ORACLE

16.사용자 관리

목차

1. 보안을 위한 권한

- 기업에서 보유하고 있는 데이터들은 자료 이상의 가치가 있으므로 외부에 노출되지 않도록 보안을 해야 한다.
- 데이터베이스를 운영하려면 데이터베이스에 대한 적절한 보안 대책을 마련해야 한다.
- 오라클은 다수의 사용자들이 데이터베이스에 저장된 정보를 공유해서 사용한다.
- 하지만 정보의 유출이나 불법적인 접근을 방지하기 위해서 철저한 보안 대책이 필요하다.
- 이러한 보안 대책을 위해서 데이터베이스 관리자가 있어야 한다.

1. 보안을 위한 권한

- 데이터베이스 관리자는 사용자가 데이터베이스의 객체(테이블, 뷰 등)에 대한 특정 권한을 가질 수 있도록 함으로서 다수의 사용자가 데이터베이스에 저장된 정보를 공유하면서도 정보에 대한 보안이 이루어지도록 한다.

- 데이터베이스에 접근하기 위해서는 사용자가 이름과 암호를 입력해서 로그인 인증을 받아야 한다.

- 이렇게 데이터베이스에 접속하는 사용자로부터 어떻게 데이터를 보안할 수 있을까?

- 사용자마다 서로 다른 권한과 롤을 부여함으로서 보안을 설정할 수 있다.

1.1 권한의 역할과 종류(1)

- 권한은 사용자가 특정 테이블을 접근할 수 있도록 하거나 해당 테이블에 SQL(SELECT/INSERT/UPDATE/DELETE) 문을 사용할 수 있도록 제한을 두는 것을 말한다.
- 데이터베이스 보안을 위한 권한은 시스템 권한(System Privileges)과 객체 권한(Object Privileges)으로 나뉜다.
- 시스템 권한은 사용자의 생성과 제거, DB 접근 및 각종 객체를 생성할 수 있는 권한 등 주로 DBA에 의해 부여되며 그 권한의 수가 80 가지가 넘기에 대표적인 시스템 권한만 정리하고 넘어간다.

시스템 권한	기능
CREATE USER	새롭게 사용자를 생성하는 권한
DROP USER	사용자를 삭제하는 권한
DROP ANY TABLE	임의의 테이블을 삭제할 수 있는 권한
QUERY REWRITE	함수 기반 인덱스를 생성하는 권한
BACKUP ANY TABLE	임의의 테이블을 백업할 수 있는 권한

1.1 권한의 역할과 종류(2)

- 데이터베이스를 관리하는 권한으로 다음과 같은 것이 있다. 이러한 권한은 시스템 관리자가 사용자에게 부여하는 권한이다.

시스템 권한	기능
CREATE SESSION	데이터베이스에 접속할 수 있는 권한
CREATE TABLE	사용자 스키마에서 테이블을 생성할 수 있는 권한
CREATE VIEW	사용자 스키마에서 뷰를 생성할 수 있는 권한
CREATE SEQUENCE	사용자 스키마에서 시퀀스를 생성할 수 있는 권한
CREATE PROCEDURE	사용자 스키마에서 함수를 생성할 수 있는 권한

- 객체 권한은 객체를 조작할 수 있는 권한이다.

- 객체는 우리가 학습한 것 중에서 테이블, 뷰 등을 예로 들 수 있고, 이미 학습한 시퀀스, 인덱스 등과 앞으로 배울 동의어가 모두 객체에 해당된다.

2. 사용자 생성(1)

- 회사에 새로운 사원이 입사하게 되면 시스템에 접속하도록 관리자가 계정을 하나 발급해준다.
- 지금까지는 개별 사용자로 접속해서 오라클 데이터베이스를 사용했지만, 사실은 부서별이나 사원의 직무에 따라 사용 가능한 테이블을 고려해서 오라클 데이터베이스에서도 사용자 계정을 발급해야한다.
- 권한은 사용자한테 부여하는 것이므로 사용자를 생성하는 것부터 살펴보도록 하자.
- 다음은 사용자 생성을 위한 CREATE USER 명령어의 형식이다.

형식	**CREATE USER *user_name*** **IDENTIFIED BY *password*;**

2. 사용자 생성(2)

- 사용자의 생성은 사용자의 이름과 암호를 지정하여 생성한다.
- 사용자를 생성하기 위해서도 권한이 필요하다.
- 우리가 지금까지 주로 사용해 왔던 개별 사용자는 사용자를 생성할 권한이 없다.
- 새로운 사용자 계정을 발급받기 전에 주의할 점이 있다.
- 1장에서 언급한 바 있지만, 사용자를 생성하기 위해서는 시스템 권한을 가지고 있어야 한다.
- 오라클 데이터베이스를 설치할 때 자동으로 생성되는 디폴트 사용자 가운데 시스템 권한을 가진 데이터베이스 관리자인 DBA는 SYS, SYSTEM이다.
- 그러므로 사용자 계정을 발급 받기 위해서 시스템 권한을 가진 SYSTEM으로 접속해야 한다.

2.1 사용자 생성하기

- CREATE USER 명령어를 사용하여 사용자명은 USER01 암호는 TIGER로 사용자를 생성해보자.
- 1. 오라클 설치할 때 지정했던 암호를 이용하여 system 사용자로 접속한다.

예 **conn system/password**

- 2. 사용자명은 USER01 암호는 TIGER로 사용자를 생성해보자. 사용자를 생성하기위해서는 CREATE USER 명령어를 사용한다.

예 **CREATE USER USER01 IDENTIFIED BY TIGER;**

- 3. 새롭게 생성된 사용자로 접속을 해보자.

예 **CONN USER01/TIGER**

```
SQL> CONN USER01/TIGER;
ERROR:
ORA-01045: user USER01 lacks CREATE SESSION privilege; logon denied


Warning: You are no longer connected to ORACLE.
SQL>
```

3. 권한 부여

- 사용자에게 시스템 권한 부여하기 위해서는 GRANT 명령어를 사용하다.

형식	**GRANT *privilege_name*, ...** **TO *user_name* ;**

- 만일 user_name 대신 PUBLIC을 기술하면 모든 사용자에게 해당 시스템 권한이 부여된다.
- PUBLIC 이란 DB 내에 있는 모든 계정 즉, 모든 계정을 의미한다.
- 우선 데이터베이스 관리자로 접속한다.
- 새로 생성된 user01에 데이터베이스에 접속할 수 있는 권한인 CREATE SESSION를 부여한다.
- 다시 user01 사용자로 접속을 시도하면 이번에는 데이터베이스에 성공적으로 접속하게 된다.

3.1 CREATE SESSION 권한 부여

- ex) USER01에 권한을 부여한다.
- 1. CREATE SESSION 권한 역시 DBA 만이 부여할 수 있으므로 system 으로 로그인한다.

예 **CONN system/password**

- 2. SYSTEM으로 로그인한 후에 다음과 같이 USER01 사용자에게 CREATE SESSION 권한을 부여한다.

예 **GRANT CREATE SESSION TO USER01;**

- 3. USER01 사용자에 데이터베이스에 연결할 수 있는 권한인 CREATE SESSION이 성공적으로 부여되었기에 USER01 사용자로 데이터베이스에 접속을 시도하면 성공적으로 접속된다.

예 **CONN USER01/TIGER;**

3.2 WITH ADMIN OPTION 옵션(1)

- 사용자에게 시스템 권한을 WITH ADMIN OPTION과 함께 부여하면 그 사용자는 데이터베이스 관리자가 아닌데도 불구하고 부여받은 시스템 권한을 다른 사용자(제 3의 일반계정)에게 부여할 수 있는 권한도 함께 부여 받게 된다.

- 데이터베이스 관리자로 로그인해서 사용자 USER02 와 USER03 를 생성한다.

- 역시 데이터베이스 관리자에서 USER02 와 USER03 에 데이터베이스에 접속할 수 있는 권한인 CREATE SESSION 권한을 부여하는데 USER02 는 WITH ADMIN OPTION을 지정하고 USER03 은 WITH ADMIN OPTION을 지정하지 않은 채 권한 부여를 할 것이다.

3.2 WITH ADMIN OPTION 옵션(2)

- USER02 사용자로 로그인하면 CREATE SESSION 권한을 USER01 사용자에게 부여할 수 있다.

- 이것이 가능해진 것은 USER02에게 WITH ADMIN OPTION을 사용하여 CREATE SESSION 권한을 부여하여 그 권한을 다른 사용자에게도 부여할 수 있도록 허용하였기 때문.

- USER03 사용자는 단순히 CREATE SESSION 권한만을 부여받았으므로 그 권한을 다른 사용자에게 부여할 수 없다.

3.2 WITH ADMIN OPTION 옵션(3)

- ex) WITH ADMIN OPTION과 함께 권한을 부여하여 그 권한을 다른 사용자에게 부여할 수 있도록 한다.
- 1. DBA 권한을 가진 SYSTEM 사용자로 접속한다.
- 2. 사용자명은 USER02 암호는 TIGER로 사용자를 생성한다. 사용자를 생성하기 위해서는 CREATE USER 명령어를 사용한다.

예 **CREATE USER USER02 IDENTIFIED BY TIGER;**

- 3. USER02에게 CREATE SESSION 권한을 WITH ADMIN OPTION을 지정하여 부여한다.

예

```
GRANT CREATE SESSION TO USER02
WITH ADMIN OPTION;
```

- 4. USER02 사용자로 접속한다.

예 **CONN USER02/TIGER;**

4. 객체 권한(1)

- 객체 권한은 특정 객체에 조작을 할 수 있는 권한이다. 객체의 소유자는 객체에 대한 모든 권한을 가진다.
- 다음은 객체와 권한을 설정할 수 있는 명령어를 매핑시켜 놓은 표이다.

권 한	TABLE	VIEW	SEQUENCE	PROCEDURE
ALTER	v		v	
DELETE	v	v		
EXECUTE				v
INDEX	v			
INSERT	v	v		
REFERENCES	v			
SELECT	v	v	v	
UPDATE	v	v		

4. 객체 권한(2)

- 객체 권한은 테이블이나 뷰나 시퀀스나 함수 등과 같은 객체별로 DML문(SELECT, INSERT, DELETE)을 사용할 수 있는 권한을 설정하는 것이다.
- 다음은 객체에 권한을 부여하기 위한 형식이다.

형식	**GRANT *privilege_name* [(*column_name*)] | ALL ①** **ON *object_name* | *role_name* | PUBLIC ②** **TO *user_name*; ③**

- GRANT 명령어의 형식은 어떤 객체(②)에 어떠한 권한(①)을 어느 사용자(③)에게 부여하는가를 설정한다. 시스템 권한과 차이점이 있다면 ON 옵션이 추가된다는 점이다. ON 다음에 테이블 객체나 뷰 객체 등을 기술한다.

4. 객체 권한(3)

- 특정 객체에 대한 권한은 그 객체를 만든 사용자에게만 기본적으로 주어진다.
- 우리가 지금까지 사용했던 EMP 테이블은 개별 사용자 소유의 테이블이다.
- 그러므로 다음과 같이 scott 사용자로 로그인해서 USER01 사용자가 테이블 객체 EMP를 조회할 수 있도록 권한 부여를 해야한다.

4.1 테이블 객체에 대한 SELECT 권한 부여

- ex) scott 사용자로 로그인해서 USER01 사용자가 테이블 객체 EMP를 조회할 수 있도록 권한 부여를 해야한다.
- 1. 개별 사용자로 접속한다.

  예 **conn scott/tiger;**

- 2. scott 사용자 소유의 EMP 테이블을 조회(SELECT)할 수 있는 권한을 USER01이란 사용자에게 부여한다.

  예 **GRANT SELECT ON EMP TO USER01;**

4.1 테이블 객체에 대한 SELECT 권한 부여

- 3. 권한 부여가 되었다면 다시 USER01로 로그인하여 EMP 테이블에 접속 해본다.

예

```
CONN USER01/TIGER
SHOW USER
SELECT * FROM EMP;
```

```
SQL> select * from emp;
select * from emp
              *
ERROR at line 1:
ORA-00942: table or view does not exist
```

- 권한 부여가 되었는데도 USER01은 EMP 테이블 객체를 조회할 수 없다. 그 이유는 객체의 소유자인 스키마를 지정하지 않았기 때문이다. 스키마에 대해 개념을 학습한 후에 특정 소유자의 테이블에 접근해보도록 하기 위해서 어떻게 해야 하는지 살펴보도록 하자.

4.2 스키마

- 다음과 같이 자신이 소유한 객체가 아닌 경우에는 그 객체를 소유한 사용자명을 반드시 기술해야한다.

예

```
CONN USER01/TIGER
SHOW USER
SELECT * FROM scott.EMP;
```

```
SQL> select * from scott.emp;

     EMPNO ENAME                JOB                       MGR HIREDATE
---------- -------------------- ------------------ ---------- --------
       SAL       COMM     DEPTNO
---------- ---------- ----------
      1001 김사랑                 사원                     1013 07/03/01
       300                    20

      1002 한예슬                 대리                     1005 07/04/02
       250         80         30

      1003 오지호                 과장                     1005 05/02/10
       500        100         30


     EMPNO ENAME                JOB                       MGR HIREDATE
---------- -------------------- ------------------ ---------- --------
       SAL       COMM     DEPTNO
---------- ---------- ----------
      1004 이병헌                 부장                     1008 03/09/02
       600                    20
```

4.3 사용자에게 부여된 권한 조회(1)

- 현재 사용자와 관련된 권한을 조회해보도록 한다.

- 사용자 권한과 관련된 데이터 딕셔너리 중에서 USER_TAB_PRIVS_MADE 데이터 딕셔너리는 현재 사용자가 다른 사용자에게 부여한 권한 정보를 알려준다.

- 만일 자신에게 부여된 사용자 권한을 알고 싶을 때에는 USER_TAB_PRIVS_RECD 데이터 딕셔너리를 조회하면 된다. USER01과 개별 사용자가 부여한 권한과 부여된 권한을 살펴보자.

4.3 사용자에게 부여된 권한 조회(2)

- USER01 계정으로 접속한다.

  예 **CONN USER01/TIGER**

- USER01 사용자가 부여한 권한을 살펴본다

  예 **SELECT * FROM USER_TAB_PRIVS_MADE;**

GRANTEE	TABLE_...	GRANTOR	PRIVILE...	GRANT...	HIERAR...

- USER01 사용자에게 부여한 권한을 살펴본다

  예 **SELECT * FROM USER_TAB_PRIVS_RECD;**

OWNER	TABLE_NAME	GRANTOR	PRIVILEGE	GRANTABLE	HIERARCHY
JOONZIS	EMP	JOONZIS	SELECT	NO	NO

4.3 사용자에게 부여된 권한 조회(3)

- scott 계정으로 접속한다.

  예 **CONN scott/tiger**

- scott 사용자가 부여한 권한을 살펴본다

  예 **SELECT * FROM USER_TAB_PRIVS_MADE;**

GRANTEE	TABLE_NAME	GRANTOR	PRIVILEGE	GRANTABLE	HIERARCHY
USER01	EMP	JOONZIS	SELECT	NO	NO

- scott 사용자에게 부여한 권한을 살펴본다

  예 **SELECT * FROM USER_TAB_PRIVS_RECD;**

GRANTEE	TABLE_...	GRANTOR	PRIVILE...	GRANT...	HIERAR...

4.4 REVOKE

- 사용자에게 부여한 객체 권한을 데이터베이스 관리자나 객체 소유자로부터 철회(권한을 취소)하기 위해서는 REVOKE 명령어를 사용한다. 다음은 REVOKE 명령어의 형식

형식	**REVOKE {*privilege_name* | all}** **ON *object_name*** **FROM {*user_name* | *role_name* | public};**

- REVOKE 명령어 다음에는 철회하고자하는 객체 권한을 기술하고 ON 다음에는 어떤 테이블에 부여된 권한인지 해당 테이블명을 기술하고 FROM 다음에는 어떤 사용자에게 부여한 권한인지 사용자명을 기술한다.

4.5 객체 권한 제거하기

- ex) SELECT 권한을 철회한다.
- 1. scott 계정으로 로그인한다.
- 2. SELECT 권한을 철회하기 전에 개별 계정에 설정된 권한을 살펴본다.

예 **SELECT * FROM USER_TAB_PRIVS_MADE;**

GRANTEE	TABLE_NAME	GRANTOR	PRIVILEGE	GRANTABLE	HIERARCHY
USER01	EMP	JOONZIS	SELECT	NO	NO

- 3. REVOKE SELECT ON EMP FROM USER01; 명령문은 USER01 사용자에게 부여된 EMP 테이블에 대한 SELECT 권한을 철회한다.

예 **REVOKE SELECT ON EMP FROM USER01;**

- 4. 권한이 철회되고 나면 데이터 딕셔너리에 객체 권한에 대한 정보도 함께 사라진다.

예 **SELECT * FROM USER_TAB_PRIVS_MADE;**

GRANTEE	TABLE_...	GRANTOR	PRIVILE...	GRANT...	HIERAR...

4.5 객체 권한 제거하기

- 5. USER01 사용자에게 부여된 EMP 테이블에 대한 SELECT 권한을 철회하였기에 USER01 사용자 계정으로 로그인해서 개별 사용자의 EMP 테이블을 사용할 수 없다.

예

```
CONN USER01/TIGER

SELECT * FROM scott.emp;
```

```
ORA-00942: table or view does not exist
00942, 00000 - "table or view does not exist"
*Cause:
*Action:
1행, 23열에서 오류 발생
```

4.6 WITH GRANT OPTION(1)

- 사용자에게 객체 권한을 WITH GRANT OPTION과 함께 부여하면 그 사용자는 그 객체를 접근할 권한을 부여 받으면서 그 권한을 다른 사용자에게 부여 할 수 있는 권한도 함께 부여받게된다.

- scott 사용자로 로그인해서 사용자 USER02와 USER03에게 EMP 테이블 객체를 SELECT 할 수 있는 권한을 부여하는데 USER02는 WITH GRANT OPTION을 지정하고 USER03은 WITH GRANT OPTION을 지정하지 않아서 차이점을 확인해본다.

- USER02는 WITH GRANT OPTION을 지정하였기에 USER02로 로그인해서 객체권한을 또 다른 사용자에게 부여할 수 있다.

4.6 WITH GRANT OPTION(2)

- USER03는 WITH GRANT OPTION을 지정하지 않았기에 USER03으로 로그인해서 객체 권한을 또 다른 사용자에게 부여할 수 없다.

```
SQL> conn joonzis/password
Connected.
SQL> grant select on joonzis.emp to user02
  2  with grant option;

Grant succeeded.
```

```
SQL> conn user02/tiger
Connected.
SQL> grant select on joonzis.emp to user01;

Grant succeeded.
```

```
SQL> conn joonzis/password
Connected.
SQL> grant create session to user03;

Grant succeeded.
```

```
SQL> conn user03/tiger
Connected.
SQL> grant select on joonzis.emp to user01;
grant select on joonzis.emp to user01
                               *
ERROR at line 1:
ORA-01031: insufficient privileges
```

- USER02는 WITH GRANT OPTION을 지정하였기에 USER02로 로그인해서 객체권한을 다른 사용자에게 부여할 수 있다.
- USER03는 WITH GRANT OPTION을 지정하지 않았기에 USER03으로 로그인해서 객체 권한을 또 다른 사용자에게 부여할 수 없다.

문제

1. kbs라는 사용자를 생성(암호는 pass)하여 connect와 resource 권한을 kbs 사용자에게 부여하여 오라클에 접속하도록 하시오.

```
SQL> conn kbs/pass;
Connected.
```

1. DBA 권한을 가진 계정으로 접속

conn /as sysdba;

2. 계정 생성

create user kbs identified by pass;

3. 권한 부여

grant connect, resource to kbs;