10. 트랜잭션 관리

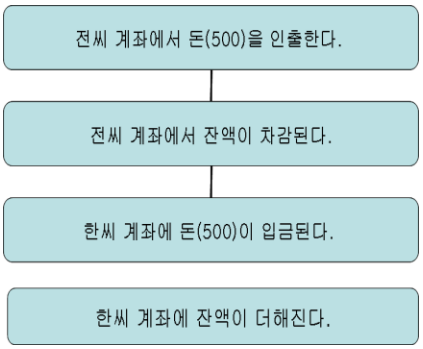
데이터를 추가, 수정, 삭제하는데 필요한 트랜잭션을 학습한다.
트랜잭션 작업을 위한 COMMIT, ROLLBACK, SAVEPOINT 문을 학습한다.

1) 트랜젝션

오라클은 트랜잭션을 기반으로 데이터의 일관성을 보장합니다. 트랜잭션(Transaction)은 데이터 처리에서 논리적으로 하나의 작업 단위를 의미합니다.

트랜젝션이란 개념을 도입한 이유는 데이터의 일관성을 유지하고 안정적으로 데이터를 복구시키기 위해서이다.

- 트랜젝션의 예



Transaction란 insert, update, delete 명령은 메모리상에서만 변경되다가 특정 단위로 하드디스크의 실제 파일인 database에 저장되는 단위이다.

이러한 과정에서 시스템의 문제로 전씨 계좌에서는 돈이 인출되었는데 한씨 계좌에 입금 처리 도중에 문제가 생겨서 정상적으로 입금되지 않았다면 전씨 계좌의 인출 작업도 취소되어야 한다. 처리 중간에 무슨 문제가 발생한다면 진행되던 인출과정 전체를 취소하고 다시 처음부터 시작해야 하기 때문인데 이것이 트랜젝션이다.

2) COMMIT과 ROLLBACK

- COMMIT

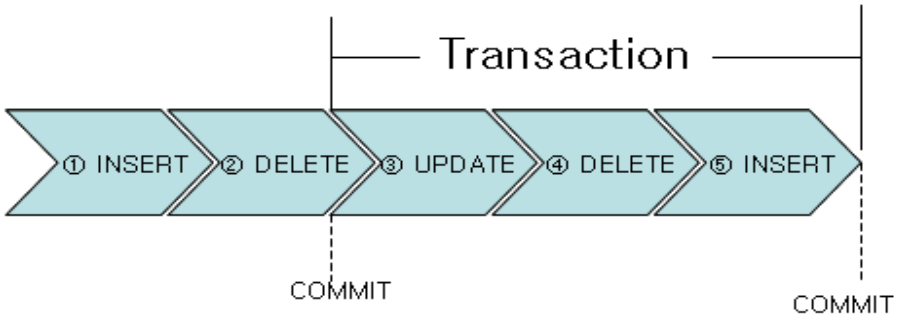
모든 작업들을 정상적으로 처리하겠다고 확정하는 명령어로 트랜잭션의 처리과정을 데이터베이스에 모두 반영하기 위해서 변경된 내용을 모두 영구 저장한다. COMMIT 명령어를 수행하게 되면 하나의 트랜잭션 과정이 종료되는 것이다.

- ROLLBACK

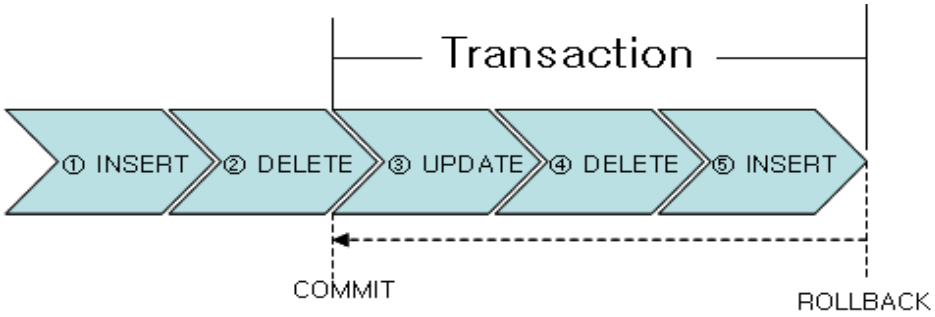
작업 중 문제가 발생되어서 트랜잭션의 처리 과정에서 발생한 변경 사항을 취소하는 명령어이다.
ROLLBACK 명령어 역시 트랜잭션 과정을 종료하게 된다. ROLLBACK은 트랜잭션으로 인한 하나의 묶음 처리가 시작되기 이전의 상태로 되돌린다.

여러 개의 DML 명령어들은 하나의 논리적인 작업 단위(트랜잭션)로 구성된다.
어떻게 여러 개의 DML 명령어들은 하나의 논리적인 작업 단위인 트랜잭션으로 묶을 수 있을까?
마지막으로 실행한 커밋(혹은 롤백) 명령 이후부터 새로운 커밋(혹은 롤백) 명령을 실행하는 시점까지 수행된 모든 DML 명령들을 의미한다.

아래 그림에서 UPDATE 문으로 데이터를 갱신하고(③), DELETE 문으로 데이터를 삭제하고(④), INSERT 문을 사용해 데이터를 삽입(⑤)한다. 만약 이 모든 과정이 오류 없이 수행되었다면 지금까지 실행한 모든 작업(③, ④, ⑤)을 "데이터베이스에 영구 저장하라"는 명령으로 COMMIT을 수행한다.



롤백 명령은 마지막으로 수행한 커밋 명령까지만 정상 처리(①, ②)된 상태로 유지하고 그 이후에 수행했던 모든 DML 명령어 작업(③, ④, ⑤)들을 취소시켜 이전 상태로 원상 복귀시킨다.



트랜잭션은 이렇듯 All-OR-Nothing 방식으로 DML 명령어들을 처리한다.

• COMMIT 명령어과 ROLLBACK 명령어의 장점

- 데이터 무결성이 보장된다.
- 영구적인 변경 전에 데이터의 변경 사항을 확인할 수 있다.
- 논리적으로 연관된 작업을 그룹화할 수 있다.

• COMMIT 명령어

- Transaction(INSERT, UPDATE, DELETE) 작업 내용을 실제 DB에 저장한다.
- 이전 데이터가 완전히 UPDATE 된다.
- 모든 사용자가 변경된 데이터의 결과를 볼 수 있다.

• ROLLBACK 명령어

- Transaction(INSERT, UPDATE, DELETE) 작업 내용을 취소한다.
- 이전 COMMIT한 곳 까지만 복구한다.

• 자동 COMMIT 명령과 자동 ROLLBACK 명령이 되는 경우

SQL* PLUS가 정상 종료되었다면 자동으로 COMMIT되지만, 비정상 종료되었다면 자동으로 ROLLBACK 한다.

DDL(데이터 정의어)**과 DCL**(데이터 제어어) **명령문이 수행된 경우 자동으로 COMMIT 된다.**

정전이 발생했거나 컴퓨터 Down시(컴퓨터의 전원이 끊긴) 자동으로 ROLLBACK 된다.

```
DROP TABLE DEPT02;

CREATE TABLE DEPT02                    --서브쿼리로 테이블 복사
AS SELECT *
FROM DEPARTMENTS;
```

```
SELECT *
FROM DEPT02;                    --확인
```

```
DELETE FROM DEPT02;                    --테이블 전체 삭제
```

```
SELECT *
FROM DEPT02;                    --확인
```

```
ROLLBACK;                              --삭제 이전으로 되돌림
```

```
SELECT *
FROM DEPT02;                    --확인
```

부서번호 20번 사원에 대한 정보만 삭제한 후 확인한다.

```
DELETE FROM DEPT02
WHERE DEPARTMENT_ID=20;
```

```
SELECT *
FROM DEPT02;
```

```
COMMIT;
```

```
ROLLBACK;
```

```
SELECT *
FROM DEPT02;
```

COMMIT 후에는 ROLLBACK 해도 소용없다.

- 자동 커밋
  DDL문에는 CREATE, ALTER, DROP, RENAME, PRUNCATE 등이 있다.
  이러한 DDL문은 자동으로 커밋(AUTO COMMIT)이 발생한다.

3) SAVEPOINT

SAVEPOINT 명령을 써서 현재의 트랜잭션을 작게 분할할 수 있다.

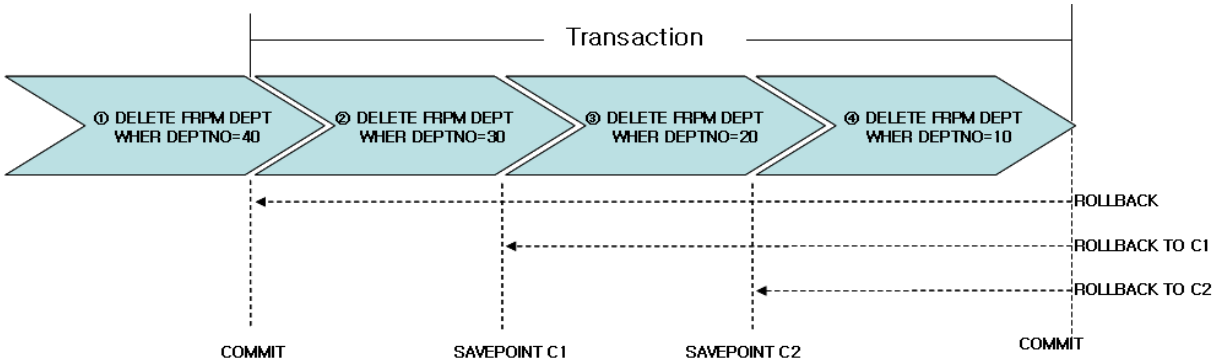
저장된 SAVEPOINT는 ROLLBACK TO SAVEPOINT 문을 사용하여 표시한 곳까지 ROLLBACK할 수 있다.

• 특정 위치를 지정하기 위한 사용 형식

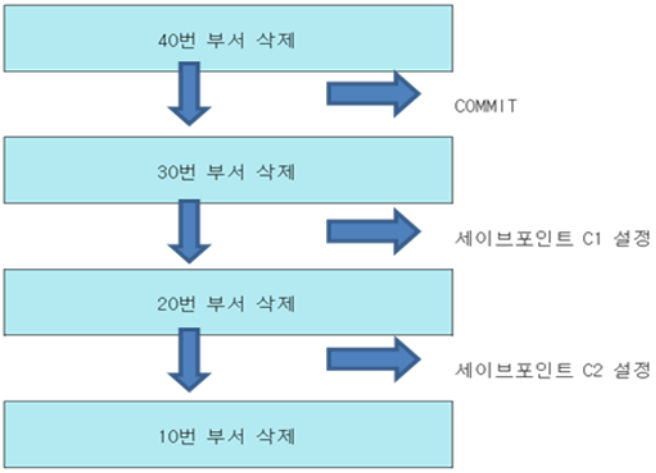
```
SAVEPOINT label_name;
```

• 지정해 놓은 특정위치로 되돌아가기 위한 형식

```
ROLLBACK TO label_name;
```



위 그림을 보면 COMMIT명령이 내려진 후 다음 COMMIT명령이 나타날 때까지가 하나의 트랜젝션으로 구성되므로 ②번에서 ④번까지가 하나의 트랜잭션이 된다. 이렇게 트랙잭션을 구성할 때 중간 중간 SAVEPOINT 명령으로 위치를 지정해 놓으면(예를 들어 C1) 하나의 트랜잭션 내에서도 ROLLBACK TO C1(SAVEPOINT 문을 사용하여 표시한 곳)까지 ROLLBACK할 수 있다.



위의 순서대로 진행할 경우의 명령어

```
DROP TABLE DEPT02;

CREATE TABLE DEPT02
AS SELECT *
FROM DEPARTMENTS;
```

```
DELETE FROM DEPT02
```

```
WHERE DEPTNO=40;
COMMIT;
```

```
DELETE FROM DEPT02
WHERE DEPTNO=30;

SAVEPOINT C1;
```

```
DELETE FROM DEPT02
WHERE DEPTNO =20;

SAVEPOINT C2;
```

```
DELETE FROM DEPT02
WHERE DEPTNO=10;
```

```
SELECT * FROM DEPT02;
```

세이브 포인트 C2로 롤백

```
ROLLBACK TO C2;
SELECT *
FROM DEPT02;
```

세이브 포인트 C1로 롤백

```
ROLLBACK TO C1;
SELECT *
FROM DEPT02;
```

커밋 이전까지 롤백

```
ROLLBACK;
SELECT *
FROM DEPT02;
```