

# 데이터베이스

- 순서 : Ch3-1( SQL\*Plus)
- 학기 : 2018학년도 2학기
- 학과 : 가천대학교 컴퓨터공학과 2학년
- 교수 : 박양재

# 데이터베이스

- 목차

## 3.1 SQL\*Plus

## 3.1 SQL\*Plus

- ❑ 오라클 설치 시 **sys, system**의 암호를 설정하고, **scott** 계정의 잠금을 해제하고 암호를 **tiger**로 설정한다.

만일 잠겨 있을 경우 : **system** 계정으로 접속 후 해제한다.

SQL>ALTER USER SCOTT ACCOUNT UNLOCK;                      또는

SQL>ALTER USER SCOTT IDENTIFIED BY tiger ACCOUNT UNLOCK;

- ❑ 오라클 사용자들

- ✓ SYS : DBMS의 데이터 사전 소유자, 오라클 데이터베이스 관리자(super user), ~~디폴트 암호 : change\_on\_install~~, DB생성가능
- ✓ System : SQL\*Forms등의 툴을 위한 데이터사전 소유자, 모든 권한이 sys와 동일하나 DB생성권한이 없다. ~~디폴트 암호:manager~~
- ✓ scott : sample 사용자 계정, 디폴트 암호 : tiger

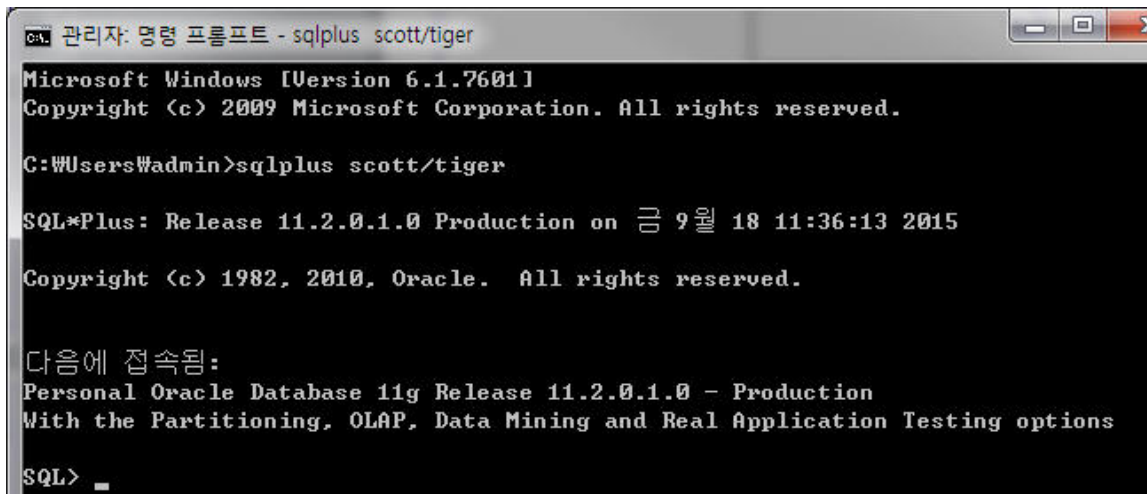
## 3.1 SQL\*Plus

### ❑ SQL\*Plus란?

- ✓ 오라클 데이터베이스 서버와 연결되어 SQL문을 저장하거나 편집하고, 실행하기 위해서 제공되는 도구
- ✓ 컬럼이나 데이터의 출력 형식을 설정하거나 환경을 설정하는 기능

### ❑ SQL\*Plus 로그인

- ✓ Command 환경에서 SQL\*Plus 로그인( c:\W> sqlplus scott/tiger)



```
관리자: 명령 프롬프트 - sqlplus scott/tiger
Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation. All rights reserved.

C:\Users\Wadmin>sqlplus scott/tiger

SQL*Plus: Release 11.2.0.1.0 Production on 금 9월 18 11:36:13 2015

Copyright (c) 1982, 2010, Oracle. All rights reserved.

다음에 접속됨:
Personal Oracle Database 11g Release 11.2.0.1.0 - Production
With the Partitioning, OLAP, Data Mining and Real Application Testing options

SQL> _
```

## 3.1 SQL\*Plus

3가

### ❑ 명령 프롬프트 환경 설정하기

- sqlplus 명령어 화면 설정법 ( )

```
SQL>set linesize 300
```

```
SQL>set pagesize 100
```

- 고정 설정법

C:\WDBNOTE\product\11.2.0\dbhome\_1\sqlplus\admin\glogin.sql 파일의 마지막 부분에 입력

```
set linesize 300
```

```
set pagesize 100
```

- cmd 옵션 변경

속성 → 레이아웃 → 화면 버퍼 크기(너비), 창크기(너비) // 115

## 3.1 SQL\*Plus

### ❑ SQL\*Plus 로그인 작업 디렉토리 만들기

✓ Command 환경에서 작업 디렉토리 만들기

① mkdir c:\wora\_work

② cd c:\wora\_work

③ c:\wora\_work>sqlplus scott/tiger

```
C:\Users\wadmin>mkdir c:\wora_work
```

```
C:\Users\wadmin>cd c:\wora_work
```

```
c:\wora_work>sqlplus scott/tiger
```

```
SQL*Plus: Release 11.2.0.1.0 Production on 금 9월 18 11:40:12 2015
```

```
Copyright (c) 1982, 2010, Oracle. All rights reserved.
```

```
다음에 접속됨:
```

```
Personal Oracle Database 11g Release 11.2.0.1.0 - Production
```

```
With the Partitioning, OLAP, Data Mining and Real Application Testing options
```

```
SQL>
```

## 3.1 SQL\*Plus

### □ SQL\*Plus 사용하기

- ✓ Command 환경에서 직접 SQL명령 사용하기(한 명령씩 사용하기)

```
SQL> select  table_name from user_tables; < -      ( ?)

TABLE_NAME
-----
SALGRADE
BONUS
EMP
DEPT

SQL>
```

- ✓ Command 환경에서 직접 SQL명령 사용하기(두 명령 이상 사용할 때는 파일명.sql 만들기)
- ✓ ED(EDITOR)-NOTEPAD.EXE에서 여러 명령어를 나열하여 일괄처리가 가능하다. batch process

## 3.1 SQL\*Plus

### □ SQL\*Plus 사용하기

- ✓ 원본 테이블은 복사해서 사용한다.

```
SQL> SELECT TABLE_NAME FROM USER_TABLES;  
  
TABLE_NAME  
-----  
SALGRADE  
BONUS  
EMP  
DEPT  
  
SQL>  
SQL> CREATE TABLE SALGRADE1 AS  
2  SELECT * FROM SALGRADE;  
  
테이블이 생성되었습니다.
```

- ✓ SALGRADE→SALGRADE1, BONUS→BONUS1, EMP→EMP1,  
DEPT→DEPT1으로 복사본을 사용해서 실습한다.



## 3.1 SQL\*Plus

### ❑ SQL\*Plus 사용하기

- ✓ Command 환경에서 직접 SQL명령 사용하기(두 명령 이상 사용할 때는 파일명.sql 만들기)

The screenshot shows a Windows environment. In the background, a Windows Explorer window displays the 'ora\_work' directory on the C: drive, containing a file named 'test.sql' modified on 2015-09-18. In the foreground, the SQL\*Plus command window is open. The prompt is 'SQL> ed test', and the editor shows the contents of 'test.sql':  
SELECT TABLE\_NAME FROM USER\_TABLES;  
SELECT \* FROM DEPT;  
Below this, the prompt is 'SQL> @test', and the output of the first query is displayed as a table:

TABLE_NAME		
-----		
SALGRADE		
BONUS		
EMP		
DEPT		

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

The prompt 'SQL>' is visible at the bottom of the command window.

## 3.1 SQL\*Plus

### □ SQL\*Plus 사용하기

- ✓ 사용자마다 사용 가능한 테이블이 어떤 테이블이 있는지 살펴보기 위해서 사용되는 데이터 사전이 TAB(TABLE)이다.
- ✓ 데이터 사전(Data dictionary): 오라클 시스템에 의해서 데이터베이스에서 객체의 정보를 추가, 삭제, 수정되어 특정 객체 정보를 기록하는 곳
- ✓ 관계형 데이터베이스에서 테이블을 정의하면 오라클 시스템에 의해서 컬럼, 도메인 및 제한사항에 대한 메타 데이터 정보를 기록해 두는 곳
- ✓ 데이터 사전의 내용은 DDL과 DML 명령이 수행시 변경되고 기록

```
SQL> SELECT * FROM TAB;
```

TNAME	TABTYPE	CLUSTERID
BONUS	TABLE	
DEPT	TABLE	
EMP	TABLE	
SALGRADE	TABLE	

```
SQL>
```

## 3.1 SQL\*Plus

### □ SQL\*Plus 사용하기

- ✓ 사용 가능한 테이블이 어떤 테이블이 있는지 확인 후 **각 테이블의 내용을 확인하는 명령어**
- ✓ DESC 테이블명

```
SQL> DESC BONUS
이름              널?       유형
-----
ENAME            VARCHAR2(10)
JOB              VARCHAR2(9)
SAL              NUMBER
COMM            NUMBER
SQL>
```

- ✓ 화면에 두줄로 표시 될 때(SQL>SELECT \* FROM EMP;) 한줄로 표시하기  
SQL>SET LINESIZE 100

## 3.1 SQL\*Plus

### □ 오라클의 데이터형

종류	Data Type	설명
numeric	NUMBER	38자리까지 유효한 부동 소수점 숫자
	NUMBER(n, p)	38자리의 범위 중에서 n의 자릿수까지 유효한 숫자값으로 전체 전체 자릿수 n, 소수점 자릿수 p
character	CHAR(n)	길이가 n인 고정길이 문자값. 기본길이는 1, 최대길이는 255
	VARCHAR2(n)	길이가 n인 가변길이 문자값. 기본길이는 1, 최대길이는 2000
	LONG	2GB까지의 값을 가질 수 있는 가변길이 문자값. Table당 1개의 LONG Column만 허용한다.
	CLOB	4GB까지의 값을 가질 수 있는 문자값
date	DATE	B.C. 4712년 1월에서 A.D. 4712년 12월 31일 사이의 일자와 시간
binary	ROW와 LONGROW	각각 VARCHAR2, LONG과 같지만 Binary Data를 저장하는데 사용한다. Index설정 불가
	BLOB	4GB까지의 값을 가질 수 있는 Binary Data를 저장

## 3.1 SQL\*Plus

### □ 오라클의 산술연산자(+, -, \*, /)

Attribute가 Number

가

```
SQL> SELECT ENAME, SAL, SAL+100000, SAL-500, SAL*02, SAL/10 FROM EMP;
```

ENAME	SAL	SAL+100000	SAL-500	SAL*02	SAL/10
SMITH	800	100800	300	1600	80
ALLEN	1600	101600	1100	3200	160
WARD	1250	101250	750	2500	125
JONES	2975	102975	2475	5950	297.5
MARTIN	1250	101250	750	2500	125
BLAKE	2850	102850	2350	5700	285
CLARK	2450	102450	1950	4900	245
SCOTT	3000	103000	2500	6000	300
KING	5000	105000	4500	10000	500
TURNER	1500	101500	1000	3000	150
ADAMS	1100	101100	600	2200	110

ENAME	SAL	SAL+100000	SAL-500	SAL*02	SAL/10
JAMES	950	100950	450	1900	95
FORD	3000	103000	2500	6000	300
MILLER	1300	101300	800	2600	130

14 개의 행이 선택되었습니다.

```
SQL>
```

## 3.1 SQL\*Plus

3

□ 오라클의 비교 연산자( =, >, <, <=, >= ) Not equal (<>, !=, ^=)

```
SQL> SELECT * FROM EMP WHERE SAL >= 3000;
```

EMPNO	ENAME	JOB	MGR	HIREDATE	SAL	COMM
DEPTNO						
7788	SCOTT	ANALYST	7566	87/04/19	3000	
20						
7839	KING	PRESIDENT		81/11/17	5000	
10						
7902	FORD	ANALYST	7566	81/12/03	3000	
20						

COMM  
Commition?( )

```
SQL> SELECT * FROM EMP WHERE SAL BETWEEN 2000 AND 3000;
```

```
SQL> SELECT * FROM EMP WHERE SAL NOT BETWEEN 2000 AND 3000;
```

# 3.1 SQL\*Plus

## ❑ 오라클의 NULL값?(COMM 값이 NULL일 때 결과?)

SQL> SELECT \* FROM EMP;

EMPNO	ENAME	JOB	MGR	HIREDATE	SAL	COMM	DEPTNO
7369	SMITH	CLERK	7902	80/12/17	800		
7499	ALLEN	SALESMAN	7698	81/02/20	1600	300	
7521	WARD	SALESMAN	7698	81/02/22	1250	500	
7566	JONES	MANAGER	7839	81/04/02	2975		
7654	MARTIN	SALESMAN	7698	81/09/28	1250	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	1500	0	
7876	ADAMS	CLERK	7788	87/05/23	1100		
7900	JAMES	CLERK	7698	81/12/03	950		
7902	FORD	ANALYST	7566	81/12/03	3000		
7934	MILLER	CLERK	7782	82/01/23	1300		

14 개의 행이 선택되었습니다.

< - null

NULL x  
NULL NULL NULL  
NVL 가

SQL> SELECT ENAME, JOB, SAL\*12, SAL\*12+COMM FROM EMP;

ENAME	JOB	SAL*12	SAL*12+COMM
SMITH	CLERK	9600	
ALLEN	SALESMAN	19200	19500
WARD	SALESMAN	15000	15500
JONES	MANAGER	35700	
MARTIN	SALESMAN	15000	16400
BLAKE	MANAGER	34200	
CLARK	MANAGER	29400	
SCOTT	ANALYST	36000	
KING	PRESIDENT	60000	
TURNER	SALESMAN	18000	18000
ADAMS	CLERK	13200	

ENAME	JOB	SAL*12	SAL*12+COMM
JAMES	CLERK	11400	
FORD	ANALYST	36000	
MILLER	CLERK	15600	

14 개의 행이 선택되었습니다.

## 3.1 SQL\*Plus

□ 오라클의 NULL값?(COMM 값이 NULL일 때 0으로 계산하게 NVL함수)

```
SQL> SELECT ENAME, JOB, SAL*12+COMM, NVL(COMM, 0), SAL*12+NVL(COMM, 0) FROM EMP;
```

ENAME	JOB	SAL*12+COMM	NVL(COMM,0)	SAL*12+NVL(COMM,0)
SMITH	CLERK		0	9600
ALLEN	SALESMAN	19500	300	19500
WARD	SALESMAN	15500	500	15500
JONES	MANAGER		0	35700
MARTIN	SALESMAN	16400	1400	16400
BLAKE	MANAGER		0	34200
CLARK	MANAGER		0	29400
SCOTT	ANALYST		0	36000
KING	PRESIDENT		0	60000
TURNER	SALESMAN	18000	0	18000
ADAMS	CLERK		0	13200

ENAME	JOB	SAL*12+COMM	NVL(COMM,0)	SAL*12+NVL(COMM,0)
JAMES	CLERK		0	11400
FORD	ANALYST		0	36000
MILLER	CLERK		0	15600

14 개의 행이 선택되었습니다.

NVL(COMM,10)  
=> COMM NULL 10  
!

COMM, 10



## 3.1 SQL\*Plus

### ❑ 컬럼 이름에 별칭 지정하기-별칭(ALIAS)-AS

SQL> SELECT ENAME, SAL\*12+NUL<COMM, 0> AS AnnSal FROM EMP;

ENAME	ANNSAL
-------	--------

SMITH	9600
-------	------

ALLEN	19500
-------	-------

WARD	15500
------	-------

JONES	35700
-------	-------

MARTIN	16400
--------	-------

BLAKE	34200
-------	-------

CLARK	29400
-------	-------

SCOTT	36000
-------	-------

KING	60000
------	-------

TURNER	18000
--------	-------

ADAMS	13200
-------	-------

ENAME	ANNSAL
-------	--------

JAMES	11400
-------	-------

FORD	36000
------	-------

MILLER	15600
--------	-------

14 개의 행이 선택되었습니다.

## 3.1 SQL\*Plus

### ❑ AS없이 컬럼 이름에 별칭 부여하기(AS생략하기)

```
SQL> SELECT ENAME, SAL*12+NVL(COMM, 0) Annsal FROM EMP;
```

ENAME	ANNSAL
-------	--------

SMITH	9600
-------	------

ALLEN	19500
-------	-------

WARD	15500
------	-------

JONES	35700
-------	-------

MARTIN	16400
--------	-------

BLAKE	34200
-------	-------

CLARK	29400
-------	-------

SCOTT	36000
-------	-------

KING	60000
------	-------

TURNER	18000
--------	-------

ADAMS	13200
-------	-------

ENAME	ANNSAL
-------	--------

JAMES	11400
-------	-------

FORD	36000
------	-------

MILLER	15600
--------	-------

14 개의 행이 선택되었습니다.

AS

!

## 3.1 SQL\*Plus

□ “ “로 별칭 부여하기-원하는 문자열 표시

```
SQL> SELECT ENAME, SAL*12+NUL<COMM, 0> "A n n s a l" FROM EMP;
```

ENAME	A n n s a l
SMITH	9600
ALLEN	19500
WARD	15500
JONES	35700
MARTIN	16400
BLAKE	34200
CLARK	29400
SCOTT	36000
KING	60000
TURNER	18000
ADAMS	13200

ENAME	A n n s a l
JAMES	11400
FORD	36000
MILLER	15600

14 개의 행이 선택되었습니다.

## 3.1 SQL\*Plus

### ❑ 별칭으로 한글 사용하기

```
SQL> SELECT ENAME, SAL*12+NUL<COMM, 0> " 연봉 " FROM EMP;
```

ENAME	연봉
SMITH	9600
ALLEN	19500
WARD	15500
JONES	35700
MARTIN	16400
BLAKE	34200
CLARK	29400
SCOTT	36000
KING	60000
TURNER	18000
ADAMS	13200

ENAME	연봉
JAMES	11400
FORD	36000
MILLER	15600

14 개의 행이 선택되었습니다.

```
SQL> SELECT ENAME "성명", SAL*12+NUL<COMM, 0> " 연봉 " FROM EMP;
```

성명	연봉
SMITH	9600
ALLEN	19500
WARD	15500
JONES	35700
MARTIN	16400
BLAKE	34200
CLARK	29400
SCOTT	36000
KING	60000
TURNER	18000
ADAMS	13200

성명	연봉
JAMES	11400
FORD	36000
MILLER	15600

14 개의 행이 선택되었습니다.

## 3.1 SQL\*Plus

### ❑ Concatenation 연산자(연결 연산자) 사용하기( ' ' )

```
SQL> SELECT ENAME , ' IS A ' , JOB FROM EMP;
```

ENAME	' IS A '	JOB
SMITH	IS A	CLERK
ALLEN	IS A	SALESMAN
WARD	IS A	SALESMAN
JONES	IS A	MANAGER
MARTIN	IS A	SALESMAN
BLAKE	IS A	MANAGER
CLARK	IS A	MANAGER
SCOTT	IS A	ANALYST
KING	IS A	PRESIDENT
TURNER	IS A	SALESMAN
ADAMS	IS A	CLERK

ENAME	' IS A '	JOB
JAMES	IS A	CLERK
FORD	IS A	ANALYST
MILLER	IS A	CLERK

14 개의 행이 선택되었습니다.

## 3.1 SQL\*Plus

### ❑ Concatenation 연산자(연결 연산자) 사용하기( || ||)

```
SQL> SELECT ENAME || ' IS A ' || JOB FROM EMP;

ENAME||'ISA' ||JOB
-----
SMITH IS A CLERK
ALLEN IS A SALESMAN
WARD IS A SALESMAN
JONES IS A MANAGER
MARTIN IS A SALESMAN
BLAKE IS A MANAGER
CLARK IS A MANAGER
SCOTT IS A ANALYST
KING IS A PRESIDENT
TURNER IS A SALESMAN
ADAMS IS A CLERK

ENAME||'ISA' ||JOB
-----
JAMES IS A CLERK
FORD IS A ANALYST
MILLER IS A CLERK

14 개의 행이 선택되었습니다.
```

## 3.1 SQL\*Plus

### ❑ DISTINCT 키워드(중복 제거)

distinct - , , X  
DISTINCT

```
SQL> SELECT DEPTNO FROM EMP;
```

DEPTNO
20
30
30
20
30
30
10
20
10
30
20

DEPTNO
30
20
10

14 개의 행이 선택되었습니다.

```
SQL> SELECT DISTINCT DEPTNO FROM EMP;
```

DEPTNO
30
20
10

## 3.1 SQL\*Plus

### ❑ SQL\*Plus 편집 명령

- ✓ 오라클은 마지막 실행된 SQL문을 SQL버퍼에 보관한다. 보관 중인 버퍼의 내용을 편집, 호출, 저장 할 수 있다.
- 명령어 버퍼의 내용 표시하기

명령어(약어)	설명
LIST(L)	버퍼에 저장된 모든 SQL문 표시
/	바로 실행
RUN(R)	저장된 SQL문 보여주고 실행



## 3.1 SQL\*Plus

### ❑ SQL\*Plus 편집 명령

- ✓ 오라클은 마지막 실행된 SQL문을 SQL버퍼에 보관한다. 보관 중인 버퍼의 내용을 편집, 호출, 저장 할 수 있다.

```
SQL> SELECT ENAME, SAL*12+NUL<COMM, 0> FROM EMP;
```

ENAME	SAL*12+NUL<COMM,0>
SMITH	9600
ALLEN	19500
WARD	15500
JONES	35700
MARTIN	16400
BLAKE	34200
CLARK	29400
SCOTT	36000
KING	60000
TURNER	18000
ADAMS	13200

ENAME	SAL*12+NUL<COMM,0>
JAMES	11400
FORD	36000
MILLER	15600

14 개의 행이 선택되었습니다.

```
SQL> L  
1* SELECT ENAME, SAL*12+NUL<COMM, 0> FROM EMP  
SQL> /
```

ENAME	SAL*12+NUL<COMM,0>
SMITH	9600
ALLEN	19500
WARD	15500
JONES	35700
MARTIN	16400
BLAKE	34200
CLARK	29400
SCOTT	36000
KING	60000
TURNER	18000
ADAMS	13200

ENAME	SAL*12+NUL<COMM,0>
JAMES	11400
FORD	36000
MILLER	15600

14 개의 행이 선택되었습니다.

## 3.1 SQL\*Plus

### ❑ SQL\*Plus 파일 명령어

= SQL

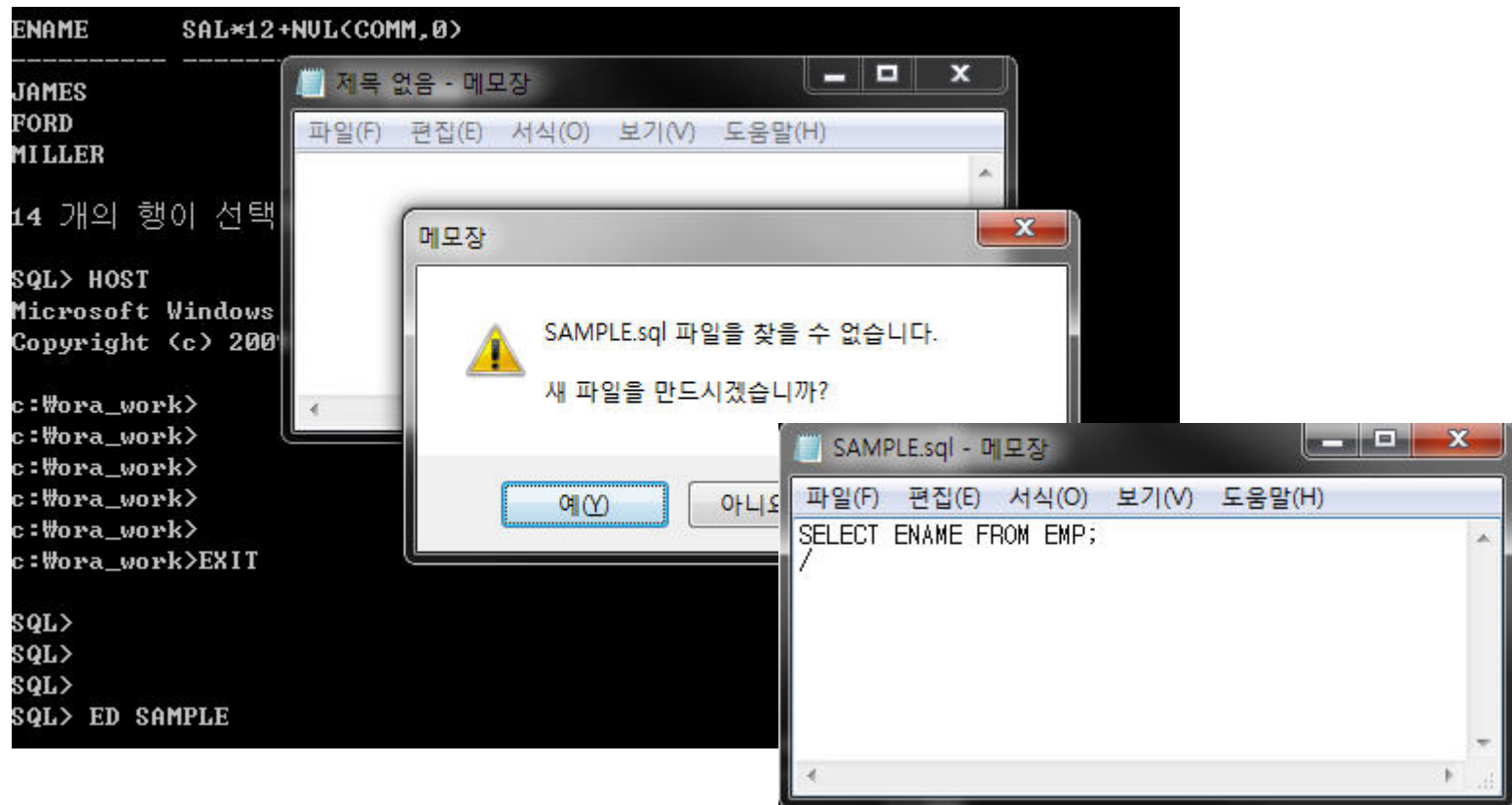
- ✓ 오라클은 현재 실행 중인 SQL문을 명령버퍼(COMMAND BUFFER)에 보관한다. 보관 중인 버퍼의 내용을 영구적으로 기록하기 위해 파일에 저장하는 SQL\*Plus명령어

명령어(약어)	설명
EDIT(ED)	파일의 내용을 VI(유닉스)나 notepad(윈도우)로 편집
HOST	오라클을 종료하지 않고 OS로 빠져나가 작업 후 EXIT로 돌아오기
SAVE	SQL버퍼 내의 현재 내용을 파일로 저장
@	SQL파일에 저장된 내용을 실행
SPOOL	오라클 화면을 갈무리하여 파일로 저장
GET	파일 내용을 SQL버퍼로 읽어들인다.
EXIT	오라클 종료

## 3.1 SQL\*Plus

### ❑ SQL\*Plus 파일 명령어

- ✓ 파일 내용을 메모장에서 편집하기



## 3.1 SQL\*Plus

### ❑ SQL\*Plus 파일 명령어

- ✓ 파일 내용을 메모장에서 실행하기(@)

```
SQL> @SAMPLE  
  
ENAME  
-----  
SMITH  
ALLEN  
WARD  
JONES  
MARTIN  
BLAKE  
CLARK  
SCOTT  
KING  
TURNER  
ADAMS  
  
ENAME  
-----  
JAMES  
FORD  
MILLER  
  
14 개의 행이 선택되었습니다.
```

## 3.1 SQL\*Plus

### ❑ SQL\*Plus 파일 명령어

- ✓ DOS 프로프트로 나가기(HOST)와 돌아오기(EXIT)

```
SQL> HOST
Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation. All rights reserved.

c:\wora_work>
c:\wora_work>
c:\wora_work>EXIT

SQL>
```

- ✓ 현재 수행중인 쿼리문 저장하기(SAVE)

## 3.1 SQL\*Plus

### ❑ SQL\*Plus 파일 명령어

✓ 현재 수행중인 쿼리문 저장하기(SAVE)

```
SQL> SELECT ENAME, SAL*12 FROM EMP;
```

ENAME	SAL*12
SMITH	9600
ALLEN	19200
WARD	15000
JONES	35700
MARTIN	15000
BLAKE	34200
CLARK	29400
SCOTT	36000
KING	60000
TURNER	18000
ADAMS	13200

14 개의 행이 선택되었습니다.

```
SQL>
```

```
SQL> SAVE A001
```

file A001.sql<이>가 생성되었습니다

```
SQL> HOST
```

Microsoft Windows [Version 6.1.7601]

Copyright (c) 2009 Microsoft Corporation. All rights reserved.

```
c:\Wora_work>DIR
```

C 드라이브의 볼륨에는 이름이 없습니다.

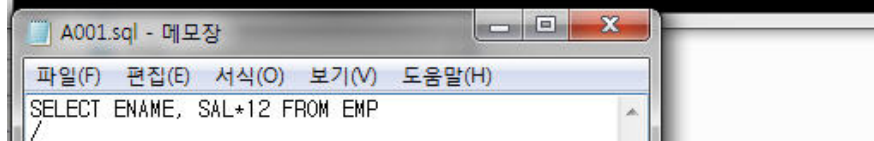
볼륨 일련 번호: C4C1-1746

c:\Wora\_work 디렉터리

2015-09-18	오후 02:11	<DIR>	.
2015-09-18	오후 02:11	<DIR>	..
2015-09-18	오후 02:11		34 A001.sql
2015-09-18	오후 02:03		25 SAMPLE.sql
2015-09-18	오전 11:57		58 test.sql
	3개 파일		117 바이트
	2개 디렉터리		350,793,056,256 바이트 남음

```
c:\Wora_work>EXIT
```

```
SQL> ED A001
```



## 3.1 SQL\*Plus

### ❑ SQL\*Plus 파일 명령어

- ✓ SQL 파일에 저장된 명령어 실행하기(@) (@파일명)

```
SQL> @TEST
TABLE_NAME
-----
SALGRADE
BONUS
EMP
DEPT

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

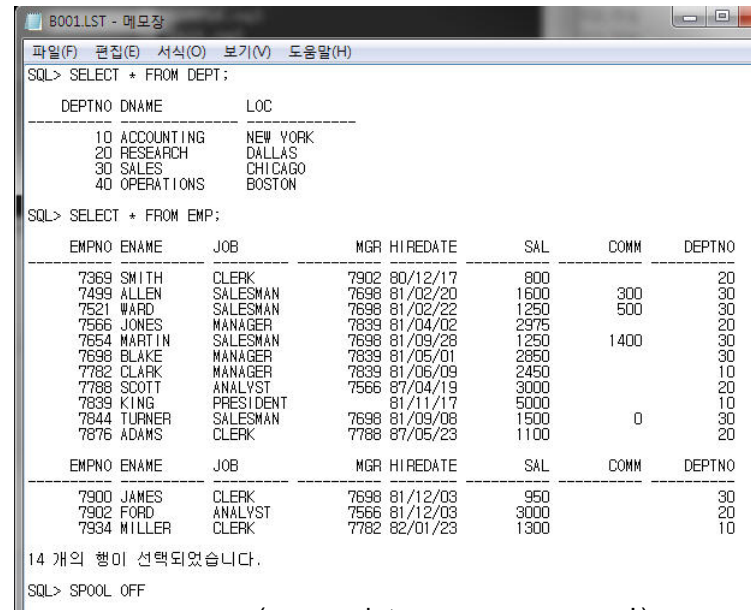
SQL> _
```

## 3.1 SQL\*Plus

### ❑ SQL\*Plus 파일 명령어

- ✓ 갈무리 하기(SPOOL) : 수업시간에 실행한 SQL문과 실행결과까지도 파일에 저장해 놓고 복습에 사용하거나 노트를 만들 때 사용한다.
- ✓ SPOOL해서 저장된 파일형식은 파일명.LST

- ① SPOOL B001
- ② SELECT \* FROM DEPT;
- ③ SELECT \* FROM EMP;
- ④ SPOOL OFF
- ⑤ HOST
- ⑥ DIR - B001.LST 파일확인
- ⑦ EXIT
- ⑧ ED B001.LST - <sup>ED B001</sup>작업했던 명령과 <sup>B001.sql</sup>결과가 모두 <sup>(</sup>저장 <sup>.lst</sup>!



```
SQL> SELECT * FROM DEPT;

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

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              30
7876 ADAMS            CLERK            7788 87/05/23          1100              20

EMPNO ENAME          JOB              MGR HIREDATE          SAL      COMM      DEPTNO
-----
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

14 개의 행이 선택되었습니다.
SQL> SPOOL OFF
```



## 3.1 SQL\*Plus

### ❑ SQL\*Plus 파일 명령어

- ✓ 저장된 SQL 명령어를 가져오기(GET)->/(실행)

```
SQL> GET A001
1* SELECT ENAME, SAL*12 FROM EMP
SQL> /
```

ENAME	SAL*12
SMITH	9600
ALLEN	19200
WARD	15000
JONES	35700
MARTIN	15000
BLAKE	34200
CLARK	29400
SCOTT	36000
KING	60000
TURNER	18000
ADAMS	13200

ENAME	SAL*12
JAMES	11400
FORD	36000
MILLER	15600

14 개의 행이 선택되었습니다.

```
SQL> _
```

## 3.1 SQL\*Plus

❑ **SQL\*Plus 시스템 변수 설정(SET) 명령어** ex) set linesize 300;

✓ 컬럼 제목의 출력여부 설정하기(HEADING 변수)

```
SQL> SET HEADING OFF
SQL> SELECT * FROM DEPT;
```

10	ACCOUNTING	NEW YORK
20	RESEARCH	DALLAS
30	SALES	CHICAGO
40	OPERATIONS	BOSTON

```
SQL> SET HEADING ON
SQL> SELECT * FROM DEPT;
```

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

```
SQL> _
```

## 3.1 SQL\*Plus

### ❑ SQL\*Plus 시스템 변수 설정(SET) 명령어

- ✓ 라인 당 출력할 문자 수 결정하기(LINESIZE 변수)
  - SET LINESIZE 100
- ✓ 컬럼에 저장된 데이터의 출력형식 변경하기(COLUMN FORMAT)
- ✓ 문자형 : A , 숫자형 : 콤마를 포함할수 있는 숫자

```
SQL> SELECT * FROM DEPT;
```

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

```
SQL> COLUMN DNAME FORMAT A25
SQL> SELECT * FROM DEPT;
```

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

```
SQL>
```

A25 - > 25  
!

```
SQL> COLUMN SAL FORMAT 9,999,999
SQL> SELECT ENAME, SAL FROM EMP;
```

ENAME	SAL
SMITH	800
ALLEN	1,600
WARD	1,250
JONES	2,975
MARTIN	1,250
BLAKE	2,850
CLARK	2,450
SCOTT	3,000
KING	5,000
TURNER	1,500
ADAMS	1,100

ENAME	SAL
JAMES	950
FORD	3,000
MILLER	1,300

14 개의 행이 선택되었습니다.

## 3.1 SQL\*Plus

### □ 사용자 생성과 권한 설정하기

✓ system 계정으로 로그인 한 후 새로운 계정을 생성하고 권한을 설정한다.

- CREATE USER [유저명] IDENTIFIED BY [비밀번호]  
DEFAULT tablespace USERS  
TEMPORARY tablespace TEMP;

- GRANT RESOURCE, CONNECT TO [유저명];

; RESOURCE : 개체를 생성, 변경, 제거 할 수 있는 권한(DDL, DML 사용이 가능)

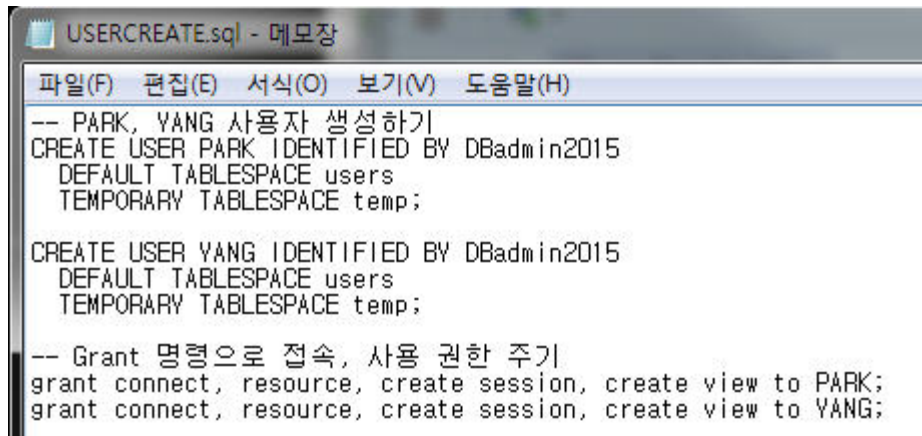
; CONNECT : 데이터베이스에 연결할 수 있는 권한

; DBA : 데이터베이스 관리자 권한 이외에도 많은 권한을 줄수 있다.

## 3.1 SQL\*Plus

### □ 사용자 생성과 권한 설정하기

- ✓ system 계정으로 로그인 한 후 새로운 계정(PARK, YANG)을 생성하고 권한을 설정
- ✓ GRANT CONNECT, RESOURCE, CREATE SESSION, CREATE VIEW TO PARK;
- ✓ GRANT CONNECT, RESOURCE, CREATE SESSION, CREATE VIEW TO YANG;



```
USERCREATE.sql - 메모장
파일(F) 편집(E) 서식(O) 보기(V) 도움말(H)
-- PARK, YANG 사용자 생성하기
CREATE USER PARK IDENTIFIED BY DBadmin2015
  DEFAULT TABLESPACE users
  TEMPORARY TABLESPACE temp;

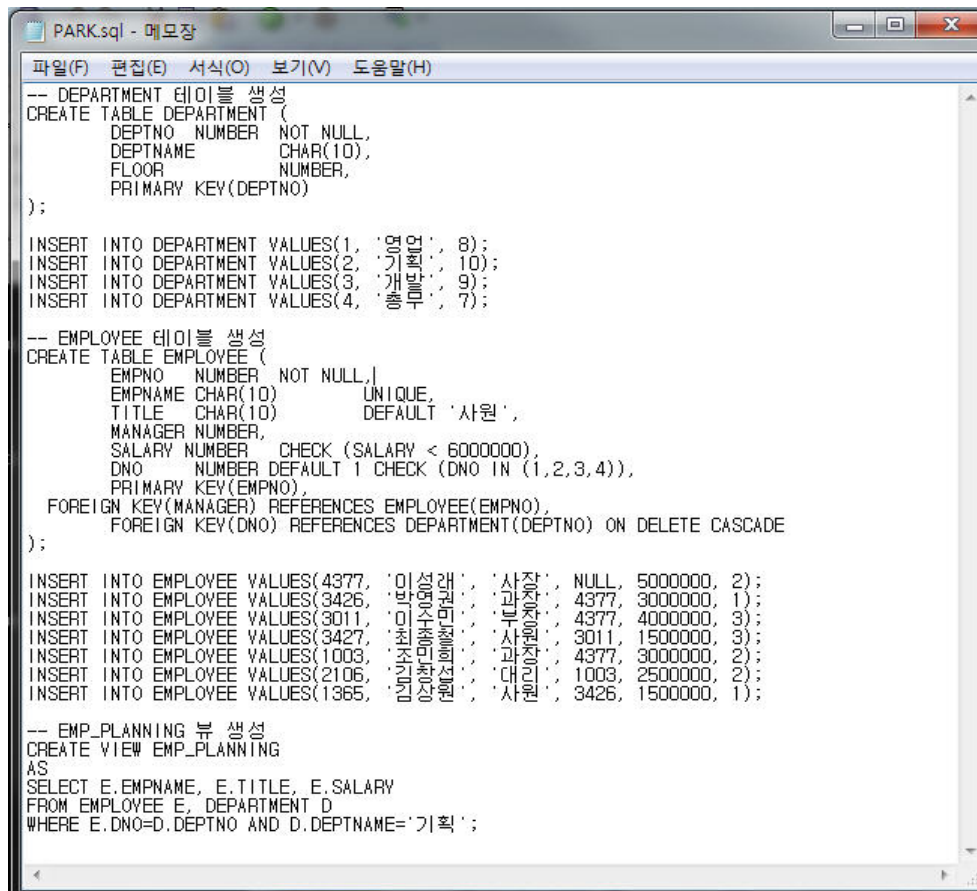
CREATE USER YANG IDENTIFIED BY DBadmin2015
  DEFAULT TABLESPACE users
  TEMPORARY TABLESPACE temp;

-- Grant 명령으로 접속, 사용 권한 주기
grant connect, resource, create session, create view to PARK;
grant connect, resource, create session, create view to YANG;
```

## 3.1 SQL\*Plus

### □ 사용자 생성과 권한 설정하기

- ✓ PARK 계정으로 로그인 후 테이블 생성과 튜플 삽입하기



```
-- DEPARTMENT 테이블 생성
CREATE TABLE DEPARTMENT (
  DEPTNO NUMBER NOT NULL,
  DEPTNAME CHAR(10),
  FLOOR NUMBER,
  PRIMARY KEY(DEPTNO)
);

INSERT INTO DEPARTMENT VALUES(1, '영역', 8);
INSERT INTO DEPARTMENT VALUES(2, '기획', 10);
INSERT INTO DEPARTMENT VALUES(3, '개발', 9);
INSERT INTO DEPARTMENT VALUES(4, '총무', 7);

-- EMPLOYEE 테이블 생성
CREATE TABLE EMPLOYEE (
  EMPNO NUMBER NOT NULL,
  EMPNAME CHAR(10) UNIQUE,
  TITLE CHAR(10) DEFAULT '사원',
  MANAGER NUMBER,
  SALARY NUMBER CHECK (SALARY < 6000000),
  DNO NUMBER DEFAULT 1 CHECK (DNO IN (1,2,3,4)),
  PRIMARY KEY(EMPNO),
  FOREIGN KEY(MANAGER) REFERENCES EMPLOYEE(EMPNO),
  FOREIGN KEY(DNO) REFERENCES DEPARTMENT(DEPTNO) ON DELETE CASCADE
);

INSERT INTO EMPLOYEE VALUES(4377, '이성재', '사장', NULL, 5000000, 2);
INSERT INTO EMPLOYEE VALUES(3426, '박영준', '과장', 4377, 3000000, 1);
INSERT INTO EMPLOYEE VALUES(3011, '김영수', '부장', 4377, 4000000, 3);
INSERT INTO EMPLOYEE VALUES(3427, '최홍민', '사원', 3011, 1500000, 3);
INSERT INTO EMPLOYEE VALUES(1003, '조민희', '과장', 4377, 3000000, 2);
INSERT INTO EMPLOYEE VALUES(2106, '김창선', '대리', 1003, 2500000, 2);
INSERT INTO EMPLOYEE VALUES(1365, '김상원', '사원', 3426, 1500000, 1);

-- EMP_PLANNING 뷰 생성
CREATE VIEW EMP_PLANNING
AS
SELECT E.EMPNAME, E.TITLE, E.SALARY
FROM EMPLOYEE E, DEPARTMENT D
WHERE E.DNO=D.DEPTNO AND D.DEPTNAME='기획';
```

### 3.1 SQL\*Plus

## □ 사용자 생성과 권한 설정하기

## ✓ PARK 계정으로 로그인 후 테이블 생성과 튜플 삽입하기

```
SQL> @PARK
```

테이블이 생성되었습니다.
1 개의 행이 만들어졌습니다.
1 개의 행이 만들어졌습니다.
1 개의 행이 만들어졌습니다.
1 개의 행이 만들어졌습니다.
테이블이 생성되었습니다.
1 개의 행이 만들어졌습니다.
1 개의 행이 만들어졌습니다.
1 개의 행이 만들어졌습니다.
1 개의 행이 만들어졌습니다.
1 개의 행이 만들어졌습니다.
1 개의 행이 만들어졌습니다.
1 개의 행이 만들어졌습니다.
뷰가 생성되었습니다.

```
SQL>
```

## 3.1 SQL\*Plus

### □ 사용자 생성과 권한 설정하기

- ✓ PARK 계정으로 테이블 생성 구조 확인 하기

```
SQL> SELECT TABLE_NAME FROM USER_TABLES;

TABLE_NAME
-----
EMPLOYEE
DEPARTMENT

SQL>
SQL> DESC EMPLOYEE;
이름                널?       유형
-----
EMPNO                NOT NULL   NUMBER
EMPNAME              CHAR(10)
TITLE                CHAR(10)
MANAGER              NUMBER
SALARY               NUMBER
DNO                  NUMBER

SQL> DESC DEPARTMENT;
이름                널?       유형
-----
DEPTNO              NOT NULL   NUMBER
DEPTNAME            CHAR(10)
FLOOR               NUMBER

SQL>
```



## 3.1 SQL\*Plus

### □ 사용자 생성과 권한 설정하기

- ✓ PARK 계정으로 테이블의 튜플 확인 하기

```
SQL> SELECT * FROM EMPLOYEE;
```

EMPNO	EMPNAME	TITLE	MANAGER	SALARY	DNO
4377	이성래	사장		5000000	2
3426	박영권	과장	4377	3000000	1
3011	이수미	부장	4377	4000000	3
3427	최종철	사원	3011	1500000	3
1003	조미희	과장	4377	3000000	2
2106	김창선	대리	1003	2500000	2
1365	김상원	사원	3426	1500000	1

7 개의 행이 선택되었습니다.

```
SQL> SELECT * FROM DEPARTMENT;
```

DEPTNO	DEPTNAME	FLOOR
1	영업	8
2	기획	10
3	개발	9
4	총무	7

```
SQL> _
```