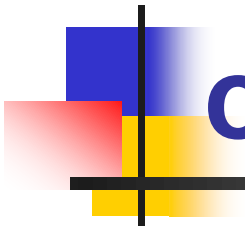
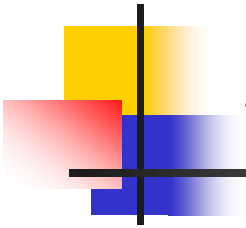


Oracle 10강 –사용자 관리

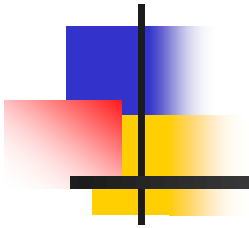
양 명 숙
[now4ever7@gmail.com]



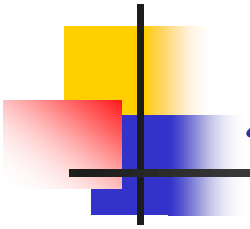
목차

- 데이터 사전
- 사용자 관리
  - 권한과 롤
- 테이블 스페이스 생성
- 사용자 생성

- 트리거



사용자 관리



사용자 계정

- 사용자 계정
  - sys, system – 오라클 데이터베이스 관리자
    - 오라클 데이터베이스 내에서 시스템적인 작업을 할 수 있는 DBA 권한이 자동으로 할당되어 있음
    - sys – 시스템 정보를 갖고 있는 모든 객체들(테이블, 뷰등)에 접근할 수 있고 이러한 객체들의 소유자임
      - 데이터 딕셔너리의 소유자
      - sys 사용자가 system 사용자보다 더 많은 권한을 가짐
    - system – 오라클 데이터베이스의 여러 옵션들과 툴 등에 대한 정보를 가진 객체들의 소유자
      - 모든 권한이 SYS와 같으나 데이터베이스 생성 권한 없음
  - scott, hr: SAMPLE 사용자 계정

사용자 관리

- user 와 스키마
  - user : 사용자, 오라클 서버에 접속하기 위해 사용하는 것이 user
  - schema : 특정 사용자(user)가 만들어 놓은 모든 object 집합

- 예) scott schema : 오라클 서버안에 scott 계정으로 로그인해서 만들어 놓은 모든 것을 다 모아 둔 것
  - scott user 가 만든 table, index, view, constraint, sequence 등을 통틀어서 scott schema 라고 함

스키마 이름.테이블 이름
scott.emp =>scott 스키마에 있는 emp 테이블, scott 사용자로 로그인해 있기 때문에 scott 생략

스키마(Schema) - 임의의 사용자가 생성한 모든 데이터베이스 객체들을 말하며, 스키마 이름은 그 사용자의 이름과 같다.

테이블 스페이스

TABLESPACE_NAME	FILE_NAME
USERS	C:₩APP₩user ₩ORADATA₩ORCL8₩USERS01.DBF
UNDOTBS1	C:₩APP₩ user₩ORADATA₩ORCL8₩UNDOTBS01.DBF
SYSAUX	C:₩APP₩uesr ₩ORADATA₩ORCL8₩SYSAUX01.DBF
SYSTEM	C:₩APP₩user ₩ORADATA₩ORCL8₩SYSTEM01.DBF
EXAMPLE	C:₩APP₩user ₩ORADATA₩ORCL8₩EXAMPLE01.DBF

- 오라클 – 데이터를 관리함
  - 데이터를 저장하고 추출, 삭제, 변경하는 작업을 함
  - 데이터는 파일에 저장됨

C:□app□user명□oradata□orcl

- 오라클 데이터베이스
  - [1] 데이터파일들을 가지고 있으며, 이 파일들에 데이터가 저장됨
    - 파일 – 데이터가 저장되는 물리적인 공간
  - [2] 오라클 내부에서는 논리적인 개념으로 데이터들을 관리
    - 데이터 블록(data block), 익스텐트(extend), 세그먼트(segment), 테이블스페이스(tablespace)
      - 데이터 블록 – 오라클에서 데이터를 저장하는 가장 최소의 논리적 단위
      - 익스텐트 – 데이터 블록이 모여 익스텐트가 됨
      - 세그먼트 – 익스텐트가 모여서 세그먼트
      - 테이블 스페이스 – 세그먼트가 모여서 테이블스페이스가 됨. (테이블, 인덱스, 프로시저, 뷰 등 여러 오라클 객체들이 저장됨)
- 실제로 물리적인 데이터 파일(.dbf나 .ora 파일)은 테이블스페이스와 대응됨
  - 하나의 테이블스페이스는 최소 1개의 데이터파일로 구성됨

오라클의 논리적 저장구조

- [1] 데이터블럭(Data Block)
  - 오라클에서 데이터를 저장하는 가장 최소의 논리적 단위
  - 데이터블럭의 size – 2K, 4K, 8K, 16K, 32K, 64K
    - 기본크기는 8K

```
-- 데이터블럭의 size 조회
select value from v$parameter where name = 'db_block_size';
```

VALUE
8192

- [2] 익스텐트(Extent)
  - 데이터블럭의 모임이며 8개의 데이터블럭이 모여서 1개의 익스텐트가 됨
  - 익스텐트(Extent)는 세그먼트(Segment)의 할당 단위임

```
select *  from dba_extents where owner='HR' and  segment_name='EMPLOYEES';
```

```
--8192*8 => 65536 byte
```

OWNER	SEGMENT_NAME	PARTITION_NAME	SEGMENT_TYPE	TABLESPACE_NAME	EXTENT_ID	FILE_ID	BLOCK_ID	BYTES	BLOCKS	RELATIVE_FNO
HR	EMPLOYEES		TABLE	EXAMPLE	0	5	200	65536	8	5

오라클의 논리적 저장구조

- [3] 세그먼트(Segment)
  - 하나 이상의 익스텐트(Extent)로 구성되며 테이블, 인덱스를 오라클에서는 세그먼트(Segment)라고 부른다.

```
select *   from dba_segments where owner='HR';
```

- [4] 테이블스페이스(Tablespace)
  - 세그먼트(Segment)들을 저장하는 논리적인 공간 이름
  - 세그먼트가 모여서 테이블스페이스가 됨. (테이블, 인덱스, 프로시저, 뷰 등 여러 오라클 객체들이 저장됨)

- [5] 데이터베이스(Database)
  - 테이블스페이스(Tablespace)들의 모임을 데이터베이스라고 함

데이터 파일(Data File)
- 디스크에 존재하는 실제 파일이며, 하나 이상의 데이터파일이 모여서 테이블스페이스(Tablespace)를 구성하게 된다.

테이블 스페이스 생성

- sys나 system 계정으로 로그인해서 사용자 생성, 테이블스페이스 생성하기
- sqlplus "/as sysdba ==>시스템dba접속

테이블스페이스의 기본 용량을 자동으로 늘려주는 옵션 (autoextend)

- 테이블 스페이스의 생성

```
<형식>  create tablespace 테이블스페이스명
          datafile  '절대경로□파일명'  size  크기
          AUTOEXTEND ON NEXT 크기(M);  --자동 증가옵션
```

예>create tablespace kim_tb
datafile 'c:₩oracle₩data₩kim.dbf' size 100M AUTOEXTEND ON NEXT 10M;

※C:□oracle□data□ 폴더는 미리 만들어야 함

- 테이블스페이스 삭제

```
DROP TABLESPACE  kim_tb
INCLUDING CONTENTS AND DATAFILES;
```

- 테이블스페이스 정보검색

```
select tablespace_name, FILE_NAME from dba_data_files;
select * from dba_tablespaces;
```

사용자 생성

- 사용자 생성

<형식> create user 사용자아이디
identified by 암호
default tablespace 테이블스페이스명;

- create user testuser2
  identified by test
  default tablespace kim_tb;

- 사용자 권한부여

grant create session to testuser2; (접속권한 부여)

• 권한 주기
- connect : 접속, 쿼리 실행, 테이블 생성
- resource : view, stored Procedure 등 서버의 자원을 만들고 쓸수 있는 권한

GRANT connect, resource TO 사용자이름;

- grant connect, resource to testuser2;

사용자 생성

- 등록된 계정 목록 보기

```
SELECT * FROM dba_users
WHERE username LIKE '사용자이름%' ORDER BY username;
```

- 사용자 삭제

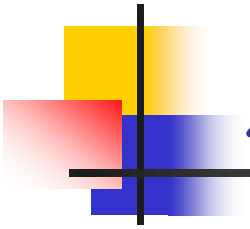
```
DROP USER 사용자이름
```

- 사용자 접속

```
conn 아이디/패스워드
conn  testuser2/test
conn / as sysdba
```

- 현재 사용자 확인

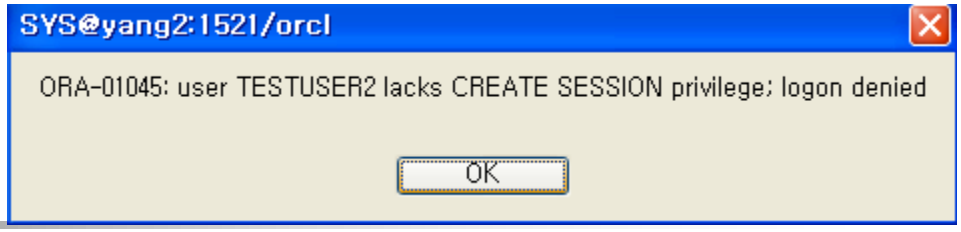
```
show user
```



사용자 관리

- 잠긴 계정을 열기
  - alter user 사용자ID account unlock;
- 계정을 잠그기.
  - alter user 사용자ID account lock;
- 기존 계정의 암호 변경
  - alter user 사용자ID idendified by 새로운 암호;

사용자 관리



- 권한 (privilege)
  - 사용자가 특정한 테이블에 접근하도록 하거나, 해당 테이블의 특정 sql 명령(insert, update 등)만을 실행할 수 있도록 제한할 때 사용
  - 사용자를 생성하기 위해서는 create user 권한을 소유하고 있는 sys나 system 계정으로 접속해야 함
  - 사용자 생성하기

```
create user testuser2
identified by test
default tablespace kim_tb;
```

  - 생성된 사용자 정보 확인하기
    - select username, default_tablespace from dba_users
      where username='TESTUSER2';
  - 생성된 사용자로 데이터베이스에 접속하기
    - conn testuser2 => 접속 거절됨(어떤 권한도 부여 받지 않았으므로)
    - 데이터베이스에 접속하려면 모든 사용자는 세션을 만들 수 있는 권한이 있어야 접속할 수 있음

사용자 관리

- system이나 sys 사용자로 접속하여 testuser2 사용자에게 create session 권한 부여하기
  - grant create session to testuser2;
  - => 접속 가능해짐
- 테이블 생성하기 ORA-01031: 권한이 불충분합니다
  - 테이블 생성 권한을 부여 받아야 함
  - grant create table to testuser2;
- 접속한 사용자가 데이터베이스에서 특정 작업을 하기 위해서는 적절한 권한이 요구됨
- 권한을 박탈하는 방법

```
SELECT * FROM dba_users
   WHERE username LIKE 'TEST%';

select * from dba_role_privs
    where grantee = 'TESTUSER2';

select * from dba_sys_privs
     where grantee='TESTUSER2';
```

revoke 권한 on object from user

revoke create session from testuser2;

사용자 관리

- 롤(Role)
  - 권한에 대한 부여와 관리를 보다 편리하게 하기 위해서 사용
  - 권한의 그룹
  - **롤 속에 여러 가지 권한을 넣어** 두고 사용자에게 롤 하나를 주면 그 안에 있는 모든 권한을 다 받게 되는 것
  - 오라클이 제공하는 유용한 role => dba_roles 참고
    - select * from dba_roles;

```
select * from dba_sys_privs where grantee = 'CONNECT';
select * from dba_sys_privs where grantee = 'RESOURCE';
```

롤

- 롤 생성하기
  - create role testrole;

```
select * from dba_roles;
```

- 롤에 create session, create table 권한 할당하기
  - grant create session, create table to testrole;

```
select * from dba_sys_privs where grantee = 'TESTROLE';
```

- TESTUSER3 사용자에게 testrole 할당하기
  - grant testrole to TESTUSER3;
- 어떤 사용자가 어떤 롤을 사용하는지 확인하기
  - select * from dba_role_privs
    where grantee = 'TESTUSER3';
- 어떤 롤에 어떤 권한이 있는지 확인하기
  - select * from dba_sys_privs where grantee='CONNECT';
  - select * from dba_sys_privs where grantee='RESOURCE';

권한

- 권한
  - 시스템 권한(System privilege)
  - 객체 권한 (Object privilege)
    - SELECT, INSERT, UPDATE, DELETE 권한

• 시스템 권한(System privilege)

권한	설명
CREATE TYPE	자신 소유의 OBJECTS를 생성하는 권한
CREATE ANY TYPE	모든 OBJECTS를 생성하는 권한
ALTER ANY TYPE	모든 OBJECTS를 변경하는 권한
DROP ANY TYPE	모든 OBJECTS를 삭제하는 권한
EXECUTE ANY TYPE	모든 OBJECTS를 사용하거나 참조하는 권한

System 관련 주요 Privilege

대분류	PRIVILEGE	설명
INDEX	CREATE ANY INDEX	소유자에 상관없이 모든 테이블에 인덱스를 생성할 수 있는 권한
	DROP ANY INDEX	소유자에 상관없이 모든 인덱스를 삭제할 수 있는 권한
	ALTER ANY INDEX	소유자에 상관없이 모든 인덱스를 수정할 수 있는 권한
TABLE	CREATE TABLE	자신 소유의 테이블을 생성할 수 있는 권한
	CREATE ANY TABLE	소유자에 상관없이 다른 user 이름으로 테이블을 생성할 수 있는 권한
	ALTER ANY TABLE	소유자에 상관없이 모든 테이블의 구조를 수정할 수 있는 권한
	DROP ANY TABLE	소유자에 상관없이 모든 사용자의 테이블을 삭제할 수 있는 권한
	UPDATE ANY TABLE	소유자에 상관없이 모든 사용자의 테이블을 업데이트할 수 있는 권한
	DELETE ANY TABLE	소유자에 상관없이 모든 사용자의 테이블의 데이터를 삭제할 수 있는 권한
	INSERT ANY TABLE	소유자에 상관없이 모든 사용자의 테이블에 데이터를 삽입할 수 있는 권한
SESSION	CREATE SESSION	서버에 접속할 수 있는 권한
	ALTER SESSION	접속 상태에서 환경값을 변경할 수 있는 권한
	RESTRICTED SESSION	Restricted 모드로 open 된 DB에 접속할 수 있는 권한

System 관련 주요 Privilege

TABLESPACE	CREATE TABLESPACE	Tablespace를 만들 수 있는 권한
	ALTER TABLESPACE	Tablespace를 수정할 수 있는 권한
	DROP TABLESPACE	Tablespace를 삭제할 수 있는 권한
	UNLIMITED TABLESPACE	Tablespace 사용 용량을 무제한으로 허용하는 권한, 즉 quota 옵션 적용을 받지 않게 됨

sysoper/ sysdba privilege

PRIVILEGE	할 수 있는 일
SYSOPER	Startup / shutdown
	Alter database mount / open
	Alter database backup control file to …
	Recover database
	Alter database archivelog
	Restricted session
SYSDBA	SYSOPER PRIVILEGE with admin option
	Create database
	Alter tablespace … begin backup / end backup
	Recover database until

사용자 관리

- system 관련 권한 할당하기/해제하기
  - grant create table, create session to TESTUSER4;
  - revoke create table from TESTUSER4;

- object 권한 할당하기/해제하기
  - grant select on hr.employees to TESTUSER4;
  - grant update on hr.employees to TESTUSER4 with grant option;
  - revoke select on hr.employees from TESTUSER4;
  - revoke update on hr.employees from TESTUSER4;

with grant option;
 => 권한을 위임하는 기능
또 다른 사용자에게 권한을 할당해 줄 수 있게 됨
object privilege 에서 사용

사용자 관리

with admin option;
=> 권한을 위임하는 기능
또 다른 사용자에게 권한을 할당해 줄 수 있게 됨

- 사용자 권한 부여
  - grant connect,resource to TESTUSER5;
  - grant connect, dba to TESTUSER5 with admin option;
  - grant create view to TESTUSER5;
  - revoke create view from TESTUSER5;

- 사용자가 가지고 있는 권한 조회
  - select * from dba_sys_privs
    where grantee = 'TESTUSER5';
  - select * from dba_role_privs (롤 조회)
    where grantee = 'TESTUSER5';
- 어떤 role(권한의 그룹) 에 어떤 권한이 있는지 확인하기
  - select * from dba_sys_privs
    where grantee = 'CONNECT';
  - select * from dba_sys_privs
    where grantee = 'RESOURCE';

사용자 관리

- 뷰의 select 권한 할당하기
  - grant select on scott.v_emp1 to hr;
  - revoke select on scott.v_emp1 from hr;

DATA Dictionary

- DATA Dictionary
  - **데이터베이스** 내에 저장된 모든 **객체의 정보를 제공**해 주는 테이블로 사용자가 생성한 테이블이 어떤 것인지, 어떤 구조로 생성되어 있는지, 데이터베이스의 성능이 어떤지 등을 보여주는 테이블
  - 데이터 사전은 테이블이나 뷰의 형태로 되어 있음
  - 데이터베이스 생성시 SYSTEM 테이블스페이스에 저장됨
- 데이터 사전에는 어떤 내용이 있나
  - 데이터베이스의 모든 스키마 오브젝트(테이블, 뷰, 시퀀스)의 정의
  - 스키마 오브젝트에 의해 현재 사용된 공간
  - 칼럼들의 기본값
  - 제약 조건 정보
  - 오라클 사용자 이름
  - 각 사용자에게 부여된 권한과 롤
  - Auditing 정보

DATA Dictionary

- Data Dictionary의 종류
  - 1) DBA_xxxx : 데이터베이스 관리를 위한 정보를 제공
  - 2) ALL_xxxx : 자신이 생성한 Object와 다른 사용자가 생성한 Object중에 자신이 볼 수 있는 정보를 제공
    - 사용자가 접근 가능한 모든 스키마의 정보를 제공
    - 권한을 부여 받음으로써 가능
  - 3) USER_xxxx : 자신이 생성한 Object정보를 제공
    - 현재 데이터베이스에 접속한 사용자가 소유한 객체의 정보를 제공
  - 4) V$_xxxx : DB의 성능 분석/통계 정보를 제공하며 X$ 테이블에 대한 View
  - 5) X$_xxxx : DB의 성능 분석/통계 정보를 제공하는 테이블

접두사 'user_' 를 갖는 사전

- 모든 자료사전의 정보를 출력
  - select * from dictionary;
- 자료사전 테이블의 각 컬럼에 대한 설명을 출력
  - SELECT * FROM DICT_COLUMNS;
- 사용자가 생성한 모든 테이블 출력
  - SELECT * FROM USER_TABLES;
  - SELECT TABLE_NAME, TABLESPACE_NAME FROM USER_TABLES;
- 사용자가 생성한 모든 인덱스를 출력
  - SELECT * FROM USER_INDEXES;
  - SELECT INDEX_NAME, INDEX_TYPE, TABLE_OWNER FROM USER_INDEXES;
- 사용자가 생성한 모든 뷰를 출력
  - SELECT * FROM USER_VIEWS;

접두사 'user_' 를 갖는 사전

```
SELECT * FROM USER_cons_columns;
--제약조건 컬럼 확인

SELECT * FROM USER_ind_columns;
--index 컬럼 확인

SELECT * FROM USER_tab_columns;
--default 값 확인
```

- 사용자가 생성한 모든 시퀀스를 출력
  - SELECT * FROM USER_sequences;
- 사용자가 생성한 모든 제약조건을 출력
  - SELECT * FROM USER_CONSTRAINTS;
  - SELECT OWNER, CONSTRAINT_NAME, TABLE_NAME,constraint_type FROM USER_CONSTRAINTS;
- 사용자가 소유하고 있는 모든 객체들의 정보 제공
  - select * from user_objects;
- 사용자가 작성한 저장 프로시저의 Text Source 정보를 보여줌
  - select text from user_source where name='프로시저 이름';

접두사 'all_'를 갖는 사전

- 자신이 생성한 것은 아니나, 접근은 가능한 경우 이런 객체들의 정보를 All_로 시작하는 테이블에서 제공함
  - 다른 사람으로부터 특정 객체에 접근할 수 있는 권한을 부여 받은 경우
  - select * from all_tables;
  - 예) conn hr

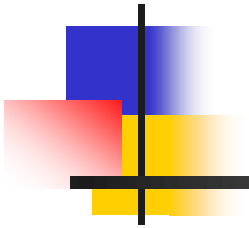
    grant select on employees to testuser;

접두사 'dba_'를 갖는 사전

- dba_ 는 데이터베이스 관리자만이 접근 가능한 테이블
  - 데이터베이스 관리를 위한 정보를 얻을 수 있음
  - dba_ 로 시작하는 데이터 사전의 정보는 dba 권한을 가진 사용자만이 접근할 수 있음
- 현재 데이터베이스에 존재하는 모든 계정의 리스트를 얻는 질의
  - conn sys
  - select username from dba_users;
    - 예) select username, default_tablespace from dba_users
      where username='HR';
- 테이블 스페이스 정보 검색
  - select tablespace_name, FILE_NAME from dba_data_files;

접두사 'dba_'를 갖는 사전

- 오라클이 제공하는 유용한 role
  - select * from dba_roles;
- 어떤 사용자가 어떤 롤을 사용하는지 확인하기
  - select * from dba_role_privs
    where grantee = 'HR';
- 어떤 롤에 어떤 권한이 있는지 확인하기
  - select * from dba_sys_privs where grantee='CONNECT';
  - select * from dba_sys_privs where grantee='RESOURCE';



트리거(Trigger)

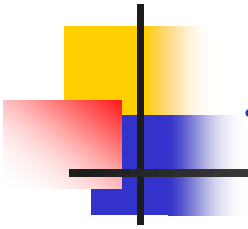
트리거(Trigger)

insert, update, delete의 **DML문이나 DDL문의 실행**을 데이터베이스에서는 **특정 이벤트가 발생되었다**고 하는데, 이런 이벤트가 발생하면 자동으로 정해진 동작을 실행하는 데이터베이스 객체를 트리거라고 함

- 트리거
  - 서브 프로그램 단위의 하나인 트리거는 테이블, 뷰, 스키마 또는 데이터베이스에 관련된 PL/SQL 블록(또는 프로시저)으로 관련된 특정 사건(event)이 발생될 때마다 자동으로 해당 PL/SQL 블록이 실행됨
  - 사전적인 의미 – 방아쇠
    - 방아쇠를 당기는 이벤트가 발생하면 총알이 발사된다는 의미처럼 오라클에서도 어떤 이벤트가 발생할 경우 연관된 다른 작업이 자동으로 수행된다는 뜻
    - 예) 입고 테이블에 상품이 입고되면(이벤트 발생) 재고 테이블에 자동으로 재고가 증가하게 만드는 것
  - 트리거는 데이터베이스 내에 오브젝트로서 저장되어 관리됨
  - 트리거 자체는 사용자가 지정해서 실행할 수 없으며, 오직 트리거 생성시 정의한 특정 사건(event)에 의해서만 묵시적으로 자동실행(Fire)이 이루어짐

트리거(Trigger)

- 트리거는 잘 사용하면 아주 편리한 기능이지만, 잘못 생성하거나 너무 많이 생성하게 되면 관련 오브젝트끼리 복잡한 종속 관계가 되어 성능 저하가 생길 수도 있으니 주의해야 함
- 트리거를 생성하려면 create trigger,
- 수정하려면 alter trigger,
- 삭제하려면 drop trigger 의 권한이 필요함
- 데이터베이스 전체의 트리거 조작은 administer database trigger 시스템 권한이 필요함
- 트리거에 대한 정보는 user_triggers 딕셔너리 조회
- trigger 를 이루는 trigger 몸체(실행부)에 TCL명령(commit, rollback, savepoint 명령)이 포함될 수 없음



주요 트리거 유형

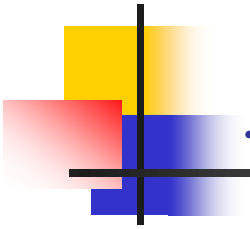
- [1] 단순 DML Trigger
  - 1) before trigger
    - 테이블에서 DML 이벤트를 trigger 하기 전에 trigger 본문을 실행함
  - 2) after trigger
    - 테이블에서 DML 이벤트를 trigger 한 후에 trigger 본문을 실행함
  - 3) instead of trigger
    - Trigger문 대신 Trigger 본문을 실행하며, 다른 방법으로는 수정이 불가능한 뷰에 사용됨
    - instead of trigger 를 사용하면 서버 프로세스가 Trigger 상에 기술된 쿼리를 직접 수행해서 원본 테이블의 데이터를 변경하게 해줌

주요 트리거 유형

- DML 트리거
  - 문장 트리거 – 영향을 받는 행이 전혀 없더라도 트리거가 한 번은 실행됨
  - 행 트리거 – 테이블이 트리거 이벤트의 영향을 받을 때마다 실행되고, 트리거 이벤트의 영향을 받는 행이 없을 경우에는 실행되지 않음
    - 행 트리거로 생성하려면 For Each Row 라는 구절을 사용하면 됨
    - 행 레벨 트리거가 실행될 때 PL/SQL 런타임 엔진은 두 개의 데이터 구조를 생성하고 채움
      - OLD – 트리거가 처리한 레코드의 원래 값을 저장함
      - NEW – 새 값을 포함함
        - => 위 두 가지 값을 사용하여 변경 전과 변경 후의 데이터를 조작할 수 있음
      - 사용할 때는 반드시 위 값 앞에 콜론(:)을 붙여서 사용해야 함
      - 예) :OLD.value, :NEW.no 등
      - 행 레벨 트리거에서만 사용 가능

주요 트리거 유형

- [2] DML 이 아닌 트리거
- 1) DDL 이벤트 트리거
  - DML 트리거와 거의 동일하지만 트리거를 활용하여 DDL 작업을 하는 것만 다름
- 2) 데이터베이스 이벤트 트리거
  - 데이터베이스 내에서 생기는 일들을 관리하기 위해서 생성하는 트리거
  - 사용자 관련 이벤트가 있고, 시스템 관련 이벤트가 있음
    - user 이벤트 트리거
      - 사용자가 발생시키는 작업에 트리거를 생성함
      - create, alter, drop
      - 로그온 또는 로그오프
    - 데이터베이스 또는 시스템 이벤트 트리거
      - 데이터베이스 전체에 영향을 주는 작업에 트리거를 생성함
      - 데이터베이스 종료 또는 시작
      - 발생한 특정 오류(또는 임의의 오류)



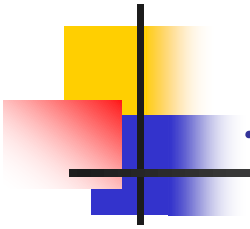
트리거 구조

- 트리거는 트리거가 실행되는 시점(Timing)
  - before – 이벤트 발생 전
  - after – 이벤트 발생 후
- 트리거를 실행시키는 사건(Event)
  - 테이블/ 뷰에 관련된 DML_event
  - 스키마/데이터베이스에 관련된 DDL_Event와 Database_Event
- 트리거와 관련된 테이블/뷰/스키마/데이터베이스
- 트리거 몸체부(Body) 로 구성됨

트리거 생성

```
create or replace Trigger 트리거 이름
        트리거 실행 시점[before|after]
        이벤트[insert|update|delete]
        on {테이블이름 | 뷰이름 | 스키마 | 데이터베이스}
[for each row]
begin
        트리거 몸체
end;
```

- on – 트리거가 발생되는 레벨 또는 대상을 지정하는 절
  - 트리거의 대상인 테이블/뷰 이름 기술
- for each row – 테이블/뷰의 트리거를 행 트리거로 명시하는 절
  - 매번 추가되는 행의 수만큼 트리거가 발생함



트리거 관리

- 트리거 관련 권한
  - 스키마에서 트리거를 생성, 변경 및 삭제할 수 있는 권한
    - grant create trigger to scott;
    - grant alter any trigger to scott;
    - grant drop any trigger to scott;
  - 데이터베이스에서 트리거를 생성할 수 있는 권한
    - grant administer database trigger to scott;

트리거 예제

```
--[1] 부서 테이블(dept)에 insert문 실행 후 메시지를 출력하는 트리거(문장 레벨 트리거)

create or replace trigger print_message
after insert on dept
begin
    dbms_output.put_line('정상적으로 추가되었습니다');
end;

--dept 테이블에 입력하기
insert into dept(deptno, dname, loc)
values(75, '정보관리팀','서울');
```

```
--[2] 테이블에 데이터를 입력할 수 있는 시간 지정하기(문장 레벨 트리거)
create table t_order(
    no number,
    ord_code    varchar2(10),
    ord_date    date
);

--입력시간이 15:20 ~ 15:30 일 경우만 입력을 허용하고, 그 외 시간일 경우는 에러를 발생시키는
    트리거
create or replace trigger t_order_trigger
before insert on t_order
begin
    if to_char(sysdate, 'hh24:mi') not between '15:20' and '15:30' then
        raise_application_error(-20100, '허용 시간이 아닙니다');
    end if;
end;

insert into t_order
values(1, 'C100',sysdate); --

select sysdate from dual;

select * from t_order;
```

```
--[3] 테이블에 입력될 데이터 값을 지정하고, 그 값 외에는 에러를 발생시키는 트리거(행 레벨 트리거)
--제품 코드가 'C100'인 제품이 입력될 경우 입력을 허용하고, 나머지 제품은 모두 에러를 발생시키는 트리거

create or replace trigger t_order_trigger2
before insert on t_order
for each row --행 레벨 트리거- 매번 추가되는 행의 수만큼 트리거가 발생함
begin
    if :new.ord_code not in('C100') then
        raise_application_error(-20200, '제품코드가 틀립니다');
    end if;
end;

insert into t_order
values(2, 'C100',sysdate); --문장 레벨 트리거 때문에 허용시간이 지나면 에러

--문장 레벨 트리거 삭제하기
drop trigger t_order_trigger;

insert into t_order
values(2, 'C100',sysdate); --성공

select * from t_order;

insert into t_order
values(2, 'C200',sysdate); --제품 코드가 틀립니다 에러
```

[4] 기존 테이블(t_test1)에 데이터가 업데이트될 때 기존 내용을 백업 테이블(t_test1_bak)로 이동시키는 트리거

```
create table t_test1(
    no number,
    name    varchar2(10)
);

create table t_test1_bak
as
select * from t_test1;

insert into t_test1 values(1, 'AAA');
insert into t_test1 values(2, 'BBB');
commit;

create or replace trigger t_move
before update on t_test1
for each row
begin
    insert into t_test1_bak
    values(:old.no, :old.name);
end;

select * from t_test1;
select * from t_test1_bak;

update t_test1 set  name='CCC' where no=1;
```

-old 연산자는 변경할 때 변경 전의 값을 가지고 있음
-new : 데이터가 추가 혹은 변경되면 new 연산자로 변경 후의 값을 얻을 수 있음
-user : 현재 접속중인 사용자를 나티냄

[5] 기존 테이블(t_test2)의 데이터가 삭제될 때 기존 내용을 백업 테이블(t_test2_bak)로 이동시키며 이때 백업 테이블에 삭제한 시간, 삭제 전 데이터를 모두 기록하는 트리거

```
create table t_test2(
    no number,
    name    varchar2(10)
);
create table t_test2_bak(
    no number,
    name    varchar2(10),
    regdate date default sysdate
);
insert into t_test2 values(1, 'AAA');
insert into t_test2 values(2, 'BBB');
commit;

create or replace trigger tr_delete
after delete on t_test2
for each row
begin
    insert into t_test2_bak(no, name, regdate)
    values(:old.no, :old.name, sysdate);
end;

select * from t_test2;
select * from t_test2_bak;
delete from  t_test2 where no=1;
```

[6] 기존 테이블(t_test3)의 추가, 변경, 삭제된 내용을 별도의 로그 테이블을 생성하여 기록을 남기도록 트리거를 설정

```
create table t_test3(
    no number,
    name    varchar2(10)
);

create table t_test3_history(
    o_no number, --변경전이나 삭제된 데이터를 저장하는 칼럼은 o로 시작
    o_name    varchar2(10),
    n_no number, --변경 후나 추가된 데이터를 저장하는 칼럼은 시작은 n으로 정의
    n_name    varchar2(10),
    who varchar2(30), --어떤 사용자가 어떤 작업을 언제작업 했는지 정보를 저장
    regdate    date default sysdate,
    chk char(1)
);

insert into t_test3 values(1, 'AAA');
insert into t_test3 values(2, 'BBB');

commit;
```

```
create or replace trigger check_t_test3_history
after update on t_test3
for each row --매번 추가되는 행의 수만큼 트리거가 발생함
begin
    if updating then
        insert into t_test3_history(o_no, o_name, n_no, n_name, who, regdate, chk)
        values(:old.no, :old.name, :new.no, :new.name, user, sysdate, 'U');
    end if;
end;
```

--old 연산자는 변경할 때 변경 전의 값을 가지고 있음

--new : 데이터가 추가 혹은 변경되면 new 연산자로 변경 후의 값을 얻을 수 있음

--user : 현재 접속중인 사용자를 나타냄

```
select * from t_test3;
select * from t_test3_history;

update  t_test3
set name='DDD'
where no=1;

rollback;
```

NO	NAME
1	DDD
2	BBB

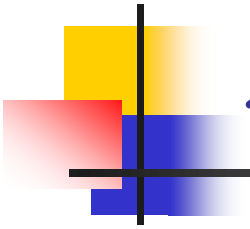
O_NO	O_NAME	N_NO	N_NAME	WHO	REGDATE	CHK
1	AAA	1	DDD	SCOTT	2014-04-08 오전 10:36:56	U

```
create table 상품(
    품번 number,
    항목명 varchar2(20),
    단가 number
);
create table 입고(
    품번 number,
    수량 number,
    금액 number
);
create table 판매(
    품번 number,
    수량 number,
    금액 number
);
create table 재고(
    품번 number,
    수량 number,
    금액 number
);
insert into 상품
values(100, '새우깡',900);
insert into 상품
values(200, '감자깡',900);
insert into 상품
values(300, '맛동산',1000);
```

실습

```
insert into 입고 values(100, 10,9000);
insert into 입고 values(200, 10,9000);
insert into 입고 values(300, 10,10000);

insert into 재고 values(100, 10,9000);
insert into 재고 values(200, 10,9000);
insert into 재고 values(300, 10,10000);
```



실습

- 상품이 입고되면 재고 테이블에서 자동으로 해당 상품의 재고 수량과 금액이 증가되는 트리거 작성하기
  - insert into 입고 values(100, 2, 1800);
- 상품이 판매되면 재고 테이블에서 자동으로 해당 상품의 재고 수량과 금액이 감소되는 트리거 작성하기
  - insert into 판매 values(100, 3, 2700);