User Management

강사 : 강병준

설치 시 생성되는 사용자

- 기본 사용자

유저 명	비밀번호
SYS	CHANGE_ON_INSTALL
SYSTEM	MANAGER
SCOTT	TIGER

설치 시 생성되는 사용자

- SYS
  - 데이터베이스의 모든 기본 테이블과 뷰는 SYS스키마에 저장
  - 기본테이블과 뷰는 oracle을 운영하는데 꼭 필요
  - SYS스키마의 테이블은 data dictionary의 무결성 유지관리를 위해 oracle에 의해 처리
  - 대부분의 database사용자는 SYS계정으로 접속하지 말아야 함.
- SCOTT
  - SCOTT유저는 일반 사용자로 오라클의 기본적인 SQL문을 테스트를 하기 위한 테이블과 데이터들이 있습니다.

설치 시 생성되는 사용자

- SYSTEM
  - 관리정보를 화면으로 보여주는 추가 테이블과 뷰, 오라클 도구가 사용하는 내부테이블과 뷰를 만들 수 있다.
  - SYSTEM유저는 모든 시스템 권한을 가지고 있다.
- 참 고
  - SYS와 SYSTEM은 사용자를 만들거나 데이터베이스를 관리 할 수 있는 권한(DBA)을 가지고 있다.
  - SYS나 SYSTEM유저에 테이블을 생성하거나, 일반 데이터들을 Insert하는 방법은 아주 좋지 않은 방법임
  - SYS나 SYSTEM유저는 데이터베이스를 관리하기 위해서만 사용되어야 함

사용자 접근 제어

- **다중 사용자 환경에서는 데이터베이스 액세스와 사용의 보안 유지가 요구 됩니다.**
  - 데이터베이스 액세스 제어
  - 데이터베이스에서 특정 객체에 대한 액세스 제공
  - 오라클 데이터 사전으로 주어지고 받는 Privilege 확인
  - 데이터베이스 객체에 대한 동의어 생성

- **데이터베이스 보안의 범주**
  - 시스템 보안: 사용자에 의해 허용된 시스템 작업 같은 시스템 수준에서의 데이터베이스의 액세스와 사용을 규정합니다.
    사용자 명
    사용자의 비밀 번호
    사용자에게 할당된 디스크 공간
  - 데이터 보안: 객체에 대해 사용자가 할 수 있는 작업을 규정합니다.

사용자 접근 제어

- **다중 사용자 환경에서는 데이터베이스 액세스와 사용의 보안 유지가 요구 됩니다.**
  - 데이터베이스 액세스 제어
  - 데이터베이스에서 특정 객체에 대한 액세스 제공
  - 오라클 데이터 사전으로 주어지고 받는 Privilege 확인
  - 데이터베이스 객체에 대한 동의어 생성

- **데이터베이스 보안의 범주**
  - 시스템 보안: 사용자에 의해 허용된 시스템 작업 같은 시스템 수준에서의 데이터베이스의 액세스와 사용을 규정합니다.

    사용자 명

    사용자의 비밀 번호

    사용자에게 할당된 디스크 공간

  - 데이터 보안: 객체에 대해 사용자가 할 수 있는 작업을 규정합니다.

사용자 생성1

- 데이타베이스내에서 사용자명은 다른 사용자와 롤에 대해 유일해야 함

```
CREATE USER user_name
IDENTIFIED {BY password|EXTERNALLY}
[DEFAULT TABLESPACE tablespace]
[TEMPORARY TABLESPACE tablespace]
[QUOTA {integer[K|M] | UNLIMITED} ON tablespace
[QUOTA {integer[K|M] | UNLIMITED} ON
tablespace]...]
[PROFILE {profile | DEFAULT}]
```

사용자 생성2

```
CREATE USER user_name
IDENTIFIED {BY password|EXTERNALLY}
[DEFAULT TABLESPACE tablespace]
[TEMPORARY TABLESPACE tablespace]
[QUOTA {integer[K|M] | UNLIMITED} ON tablespace
[QUOTA {integer[K|M] | UNLIMITED} ON tablespace]...]
[PROFILE {profile | DEFAULT}]
```

- IDENTIFIED BY

  : 데이타베이스 인증

  방식에 의한 사용자 인증
- IDENTIFIED EXTERNALLY : 운영체제 인증 방식에 의한 사용자 인증
- DEFAULT / TEMPORARY : 사용자에 대한 기본 / 임시 테이블스페이스 지정
- QUOTA : 사용자에게 허용된 기본 테이블스페이스내의 공간 크기 지정
- UNLIMITED : 기본 테이블스페이스의 전체 가용 공간 사용 가능
- PROFILE : 자원에 대한 제한 사항 지정

사용자 생성 2

```
SQL> connect /as sysdba
SQL> create user jason identified by price;
```

```
SQL> create user henry identified by hooray
  2  default tablespace users
  3  temporary tablespace temp;
```

```
SQL> connect jason/price
```

```
ORA-01045: user JASON lacks CREATE SESSION
                privilege; logon denied
```

```
SQL> connect /as sysdba
SQL> grant create session to jason;
```

사용자(users)

```
SQL> connect jason/price
```

```
SQL> connect /as sysdba
SQL> create user steve identified by button;
SQL> create user gail identified by seymour;
SQL> grant create session to steve, gail;
```

사용자 생성

- DBA는 CREATE USER문장을 사용하여 사용자를 생성 합니다. 사용자는 생성성 후 어떠한 권한도 가지지 않습니다. DBA는 이때 그 사용자에게 여러 권한을 부여 합니다. 이 권한은 데이터베이스 수준에서 사용자가 할 수 있는 것이 무엇인가를 결정 합니다.

```
CREATE  USER  user_name
IDENTIFIED  BY  password;
```

- 사용자 명은 yjb, 패스워드는 yoon인 사용자를 생성하고 CONNECT, RESOURCE권한을 부여하여라.

```
SQL> conn system/manager
Connected.
SQL> CREATE USER yjb
  2  IDENTIFIED BY yoon;

User created.
SQL> GRANT connect,resource TO yjb;

Grant succeeded.
```

사용자 변경

- ALTER USER 명령문을 사용하여 암호, 기본 테이블스페이스, 임시 테이블스페이스, 테이블스페이스 할당 크기 변경 가능

```
ALTER USER user_name
[IDENTIFIED {BY password|EXTERNALLY}]
[DEFAULT TABLESPACE tablespace]
[TEMPORARY TABLESPACE tablespace]
[QUOTA {integer[K|M] | UNLIMITED} ON tablespace
[QUOTA {integer[K|M] | UNLIMITED} ON
tablespace]...]
[PROFILE {profile | DEFAULT}]
```

사용자(users)

- 사용자(users) 암호(password) 변경(change)

```
SQL> alter user jason identified by marcus;
```

```
SQL> connect jason/marcus
SQL> password
  2  changing password for jason
  3  old password : ******
  4  new password : ******
  5  retype new password : ******
  6  password changed
```

- 사용자(users) 삭제(delete)

```
SQL> drop user jason [cascade];
```

권한

- **권한은 특정 SQL문장을 실행하기 위한 권한입니다. 데이터베이스 관리자는 데이터베이스와 그 객체에 대한 액세스를 사용자에게 부여하는 능력을 가진 상급 사용자 입니다. 사용자는 데이터베이스에 액세스하기 위해 system privilege가 필요하고 데이터베이스에서 객체의 내용을 조작하기 위해 object privilege가 필요합니다. 사용자는 관련 권한들의 이름있는 그룹인 role이나 다른 사용자에게 추가적으로 권한을 부여하기 위해 권한을 가질 수 있습니다.**

- 시스템 권한과 객체 권한으로 구분
  - 시스템 권한은 시스템 차원의 자원 관리, 사용자 스키마 객체 관리 등과 같은 데이타베이스 관리 작업을 할 수 있는 권한
  - 객체 권한은 테이블, 뷰, 시퀀스, 함수 등과 같은 스키마 객체에 특정 작업을 실행할 수 있는 권한

권한

시스템 권한	허가된 내용(Grantee:권한을 받은 사용자)
ALTER ANY TABLE	Grantee가 Schema에 있는 Index를 Alter할 수 있다.
ALTER ANY PROCEDURE	Grantee가 Schema에 내장 프로시저,함수,또는 패키지 바꾸기를 할 수 있다.
ALTER ANY ROLE	Grantee가 데이터베이스에서 역할 바꾸기를 할 수 있다.
ALTER ANY TABLE	Grantee가 Schema에서 TABLE이나 VIEW를 바꾸도록 할 수 있다.
ALTER ANY TRIGGER	Grantee가 Schema에서 데이터베이스 TRIGGER를 활성화,비활성화 또는 Compile 하게할 수 있다.
ALTER DATABASE	Grantee가 데이터베이스 바꾸기를 허용한다.
ALTER USER	Grantee가 사용자 바꾸기를 할 수 있다. 이 권한은 Grantee가 다른 사용자의 Password나 확인 방법을 바꾸도록 권한을 주고 DEFAULT TABLESPACE, TEMPORARY TABLESPACE, PROFILE, QUOTA의 양을 바꿀 수 있도록 한다.
CREATE ANY INDEX	Grantee가 어떤 Schema에서나 테이블에 인덱스 만들기를 허용한다.
CREATE ANY PROCEDURE	Grantee가 어떤 Schema에서 내장 프로시저,함수,패키지를 만들 수 있도록 허용한다.
CREATE ANY TABLE	Grantee가 어떤 Schema에서나 테이블을 만들 수 있도록 허용한다.
CREATE ANY TRIGGER	Grantee가 어떤 Schema에서나 테이블과 연관된 Schema에서 데이터베이스 트리거를 만들 수 있도록 허용한다.
CREATE ANY VIEW	Grantee가 어떤 Schema에서나 VIEW를 만들 수 있도록 허용한다.
CREATE PROCEDURE	Grantee가 자체 Schema에서 내장 프로시저,함수,패키지를 만들 수 있도록 허용한다.

권한

시스템 권한	허가된 내용(Grantee:권한을 받은 사용자)
CREATE PROFILE	Grantee가 PROFILE을 만들 수 있도록 허용한다.
CREATE ROLE	Grantee가 ROLE을 만들 수 있도록 허용한다.
CREATE SYNONYM	Grantee가 자체 Schema에서 시너임을 만들 수 있도록 허용한다.
CREATE TABLE	Grantee가 자체 Schema에서 테이블을 만들 수 있도록 허용한다.
CREATE TRIGGER	Grantee가 자체 Schema에서 트리거를 만들 수 있도록 허용한다.
CREATE USER	Grantee가 사용자를 만들 수 있도록 허용한다.
CREATE VIEW	Grantee가 자체 Schema에서 VIEW을 만들 수 있도록 허용한다.
DELETE ANY TABLE	Grantee가 어떤 Schema에서 테이블의 자료를 삭제할 수 있도록 허용한다.
DROP ANY INDEX	Grantee가 어떤 Schema에서나 인덱스를 삭제할 수 있다.
DROP ANY PROCEDURE	Grantee가 어떤 Schema에서나 내장 프로시저,함수,패키지를 삭제할 수 있도록 허용한다.
DROP ANY ROLE	Grantee가 ROLE을 삭제하도록 허용한다.
DROP ANY SYNONYM	Grantee가 어떤 Schema에서나 시너임을 삭제할 수 있도록 허용한다 .
DROP ANY TABLE	Grantee가 어떤 Schema에서나 테이블을 식제할 수 있도록 허용한다 .

권한

시스템 권한	허가된 내용(Grantee:권한을 받은 사용자)
DROP ANY TRIGGER	Grantee가 어떤 Schema에서나 데이터베이스 트리거를 삭제할 수 있도록 허용한다.
DROP ANY VIEW	Grantee가 어떤 Schema에서나 VIEW를 삭제할 수 있도록 허용한다.
DROP USER	Grantee가 사용자를 삭제할 수 있도록 허용한다.
EXECUTE ANY PROCEDURE	Grantee가 어떤 Schema에서나 프로시저,함수,패키지를 실행할 수 있도록 허용한다.
TRANSACTION	Local Database에서 자체의 불안정한 분산 Transaction의 BACK을 허용한다.
GRANT ANY PRIVILEGE	Grantee가 시스템 권한을 주는 것을 허용한다.
GRANT ANY ROLE	Grantee가 데이터베이스에서 어떠한 ROLE이라도 GRANT할 수 있는 권한을 허용한다.
INSERT ANY TABLE	Grantee가 어떠한 Schema에서나 테이블과 VIEW에 자룔를 삽입할 수 있도록 허용한다.
LOCK ANY TABLE	Grantee가 어떤 Schema에서나 테이블과 VIEW에 LOCK을 걸도록 허용한다.
SELECT ANY SEQUENCE	Grantee가 어떤 Schema에서나 시퀀스를 참조할 수 있도록 허용한다.
SELECT ANY TABLE	Grantee가 어떤 Schema에서나 테이블,VIEW, Snapshot을 참조할 수 있도록 허용한다.
UPDATE ANY	Grantee가 테이블에서 행을 수정하도록 허용한다.

권한 설정

```
GRANT {system_priv|role]
      [,{system_priv|role} ]....
      TO  {user|role|PUBLIC}
       [, {user|role|PUBLIC}]...
[WITH ADMIN OPTION]
```

- PUBLIC : 모든 사용자에게 해당 시스템 권한 부여
- WITH ADMIN OPTION : 부여받은 시스템 권한을 다른 사용자나 롤에게 재부여 가능

시스템(system) 특권(privileges)

- 사용자(user)에게 시스템 특권(privilege) 부여(grant)
  - With admin option 을 사용하면 부여받은 권한에 대한 부여 권한도 같이 받을 수 있다.

```
SQL> grant create session, create user, create table
  2        to steve;
```

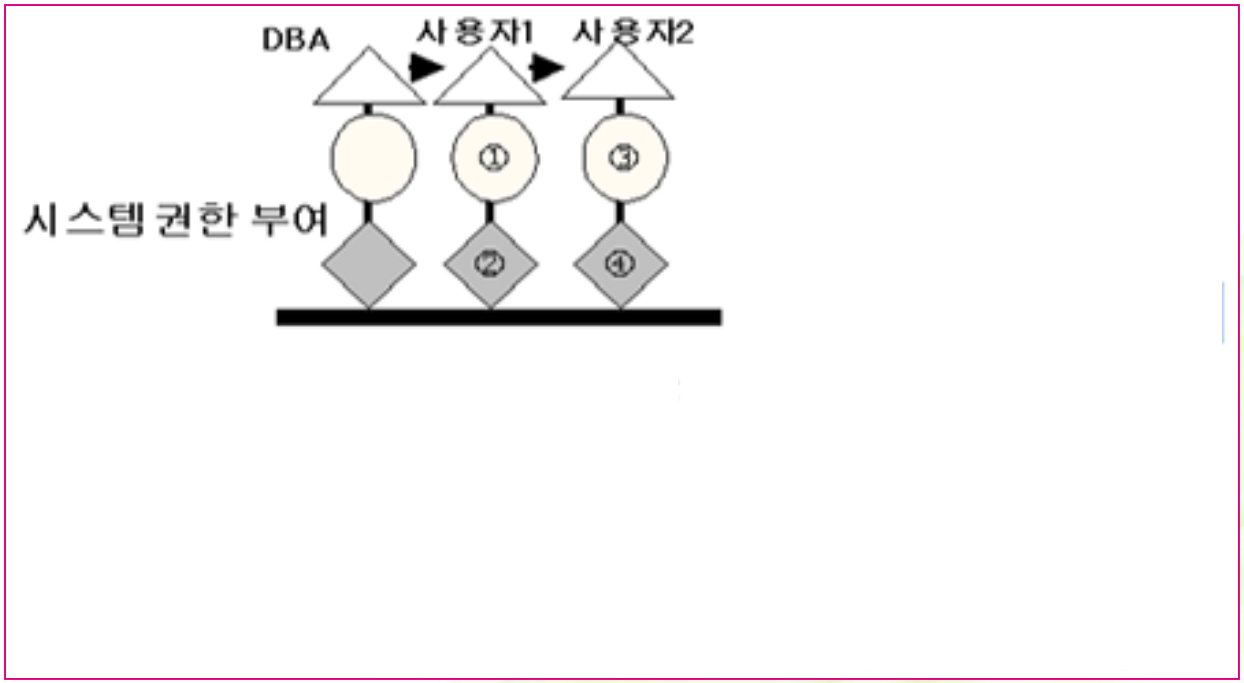
```
SQL> grant execute any procedure to steve
           with admin option;
```

```
SQL> connect steve/button
SQL> grant execute any procedure to gail;
```

```
SQL> connect system/manager
SQL> grant execute any procedure to public;
```

WITH ADMIN OPTION

- 부여 받은 시스템 권한을 다른 사용자나 롤에게 재부여 할 수 있는 기능 (dba_sys_privs에서 조회 가능)



시스템(system) 특권(privileges)

- 사용자(user)에게 부여된 시스템 특권(privilege) 점검(check)

```
SQL> connect steve/button
SQL> select *
  2  from user_sys_privs;
```

```
USERNAME     PRIVILEGE                                ADM
------------ ---------------------------------------- ---
PUBLIC       EXECUTE ANY PROCEDURE                     NO
STEVE        CREATE SESSION                            NO
STEVE        CREATE TABLE                              NO
STEVE        CREATE USER                               NO
STEVE        EXECUTE ANY PROCEDURE                    YES
```

시스템(system) 특권(privileges)

- 시스템 특권(privilege)의 사용(use)
  - 부여된 시스템 특권(privilege)에 대하여 지정된 작업만 시행(perform)할 수 있다.
  - 따라서 사용하고자 하는 시스템 특권(privilege)은 모두 부여(grant)하여야 한다.

```
SQL> connect steve/button
SQL> create user roy identified by williams;
```

```
SQL> drop user roy;
```

```
Ora-01031 : insufficient privileges
```

객체(object) 특권(privileges)

- 객체(object) 특권(privileges)
  - 사용자에게 데이터베이스 객체에 어떠한 행위(테이블에 DML문의 실행 같은)를 시행하도록 허락하는 것이다.

객체 특권	
SELECT	SELECT 시행
INSERT	INSERT 시행
UPDATE	UPDATE 시행
DELETE	DELETE 시행
EXECUTE	저장 프로시저에 대한 시행

권한

객체 권한	TABLE	VIEW	SEQUENCE	PROCEDURE	SNAPSHOT
ALTER	♣		♣		
DELETE	♣	♣			
EXECUTE				♣	
INDEX	♣				
INSERT	♣	♣			
REFERENCES	♣				
SELECT	♣	♣	♣		♣
UPDATE	♣	♣			

권한

OBJECT 권한	허가된 내용(Grantee:권한을 받은 사용자)
ALTER	Grantee가 OBJECT에 대해 ALTER할 수 있도록 허용한다.
AUDIT	Grantee가 OBJECT에 대해 감사할 수 있도록 허용한다.
COMMENT	Grantee가 OBJECT에 대해 COMMENT할 수 있도록 허용한다.
DELETE	Grantee가 OBJECT에 대해 자료를 삭제할 수 있도록 허용한다.
GRANT	Grantee가 OBJECT에 대해 GRANT할 수 있도록 허용한다.
INDEX	Grantee가 OBJECT에 대해 인덱스를 생성할 수 있도록 허용한다.
INSERT	Grantee가 OBJECT에 대해 자료를 삽입할 수 있도록 허용한다.
LOCK	Grantee가 OBJECT에 대해 Locking할 수 있도록 허용한다.
RENAME	Grantee가 OBJECT에 대해 이름을 변경할 수 있도록 허용한다.
SELECT	Grantee가 OBJECT에 대해 자료를 조회할 수 있도록 허용한다.
UPDATE	Grantee가 OBJECT에 대해 자료를 갱신할 수 있도록 허용한다.
REFERENCES	Grantee가 OBJECT에 대해 자료를 참조할 수 있도록 허용한다.
EXECUTE	Grantee가 프로시저,함수,패키지에 대해 실행할 수 있도록 허용한다.

객체 권한 부여

- 사용자와 롤에게 객체 권한을 부여하는 가능
- GRANT 명령문 사용

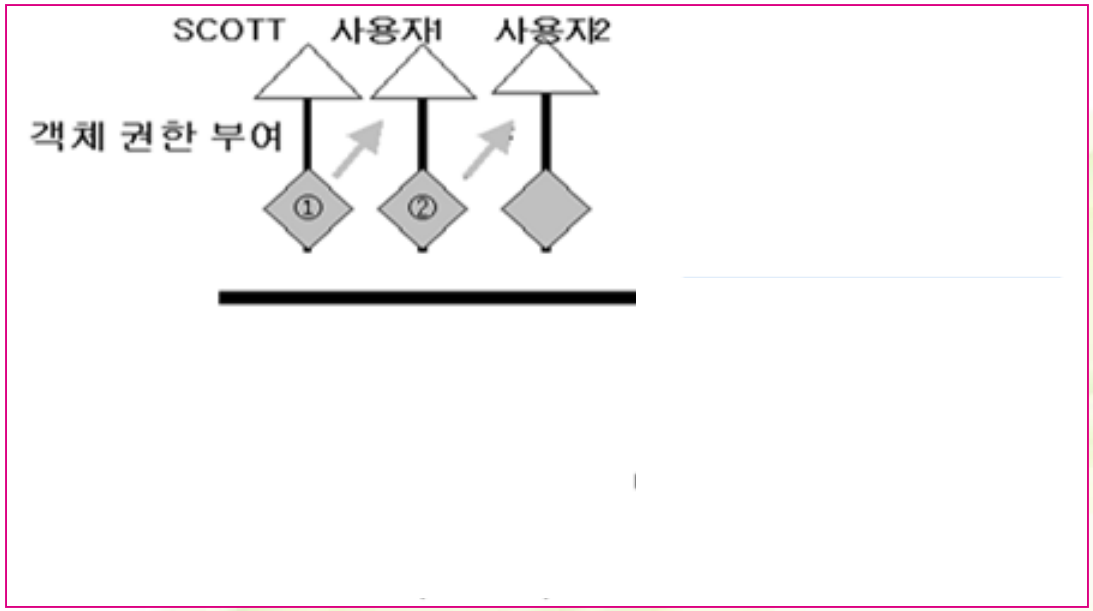
```
GRANT {object_priv [(column_list)]
[,object_priv[(column_list)] ]....
| ALL [PRIVILEGES]}
ON   [schema.]object
TO   {user|role|PUBLIC}
[, {user|role|PUBLIC}]...
[WITH GRANT OPTION]
```

- ALL : 모든 객체 권한을 사용자에게 부여
- PUBLIC : 모든 사용자에게 해당 객체 권한 부여
- WITH GRANT OPTION : 사용자의 객체 권한을 다른 사용자나 롤에게 재부여하는 기능

WITH GRANT OPTION

- WITH GRANT OPTION : 부여받은 객체 권한을 다른 사용자나 롤에게 재부여하는 기능

- WITH GRANT OPTION으로 부여된 객체 권한을 철회하면 다른 사용자나 롤에 게 재부여한 객체 권한도 연쇄적으로 철회



객체(object) 특권(privileges)

- 사용자(user)에게 객체 특권(privilege) 부여(grant)
  - With grant option 을 사용하면 부여 받은 권한에 대한 부여 권한도 같이 받을 수 있다.

```
SQL> connect scott/tiger
SQL> grant select, insert, delete on scott.emp
  2        to steve;
SQL> grant select on scott.salgrade to steve;
```

```
SQL> grant update (ename, sal)
  2        on scott.emp to steve;
```

```
SQL> grant select on scott.dept
           to steve with grant option;
```

```
SQL> connect steve/button
SQL> grant select on scott.dept to gail;
```

객체(object) 특권(privileges)

- 만들어진 개체 특권(privilege) 점검(check)
  - 테이블 단위로 만들어진 사용자들의 특권을 보여준다.

```
SQL> connect store/store
SQL> select *
  2  from user_tab_privs_made;
```

```
GRANTEE                        TABLE_NAME
------------------------------ ------------------------------
GRANTOR                        PRIVILEGE                 GRA HIE
------------------------------ ------------------------- --- ---
PRODUCT_MANAGER                PRODUCTS
STORE                          DELETE                    NO  NO

PRODUCT_MANAGER                PRODUCTS
STORE                          INSERT                    NO  NO
                               ...
```

객체(object) 특권(privileges)

- 컬럼 단위로 만들어진 사용자들의 특권을 보여준다.

```
SQL> select *
  2  from user_col_privs_made;
```

```
GRANTEE                        TABLE_NAME
------------------------------ ------------------------------
COLUMN_NAME                    GRANTOR
------------------------------ ------------------------------
PRIVILEGE                                                GRA
-------------------------------------------------------- ---
STEVE                          EMPLOYEES
LAST_NAME                      STORE
UPDATE                                                   NO

STEVE                          EMPLOYEES
SALARY                         STORE
UPDATE                                                   NO
                               ...
```

객체(object) 특권(privileges)

- 부여 받은 객체 특권(privilege) 점검(check)
  - 테이블(table) 단위로 받은(received) 객체 특권(privilege)을 보여준다.

```
SQL> connect steve/button
SQL> select *
  2  from user_tab_privs_recd;
```

```
OWNER                          TABLE_NAME
------------------------------ ------------------------------
GRANTOR                        PRIVILEGE                 GRA HIE
------------------------------ ------------------------- --- ---
STORE                          CUSTOMERS
STORE                          SELECT                    YES NO

STORE                          PRODUCTS
STORE                          INSERT                    NO  NO
                               ...
```

객체(object) 특권(privileges)

- 컬럼(column) 단위로 받은(received) 객체 특권(privilege)을 보여준다.

```
SQL> connect steve/button
SQL> select *
  2  from user_col_privs_recd;
```

```
OWNER                          TABLE_NAME
------------------------------ ------------------------------
COLUMN_NAME                    GRANTOR
------------------------------ ------------------------------
PRIVILEGE                                                GRA
-------------------------------------------------------- ---
STORE                          EMPLOYEES
LAST_NAME                      STORE
UPDATE                                                   NO

STORE                          EMPLOYEES
SALARY                         STORE
UPDATE                                                   NO
                               ...
```

객체(object) 특권(privileges)

- 객체 특권(privilege)의 사용(use)
  - 부여된 객체 특권(privilege)에 대하여 지정된 작업만 시행(perform)할 수 있다.
  - 따라서 사용하고자 하는 시스템 특권(privilege)은 모두 부여(grant)하여야 한다.

```
SQL> connect steve/button
SQL> select *
  2  from store.customers;
```

```
SQL> select *
  2  from store.purchases;
```

```
Ora-00942 : table or view does not exist
```

객체 권한 철회

- 사용자와 롤에게 부여된 객체 권한을 취소하는 기능
- REVOKE 명령문 사용

```
REVOKE {object_priv
[,object_priv]....
| ALL [PRIVILEGES]}
ON     [schema.]object
FROM   {user|role|PUBLIC}
[, {user|role|PUBLIC}]...
[CASCADE CONSTRAINTS]
```

- CASCADE CONSTRAINTS : REFERENCES나 ALL 권한 철회시 관련 참조 무결성 제약 조건도 함께 삭제

시스템(system) 특권(privileges)

- 사용자(user)의 시스템 특권(privilege) 폐지(revoke)
  - Revoke 명령을 사용하여 부여된 시스템 특권(privilege)을 폐지(revoke) 할 수 있다.

```
SQL> connect system/system_password
SQL> revoke create table from steve;
```

```
SQL> revoke execute any procedure from steve;
```

```
SQL> connect gail/seymour
SQL> select *
  2  from user_sys_privs;
```

```
USERNAME     PRIVILEGE                                ADM
-----------  ---------------------------------------- ---
PUBLIC       EXECUTE ANY PROCEDURE                     NO
GAIL         CREATE SESSION                            NO
GAIL         EXECUTE ANY PROCEDURE                     NO
```

Example

The user Sue issues this SQL statement:

```
GRANT SELECT ON sue.EMP
TO alice
WITH GRANT OPTION;
```

The user Alice issues this SQL statement:

```
GRANT SELECT ON sue.EMP
TO reena
WITH GRANT OPTION;
```

The user Reena issues this SQL statement:

```
GRANT SELECT ON sue.EMP
TO timber;
```

The user Sue issues this SQL statement:

```
REVOKE select on sue.EMP
FROM alice;
```

- Result ?

객체(object) 특권(privileges)

- 동의어(synonym)
  - Create synonym 문장을 사용하여 다른 스키마(schema)에 속한 명칭을 간략하게 만들 수 있다.

```
SQL> connect system/system_password
SQL> grant create synonym to steve;
```

```
SQL> connect steve/button
SQL> create synonym semp for scott.emp;
```

```
SQL> select *
  2  from semp;
```

```
CUST_ID FIRST_NAME LAST_NAME DOB       PHONE
------- ---------- --------- --------- ------------
      1 JOHN       BROWN     01-JAN-65 800-555-1111
      2 CYNTHIA    GREEN     05-FEB-77 800-555-1222
                        ...
```

객체(object) 특권(privileges)

- 일반 동의어(public synonym)를 사용하면 다른 사용자(other user)가 사용 가능한 동의어(synonym)를 만들 수 있다.
- 적절한 객체 권한(privilege)을 가진 사용자(user)만 접근 가능하다.

```
SQL> connect system/system_password
SQL> grant create public synonym to store;
```

```
SQL> connect store/button
SQL> create public synonym products
            for store.products;
```

```
SQL> connect steve/button
SQL> select *
  2  from products;
```

객체(object) 특권(privileges)

```
SQL> connect gail/seymour
SQL> select *
  2  from products;
```

```
Ora-00942 : table or view does not exist
```

- Steve 사용자는 store.products에 대한 select 권한이 있었기 때문에 store 사용자가 생성한 일반 동의어(public synonym)를 사용할 수 있다.
- Gail 사용자는 store.products에 대한 select 권한이 없기 때문에 store 사용자가 생성한 일반 동의어(public synonym)를 사용할 수 없다.

```
SQL> select *
  2  from user_synonym;
```

롤(role)의 개념

- 롤이란 다중 사용자 시스템에서 사용자에 대한 권한 관리를 효율적으로 하기위해 관련된 권한을 그룹화한 개념
- 롤은 데이타베이스 응용 프로그램에 대한 권한 관리 또는 사용자 그룹에 대한 효율적인 권한 관리 목적으로 사용
- 오라클 서버에서 사전에 정의된 롤과 사용자가 정의한 롤로 구분

롤(role)의 특징

- 롤은 사용자 또는 다른 롤에 부여 또는 철회 가능
- 다른 롤에 롤을 부여하는 것은 가능하지만, 자신에 대한 롤 부여나 순환적인 부여는 불가능. 즉, A롤에 B롤을 부여했다면 B롤에 A롤을 재부여하는 것은 불가능
- 롤은 특정 소유자나 특정 객체에 속하지 않음
- 롤명은 사용자명이나 다른 롤명과는 구분되는 유일한 이름 사용

롤(role)의 생성

```
CREATE ROLE role [NOT IDENTIFIED | IDENTIFIED
{BY password | EXTERNALLY }]
```

- NOT IDENTIFIED : 롤 활성화시 암호에 의한 검증 과정 생략
- IDENTIFIED : 롤 활성화시 암호에 의한 검증 과정 필요
- BY password : 롤 활성화시 사용될 암호 지정
- EXTERNALLY : 롤 활성화시 운영체제 인증을 통한 롤에 대한 사용자 검증

롤(role)의 권한 부여

- 롤에 시스템 권한이나 객체 권한 또는 다른 롤을 부여 가능
- GRANT 명령문을 사용

```
GRANT role [, role ] ...
TO {user|role|PUBLIC}
[, {user|role|PUBLIC} ]...
[WITH ADMIN OPTION]
```

- PUBLIC : 모든 사용자에게 해당 롤 부여
- WITH ADMIN OPTION : 다른 사용자나 롤에 해당 롤을 재부여할 수 있는 옵션

롤(role)의 부여

- 롤을 다른 롤이나 사용자에게 부여 가능
- 롤을 부여받은 다른 롤이나 사용자는 부여받은 롤이 가지는 모든 권한 사용 가능
- 다른 롤이나 사용자에게 롤을 부여할 때 WITH ADMIN OPTION 지정 가능
- WITH ADMIN OPTION을 포함하여 부여 받은 롤은 다른 롤이나 사용자에 재부여 가능
- 하나의 GRANT 명령문에서 시스템 권한이나 롤을 객체 권한과 함께 부여 불가능

역할(roles)

- 역할(roles)
  - 권한의 집합이다.
  - 직접적으로 권한을 부여하기 보다는 역할을 생성한 후 역할을 부여하는 것이 관리에 쉽다.
  - 역할에 권한을 추가하거나 삭제 함으로써 부여받은 모든 사용자에게 영향을 줄 수 있다.
  - 사용자나 역할에 여러가지 역할을 할당할 수 있다.
  - 역할에 암호를 설정 할 수 있다.

역할(roles) 종류

오라클을 설치하면 기본적으로 3가지의 롤(CONNECT, RESOURCE, DBA)을 제공한다.

먼저 CONNECT 롤은 사용자가 데이터베이스에 접속 가능하도록 하기 위한 아래와 같은 기본적인 시스템 권한을 제공한다.

ALTER SESSION, CREATE CLUSTER,
CREATE DATABASE LINK, CREATE SEQUENCE,
CREATE SESSION, CREATE SYNONYM,
CREATE TABLE, CREATE VIEW

RESOURCE 롤은 사용자가 객체(테이블, 뷰, 인덱스)를 생성하기 위한 시스템 권한을 제공한다.

CREATE CLUSTER, CREATE PROCEDURE,
CREATE SEQUENCE, CREATE TABLE,
CREATE TRIGGER

DBA 롤은 시스템 자원을 무제한적으로 사용하며 시스템 관리에 필요한 모든 권한을 보유한 강력한 롤이다. 따라서 일반 계정에게 부여하는 것은 위험하다.

역할(roles)

- 역할(role) 생성(create)

```
SQL> connect system/system_password
SQL> grant create role to store
SQL> grant create user to store with admin option;
```

```
SQL> connect store/store_password
SQL> create role product_manager;
SQL> create role hr_manager;
SQL> create role overall_manager
            identified by manager_password;
```

역할(roles)

- 역할(role)에 특권(privilege) 부여(grant)

```
SQL> grant select, insert, update, delete
           on dept to product_manager;

SQL> grant select, insert, update, delete
           on salgrade to hr_manager;

SQL> grant select, insert, update, delete
           on emp to hr_manager;

SQL> grant create user to hr_manager;

SQL> grant product_manager, hr_manager
           to overall_manager;
```

역할(roles)

- 사용자(user)에게 역할(privilege) 부여(grant)

```
SQL> grant overall_manager to steve;
```

- 사용자(user)에게 부여된 역할(privilege) 점검(check)

```
SQL> conn steve/button;
SQL> select *
  2  from user_role_privs;
```

```
USERNAME                       GRANTED_ROLE                   ADM DEF OS_
------------------------------ ------------------------------ --- --- ---
STEVE                          OVERALL_MANAGER                YES YES NO
```

역할(roles)

```
SQL> conn store/store_password;
SQL> select *
  2  from user_role_privs;
```

```
USERNAME                       GRANTED_ROLE                   ADM DEF OS_
------------------------------ ------------------------------ --- --- ---
STORE                          OVERALL_MANAGER                YES YES NO
STORE                          CONNECT                        NO  YES NO
STORE                          RESOURCE                       NO  YES NO
STORE                          HR_MANAGER                     YES YES NO
STORE                          PRODUCT_MANAGER                YES YES NO
```

역할(roles)

- 역할(role)에 부여된 시스템 특권 점검

```
SQL> conn store/store_password;
SQL> select *
  2  from role_sys_privs;
```

```
ROLE                 PRIVILEGE                      ADM
-------------------- ------------------------------ ---
CONNECT              ALTER SESSION                   NO
CONNECT              CREATE CLUSTER                  NO
CONNECT              CREATE DATABASE LINK            NO
CONNECT              CREATE SEQUENCE                 NO
                          ...
```

역할(roles)

- 역할(role)에 부여된 객체 특권 점검

```
SQL> conn store/store_password;
SQL> select *
  2  from role_tab_privs;
```

```
ROLE                 OWNER
-------------------- ------------------------------
TABLE_NAME           COLUMN_NAME
-------------------- ------------------------------
PRIVILEGE            GRA
-------------------- ------------------------------
HR_MANAGER           STORE
EMPLOYEES
DELETE               NO
HR_MANAGER           STORE
EMPLOYEES
INSERT               NO
...
```

역할(roles)

- 역할(role)에 부여된 특권 사용(use)

```
SQL> conn steve/button;
SQL> select p.name, pt.name
  2  from store.products p, store.product_types pt
  3  where p.product_type_id = pt.product_type_id;
```

```
NAME                 NAME
-------------------- ------------------------------
Modern science       book
Chemistry            book
Supernova            video
Tank war             video
                          ...
```

역할(roles)

- 기본 역할(role)
  - 역할은 기본적으로 부여시 자동 사용이다.
  - 비활성을 할 수 있다.

```
SQL> conn system/system_password;
SQL> alter user steve default role all
           except overall_manager;
```

```
SQL> conn steve/button;
SQL> set role overall_manager
              identified by manager_password;
```

```
SQL> set role none;
```

```
SQL> set role all except overall_manager;
```

역할(roles)

- 역할(role) 폐지

```
SQL> conn store/store_password;
SQL> revoke overall_manager from steve;
```

- 역할(role)에서 특권(privilege) 폐지

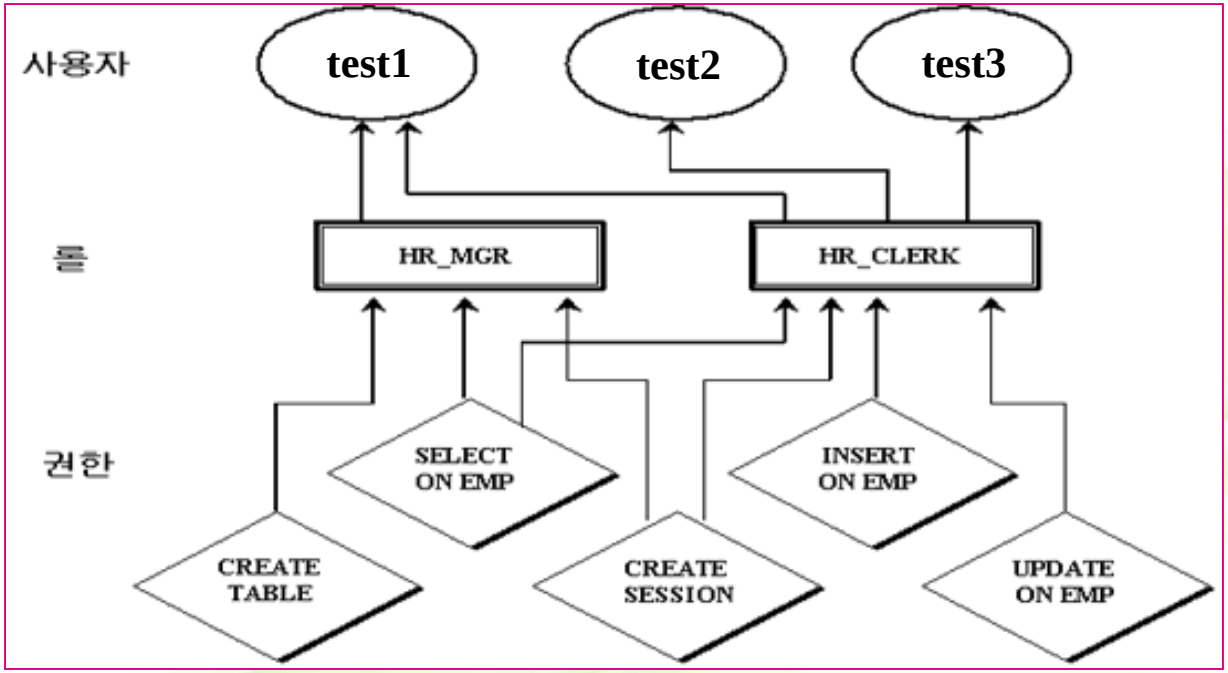
```
SQL> revoke all on products from product_manager;
SQL> revoke all on product_types from product_manager;
```

- 역할(role) 삭제(drop)

```
SQL> drop role overall_manager;
SQL> drop role product_manager;
SQL> drop role hr_manager;
```

실습) role의 사용

- 롤 자체에 대한 보안을 위해 암호 부여 가능
- 롤 생성 명령문으로 Role을 생성한 후, 특정 권한을 부여
- 사용자 생성시 해당 테이블스페이스에 Quota 지정해야 테이블 생성 가능
  - **사용자 생성시 설정 : Quota unlimited on system**



복습

1. 계정이 KIM, 암호가 LION인 사용자 계정을 작성하시오.

2. KIM에게 CREATE TABLE과 CREATE SESSION 권한을 부여하시오.

3. KIM에게 SCOTT의 DEPT, EMP 테이블의 SELECT 권한을 부여하시오.

4. KIM에게 SCOTT의 EMP 테이블에 SAL, COMM 컬럼을 UPDATE 할 수 있는 권한을 부여하시오.

5. KIM에게 부여된 EMP 테이블의 UPDATE 권한을 회수하시오.

연습문제

1. 사용자는 KSH이고 패스워드는 KIM인 사용자를 생성하여라.
2. 생성된 사용자에게 CONNECT와 RESOURCE권한을 부여하여라.
3. 부여한 권한을 취소하고 KSH사용자를 삭제하여라.

백업과 복원

- 백업(Export) :데이터와 구조를 바이너리 파일로 저장

>exp userid=아이디/비밀번호@전역데이터베이스명 file=저장경로

system계정으로 전체 백업

>exp userid=system/비밀번호@전역데이터베이스명 full=y file=c:\dump.dmp

system 계정으로 scott 계정에 있는 DB백업

>exp userid=system/비밀번호@전역데이터베이스명 owner=scott file=c:\dump.dmp

scott계정으로 자신의 모든 데이터 백업

dos>exp userid=scott/비밀번호@전역데이터베이스명 file=c:\dump.dmp

scott계정으로 emp테이블만 백업

dos>exp userid=scott/비밀번호@전역데이터베이스명 file=c:\dump.dmp tables=emp

여러개 테이블을 동시에 받으려면 tables=(테이블1,테이블2,...)

백업 파일의 확장자는 보통 .dmp 혹은 .dat .bak으로 한다 한다.

백업과 복원

- 복원(Import)

imp 아이디/비밀번호@전역데이터베이스명 file=백업경로

system계정으로 전체 복원

>imp system/비밀번호@전역데이터베이스명 file=c:\dump.dmp

system 계정으로 scott 계정에 있는 DB복원

>imp system/비밀번호@전역데이터베이스명 fromuser=scott touser=scott file=c:\dump.dmp

scott계정으로 자신의 모든 데이터 백업

>imp scott/비밀번호@전역데이터베이스명 file=c:\dump.dmp

복원하고자하는 DB에 같은 이름의 Object가 있을때,오류를 무시하고 건너 띄고 싶을때 ignore 옵션사용dos>imp 아이디/비밀번호@전역데이터베이스명 file=c:\dump.dmp ignore=y

- 5. system계정으로 들어가 scott에서 Export한 데이터를 scott2에게 Import
- dos>imp system/비밀번호@전역데이터베이스명 fromuser=scott touser=scott2 file=c:\dump.dmp

백업과 복원

Table백업 및 복구

exp 유저/암호 file=a.dmp tables=emp,dept,..

imp 유저/암호 file=a.dmp tables=emp,dept,..

XP 방화벽 예외처리

window제어판 - 보안센터 윈도우 방화벽 예외 탭 클릭

프로그램추가 위치 : C:\oracle\product\10.2.0\db_1\BIN oracle.exe

TNSLSNR.EXE 포트

```
SQL> DROP TABLE EMP;

테이블이 삭제되었습니다.

SQL> FLASHBACK TABLE EMP TO BEFORE DROP;

플래시백이 완료되었습니다.

SQL> SELECT COUNT(*) FROM EMP;

  COUNT(*)
----------
        14

SQL> CREATE TABLE EMP2
  2  AS SELECT * FROM EMP;

테이블이 생성되었습니다.

SQL> DROP TABLE EMP2 PURGE;

테이블이 삭제되었습니다.

SQL> FLASHBACK TABLE EMP2 TO BEFORE DROP;
```

각 사용자는 자유롭게 휴지통을 검색 할 수 있지만, SYSDBA 권한을 부여 받지 않는 한, 자신이 소유한 객체에 대해서만 접근 할 수 있다. 사용자는 다음과 같은 문장으로 휴지통에서 자신의 객체를 검색 할 수 있다.

```
SELECT * FROM RECYCLEBIN;
```

사용자가 삭제되는 경우에는 해당 사용자가 소유하고 있는 모든 객체가 휴지통으로 이동하지 않으며, 휴지통에 있던 어떤 객체도 삭제되지 않는다. 만약, 휴지통에 저장된 객체를 영구히 삭제하고자 하는 경우에 다음과 같은 명령을 실행하면 된다.

```
PURGE RECYCLEBIN;
```