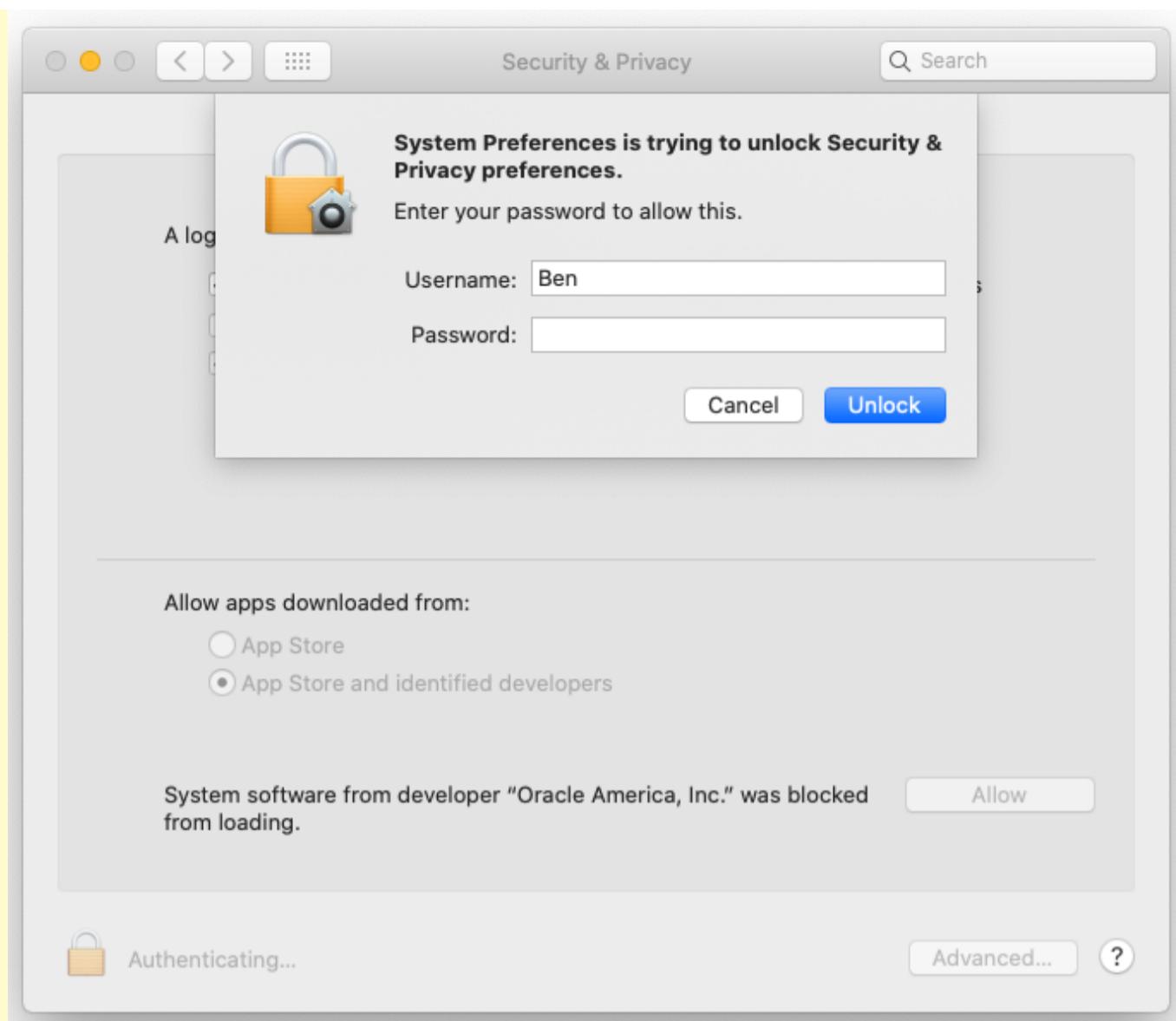


Click on **Security & Privacy** (second row, far right).

Click on the **General** tab.

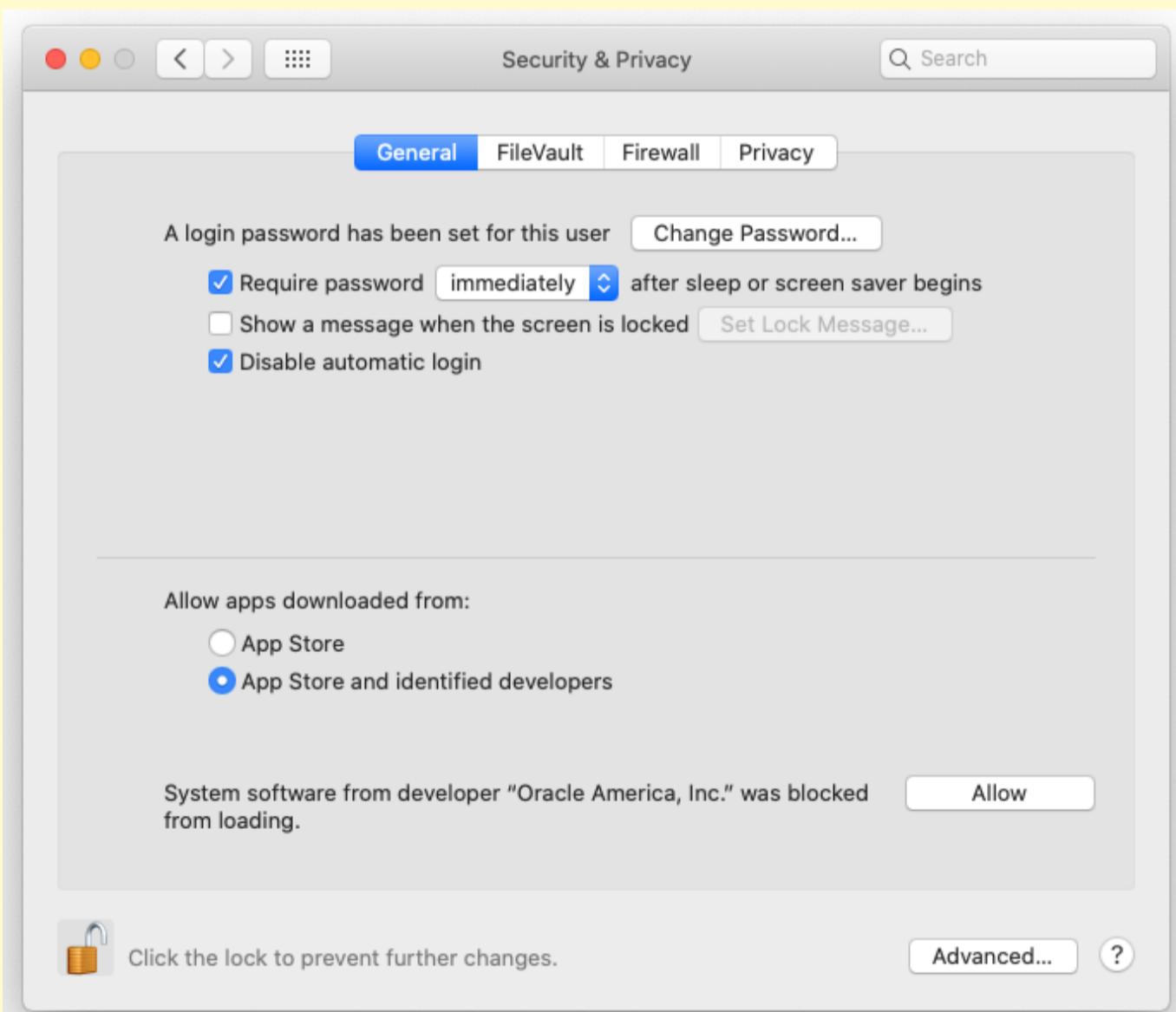


You should see a greyed-out message at the bottom of the window saying "System software from developer "Oracle America Inc." was blocked from loading.



Click the padlock icon on the bottom left of the window to unlock this window.

Enter your password and click OK.



The disabled text and buttons should now be enabled.

**Click Allow.**

Enter your password again.

Now, return to the VirtualBox installer and click Install.

Resume the steps above from 1H. The installation should now be successful.

## Step 2: Download the Oracle Developer VM

Now you've got VirtualBox installed, you need to download the file that contains the Oracle database. This is the virtual machine file and is provided by Oracle.

It's called the "Oracle Developer VM" or "Oracle Developer Day VM", as it was originally created for developers but it has expanded since then.

To download this file:

**Step 2A:** Visit the Oracle Developer Day [download page here](#) (last updated 20 June 2019).

The screenshot shows a web browser window with the Oracle Technology Network logo at the top. The main content area displays the following information:

**Database / Technical Details /**  
**Developer Day - Hands-on Database Application Development**

**Oracle Technology Network**

**Database Virtual Box Appliance / Virtual Machine**

*Set-Up instructions: Database Application Development Hands On Labs*

**Updated: 6/20/2019**

Welcome to the **Oracle Technology Network - Hands-on Database Application Development** lab installation instr provides pre-configured Oracle software for your use.

Please note that this appliance is **for testing purposes only**, as such it is **unsupported** and should not to be used in

- Oracle Linux 7
- Oracle Database 19.3 Linux x86-64
- Oracle SQL Developer 19.1
- Oracle Application Express 19.1
- Hands-On-Labs (accessed via the Toolbar Menu in Firefox)
  - Oracle REST Data Services 19.1
  - Oracle SQL Developer Data Modeler 19.1
  - Oracle XML DB

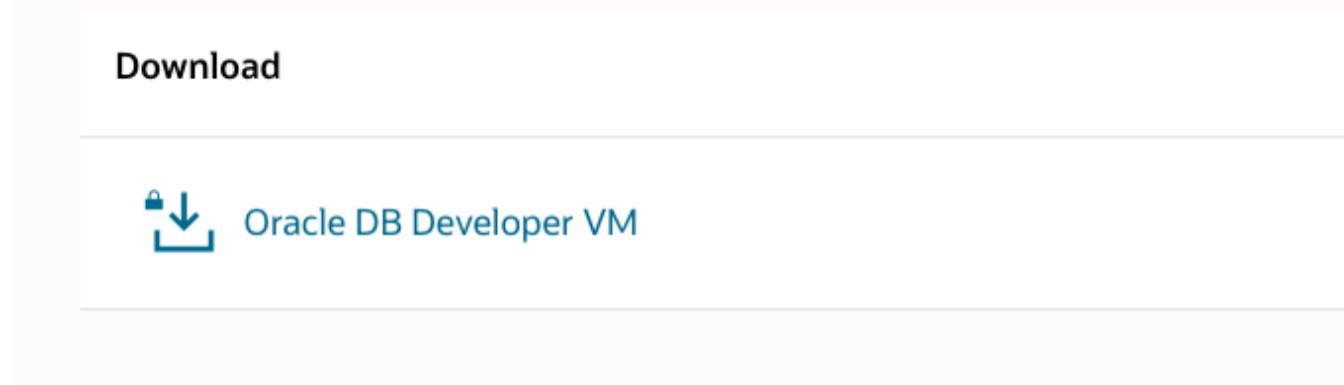
This page lists the versions of software it comes with. At the time of writing, this is Oracle Linux 7 with **Oracle Database 19.3**.

**Step 2B:** Scroll to the bottom of the page and click the **Oracle DB Develop VM** link.

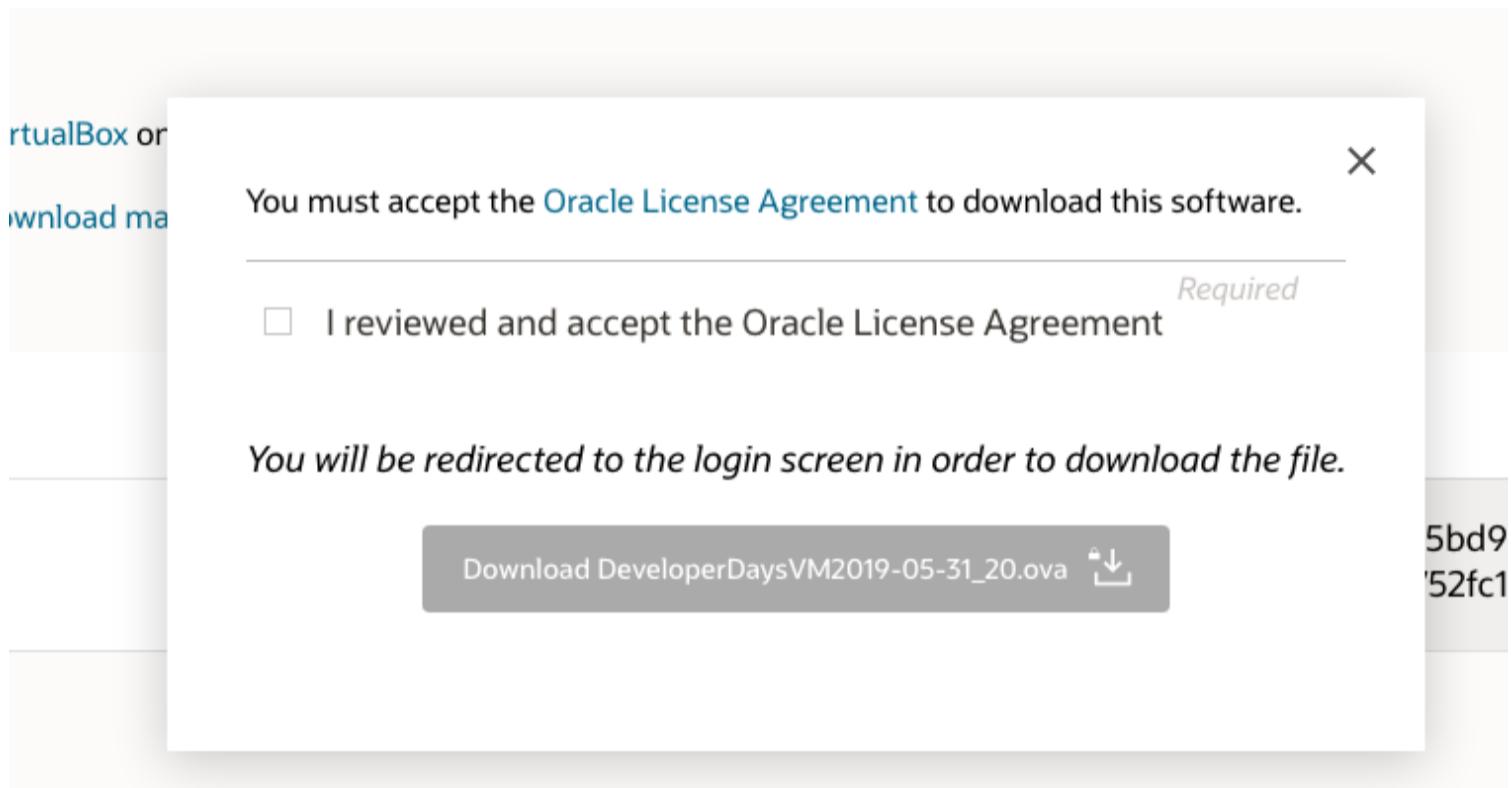
## Setup

**Step 1.** Download and install [Oracle VM VirtualBox](#) on your host system.

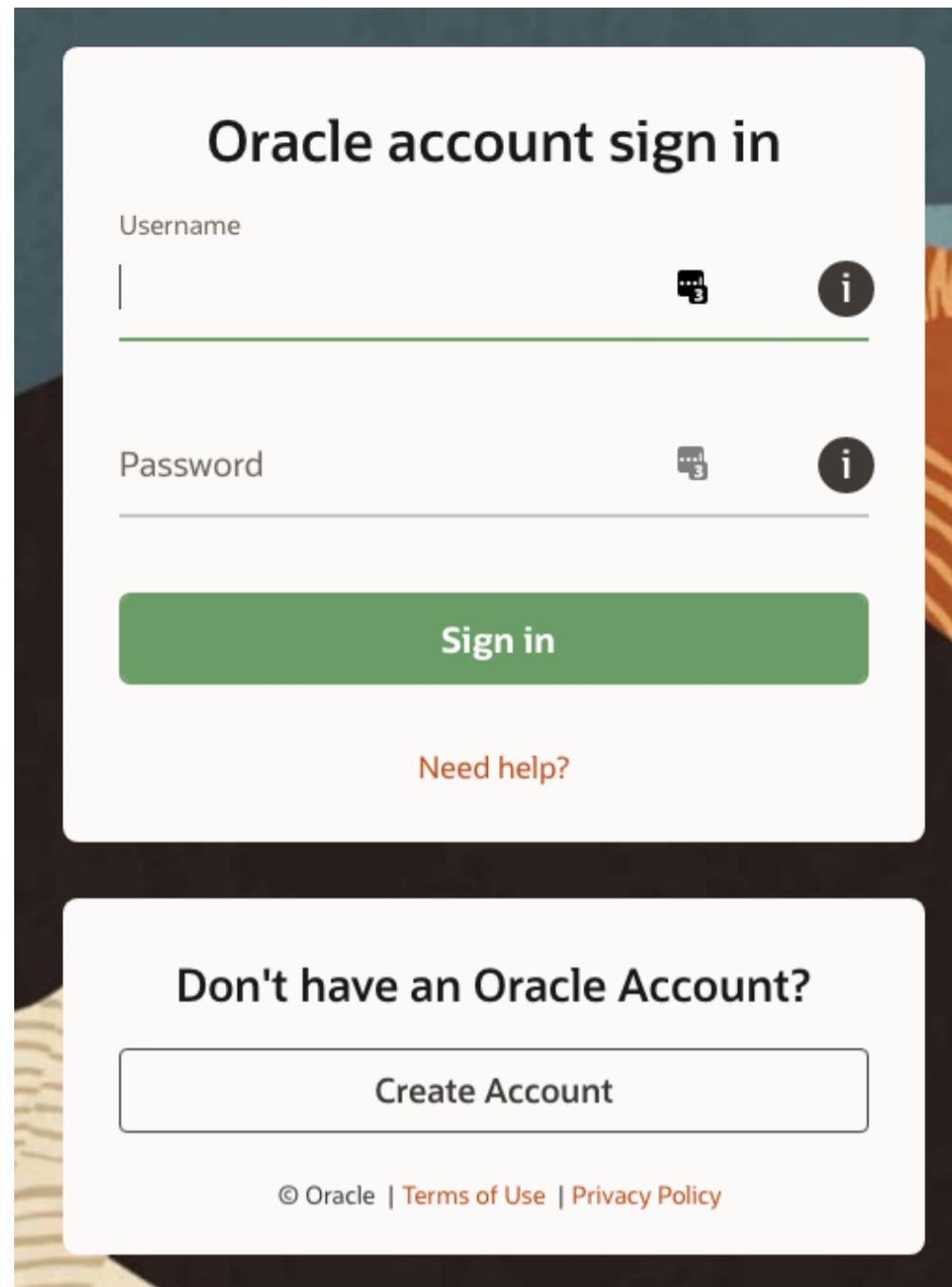
**Step 2.** Download the files (the use of a [download manager](#) is **highly recommended**):



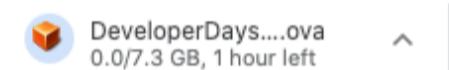
**Step 2C:** On the popup that appears, check the box to accept the license agreement, and click the green Download button.



**Step 2D.** You'll be asked to log in to your Oracle account. If you have an account, enter your details. If you don't have an account, you can create one. It's easy and free.



**Step 2E:** Once you have logged in, the Developer Days VM file will start downloading. It has an “ova” extension.



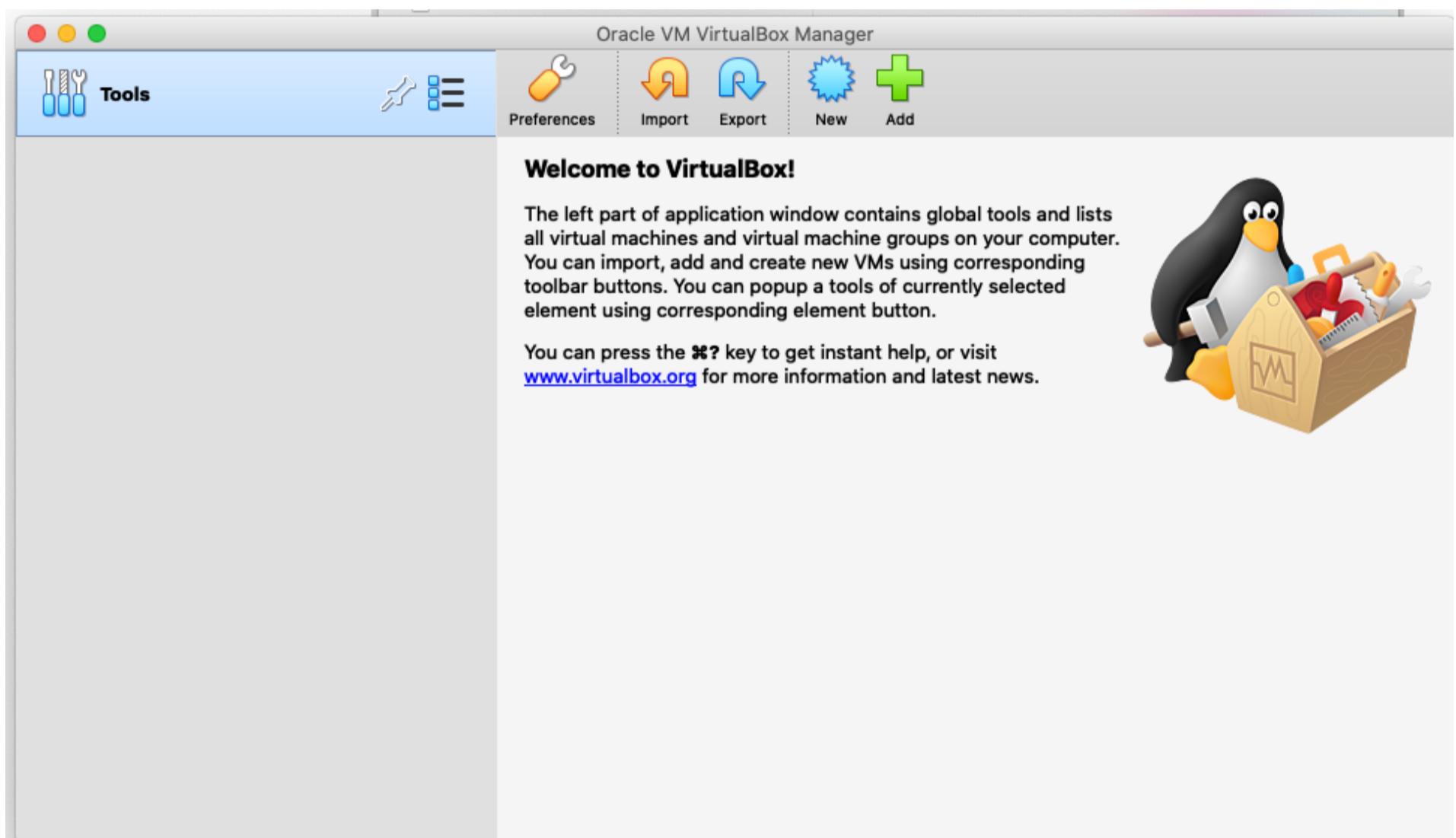
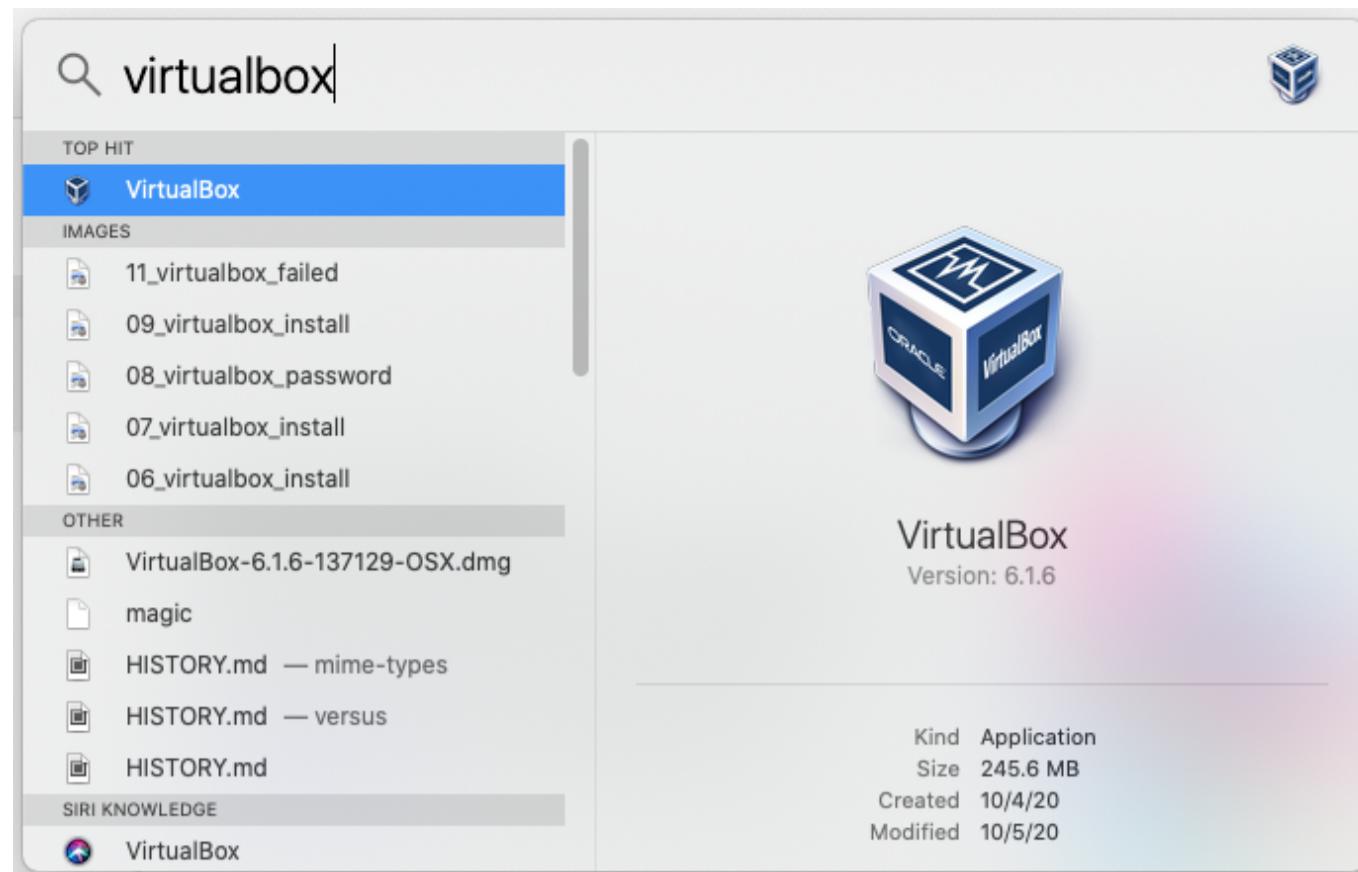
This **file is about 7GB** so may take a while. I've had no issues downloading it with Chrome, but you may want to use a download manager if your connection isn't very good.

## Step 3: Set Up the Oracle VM

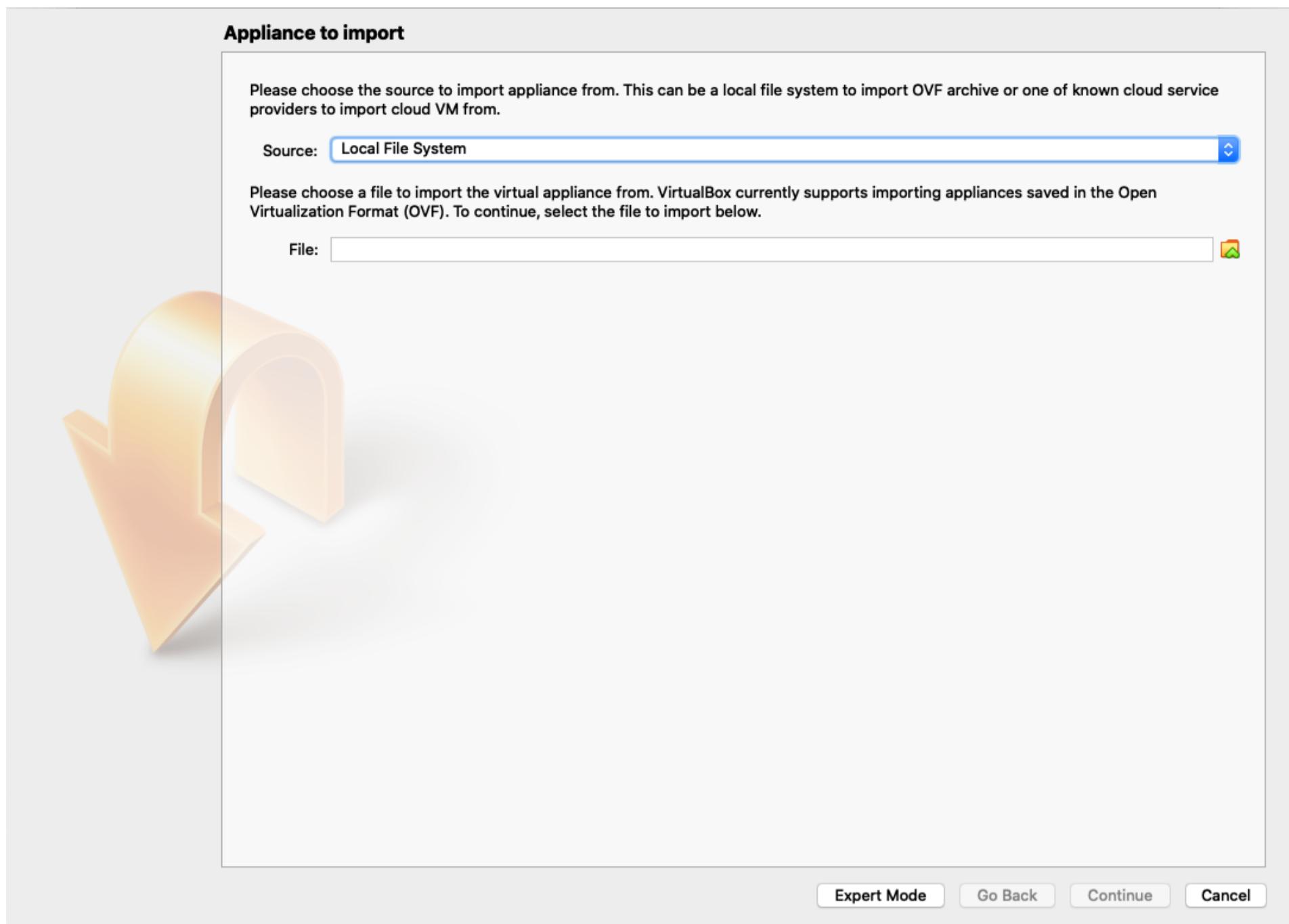
---

Now you have downloaded the Oracle VM file, it's time to set it up.

**Step 3A:** Open VirtualBox. I usually do this by pressing **Command + Space** and typing in VirtualBox.



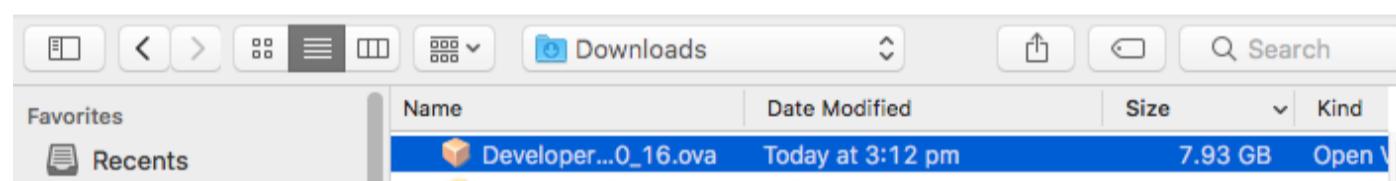
**Step 3B:** Click File, then click Import Appliance (or click the Import button at the top of the main screen). The Appliance to Import screen is displayed.



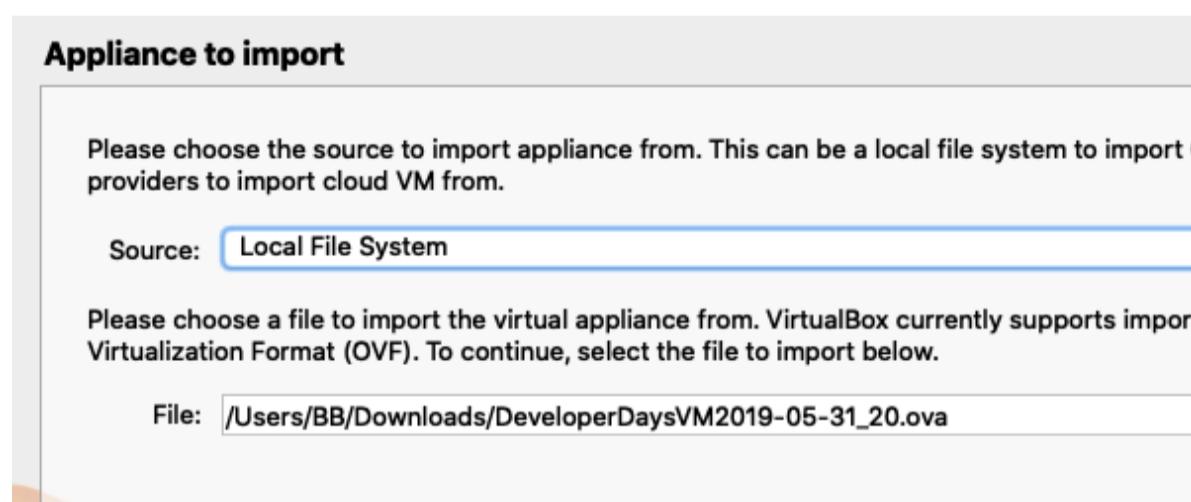
**Step 3C:** Click the Choose File button on the right of the textbox, which is the little yellow folder icon with the green arrow. (Leave the Source textbox as the default “Local File System”).

**Step 3D:** Select the DeveloperDays OVA file you have just downloaded, and click Open.

There's no need to extract the file before opening it.



**Step 3E:** Click the Continue button on the main installer.



**Step 3F:** Click Import on the Application Settings page.

## Appliance settings

These are the virtual machines contained in the appliance and the suggested settings of the imported VirtualBox machines. You can change many of the properties shown by double-clicking on the items and disable others using the check boxes below.

Virtual System 1

|                           |                                                                                          |
|---------------------------|------------------------------------------------------------------------------------------|
| Name                      | Oracle DB Developer VM                                                                   |
| Product                   | Oracle RDBMS 19.3, Application Express 19.1, REST Data Services 19.1, SQL Developer 19.1 |
| Product-URL               | <a href="http://otn.oracle.com">http://otn.oracle.com</a>                                |
| Vendor                    | Oracle                                                                                   |
| Vendor-URL                | <a href="http://www.oracle.com">http://www.oracle.com</a>                                |
| Version                   | April_2019                                                                               |
| Guest OS Type             | Oracle (64-bit)                                                                          |
| CPU                       | 1                                                                                        |
| RAM                       | 2048 MB                                                                                  |
| DVD                       | <input checked="" type="checkbox"/>                                                      |
| Network Adapter           | <input checked="" type="checkbox"/> Intel PRO/1000 MT Desktop (82540EM)                  |
| Storage Controller (IDE)  | PIIX4                                                                                    |
| Storage Controller (IDE)  | PIIX4                                                                                    |
| Storage Controller (SATA) | AHCI                                                                                     |
| Virtual Disk Image        | Oracle DB Developer VM-disk002.vmdk                                                      |
| Virtual Disk Image        | Oracle DB Developer VM-disk001.vmdk                                                      |

Machine Base Folder:  /Users/BB/VirtualBox VMs

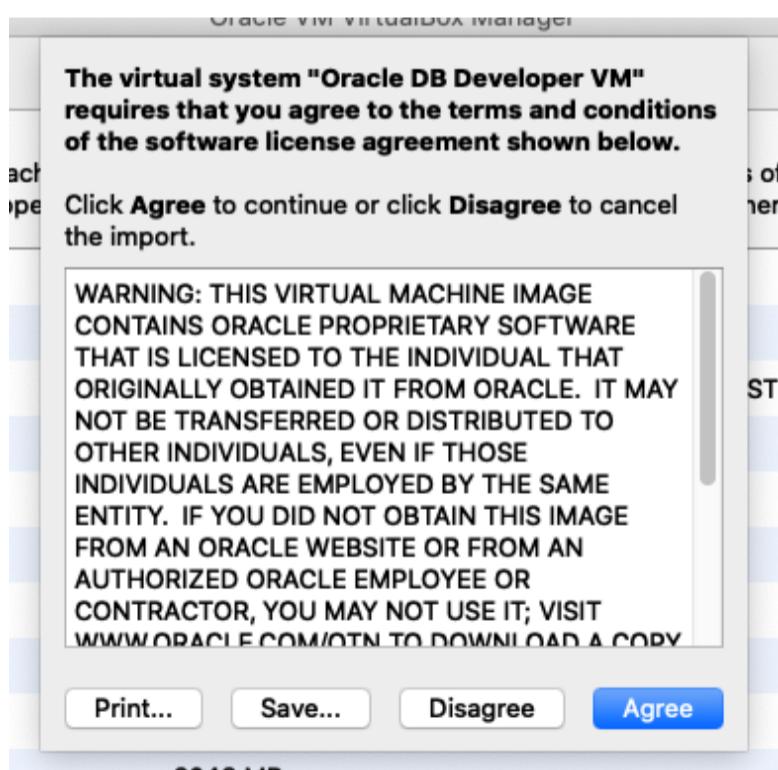
MAC Address Policy:  Include only NAT network adapter MAC addresses

Additional Options:  Import hard drives as VDI

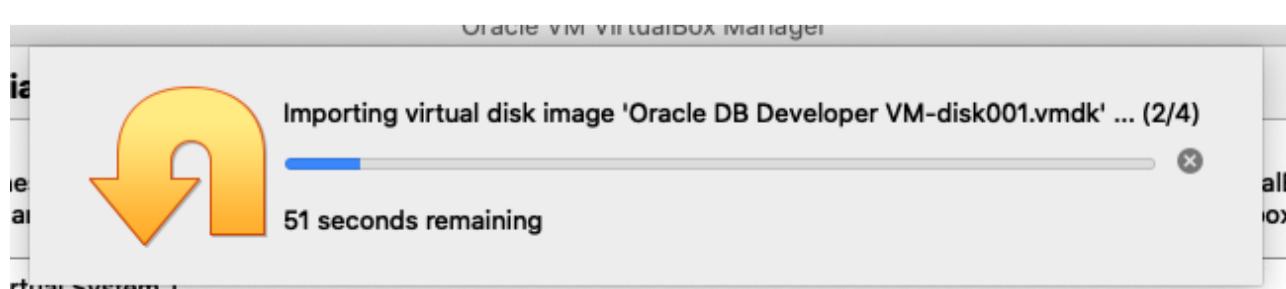
Appliance is not signed

Restore Defaults   Go Back   **Import**   Cancel

Step 3G: Click Agree on the popup that appears.

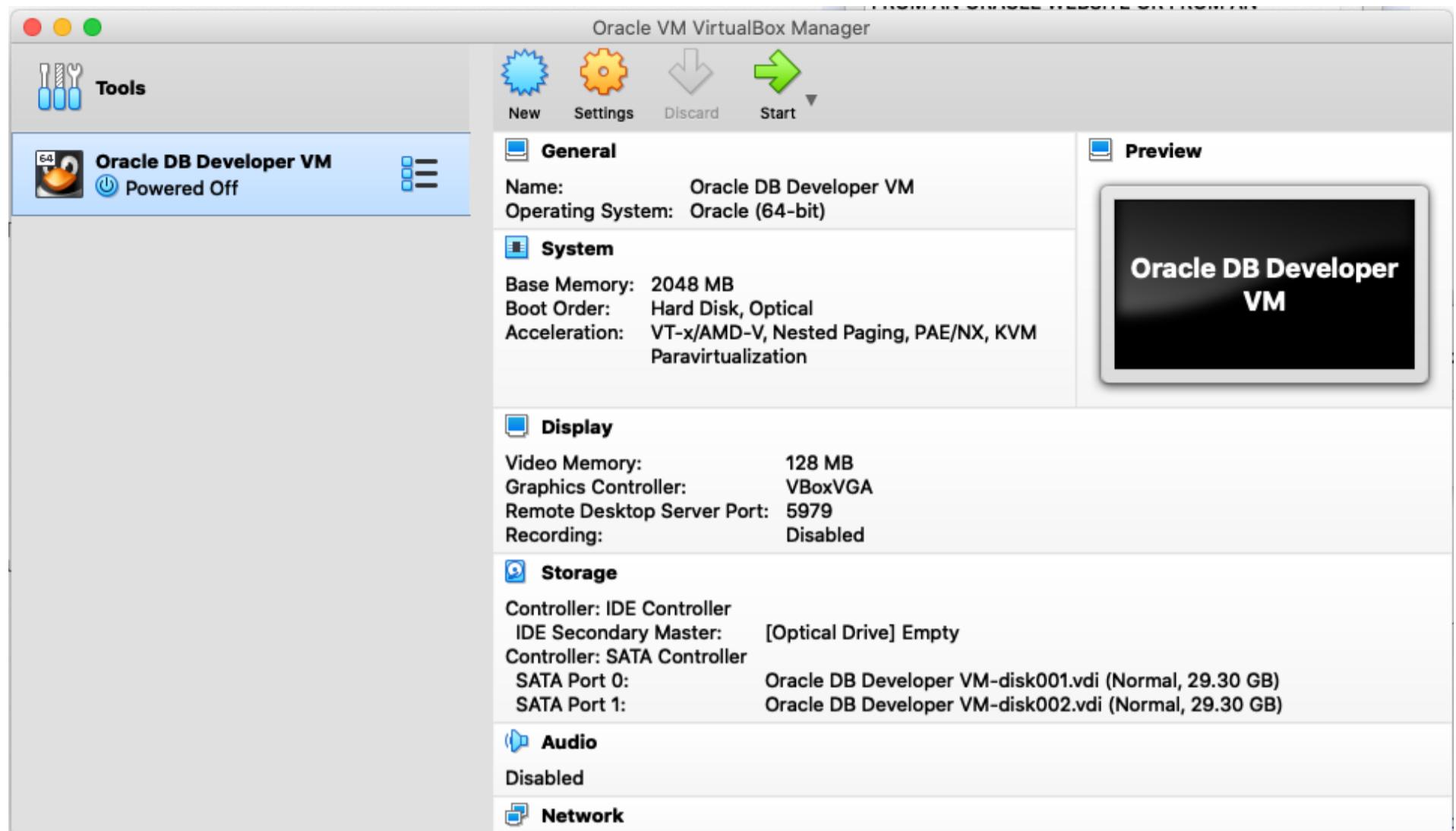


Step 3H: The import process will begin. This process took about 2 minutes for me.



Step 3I: Once the installer is finished, you should see "Oracle Developer Days (Powered Off)" on the left panel of the VirtualBox window.

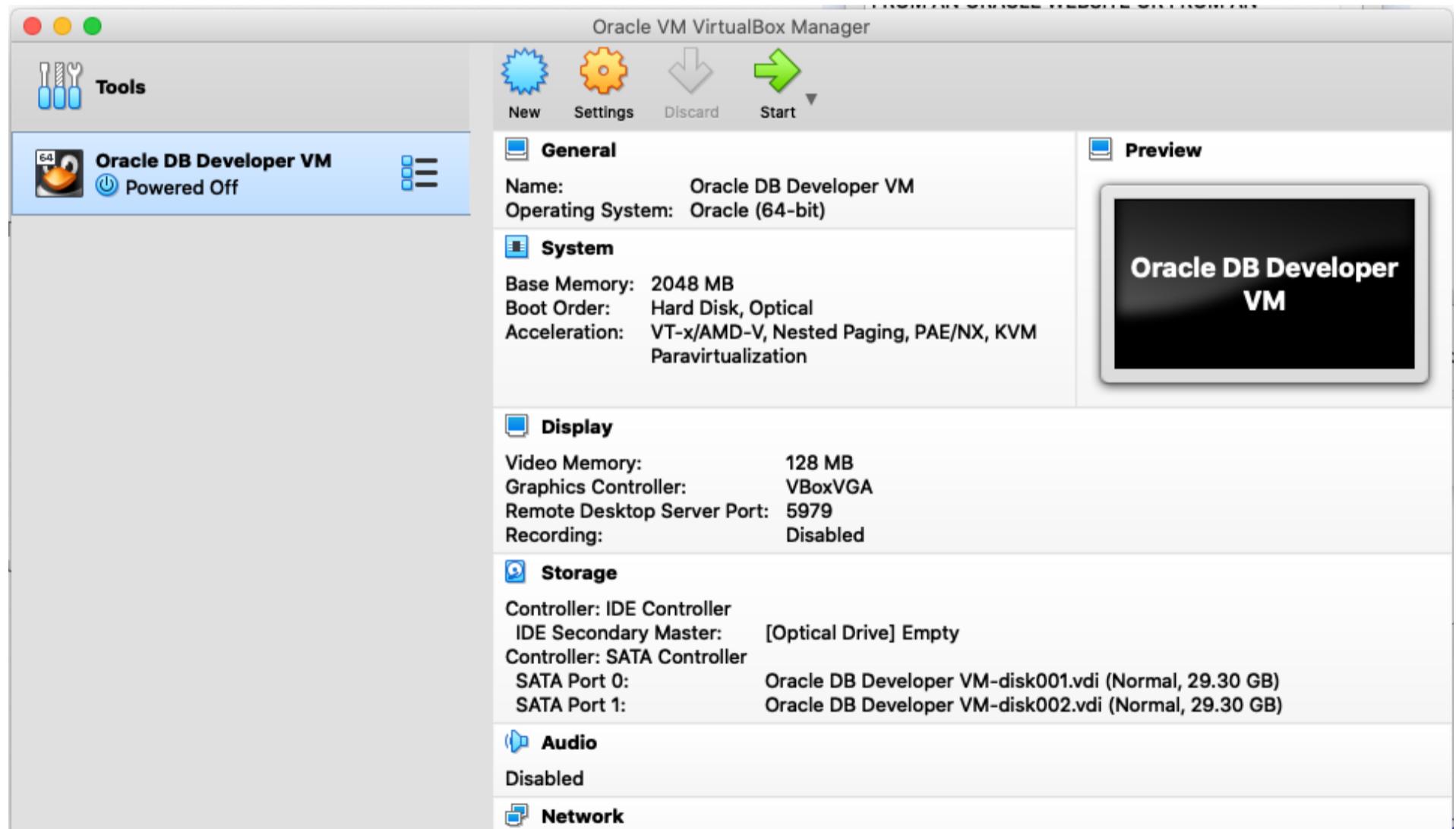
This means the VM is installed but not running.



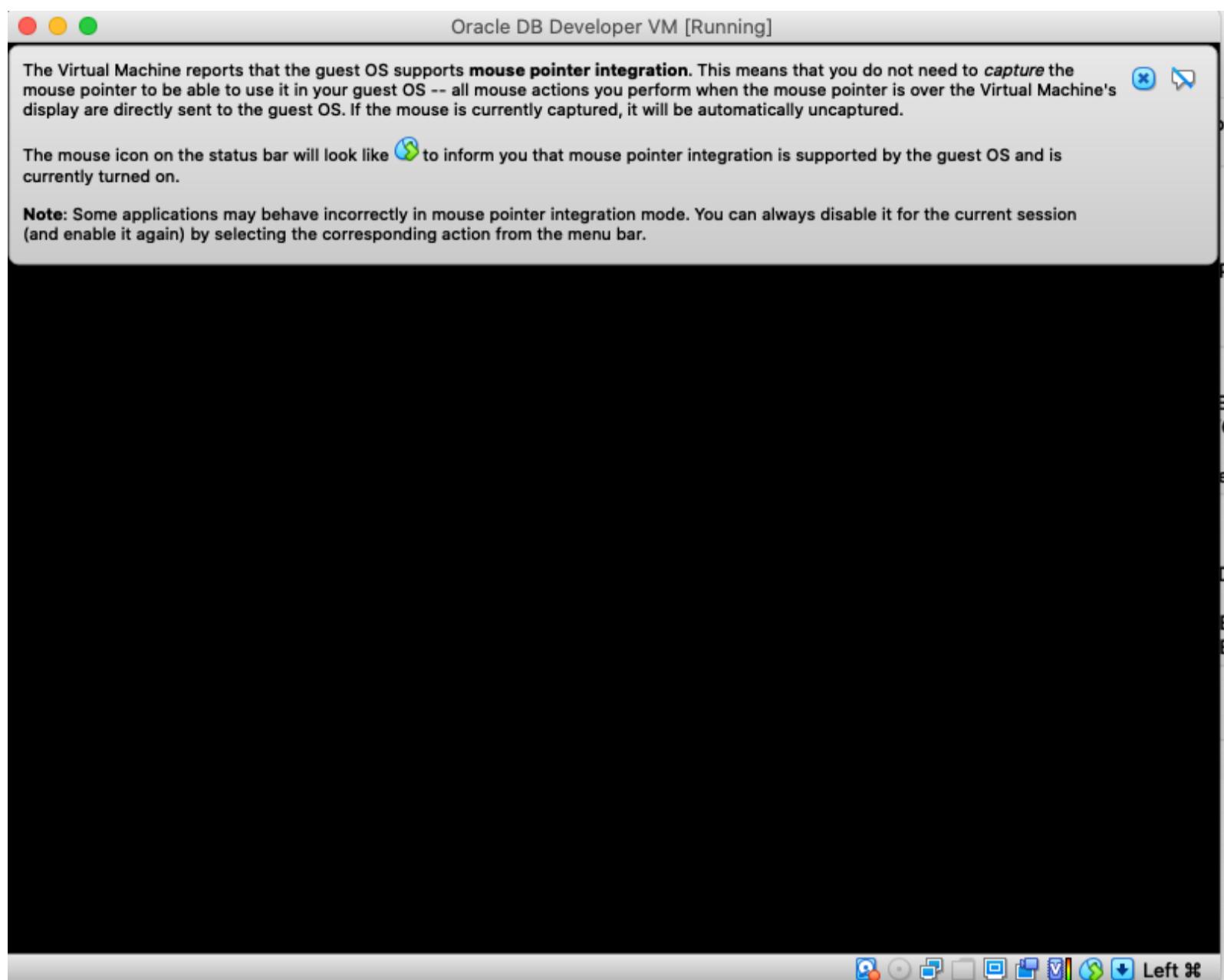
## Step 4: Run the Oracle VM

Now the installation has finished, it's time to run the Oracle VM.

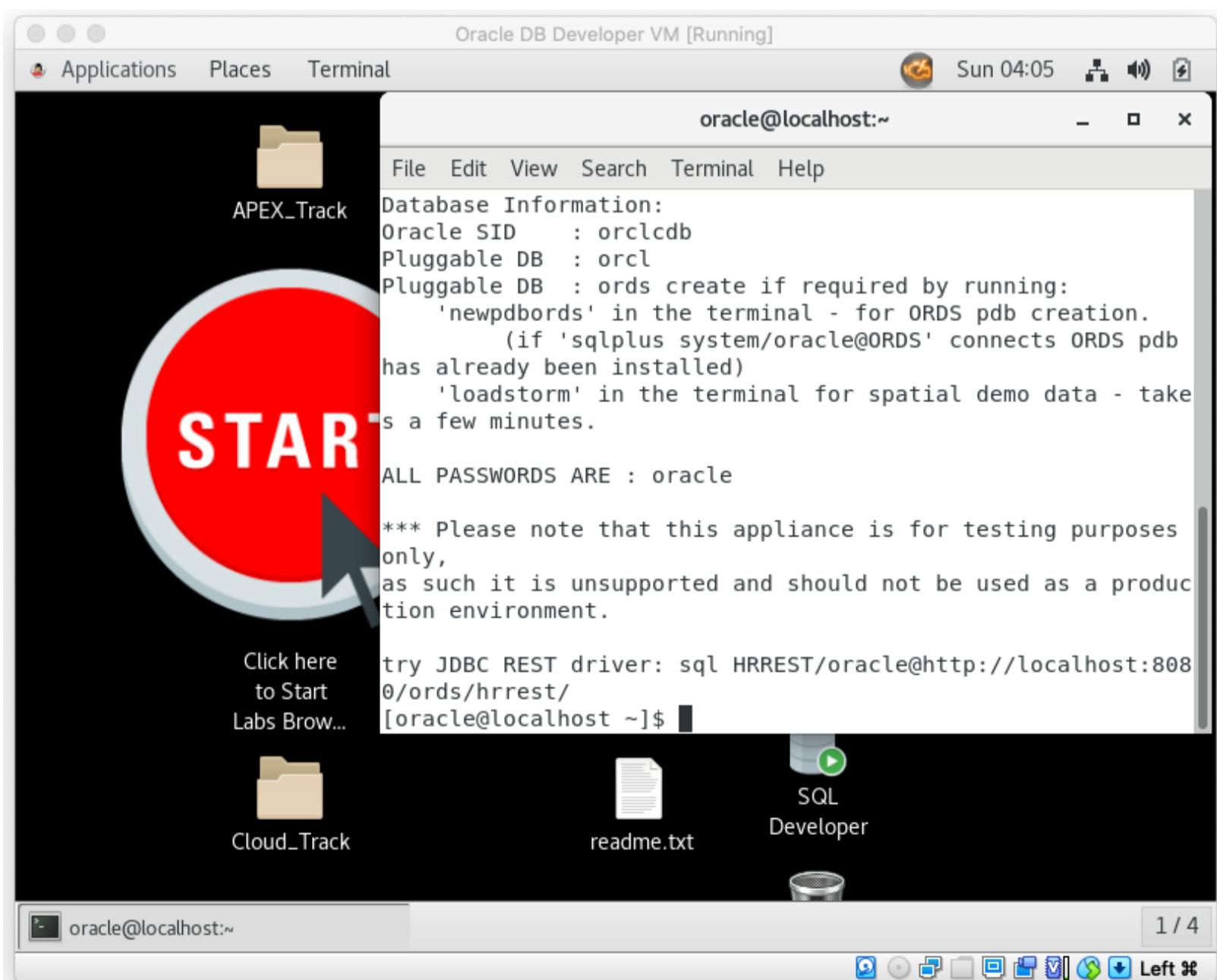
**Step 4A:** To run the Virtual Machine, click once to select “Oracle DB Developer VM” and click the green Start arrow.



There will be a minute or so of a black screen with white text as the virtual machine loads, and a black screen.



A Linux desktop will appear, and a moment later a terminal window will appear:



The virtual machine is now ready to use.

## A Note About Firewall and Port Forwarding

In previous versions of VirtualBox and this VM, we had to take extra steps to make the VM visible to the host (the Mac that you are running the VM on). We also had to set up port forwarding, which would allow us to enter in an IP address such as “localhost” on the Mac and have it forward to the Virtual Machine.

However, this configuration is already done in the latest version. So, these steps are no longer needed. Once the VM is running, as mentioned above, we can connect to it using SQL Developer (or another IDE).

I've removed these steps from the guide, but the comments below still refer to them.

Let's continue!

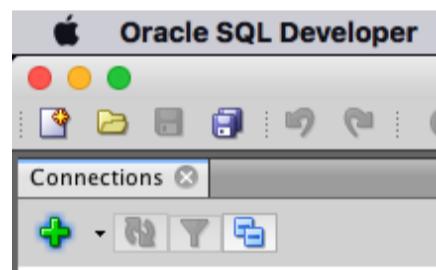
## Step 5: Test Using SQL Developer

---

The last step is to test this connection using SQL Developer.

**Step 5A:** Open SQL Developer on your mac. If you haven't downloaded it, you can download it from the Oracle website.

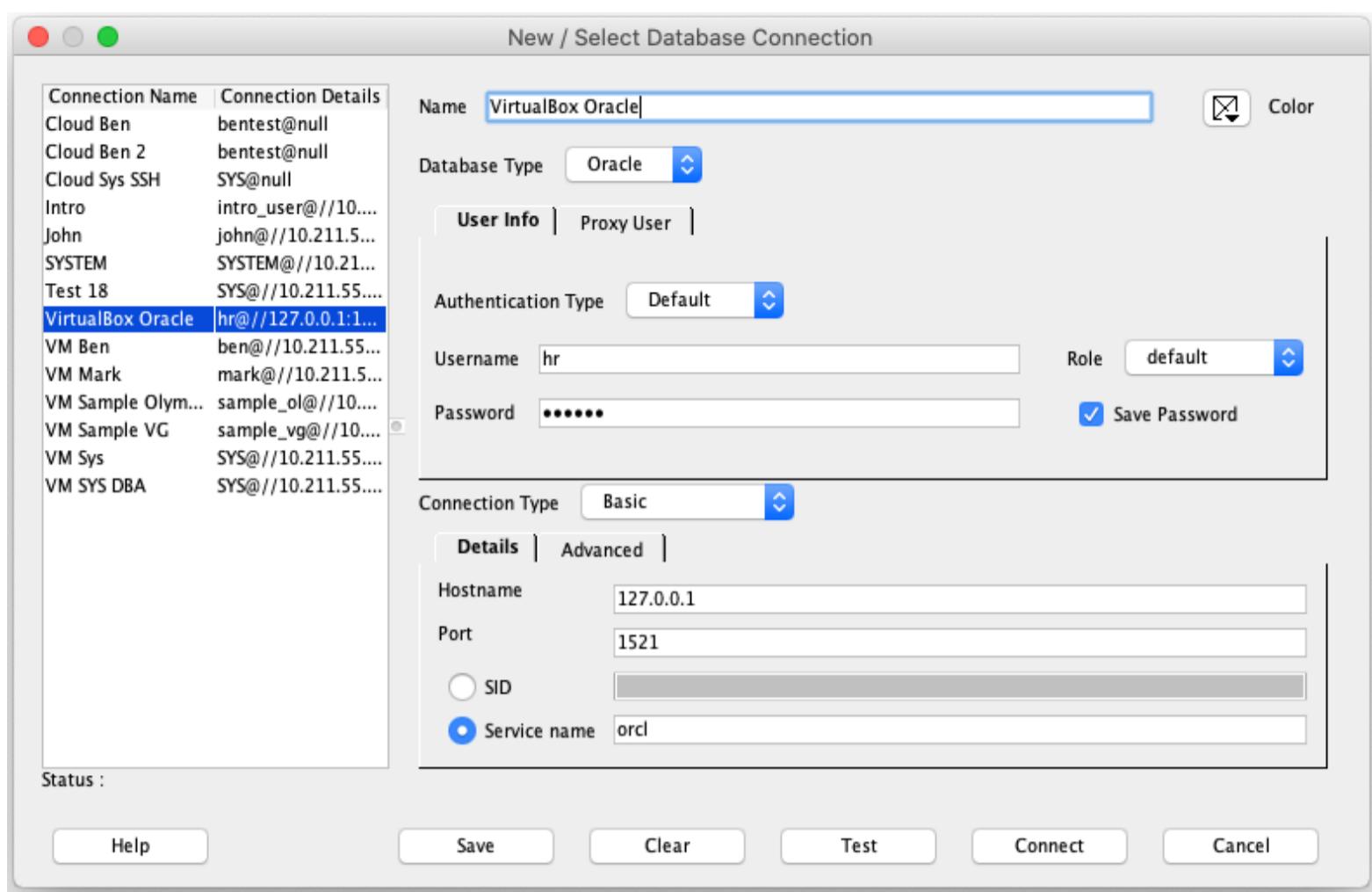
**Step 5B:** Create a new connection by clicking on the green + icon.



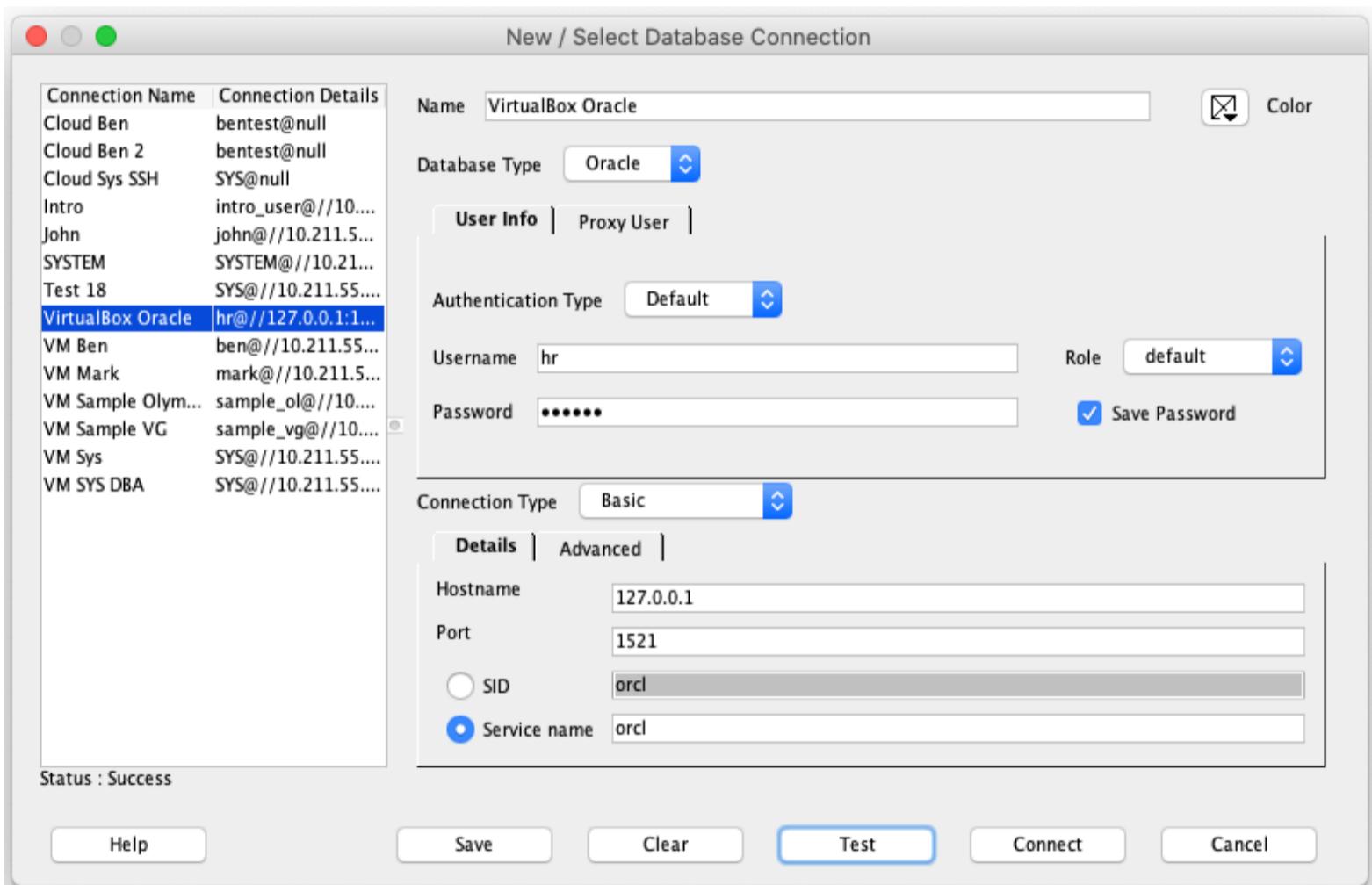
**Step 5C:** Enter these details for your connection:

- **Name:** whatever you like. A name such as “VirtualBox Oracle” is OK if you can’t think of anything.
- **Username:** hr
- **Password:** oracle
- Check **Save Password** if you don’t want to enter the password every time.
- **Hostname:** localhost
- **Port:** 1521
- Select “**Service Name**” instead of “SID”
- **Service Name:** orcl
- **Connection Type:** Basic

Your connection screen should look something like this:



**Step 5D:** Click “Test” to test the connection. The message should say “Success”, which means the connection has worked.



- If it says “Listener refused the connection with the following error: 12514”, then you likely have the Service Name incorrect.
- If it says “Listener refused the connection with the following error: 12505”, then you likely have selected SID instead of Service Name.
- If it says “Test failed: ORA0-1017: invalid username/password, logon denied”, then you likely have an incorrect username or password, or one of several other issues.

I've written more about [resolving Oracle connection issues here](#).

**Step 5E:** Click Save to save your connection in the list.

**Step 5F:** Click Connect. After a moment, a new SQL worksheet window is opened on this new connection.

You can now run SQL queries on Oracle on your Mac. This will work for as long as you have the Virtual Machine running in the background.

To test this, you can run this query:

```
SELECT * FROM v$version;
```

This will show the current version of the Oracle database.:

Your Mac can now access your Oracle database running on your own virtual machine!

## Shutting Down

---

Once you have finished with your Oracle SQL session, you can shut down your VM. To do this, go to **Machine > ACPI Shutdown**. The Virtual Machine will go back to a “Powered Off” when it’s done.

## Alternative to VirtualBox: Use Parallels

One alternative to this process of using VirtualBox and the Oracle VM is to use Parallels. Parallels is a VM application, just like VirtualBox. The key differences are:

- It has a different VM file format, so you can't use Oracle's VM files.
- It's a paid tool, unlike VirtualBox which is free.
- You can install your own operating system on it, so you can run Windows 10 for example, instead of Oracle Linux. VirtualBox allows this as well.
- You can install your own version of Oracle on there, such as Oracle Express or an earlier/later version of Oracle.

The steps are quite similar to setting up VirtualBox. To set up Parallels for an Oracle database on your Mac, the high-level steps are:

1. Purchase and download Parallels for Mac (or use the free trial).
2. Install Parallels.
3. Set up a new Virtual Machine on Windows 10. I don't know if you have to download the Windows 10 (or other Windows version) ISO file, or if Parallels can do that for you.
4. Once the operating system is installed, download Oracle or Oracle Express from the Oracle website, inside your virtual machine.
5. Install it on your virtual machine as though you're running it on a Windows computer.
6. Configure the port settings in a similar way as you did for VirtualBox
7. Connect to the virtual machine from SQL Developer on your Mac.

I've used Parallels in the past for this. The only reason I stopped was because my MacBook air didn't have the specs to keep up with running a VM.



### Download This Guide in a PDF

Save this guide as a PDF so you can refer to it later. All steps and screenshots included.

Email



DOWNLOAD

## Summary

Some developers want to run an Oracle database on a Mac. While Oracle is not natively available on a Mac computer, it's possible to run Oracle or Oracle Express on a Mac using a virtual machine.

This article describes the steps you need to take to set up the existing Oracle Developer VM and VirtualBox on your Mac to get it up and running.

## 59 Comments

**sushma** on March 19, 2019 at 12:52 am

[Reply](#)

Hi, I've followed the steps but when I tried to import .ova file, I'm getting below error. Do you know how can I resolve it?

Error reading OVA '/Users/username/Downloads/DeveloperDaysVM2018-10-16\_09.ova'  
(VERR\_TAR\_UNEXPECTED\_EOS).

Result Code:

VBOX\_E\_IPRT\_ERROR (0x80BB0005)

Component:

ApplianceWrap

Interface:

IAppliance {8398f026-4add-4474-5bc3-2f9f2140b23e}

**Ben** on April 1, 2019 at 4:42 am

[Reply](#)

Hi, I've never seen that error before, it might be something you need to Google or check with Oracle.

**Shweta** on October 7, 2019 at 9:09 am

[Reply](#)

Step 5: make visible on mac host

Not able to do please explain

Step 6: Setup port forwarding

On testing sql getting error please help me.

**Dipika Sharma** on April 14, 2019 at 10:11 am

[Reply](#)

When trying to test the database, I am getting this error.

Status: Failure-Test failed: Listener refused the connection with the following error: ORA 12514, TNS listener does not currently know the service request

**Ben** on April 19, 2019 at 9:03 am

[Reply](#)

Hi Dipika, I would suggest checking your connection parameters, especially the IP address of the virtual machine.

I've got an [article here](#) on this error but it may not solve the issue for connecting using a Mac.

**Sai** on January 28, 2021 at 3:21 pm

[Reply](#)

Hi,

I tried following the installation process on my New MacBook Air with M1 chipset. However , I got the following error while accessing oracle DB developer through VM manager :

Kernel driver not installed (rc = -1908) and it's failing to open a session for virtual Machine Oracle DB developer VM.

Kindly help

**M** on January 16, 2020 at 7:02 pm

[Reply](#)

Hello Dipika, i encountered this error as well. I am running 19c. When you start your oracle database on the virtual machine, it gives you some database information. Please check to see what those credentials are, they may be different from what was used above. Check specifically for these:

Oracle SID: sid\_here (Connect using SID instead of service name as described above)

Pluggable DB: check for the 'username/password@ORDS' credentials given in this section. you'll notice a place like ('if sqlplus username/oracle@ORDS' connects ORDS pdb has already been installed)

**Yonas** on April 14, 2019 at 2:06 pm

[Reply](#)

Hi,

While I am trying to import the OVA file on mac os, I get the below error

Failed to import appliance .../Downloads/DeveloperDaysVM2018-10-16\_09.ova.

Result Code: NS\_ERROR\_INVALID\_ARG (0x80070057). Any thing I can do to resolve it?

Thanks

**Ben** on April 19, 2019 at 9:01 am

[Reply](#)

Hi Yonas, I haven't seen that error before unfortunately. I would suggest Googling the error – perhaps others have gotten it. If that doesn't give you a solution, you can try the [Oracle forums here](#).

**Matt** on May 13, 2019 at 6:45 am

[Reply](#)

Hi

Thanks for your write up.

I'm on a Mac on Mojave 10.14, I can get the Oracle DB Developer VM running but I can't past step 5.

I have no admin setting to reach the firewall settings, port etc?

Thanks

**Ben** on May 18, 2019 at 3:34 pm

[Reply](#)

Hi Matt,

Good question. I'm not sure why the Admin section would not be appearing. I remember having this issue a while ago but I don't remember how I resolved it. I admin setting up a VM for this is a bit tricky so I would suggest Googling why the Admin menu does not appear.

Sorry I couldn't be more helpful!

Ben

**Digitalman42** on June 12, 2019 at 10:36 pm

[Reply](#)

On an enterprise (corporate) system, the MacOS might be locked down from the user and is denied permission to get to the admin section. Unfortunately, this is pretty common. It would help if you mentioned that it was the admin section of MacOS. It's confusing to direct us to the VM and then switch gears to the MacOS without any indication that you're doing this. Otherwise, great and helpful article.

**Mike** on August 29, 2019 at 4:50 am

[Reply](#)

Did you resolve this? I am stuck too. He said system menu, admin then firewall but none of those options are there. I don't see the system menu. I click settings but there was no admin tab. Any help?

**John** on May 14, 2019 at 10:09 pm

[Reply](#)

Hi Ben,

Thanks so much for this article. It is great help.

It appears Oracle has come out with a new release of VirtualBox (6.0) since you wrote this article which I have installed. The interface is slightly different than you describe. I am working on the step to set up the firewall rule. In the upper right hand corner of the 6.0 interface is a icon (which I might label a 3 server network icon). When clicking on it a drop down displays, with a variety of options including a wrench/tools icon. When I click on that, I get to a screen with a Wifi option. It displays a message, No Wi-Fi Adapter Found. I am able to open up Firefox and surf around on the VM so must have WiFi connectivity, but still have that message.

In the upper left hand corner of the VM, is a menu labeled Applications. Underneath it, is a submenu named Sundry which has an option for Firewall. When I open it up I get a message: Trying to connect to firewalld, waiting...

A quick Google search about the No Wi-Fi Adapter Found returned some entries from 2013 about installing an extension to VirtualBox. Before I go down that path, I thought I would ask you if that is what you recommend or maybe I missed something.

Also, I noticed on my virtual machine the command ipconfig is not installed. On my MacBook, it is in /usr/sbin, but nowhere to be found on the VM. Any thoughts on that?

Thanks again. This has been a great start for me.

**Ben** on May 18, 2019 at 3:36 pm

[Reply](#)

Hi John,

Glad you like the article.

Thanks for the question. Yeah it looks like a recent version has come out since writing this. I've only set up this VM once or twice so I'm not that familiar with all of the issues or intricacies of it.

It's strange that the icon is different and the firewall setup is not working.

I would suggest asking a question on AskTOM – that's a support forum for Oracle issues. They may be able to advise what the issue is or at least advise what else you can try.

Sorry I couldn't be more helpful!

Thanks,

Ben

**Leonardo Forero** on September 26, 2019 at 5:03 am

[Reply](#)

Hi,

You need to start the firewall using the terminal.

Follow this steps

<https://www.liquidweb.com/kb/how-to-start-and-enable-firewalld-on-centos-7/>

**George E** on May 16, 2019 at 3:29 am

[Reply](#)

Hello,

Your steps have worked just fine for me thus far (up to Step 5, haven't tried this yet). This is by far the simplest tutorial on running Oracle on Mac that I've found.

I notice SQL Developer runs a little slower in the VM (Oracle Live SQL is worse). I suspect Step 5 may make things better but my question is in regards to the safety of it. This is a little new to me so I have some questions. Does opening ports expose my Mac to security issues, is this port only open when VirtualBox is running? Is there a way to make this more secure for peace of mind? I guess I'm just concerned with how safe making the VM visible to my Mac would be.

Thanks so much for your detailed tutorial.

**George E.** on May 16, 2019 at 4:45 am

[Reply](#)

Also, when logging into SQL Developer and it says there's an updated version, if I click on that link it will open the VM Firefox browser and take me to the download page from the Oracle site. Can we actually update our version of SQL Developer this way or is that not possible since it's still technically running on my Mac.?

**Ben** on May 18, 2019 at 3:40 pm

[Reply](#)

Hi,

Are you opening SQL Developer inside your VM or on the Mac?

I think you can update it either way – the version on the VM or the version on the Mac.

I don't use the version inside the VM, but if I did, I think it would update OK.

If you're not sure, you can manually download it from the Oracle website and install it.

**Ben** on May 18, 2019 at 3:38 pm

[Reply](#)

Hi George,

Thanks, glad you like the article!

I'm not sure about the security issue behind it. Networks and ports is not a strong point for me. I would suggest asking the AskTOM forums, they can probably advise on the impacts I think.

What are the specifications of your Mac? I ran a VM on a 2015 MacBook Air and it ran quite slowly, but on a 2018 MacBook Pro it's running very well.

Thanks,

Ben

**Digitalman42** on June 12, 2019 at 11:10 pm

[Reply](#)

Nice article! Helped me a lot. Needs to be updated to the current version of the VM and some extra screenshots would be nice. I also found that the auth should be system/oracle and service name=orcl. Also, the port forwarding was already set up in the VM, so no need to actually do this step, but nice to have it document in case it gets deleted. Also don't need the IP for it, just leave it blank. Thanks for posting! I'm happily connected to my VM with Sql Developer on my Mac. Good times!

**Ben** on June 13, 2019 at 5:01 am

[Reply](#)

Thanks! I'll update the article with those points and look to get new screenshots. Glad you found it useful!

**Ron** on July 19, 2019 at 7:43 am

[Reply](#)

You filled in/clarified the missing info. Thanks! Great article Ben. Just needs a couple of updates to align with latest version of VM. As mentioned by Digitalman42, it's actually a bit easier now with some of the configuration already installed.

**TJ** on July 24, 2019 at 5:06 pm

[Reply](#)

Hi, At step 5 when I'm trying to open firewall configurations it is giving me an error stating – "Failed to connect to firewalld. Please make sure that the service has started correctly and try again."

Thank you so much for the tutorial and I'd appreciate any HELP on resolving the ERROR I got.

Tj

**Ben** on July 25, 2019 at 5:24 am

[Reply](#)

Hi Tj,

I'll take a look at the error, however it's not something I've seen before. It might be easier for you to Google it (if you haven't done so already) as others may have experienced it.

Ben

**Santosh** on September 27, 2019 at 3:16 am

[Reply](#)

Hi,

Checkout the below link. It will help.

<https://oracle-base.com/articles/linux/linux-firewall-firewalld>

Santosh.

**Laura Prado** on September 12, 2019 at 11:19 pm

[Reply](#)

thanks so much for this tutorial, it worked like a charm. I have the same issue with configuring the firewall and opening ports, as mine shows as having a wired connection, but it's fast enough using SQL dev within the VM, so I'm ok with that. Anyway, just wanted to say I really appreciate you taking the time to do this!

**kadir malak** on October 1, 2019 at 10:43 pm

[Reply](#)

Hi, I've used ORCL as service name and it worked!

**Pankaj Pilkhwal** on October 7, 2019 at 4:24 pm

[Reply](#)

System configurations

macOS Mojave – Version 10.14.6

VirtualBox – Version 6.0.12 r133076 (Qt5.6.3)

---

I am getting the below errors once I click on the VM

Error1

Failed to open a session for the virtual machine Oracle DB Developer VM.

Error2

Kernel driver not installed (rc=-1908)

Make sure the kernel module has been loaded successfully.

where: suplibOsInit what: 3 VERR\_VM\_DRIVER\_NOT\_INSTALLED (-1908) - The support driver is not installed. On linux, open returned ENOENT.

**Pankaj Pilkhwal** on October 7, 2019 at 4:57 pm

[Reply](#)

The issue was with the VirtualBox version. Installed Version 5.2.32 r132073 (Qt5.6.3) to fix the issue

**JinGu** on January 7, 2020 at 11:29 am

[Reply](#)

Can you please tell me more detail about it?

**Ben** on January 7, 2020 at 12:31 pm

[Reply](#)

What would you like to know more about, JinGu?

**JinGu** on January 7, 2020 at 5:28 pm

[Reply](#)

Hi ben

Can you help me check this error

Status : Failure -Test failed: IO Error: Connection reset by peer,  
Authentication lapse 0 ms.

**JinGu** on January 7, 2020 at 5:31 pm

[Reply](#)

Connecting sql developer mac to Oracle database

**Trent** on May 27, 2020 at 8:41 am

[Reply](#)

Thank you so much Ben! I kept wondering why it didn't work on my mac, and then i realized that it wasn't made to work on macs, UNLESS we use your process. You are such a huge relief.

**Ben** on May 29, 2020 at 5:29 am

[Reply](#)

Glad you found it useful!

**Patrick Wadkins** on May 28, 2020 at 9:32 am

[Reply](#)

This post was very helpful and the steps were easy to follow and install each component. My environment works correctly; however, when I try to install sample schemas from the Oracle site, I am having problem with the users being setup. I don't seem to have access to create these schemas. Probably user error, but trying to figure out how to get these sample schemas created.

<https://docs.oracle.com/en/database/oracle/oracle-database/18/comsc/installing-sample-schemas.html#GUID-B0BEE222-D8B0-4B68-B359-DEA153956EF6>

**swornim maharjan** on June 18, 2020 at 6:53 pm

[Reply](#)

ORA-12162: TNS:net service name is incorrectly specified

What should I do...?

**ririn** on August 20, 2020 at 1:21 pm

[Reply](#)

Hi Ben, i'm a new mac user. I need an oracle jinitiator for my mac. Would you tell me how to install it? I really need ur help :"

**Ben** on August 27, 2020 at 5:12 am

[Reply](#)

Hi Ririn, I've never heard of Oracle Jinitiator. Have you tried the instructions that Oracle provide?

**Ghadi** on September 8, 2020 at 3:44 pm

[Reply](#)

Error report –

ORA-01031: insufficient privileges

01031. 00000 – “insufficient privileges”

\*Cause: An attempt was made to perform a database operation without the necessary privileges.

\*Action: Ask your database administrator or designated security administrator to grant you the necessary privileges

what should i do?

**Brandon** on September 20, 2020 at 4:40 pm

[Reply](#)

Thanks for this guide. I was able to perform all the steps and the test using SQL developer worked. My confusion is related to my classwork for a course I'm taking on database systems using Oracle.

I'm completely new at this, so my questions or confusion may not even make sense, sorry in advance, but hopefully I can get some guidance.

My class instructs me to download 11g Express Edition from Oracle (which I obviously can't run on my mac, which is why I am here). This guide seems to indicate I can do what I need using this method, and my instructor said that students had used a VM with some success in the past. Part of my confusion is understanding the difference (and how they relate) between 11g Express and SQL Developer. I think that 11g is the database, and SQL is a way to access and interact with it?

What is throwing me off is your example to test using SQL and using the hr username and oracle password, is this simply a sample database? How do I set up my own database on my Mac to do my coursework?

This is where the dots aren't connecting for me. I can't tell if what I am trying to do is even possible, or if I am just missing something. My only other option to use an old PC I have access to, but this is less than ideal. I would prefer to run everything out of my Mac if possible.

Thanks in advance.

**Ben** on September 25, 2020 at 8:35 am

[Reply](#)

Hi Brandon, thanks for the questions.

Oracle Express is the database itself, which is the software and files that store all of the tables and data and everything that makes the database work. Oracle Express is a smaller but easier to set up version of Oracle's full database, and it's great for getting started. Oracle Express 11g was the most recent version of Oracle Express up until a year or so ago, and now I think the latest version is Oracle Express 19c.

Oracle SQL Developer is an application that lets you access and interact with the database, as you mentioned. You'll need Oracle Express to work with the database, but you can use another application instead of SQL Developer if you like. SQL Developer is quite easy to use, and free, and is often recommended by those working with Oracle (myself included).

The difference with my article and what you're referring to is that my article uses a VM that's provided by Oracle. Their VM includes the full Oracle Database (not Oracle Express), and this database includes a sample database called "hr". I suggest testing with this because if you're able to connect to this "hr" database that's inside the VM, then the setup process has worked.

The good thing is that once this has been done, you can then set up your own database and tables and use that instead of the "hr" database.

So, in short, if you can complete everything in this article and get the VM working on your Mac, you should then be able to create the database you need. Hope that helps. Let me know if you have any other questions.

**Fisayo Nana** on October 9, 2020 at 7:15 am

[Reply](#)

Hello Ben,

Your article is amazing and it worked perfectly for me too.

However, I am struggling with creating my own database, please can you share a guide on how to do this properly after VM is working without using the "hr" database

**Tanya** on October 13, 2020 at 7:40 am

[Reply](#)

Hi,

I installed this and it's ok but I need from Northwind database because my SQL course works with this database. Please, can you help me with this!

**Ben** on October 16, 2020 at 4:17 am

[Reply](#)

Hi Tanya, I don't think Oracle has released a Northwind database as it's an SQL Server data set. But if you have the SQL scripts to create it from SQL Server you can convert it to Oracle by changing some of the data types and syntax I think.

**Jose** on October 24, 2020 at 11:41 pm

[Reply](#)

Great tutorial :) Thanks a lot, everything works as I expected :)

**Thomas** on November 6, 2020 at 8:14 pm

[Reply](#)

How can I create a new user and database/schema?

**Cynthia** on November 12, 2020 at 7:23 pm

[Reply](#)

Hi, I've installed the VM on my MAC and when I tried to run the Oracle DB Developer, I have a failure :

Kernel driver not installed (rc=-1908)

Make sure the kernel module has been loaded successfully.

where: suplibOsInit what: 3 VERR\_VM\_DRIVER\_NOT\_INSTALLED (-1908) - The support driver is not installed. On linux, open returned ENOENT.

Can you help me please ?

**Ben** on November 16, 2020 at 2:50 pm

[Reply](#)

Hi Cynthia, I'm not sure about that error. I would consider Googling it (if you haven't done so already).

**Kamilka** on November 30, 2020 at 3:03 pm

[Reply](#)

I just want to say thank you so so so much!! the article is so insanely helpful, everything worked for me perfectly, I appreciate your help!

**Kamila** on December 2, 2020 at 9:55 am

[Reply](#)

"You must specify a machine to start, using the command line.

Usage: VirtualBoxVM –startvm <name|UUID>

Starts the VirtualBox virtual machine with the given name or unique identifier (UUID)."

For some reason I'm getting this message after I shut down the Virtual Box and I can not do anything now. do you guys know how to solve this problem ?

**Steve** on January 14, 2021 at 1:55 am

[Reply](#)

Followed your instructions to the letter and everything seems to work perfectly. Thanks!!

**Steve** on January 14, 2021 at 1:56 am

[Reply](#)

Forgot to say, I'm using DBeaver SQL client rather than Oracle SQL Developer and this works great with Oracle VM.

**Robert** on January 15, 2021 at 7:42 pm

[Reply](#)

The Open SQL Developer won't start on my Mac(BookPro 2020 – Catalina) – I used DBeaver (<https://dbeaver.io/download/>) instead which worked out pretty well besides – thank you for this awesome HowTo

**santosh kumar choudhary** on January 24, 2021 at 2:21 am

[Reply](#)

Excellent blog, I followed steps and am able to install oracle and connected successfully from mac.

Thanks for such wonderful detailed and needed guid!

**shoug** on February 16, 2021 at 10:01 pm

[Reply](#)

I am stuck at step " Step 3D: Select the DeveloperDays OVA file you have just downloaded, and click Open." when I try opening it, it's all grey meaning that they are not compatible maybe? I don't know what the problwm is do i need an application to open it?

**Ben** on February 17, 2021 at 3:02 pm

[Reply](#)

Hi, I think the OVA file is opened by VirtualBox. I think you can open VirtualBox and follow the Import process to import this OVA file to start the virtual machine.

**Zhanserik Izmakov** on March 31, 2021 at 12:22 am

[Reply](#)

Hello, I do not start the Oracle DB Developer virtual machine and I also have the second Windows 10 operating system may not be started because of this

This site uses Akismet to reduce spam. [Learn how your comment data is processed.](#)

Ben Brumm  
DatabaseStar

Database Star Academy:

[Login](#)

## Popular Posts

[Get my book: Beginning Oracle SQL for Oracle Database 18c](#)



[Terms of Service](#)    [Privacy](#)

# hr 계정

