03. DML

목차

1. SQL 명령어 분류 와 DML

분 류	종 류
QUERY	SELECT(데이터 조회)
DML(Data Manipulation Language)	INSERT(데이터 입력), UPDATE(데이터 수정), DELETE(데이터 삭제)
TCL(Transaction Control Language)	COMMIT(트랜잭션 저장),ROLLBACK(트랜잭션 취소), SAVEPOINT(트랜잭션 임시 저장점)
DDL(Data Definition Language)	CREATE(데이터베이스 Object 생성), ALTER(데이타베이스 Object 변경), DROP(데이타베이스 Object 삭제)
DCL(Data Control Language)	GRANT(권한 부여),REVOKE(권한 취소)

Google →

oracle insert syntax 12c

→ INSERT(클릭)

→ SYNTAX(클릭)

2. 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 ??
④ INSERT INTO DEPT VALUES('중부영업점','대구');                          // ERROR의 이유는?
⑤ INSERT 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
⑧ INSERT 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
```

3. UPDATE

UPDATE 테이블명 SET [수정할 컬럼명=수정할값] WHERE 조건절

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

① UPDATE DEPT SET DNAME = '중부연구소' WHERE DEPTNO = 50; // 단일 COLUMN수정
② UPDATE DEPT SET DNAME = '북서부연구소', LOC='인천' WHERE DEPTNO = 51; // 복수 COLUMN수정

③ SELECT * FROM DEPT WHERE DEPTNO IN (50,51); // 결과 조회
④ COMMIT;

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

4. 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 처리 된다

③ SELECT * FROM DEPT;
④ ROLLBACK;

5. 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] 첫번째 실행가능(Executable)한 SQL 실행시
[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
⑧ SELECT * FROM DEPT;                                                        // 종료후 조회
⑨ SELECT * FROM EMP WHERE DEPTNO = 30;
```

5. TRANSACTION

❏ 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
```

5. TRANSACTION

❑ Rollback Level

1) SQL Script 파일생성하여 실습 (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;
```

@ TST_TRANS.SQL // **트랜잭션 원자성** ??

5. TRANSACTION

❑ Rollback Level

2) SQL Script 파일생성하여 실습 (TST_TRANS_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);
```

@ TST_TRANS_P.SQL // **트랜잭션 원자성 ??**

5. TRANSACTION 과 읽기 일관성(READ CONSISTENCY)

// 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; //??	
④ commit;	
	⑤ ②번 SQL 재실행
⑥ ③번 SQL재실행	

5. TRANSACTION 과 Row Level Lock

// 2개의 세션을 생성후 실습

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…

5. TRANSACTION 과 Repeatable Read

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

5. TRANSACTION 과 Repeatable Read

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

5. TRANSACTION

과제

ⓐ SAVEPOINT 예제 만들어 부분 롤백이 가능한지 증명

ⓑ AUTOCOMMIT 사례 재현

DDL 수행시 , 데이터베이스 정상적으로 접속 종료시

비정상 접속 종료시, DBMS 비정상 종료시

ⓒ SELECT ~ FOR UPDATE 의 기능 및 트랜잭션 시작/종료 설명