8. 데이터 조작

목적

- 본 과정을 마치면 다음을 할 수 있어야 합니다.
  - 각 DML 문장을 기술합니다.
  - 테이블에 행을 삽입합니다.
  - 테이블에 행을 갱신합니다.
  - 테이블로부터 행을 삭제합니다.

데이터 조작어

- DML 문장은 다음의 경우에 실행됩니다.
  - 테이블에 새로운 행을 추가
  - 테이블에 있는 기존의 행을 변경
  - 테이블로부터 기존의 행을 제거

테이블에 새로운 행 추가

새로운 행

50	DEVELOPMENT	DETROIT

DEPT

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

"새로운 행을 DEPT 테이블에 삽입…"

DEPT

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

INSERT 문장

- INSERT 문장을 사용하여 테이블에 새로운 행을 추가합니다.

```
INSERT INTO table [(column [, column …])]
VALUES            (value [, value …]);
```

- 이 구문형식으로는 한번에 오직 하나의 행만이 삽입됩니다.

새로운 행 삽입

- 각각의 열에 대한 값을 포함하는 새로운 행을 삽입합니다.
- 테이블에 있는 열의 디폴트 순서로 값을 나열합니다.
- INSERT 절에서 열을 선택적으로 나열합니다.

```
SQL> INSERT INTO dept (deptno, dname, loc)
  2  VALUES      (50, 'DEVELOPMENT', 'DETROIT');

1 개의 행이 만들어졌습니다.
```

- 문자와 날짜 값은 단일 따옴표 내에 둡니다.

Null 값을 가진 새로운 행 추가

- 암시적 방법 : 열 목록으로부터 열을 생략합니다.

```
SQL> INSERT  INTO  dept (deptno, dname )
  2  VALUES        (60, 'MIS');

1 개의 행이 만들어졌습니다.
```

- 명시적 방법 : NULL 키워드를 명시합니다.

```
SQL> INSERT  INTO  dept
  2  VALUES        (70, 'FINANCE', NULL);

1 개의 행이 만들어졌습니다.
```

특수 값 삽입

- SYSDATE 함수는 현재 날짜와 시간을 기록합니다.

```
SQL> INSERT INTO emp (empno, ename, job, mgr,
  2                   hiredate, sal, comm, deptno)
  3  VALUES           (7989, 'JUNE', 'SALESMAN', 7782,
  4                   SYSDATE, 2000, NULL, 10);

1 개의 행이 만들어졌습니다.
```

특정 날짜 값 삽입

- 특정 입사일을 지정하여 새로운 종업원을 추가합니다.

```
SQL> INSERT  INTO  emp
  2  VALUES     (7979, 'HYUN', 'SALESMAN', 7782,
  3               TO_DATE('2007/10/24', 'YYYY/MM/DD'),
  4               1300, NULL, 10);

1 개의 행이 만들어졌습니다.
```

- 추가 내용을 확인합니다.

```
     EMPNO ENAME      JOB       HIREDATE        COMM
---------- ---------- --------- -------- ----------
      7979 HYUN       SALESMAN  07/10/24
```

치환 변수를 이용한 데이터 입력

- 치환변수는 사용자로부터 입력 받은 값을 SQL 문장간에 전달하기 위해 사용되는 변수입니다.
- 치환변수는WHERE절, ORDER BY절, COLUMN 표현식, 테이블이름, 전체 SELECT문에 올 수 있습니다.

종류	설명
&치환변수	SQL문장이 실행될 때마다 새로운 값을 요구합니다.
&&치환변수	처음 SQL문이 실행되어 입력된 치환변수 값이 다음 실행시에도 계속 사용됩니다.
ACCEPT명령어와 &치환변수	PROMPT 문장에 의한 메시지와 함께 치환변수를 입력합니다.

치환변수를 이용한 데이터 입력 사용

- 치환변수를 이용해 부서 테이블에 정보를 입력합니다.

```
SQL> SQL> INSERT INTO dept(deptno, dname, loc)
  2  VALUES (&department, '&name', '&location');
```

- 추가 내용을 확인합니다.

```
department의 값을 입력하십시오: 50
name의 값을 입력하십시오: 영업부
location의 값을 입력하십시오: 서울
구   2: VALUES (&department, '&name', '&location')
신   2: VALUES (50, '영업부', '서울')

1 개의 행이 만들어졌습니다.
```

다른 테이블로부터 행 복사

- 서브쿼리로 INSERT 문장을 작성합니다.

```
SQL> INSERT INTO managers (id, name, salary, hiredate)
  2       SELECT empno, ename, sal, hiredate
  3       FROM   emp
  4       WHERE  job='MANAGER';

3 개의 행이 만들어졌습니다.
```

- VALUES 절을 사용하지 않습니다.
- 서브쿼리의 열 수와 INSERT 절의 열 수는 일치해야 합니다.

MULTIPLE INSERT 문

- 하나의 INSERT문에서 여러 개의 테이블에 동시에 하나의 행을 입력할 수 있습니다.

```
INSERT [ALL]
[WHEN  조건절1  THEN
  INTO [table1] VALUES (col1, col2, …)]
[WHEN  조건절2  THEN
  INTO [table2] VALUES (col1, col2, …)]
[ELSE
  INTO [table3] VALUES (col1, col2, …)]
[Sub-Query];
```

MULTIPLE INSERT 문 – UNCONDITIONAL INSERT ALL

- 조건과 상관없이 기술되어진 여러 개의 테이블에 데이터를 입력합니다.

```
SQL> CREATE TABLE dept1 AS SELECT * FROM dept WHERE 1=2;

SQL> CREATE TABLE dept2 AS SELECT * FROM dept WHERE 1=2;

SQL> INSERT ALL
  2  INTO dept1 VALUES(deptno, dname, loc)
  3  INTO dept2(deptno, dname) VALUES(deptno, dname)
  4  SELECT * FROM dept;

SQL> SELECT * FROM dept1;
```

MULTIPLE INSERT 문 – CONDITIONAL INSERT ALL

- 특정 조건들을 기술하여 그 조건에 맞는 행들을 원하는 테이블에 나누어 삽입합니다.

```
SQL> CREATE TABLE dept3 AS SELECT * FROM dept WHERE 1=2;

SQL> CREATE TABLE dept4 AS SELECT * FROM dept WHERE 1=2;

SQL> INSERT ALL
  2  WHEN deptno <= 20 THEN
  3       INTO dept3 VALUES(deptno, dname, loc)
  4  WHEN deptno > 20 THEN
  5       INTO dept4(deptno, dname) VALUES(deptno, dname)
  6  SELECT deptno, dname, loc FROM dept;

SQL> SELECT * FROM dept1;
```

MULTIPLE INSERT 문 – UNCONDITIONAL FIRST INSERT

- 첫 번째 when 절에서 조건을 만족할 경우 다음의 when 절은 수행하지 않습니다.

```
SQL> CREATE TABLE dept2 AS SELECT * FROM dept WHERE 1=2;
SQL> CREATE TABLE dept3 AS SELECT * FROM dept WHERE 1=2;
SQL> CREATE TABLE dept4 AS SELECT * FROM dept WHERE 1=2;

SQL> INSERT FIRST
  2  WHEN deptno <= 20 THEN
  3       INTO dept2 VALUES(deptno, dname, loc)
  4  WHEN deptno = 30 THEN
  5       INTO dept3(deptno, loc) VALUES(deptno, loc)
  6  ELSE
  5       INTO dept4(deptno, dname) VALUES(deptno, dname)
  6  SELECT * FROM dept;
```

MULTIPLE INSERT 문 – PIVOTING INSERT

- 여러 개의 into 절을 사용할 수 있지만, into 절 뒤에 오는 테이블은 모두 동일해야 합니다.
- 비 관계형 데이터베이스를 관계형 데이터베이스 구조로 만들때 사용합니다.

```
SQL> INSERT ALL
  2  INTO sales_data VALUES(employee_id, week_id, sales_mon)
  3  INTO sales_data VALUES(employee_id, week_id, sales_tue)
  4  INTO sales_data VALUES(employee_id, week_id, sales_wed)
  5  INTO sales_data VALUES(employee_id, week_id, sales_thu)
  6  INTO sales_data VALUES(employee_id, week_id, sales_fri)
  7  SELECT employee_id, week_id, sales_mon, sales_tue,
  8         sales_wed, sales_thu, sales_fri
  9  FROM sales;
```

테이블 데이터 변경

EMP

EMPNO	ENAME	JOB	...	DEPTNO
7839	KING	PRESIDENT		10
7698	BLAKES	MANAGER		30
7782	CLARK	MANAGER		10
7566	JONES	MANAGER		20
...				

"EMP 테이블의 행을 갱신..."

EMP

EMPNO	ENAME	JOB	...	DEPTNO
7839	KING	PRESIDENT		10
7698	BLAKES	MANAGER		30
7782	CLARK	MANAGER		20
7566	JONES	MANAGER		20
...				

UPDATE 문장

- UPDATE 문장으로 기존의 행을 갱신합니다.

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

- 필요하다면 하나 이상의 행을 갱신할 수 있습니다.

테이블 행 갱신

- 특정 열이나 행은 WHERE 절을 명시함으로써 수정될 수 있습니다.

```
SQL> UPDATE  emp
  2  SET     deptno = 20
  3  WHERE   empno = 7782;

1 행이 갱신되었습니다.
```

- WHERE 절을 생략하며 테이블에 있는 모든 행이 수정됩니다.

```
SQL>  UPDATE  employee
  2   SET     deptno = 20;

14 행이 갱신되었습니다.
```

다중 열 서브쿼리로 갱신

- 종업원 7499의 업무와 부서에 일치하도록
  종업원 7698의 업무와 부서를 갱신합니다.

```
SQL> UPDATE  emp
  2  SET     (job, deptno) =
  3                          (SELECT job, deptno
  4                           FROM   emp
  5                           WHERE  empno = 7499)
  6  WHERE   empno = 7698;

1 행이 갱신되었습니다.
```

테이블로부터 행 제거

DEPT

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

"DEPT 테이블로부터 행을 삭제…"

DEPT

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

DELETE 문장

- DELETE 문장을 사용하여 테이블로부터 기존의 행을 제거할 수 있습니다.

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

- 참조 무결성 제약 조건에 주의해야 합니다.

테이블로부터 행 삭제

- WHERE 절을 명시하여 특정 행이나 행들을 삭제할 수 있습니다.

```
SQL> DELETE  FROM  department
  2  WHERE         dname = 'DEVELOPMENT';

1 개의 행이 삭제되었습니다.
```

- WHERE 절을 생략하면 테이블의 모든 행이 삭제됩니다.

```
SQL> DELETE  FROM  department;

4 개의 행이 삭제되었습니다.
```

다른 테이블을 근거로 한 행 삭제

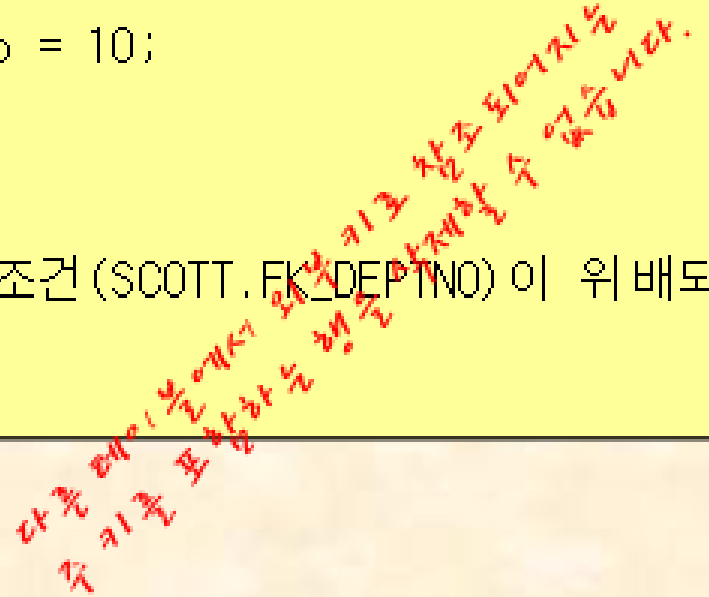
- 다른 테이블의 값을 근거로 테이블로부터 행을 제거하기 위해서 DELETE 문장에서 서브쿼리를 사용할 수 있습니다.

```
SQL> DELETE FROM employee
  2  WHERE       deptno =
  3                      (SELECT  deptno
  4                       WHERE   dept
  5                       WHERE   dname = 'SALES');

6 개의 행이 삭제되었습니다.
```

행 삭제 : 무결성 제약조건 에러

```
SQL> DELETE FROM dept
  2  WHERE       deptno = 10;
DELETE FROM dept
*
1행에 오류:
ORA-02292: 무결성 제약조건(SCOTT.FK_DEPTNO)이 위배되었습
니다- 자식 레코드가
발견되었습니다
```



RETURNING 절(10g)

- DML 조작시 특정 값을 임시로 저장할 수 있습니다.

```
SQL> VARIABLE TOT_SAL NUMBER;
SQL> DELETE emp
  2  WHERE  deptno=30
  3  RETURNING SUM(sal) INTO :tot_sal

6 행이 삭제되었습니다.

SQL> PRINT tot_sal;

   TOT_SAL
----------
      9400
```

MERGE 문(9i)

- INSERT문과 UPDATE문을 동시에 사용할 수 있습니다.
- Oracle 9i에 추가되었습니다.

```
MERGE INTO  table_name [alias]
  USING (target|view|subquery) [alias]
    ON (join_condition)
  WHEN MATCHED THEN
    UPDATE SET col1=val1[, …]
  WHEN NOT MATCHED THEN
    INSERT (column lists) VALUES (value lists);
```

MERGE 문(9i) 사용

```
SQL> CREATE TABLE copy_emp AS SELECT * FROM emp WHERE 1=2;
SQL> INSERT INTO copy_emp (empno, ename) VALUES(7369, 'SMITH');
SQL> COMMIT;

SQL> MERGE INTO copy_emp c
  2    USING emp e
  3    ON (c.empno = e.empno)
  4  WHEN MATCHED THEN
  5    UPDATE SET
  6      c.ename = e.ename,
  7      c.job = e.job,
  8      c.mgr = e.mgr,
  9      c.hiredate = e.hiredate,
 10      c.sal = e.sal,
 11      c.comm = e.comm,
 12      c.deptno = e.deptno
 13  WHEN NOT MATCHED THEN
 14    INSERT VALUES(e.empno, e.ename, e.job, e.mgr,
 15                  e.hiredate, e.sal, e.comm, e.deptno);
```