Chapter 14. 사용자 접근 제어

이번 장에서는 사용자를 생성하고 롤(Role)을 이용하여 사용자에게 여러 가지 권한(Privilege)을 효과적으로 부여 및 관리하는 방법을 설명한다. 권한 부여 및 회수를 위한 CREATE 및 REVOKE 문장을 살펴보고 원격 데이터베이스 연결을 위한 데이터베이스 링크의 작성 및 사용방법을 알아본다.

Oracle 데이터베이스의 보안

다중 사용자 환경에서 개별 사용자들은 데이터베이스 접근 및 사용에 있어서 적절한 보안을 유지하여야만 한다. 이를 위하여 Oracle 서버에서는 다음과 같은 작업을 수행 할 수 있다.

· 데이터베이스 접근 제어
· 데이터베이스의 특정 객체에 대한 접근 권한 부여
· Oracle 데이터 딕셔너리로부터 부여하거나 부여 받은 권한 확인
· 데이터베이스 객체에 대한 동의어 작성

데이터베이스 보안은 시스템 보안과 데이터 보안으로 분류 할 수 있다. 시스템 보안은 사용자 계정 생성, 암호 변경, 디스크 공간 할당, 시스템 작업 등과 같이 시스템 수준에서의 데이터베이스 접근 및 사용을 관리하는 것이며 데이터베이스 보안은 데이터베이스 객체에 대한 사용자들의 접근 및 사용을 관리하는 것이다.

사용자 생성

데이터베이스 관리자(DBA)는 CREATE USER 명령을 이용하여 사용자를 생성 할 수 있다. CREATE USER 명령의 문법은 다음과 같다.

```
CREATE USER user
IDENTIFIED BY password;
```

계정은 TOM, 암호는 JERRY인 사용자를 생성하는 방법은 다음과 같다.

```
SQL> CONNECT / AS SYSDBA
연결되었습니다.
SQL> CREATE USER TOM
  2  IDENTIFIED BY JERRY;

사용자가 생성되었습니다.
```

위에서 생성된 TOM은 아직 아무런 권한이 부여되지 않았기 때문에 어떠한 작업도 불가능하다.

TOM 사용자의 암호를 TIGER로 변경하는 방법은 다음과 같다.

```
SQL> ALTER USER TOM
  2  IDENTIFIED BY TIGER;

사용자가 변경되었습니다.
```

권한

권한이란 특별한 SQL 문장을 실행 할 수 있는 권리를 의미한다. 데이터베이스 관리자는 사용자들에게 데이터베이스와 데이터베이스 객체에 접근 할 수 있는 권한을 부여 할 수 있는 고급 사용자이며, 일반 사용자들은 데이터베이스에 접근 할 수 있는 시스템 권한과 데이터베이스 객체의 내용에 접근할 수 있는 객체 권한을 부여 받아야 한다. 또한, 사용자는 다른 사용자 또는 롤(Role)에게 권한을 부여 할 수 있는 권한을 부여 받을 수도 있다. 롤이란 관련 권한들의 논리적 집합이다.

참고로 스키마(Schema)란 테이블, 뷰, 시퀀스와 같은 객체들의 모음이다. 스키마는 데이터베이스 사용자가 소유하며 사용자 이름과 동일한 이름을 갖는다.

시스템 권한

Oracle에서 사용가능한 시스템 권한은 약 100여개 이상이며, 일반적으로 데이터베이스 관리자에 의해 부여된다. 다음은 데이터베이스 관리자가 가지고 있는 일반적인 시스템 권한이다.

표 14-1. 일반적인 데이터베이스 관리자의 시스템 권한

시스템 권한	수행 가능한 작업
CREATE USER	사용자 생성
DROP USER	사용자 삭제
DROP ANY TABLE	모든 스키마에서 테이블 삭제 가능
BACKUP ANY TABLE	모든 스키마에서 Exp 유틸리티를 이용하여 백업 가능
SELECT ANY TABLE	모든 스키마에서 테이블, 뷰, 스냅샷을 검색 가능
CREATE ANY TABLE	모든 스키마에서 테이블 생성 가능

사용자가 생성되면 데이터베이스 관리자는 특정한 시스템 권한을 사용자에게 부여해야 한다. 권한을 부여하는 명령은 다음과 같다.

```
GRANT privilege [, privilege ...]
TO user [, user | role | PUBLIC ...];
```

위에서 PUBLIC은 모든 사용자에게 지정된 권한을 부여하는 것이며, 다음은 일반 사용자에게 부여하는 일반적인 시스템 권한이다.

표 14-2. 일반적인 사용자에게 부여하는 시스템 권한

시스템 권한	수행 가능한 작업
CREATE SESSION	데이터베이스 연결
CREATE TABLE	사용자 스키마에 테이블 생성
CREATE SEQUENCE	사용자 스키마에 시퀀스 생성
CREATE VIEW	사용자 스키마에 뷰 생성
CREATE PROCEDURE	사용자 스키마에 저장 프로시져, 함수, 패키지 생성

TOM 사용자에게 시스템 권한을 부여하면 다음과 같다.

```
SQL> GRANT CREATE SESSION, CREATE TABLE,
  2  CREATE SEQUENCE, CREATE VIEW
  3  TO TOM;

권한이 부여되었습니다.
```

객체 권한

객체 권한은 특정 테이블, 뷰, 시퀀스, 프로시저 등에 특별한 작업을 수행 할 수 있는 권리이다. 각 객체는 아래 표와 같이 각각 부여 가능한 권리들의 집합을 가지고 있다. 예를 들어, 시퀀스에는 ALTER와 SELECT 할 수 있는 권한만 부여 가능하다.

표 14-3. 객체 권한

객체 권한	테이블	뷰	시퀀스	프로시저
ALTER	√		√	
DELETE	√	√		
EXECUTE				√
INDEX	√			
INSERT	√	√		
REFERENCES	√	√		
SELECT	√	√	√	
UPDATE	√	√		

위에서 보는 것과 같이 객체 권한은 객체에 따라 부여 할 수 있는 권한이 다르다. 사용자는 일반적으로 자신의 스키마에 저장된 모든 객체에 대하여 모든 권한을 부여 받기 때문에 다른 사용자 또는 롤에게 자신이 소유한 권한을 부여 할 수 있다. 객체 권한을 부여하는 명령은 다음과 같다.

```
GRANT object_priv [(columns)]
ON object
TO {user | role | PUBLIC}
[WITH GRANT OPTION];
```

객체 권한을 부여할 때 WITH GRANT OPTION을 사용하면 권한을 부여 받은 사람이 받은 권한을 다른 사용자에게 다시 부여 할 수 있다.

SCOTT가 자신의 EMP 테이블을 SELECT 할 수 있는 권한을 TOM에게 부여하는 방법은

다음과 같다.

```
SQL> CONNECT SCOTT/TIGER
연결되었습니다.
SQL> GRANT SELECT ON EMP TO TOM;

권한이 부여되었습니다.

SQL> CONNECT TOM/TIGER
연결되었습니다.
SQL> SELECT ENAME, JOB FROM SCOTT.EMP
  2  WHERE ENAME = 'SMITH';

ENAME      JOB
---------- ---------
SMITH      CLERK
```

DEPT 테이블의 특정 컬럼을 UPDATE 할 수 있는 권한을 TOM에게 부여하면 다음과 같다.

```
SQL> CONNECT SCOTT/TIGER
연결되었습니다.
SQL> GRANT UPDATE (DNAME, LOC) ON DEPT TO TOM;

권한이 부여되었습니다.

SQL> CONNECT TOM/TIGER
연결되었습니다.

SQL> UPDATE SCOTT.DEPT
  2  SET LOC='서울'
  3  WHERE DEPTNO = 10;

1 행이 갱신되었습니다.
```

WITH GRANT OPTION을 사용하면 부여된 권한을 받은 사용자가 해당 권한을 다른 사용자에게 부여 할 수 있다.

```
SQL> CONNECT SCOTT/TIGER
연결되었습니다.
SQL> GRANT SELECT, INSERT
  2  ON DEPT
  3  TO TOM
  4  WITH GRANT OPTION;

권한이 부여되었습니다.

SQL> CONNECT TOM/TIGER
연결되었습니다.
SQL> GRANT SELECT
  2  ON SCOTT.DEPT
  3  TO PUBLIC;

권한이 부여되었습니다.
```

부여된 권한의 확인

사용자에게 부여된 권한을 확인 할 수 있는 데이터 딕셔너리는 다음과 같다.

표 14-4. 권한 관련 데이터 딕셔너리

데이터 딕셔너리	설명
ROLE_SYS_PRIVS	롤에 부여된 시스템 권한
ROLE_TAB_PRIVS	롤에 부여된 테이블 권한
USER_ROLE_PRIVS	사용자가 접근 가능한 롤
USER_TAB_PRIVS_MADE	사용자가 부여한 객체 권한
USER_TAB_PRIVS_RECD	사용자에게 부여된 객체 권한
USER_COL_PRIVS_MADE	사용자가 부여한 컬럼에 대한 객체 권한
USER_COL_PRIVS_RECD	사용자에게 부여된 컬럼에 대한 객체 권한
USER_SYS_PRIVS	사용자에게 부여된 시스템 권한

SCOTT가 부여한 권한과 TOM이 부여 받은 권한을 확인하는 방법은 다음과 같다.

```
SQL> CONNECT SCOTT/TIGER
연결되었습니다.
SQL> SELECT GRANTEE, TABLE_NAME, GRANTOR, PRIVILEGE
  2  FROM USER_TAB_PRIVS_MADE;

GRANTEE    TABLE_NAME           GRANTOR    PRIVILEGE
---------- -------------------- ---------- --------------------
TOM        DEPT                 SCOTT      INSERT
TOM        DEPT                 SCOTT      SELECT
PUBLIC     DEPT                 TOM        SELECT
TOM        EMP                  SCOTT      SELECT

SQL> CONNECT TOM/TIGER
연결되었습니다.
SQL> SELECT OWNER, TABLE_NAME, GRANTOR, PRIVILEGE
  2  FROM USER_TAB_PRIVS_RECD;

OWNER      TABLE_NAME GRANTOR    PRIVILEGE
---------- ---------- ---------- --------------------
SCOTT      DEPT       SCOTT      INSERT
SCOTT      DEPT       SCOTT      SELECT
SCOTT      EMP        SCOTT      SELECT
```

객체 권한의 회수

객체 권한을 회수하면 WITH GRANT OPTION에 의해 다른 사람에게 부여된 권한도 모두 회수된다. 예를 들어, A 사용자가 B 사용자에게 특정 테이블의 SELECT 권한을 WITH GRANT OPTION으로 부여하고, B 사용자가 부여 받은 권한을 다시 WITH GRANT OPTION으로 C 사용자에게 부여한 후에 C 사용자가 부여 받은 권한을 다시 D 사용자에게 부여했다면 A 사용자가 B 사용자에게 부여된 권한을 회수하면 C, D 사용자에게 부여된 모든 권한도 회수된다. 부여된 객체 권한을 회수하는 명령은 다음과 같다.

```
REVOKE {privilege [, privilege ...] | ALL}
ON object
FROM {user [, user ...] | role | PUBLIC}
[CASCADE CONSTRAINTS];
```

위에서 CASCADE CONSTRAINTS 옵션을 추가하면 REFERENCES 권한에 의해 객체에 부여된 참조 무결성 제약조건도 삭제한다.

SCOTT가 TOM에게 부여한 부서 테이블의 SELECT, INSERT 권한을 회수하는 방법은 다음과 같다.

```
SQL> REVOKE SELECT, INSERT ON DEPT
  2  FROM TOM;

권한이 취소되었습니다.
```

롤(Role)

롤이란 사용자에게 부여 할 관련 권한들의 논리적인 집합이다. 롤을 사용하면 권한의 부여, 회수 작업을 단순화 할 수 있다. 사용자는 여러 개의 롤을 할당 받을 수 있으며, 여러 사용자는 같은 롤을 할당 받을 수도 있다.

먼저, 데이터베이스 관리자는 롤을 생성하고, 관련 권한들을 롤에 할당한 다음 사용자에게 해당 롤을 할당하면 된다.

```
SQL> CONNECT / AS SYSDBA
연결되었습니다.
SQL> CREATE ROLE MANAGER;

롤이 생성되었습니다.

SQL> GRANT CREATE TABLE, CREATE VIEW TO MANAGER;

권한이 부여되었습니다.

SQL> GRANT MANAGER TO TOM;

권한이 부여되었습니다.
```

데이터베이스 링크

데이터베이스 링크는 로컬 Oracle 데이터베이스 서버에서 원격 Oracle 데이터베이스 서버로의 단방향 통신 경로를 정의하는 포인터이다. 링크 포인터는 데이터 딕셔너리에 저장되며, 저장된 링크 포인터를 사용하기 위해서는 해당 데이터 딕셔너리가 있는 로컬 데이터베이스에 먼저 연결해야 한다.

데이터베이스 링크는 단방향이므로 A 데이터베이스에 연결된 사용자는 저장된 링크를 사용

하여 B 데이터베이스에 연결 할 수 있지만 B 데이터베이스에 연결된 사용자는 A 데이터베이스에 연결 할 수 없다. 만약, B 데이터베이스에 연결된 사용자가 A 데이터베이스에 연결하려면 B 데이터베이스에 A 데이터베이스로의 링크가 데이터 딕셔너리에 저장되어 있어야 한다.

데이터베이스 링크는 로컬 사용자가 원격 데이터베이스의 데이터에 접근 할 수 있도록 해주며, 이러한 연결을 수행하기 위해서는 각 데이터베이스가 고유한 전역 데이터베이스 이름을 갖고 있어야 한다. 데이터베이스 링크를 사용함으로서 얻을 수 있는 가장 큰 장점은 로컬 데이터베이스의 사용자들이 원격 데이터베이스내 객체에 접근하면 모든 권한은 해당 객체의 소유자 권한으로 한정된다는 점이다. 데이터베이스 링크는 데이터베이스 관리자가 생성하며 USER_DB_LINKS 데이터 딕셔너리를 이용하여 확인할 수 있다.

데이터베이스 링크를 생성하기에 앞서 TNSNAME.ORA 파일에 원격 데이터베이스의 접속 연결자를 추가한다. 여기서, 원격 Oracle 서버는 192.168.0.100, 서비스명은 ora92, 접속 연결자는 MYORA92이다.

```
# TNSNAMES.ORA Network Configuration File:
# C:WoracleWora92WnetworkWadminWtnsnames.ora
# Generated by Oracle configuration tools.

MYORA92 =
  (DESCRIPTION =
    (ADDRESS_LIST =
      (ADDRESS = (PROTOCOL = TCP)(HOST = 192.168.0.100)(PORT = 1521))
    )
    (CONNECT_DATA =
      (SERVER = DEDICATED)
      (SERVICE_NAME = ora92)
    )
  )
```

데이터베이스 링크를 작성한다.

```
SQL> CONNECT / AS SYSDBA
연결되었습니다.
SQL> CREATE PUBLIC DATABASE LINK LINK_TEST
  2  CONNECT TO SCOTT IDENTIFIED BY TIGER
  3  USING 'MYORA92';

데이타베이스 링크가 생성되었습니다.

SQL> SELECT * FROM DEPT@LINK_TEST;

    DEPTNO DNAME          LOC
---------- -------------- -------------
        10 ACCOUNTING     NEW YORK
        20 RESEARCH       DALLAS
        30 SALES          CHICAGO
        40 OPERATIONS     BOSTON
```

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.