

## 09. 테이블의 내용 추가, 수정, 삭제하는DML

# 01. 테이블에 새로운 행을 추가하는 INSERT 문

- ❖ INSERT 문은 테이블에 새로운 데이터를 입력하기 위해 사용하는 데이터 조작어입니다.
- ❖ 다음은 INSERT 문의 기본 형식입니다.

```
INSERT INTO table_name  
(column_name, ...)  
VALUES(column_value, ...);
```

# INSERT 문을 위한 실습에 사용할 테이블 생성하기

- ❖ INSERT 문을 위한 실습을 하기에 앞서서 실습에 사용할 테이블을 CREATE TABLE 명령어 명령어로 새롭게 만듭시다.
- ❖ 1. 이미 사용하던 테이블과 동일한 이름이 테이블을 새로 만들려면 DROP TABLE 명령어로 테이블을 삭제한 후에 기존에 있던 부서 테이블(DEPT)과 동일한 구조를 갖되 데이터는 복사하지 않는 부서 테이블(DEPT01)을 생성하기 위해서 서브 쿼리문을 사용하여 테이블을 생성합시다.

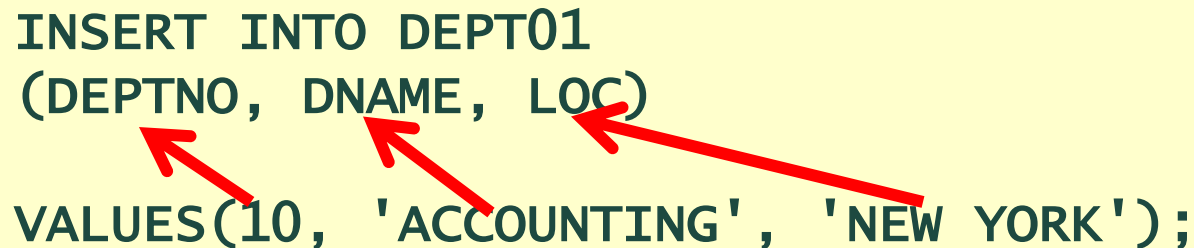
```
DROP TABLE DEPT01;
```

```
CREATE TABLE SAM02(  
YEAR01 INTERVAL YEAR(3) TO MONTH);
```

## 1.2 테이블의 구조만 복사하기

- ❖ 새로운 데이터를 추가하기 위해서 사용할 명령어 INSERT INTO ~ VALUES ~는 칼럼 명에 기술된 목록의 수와 VALUES 다음에 나오는 괄호에 기술한 값의 개수가 같아야 합니다.

```
INSERT INTO DEPT01  
(DEPTNO, DNAME, LOC)  
VALUES(10, 'ACCOUNTING', 'NEW YORK');
```



- ❖ 칼럼 DEPTNO에 10을, 칼럼 DNAME에는 'ACCOUNTING'을, 칼럼 LOC에는 'NEW YORK'을 추가합니다.

# 빈 테이블에 데이터 추가하기

- ❖ 1. DEPT01에 데이터를 추가합니다.

```
INSERT INTO DEPT01  
(DEPTNO, DNAME, LOC)  
VALUES(10, 'ACCOUNTING', 'NEW YORK');
```

## 1.1 INSERT 구문에서 오류 발생의 예

- ❖ 칼럼 명에 기술된 목록의 수보다 VALUES 다음에 나오는 괄호 안에 기술한 값의 개수가 적으면 에러가 발생합니다.

```
INSERT INTO DEPT01  
(DEPTNO, DNAME, LOC)  
VALUES (10, 'ACCOUNTING');
```

- ❖ 칼럼 명에 기술된 목록의 수보다 VALUES 다음에 나오는 괄호에 기술한 값의 개수가 많으면 에러가 발생합니다.

```
INSERT INTO DEPT01  
(DEPTNO, DNAME, LOC)  
VALUES(10, 'ACCOUNTING', 'NEW YORK', 20);
```

## 1.1 INSERT 구문에서 오류 발생의 예

- ❖ 칼럼 명이 잘못 입력되었을 때에도 에러가 발생합니다.

```
INSERT INTO DEPT01  
(NUM, DNAME, LOC)  
VALUES(10, 'ACCOUNTING', 'NEW YORK');
```

- ❖ 칼럼과 입력할 값의 데이터 타입이 서로 맞지 않을 경우에도 에러가 발생합니다.

```
INSERT INTO DEPT01  
(DEPTNO, DNAME, LOC)  
VALUES(10, ACCOUNTING, 'NEW YORK');
```

## 1.2 칼럼 명을 생략한 INSERT 구문

- ❖ 테이블에 로우를 추가할 때 모든 칼럼에 모두 자료를 입력하는 경우에는 굳이 칼럼 목록을 기술하지 않아도 됩니다.
- ❖ 칼럼 목록이 생략되면 VALUES 절 다음의 값들이 테이블의 기본 칼럼 순서대로 입력됩니다.
- ❖ 테이블의 컬럼 순서는 CREATE TABLE로 테이블을 생성할 때의 순서를 따릅니다.
- ❖ 테이블의 기본 칼럼 순서는 DESC 문으로 조회했을 때 보여 지는 순서입니다.



## 칼럼 명을 생략한 INSERT 구문 사용하기

- ❖ 칼럼명을 생략한 채 테이블이 갖은 모든 컬럼에 데이터를 추가해 봅시다.

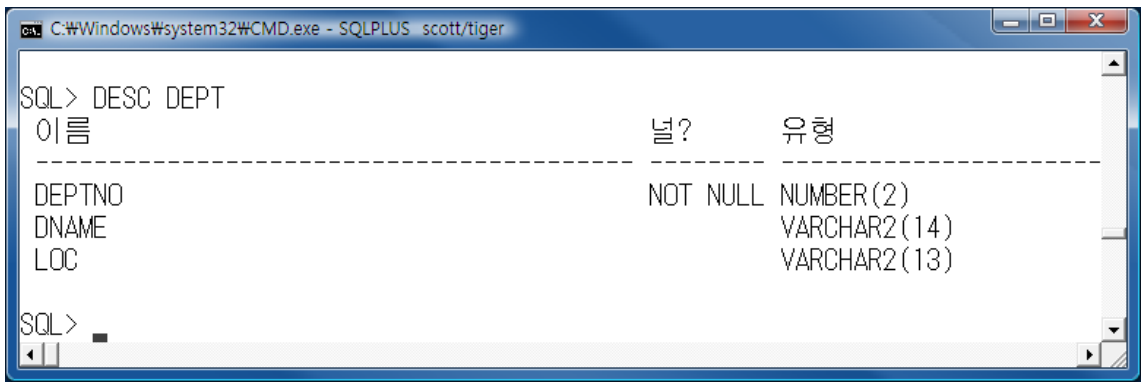
```
INSERT INTO DEPT01  
VALUES (20, 'RESEARCH', 'DALLAS');
```

## 1.3 NULL 값 삽입하는 다양한 방법

- ❖ 데이터를 입력하는 시점에서 해당 컬럼 값을 모르거나 확정되지 않았을 경우에는 NULL값을 입력해야 합니다.
- ❖ NULL 값 삽입은 암시적인 방법과 명시적인 방법이 있습니다.
- ❖ 암시적 방법은 컬럼 명 리스트에 컬럼을 생략하는 것입니다. 즉, 다른 컬럼은 값을 입력하지만 이렇게 생략한 컬럼에는 암시적으로 NULL 값이 할당되는 것입니다.
- ❖ 명시적 방법은 VALUES 리스트에 명시적으로 NULL을 입력합니다.

# 1.3 NULL 값 삽입하는 다양한 방법

- ❖ 부서 테이블에 컬럼이 NULL값을 허용하는지 살펴보기 위해서 DESC 명령을 실행합니다.
- ❖ DEPT 테이블의 DEPTNO 컬럼은 NOT NULL 제약조건이 지정되어 있습니다.



- ❖ NOT NULL 제약조건이 지정된 DEPTNO 컬럼은 널 값을 입력하지 못합니다.
- ❖ 오라클이 제공해 주는 DEPT 테이블의 DEPTNO 컬럼에 널값을 허용하지 못하도록 오라클 내부에서 이미 컬럼에 제약조건을 지정해 놓은 상태입니다.
- ❖ 컬럼에 널값을 허용하지 못하도록 하려면 컬럼에 제약조건을 지정해야 합니다.

# 암시적으로 NULL 값의 삽입

- ❖ 다음은 지역명이 결정되지 않은 30번 부서에 부서명만 입력하려고 합니다.
- ❖ 저장할 값을 명확하게 알고 있는 컬럼 명만 명시적으로 기술한 후에 그에 매칭되는 값을 VALUES 절 다음에 기술합니다.

```
INSERT INTO DEPT01  
(DEPTNO, DNAME)  
VALUES (30, 'SALES');
```

# 명시적으로 NULL 값의 삽입

- ❖ 컬럼명을 명시적으로 기술하지 않으면 테이블이 갖고 있는 모든 컬럼에 값을 지정해야 합니다.

```
INSERT INTO DEPT01  
VALUES (40, 'OPERATIONS', NULL);
```

- ❖ 지역명이 결정되어 지지 않았더라도 반드시 값을 3개 지정해야 하기 때문에 명시적으로 VALUES 리스트에서 지역명에 NULL을 입력해야 합니다.

# 명시적으로 NULL 값의 삽입

- ❖ NULL 값을 갖는 칼럼을 추가하기 위해서 NULL 대신 ''를 사용할 수 있습니다.
- ❖ 이번에는 지역명이 아닌 부서명이 결정되지 않아 부서명에 NULL 값을 입력한 예입니다.

```
INSERT INTO DEPT01  
VALUES (50, '', 'CHICAGO');
```

## 1.4 서브 쿼리로 데이터 삽입하기

- ❖ INSERT INTO 다음에 VALUES 절을 사용하는 대신에 서브 쿼리를 사용할 수 있습니다.
- ❖ 이렇게 하면 기존의 테이블에 있던 여러 행을 복사해서 다른 테이블에 삽입할 수 있습니다.
- ❖ 이 때 주의할 점은 INSERT 명령문에서 지정한 컬럼의 개수나 데이터 타입이 서브 쿼리를 수행한 결과와 동일해야 한다는 점입니다.

# 서브 쿼리로 데이터 삽입하는 예제

1. 서브 쿼리로 데이터 삽입하기 위해서 우선 테이블을 생성하되 데이터는 복사하지 않고 빈 테이블만 생성합시다.

```
DROP TABLE DEPT02;  
CREATE TABLE DEPT02  
AS  
SELECT * FROM DEPT WHERE 1=0;
```

- ❖ 2. 테이블 구조만을 복사해서 내용을 갖지 않는 테이블에 서브 쿼리로 row를 입력해 봅시다.

```
INSERT INTO DEPT02  
SELECT * FROM DEPT;
```



## 02. 다중 테이블에 다중 행 입력하기

- ❖ INSERT ALL문을 사용하면 서브 쿼리의 결과를 조건 없이 여러 테이블에 동시에 입력할 수 있습니다 .
- ❖ 사원번호, 사원명, 입사일자로 구성된 EMP\_HIR 테이블과 사원번호, 사원명, 해당관리자(상관))로 구성된 EMP\_MGR 테이블이 빈 테이블로 존재한다고 합시다.
- ❖ 사원 테이블(EMP)에서 부서 번호가 20인 직원들을 검색하여 EMP\_HIR 테이블에는 사원 번호, 사원 명, 급여를 EMP\_MGR 테이블에는 사원 번호, 사원 명, 해당관리자(상관)를 입력하려면 어떻게 해야 할까요?
- ❖ INSERT ALL 명령문을 사용하면 두 번의 쿼리문을 수행하지 않고도 하나의 쿼리문으로 두 개의 테이블에 원하는 컬럼 값을 삽입할 수 있습니다.

## 02. 다중 테이블에 다중 행 입력하기

- ❖ INSERT ALL 명령문은 서브 쿼리의 결과 집합을 조건 없이 여러 테이블에 동시에 입력하기 위한 명령문입니다.
- ❖ 이때 주의할 점은 서브 쿼리의 컬럼명과 데이터가 입력되는 테이블의 컬럼명이 동일해야 한다는 점입니다.

```
INSERT ALL  
  INTO EMP_HIR VALUES(EMPNO, ENAME, HIREDATE)  
  INTO EMP_MGR VALUES(EMPNO, ENAME, MGR)  
  SELECT EMPNO, ENAME, HIREDATE, MGR  
  FROM EMP  
  WHERE DEPTNO=20;
```

## 2.1 조건(WHEN)에 의해 다중 테이블에 다중 행 입력하기

- ❖ INSERT ALL 명령문에 WHEN 절을 추가해서 조건을 제시하여 조건에 맞는 행만 추출하여 테이블에 추가합니다.
- ❖ EMP\_HIR02 테이블에는 1982 년 01 월01 일 이후에 입사한 직원들의 번호, 직원 명, 입사일을 추가합니다.
- ❖ EMP\_SAL 테이블에는 급여가 2000 이상인 직원들의 번호, 직원 명, 급여를 추가합니다.

```
INSERT ALL
WHEN HIREDATE > '1982/01/01' THEN
  INTO EMP_HIR02 VALUES(EMPNO, ENAME,
    HIREDATE)
WHEN SAL > 2000 THEN
  INTO EMP_SAL VALUES(EMPNO, ENAME, SAL)
SELECT EMPNO, ENAME, HIREDATE, SAL FROM
EMP;
```

### 03. 테이블의 내용을 수정하기 위한 UPDATE 문

- ❖ UPDATE 문은 테이블에 저장된 데이터를 수정하기 위해서 사용합니다.

```
UPDATE table_name  
SET column_name1 = value1, column_name2 = value2, ...  
WHERE conditions;
```

- ❖ UPDATE 문은 기존의 행을 수정하는 것입니다. 따라서 어떤 행의 데이터를 수정하는지 WHERE 절을 이용하여 조건을 지정합니다.
- ❖ WHERE 절을 사용하지 않을 경우는 테이블에 있는 모든 행이 수정됩니다.
- ❖ 정말 테이블의 전체 행을 수정하려고 했던 것이 아니라면 큰 문제가 발생하므로 WHERE 절의 사용 유무를 신중히 판단하여야 합니다.

## 테이블의 모든 행 변경

- ❖ 1. 모든 사원의 부서번호를 30번으로 수정합시다.

```
UPDATE EMP01  
SET DEPTNO=30;
```

- ❖ 2. 이번엔 모든 사원의 급여를 10% 인상시키는 UPDATE 문을 보겠습니다.

```
UPDATE EMP01  
SET SAL = SAL * 1.1;
```

- ❖ 3. 모든 사원의 입사일을 오늘로 수정하려면 다음과 같이 합니다.

```
UPDATE EMP01  
SET HIREDATE = SYSDATE;
```

## 3.2 테이블의 특정 행만 변경

- ❖ UPDATE 문에 WHERE 절을 추가하면 테이블의 특정 행이 변경됩니다.
- ❖ UPDATE 문을 이용하여 테이블의 특정 행을 변경하기 위한 실습을 하기에 앞서서 실습에 사용할 테이블을 먼저 만들자.
- ❖ 이전 실습을 위해서 사용하였던 사원 테이블(EMP01)을 제거한 후 다시 기존에 있던 사원 테이블(EMP)과 동일한 구조와 데이터를 갖는 사원 테이블(EMP01)을 생성합시다.

## 테이블의 특정 행만 변경

1. 부서번호가 10번인 사원의 부서번호를 30번으로 수정합시다.

```
UPDATE EMP01  
SET DEPTNO=30  
WHERE DEPTNO=10;
```

- ❖ 2. 급여가 3000 이상인 사원만 급여를 10% 인상합시다.

```
UPDATE EMP01  
SET SAL = SAL * 1.1  
WHERE SAL >= 3000;
```

## 테이블의 특정 행만 변경

- ❖ 3. 1987년에 입사한 사원의 입사일이 오늘로 수정합시다. 사원의 입사일을 오늘로 수정한 후에 테이블 내용을 살펴봅시다.

```
UPDATE EMP01  
SET HIREDATE = SYSDATE  
WHERE SUBSTR(HIREDATE, 1, 2)='87';
```



### 3.3 테이블에서 2개 이상의 칼럼 값 변경

- ❖ 테이블에서 하나의 칼럼이 아닌 복수 개 칼럼의 값을 변경하려면 기존 SET 절에 콤마를 추가하고 칼럼=값을 추가 기술하면 됩니다.

# 테이블에서 2개 이상의 칼럼 값 변경

1. SCOTT 사원의 부서번호는 20번으로, 직급은 MANAGER로 한꺼번에 수정하도록 합시다.

```
UPDATE EMP01  
SET DEPTNO=20, JOB='MANAGER'  
WHERE ENAME='SCOTT';
```

2. SCOTT 사원의 입사일자는 오늘로, 급여를 50 으로 커미션을 4000 으로 수정합시다.

```
UPDATE EMP01  
SET HIREDATE = SYSDATE, SAL=50, COMM=4000  
WHERE ENAME='SCOTT';
```

## 3.4 서브 쿼리를 이용한 데이터 수정하기

- ❖ UPDATE 문의 SET 절에서 서브 쿼리를 기술하면 서브 쿼리를 수행한 결과로 내용이 변경됩니다.
- ❖ 이러한 방법으로 다른 테이블에 저장된 데이터로 해당 컬럼 값을 변경할 수 있습니다.

# 서브 쿼리를 이용한 데이터 수정하기

- ❖ 1. 20번 부서의 지역명을 40번 부서의 지역명으로 변경하기 위해서 서브 쿼리문을 사용해 봅시다.

```
UPDATE DEPT01  
SET LOC=(SELECT LOC  
          FROM DEPT01  
          WHERE DEPTNO=40)  
WHERE DEPTNO=20;
```

### 3.5 서브 쿼리를 이용한 두개 이상의 칼럼에 대한 값 변경

- ❖ 서브 쿼리를 사용한 UPDATE 형식은 다음과 같이 2가지입니다. 사용면에서 형식 2가 보다 편리합니다.

- ❖ 형식 1

```
UPDATE table_name  
SET column_name1 = (sub_query1),  
    column_name2 = (sub_query2), ...  
WHERE 조건
```

- ❖ 형식 2

```
UPDATE table_name  
SET (column_name1, column_name2, ...) =  
    (sub_query)  
WHERE 조건
```

## 서브 쿼리를 이용한 한꺼번에 두 개의 컬럼 값 변경하기

- ❖ 서브 쿼리를 이용해서 부서번호가 20인 부서의 부서명과 지역명을 부서번호가 40번인 부서와 동일하게 변경하도록 해 봅시다.
- ❖ 1. 부서 번호가 20번인 부서의 이름과 지역은 RESEARCH와 DALLAS입니다. 다음은 부서번호가 20인 부서의 부서명과 지역명을 부서번호가 40번인 부서와 동일하게 변경하기 위한 UPDATE 명령문입니다.

```
UPDATE DEPT01  
SET (DNAME, LOC)=(SELECT DNAME, LOC  
                   FROM DEPT  
                   WHERE DEPTNO=40)  
WHERE DEPTNO=20;
```

## 04. 테이블에 불필요한 행을 삭제하기 위한 DELETE 문

- ❖ DELETE 문은 테이블에 저장되어 있는 데이터를 삭제합니다.

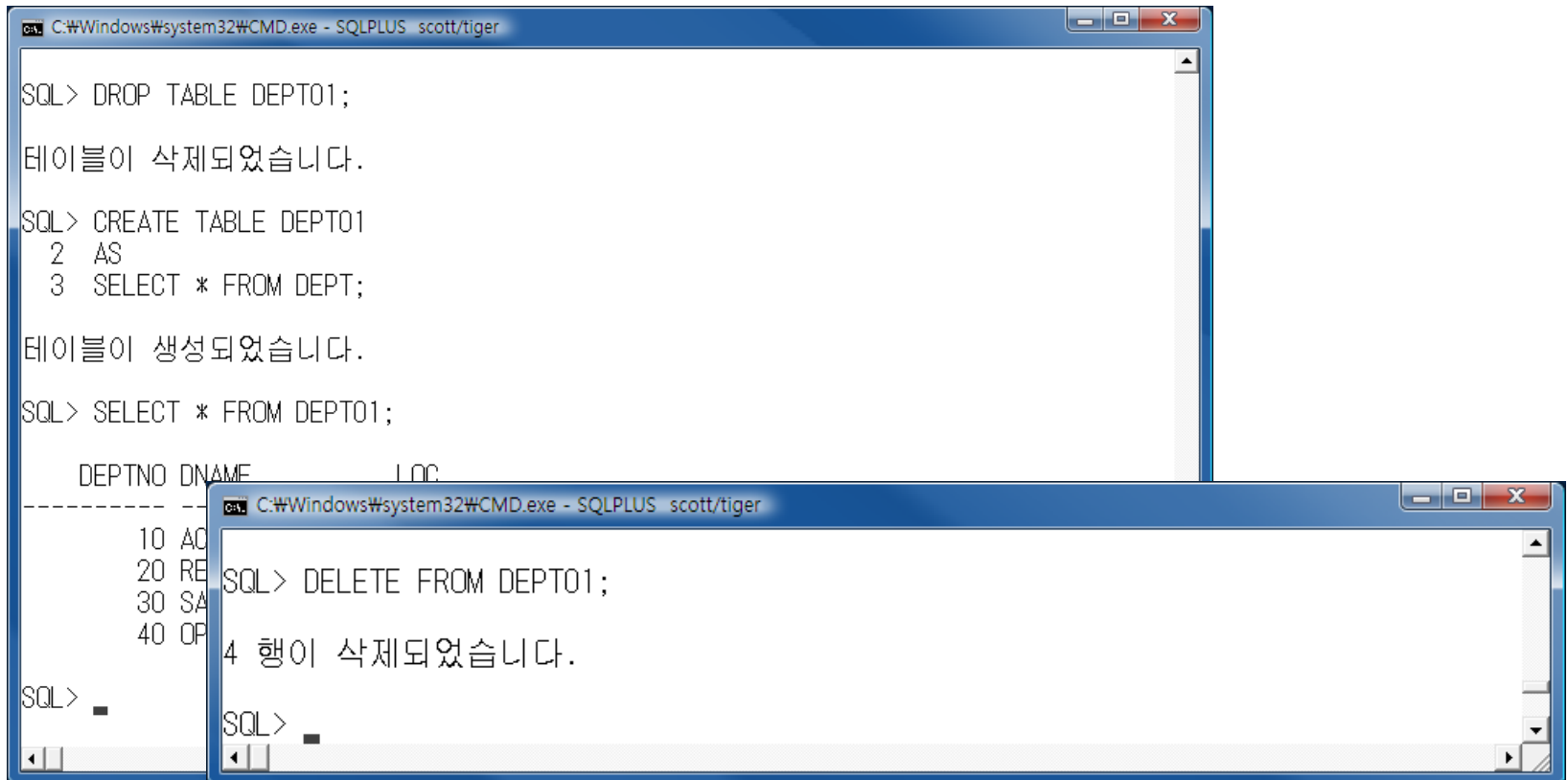
```
DELETE FROM table_name  
WHERE conditions;
```

- ❖ DELETE 문은 테이블의 기존 행을 삭제하며 특정한 로우(행)을 삭제하기 위해서 WHERE 절을 이용하여 조건을 지정합니다.
- ❖ 만약 DELETE 문에 WHERE 절을 사용하지 않을 경우 테이블에 있는 모든 행이 삭제되므로 매우 신중하게 명령문을 사용해야 합니다.

# DELETE 문으로 행을 삭제하기

❖ DELETE 문으로 부서 테이블의 모든 행을 삭제합니다.

**DELETE FROM DEPT01;**



The image shows two overlapping screenshots of an SQL\*Plus command window. The top screenshot shows the process of dropping and recreating the DEPT01 table. The bottom screenshot shows the deletion of all data from the DEPT01 table.

**Top Screenshot:**

```
C:\Windows\system32\CMD.exe - SQLPLUS scott/tiger

SQL> DROP TABLE DEPT01;

테이블이 삭제되었습니다.

SQL> CREATE TABLE DEPT01
  2  AS
  3  SELECT * FROM DEPT;

테이블이 생성되었습니다.

SQL> SELECT * FROM DEPT01;
```

DEPTNO	DNAME	LOC
10	AC	
20	RE	
30	SA	
40	OP	

**Bottom Screenshot:**

```
C:\Windows\system32\CMD.exe - SQLPLUS scott/tiger

SQL> DELETE FROM DEPT01;

4 행이 삭제되었습니다.

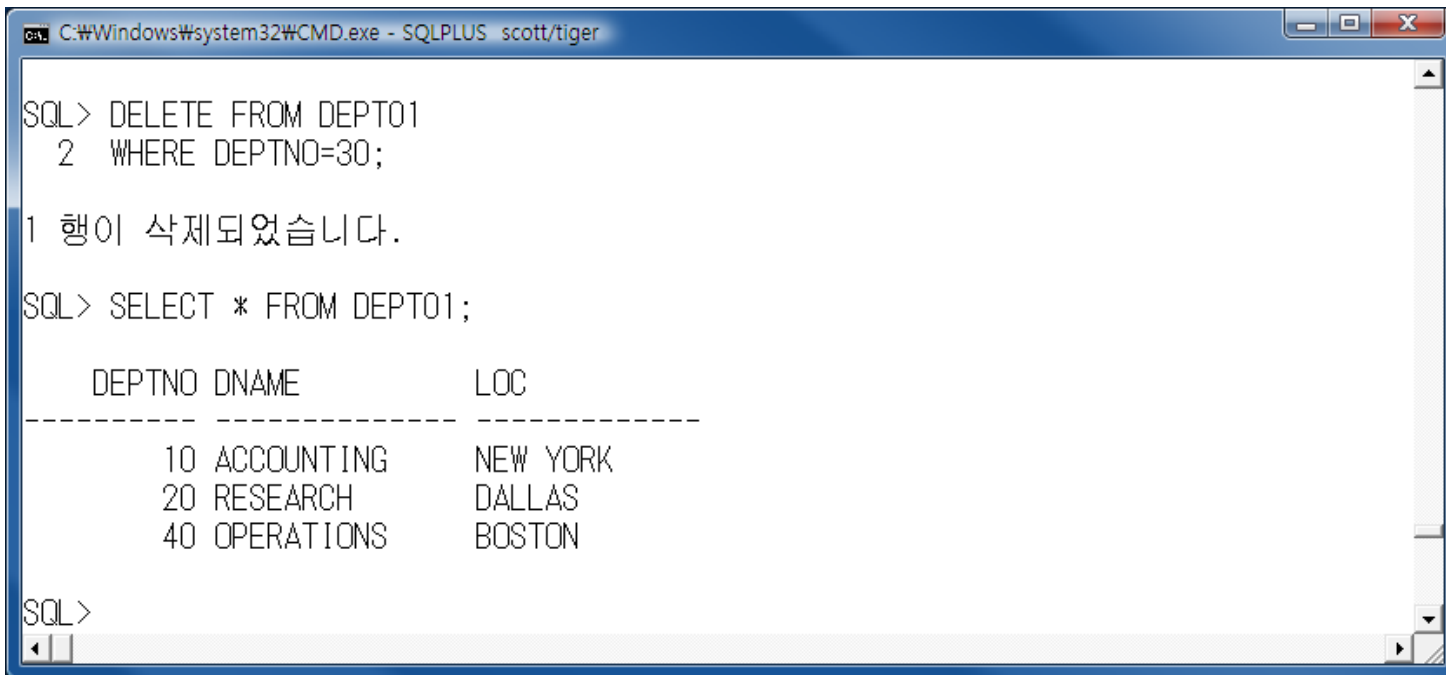
SQL>
```



# 조건을 제시하여 특정 행만 삭제하기

❖ 부서 테이블에서 30번 부서만 삭제합니다.

```
DELETE FROM DEPT01  
WHERE DEPTNO=30;
```



```
SQL> DELETE FROM DEPT01  
2  WHERE DEPTNO=30;  
  
1 행이 삭제되었습니다.  
  
SQL> SELECT * FROM DEPT01;
```

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

```
SQL>
```

## 4.1 서브 쿼리를 이용한 데이터 삭제

- ❖ DELETE 문을 사용하기에 앞서 사원 테이블을 복사합시다. 사원 테이블에서 부서명이 SALES인 사원을 모두 삭제해보도록 하겠습니다.
- ❖ 사원 테이블에는 부서명이 기록되어 있지 않습니다.
- ❖ 부서명은 부서 테이블에 기록되어 있으므로 부서 테이블에서 부서명이 SALES인 부서의 번호부터 알아내야 합니다.
- ❖ 이렇게 알아낸 부서번호를 사원 테이블에 적용하기 위해서는 서브 쿼리를 이용해야 합니다.

# 서브 쿼리를 이용한 데이터 삭제하기

- ❖ 사원 테이블에서 부서명이 SALES인 사원을 모두 삭제해봅시다.

```
DELETE FROM EMP01  
WHERE DEPTNO=(SELECT DEPTNO  
                FROM DEPT  
                WHERE DNAME='SALES');
```