다양한 예제로 쉽게 배우는

오라클 SQL 과 PL/SQL

6장. DML을 배웁니다

6. DML을 배웁니다

- SQL 명령어들

•**DML** (Data Manipulation Language) : INSERT(입력) , UPDATE(변경) , DELETE(삭제) , MERGE(병합)

•**DDL** (Data Definition Language) : CREATE (생성) , ALTER (수정) , TRUNCATE (잘라내기) ,DROP (삭제)

* **DCL** (Data Control Language) : GRANT (권한 주기) , REVOKE (권한 뺏기)

* **TCL** (Transaction Control Language): COMMIT (확정) , ROLLBACK (취소)

* **SELECT** : 어떤 분류에서는 DQL (Data Query Language) 라고 하기도 합니다.

6. DML을 배웁니다

1. INSERT : 데이터 입력 명령어

1) INSERT 를 사용하여 단일 행 입력하기

```
INSERT INTO  table [(column1, column2,……)]
VALUES (value 1 , value 2,….) ;
```

- 사용 예 1:
Dept2 테이블에 아래와 같은 내용으로 새로운 부서 정보를 입력하세요.

* 부서번호 : 9000
* 부서명 : 특판1팀
* 상위부서 : 영업부
* 지역 : 임시지역

```
SCOTT>INSERT INTO dept2(dcode , dname , pdept ,area )
  2  VALUES (9000 , '특판1팀','영업부','임시지역') ;
```

```
SCOTT>INSERT INTO dept2
  2  VALUES(9001 , '특판2팀','영업부','임시지역') ;
```

6. DML을 배웁니다

- 사용 예 2: 특정 칼럼만 입력하기

부서번호와 부서명, 상위부서 값만 아래의 값으로 입력하세요.

* 부서번호 : 9002
* 부서명 : 특판3팀
* 상위부서 : 영업부

```
SCOTT>INSERT INTO dept2(dcode,dname,pdept)
  2  VALUES(9002, '특판3팀' , '영업부') ;
```

6. DML을 배웁니다

- 사용 예 3: 날짜 데이터 입력하기

아래 정보를 professor 테이블에 입력하세요.

* 교수번호 : 5001
* 교수이름 : 김설희
* ID : Love_me
* POSITION : 정교수
* PAY : 510
* 입사일 : 2011년 11월 14일 <- 이 부분을 주의 깊게 보세요.

```
SCOTT>INSERT INTO professor (profno , name , id , position , pay , hiredate)
  2  VALUES (5001,'김설희','Love_me','정교수',510,'2011-11-14');
```

- 윈도 용과 유닉스 용은 날짜 포맷이 다르므로 주의해야 함.

6. DML을 배웁니다

- 사용 예 4: Null 값 입력하기

* 자동 NULL 값 입력하기
데이터를 입력할 때 칼럼에 값을 안 주면 자동으로 NULL 값이 들어 갑니다.

* 수동 NULL 값 입력하기
데이터부분에 NULL 값을 적어주면 됩니다.

6. DML을 배웁니다

2) INSERT 를 사용하여 여러 행 입력하기

```
SCOTT>CREATE TABLE professor2
  2  AS SELECT * FROM professor ;
```

실습을 위해 professor2 테이블을 생성합니다.

```
SQL> INSERT INTO professor2
 2   SELECT * FROM  professor ;
```

이 방식은 이미 생성되어 있는 테이블에서 대량의 데이터를 복사 해 올 때 아주 많이 사용하는 방법입니다. ITAS 라고 부르기도 합니다.

6. DML을 배웁니다

3) INSERT ALL 을 이용한 여러 테이블에 여러 행 입력하기

- 사용 예 1 : 다른 테이블에 한꺼번에 데이터 입력하기

```
SCOTT> INSERT ALL
  2   INTO p_01 (no , name)
  3     VALUES (1,'AAA')
  4   INTO p_02 (no , name)
  5     VALUES(2,'BBB')
  6   SELECT  *  FROM  dual ;
```

이 예제는 p_01 테이블과 P_02 테이블에 각각 서로 다른 데이터를 동시에 입력하는 방법을 보여주고 있습니다.

6. DML을 배웁니다

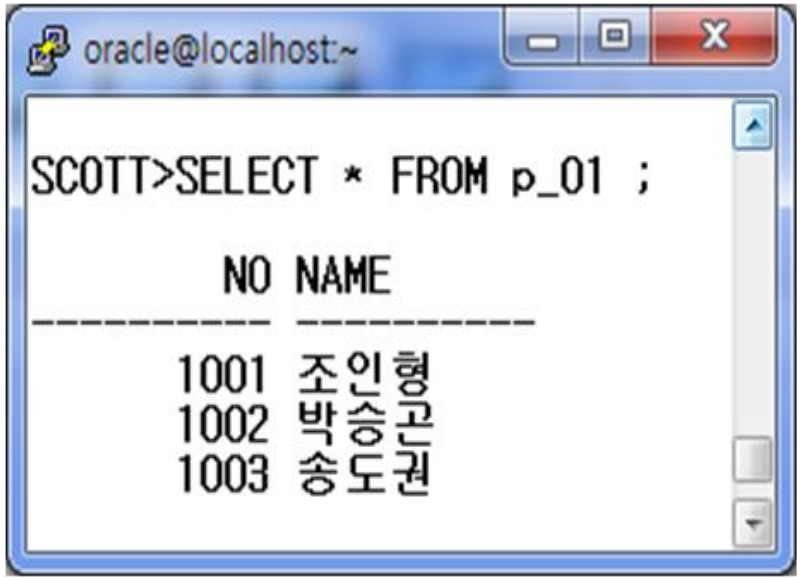
- 사용 예 2 : 다른 테이블의 데이터를 가져와서 입력하기

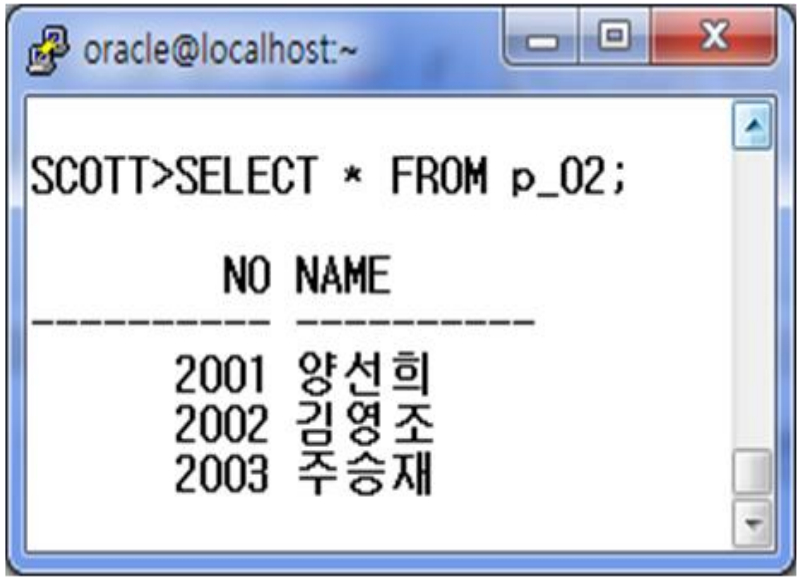
Professor 테이블에서 교수번호가 1000 번 에서 1999번까지 인 교수의 번호와 교수이름은 p_01 테이블에 입력하고 교수번호가 2000 번에서 2999 번까지 인 교수의 번호와 이름은 p_02 테이블에 입력하세요.

```
SCOTT>INSERT ALL
  2  WHEN profno BETWEEN 1000 AND 1999 THEN
  3    INTO p_01 VALUES(profno,name)
  4  WHEN profno BETWEEN 2000 AND 2999 THEN
  5    INTO p_02 VALUES(profno,name)
  6  SELECT profno,name
  7  FROM professor ;
```

6. DML을 배웁니다

- 사용 예 2번 결과 화면





6. DML을 배웁니다

- 사용 예 3 : 다른 테이블에 동시에 같은 데이터 입력하기

Professor 테이블에서 교수번호가 3000번 에서 3999 번인 교수들의 교수 번호와 이름을 p_01 테이블과 p_02 테이블에 동시에 입력하세요.

```
SCOTT>INSERT ALL
  2      INTO p_01 VALUES (profno,name)
  3      INTO p_02 VALUES (profno,name)
  4   SELECT profno,name
  5   FROM professor
  6   WHERE profno BETWEEN 3000 AND 3999 ;
```

6. DML을 배웁니다

2. UPDATE (데이터 변경하기)

```
UPDATE  table
SET  column = value
WHERE  조건 ;
```

- 사용 예 1:

Professor 테이블에서 직급이 조교수 인 교수들의 BONUS 를 100 만원 으로 인상하세요.

```
SCOTT>UPDATE  professor
  2  SET bonus = 100
  3  WHERE position ='조교수';
```

6. DML을 배웁니다

- 사용 예 2:

Professor 테이블에서 차범철 교수의 직급과 동일한 직급을 가진 교수들 중 현재 급여가 250 만원이 안 되는 교수들의 급여를 15% 인상하세요.

```
SCOTT>UPDATE professor
  2  SET pay = pay * 1.15
  3  WHERE position = ( SELECT position
  4                     FROM professor
  5                     WHERE name = '차범철')
  6  AND  pay < 250 ;
```

6. DML을 배웁니다

3. DELETE (데이터 삭제하기)

```
DELETE   FROM   table
WHERE   조건   ;
```

- 사용 예 :

Dept2 테이블에서 부서번호(DCODE)가 9000 번에서 9100 번 사이인 매장들을 삭제하세요.

```
SCOTT>DELETE FROM dept2
  2  WHERE dcode between 9000 and 9100 ;
```

DELETE 는 데이터는 삭제되나 용량은 변함이 없다는 것 !!!

6. DML을 배웁니다

-Table Reorg 하기 (DELETE 후 용량 줄이기)

```
SCOTT>CONN  /  AS  SYSDBA ;
SYS> CREATE TABLE scott.test01 (
  2    no NUMBER, name  VARCHAR2(20),  addr VARCHAR2(20));

SYS>BEGIN
  2    FOR  i  IN  1..500000  LOOP
  3      INSERT  INTO  scott.test01
  4      VALUES ( i, DBMS_RANDOM.STRING('A',19) ,
  5                  DBMS_RANDOM.STRING('Q',19) );
  6    END LOOP;
  7    COMMIT;
  8  END;
  9  /
PL/SQL procedure successfully completed.
```

6. DML을 배웁니다

- 테이블 크기 확인

```
SYS>SELECT COUNT(*) FROM SCOTT.TEST01;

  COUNT(*)
----------
    500000   <- 50만 건의 데이터가 확인됩니다.

SYS>ANALYZE TABLE scott.test01 COMPUTE STATISTICS ;
Table analyzed.

SYS>SELECT SUM(BYTES)/1024/1024 MB
  2  FROM DBA_SEGMENTS
  3  WHERE OWNER='SCOTT'
  4  AND SEGMENT_NAME='TEST01';

        MB
----------
        28    <- 테이블 크기가 28 MB 로 확인됩니다.
```

6. DML을 배웁니다

```
SYS>SELECT table_name, num_rows, blocks, empty_blocks
  2  FROM dba_tables
  3  WHERE owner='SCOTT'
  4  AND  table_name='TEST01';

TABLE_NAME        NUM_ROWS      BLOCKS    EMPTY_BLOCKS
----------------  ------------------- ----------  ----------------
TEST01                      500000       3520              64
```

```
SYS> SELECT  COUNT(DISTINCT  DBMS_ROWID.ROWID_BLOCK_NUMBER(rowid) ||
 2                    DBMS_ROWID.ROWID_RELATIVE_FNO(rowid)) "실사용 블록수"
 3 FROM  scott.test01 ;

실사용 블록수
---------------
        3447   ← 실제 사용하고 있는 블록 개수입니다.
```

6. DML을 배웁니다

```
SYS>DELETE  FROM  SCOTT.TEST01;
500000 rows deleted.

SYS>COMMIT;
Commit complete.

SYS>SELECT COUNT(*) FROM SCOTT.TEST01;

  COUNT(*)
------------
         0  <- 모든 데이터가 전부 삭제됨이 확인됩니다.
```

6. DML을 배웁니다

```
SYS>SELECT SUM(BYTES)/1024/1024 MB
  2  FROM DBA_SEGMENTS
  3  WHERE OWNER='SCOTT'
  4  AND SEGMENT_NAME='TEST01';

        MB
----------
        28    <- 용량은 변함없이 그대로 입니다.

SYS>SELECT COUNT(DISTINCT  DBMS_ROWID.ROWID_BLOCK_NUMBER(rowid) ||
  2                        DBMS_ROWID.ROWID_RELATIVE_FNO(rowid)) "실사용 블록수"
  3    FROM  scott.test01 ;

실사용 블록수
--------------
             0
```

DELETE 가 발생해서 모든 데이터가 삭제 되었지만
테이블의 용량은 전혀 변하지 않음이 확인되었습니다.

6. DML을 배웁니다

- DELETE 후 용량까지 줄이는 Reorg 작업을 수행합니다.

1. 위에서 생성했던 SCOTT.TEST01 테이블에 데이터를 추가합니다.

```
SYS>BEGIN
  2  FOR i  IN 1..1000 LOOP
  3    INSERT INTO scott.test01
  4    VALUES (i , DBMS_RANDOM.STRING('A',19)
  5                , DBMS_RANDOM.STRING('B',19) );
  6    END LOOP;
  7    COMMIT;
  8  END;
  9  /

PL/SQL procedure successfully completed.
```

6. DML을 배웁니다

2. 데이터건수와 테이블 용량을 측정합니다.

```
SYS>SELECT COUNT(*) FROM SCOTT.TEST01;

  COUNT(*)
------------
      1000    <- 1000 건이 입력되었습니다.

SYS>SELECT SUM(BYTES)/1024/1024 MB
  2  FROM DBA_SEGMENTS
  3  WHERE OWNER='SCOTT'
  4  AND SEGMENT_NAME='TEST01';

        MB
------------
        28    <- 용량은 여전히 28MB 입니다.
```

6. DML을 배웁니다

3. 1000 건의 데이터 중 300 건만 삭제합니다.

```
SYS>DELETE FROM SCOTT.TEST01
  2  WHERE no BETWEEN 1 AND 300 ;

300 rows deleted.

SYS>COMMIT;

Commit complete.

SYS>SELECT COUNT(*) FROM SCOTT.TEST01;

  COUNT(*)
------------
       700
```

6. DML을 배웁니다

```
SYS>SELECT SUM(BYTES)/1024/1024 MB
  2  FROM DBA_SEGMENTS
  3  WHERE OWNER='SCOTT'
  4  AND SEGMENT_NAME='TEST01';

        MB
------------
        28    <- 여전히 예상대로 28 MB 입니다.
```

6. DML을 배웁니다

4. 테이블 REORG(리오그) 작업을 합니다.

```
SYS>SELECT TABLE_NAME,TABLESPACE_NAME
  2  FROM DBA_TABLES
  3  WHERE TABLE_NAME='TEST01';

TABLE_NAME     TABLESPACE_NAME
-------------- ------------------------
TEST01                         USERS   <- 현재 USERS 테이블스페이스입니다.
```

```
SYS>ALTER TABLE SCOTT.TEST01 MOVE TABLESPACE USERS;

Table altered.
```

6. DML을 배웁니다

```
SYS>SELECT SUM(BYTES)/1024/1024 MB
  2  FROM DBA_SEGMENTS
  3  WHERE OWNER='SCOTT'
  4  AND SEGMENT_NAME='TEST01';

        MB
------------
     .0625   <-  용량이 현저하게 줄어 든 것이 확인됩니다.

SYS>SELECT COUNT(*) FROM SCOTT.TEST01 ;

  COUNT(*)
--------------
       700   <- 데이터는 700 건 있습니다.
```

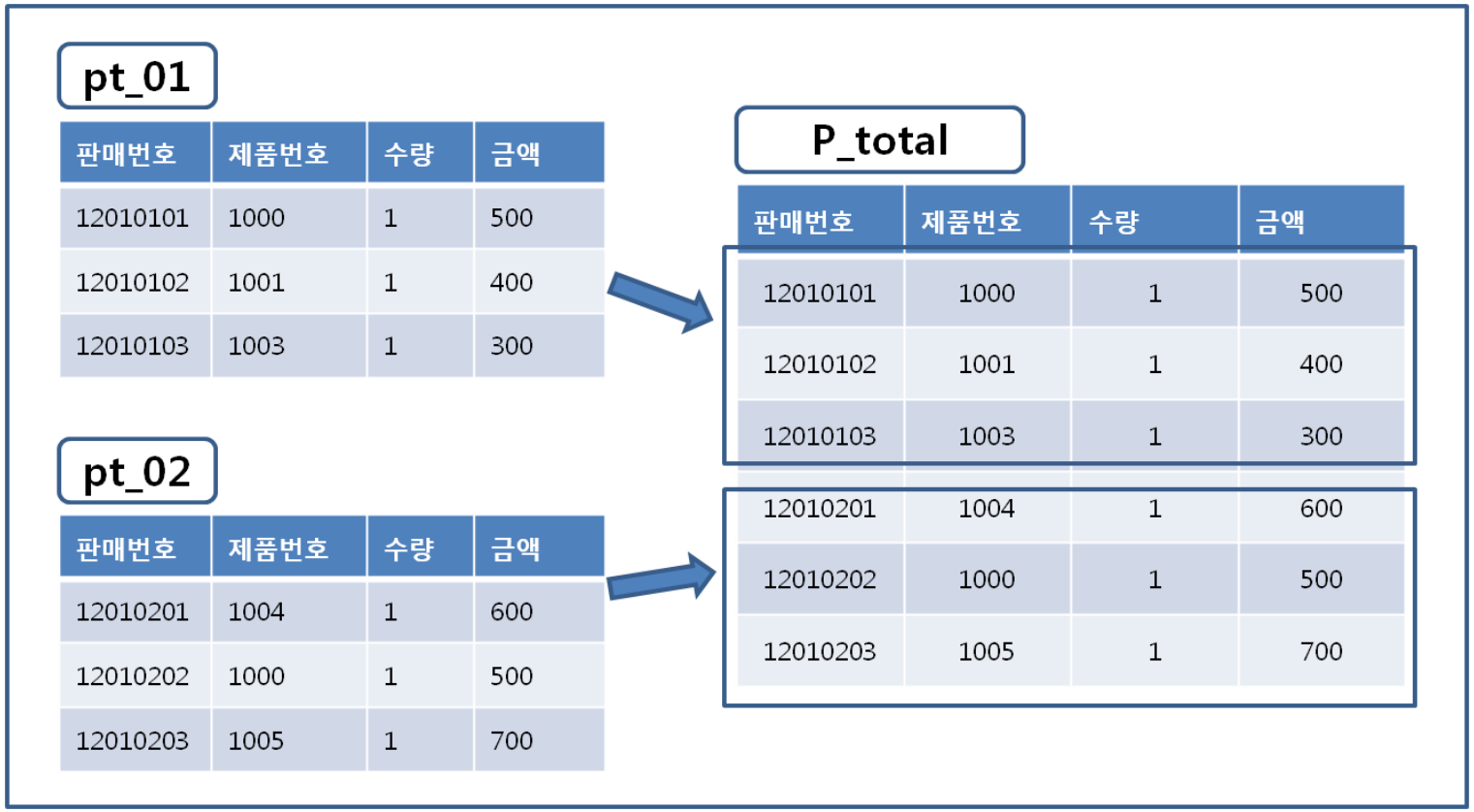
6. DML을 배웁니다

4. MERGE (테이블 합치기)

```
SQL>  MERGE  INTO  Table1
 2  USING  Table2
 3  ON  ( 병합 조건절 )
 4  WHEN  MATCHED  THEN
5    UPDATE  SET  업데이트 내용
6    DELETE  WHERE  조건
 7  WHEN  NOT  MATCHED  THEN
 8   INSERT  VALUES(컬럼 이름) ;
```

6. DML을 배웁니다

- Merge 실습



6. DML을 배웁니다

- Merge 전 테이블 내용 확인

```
SCOTT>SELECT  *  FROM  pt_01 ;

판매번호   제품번호    수량     금액
---------- ---------- ---------- ----------
12010101   1000         1        500
12010102   1001         1        400
12010103   1003         1        300
```

```
SCOTT>SELECT  *  FROM  pt_02 ;

판매번호   제품번호    수량     금액
---------- ---------- ---------- ----------
12010201   1004         1        600
12010202   1000         1        500
12010203   1005         1        700
```

```
SCOTT>SELECT  *  FROM  p_total ;

no rows selected
```

6. DML을 배웁니다

- **MERGE 작업 QUERY 1 (pt_01 테이블과 p_total 테이블 병합)**

```
SCOTT>MERGE  INTO  p_total  total
  2  USING  pt_01  p01
  3  ON  (total.판매번호=p01.판매번호)
  4  WHEN  MATCHED  THEN
  5    UPDATE  SET  total.제품번호 = p01.제품번호
  6  WHEN  NOT  MATCHED  THEN
  7    INSERT  VALUES(p01.판매번호 , p01.제품번호 , p01.수량 , p01.금액) ;
```

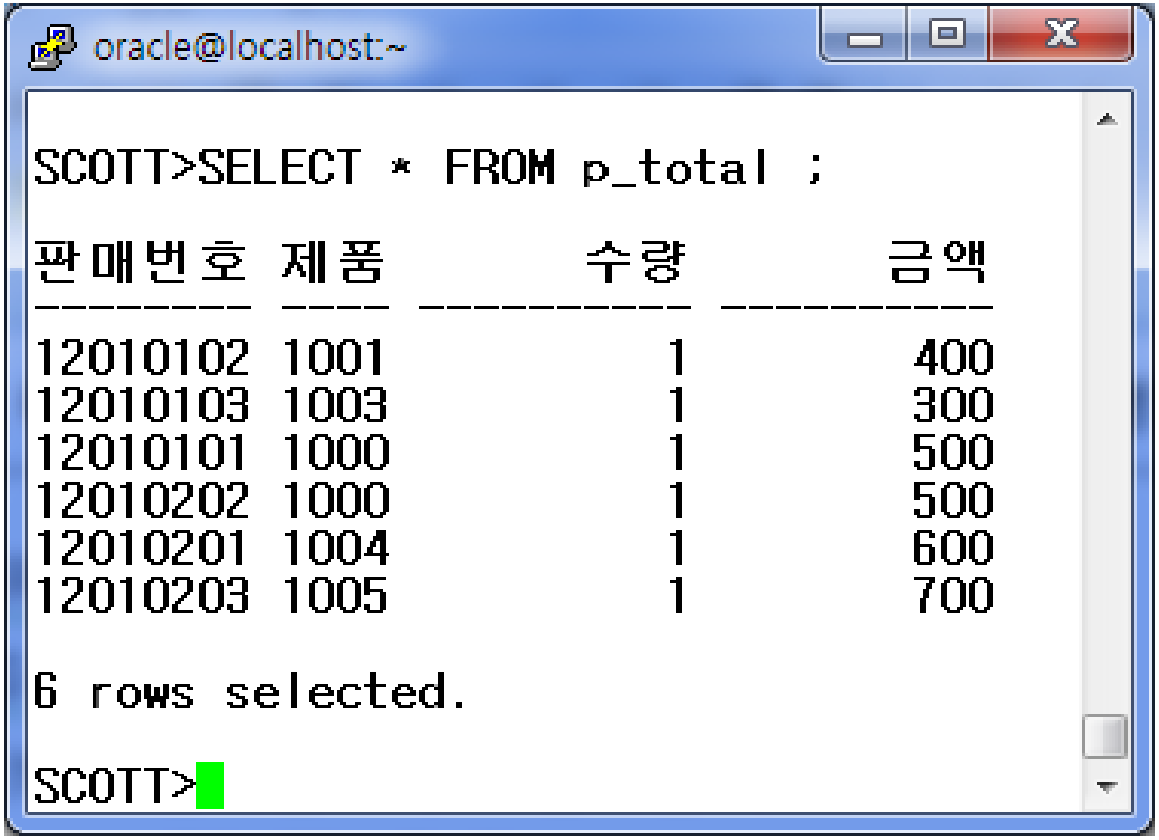
6. DML을 배웁니다

- MERGE 작업 QUERY 2 (pt_02 테이블과 p_total 테이블 병합)

```
SCOTT>MERGE INTO p_total total
  2 USING pt_02 p02
  3 ON (total.판매번호=p02.판매번호)
  4 WHEN MATCHED THEN
  5   UPDATE SET total.제품번호 = p02.제품번호
  6 WHEN NOT MATCHED THEN
  7   INSERT VALUES(p02.판매번호 , p02.제품번호 , p02.수량 , p02.금액) ;
```

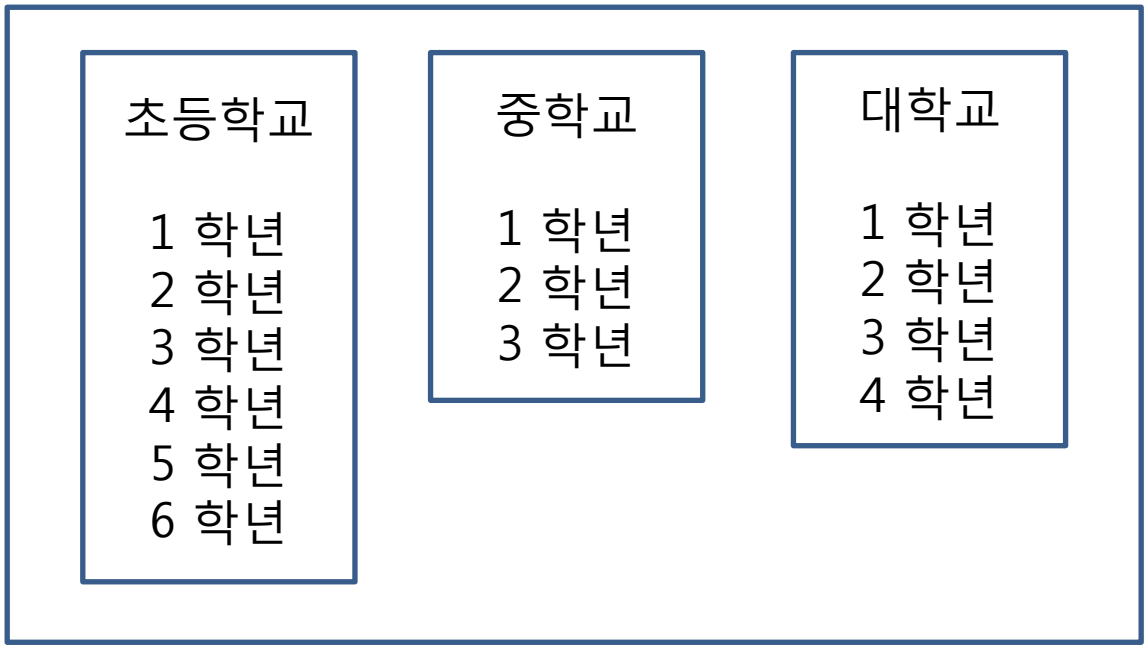
6. DML을 배웁니다

- Merge 작업 완료 후 결과 조회하기



6. DML을 배웁니다

5. TRANSACTION 관리하기



- Commit – 트랜잭션 확정하기

- Rollback – 트랜잭션 취소하기

6. DML을 배웁니다

6. DML ERROR LOGGING 하기 (10g R2 부터 지원됨)

1. DBMS_ERRLOG 패키지를 수행해서 에러 로깅 테이블 DML_ERRORS 생성합니다.

```
SCOTT>SELECT * FROM dml_err_test ;

        NO   NAME
----------  ----------
         1   AAA
         2   BBB
```

이 테이블은 NO 컬럼에 Primary Key가 설정이 되어 있습니다.

6. DML을 배웁니다

- 에러 내용을 로깅하기 위해 DBMS_ERROR 패키지를 수행합니다.

```
SCOTT>BEGIN
  2  DBMS_ERRLOG.CREATE_ERROR_LOG (
  3    dml_table_name => 'DML_ERR_TEST',
  4    err_log_table_name => 'DML_ERRORS' );
  5  END ;
  6  /

PL/SQL procedure successfully completed.
```

6. DML을 배웁니다

- 에러가 기록되는 DML_ERRORS 테이블을 살펴봅니다.

```
SCOTT>DESC dml_errors ;

 Name                          Null?    Type
 ----------------------------- -------- ----------------------------
 ORA_ERR_NUMBER$                        NUMBER
 ORA_ERR_MESG$                          VARCHAR2(2000)
 ORA_ERR_ROWID$                         ROWID
 ORA_ERR_OPTYP$                         VARCHAR2(2)
 ORA_ERR_TAG$                           VARCHAR2(2000)
 NO                                      VARCHAR2(4000)
 NAME                                    VARCHAR2(4000)
```

6. DML을 배웁니다

2. DML_ERR_TEST 테이블에 에러를 발생하는 DML 을 수행합니다.

```
SCOTT>INSERT INTO dml_err_test
  2  VALUES (1,'CCC')
  3     LOG ERRORS INTO dml_errors('INSERT..RL=UNLIMITED')
  4  REJECT LIMIT UNLIMITED ;

0 rows created.
```

6. DML을 배웁니다

3. 에러를 확인합니다 (관련 스크립트는 교재를 참조하세요)

```
SCOTT>exec print_table('SELECT * FROM dml_errors');
ORA_ERR_NUMBER$                 : 1
ORA_ERR_MESG$                   : ORA-00001: unique constraint
(SCOTT.SYS_C0014256) violated
ORA_ERR_ROWID$                  :
ORA_ERR_OPTYP$                  : I
ORA_ERR_TAG$                    : INSERT..RL=UNLIMITED
NO                              : 1
NAME                            : CCC
-----------------
PL/SQL procedure successfully completed.
```