

# **Introduction to Oracle : SQL**

---

## Chapter7. 트랜잭션 제어 및 사용자 관리

- ▶ 트랜잭션의 의미를 이해하고 트랜잭션을 제어하는 명령어를 익힌다.
  - ▶ 사용자를 생성하고 권한을 부여, 박탈한다.
-

## 1. 트랜잭션의 정의

- 트랜잭션은 다음 중 하나로 구성된다.
    - 여러 개의 DML문
    - 하나의 DDL문
    - 하나의 DCL문
  - 트랜잭션은 DML문장이 처음 실행될 때 시작된다
  - 트랜잭션은 다음 이벤트 중 하나로 종료된다.
    - COMMIT 또는 ROLLBACK을 수행
    - DDL문이나 DCL문을 수행
    - SQL\*Plus를 종료
    - 시스템 손상
-

## 2. 트랜잭션 제어어 (TCL)

### (1) COMMIT과 ROLLBACK

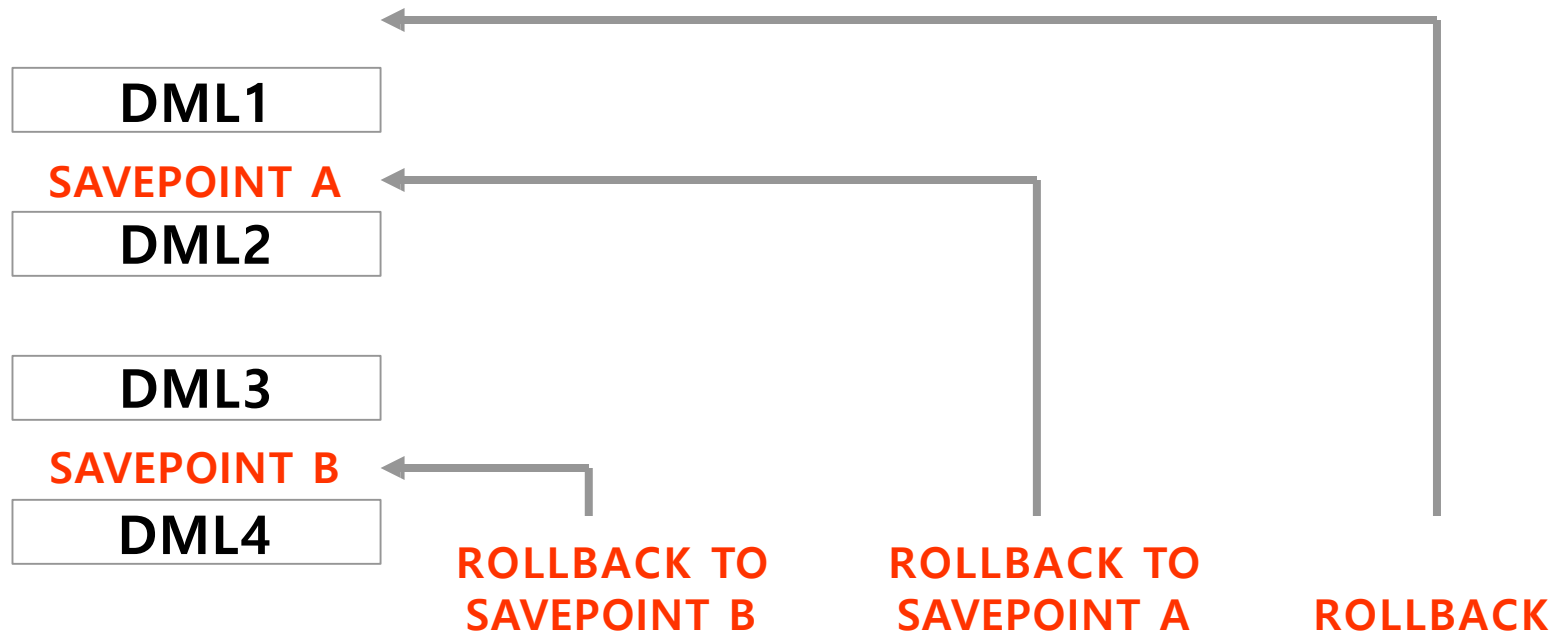
#### 1) COMMIT과 ROLLBACK의 장점

- 데이터 일관성 유지
  - 데이터를 영구적으로 변경하기 전 데이터 값의 확인 가능
  - 논리적으로 관련있는 명령끼리 그룹화 가능
-

## 2. 트랜잭션 제어어 (TCL)

### (1) COMMIT과 ROLLBACK

#### 2) COMMIT과 ROLLBACK의 기능



## 2. 트랜잭션 제어어 (TCL)

### (1) COMMIT과 ROLLBACK

#### 3) SAVEPOINT의 활용

- SAVEPOINT를 사용해서 현재 일으키고 있는 트랜잭션의 중간에 표시를 할 수 있다
  - 표시를 해둔 SAVEPOINT 지점으로 ROLLBACK TO SAVEPOINT 구문을 사용하여 트랜잭션의 진행 상황을 되돌릴 수 있다.
-

## 2. 트랜잭션 제어어 (TCL)

### (1) COMMIT과 ROLLBACK

#### 4) COMMIT 또는 ROLLBACK 전의 데이터 상태

- 데이터를 변경 이전 상태로 되돌릴 수 있다.
  - 현재 DML문을 수행하고 있는 사용자만 변경된 데이터 값 확인 가능
  - 다른 사용자들이 SELECT할 경우 변경되기 전의 값을 확인
  - 현재 DML 작업중인 행은 LOCK에 걸려 다른 유저가 DML 작업을 할 수 없게 된다.
-

## 2. 트랜잭션 제어어 (TCL)

### (1) COMMIT과 ROLLBACK

#### 5) COMMIT 후 데이터 상태

- 데이터의 변경 내용이 데이터베이스에 영구적 반영
  - 데이터의 변경 전 상태는 영구적으로 손실
  - 데이터베이스의 모든 사용자들이 변경 내용 확인 가능
  - 데이터 변경 작업동안 걸린 LOCK이 해제
  - 모든 SAVEPOINT는 지워짐
-



## 2. 트랜잭션 제어어 (TCL)

### (1) COMMIT과 ROLLBACK

#### 6) ROLLBACK 후 데이터 상태

- 데이터의 이전 상태로 복원
- 행에 걸린 LOCK이 해제

## 2. 트랜잭션 제어어 (TCL)

### (1) COMMIT과 ROLLBACK

#### 7) COMMIT 수행

```
DELETE FROM emp  
WHERE      employee_id = 10000;  
1행이 삭제되었습니다.
```

```
COMMIT ;  
커밋이 완료되었습니다.
```

---

## 2. 트랜잭션 제어어 (TCL)

### (1) COMMIT과 ROLLBACK

#### 8) ROLLBACK 수행

```
DELETE FROM emp ;  
107행이 삭제되었습니다.  
ROLLBACK ;  
롤백이 완료되었습니다.
```

```
UPDATE emp  
SET      department_id = 60  
WHERE    employee_id = 101  
1행이 변경되었습니다.  
SAVEPOINT A ;  
DELETE FROM emp ;  
ROLLBACK TO A ;  
롤백이 완료되었습니다.
```

## 2. 트랜잭션 제어어 (TCL)

### (2) 문장 단위의 ROLLBACK

- 단독 DML문이 실행 도중 실패하면 해당하는 문장 전으로만 ROLLBACK
  - 오라클 서버는 암시적인 SAVEPOINT를 실행시킨다.  
따라서 문제가 발생한 문장을 제외한 나머지 작업은 그래도 유지된다.
  - 문장단위의 ROLLBACK이 수행되고 난 후, 사용자는 반드시 COMMIT이나 ROLLBACK을 수행해서 명시적으로 트랜잭션을 종료해야 한다.
-

## 3. LOCK

### (1) 읽기 일관성

- 읽기 일관성은 데이터에 대해 항상 유지된다.
  - 같은 데이터에 대해 읽기 일관성을 보장한다.
    - SELECT하는 사람은 DML 수행을 기다리지 않는다.
    - DML을 수행하는 사람은 SELECT 수행을 기다리지 않는다.
-

## 3. LOCK

### (2) LOCK의 정의

- 동시에 일어나는 트랜잭션 사이의 해가 되는 상호작용을 보호
  - 사용자의 특별한 액션을 요구하지 않음
  - 트랜잭션이 종료되기 전까지 유지
  - LOCK이 걸린 행에 대해 다른 사용자들은 SELECT만 가능
-

## 4. 사용자 관리

### (1) 사용자 생성

#### 1) 사용자 생성 구문 및 사용의 예

- DBA는 CREATE USER문을 사용해서 사용자를 생성

```
CREATE USER 사용자명  
IDENTIFIED BY 패스워드 ;
```

```
CREATE USER user01  
IDENTIFIED BY pass ;
```

사용자가 생성되었습니다.

## 4. 사용자 관리

### (1) 사용자 생성

#### 2) 사용자 패스워드 변경

- DBA는 사용자에게 초기 비밀번호를 제공
- 해당 사용자는 본인의 패스워드를 변경 가능

```
ALTER USER user01  
IDENTIFIED BY pass01 ;
```

사용자가 변경되었습니다.



## 4. 사용자 관리

### (2) 권한

- 시스템 권한과 객체 권한으로 구분
  - 시스템 권한 : 데이터베이스에 대한 접근 권한
  - 객체 권한 : 데이터베이스 객체의 내용을 조작

## 4. 사용자 관리

### (2) 권한

#### 1) 시스템 권한

##### • 권한 부여 구문

```
GRANT 권한명 [, 권한명...]  
TO 사용자명 [, 사용자명 | 롤 | PUBLIC];
```

```
GRANT create session, create table, create view  
TO user01 ;
```

권한이 부여되었습니다.

## 4. 사용자 관리

### (2) 권한

#### 2) 객체 권한

- 권한 부여 구문

```
GRANT 객체권한명 [ (컬럼명) ]  
ON 객체명  
TO 사용자명 | 롤 | PUBLIC [WITH GRANT OPTION] ;
```

- 객체의 소유자는 해당 객체에 대한 모든 권한을 갖는다.
  - 객체의 소유자는 해당 객체에 대한 모든 권한을 다른 사용자에게 부여할 수 있다.
-

## 4. 사용자 관리

### (2) 권한

#### 2) 객체 권한

##### • 권한의 종류

객체 권한	Table	View	Sequence	Procedure
ALTER	○		○	
DELETE	○	○		
EXECUTE				○
INDEX	○			
INSERT	○	○		
REFERENCES	○	○		
SELECT	○	○	○	
UPDATE	○	○		

## 4. 사용자 관리

### (2) 권한

#### 2) 객체 권한

- 권한 부여 사용의 예

```
GRANT select  
ON      dept  
TO      user01, scott ;
```

```
GRANT update (location_id, city)  
ON      dept  
TO      user01, man_role ;
```

## 4. 사용자 관리

### (2) 권한

#### 2) 객체 권한

- WITH GRANT OPTION

- 권한에 대한 상속 권한도 함께 부여

```
GRANT  select, update
ON      dept
TO      user01
WITH    GRANT OPTION ;
```

- PUBLIC 사용자에게 대한 권한 부여

```
GRANT  update
ON      hr.dept
TO      PUBLIC ;
```

## 4. 사용자 관리

### (2) 권한

#### 2) 객체 권한

- 권한 박탈 구문

```
REVOKE 권한명 | ALL  
ON      객체명  
FROM    사용자명 | 롤 | PUBLIC ;
```

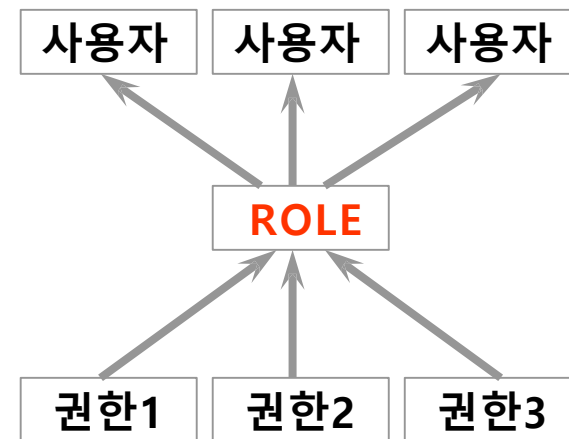
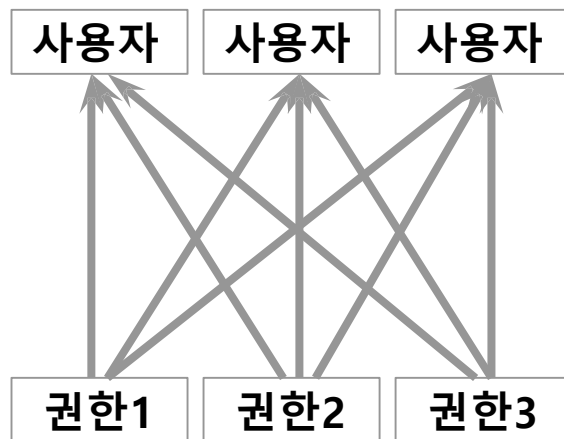
```
REVOKE select, update  
ON      departments u  
FROM    ser01 ;
```

## 4. 사용자 관리

### (3) ROLE

#### 1) ROLE의 정의

- 사용자에게 많은 권한을 일일이 줘야 하는 번거로움을 줄이기 위해 사용하는 권한의 집합





## 4. 사용자 관리

### (3) ROLE

#### 2) ROLE의 생성

```
CREATE ROLE man_role ;
```

#### 3) ROLE에 권한을 부여

```
GRANT create table, create view  
TO man_role ;
```

#### 3) ROLE을 사용자에게 부여

```
GRANT man_role  
TO user01, scott ;
```

---

## 4. 사용자 관리

### (4) 부여한 권한 확인

데이터디렉터리	설 명
ROLE_SYS_PRIVS	ROLE에게 부여된 시스템 권한
ROLE_TAB_PRIVS	ROLE에게 부여된 테이블 권한
USER_ROLE_PRIVS	사용자에 의해 액세스 가능한 ROLE
USER_TAB_PRIVS_MADE	사용자가 부여된 객체 권한
USER_TAB_PRIVS_RECO	사용자에게 부여된 객체 권한
USER_COL_PRIVS_MADE	사용자가 객체의 열에 대해 부여한 객체 권한
USER_COL_PRIVS_RECO	특정 열에 대해 사용자에게 부여된 객체 권한