10 DML

10.1 테이블에 새로운 행을 추가하는 INSERT문

- INSERT 문은 테이블에 새로운 데이터를 입력하기 위해 사용하는 데이터 조작어이다.

```
-- 형식
INSERT INTO table_name
(column_name, …)
VALUES(column_value, …);

-- 예
INSERT INTO DEPT01
VALUES(10, 'ACCOUNTING', 'NEW YORK');

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

[실습] INSERT문 실습에 사용할 테이블 생성하기

- INSERT문을 위한 실습을 하기에 앞서 실습에 사용할 테이블을 CREATE TABLE 명령어로 새롭게 만들어 보겠다.

```
-- 기존에 있던 부서 테이블(DEPT)과 동일한 구조를 갖되 데이터는 복사하지 않는 테이블을 생성한다.
DROP TABLE DEPT01;
CREATE TABLE DEPT01
AS
SELECT * FROM DEPT WHERE 1=0;

-- 생성된 테이블의 구조를 살펴본다.
DESC DEPT01

-- 수록된 데이터를 살펴본다.
SELECT * FROM DEPT01;
```

10.1.1 INSERT 구문의 오류 발생 예

- 칼럼 명에 기술된 목록의 수보다 VALUES 다음에 나오는 괄호 안에 기술한 값의 개수가 적으면 에러가 발생한다.
- 칼럼 명에 기술된 목록의 수보다 VALUES 다음에 나오는 괄호에 기술한 값의 개수가 많으면 에러가 발생한다.
- 칼럼 명이 잘못 입력되었을 때에도 에러가 발생한다.
- 칼럼과 입력할 값의 데이터 타입이 서로 맞지 않을 경우에도 에러가 발생한다.

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

INSERT INTO DEPT01
(DEPTNO, DNAME, LOC)
```

```
VALUES(10, 'ACCOUNTING', 'NEW YORK', 20);

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

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

10.1.2 컬러명을 생략한 INSERT 구문

- 테이블에 로우를 추가할 때 모든 칼럼에 모두 자료를 입력하는 경우에는 굳이 칼럼 목록을 기술하지 않아도 된다.
- 칼럼 목록을 생략하면 VALUES 절 다음의 값들을 테이블의 기본 칼럼 순서대로 입력한다.
- 테이블의 컬럼 순서는 CREATE TABLE로 테이블을 생성할 때의 순서를 따른다.
- 테이블의 기본 칼럼 순서는 DESC 문으로 조회했을 때 보여 지는 순서이다.

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

SELECT * FROM DEPT01;
```

10.1.3 NULL 값을 삽입하는 다양한 방법

- 부서 테이블에 컬럼이 NULL값을 허용하는지 살펴보기 위해서 DESC 명령을 실행한다.
- DEPT 테이블의 DEPTNO 컬럼은 NOT NULL 제약조건이 지정되어 있다.
- NOT NULL 제약조건이 지정된 DEPTNO 컬럼은 널 값을 입력하지 못한다.
- 오라클이 제공해 주는 DEPT 테이블의 DEPTNO 컬럼에 널값을 허용하지 못하도록 오라클 내부에서 이미 컬럼에 제약조건을 지정해 놓은 상태이다.
- 컬럼에 널값을 허용하지 못하도록 하려면 컬럼에 제약조건을 지정해야 한다.

(1) 암시적으로 NULL 값의 삽입

- 저장할 값을 명확하게 알고 있는 컬럼 명만 명시적으로 기술한 후에 그에 매칭되는 값을 VALUES 절 다음에 기술한다.

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

(2) 명시적으로 NULL 값의 삽입

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

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

10.1.4 서브 쿼리로 데이터 삽입하기

- INSERT INTO 다음에 VALUES 절을 사용하는 대신에 서브 쿼리를 사용할 수 있다.
- 주의할 점은 INSERT 명령문에서 지정한 컬럼의 개수나 데이터 타입이 서브 쿼리를 수행한 결과와 동일해야 한다.

[실습] 서브 쿼리로 데이터를 삽입하는 예제

- INSERT INTO 다음에 VALUES 절을 사용하는 대신 서브 쿼리를 사용하여 데이터를 추가한다.

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

SELECT * FROM DEPT02;

INSERT INTO DEPT02
SELECT * FROM DEPT;

SELECT * FROM DEPT02;
```

10.2 다중 테이블에 다중행 입력하기

- INSERT ALL 명령문은 서브 쿼리의 결과 집합을 조건 없이 여러 테이블에 동시에 입력하기 위한 명령문입니다.

[실습] INSERT ALL 명령문으로 다중 테이블에 다중행 입력하기

```
DESC EMP
SELECT * FROM EMP
WHERE DEPTNO=20;

DROP TABLE EMP_HIR;
CREATE TABLE EMP_HIR
AS
SELECT EMPNO, ENAME, HIREDATE FROM EMP
WHERE 1=0;
SELECT * FROM EMP_HIR;

DROP TABLE EMP_MGR;
CREATE TABLE EMP_MGR
AS
SELECT EMPNO, ENAME, MGR FROM EMP
WHERE 1=0;
SELECT * FROM EMP_MGR;

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;

SELECT * FROM EMP_HIR;
SELECT * FROM EMP_MGR;
```

[실습] INSERT ALL 명령에 조건(WHEN)으로 다중 테이블에 다중행 입력하기

```
DROP TABLE EMP_HIR02;
CREATE TABLE EMP_HIR02
AS
SELECT EMPNO, ENAME, HIREDATE FROM EMP
WHERE 1=0;
SELECT * FROM EMP_HIR02;

DROP TABLE EMP_SAL;
CREATE TABLE EMP_SAL
AS
SELECT EMPNO, ENAME, SAL FROM EMP
WHERE 1=0;
SELECT * FROM EMP_SAL;

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;

SELECT * FROM EMP_HIR02;
SELECT * FROM EMP_SAL;
```

10.3 테이블의 내용을 수정하기 위한 UPDATE문

■ UPDATE 문은 테이블에 저장된 데이터를 수정하기 위해서 사용한다.

```
-- 형식
UPDATE table_name
SET column_name1 = value1, column_name2 = value2, …
WHERE conditions;
```

10.3.1 테이블의 모든 행 변경

```
DROP TABLE EMP01;
DROP TABLE EMP01 CASCADE CONSTRAINTS;
CREATE TABLE EMP01
AS
SELECT * FROM EMP;
SELECT * FROM EMP01;

UPDATE EMP01
```

```
SET DEPTNO=30;
SELECT * FROM EMP01;

UPDATE EMP01
SET SAL = SAL * 1.1;
SELECT * FROM EMP01;

UPDATE EMP01
SET HIREDATE = SYSDATE;
SELECT * FROM EMP01;
```

10.3.2 테이블의 특정 행만 변경

■ UPDATE 문에 WHERE 절을 추가하면 테이블의 특정 행이 변경된다.

```
DROP TABLE EMP01;
CREATE TABLE EMP01
AS
SELECT * FROM EMP;
SELECT * FROM EMP01;

UPDATE EMP01
SET DEPTNO=30
WHERE DEPTNO=10;
SELECT * FROM EMP01;

UPDATE EMP01
SET SAL = SAL * 1.1
WHERE SAL >= 3000;
SELECT * FROM EMP01;

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

10.3.3 테이블에서 2개 이상의 컬럼값 변경

■ 테이블에서 하나의 칼럼이 아닌 복수 개 칼럼의 값을 변경하려면 기존 SET 절에 콤마를 추가하고 칼럼=값을 추가 기술하면 된다.

```
DROP TABLE EMP01;
CREATE TABLE EMP01
AS
SELECT * FROM EMP;
SELECT * FROM EMP01;

UPDATE EMP01
SET DEPTNO=20, JOB='MANAGER'
WHERE ENAME='SCOTT';
SELECT * FROM EMP01 WHERE ENAME='SCOTT';

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

10.3.4 서브 쿼리를 이용한 데이터 수정하기

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

```
-- 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

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

10.4 테이블의 불필요한 행을 삭제하기 위한 DELETE문

10.4.1 DELETE문을 이용한 행 삭제

- DELETE 문은 테이블의 기존 행을 삭제하며 특정한 로우(행)을 삭제하기 위해서 WHERE 절을 이용하여 조건을 지정한다.

```
-- 형식
DELETE FROM table_name
WHERE conditions;

-- 예
DELETE FROM DEPT01;
```

10.4.2 조건을 제시하여 특정 행만 삭제하는 법

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

10.4.3 서브 쿼리를 이용한 데이터 삭제

.

```
DELETE FROM DEPT01
WHERE DEPTNO=(SELECT DEPTNO
              FROM DEPT
```

```
            WHERE DNAME='SALES');
```

10.5 테이블을 합병하는 MERGE

- MERGE는 합병이란 의미이므로 구조가 같은 두개의 테이블을 하나의 테이블로 합치는 기능을 한다.
- MERGE 명령을 수행하기 위해서 수행하는 테이블에 기존에 존재하는 행이 있다면 새로운 값으로 갱신(UPDATE)되고, 존재하지 않으면 새로운 행으로 추가(INSERT)된다.

```
DROP TABLE EMP01;
CREATE TABLE EMP01
AS
SELECT * FROM EMP;
SELECT * FROM EMP01;

DROP TABLE EMP02;
CREATE TABLE EMP02
AS
SELECT * FROM EMP
WHERE JOB='MANAGER';
SELECT * FROM EMP02;

UPDATE EMP02
SET JOB='TEST';
SELECT * FROM EMP02;

INSERT INTO EMP02
VALUES(8000, 'SYJ', 'TOP', 7566, '2009/01/12', 1200, 10, 20);

MERGE INTO EMP01
USING EMP02
ON(EMP01.EMPNO=EMP02.EMPNO)
WHEN MATCHED THEN
UPDATE SET
EMP01.ENAME=EMP02.ENAME,
EMP01.JOB=EMP02.JOB,
EMP01.MGR=EMP02.MGR,
EMP01.HIREDATE=EMP02.HIREDATE,
EMP01.SAL=EMP02.SAL,
EMP01.COMM=EMP02.COMM,
EMP01.DEPTNO=EMP02.DEPTNO
WHEN NOT MATCHED THEN
INSERT VALUES(EMP02.EMPNO, EMP02.ENAME, EMP02.JOB,
EMP02.MGR, EMP02.HIREDATE, EMP02.SAL,
EMP02.COMM, EMP02.DEPTNO);

SELECT * FROM EMP01;
```

[과제] 과제-10-01.TXT

SQL> CONN ORA_USER/HONG 로 접속하여 SQL문을 작성하세요.

1. ex3_6란 테이블을 만들고, 사원테이블(employees)에서 관리자사번이 124번이고 급여가 2000에서 3000 사이에 있는 사원의 사번, 사원명, 급여, 관리자사번을 입력하는 INSERT문을 작성하라.

〈정답〉

2. 다음 문장을 실행해보자.

```
CREATE TABLE EX3_3 (
       EMPLOYEE_ID NUMBER,
       BONUS_AMT   NUMBER DEFAULT 0);
INSERT INTO EX3_3 VALUES(148,0);
INSERT INTO EX3_3 VALUES(153,0);
INSERT INTO EX3_3 VALUES(154,0);
INSERT INTO EX3_3 VALUES(155,0);
INSERT INTO EX3_3 VALUES(161,0);
COMMIT;
```

관리자사번(manager_id)이 145번인 사원을 찾아 위 테이블에 있는 사원의 사번과 일치하면 보너스 금액(bonus_amt)에 자신의 급여의 1%를 보너스로 갱신하고, 사원의 사번과 일치하지 않는 사원을 EX3_3 테이블에 신규 입력 (이때 보너스 금액은 급여의 0.5%로 한다) 하는 MERGE 문을 작성하라.

〈정답〉