사용자 권한 제어

배재대학교 컴퓨터공학과
김 창 수

Contents

데이터베이스 보안

❖ 다중 사용자 환경(multi-user environment)

- 불법적인 접근 및 유출 방지를 위해 보안 대책 필요
- 오라클에서는 사용자는 자신이 생성한 객체에 대해 소유권을 가지고 데이터에 대한 조작이나 조회 가능
- 다른 사용자가 소유한 객체는 소유자로부터 접근 권한을 부여받지 않는 한 접근 불가
- 다중 사용자 환경에서는 데이터베이스 관리자의 암호를 철저하게 관리

❖ 중앙 집중적인 데이터 관리

- 분산적으로 관리되는 기존의 파일 시스템보다 보안이 취약할 수 있으므로 철저한 보안 대책 필요

오라클에서 지원하는 데이터베이스 보안 정책

❖ 시스템 보안

- 시스템 관리 차원에서 데이터베이스 자체에 대한 접근 권한을 관리
- 데이터베이스 관리자는 사용자 계정, 암호 관리, 사용자별 허용 가능한 디스크공간 할당
- 예
  - 사용자의 접근 패턴을 감시하여 의심스러운 사용자 시스템 접속을 차단
  - 일정 주기로 암호를 변경하지 않는 사용자의 접속을 차단

❖ 데이터 보안

- 사용자별로 객체를 조작하기 위한 동작 관리
- 데이터베이스 객체에 대한 접근 권한을 관리
- 예
  - scott에게 테이블, 뷰, 인덱스와 같은 객체를 생성할 수 있는 권한을 부여
  - scott가 만든 테이블을 다른 사용자가 접근할 수 없도록 접근 권한 제한

권한(Privilege)

❖ 권한이란?

- 권한은 사용자가 데이터베이스 시스템을 관리하거나 객체를 이용할 수 있는 권리

❖ 권한의 종류

- 시스템 권한
  - 시스템 차원의 자원 관리나 사용자 스키마 객체 관리 등과 같은 데이터베이스 관리 작업을 할 수 있는 권한
  - 오라클에서는 테이블, 뷰, 롤백 세그먼트, 프로시져와 같은 객체를 생성, 삭제, 수정하는 작업과 관련된 140여 종류의 시스템 권한을 지원
- 객체 권한
  - 테이블, 뷰, 시퀀스, 함수 등과 같은 객체를 조작할 수 있는 권한
  - 객체의 종류에 따라 서로 다른 유형의 객체 권한 지원
  - 테이블의 칼럼별로 insert, update, references등의 권한을 상세 부여 가능

시스템 권한

❖ 시스템 권한

- 데이터베이스 관리자나 일반 사용자에 의해 데이터베이스를 관리하는 권한

❖ 데이타베이스 관리자가 가지는 시스템 권한

- 사용자 생성, 삭제, 사용자 계정에서 객체의 생성 또는 수정, 데이터베이스 백업 관리

시스템 권한	기능
CREATE USER	사용자를 생성할 수 있는 권한
DROP USER	사용자를 삭제할 수 있는 권한
DROP ANY TABLE	임의의 테이블을 삭제할 수 있는 권한
QUERY REWRITE	함수 기반 인덱스를 생성하기 위한 권한
BACKUP ANY TABLE	익스포트 유틸리티를 사용해서 임의의 테이블을 백업할 수 있는 권한

일반 사용자

❖ 일반 사용자가 가지는 시스템 권한

- 사용자가 생성한 객체를 관리, 내장 프로시저를 관리

시스템 권한	기능
CREATE SESSION	데이터베이스에 접속할 수 있는 권한
CREATE TABLE	사용자 스키마에서 테이블을 생성할 수 있는 권한
CREATE SEQUENCE	사용자 스키마에서 시퀀스를 생성할 수 있는 권한
CREATE VIEW	사용자 스키마에서 뷰를 생성할 수 있는 권한
CREATE PROCEDURE	사용자 스키마에서 프로시저, 함수, 패키지를 생성할 수 있는 권한

시스템 권한 부여

❖ 시스템 권한 부여

- 시스템 권한은 특정 사용자나 모든 사용자에게 부여 가능
- 롤에도 시스템 권한 부여 가능
- 데이터베이스 관리자가 GRANT 명령문을 사용하여 일반 사용자에게 시스템 권한을 부여 가능

사용법

```
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 : 해당 시스템 권한을 다른 사용자나 롤에 재부여 허용

시스템 권한 부여 사용 예

사용예

query rewrite 시스템 권한을 모든 사용자에게 부여하여라. quey rewrite 권한은 함수 기반 인덱스를 생성하기 위해 필요한 권한이다.

```
SQL> CONNECT system/manager

SQL> GRANT query rewrite TO PUBLIC;
권한이 부여되었습니다.

SQL> CONNECT scott/tiger

SQL> SELECT * FROM user_sys_privs;
```

모든 사용자에게 query rewrite 권한을 부여한다.

사용자와 롤에 부여된 시스템 권한을 조회한다.

☞ 출력 결과

USERNAME	PRIVILEGE	ADMIN_
PUBLIC	**QUERY REWRITE**	**NO**
SCOTT	**QUERY REWRITE**	**NO**
SCOTT	UNLIMITED TABLESPACE	NO

현재 세션에 부여된 시스템 권한 조회

❖ SESSION_PRIVS

- 현재 세션에서 사용자와 롤에 부여된 시스템 권한을 조회 가능

```
SQL> SELECT * FROM session_privs;
```

현재 세션에서 사용자와 롤에 부여된 시스템 권한을 조회한다.

☞ 출력 결과

```
PRIVILEGE
----------------------------------------
CREATE SESSION
ALTER SESSION
UNLIMITED TABLESPACE
CREATE TABLE
...
QUERY REWRITE
15 개의 행이 선택되었습니다.
```

시스템 권한 철회

❖ 시스템 권한 철회

- 데이터베이스 관리자나 권한을 부여한 사용자는 다른 사용자에게 부여한 시스템 권한을 철회 가능
- REVOKE 명령문 사용하여 시스템 권한 철회

사용법

```
REVOKE { system_priv|role}
      [ ,{ system_priv|role} ] ...
  FROM  { user|role|PUBLIC}
      [ ,{ user|role|PUBLIC} ] ...
```

사용예

scott 사용자에게 부여한 query rewrite 시스템 권한을 철회하여라.

```
SQL> CONNECT system/manager
SVRMGR> REVOKE query rewrite FROM scott;
권한이 취소되었습니다.
```

객체 권한

❖ 객체 권한

- 객체 권한은 테이블, 뷰 , 시퀀스, 함수 등과 같이 객체별로 사용할 수 있는 권한

객체 권한 종류	테이블	뷰	시퀀스	프로시저[1]
ALTER	○		○	
DELETE	○	○		
EXECUTE				○
INDEX	○[2]			
INSERT	○	○		
REFERENCES	○[2]			
SELECT	○	○[3]	○	
UPDATE	○	○		

1. 독립형 내장 프로시저와 함수, 공용 패키지 구성자 포함
2. 롤에는 해당 권한의 부여 불가능
3. 스냅샷에는 해당 권한의 부여 불가능

객체 권한 부여

❖ 객체 권한 부여

- 데이터베이스 관리자나 객체 소유자가 사용자와 롤에게 객체 권한 부여 가능
- GRANT 명령문을 사용해 권한 부여

사용법

```
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 해당 사용자에게 부여
- PUBLIC : 모든 사용자에게 해당 객체 권한 부여
- WITH GRANT OPTION : 해당 객체 권한을 다른 사용자나 롤에 재부여 허용

객체 권한 부여 사용 예1

사용예

tiger 사용자에게 scott 소유의 테이블을 SELECT할 수 있는 권한을 부여하여라.

```
SQL> CONNECT system/manager

SQL> CREATE USER tiger IDENTIFIED BY tiger123
  2  DEFAULT TABLESPACE users
  3  TEMPORARY TABLESPACE temp;
사용자가 생성되었습니다.

SQL> GRANT connect, resource TO tiger;
권한이 부여되었습니다.

SQL> CONNECT scott/tiger

SQL> GRANT SELECT ON scott.student TO tiger;
권한이 부여되었습니다.

SQL> CONNECT tiger/tiger123

SQL> SELECT * FROM scott.student;
```

데이터베이스 관리자에 의해 tiger 사용자를 생성한다.

tiger 사용자에게 데이터베이스에 대한 접속 객체 관리 및 롤 사용 권한을 부여한다.

scott가 tiger에게 scott 소유의 student 테이블을 조회할 수 있는 권한을 부여한다.

tiger 사용자가 student 테이블을 검색할 수 있다.

객체 권한 부여 사용 예2

사용예

tiger 사용자에게 scott가 소유한 student 테이블의 키와 몸무게를 수정할 수 있는 권한을 부여하여라.

```
SQL> CONNECT tiger/tiger123

SQL> UPDATE scott.student
  2  SET    height = 180, weight =80
  3  WHERE  deptno = 101;
UPDATE scott.student
            *
1행에 오류:
ORA-01031: 권한이 불충분합니다

SQL> CONNECT scott/tiger

SQL> GRANT UPDATE(height, weight) ON student TO tiger;
권한이 부여되었습니다.

SQL> CONNECT tiger/tiger123

SQL> UPDATE scott.student
  2  SET    height = 180, weight =80
  3  WHERE  deptno = 101;

8행이 갱신되었습니다.
```

tiger 사용자는 scott 소유의 student 테이블에 대한 수정 권한이 없다.

scott가 tiger에게 scott 소유의 student 테이블을 수정할 수 있는 권한을 부여한다.

tiger 사용자가 student 테이블을 수정할 수 있다.

객체 권한 조회

사용예

tiger 사용자에게 부여된 사용자 객체, 칼럼에 부여된 객체 권한을 조회하여라.

```
SQL> CONNECT tiger/tiger123

SQL> SELECT *
  2  FROM   user_tab_privs_made;
```

tiger 사용자가 다른 사용자에게 부여한 객체 권한을 조회한다.

☞ 출력 결과

선택된 레코드가 없습니다.

```
SQL> SELECT *
  2  FROM   user_tab_privs_recd;
```

tiger 사용자에게 부여된 객체권한, 객체 이름을 조회한다.

☞ 출력 결과

```
OWNER      TABLE_NAME GRANTOR    PRIVILEGE  GRANTA HIERAR
---------- ---------- ---------- ---------- ------ ------
SCOTT      STUDENT    SCOTT      SELECT     NO     NO
```

객체 권한 조회

```
SQL> SELECT *
  2  FROM   user_col_privs_made;
```

tiger 사용자가 다른 사용자에게 부여한 칼럼에 대한 객체 권한과 칼럼 이름을 조회한다.

☞ 출력 결과

선택된 레코드가 없습니다.

```
SQL> SELECT *
  2  FROM   user_col_privs_recd;
```

tiger 사용자에게 부여된 칼럼에 대한 객체 권한과 칼럼 이름을 조회한다.

☞ 출력 결과

OWNER	TABLE_NAME	COLUMN_NAME	GRANTOR	PRIVILEGE	GRANTA
SCOTT	STUDENT	HEIGHT	SCOTT	UPDATE	NO
SCOTT	STUDENT	WEIGHT	SCOTT	UPDATE	NO

객체 권한의 철회

❖ 객체 권한의 철회

- 사용자와 롤에 부여된 객체 권한은 데이타베이스 관리자나 객체 소유자가 철회 가능
- REVOKE 명령문을 사용하여 철회

사용법

```
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 제약조건도 함께 삭제

객체 권한의 철회 사용 예

사용예

scott에 의해 tiger에게 부여된 student 테이블에 대한 SELECT, UPDATE 권한을 철회하여라.

```
SQL> CONNECT scott/tiger

SQL> REVOKE UPDATE ON student FROM tiger;
권한이 취소되었습니다.

SQL> REVOKE SELECT ON student FROM tiger;
권한이 취소되었습니다.

SQL> CONNECT tiger/tiger123

SQL> SELECT * FROM scott.student;

☞ 출력 결과
SQL> SELECT * FROM scott.student;
SELECT * FROM scott.student
                    *
1행에 오류:
ORA-00942: 테이블 또는 뷰가 존재하지 않습니다.
```

tiger 사용자에게 부여된 student 테이블의 update 권한을 철회한다.

tiger 사용자에게 부여된 student 테이블의 select 권한을 철회한다.

tiger 사용자가 scott의 student 테이블을 조회하면 권한이 부족하므로 조회할 수 없다.

롤(role)

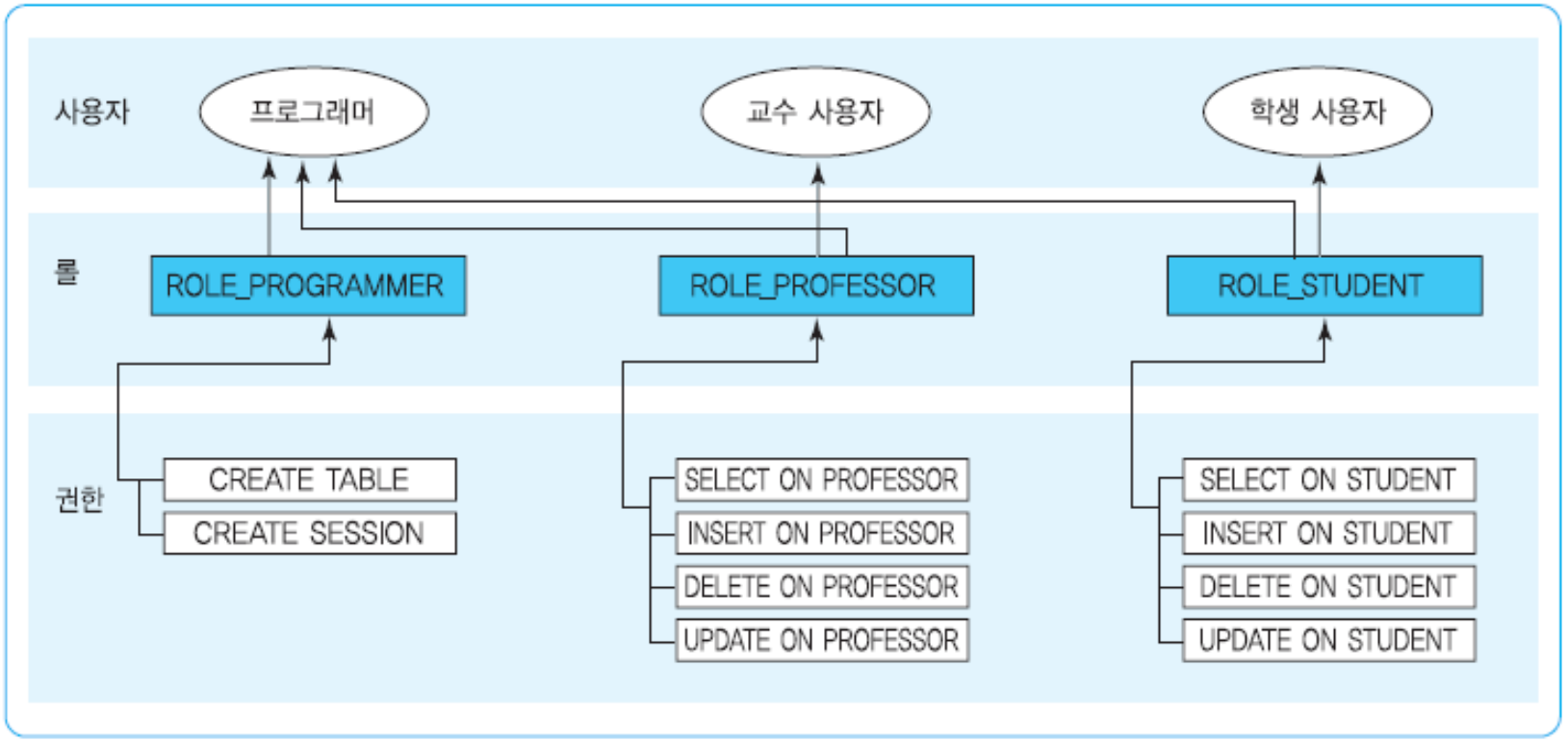
❖ 롤 개념

- 다수 사용자와 다양한 권한을 효과적으로 관리하기 위하여 서로 관련된 권한을 그룹화한 개념
- 일반 사용자가 데이터베이스를 이용하기 위한 공통적인 권한(데이터베이스 접속권한, 테이블 생성, 수정, 삭제, 조회 권한, 뷰 생성 권한)을 그룹화
- 이러한 권한을 그룹화하여 롤로 정의해서 사용자에게 롤에 대한 접근 권한을 부여하면 권한 관리를 효과적으로 함

❖ 롤 부여

- 활성화 또는 비활성화를 통한 일시적 권한 부여 철회 가능
- 암호 부여 가능
- 사용자 또는 다른 롤에 대한 접근 권한 부여 및 철회 가능
- 자신에 대한 롤 부여나 순환적인 부여는 불가능
- 롤은 특정 소유자나 특정 객체에 속하지 않음

사용자, 롤, 권한간의 관계


사용자
프로그래머
교수 사용자
학생 사용자
롤
ROLE_PROGRAMMER
ROLE_PROFESSOR
ROLE_STUDENT
권한
CREATE TABLE
CREATE SESSION
SELECT ON PROFESSOR
INSERT ON PROFESSOR
DELETE ON PROFESSOR
UPDATE ON PROFESSOR
SELECT ON STUDENT
INSERT ON STUDENT
DELETE ON STUDENT
UPDATE ON STUDENT

롤 종류

❖ 사전에 정의된 롤

- 사용자 접속, 자원 생성, DBA 등과 같이 공통적으로 사용되는 권한은 사전에 롤로 정의하여 제공됨
- 사전에 정의된 롤도 부여, 철회 가능

❖ 사용자 정의 롤

롤 종류	롤에 부여된 권한
CONNECT	ALTER SESSION, CREATE CLUSTER, CREATE DATABASE LINK, CREATE SEQUENCE, CREATE SESSION, CREATE SYNONYM, CREATE TABLE, CREATE VIEW
RESOURCE	CREATE CLUSTER, CREATE PROCEDURE, CREATE SEQUENCE, CREATE TABLE, CREATE TRIGGER
DBA	WITH ADMIN OPTION이 있는 모든 시스템 권한

사전에 정의된 롤

- **CONNECT 롤**
  - 사용자가 데이터베이스에 접속하여 세션을 생성할 수 있는 권한
  - 테이블 또는 뷰와 같은 객체를 생성할 수 있는 권한
- **RESOURCE 롤**
  - 사용자에게 자신의 테이블, 시퀀스, 프로시져, 트리거 객체 생성 할 수 있는 권한
  - 사용자 생성시 CONNECT 롤과 RESOURCE 롤을 부여
- **DBA 롤**
  - 시스템 자원의 무제한적인 사용이나 시스템 관리에 필요한 모든 권한
  - DBA 권한을 다른 사람에게 부여할 수 있음
  - 모든 사용자 소유의 CONNECT, RESOURCE, DBA 권한을 포함한 모든 권한을 부여 및 철회 가능

롤 생성

❖ 롤 생성

- CREATE ROLE 명령문으로 생성
- 롤에 암호 부여 가능
- 롤 이름은 사용자나 다른 롤과 중복될 수 없음

사용법

```
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 사용되는 암호 지정
- EXTERNALLY : 롤 활성화시 운영체제 인증을 통한 사용자 검증

롤 생성 예

사용예

암호를 지정한 롤과 지정하지 않은 롤을 생성하여라.

```
SQL> CONNECT system/manager

SQL> CREATE ROLE hr_clerk;
롤이 생성되었습니다.

SQL> CREATE ROLE hr_mgr
  2  IDENTIFIED BY manager;
롤이 생성되었습니다.
```

hr_mgr 롤에 'manager' 암호 지정

롤에 권한 부여

❖ 롤에 권한 또는 다른 롤 부여

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

사용법

```
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

롤에 권한 부여

❖ 롤에 시스템 권한 부여

- DBA 또는 GRANT ANY PRIVILEGE 권한을 가진 사용자는 롤에 시스템 권한을 부여 가능

사용예

hr_mgr 롤에 CREATE SESSION 시스템 권한을 부여하여라.

```
SQL> GRANT create session TO hr_mgr;
```

❖ 롤에 객체 권한 부여

- 사용자가 롤에 객체을 부여할 수 있는 경우
  - 사용자가 객체의 소유자인 경우
  - WITH GRANT OPTION 옵션과 함께 객체 권한을 부여 받은 경우

사용예

scott 사용자의 student 테이블의 모든 칼럼에 대한 select, insert, delete 객체 권한을 hr_clerk 롤에 부여하여라.

```
SQL> GRANT select, insert, delete ON SCOTT. student TO hr_clerk;
```

롤에 롤 부여

❖ 롤 부여

- 롤은 사용자 또는 다른 롤에게 롤 부여
- 롤을 부여받은 사용자나 다른 롤은 해당 롤이 가지는 모든 권한 사용
- WITH ADMIN OPTION
  - WITH ADMIN OPTION을 부여 받은 롤은 사용자나 다른 롤에게 해당 롤을 재부여 가능

사용예

hr_clerk 롤을 hr_mgr 롤과 tiger 사용자에게 부여하여라.

```
SQL> GRANT hr_clerk TO hr_mgr;
권한이 부여되었습니다.

SQL> GRANT hr_clerk TO tiger;
권한이 부여되었습니다.
```

hr_clerk 롤을 hr_mgr 롤에 부여

hr_clerk 롤을 tiger 사용자에 부여

롤 조회

❖ role_sys_privs

- 롤에 부여한 시스템 권한 조회

사용예

데이터 딕셔너리를 이용하여 롤 정의를 조회하여라.

```
SQL> SELECT * FROM role_sys_privs;        <롤에 부여한 시스템 권한 조회

ROLE                           PRIVILEGE                                ADM
------------------------------ ---------------------------------------- ---
AQ_ADMINISTRATOR_ROLE          DEQUEUE ANY QUEUE                        YES
AQ_ADMINISTRATOR_ROLE          ENQUEUE ANY QUEUE                        YES
AQ_ADMINISTRATOR_ROLE          MANAGE ANY QUEUE                         YES
DBA                            ADMINISTER DATABASE TRIGGER              YES
...                            ...                                      ...
248 개의 행이 선택되었습니다.
```

롤 조회

❖ role_tab_privs, user_role_privs

- 롤에 부여한 시스템 권한 조회

```
SQL> SELECT * FROM role_tab_privs;        롤에 부여한 객체 권한 조회

ROLE                    OWNER     TABLE_NAME             COLUMN_NAME
----------------------- --------- ---------------------- -----------
AQ_ADMINISTRATOR_ROLE   SYS       AQ$INTERNET_USERS
AQ_ADMINISTRATOR_ROLE   SYS       AQ$_PROPAGATION_STATUS
AQ_ADMINISTRATOR_ROLE   SYS       DBA_QUEUES
AQ_ADMINISTRATOR_ROLE   SYS       DBA_QUEUE_SCHEDULES
...                     ...       ...
1253 개의 행이 선택되었습니다.

SQL> SELECT * FROM user_role_privs;        사용자가 부여받은 롤 조회

USERNAME              GRANTED_ROLE                   ADM DEF OS_
--------------------- ------------------------------ --- --- ---
SYSTEM                AQ_ADMINISTRATOR_ROLE          YES YES NO
SYSTEM                DBA                            YES YES NO
SYSTEM                HR_CLERK                       YES YES NO
SYSTEM                HR_MGR                         YES YES NO
...                   ...                            ...
25 개의 행이 선택되었습니다.
```

동의어

❖ 동의어의 개념

- 하나의 객체에 대해 다른 이름을 정의하는 방법

❖ 동의어의 필요성

- 데이터베이스 객체의 소유권은 해당 객체를 생성한 사용자에게 있음
  - 사용자가 소유한 객체에 접근하기 위해서는 소유자로부터 접근 권한을 부여받아야만 함
  - 다른 사용자가 소유한 객체를 조회할 때에는 소유자의 아이디를 객체 이름 앞에 첨부해야 함
  - 예 : 학생 테이블의 소유자는 scott,
    홍길동이 scott로부터 학생 테이블에 대한 접근 권한을 부여받아 학생
    테이블을 조회하려면?
    - SELECT * FROM scott.student; <= 소유자 아이디를 테이블 이름 앞에 지정
  - 객체를 조회할 때마다 객체의 소유자를 일일이 지정하는 방법은 매우 번거러우므로 이런 번거로움을 최소화하기 위해 주로 동의어 사용

동의어

❖ 동의어와 별명(Alias) 차이점

- 동의어는 데이터베이스 전체에서 사용할 수 있는 객체
- 별명은 해당 SQL 명령문에서만 사용

❖ 동의어 생성

- 개별 사용자를 대상으로 한 전용 동의어, 전체 사용자를 대상으로 한 공용 동의어 존재

사용법

```
CREATE [ PUBLIC]  SYNONYM [ schema.] synonym
FOR [ schema.] object;
```

▶ PUBLIC : 모든 사용자가 사용 가능

▶ 기본 값은 전용 동의어

동의어의 종류

❖ 전용 동의어(private synonym)

- 객체에 대한 접근 권한을 부여 받은 사용자가 정의한 동의어로 해당 사용자만 사용

❖ 공용 동의어(public sysnonym)

- 권한을 주는 사용자가 정의한 동의어로 누구나 사용
- DBA 권한을 가진 사용자만 생성 (예 : 데이터 딕셔너리)

전용 동의어 생성 예

사용예

SYSTEM 사용자 소유의 project 테이블에 my_project 전용 동의어를 생성하여라.

```
SQL> CONNECT system/manager

SQL> CREATE TABLE project (
  2  project_id     NUMBER(5)
           CONSTRAINT pro_id_pk PRIMARY KEY,
  3  project_name   VARCHAR2(100),
  4  studno         NUMBER(5),
  5  profno         NUMBER(5));
테이블이 생성되었습니다.

SQL> INSERT INTO project VALUES(12345,'portfolio',10101,9901);
1 개의 행이 만들어졌습니다.

SQL> COMMIT;
커밋이 완료되었습니다.

SQL> SELECT * FROM project;

☞ 출력 결과
PROJECT_ID PROJECT_NAME  STUDNO  PROFNO
---------- ------------ ------- -------
     12345 portfolio      10101    9901

SQL> GRANT SELECT ON project TO scott;
권한이 부여되었습니다.
```

system 사용자로 접속

project 테이블 생성

전용 동의어 생성 예

```
SQL> CONNECT scott/tiger
연결되었습니다.

SQL> SELECT *
  2  FROM project;
SELECT * FROM project
              *
1행에 오류:
ORA-00942: 테이블 또는 뷰가 존재하지 않습니다.

SQL> SELECT *
  2  FROM system.project;

☞ 출력 결과
PROJECT_ID PROJECT_NAME  STUDNO  PROFNO
---------- ------------ ------- -------
     12345 portfolio      10101    9901

SQL> CREATE SYNONYM my_project FOR system.project;
동의어가 생성되었습니다.

SQL> SELECT * FROM my_project;

☞ 출력 결과
PROJECT_ID PROJECT_NAME  STUDNO  PROFNO
---------- ------------ ------- -------
     12345 portfolio      10101    9901
```

scott 사용자로 접속

소유자 지정없이 project 테이블을 조회하면 오류가 발생한다.

소유자를 지정하여 project 테이블을 조회한다.

my_project 전용 동의어 생성

전용 동의어에 의해 system 소유의 project 테이블을 조회한다.

공용 동의어 생성 예

사용예

system 사용자의 project 테이블에 대해 공용 동의어를 생성하여라.

```
SQL> CONNECT system/manager

SQL> CREATE PUBLIC SYNONYM pub_project FOR project;
동의어가 생성되었습니다.

SQL> CONNECT scott/tiger
연결되었습니다.

SQL> SELECT * FROM pub_project;

☞ 출력 결과
PROJECT_ID PROJECT_NAME   STUDNO   PROFNO
---------- ------------ ------- -------
     12345 portfolio      10101    9901
```

공용 동의어를 생성하기 위하여 sys 또는 system으로 접속한다.

pub_project 공용 동의어 생성

scott 사용자가 별도로 동의어를 생성하지 않아도 공용 동의어에 의해 system소유의 project 테이블을 조회한다.

동의어 삭제

❖ 동의어 삭제

- DROP SYNONYM 명령문 사용

사용법

```
DROP [ PUBLIC]  SYNONYM synonym;
```

❖ 전용 동의어 삭제 예

사용예

my_project와 pub_project 동의어를 삭제하여라.

```
SQL> CONNECT scott/tiger

SQL> DROP SYNONYM my_project;
동의어가 삭제되었습니다.
```

my_project를 생성한 scott에 의해 동의어를 삭제할 수 있다.

동의어 삭제 예

❖ 공용 동의어 삭제 예

```
SQL> CONNECT system/manager

SQL> DROP SYNONYM pub_project;

DROP SYNONYM pub_project
             *
1행에 오류:
ORA-01434: 삭제할 비공개 동의어가 존재하지 않습니다


SQL> DROP PUBLIC SYNONYM pub_project;

동의어가 삭제되었습니다.
```

공용 동의어는 DROP SYNONYM 명령으로 삭제할 수 없다.

공용 동의어는 DROP PUBLIC SYNONYM 명령으로 사용하여 삭제할 수 있다.