

이 장에서 다룰 내용



1 SQL*Plus 명령어의 개념

2 SQL*Plus 편집 명령

3 SQL*Plus 파일 명령어

4 시스템 변수 조작을 위한 SET 명령어

01. SQL*Plus 명령어의 개념



- ❖ SQL*Plus는 SQL문을 실행시키고 그 결과를 볼 수 있도록 오라클에서 제공하는 툴입니다.
- ❖ SQL*Plus 명령어를 SQL 명령문과 혼동하는 사람들이 많은데, SQL은 데이터베이스에서 자료를 검색하고 수정하고 삭제하는 등을 위한 데이터베이스 언어인 반면 SQL*Plus 명령어는 툴에서 출력 형식을 지정하는 등 환경을 설정합니다.

01. SQL*Plus 명령어의 개념



❖ 다음은 SQL과 SQL*Plus 명령어의 차이점을 정리한 표입니다.

SQL 문	SQL*Plus 명령문
관계형 데이터베이스의 ANSI 표준 언어	SQL 문을 실행 시킬 수 있는 오라클의 툴
여러 줄 실행	한줄 실행
종결문자(;) 필요	종결문자 불요
키워드 단축 불가	키워드 단축 가능(describe->desc)
버퍼에 마지막 명령문 저장	버퍼 저장 안함

01. SQL*Plus 명령어의 개념



❖ SQL*Plus 명령어 중에서 자주 사용하는 명령어들은 다음과 같습니다.

명령어	기능
LIST, RUN, @, /	편집 명령
SAVE, GET, EDIT, SPOOL	파일 명령
HOST, EXIT	데이터베이스 접속 및 종료
LINE, PAGE	출력 형식

02. SQL*Plus 편집 명령



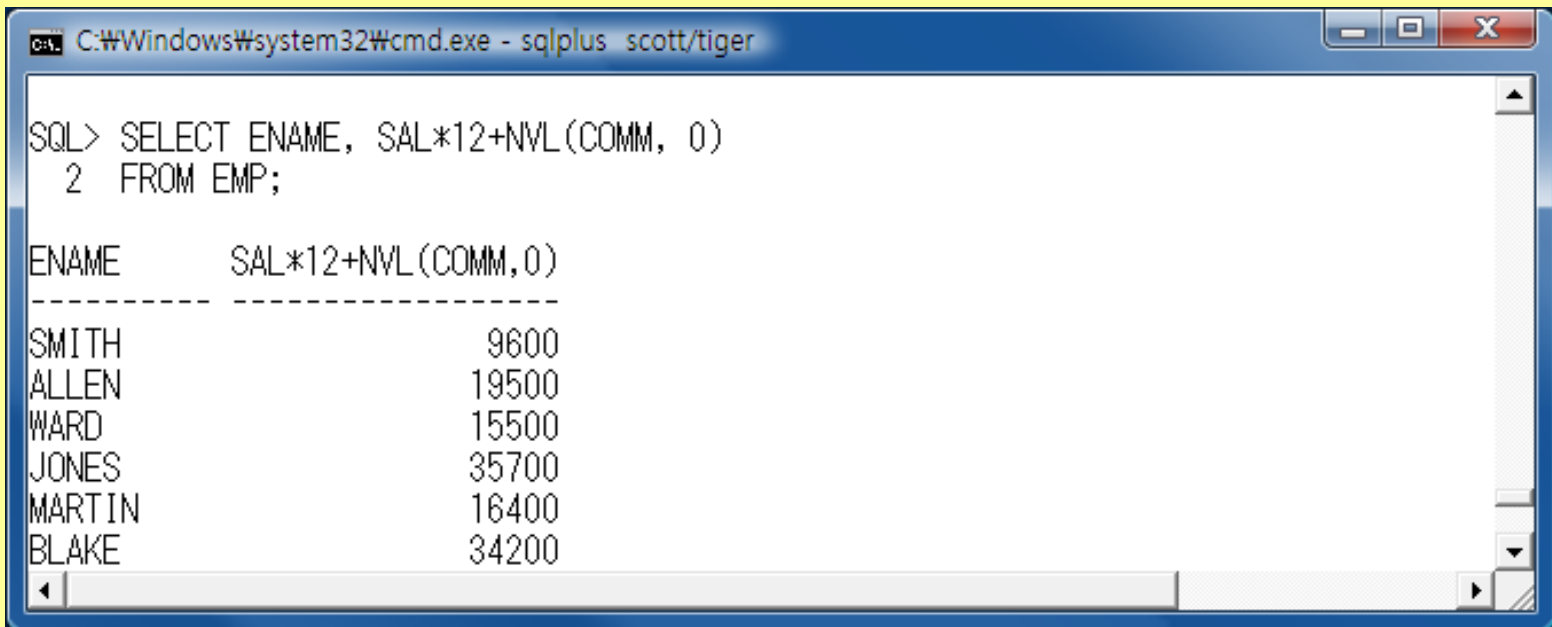
명령어(약어)	설 명
LIST (L)	버퍼에 저장된 모든 SQL 문 또는 검색한 라인의 SQL 문을 나타낸다.
/	SQL 문을 보여주지 않고 바로 실행한다.
RUN (R)	버퍼에 저장된 SQL 문을 보여주고 실행한다.

〈실습하기〉 명령어 버퍼의 내용을 출력해 보기

연봉을 계산하기 위한 쿼리문을 작성하여 실행한 후 이를 다시 한번 출력하기 위한 LIST 명령어를 수행해 봅시다.

1. 연봉을 계산하기 위해서 다음과 같이 입력합니다.

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



```
C:\Windows\system32\cmd.exe - sqlplus scott/tiger

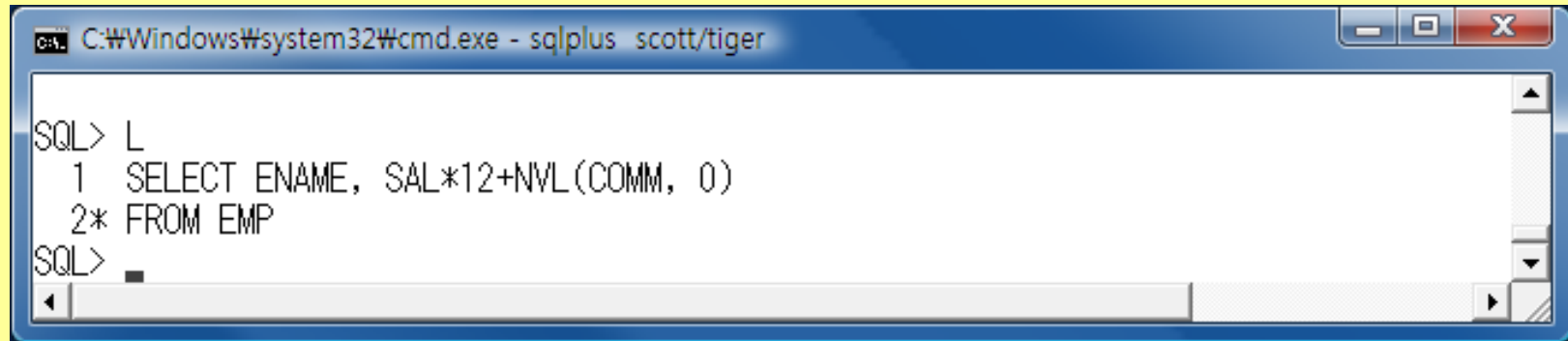
SQL> SELECT ENAME, SAL*12+NVL(COMM, 0)
2  FROM EMP;

ENAME          SAL*12+NVL(COMM,0)
-----
SMITH              9600
ALLEN             19500
WARD              15500
JONES             35700
MARTIN            16400
BLAKE             34200
```

〈실습하기〉 명령어 버퍼의 내용을 출력해 보기

2. 앞에서 사용했던 연봉을 계산하는 쿼리문을 다시 한 번 나타내기 위해서 LIST란 명령어를 사용할 수 있습니다.

SQL*Plus L



```
C:\Windows\system32\cmd.exe - sqlplus scott/tiger

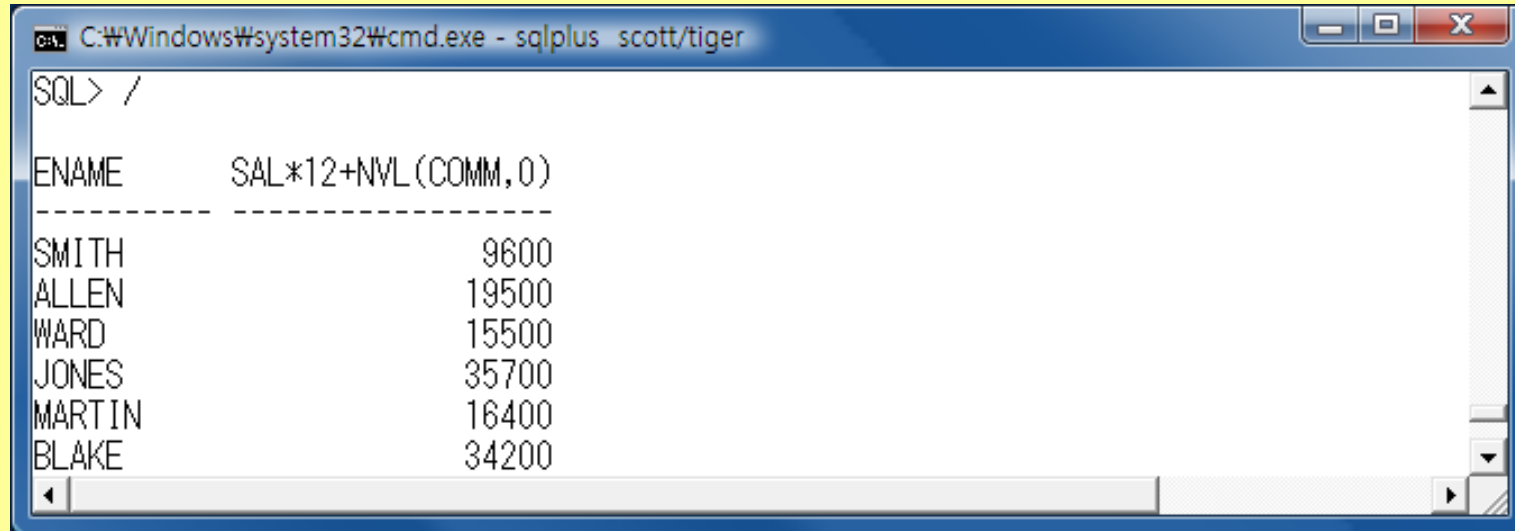
SQL> L
  1  SELECT ENAME, SAL*12+NVL(COMM, 0)
  2* FROM EMP
SQL>
```

<실습하기> SQL 버퍼에 저장된 쿼리문을 실행하는 /

버퍼에 저장된 쿼리문을 실행시키기 위해서 /을 입력합니다.

SQL*Plus

/



```
C:\Windows\system32\cmd.exe - sqlplus scott/tiger
SQL> /

ENAME          SAL*12+NVL(COMM,0)
-----
SMITH              9600
ALLEN             19500
WARD              15500
JONES             35700
MARTIN            16400
BLAKE             34200
```

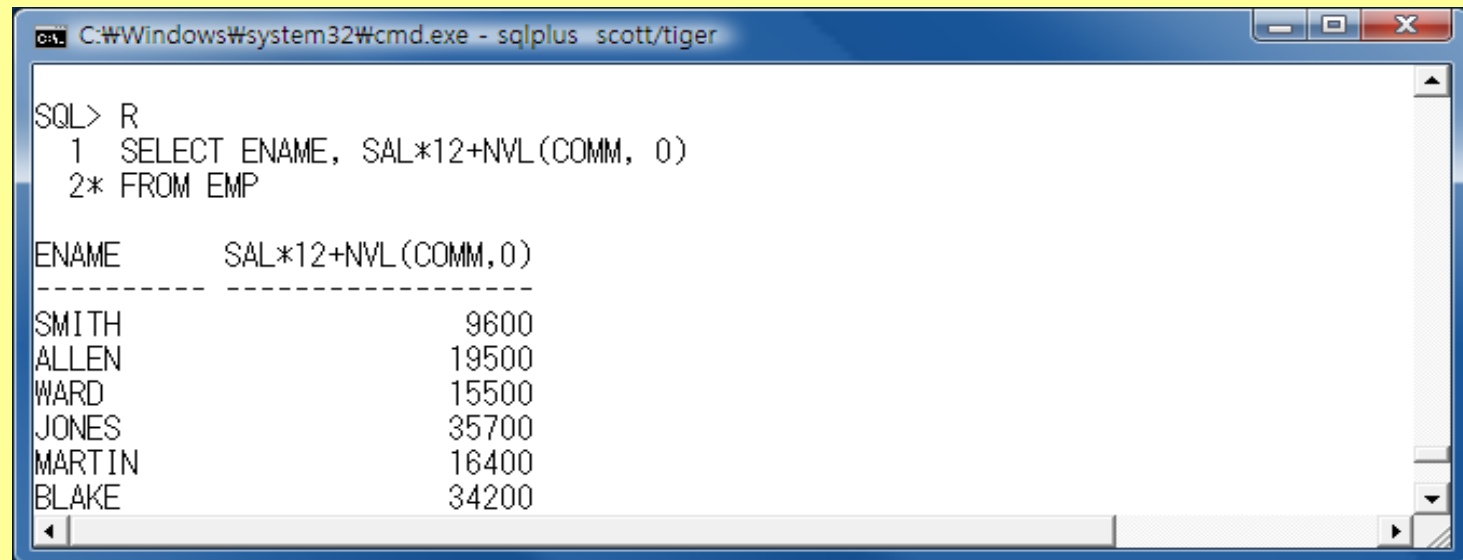

SQL 버퍼 내용을 보여주고 실행하는 RUN (R)

R[UN] 명령어는 /와 같이 버퍼에 저장된 명령을 수행하지만, /와 R[UN]의 차이점은 R[UN]은 명령 버퍼에 저장된 내용을 다시 한 번 출력한다는 점입니다.

R[UN] 명령어는 L[IST] 명령어와 / 명령어를 결합한 형태라고 말할 수 있습니다.

SQL*Plus

R



```
C:\Windows\system32\cmd.exe - sqlplus scott/tiger

SQL> R
  1  SELECT ENAME, SAL*12+NVL(COMM, 0)
  2* FROM EMP

ENAME          SAL*12+NVL(COMM,0)
-----
SMITH              9600
ALLEN             19500
WARD              15500
JONES             35700
MARTIN            16400
BLAKE             34200
```

03. SQL*Plus 파일 명령어

- ❖ 보관 중인 명령 버퍼의 내용을 영구적으로 기록하기 위해서 파일에 저장하는 SQL*Plus 파일 명령어에 대해서 살펴봅시다.



명령어(약어)	설 명
EDIT (ED)	파일의 내용을 vi(유닉스)나 notepad(윈도우즈)와 같은 에디터로 읽어 편집할 수 있도록 한다.
HOST	오라클을 종료하지 않고 OS 명령을 수행할 수 있도록 OS 환경으로 잠시 빠져 나갈 수 있도록 한다. OS Prompt 상에서 Exit 하면 다시 오라클 환경으로 돌아온다.
SAVE	SQL 버퍼 내의 현재 내용을 실제 파일로 저장한다.
@	SQL 파일에 저장된 내용을 실행한다.
SPOOL	오라클 화면을 갈무리하여 파일로 저장한다.
GET	파일의 내용을 SQL 버퍼로 읽어 들인다.
EXIT	오라클을 종료한다.

3.1 파일에 내용을 메모장에서 편집하게 하는 EDIT (ED)



- ❖ SQL은 파일의 내용을 메모장에서 쉽게 편집할 수 있도록 ED[IT] 명령어를 제공합니다.

형식	EDIT <i>filename</i>
----	----------------------

- ❖ ED[IT] 명령어를 사용할 때 파일이름을 생략하면 버퍼에 저장된 명령어를 메모장에서 쉽게 편집할 수 있도록 합니다.

SQL*Plus	ED
----------	----

- ❖ ED[IT] 명령어만 입력한 후 [Enter]키를 누르면 메모장이 실행되면서 마지막에 입력했던 SQL 명령문이 불러옵니다.
- ❖ 주의 할 점은 SQL 버퍼를 편집기로 열었을 때 명령어문 끝에 붙였던 종결문자 ; 가 편집화면에서는 /로 대체 된다는 점입니다.

; --> /

3.1 파일에 내용을 메모장에서 편집하게 하는 EDIT (ED)



- ❖ SQL 버퍼 편집할 때에도 ;를 /로 대체 되어야만 합니다.
- ❖ **SQL 문장은 반드시 한 개가 와야 하며** SQL*Plus 명령어를 저장한 후 실행 시켜도 에러가 발생함으로 주의하기 바랍니다.
- ❖ SQL 명령문이 저장된 버퍼는 파일 형태인데 그 파일 이름은 **“afiedt.buf”** 입니다.

<실습하기> 버퍼에 있는 내용을 메모장에서 편집하기

연봉 계산을 보다 정확하게 하기 위해서 커미션까지 포함해서 계산하기 위한 쿼리문으로 변경하기 위해서 ED[IT] 명령어를 사용해봅시다.

1. 연봉을 계산하기 위해서 다음과 같이 입력합니다.

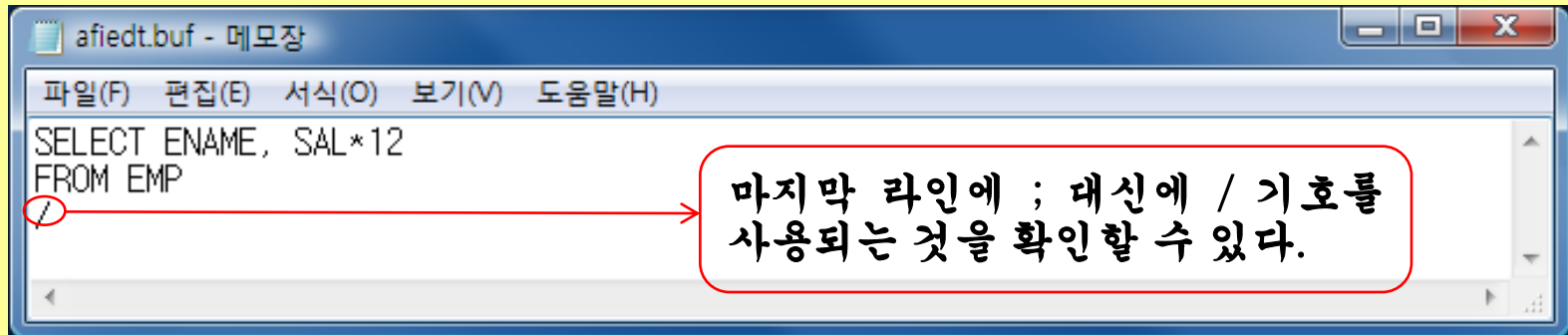
SQL

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

2. 정확한 연봉계산을 위해서 커미션을 연봉에 합산하도록 SQL 문을 변경합니다. 그러기 위해서는 SQL 프롬프트에서 ED를 입력하여 버퍼에 저장된 가장 최근의 SQL 문을 메모장으로 불러와야 합니다.

SQL*Plus

ED



<실습하기> 버퍼에 있는 내용을 메모장에서 편집하기

3. 그런 후 메모장을 다음과 같이 변경합니다. 변경 후에 끝내기 버튼[x]을 누릅니다.

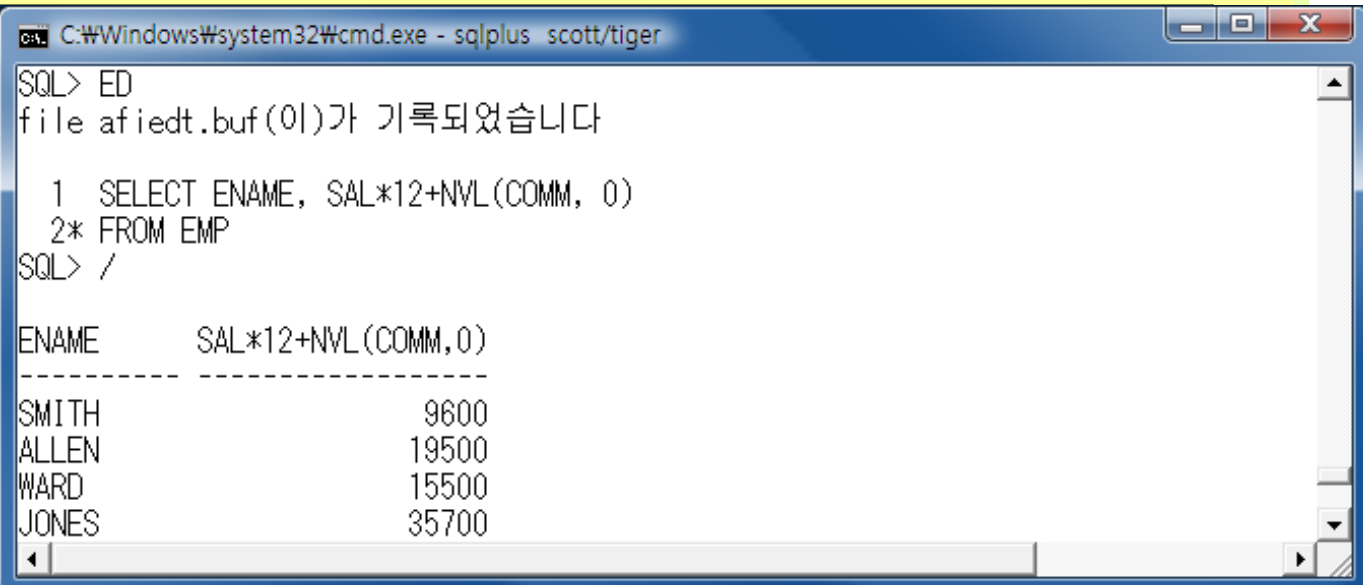
SQL	SELECT ENAME, SAL*12+NVL(COMM, 0) FROM EMP /
-----	--

4. 변경된 내용을 저장하겠냐고 물어보는 대화상자가 나타납니다. 그러면 [저장] 버튼을 클릭합니다.

5. 메모장에 수정한 내용이 저장되었음을 확인한 후에 이를 실행하기 위해서는 SQL 프롬프트에서 “/”를 입력하면 SQL문이 실행됩니다.

SQL*Plus

/



```
C:\Windows\system32\cmd.exe - sqlplus scott/tiger
SQL> ED
file afiedt.buf(이)가 기록되었습니다

  1  SELECT ENAME, SAL*12+NVL(COMM, 0)
  2*  FROM EMP
SQL> /

ENAME          SAL*12+NVL(COMM,0)
-----
SMITH              9600
ALLEN             19500
WARD              15500
JONES             35700
```

3.2 DOS 프롬프트로 나가게 하는 HOST



- ❖ SQL 명령문이 저장된 버퍼는 파일 형태인데 그 파일 이름은 "afiedt.buf"입니다.
- ❖ 오라클을 종료하지 않고 DOS 명령어인 dir를 사용하여 afiedt.buf 파일이 존재하는 것을 확인하려면 어떻게 해야 할까요?
- ❖ 바로 이럴 때 요기 나가 사용할 수 있는 명령어가 DOS 환경으로 나가는 HOST 입니다.

SQL*Plus	HOST
----------	------

- ❖ SQL*Plus 로 돌아가기 위해서는 EXIT를 입력합니다.

SQL*Plus	EXIT
----------	------

3.3 사용자가 현재 수행 중인 쿼리문을 저장하는 SAVE



- ❖ SQL*Plus에서는 사용자가 가장 최근에 수행한 쿼리문을 파일로 저장할 수 있도록 하는 SAVE 명령어를 제공합니다.

형식

SAVE *filename[.ext]* [REPLACE|APPEND]

- ❖ 파일 이름만 기술하고 확장자를 기술하지 않으면 확장자는 디폴트로(기본적으로) .sql로 지정합니다.
- ❖ SAVE 명령어는 옵션으로 REPLACE 혹은 APPEND를 사용할 수 있습니다.
- ❖ 이미 존재하는 파일에 새로운 내용을 덮어쓰길 원하면 replace 옵션을 사용합니다.
- ❖ APPEND 옵션은 이미 존재하는 파일 끝에 마지막으로 실행한 명령어가 추가됩니다.

<실습하기> 마지막에 실행한 명령어를 확장자가 sql 인 파일에 저장하기

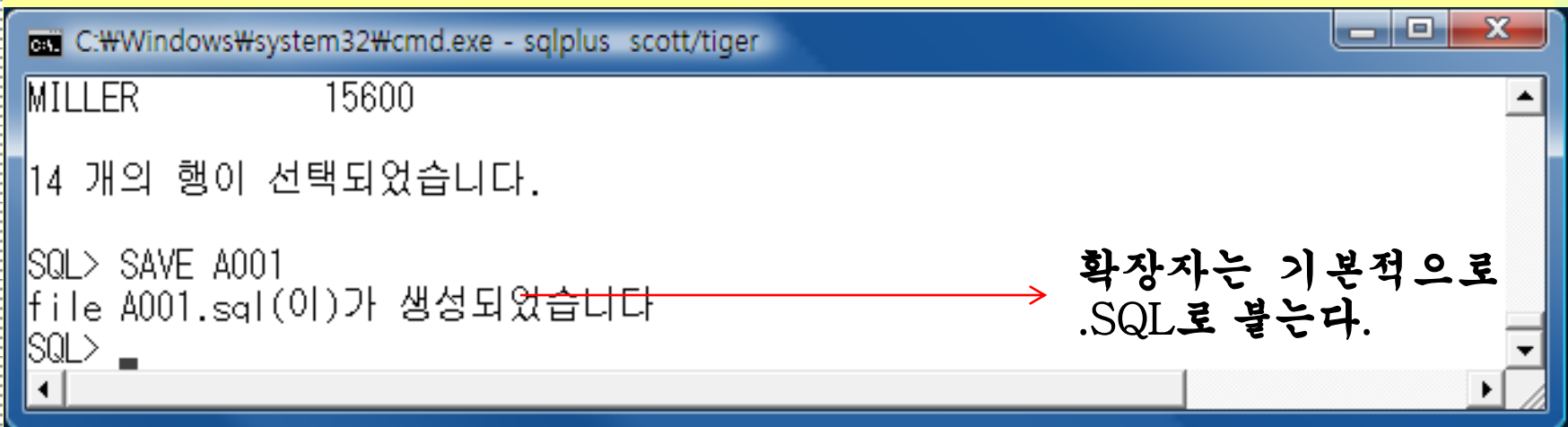
연봉 계산을 위한 쿼리문을 파일에 저장합니다.

1. 연봉을 계산하기 위해서 다음과 같이 입력합니다.

SQL	SELECT ENAME, SAL*12 FROM EMP;
-----	-----------------------------------

2. 마지막에 실행한 명령어를 a001.sql 로 저장하기 위해서 다음과 같이 입력합니다.

SQL*Plus	SAVE A001
----------	-----------



The screenshot shows a Windows command prompt window titled "C:\Windows\system32\cmd.exe - sqlplus scott/tiger". The window displays the output of a query: "MILLER 15600" and "14 개의 행이 선택되었습니다." (14 rows selected). Below this, the command "SQL> SAVE A001" is entered, followed by the message "file A001.sql(이)가 생성되었습니다" (file A001.sql has been created). A red arrow points from this message to the Korean text on the right: "확장자는 기본적으로 .SQL로 붙는다." (The extension is basically .SQL).

```
C:\Windows\system32\cmd.exe - sqlplus scott/tiger  
MILLER          15600  
  
14 개의 행이 선택되었습니다.  
  
SQL> SAVE A001  
file A001.sql(이)가 생성되었습니다  
SQL>
```

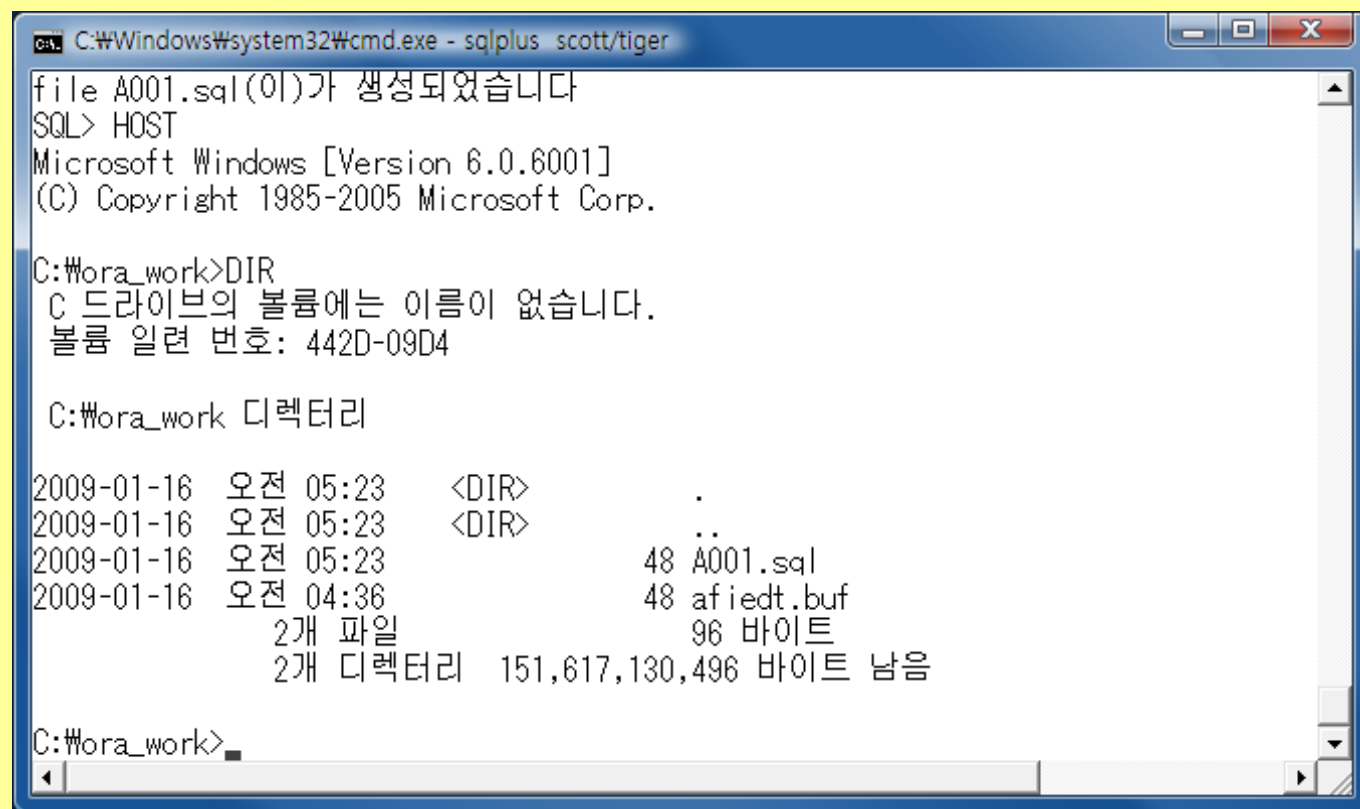
확장자는 기본적으로
.SQL로 붙는다.

<실습하기> 마지막에 실행한 명령어를 확장자가 sql 인 파일에 저장하기

3. DOS 프롬프트로 나가서 파일 목록을 살펴보면 a001.sql이 생성되어 있는 것을 확인 할 수 있습니다.

SQL*Plus

HOST



```
C:\Windows\system32\cmd.exe - sqlplus scott/tiger
file A001.sql(이)가 생성되었습니다
SQL> HOST
Microsoft Windows [Version 6.0.6001]
(C) Copyright 1985-2005 Microsoft Corp.

C:\Wora_work>DIR
C 드라이브의 볼륨에는 이름이 없습니다.
볼륨 일련 번호: 442D-09D4

C:\Wora_work 디렉터리

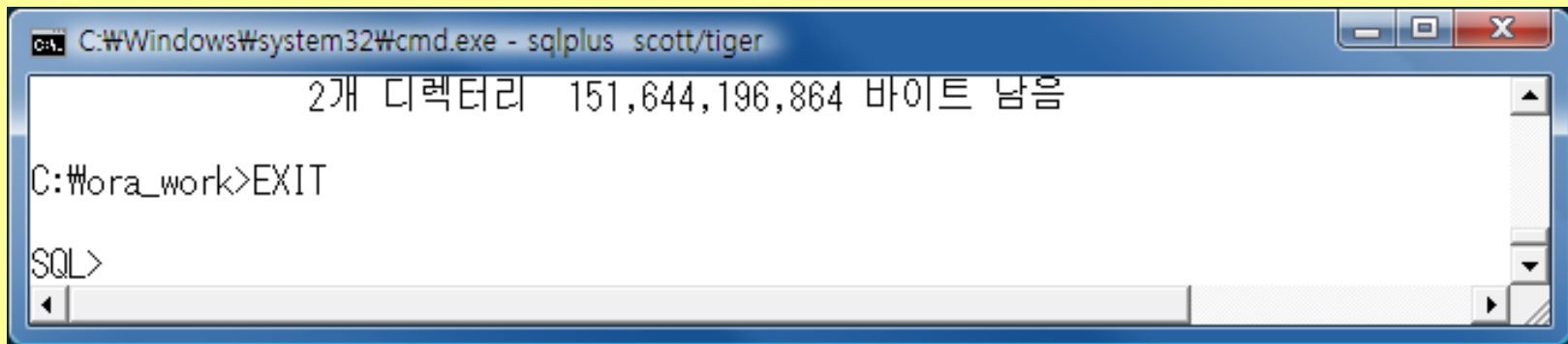
2009-01-16 오전 05:23 <DIR> .
2009-01-16 오전 05:23 <DIR> ..
2009-01-16 오전 05:23          48 A001.sql
2009-01-16 오전 04:36          48 afiedt.buf
                2개 파일             96 바이트
                2개 디렉터리 151,617,130,496 바이트 남음

C:\Wora_work>
```

<실습하기> 마지막에 실행한 명령어를 확장자가 sql 인 파일에 저장하기

4. SQL*Plus 로 돌아가기 위해서는 EXIT를 입력합니다.

SQL*Plus	EXIT
----------	------



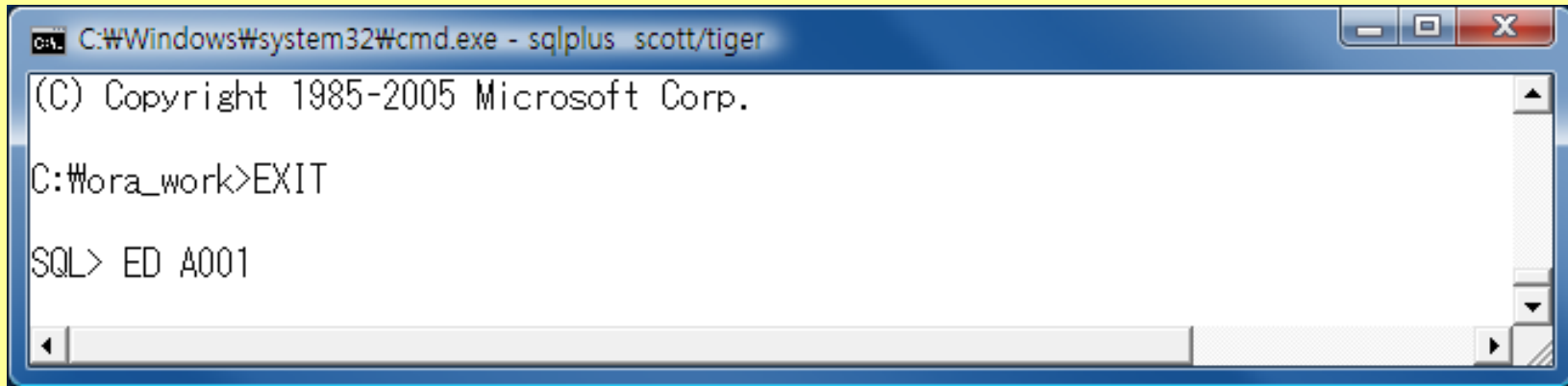
```
C:\Windows\system32\cmd.exe - sqlplus scott/tiger
2개 디렉터리 151,644,196,864 바이트 남음
C:\Wora_work>EXIT
SQL>
```

<실습하기> 마지막에 실행한 명령어를 확장자가 sql 인 파일에 저장하기

5. ED 명령어는 두 가지 용도로 사용되는데 파일 명을 생략한 채 사용하면 메모장이 실행되면서 명령 버퍼인 afiedt.buf 파일을 열지만 확장자를 생략한 채 파일 명 A001만 입력하면 디폴트로 확장자가 .sql인 파일을 찾습니다. A001.sql 파일의 내용을 확인하기 위해서 다음과 같이 입력합니다.

SQL*Plus

ED A001



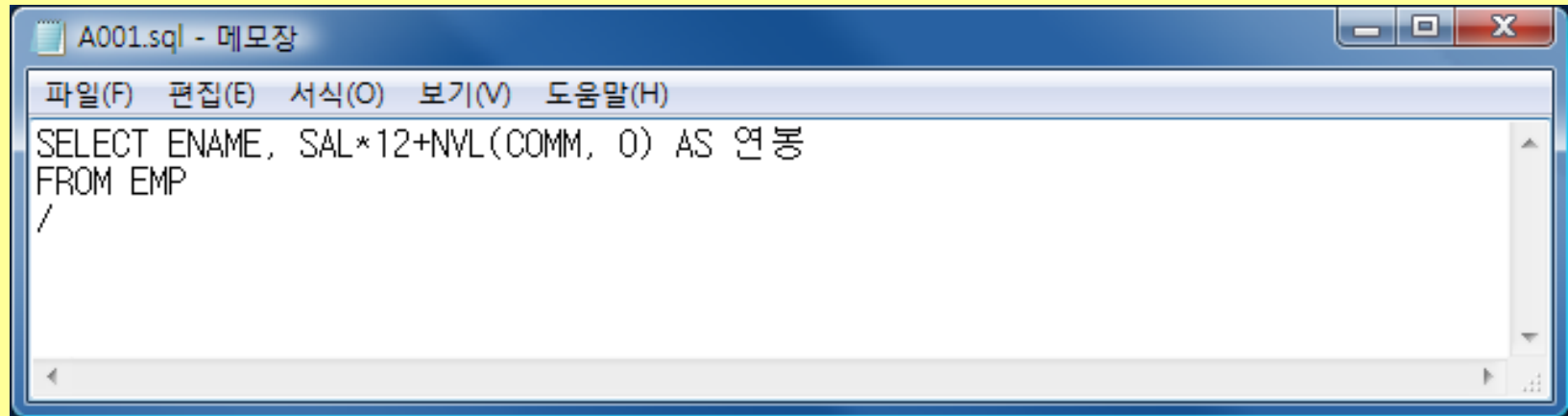
```
C:\Windows\system32\cmd.exe - sqlplus scott/tiger
(C) Copyright 1985-2005 Microsoft Corp.
C:\Wora_work>EXIT
SQL> ED A001
```

<실습하기> 마지막에 실행한 명령어를 확장자가 sql 인 파일에 저장하기

6. 메모장에 A001.sql 파일이 열립니다. 파일 안의 내용을 변경한 후 저장합니다.

SQL

```
SELECT ENAME, SAL*12+NVL(COMM, 0) AS 연봉  
FROM EMP;
```



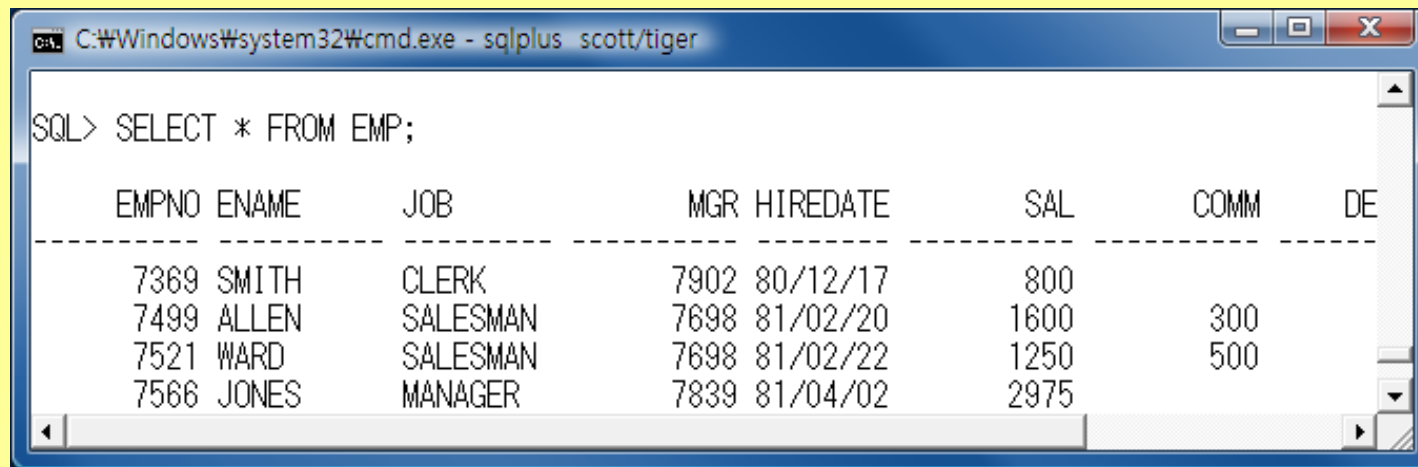
〈실습하기〉 이미 존재하는 파일의 내용 대체하기

SAVE 명령어 다음에 기술한 파일이 이미 존재 한다면 에러가 납니다. 만일 존재하는 파일의 내용을 지금 막 수행한 쿼리문으로 대체시키려면 REPLACE 라는 옵션을 사용합니다.

1. 사원의 모든 정보를 출력하는 쿼리문을 입력합니다.

SQL

SELECT * FROM EMP;



The screenshot shows a Windows command prompt window titled "C:\Windows\system32\cmd.exe - sqlplus scott/tiger". The prompt is "SQL> SELECT * FROM EMP;". The output is a table with 8 columns: EMPNO, ENAME, JOB, MGR, HIREDATE, SAL, COMM, and DE. The data is as follows:

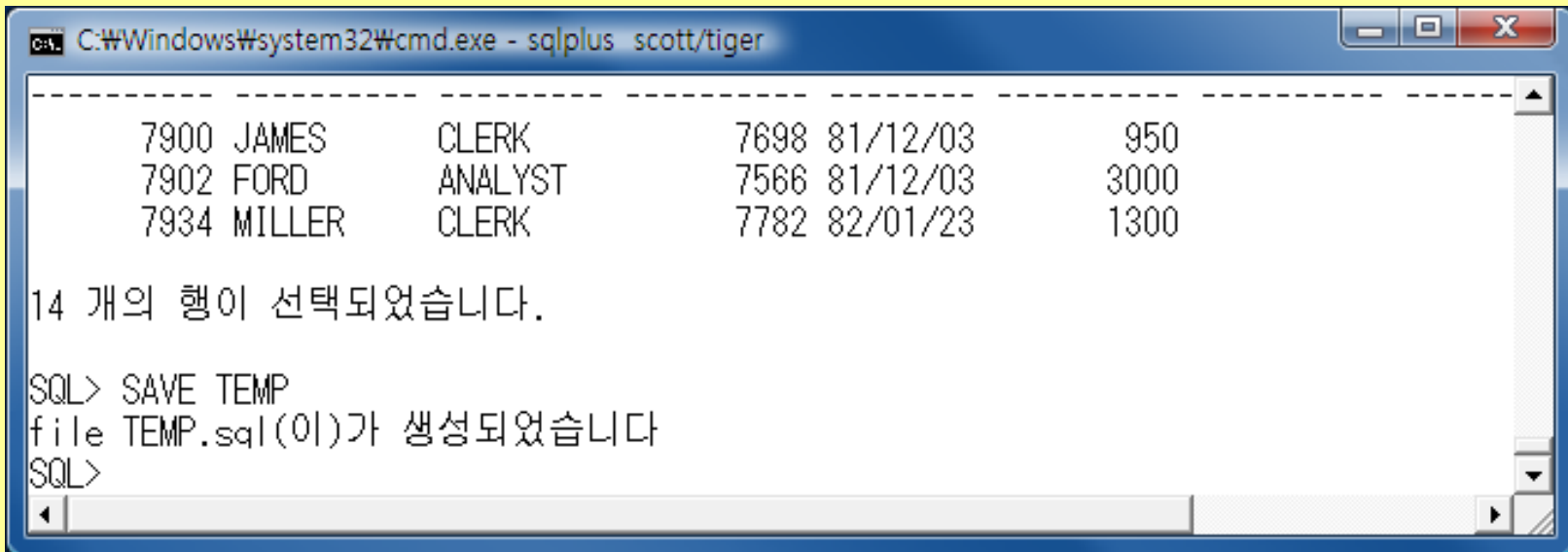
EMPNO	ENAME	JOB	MGR	HIREDATE	SAL	COMM	DE
7369	SMITH	CLERK	7902	80/12/17	800		
7499	ALLEN	SALESMAN	7698	81/02/20	1600	300	
7521	WARD	SALESMAN	7698	81/02/22	1250	500	
7566	JONES	MANAGER	7839	81/04/02	2975		

〈실습하기〉 이미 존재하는 파일의 내용 대체하기

2. TEMP.sql 라는 파일에 저장합니다.

SQL*Plus

SAVE TEMP



```
C:\Windows\system32\cmd.exe - sqlplus scott/tiger

-----
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.

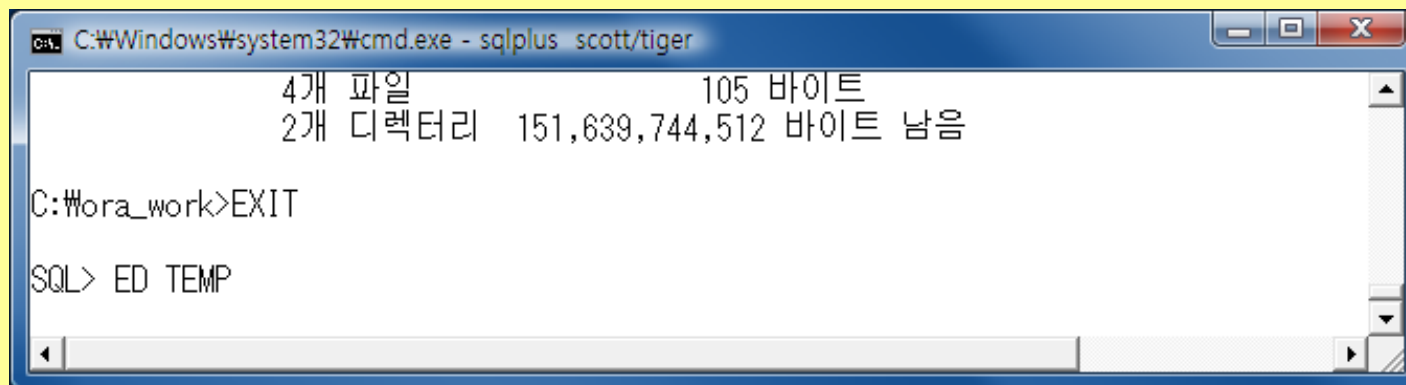
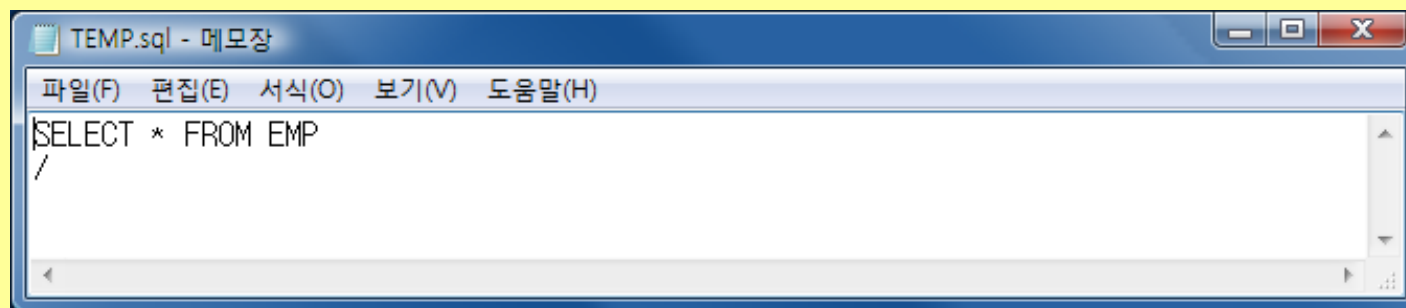
SQL> SAVE TEMP
file TEMP.sql(이)가 생성되었습니다
SQL>
```

<실습하기> 이미 존재하는 파일의 내용 대체하기

3. 다음과 같이 입력하여 파일의 내용이 변경되었음을 확인합니다.

SQL*Plus

ED TEMP

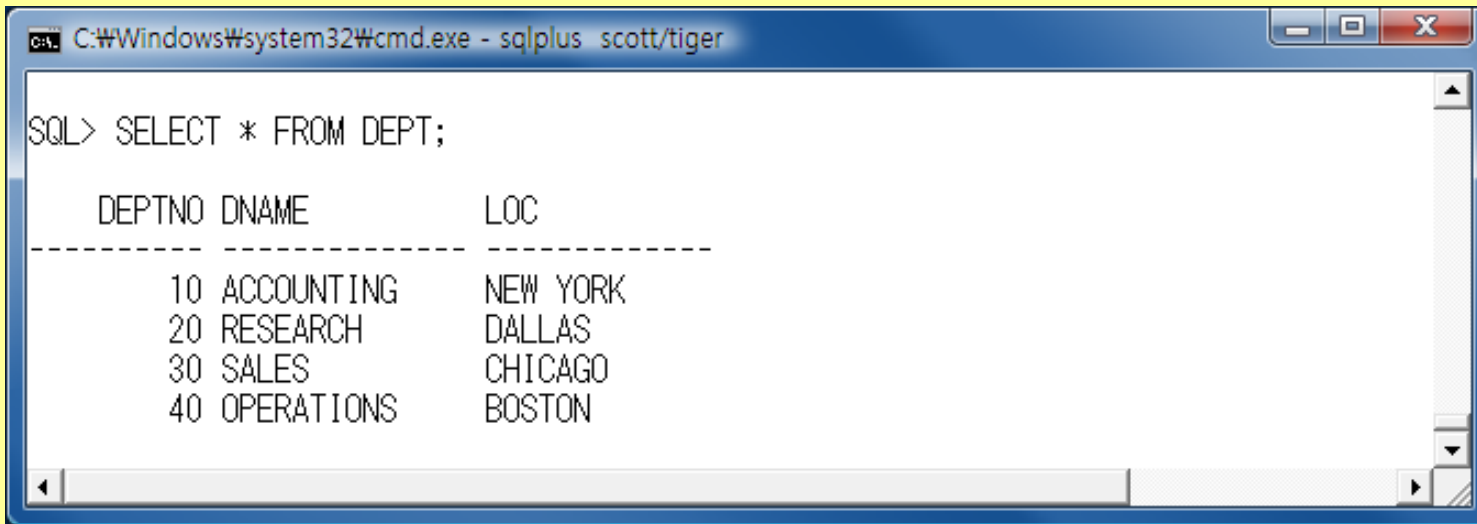


<실습하기> 이미 존재하는 파일의 내용 대체하기

4. 부서의 모든 정보를 출력하는 명령문을 수행합니다.

SQL*Plus

SELECT * FROM DEPT;



```
C:\Windows\system32\cmd.exe - sqlplus scott/tiger

SQL> SELECT * FROM DEPT;

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

〈실습하기〉 이미 존재하는 파일의 내용 대체하기

5. 지금 막 수행한 쿼리문을 TEMP 파일로 저장해 봅시다.

SQL*Plus

SAVE TEMP

```

C:\Windows\system32\cmd.exe - sqlplus scott/tiger
-----
10 ACCOUNTING      NEW YORK
20 RESEARCH        DALLAS
30 SALES           CHICAGO
40 OPERATIONS      BOSTON

SQL> SAVE TEMP
SP2-0540: "TEMP.sql" 파일은 이미 존재합니다.
"SAVE 파일명[.ext] REPLACE"을 사용합니다.
SQL>

```

파일이 이미 존재 하므로
에러가 발생합니다.

<실습하기> 이미 존재하는 파일의 내용 대체하기

6. 이미 존재하는 파일에 새로운 내용을 덮어쓰길 원하면 REPLACE를 사용합니다. 그러면 TEMP.sql 파일에 새로운 내용이 덮어 써집니다.

SQL*Plus

SAVE TEMP REPLACE

```

C:\Windows\system32\cmd.exe - sqlplus scott/tiger

      20 RESEARCH      DALLAS
      30 SALES         CHICAGO
      40 OPERATIONS    BOSTON

SQL> SAVE TEMP
SP2-0540: "TEMP.sql" 파일은 이미 존재합니다.
"SAVE 파일명[.ext] REPLACE"을 사용합니다.
SQL> SAVE TEMP REPLACE
file TEMP.sql(이)가 기록되었습니다
SQL>

```

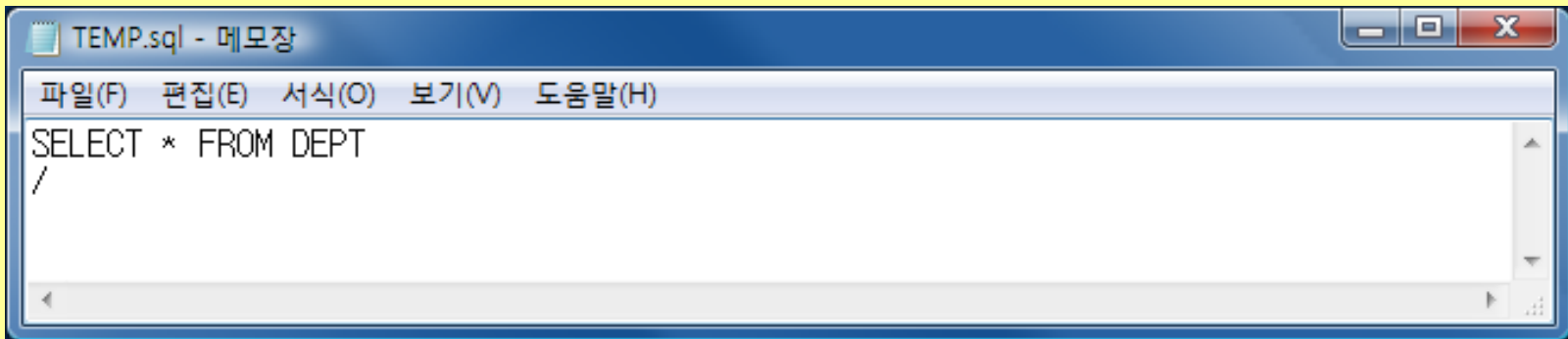
TEMP.sql 파일에 새로운
내용이 덮어 써집니다.

〈실습하기〉 이미 존재하는 파일의 내용 대체하기

7. 다음과 같이 입력하여 파일의 내용이 변경되었음을 확인합니다.

SQL*Plus

ED TEMP

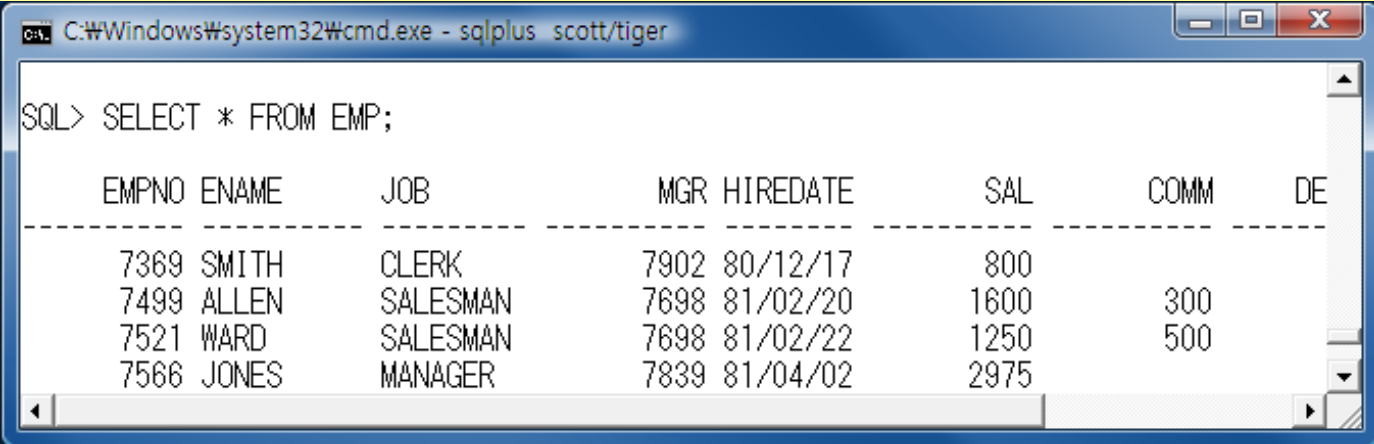


〈실습하기〉 이미 존재하는 파일에 내용 추가하기

확장자가 sql 인 파일에는 한 개 이상의 SQL문을 저장합니다. 이미 쿼리문이 저장되어 있는 sql 파일에 SAVE 명령어 APPEND 옵션을 사용하여 현재 수행한 SQL 문을 추가로 저장해 봅시다.

1. 사원의 모든 정보를 출력하는 명령문을 수행합니다.

SQL	SELECT * FROM EMP;
-----	--------------------



C:\Windows\system32\cmd.exe - sqlplus scott/tiger

SQL> SELECT * FROM EMP;

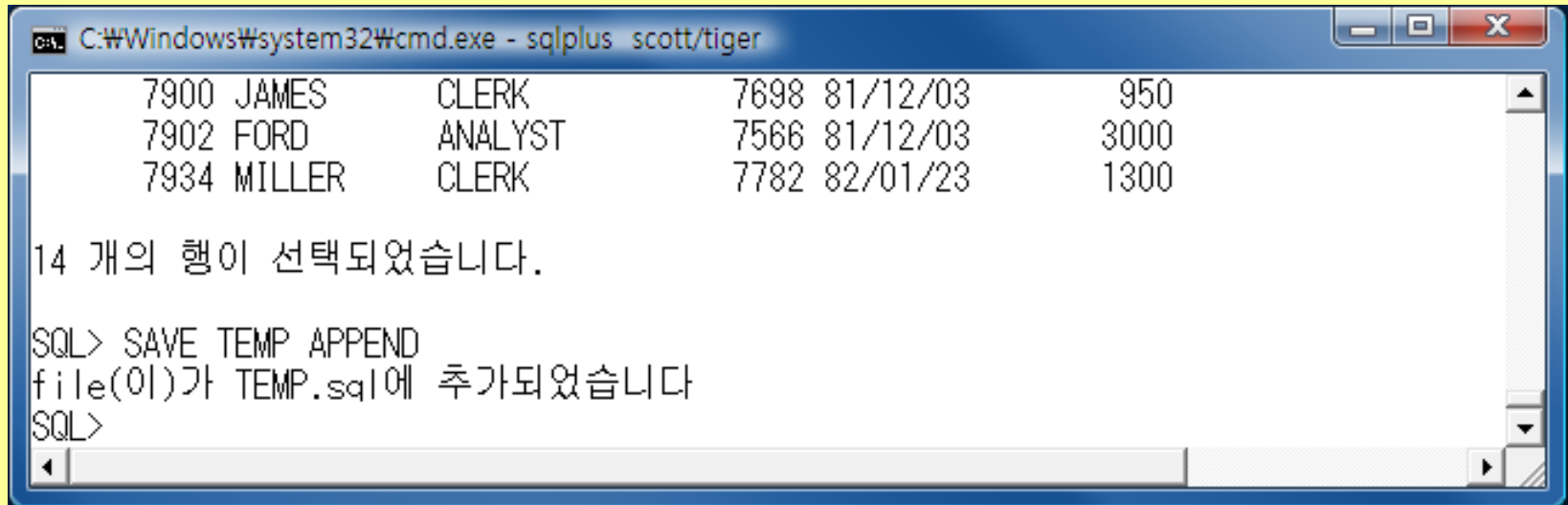
EMPNO	ENAME	JOB	MGR	HIREDATE	SAL	COMM	DE
7369	SMITH	CLERK	7902	80/12/17	800		
7499	ALLEN	SALESMAN	7698	81/02/20	1600	300	
7521	WARD	SALESMAN	7698	81/02/22	1250	500	
7566	JONES	MANAGER	7839	81/04/02	2975		

〈실습하기〉 이미 존재하는 파일에 내용 추가하기

2. TEMP.sql 파일에는 부서의 모든 정보를 출력하는 명령문인 “SELECT * FROM DEPT;”가 저장되어 있습니다. 이 내용을 저장하고 있는 TEMP.sql 파일에 사원의 모든 정보를 출력하는 명령문인 “SELECT * FROM EMP;”을 추가합니다. 그러기 위해서는 APPEND 옵션을 사용해야 합니다.

SQL*Plus

SAVE TEMP APPEND



```
C:\Windows\system32\cmd.exe - sqlplus scott/tiger

  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.

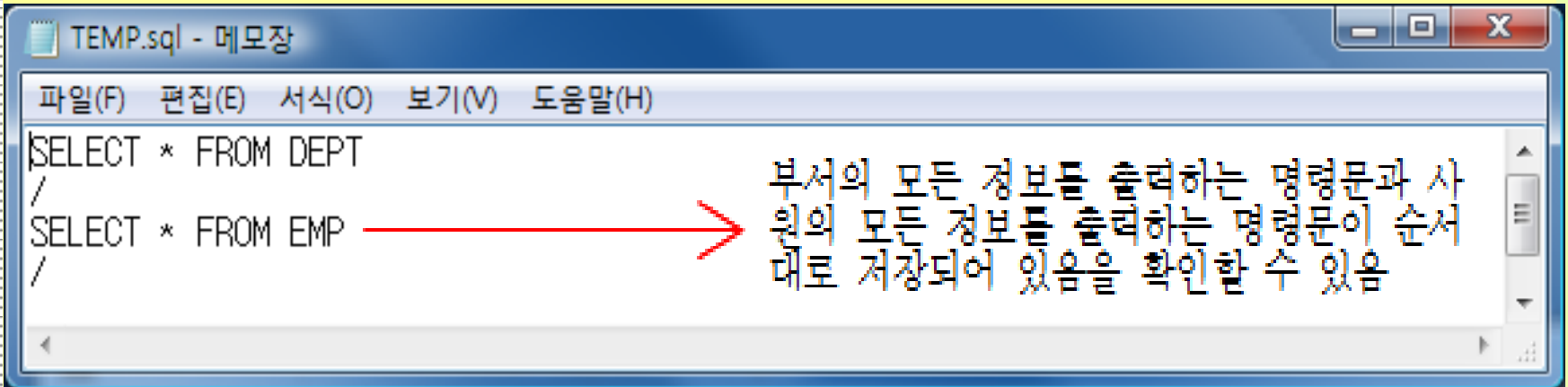
SQL> SAVE TEMP APPEND
file(이)가 TEMP.sql에 추가되었습니다
SQL>
```

〈실습하기〉 이미 존재하는 파일의 내용 대체하기

3. 다음과 같이 입력하여 파일의 내용이 변경되었음을 확인합니다.

SQL*Plus

ED TEMP



TEMP.sql - 메모장

파일(F) 편집(E) 서식(O) 보기(V) 도움말(H)

```
SELECT * FROM DEPT  
/  
SELECT * FROM EMP  
/
```

부서의 모든 정보를 출력하는 명령문과 사
원의 모든 정보를 출력하는 명령문이 순서
대로 저장되어 있음을 확인할 수 있음

3.4 SQL 파일에 저장된 명령어를 실행하는 @



- ❖ @ 명령어는 확장자가 .SQL인 파일에 저장되어 있는 쿼리문을 실행시키기 위해서 사용합니다.

형식	@ <i>filename</i>
----	-------------------

- ❖ @ 다음에는 실행시키고자 하는 파일을 지정합니다. 확장자를 생략한 채 파일명만 기술하면 확장자는 디폴트로 .SQL로 인식합니다. SQL 파일에는 일반적으로 여러 개의 쿼리문을 저장해 두기에 기술된 쿼리문을 순차적으로 수행합니다.
- ❖ TEMP.sql 파일에 저장된 쿼리문을 실행시킵시다.

형식	@TEMP
----	-------

```
SQL> @TEMP
```

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

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

3.5 갈무리 기능을 하는 SPOOL



- ❖ SAVE 명령어가 SQL문 자체를 저장하는데 비해 SPOOL명령어는 SQL문과 실행된 쿼리 결과를 파일로 기록하는 명령어 입니다.
- ❖ 즉 화면에 보여지는 내용 전체를 갈무리 해서 하나의 파일로 만듭니다.

형식	SPOOL <i>filename</i>
----	-----------------------

- ❖ SPOOL 명령은 SQL 명령문의 실행 결과 화면이 갈무리되어 지정한 파일에 기록되며, 확장자는 기본으로 .LST가 부여됩니다.

3.5 갈무리 기능을 하는 SPOOL



- ❖ SPOOL를 해제하기 위한 명령어로 SPOOL OFF가 제공됩니다.

형식	SPOOL OFF
----	-----------

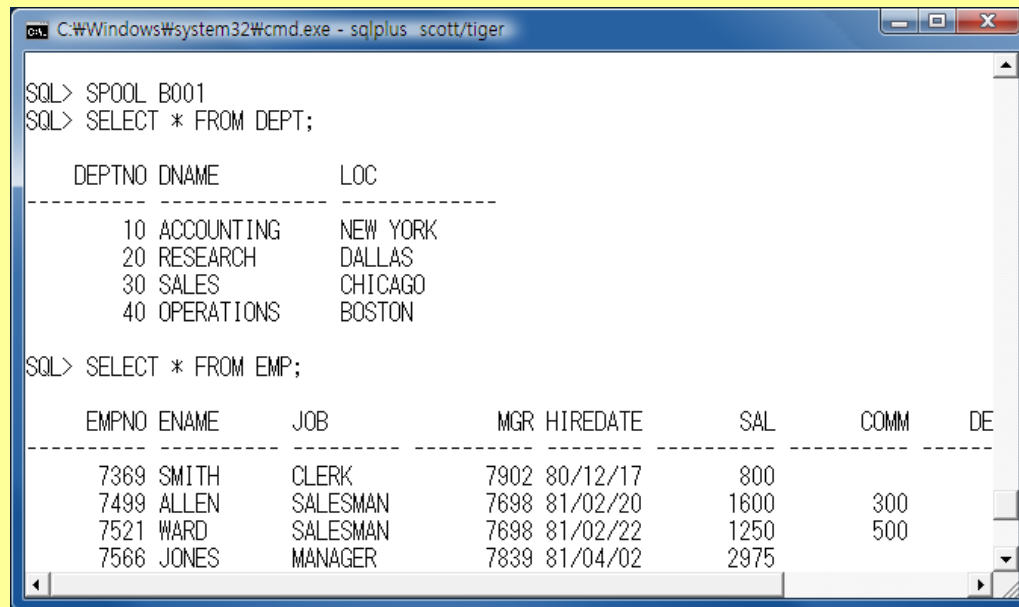
- ❖ SPOOL OFF 명령어는 SPOOL를 화면 갈무리 작업을 중단하며 해제하기 전까지의 여러 SQL 명령문이 모두 저장합니다.
- ❖ SPOOL 명령어를 사용할 때 주의할 점은 화면 갈무리한 내용을 저장하기 위해서는 반드시 SPOOL OFF를 해주어야한다는 점입니다.
- ❖ 혹시 SPOOL OFF를 하지 않고 오라클을 종료하게 되면 지금까지 갈무리한 내용이 저장되지 않고 날아가 버립니다.

〈실습하기〉 화면 갈무리하기

SQL*Plus 화면을 갈무리하여 B001.LST 파일에 저장해 봅시다.

1. SQL 프롬프트에 SPOOL을 입력하면 갈무리 기능을 시작합니다.

명령어	SPOOL B001
	SELECT * FROM DEPT;
	SELECT * FROM EMP;



```
C:\Windows\system32\cmd.exe - sqlplus scott/tiger

SQL> SPOOL B001
SQL> SELECT * FROM DEPT;

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

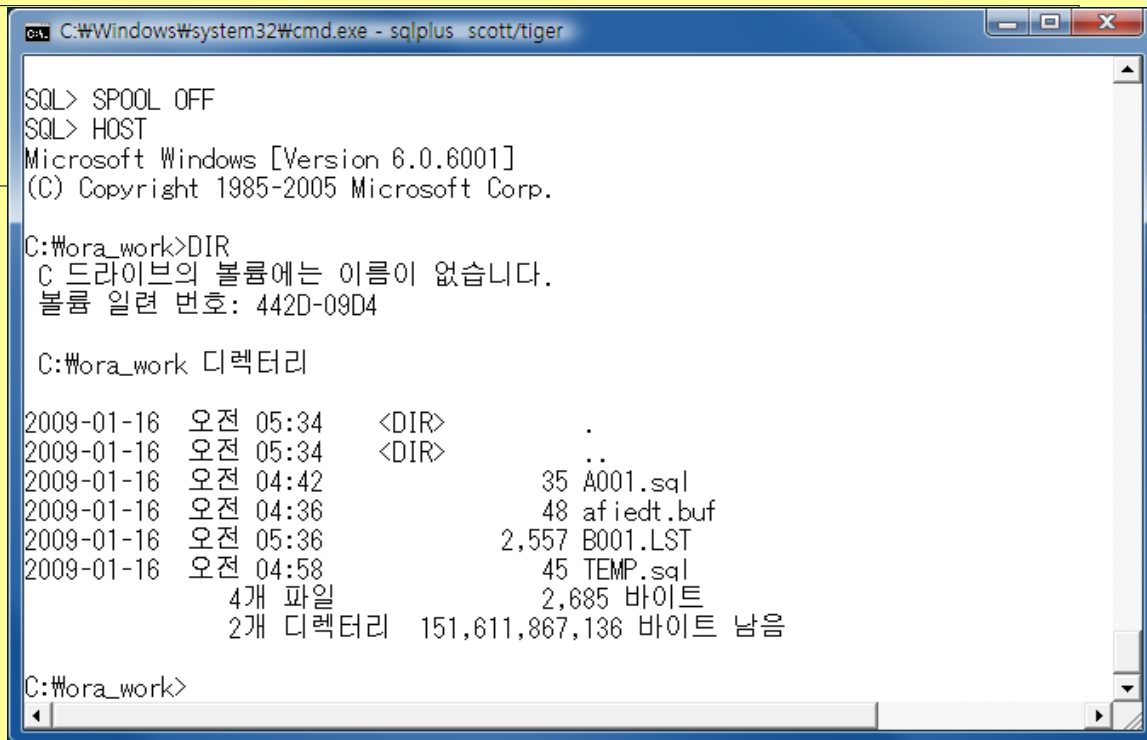
SQL> SELECT * FROM EMP;

  EMPNO ENAME      JOB              MGR HIREDATE          SAL
-----
    7369 SMITH      CLERK            7902 80/12/17          800
    7499 ALLEN      SALESMAN         7698 81/02/20         1600
    7521 WARD        SALESMAN         7698 81/02/22         1250
    7566 JONES      MANAGER          7839 81/04/02         2975
```

<실습하기> 화면 갈무리하기

2. 저장하기 위해선 반드시 SPOOL OFF를 해주어야 합니다 SPOOL OFF 명령어는 SPOOL를 해제함으로써 화면 갈무리를 중단합니다. 해제하기 전까지 여러 SQL 명령문이 모두 갈무리됩니다. SQL 명령문과 함께 실행결과 화면까지 갈무리하여 B001.LST로 저장됩니다. DOS 프롬프트로 나갑니다. 파일 목록을 보면 B001.LST가 생성되어 있는 것을 확인 할 수 있습니다.

SQL*Plus SPOOL OFF
HOST
DIR



```
C:\Windows\system32\cmd.exe - sqlplus scott/tiger

SQL> SPOOL OFF
SQL> HOST
Microsoft Windows [Version 6.0.6001]
(C) Copyright 1985-2005 Microsoft Corp.

C:\Wora_work>DIR
C 드라이브의 볼륨에는 이름이 없습니다.
볼륨 일련 번호: 442D-09D4

C:\Wora_work 디렉터리

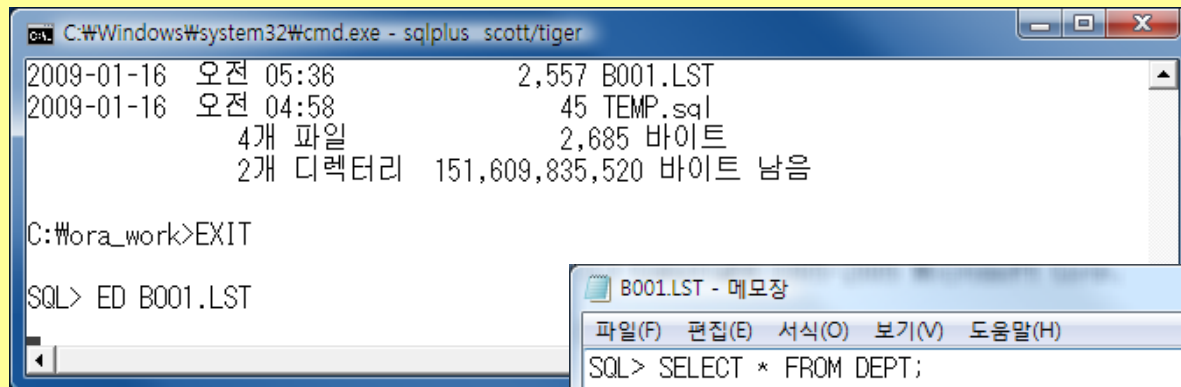
2009-01-16 오전 05:34 <DIR> .
2009-01-16 오전 05:34 <DIR> ..
2009-01-16 오전 04:42      35 A001.sql
2009-01-16 오전 04:36      48 afiedt.buf
2009-01-16 오전 05:36   2,557 B001.LST
2009-01-16 오전 04:58      45 TEMP.sql
                4개 파일          2,685 바이트
                2개 디렉터리 151,611,867,136 바이트 남음

C:\Wora_work>
```

〈실습하기〉 화면 갈무리하기

3. 다시 SQL로 되돌아 온 후에 해당 파일을 에디터(메모장)에서 열어봅시다.
ED 명령어는 디폴트 확장자를 .sql 로 인식하기 때문에 B001.LST 파일일 경우에는 확장자까지 명확하게 적어 주어야 합니다.

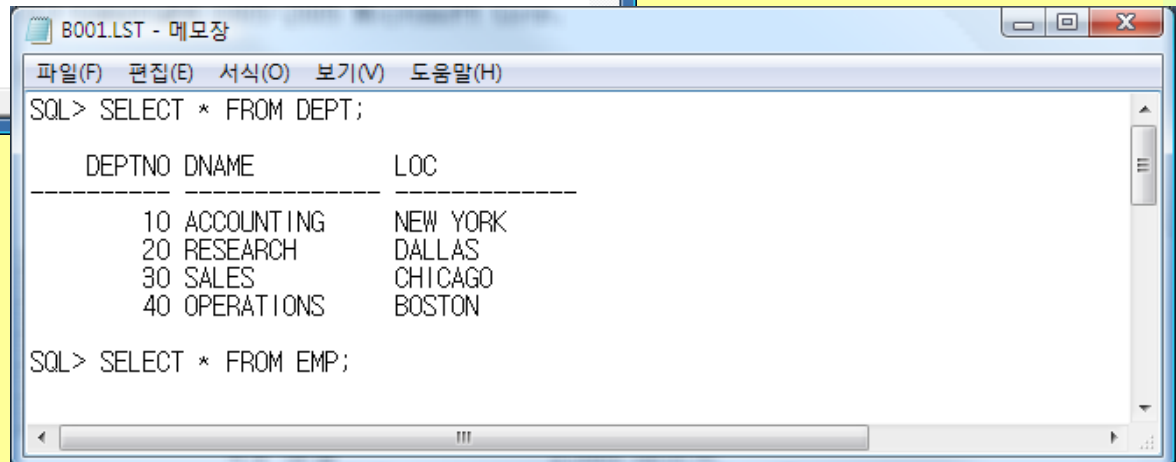
```
SQL*Plus EXIT
ED B001.LST
```



```
C:\Windows\system32\cmd.exe - sqlplus scott/tiger
2009-01-16 오전 05:36      2,557 B001.LST
2009-01-16 오전 04:58       45 TEMP.sql
      4개 파일      2,685 바이트
      2개 디렉터리 151,609,835,520 바이트 남음

C:\Wora_work>EXIT

SQL> ED B001.LST
```



```
B001.LST - 메모장
파일(F) 편집(E) 서식(O) 보기(V) 도움말(H)
SQL> SELECT * FROM DEPT;

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

SQL> SELECT * FROM EMP;
```

탄탄히 다지기

1. 명령어 버퍼의 내용을 나타내기 위한 SQL*Plus 명령어는 무엇일까요?
2. LIST와 /를 합하여 놓은 명령어로 명령어의 내용을 출력한 후 실행하는 명령어는 무엇일까요?
3. 다음 빈 칸을 채우시오.
사용자가 현재 실행한 쿼리문을 확장자가 .sql 인 SQL 파일에 저장하도록 하는 명령어는 ()이고 이렇게 저장된 SQL 파일을 실행시킬 때 사용하는 기호는 ()입니다.
4. 현재 실행중인 쿼리문은 물론 실행결과까지 모두 갈무리하여 저장하는 명령어는 무엇일까요?

3.6 저장한 SQL 명령어를 가져오는 GET



- ❖ SAVE 명령어를 사용하여 저장한 SQL 명령어를 다시 사용할 수 있는데 이때 사용하는 명령어가 GET입니다. GET 명령어도 SAVE 명령어와 마찬가지로 파일 이름만 기술하고 확장자를 기술하지 않으면 기본적으로 확장자를 .SQL로 인식합니다.

형식

GET *filename*

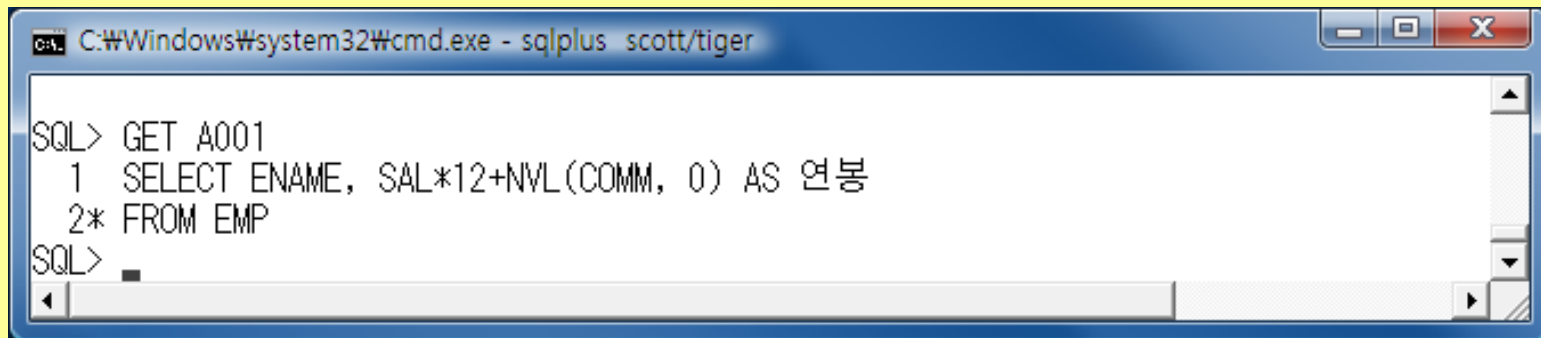
- ❖ /를 입력하면 버퍼(afiedt.buf)에 저장된 쿼리문을 실행한다고 하였습니다. 바로 전에 실행했던 쿼리문이 아니라 훨씬 이전에 실행했던 명령어를 /를 입력해서 실행하려면 어떻게 해야 할까요?
- ❖ 만일 훨씬 이전에 실행했던 명령어가 SQL 파일(SAVE 명령어로 저장해 두었겠지요^^)로 저장되어 있다면 GET 명령어로 SQL 파일에 저장된 쿼리문을 버퍼에 불러온 후에 /을 입력하여 이전에 실행했던 쿼리문을 재실행시킬 수 있습니다.

〈실습하기〉 GET을 사용하여 쿼리문을 가져오기

SAVE 명령어를 사용하여 파일에 저장한 SQL 명령문을 SQL 버퍼로 가져 오도록 합니다.

1. GET 다음에 파일이름을 기술하면 해당 쿼리문이 SQL 버퍼로 불러옵니다.

SQL*Plus GET A001



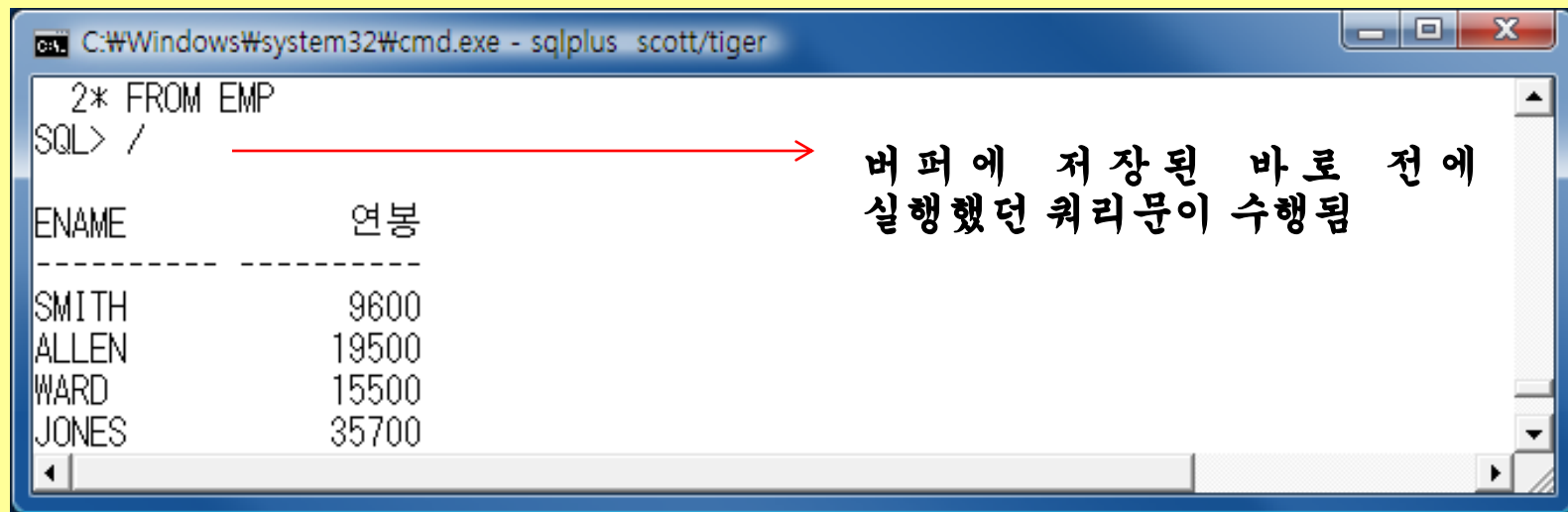
```
C:\Windows\system32\cmd.exe - sqlplus scott/tiger

SQL> GET A001
 1  SELECT ENAME, SAL*12+NVL(COMM, 0) AS 연봉
 2*  FROM EMP
SQL>
```


〈실습하기〉 GET을 사용하여 쿼리문을 가져오기

2. 이렇게 SQL버퍼에 올려진 SQL문은 앞서 학습한 /나 R 로 실행합니다.

SQL*Plus /



2* FROM EMP
SQL> /

ENAME	연봉
SMITH	9600
ALLEN	19500
WARD	15500
JONES	35700

버퍼에 저장된 바로 전에
실행했던 쿼리문이 수행됨

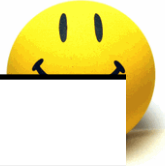
04. 시스템 변수 조작을 위한 SET 명령어



- ❖ 오라클은 다양한 시스템 변수를 제공되며 이러한 변수들은 디폴트 값이 지정되어 있습니다.
- ❖ 현재 시스템 변수의 값을 확인하기 위해서는 SHOW 명령어를 사용하고 변경하기 위해서는 SET 명령어를 사용합니다.

환경 변수(약어)	설 명
HEADING (HEA) on off	SELECT 명령어를 수행한 후 실행결과가 출력될 때 컬럼의 제목을 출력할 것인지의 여부를 제어한다. 디폴트 값은 ON이므로 컬럼 제목이 출력된다. 컬럼 제목이 출력되지 않도록 하려면 다음과 같이 설정한다. SET HEADING OFF

04. 시스템 변수 조작을 위한 SET 명령어



환경 변수(약어)	설 명
LINESIZE (LIN) n	<p>SELECT 명령어를 수행한 후 결과를 출력할 때 한 라인에 출력할 최대 문자(Character) 수를 결정한다. 디폴트 값은 80 이다. 출력할 문자 수가 80 이상이면 다음과 같이 설정한다.</p> <p>SET LINESIZE 100</p>
PAGESIZE (PAGES) n	<p>SELECT 명령어를 수행한 후 결과를 출력할 때 한 페이지에 출력할 최대 라인 수를 결정한다. 디폴트 값은 14 이다. 페이지당 출력할 라인수를 10으로 조절하려면 다음과 같이 설정한다.</p> <p>SET PAGESIZE 10</p>
COLUMN FORMAT	<p>칼럼 데이터에 대한 출력 형식을 다양하게 지정하기 위한 명령어이다.</p>

4.1 컬럼 제목의 출력 여부를 결정하는 HEADING(HEA) 변수



- ❖ HEADING은 SELECT 명령어를 수행한 후 실행결과가 출력될 때 컬럼의 제목을 출력할 것인지의 여부를 제어합니다. 일반적으로 SELECT 명령어를 수행한 실행 결과에는 컬럼 제목이 출력됩니다.

예

SET HEADING OFF

- ❖ 다음과 같이 시스템 변수 HEADING에 OFF를 지정하면 컬럼 제목이 출력되지 않습니다. 다시 컬럼 제목이 출력되도록 하려면 시스템 변수 HEADING에 ON을 지정해야 합니다.

예

SET HEADING ON

〈실습하기〉 컬럼 제목의 출력 여부를 결정하기

시스템 변수 HEADING(HEA) 로 조회되는 결과 화면에 컬럼 제목을 출력할지 말지를 결정해 봅시다.

1. 부서의 모든 정보를 출력하는 쿼리문을 수행해 봅시다.

SQL

SELECT * FROM DEPT;

C:\Windows\system32\cmd.exe - sqlplus scott/tiger

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

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

SQL>

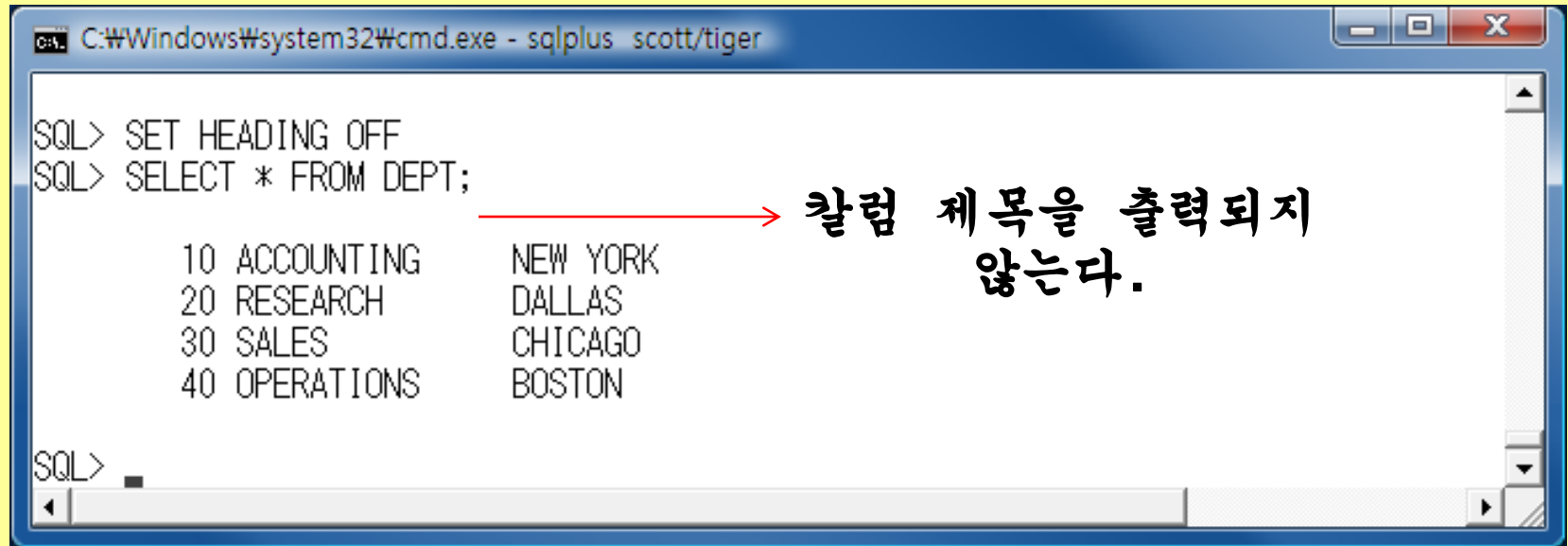
시스템 변수 HEADING 의 디폴트 값은 ON이므로 컬럼 제목이 출력됩니다.

〈실습하기〉 컬럼 제목의 출력 여부를 결정하기

2. HEADING에 OFF를 지정한 후에 부서의 모든 정보를 출력하는 쿼리문을 수행해 보시다. 컬럼 제목이 출력되지 않음을 확인할 수 있습니다.

SQL

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



The screenshot shows a Windows command prompt window titled "C:\Windows\system32\cmd.exe - sqlplus scott/tiger". Inside the window, the following SQL commands are entered:

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

The output of the query is displayed as follows:

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

A red arrow points from the output to the Korean text: **컬럼 제목을 출력되지 않는다.**

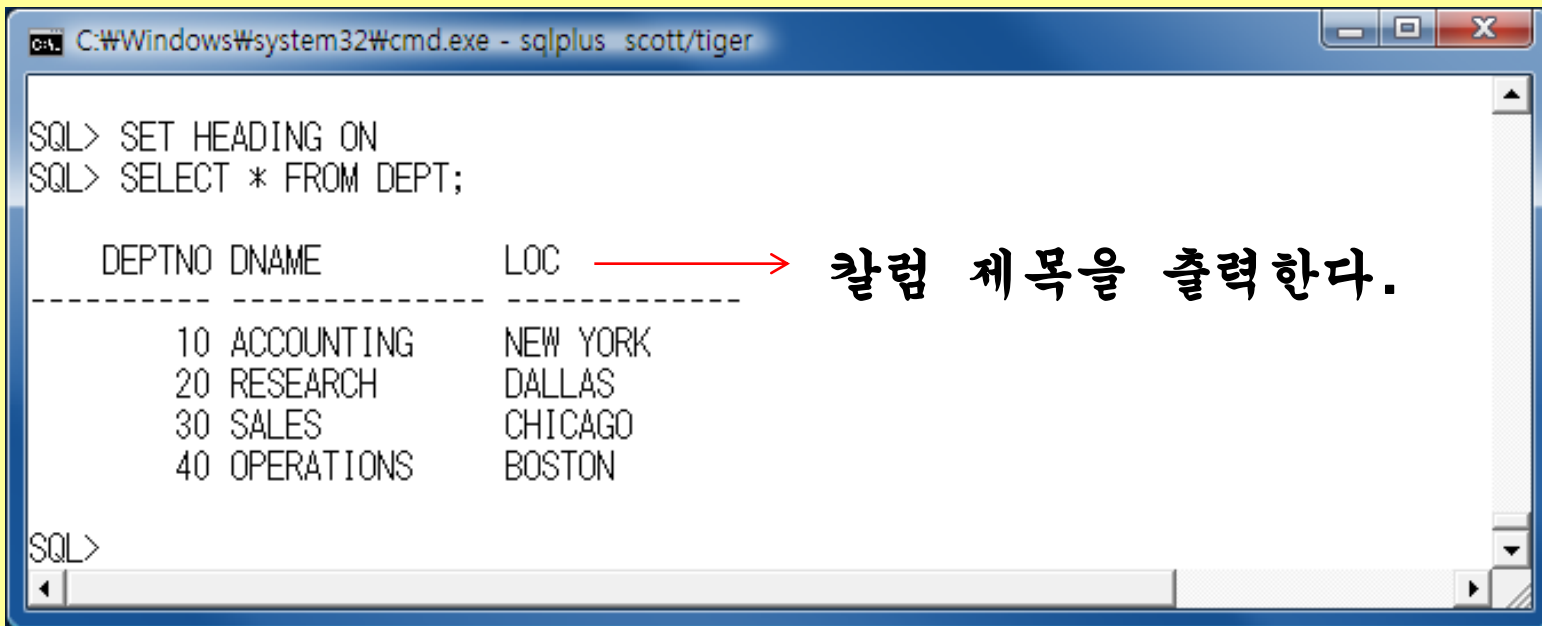
SQL>

<실습하기> 컬럼 제목의 출력 여부를 결정하기

3. 시스템 변수 HEADING에 ON을 지정하여 원상 복귀한 후에 부서의 모든 정보를 출력하는 쿼리문을 수행해 보시다.

SQL

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



The screenshot shows a Windows command prompt window titled "C:\Windows\system32\cmd.exe - sqlplus scott/tiger". The prompt is "SQL>". The user has entered the commands "SET HEADING ON" and "SELECT * FROM DEPT;". The output is a table with three columns: DEPTNO, DNAME, and LOC. The first column is right-aligned, the second is left-aligned, and the third is left-aligned. A red arrow points from the text "컬럼 제목을 출력한다." to the "LOC" column header.

```
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>
```

컬럼 제목을 출력한다.

4.2 라인 당 출력할 문자 수를 결정하는 LINESIZE 변수



- ❖ 시스템 변수 LINESIZE 는 라인 당 출력될 문자 수를 결정합니다.
- ❖ 디폴트 값은 80 이므로 SELECT 문의 출력 결과를 출력할 때 한 라인에 80까지만 출력합니다.
- ❖ 만일 한 라인에 출력할 문자 수가 80자 이상일 경우에는 다음 줄에 출력됩니다.

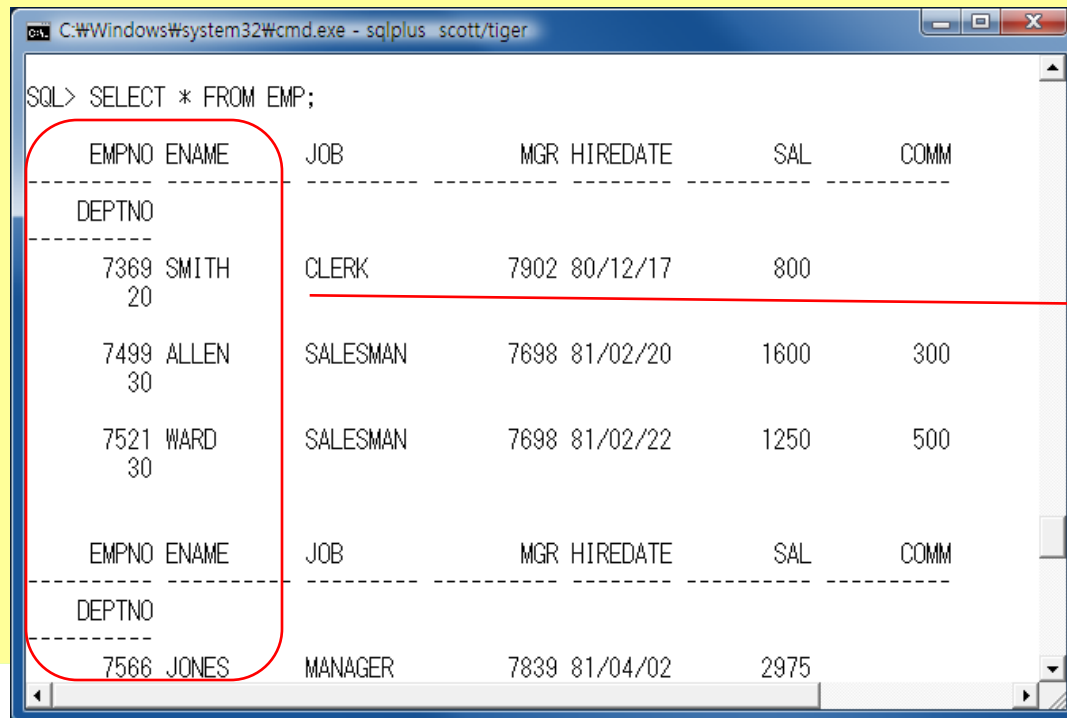
<실습하기> 라인 당 출력될 문자 수 변경하기

사원 테이블의 전체 내용을 출력하다 보면 사원 한 사람에 대한 정보가 2라인에 걸쳐 출력됨을 확인할 수 있습니다. 한 개의 로우의 내용이 80자 이상이기 때문입니다. 한 명의 사원 정보가 한 줄에 출력되도록 하기 위해서 한 라인에 출력될 문자 수를 100으로 조정합니다.

1. 사원 테이블의 전체 내용을 출력하는 쿼리문을 수행합니다.

SQL

SELECT * FROM EMP;



SQL> SELECT * FROM EMP;

EMPNO	ENAME	JOB	MGR	HIREDATE	SAL	COMM
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	

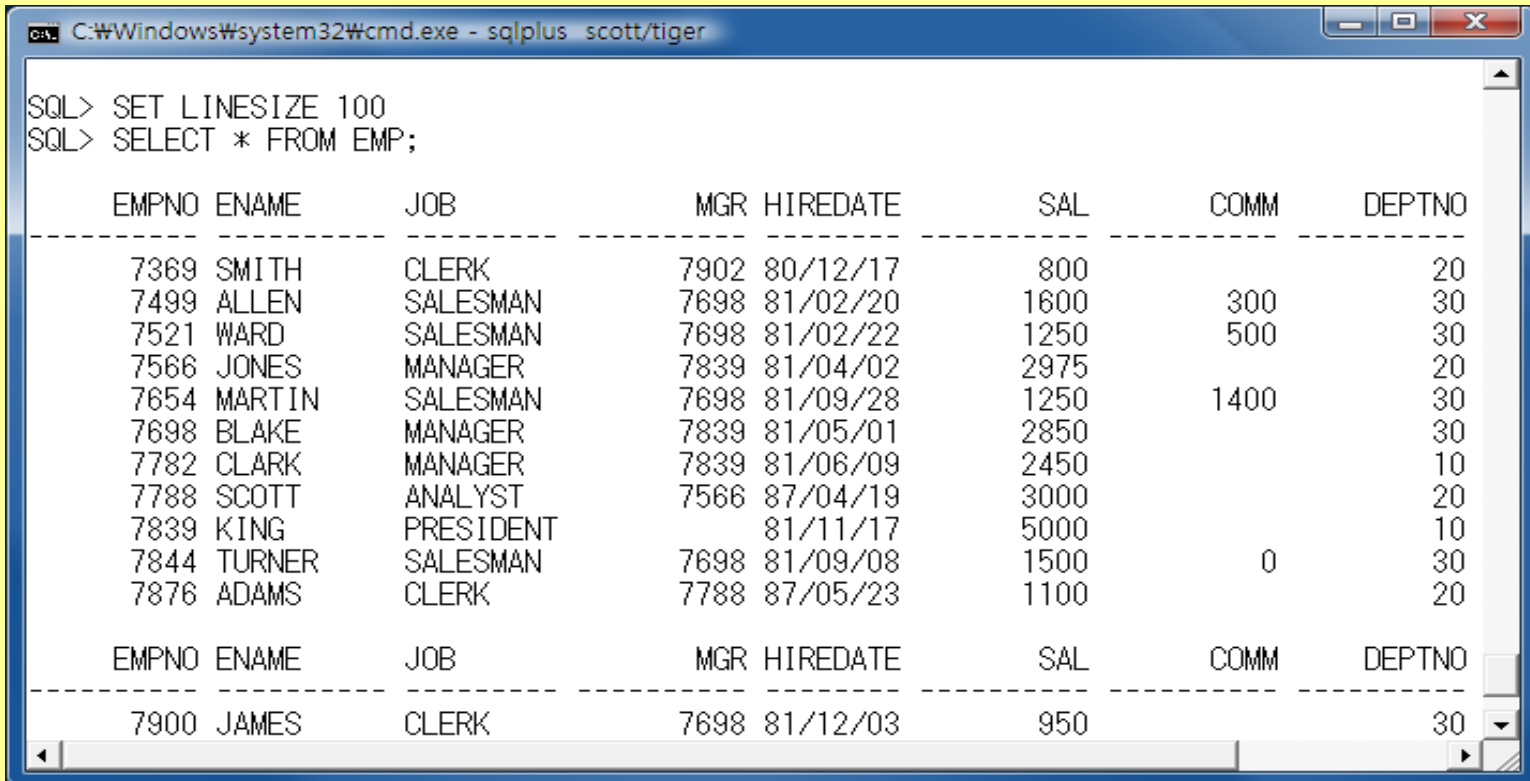
출력 결과가
깔끔하지 않음

〈실습하기〉 라인 당 출력될 문자 수 변경하기

2. 한 명의 사원 정보가 한 줄에 출력되도록 하기 위해서 한 라인에 출력될 문자 수를 100으로 조정합시다.

SQL*Plus

SET LINESIZE 100



```
SQL> SET LINESIZE 100
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

4.3 페이지 당 출력할 라인 수를 결정하는 PAGESIZE 변수



- ❖ PAGESIZE 변수는 SQL 명령문의 실행 결과에 대해서 출력될 수 있는 페이지의 크기를 설정하는 변수로서 한 페이지에 컬럼 제목, 컬럼 제목과 데이터 구분선, 페이지를 구분하기 위한 공백 라인을 포함합니다.
- ❖ 사원의 모든 정보를 출력해 봅시다.

SQL

SELECT * FROM EMP;

C:\Windows\system32\cmd.exe - sqlplus scott/tiger

```
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

14 개의 행이 선택되었습니다.

SQL>

디폴트 값은 14이기에 컬럼 제목, 컬럼 제목과 데이터 구분선, 페이지를 구분하기 위한 공백 라인을 위한 3 라인을 제외하면 한 페이지에 출력되는 사원은 11명이 됩니다.

4.3 페이지 당 출력할 라인 수를 결정하는 PAGESIZE 변수

- ❖ 페이지 크기를 10으로 조정하기 위해 PAGESIZE 변수 값을 변경해 보시다.



SQL*Plus

SET PAGESIZE 10

```
C:\Windows\system32\cmd.exe - sqlplus scott/tiger

SQL> SET PAGESIZE 10
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

14 개의 행이 선택되었습니다.

```
SQL>
```

페이지 크기를 10으로 지정하면 컬럼 제목, 컬럼 제목과 데이터 구분선, 페이지를 구분하기 위한 공백 라인을 위한 3 라인이 할당되므로 실제 한 페이지에 출력되는 사원은 7명이 됩니다.

4.4 컬럼에 저장된 데이터의 출력형식 변경을 위한 COLUMN FORMAT



- ❖ DESC 명령어로 테이블의 구조를 살펴보면 각 컬럼이 숫자 형태인지 문자 형태인지를 알 수 있습니다.
- ❖ 또한 컬럼에 저장할 수 있는 최대 크기도 알 수 있습니다. 다음은 사원 테이블의 구조를 살펴보기 위한 명령어입니다.

SQL*Plus

DESC EMP

```
C:\Windows\system32\cmd.exe - sqlplus scott/tiger

SQL> DESC EMP
이름                널?       유형
-----
EMPNO                NOT NULL   NUMBER(4)
ENAME                VARCHAR2(10)
JOB                  VARCHAR2(9)
MGR                  NUMBER(4)
HIREDATE              DATE
SAL                  NUMBER(7,2)
COMM                 NUMBER(7,2)
DEPTNO               NUMBER(2)
```

4.4 컬럼에 저장된 데이터의 출력형식 변경을 위한 COLUMN FORMAT



❖ 문자형식의 컬럼의 출력 길이 조정

- 문자형식의 컬럼의 출력 길이를 지정하기 위해서는 다음과 같이 A다음에 컬럼의 길이를 지정합니다.

예

COLUMN ENAME FORMAT A25

```
C:\Windows\system32\cmd.exe - sqlplus scott/tiger
SQL> SELECT * FROM EMP;

EMPNO  ENAME      JOB      MGR HIREDATE          SAL
-----
7369 SMITH      CLERK      7902 80/12/17          800
7499 ALLEN      SALESMAN   7698 81/02/20         1600
7521 WARD       SALESMAN   7698 81/02/22         1250
7566 JONES      MANAGER    7839 81/04/02         2975
7654 MARTIN     SALESMAN   7698 81/09/28         1250
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
7876 ADAMS      CLERK      7788 87/05/23         1100

EMPNO  ENAME      JOB      MGR HIREDATE          SAL
-----
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.

SQL>
```

```
C:\Windows\system32\cmd.exe - sqlplus scott/tiger
SQL> COLUMN ENAME FORMAT A25
SQL> SELECT * FROM EMP;

EMPNO  ENAME      JOB      MGR HIREDATE          SAL
-----
7369 SMITH      CLERK      7902 80/12/17          800
7499 ALLEN      SALESMAN   7698 81/02/20         1600
7521 WARD       SALESMAN   7698 81/02/22         1250
7566 JONES      MANAGER    7839 81/04/02         2975
7654 MARTIN     SALESMAN   7698 81/09/28         1250
7698 BLAKE      MANAGER    7839 81/05/01         2850
7782 CLARK      MANAGER    7839 81/06/09         2450

EMPNO  ENAME      JOB      MGR HIREDATE          SAL
-----
7788 SCOTT      ANALYST    7566 87/04/19         3000
7839 KING       PRESIDENT      81/11/17         5000
7844 TURNER    SALESMAN   7698 81/09/08         1500
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.

SQL>
```

4.4 컬럼에 저장된 데이터의 출력형식 변경을 위한 COLUMN FORMAT



❖ 숫자형식의 컬럼의 출력 길이 조정

- 숫자형식인 경우는 출력되는 길이에 맞게 컬럼의 길이가 자동 조정되므로 이런 명령어를 쓸 경우가 없지만 굳이 사용해야 한다면 A길이 형식 대신 9999999나 0000000을 사용해야 합니다.
- 9999999는 숫자 7 자리를 의미합니다.
- 만일 출력할 숫자가 7자리가 안될 경우에는 공란으로 출력됩니다.
- 반면 0000000은 9999999와 마찬가지로 숫자 7 자리를 의미하지만, 출력할 자리수보다 작은 데이터를 출력할 경우에는 0으로 채웁니다.

데이터	출력형식	출력결과
123456	9999999	123456
123456	0000000	0123456

4.4 컬럼에 저장된 데이터의 출력형식 변경을 위한 COLUMN FORMAT



❖ 숫자형식의 컬럼의 출력 길이 조정

- 세자리마다 ,로 구분하여 출력하고자 할 경우에는 9,999,999 혹은 0,000,000로 표현합니다.

데이터	출력형식	출력결과
123456	9,999,999	123,456
123456	0,000,000	0,123,456

4.4 컬럼에 저장된 데이터의 출력형식 변경을 위한 COLUMN FORMAT

- ❖ 급여 컬럼은 숫자 7자리로 설정하되 남은 자리는 공란으로 채우도록 하고, 커미션 컬럼을 7자리로 주되 남은 자리는 0으로 채우도록 합시다.



예

COLUMN SAL FORMAT 9,999,999
COLUMN COMM FORMAT 0,000,000

C:\Windows\system32\cmd.exe - sqlplus scott/tiger

```
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	1,600	0,000,300	30
7521	WARD	SALESMAN	7698	81/02/22	1,250	0,000,500	30
7566	JONES	MANAGER	7839	81/04/02	2,975		20
7654	MARTIN	SALESMAN	7698	81/09/28	1,250	0,001,400	30
7698	BLAKE	MANAGER	7839	81/05/01	2,850		30
7782	CLARK	MANAGER	7839	81/06/09	2,450		10
7788	SCOTT	ANALYST	7566	87/04/19	3,000		20
7839	KING	PRESIDENT		81/11/17	5,000		10
7844	TURNER	SALESMAN	7698	81/09/08	1,500	0,000,000	30
7876	ADAMS	CLERK	7788	87/05/23	1,100		20
7900	JAMES	CLERK	7698	81/12/03	950		30
7902	FORD	ANALYST	7566	81/12/03	3,000		20
7934	MILLER	CLERK	7782	82/01/23	1,300		10

14 개의 행이 선택되었습니다.

C:\Windows\system32\cmd.exe - sqlplus scott/tiger

```
SQL> COLUMN SAL FORMAT 9,999,999
SQL> COLUMN COMM FORMAT 0,000,000
SQL> SELECT * FROM EMP;
```

EMPNO	ENAME	JOB	MGR	HIREDATE	SAL	COMM	DEPTNO
7902				80/12/17	800		20
7698				81/02/20	1600	300	30
7698				81/02/22	1250	500	30
7839				81/04/02	2975		20
7698				81/09/28	1250	1400	30
7839				81/05/01	2850		30
7839				81/06/09	2450		10
7566				87/04/19	3000		20
				81/11/17	5000		10
7698				81/09/08	1500	0	30
7788				87/05/23	1100		20
7698				81/12/03	950		30
7566				81/12/03	3000		20
7782				82/01/23	1300		10

4.4 컬럼에 저장된 데이터의 출력형식 변경을 위한 COLUMN FORMAT

- ❖ 급여 컬럼은 숫자 7자리로 설정하되 남은 자리는 공란으로 채우도록 하고, 커미션 컬럼을 7자리로 주되 남은 자리는 0으로 채우도록 합시다.



예

COLUMN SAL FORMAT 9,999,999
COLUMN COMM FORMAT 0,000,000

C:\Windows\system32\cmd.exe - sqlplus scott/tiger

```
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	1,600	0,000,300	30
7521	WARD	SALESMAN	7698	81/02/22	1,250	0,000,500	30
7566	JONES	MANAGER	7839	81/04/02	2,975		20
7654	MARTIN	SALESMAN	7698	81/09/28	1,250	0,001,400	30
7698	BLAKE	MANAGER	7839	81/05/01	2,850		30
7782	CLARK	MANAGER	7839	81/06/09	2,450		10
7788	SCOTT	ANALYST	7566	87/04/19	3,000		20
7839	KING	PRESIDENT		81/11/17	5,000		10
7844	TURNER	SALESMAN	7698	81/09/08	1,500	0,000,000	30
7876	ADAMS	CLERK	7788	87/05/23	1,100		20
7900	JAMES	CLERK	7698	81/12/03	950		30
7902	FORD	ANALYST	7566	81/12/03	3,000		20
7934	MILLER	CLERK	7782	82/01/23	1,300		10

14 개의 행이 선택되었습니다.

SQL>

4.4 컬럼에 저장된 데이터의 출력형식 변경을 위한 COLUMN FORMAT



❖ 별칭(가상 컬럼)에 형식 지정하기

예

COLUMN 연봉 FORMAT 9,999,999
COLUMN 연봉 FORMAT 0,000,000

C:\Windows\system32\cmd.exe - sqlplus scott/tiger

```
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	1,600	300	30
7521	WARD	SALESMAN	7698	81/02/22	1,250	500	30
7566	JONES	MANAGER	7839	81/04/02	2,975		20
7654	MARTIN	SALESMAN	7698	81/09/28	1,250	1400	30
7698	BLAKE	MANAGER	7839	81/05/01	2,850		30
7782	CLARK	MANAGER	7839	81/06/09	2,450		10

C:\Windows\system32\cmd.exe - sqlplus scott/tiger

```
SQL> COLUMN SAL FORMAT 9,999,999
SQL> COLUMN COMM FORMAT 0,000,000
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	1,600	0,000,300	30
7521	WARD	SALESMAN	7698	81/02/22	1,250	0,000,500	30
7566	JONES	MANAGER	7839	81/04/02	2,975		20
7654	MARTIN	SALESMAN	7698	81/09/28	1,250	0,001,400	30
7698	BLAKE	MANAGER	7839	81/05/01	2,850		30
7782	CLARK	MANAGER	7839	81/06/09	2,450		10

14 개의 행이 선택되었습니다.

SQL>