DML(Data Manipulation Language)
[정의] 데이터 조작어
[종류] INSERT
UPDATE
DELETE
MERGE(9i)

INSERT

[설명] 1번에 1개의 Row 입력, 2가지 유형중, 좋은 방식은 ?,

[주의] 2번째 방식은 컬럼명과 값을 1:1로 매핑-> 테이블에 정의된 컬럼 순서 필요(X)

① INSERT INTO DEPT VALUES(50,'연구소1','서울'); // 컬럼명 생략시?

② INSERT INTO DEPT(DEPTNO,DNAME,LOC) VALUES(51,'연구소2','대전'); // 좋은방식의 SQL문은?

③ SELECT * FROM DEPT;

// 해당 컬럼을 생략하는 경우 NULL로 지정된다.

④ INSERT INTO DEPT VALUES('중부영업점','대구'); // ERROR의 이유는?

⑤INSERT INTO DEPT(DNAME,LOC) VALUES('중부영업점','대구'); // 여전히 ERROR인 이유는?

// ORA-01400: NULL을 ("SCOTT"."DEPT"."DEPTNO") 안에 삽입할 수 없습니다.

// CONSTRAINT 과정에서 설명

// INSERT시에 특정 COLUM에 NULL값 삽입방법

-- EXPLICIT방법 2가지

⑥ INSERT INTO DEPT(DEPTNO,DNAME,LOC) VALUES(52, '북부영업점',NULL); // 'NULL' 과 다른점은?

⑦ INSERT INTO DEPT(DEPTNO,DNAME,LOC) VALUES(53, '남부영업점','');

-- IMPLICIT방법 1가지

⑧INSERT INTO DEPT(DEPTNO,DNAME) VALUES(54,'서부영업점'); // 대상 컬럼 생략

⑨ SELECT DEPTNO,DNAME,NVL(LOC,'미지정지역') AS LOC FROM DEPT; // 결과 조회

⑩ COMMIT; //변경사항을 DB에 영구히 반영

//TRANSACTION 과정에서 설명

UPDATE

UPDATE 테이블명 SET [수정할 컬럼명=수정할값] WHERE~

// 연구소 조직 개편
// 50번 조직 연구소1 → 중부연구소로 변경
// 51번 조직 연구소2 → 북서부연구소, 대전→ 인천 변경.

⑪ UPDATE DEPT SET DNAME = '중부연구소' WHERE DEPTNO = 50; // 단일 COLUMN수정
⑫ UPDATE DEPT SET DNAME = '북서부연구소', LOC='인천' WHERE DEPTNO = 51; // 복수 COLUMN수정

① SELECT * FROM DEPT WHERE DEPTNO IN (50,51); // 결과 조회
② COMMIT;

③ UPDATE DEPT SET LOC='미개척지역'; // 주의사항: WHERE절 생략시 전체 ROW가 처리됩니다.
④ SELECT * FROM DEPT;
⑤ ROLLBACK; // 잘못된 데이터 처리시 해당 변경사항을 취소
⑥ SELECT * FROM DEPT;

DELETE

// 미개척 지역을 폐쇄

① DELETE FROM DEPT WHERE LOC IS NULL; // NULL 값인 데이터를 찾는 방법

② DELETE DEPT; // FROM 생략 가능합니다.
// 주의사항: WHERE절 생략시 전체 ROW가 처리됩니다.

③ SELECT * FROM DEPT;
④ ROLLBACK;

MERGE(9i 의 new feature)

[요구]MERGE SQL 작성

DML SUBQUERY

// SUBQUERY로 한번에 여러ROW를 INSERT 합니다.
⑤ INSERT INTO BONUS(ENAME,JOB,SAL,COMM)
SELECT ENAME,JOB,SAL,COMM FROM EMP;
⑥ SELECT * FROM BONUS;
⑦ ROLLBACK; // 다음번 실습을 위해서
⑧ SELECT * FROM BONUS;

// 부서별로 보너스를 계산한후(데이터 연산) INSERT 작업
⑨ INSERT INTO BONUS(ENAME,JOB,SAL,COMM)
SELECT ENAME,JOB,SAL,DECODE(DEPTNO,10,SAL*0.3,20,SAL*0.2)+NVL(COMM,0)
FROM EMP
WHERE DEPTNO IN (10,20);
⑩ SELECT * FROM BONUS;
⑪ COMMIT;
⑫ ROLLBACK;

// 평상시에 COMM을 받지 못하는 사원들에게 평균 COMM 금액의 50%를 보너스로 지급

⑬ UPDATE EMP SET COMM = (SELECT AVG(COMM)/2 FROM EMP)
WHERE COMM IS NULL OR COMM = 0;

⑭ COMMIT;

// 평균 이상의 급여를 받는 사원들은 보너스 지급 대상자에서 제외

⑮ DELETE FROM BONUS WHERE SAL > (SELECT AVG(SAL) FROM EMP ;
COMMIT;

TRANSACTION

[정의] 데이터 베이스의 논리적 연산 단위(?), 밀접히 관련되어서 분리될수 없는 한 개 이상의 데이터 조작
하나의 논리적인 작업단위를 구성하는 세부적인 연산들의 집합

[특징] ACID -> 일원고지

[시작]

[종료]

COMMIT ~ 트랜잭션에서 변경된 사항을 데이터베이스에 영구히 반영하는 것

ROLLBACK ~ 트랜잭션 시작 이전의 상태로 되돌리는 것 입력/수정/삭제를 취소하고 lock을 해제

SAVEPOINT ~ 저장점, 현재 시점부터 SAVEPOINT까지 트랜잭션의 일부만 ROLLBACK

*복잡한 대규모 TRANSACTION에서 에러가 발생했을 때

// TRANSACTION 시작 과 종료

① ROLLBACK; // TRANSACTION END

② INSERT INTO DEPT(DEPTNO,DNAME,LOC) VALUES(90,'신규사업부','경기도'); // TRANSACTION START

③ UPDATE EMP SET DEPTNO = 90 WHERE DEPTNO = 30; // 진행중

④ DELETE FROM DEPT WHERE DEPTNO = 30; // 진행중

⑤ SELECT * FROM DEPT; // 진행중인 상태에서 조회

⑥ SELECT * FROM EMP WHERE DEPTNO = 30;

⑦ ROLLBACK; // TRANSACTION END

⑧SELECT * FROM DEPT; // 종료후 조회

⑨SELECT * FROM EMP WHERE DEPTNO = 30;

// TRANSACTION 종료후 ROLLBACK 처리범위

⑧ INSERT INTO EMP(EMPNO,ENAME,JOB,SAL) VALUES(1111,'오라클','DBA',3500); // TRANSACTION START

⑨ UPDATE EMP SET SAL = SAL* 1.3;

⑩ COMMIT; // TRANSACTION END

⑪ ROLLBACK WORK;

⑫ SELECT * FROM EMP;

// TRANSACTION 과 DDL

① INSERT INTO EMP(EMPNO,ENAME,DEPTNO) VALUES(9999,'OCPOK',20); //TRANSACTION START

② ALTER TABLE EMP ADD(SEX CHAR(1) DEFAULT 'M'); // DDL

③ ROLLBACK; // 취소 범위는?

④ DESC EMP;

⑤ ALTER TABLE EMP DROP COLUMN SEX; // DDL

⑥ ROLLBACK; // 취소 범위는?

⑦ DESC EMP

[요구]

ⓐ SAVEPOINT 예제 만들어 부분 롤백이 가능한지 증명 하십시요

ⓑ AUTOCOMMIT 예를 재현 하십시요

1) DDL 수행시 , 데이터베이스 정상적으로 접속 종료시

2) 비정상 접속 종료시, DBMS 비정상 종료시

ⓒ SELECT ~ FOR UPDATE 의 기능 및 트랜잭션 시작/종료를 설명 하십시요.

Rollback Level

// STATEMENT Level ROLLBACK , SQL*PLUS 에서 실습

ED C:₩03_SQL₩TST_TRANS.SQL

```
① ROLLBACK;
② SELECT /*  Before Transaction */ EMPNO,SAL FROM EMP WHERE EMPNO IN (7788,7902);
③ DELETE FROM  EMP WHERE DEPTNO = 10;
④ UPDATE  /* STATEMENT LEVEL ROLLBACK */  EMP SET SAL = 123456789 WHERE EMPNO = 7788;
⑤ UPDATE  EMP SET  SAL = 1234        WHERE  EMPNO = 7902;
⑥ COMMIT;
⑦ -- 데이터 확인
   SELECT /* After Transaction */ EMPNO,SAL FROM EMP WHERE EMPNO IN (7788,7902);
   SELECT /* After Transaction */ EMPNO,SAL FROM EMP WHERE DEPTNO = 10;
```

[파일 저장후 EDITOR 종료]

@ TEST.SQL // 왜 실행이 안되는지 ?

// TRANSACTION LEVEL ROLLBACK

ED C:₩03_SQL₩TST_TRAN_P.SQL

```
SELECT /* Before Transaction */ EMPNO,SAL FROM EMP WHERE EMPNO IN (7499,7698);
SELECT /* Before Transaction */ EMPNO,SAL FROM EMP WHERE DEPTNO = 20;

BEGIN
        /* 1. 멀티행 라인
           2. 주석 테스트   */
    ① DELETE FROM EMP WHERE DEPTNO = 20;
    ② -- 자리수 초과 에러 발생
       UPDATE  EMP SET  SAL = 123456789 WHERE  EMPNO = 7499;
    ③ UPDATE  EMP SET  SAL = 1234       WHERE  EMPNO = 7698;
    ④ COMMIT;
EXCEPTION
    WHEN  OTHERS  THEN
       ⑤ ROLLBACK;                                  --  TRANSACTION LEVEL ROLLBACK
END;
/

SELECT /* After Transaction */ EMPNO,SAL FROM EMP WHERE DEPTNO = 20;
SELECT /* After Transaction */ EMPNO,SAL FROM EMP WHERE EMPNO IN (7499,7698);
```

[파일 저장후 EDITOR 종료]

@ C:₩03_SQL₩TST_TRAN_P.SQL

// 다른 SQL 실습을 위해서 SAMPLE DATA 재생성

// 학과목 자료 폴더에서 demobld.sql 파일을 C:₩03_SQL 폴더로 다운 받기

SQL>@C:₩03_SQL ₩demobld.sql

TRANSACTION과 읽기일관성(READ CONSISTENCY

// SQL*PLUS 에서 2개의 세션을 생성후 실습

Connect scott/tiger	Connect scott/tiger
① update emp set sal=0 where deptno= 10;	
	② select deptno,ename,sal from emp where deptno = 10; //??
③ select deptno,ename,sal from emp where deptno = 10; //??	
	⑤ ②재실행
④ commit;	
⑥ ③재실행	

TRANSACTION 과 Row Level Lock

Connect scott/tiger	Connect scott/tiger
① update emp set sal=9999 where deptno= 10;	
	② delete from emp where deptno = 20; ③ delete from emp where deptno = 10; //??
④ commit; // or rollback;	
	⑤ rollback; //다음번 test를 위해서…

SELECT * FOR UPDATE (ROW LEVEL LOCK),

// Repeatable Read 보장

Connect scott/tiger	Connect scott/tiger
① SELECT * FROM EMP WHERE DEPTNO = 10 FOR UPDATE [WAIT]; ;	
	② delete from emp where deptno = 20; ③ delete from emp where deptno = 10; //??
④ commit; // or rollback;	
	⑤ rollback; //다음번 test를 위해서…

Connect scott/tiger
 	
② SELECT * FROM EMP
WHERE DEPTNO = 10 FOR UPDATE ;
 	
// 개인의 취향에 따라 기다려봄
 	
④ commit; // or rollback;
	Connect scott/tiger
	
	① delete from emp where deptno = 10
	
	③ rollback;

Connect scott/tiger ② SELECT * FROM EMP WHERE DEPTNO = 10 FOR UPDATE WAIT 10 // 10초후기다리면….??? 	Connect scott/tiger ① delete from emp where deptno = 10 ③ rollback;

[요구] 아래의 PL/SQL 구문을 SQL SCRIPT 파일로 만들어서 실습 해보시길

```
SET SERVEROUTPUT ON
DECLARE
        V_SAL  EMP.ESAL%TYPE;
BEGIN
        SELECT SAL INTO V_SAL
        FROM    EMP
        WHERE   EMPNO = 7369
        FOR UPDATE WAIT 4;
EXCEPTION
        WHEN OTHERS THEN
                DBMS_OUTPUT.PUT_LINE('EXCEPTION:'||SQLERRM);
END;
/
```