

17장 사용자 관리

이 장에서 다룰 내용

1 데이터베이스 보안을 위한 권한

2 사용자 생성

3 권한 부여하는 GRANT 명령어

4 객체 권한

01. 데이터베이스 보안을 위한 권한

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

01. 데이터베이스 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 합니다.
- ❖ 데이터베이스에 접근하기 위해서는 사용자가 이름과 암호를 입력해서 로그인 인증을 받아야 합니다.
- ❖ 이렇게 데이터베이스에 접속하는 사용자로부터 어떻게 데이터를 보안할 수 있을까요?
- ❖ 사용자마다 서로 다른 권한과 롤을 부여함으로써 보안을 설정할 수 있습니다.

1.1 권한의 역할과 종류

- ❖ 권한은 사용자가 특정 테이블을 접근할 수 있도록 하거나 해당 테이블에 SQL(SELECT/INSERT/UPDATE/DELETE) 문을 사용할 수 있도록 제한을 두는 것을 말합니다.
- ❖ 데이터베이스 보안을 위한 권한은 시스템 권한(System Privileges)과 객체 권한(Object Privileges)으로 나뉩니다.
- ❖ 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 넘어갑시다.

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

- ❖ 데이터베이스를 관리하는 권한으로 다음과 같은 것이 있습니다. 이러한 권한은 시스템 관리자가 사용자에게 부여하는 권한입니다.

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



02. 사용자 생성

- ❖ 회사에 새로운 사원이 입사하게 되면 시스템에 접속하도록 관리자가 계정을 하나 발급해 줍니다.
- ❖ 지금까지는 SCOTT 사용자로 접속해서 오라클 데이터베이스를 사용했지만, 사실은 부서별이나 사원의 직무에 따라 사용 가능한 테이블을 고려해서 오라클 데이터베이스에서도 사용자 계정을 발급해야 합니다.
- ❖ 권한은 사용자한테 부여하는 것이므로 사용자를 생성하는 것부터 살펴보도록 합시다.
- ❖ 다음은 사용자 생성을 위한 CREATE USER 명령어의 형식입니다.

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

02. 사용자 생성

- ❖ 사용자의 생성은 사용자의 이름과 암호를 지정하여 생성합니다.
- ❖ 사용자를 생성하기 위해서도 권한이 필요합니다.
- ❖ 우리가 지금까지 주로 사용해 왔던 SCOTT이란 사용자는 사용자를 생성할 권한이 없습니다.
- ❖ 새로운 사용자 계정을 발급받기 전에 주의할 점이 있습니다.
- ❖ 1장에서 언급한 바 있지만, 사용자를 생성하기 위해서는 시스템 권한을 가지고 있어야 합니다.
- ❖ 오라클 데이터베이스를 설치할 때 자동으로 생성되는 디폴트 사용자 가운데 시스템 권한을 가진 데이터베이스 관리자인 DBA는 SYS, SYSTEM입니다.
- ❖ 그러므로 사용자 계정을 발급 받기 위해서 시스템 권한을 가진 SYSTEM으로 접속해야 합니다.

〈실습하기〉 사용자 생성하기

CREATE USER 명령어를 사용하여 사용자명은 USER01 암호는 TIGER로 사용자를 생성해봅시다.

1. 오라클 설치할 때 system 사용자의 암호를 manager 로 지정하였기 때문에 위와 같이 입력하였습니다. 오라클 설치할 때 암호를 변경하지 않았다면 다음과 같이 접속합니다.

CONN system/manager

2. 사용자명은 USER01 암호는 TIGER로 사용자를 생성해봅시다. 사용자를 생성하기 위해서는 CREATE USER 명령어를 사용합니다.

CREATE USER USER01 IDENTIFIED BY TIGER;

3. 새롭게 생성된 사용자로 접속을 해봅시다.

CONN USER01/TIGER

C:\Windows\system32\cmd.exe - SQLPLUS scott/tiger

SQL> CONN USER01/TIGER

ERROR:

ORA-01045: user USER01 lacks CREATE SESSION privilege; logon denied

경고: 이제는 ORACLE에 연결되어 있지 않습니다.

SQL>

03. 권한 부여하는 GRANT 명령어

- ❖ 사용자에게 시스템 권한 부여하기 위해서는 GRANT 명령어를 사용합니다.

```
GRANT privilege_name, ...  
TO user_name;
```

- ❖ 만일 `user_name` 대신 `PUBLIC`을 기술했다면 모든 사용자에게 해당 시스템 권한이 부여됩니다.
- ❖ `PUBLIC` 이란 DB 내에 있는 모든 계정 즉, 모든 계정을 의미합니다.
- ❖ 우선 데이터베이스 관리자로 접속합니다.
- ❖ 새로 생성된 `user01`에 데이터베이스에 접속할 수 있는 권한인 `CREATE SESSION`를 부여합니다.
- ❖ 다시 `user01` 사용자로 접속을 시도하면 이번에는 데이터베이스에 성공적으로 접속하게 됩니다.

〈실습하기〉 CREATE SESSION 권한 부여하기

새로 생성된 user01에 데이터베이스에 접속할 수 있는 권한인 CREATE SESSION를 부여합니다. 다시 user01 사용자로 접속을 시도하면 이번에는 데이터베이스에 성공적으로 접속하게 됩니다.

1. CREATE SESSION 권한 역시 DBA 만이 부여할 수 있으므로 system 으로 로그인합니다.

CONN system/manager

2., SYSTEM으로 로그인한 후에 다음과 같이 USER01 사용자에게 CREATE SESSION 권한을 부여합니다.

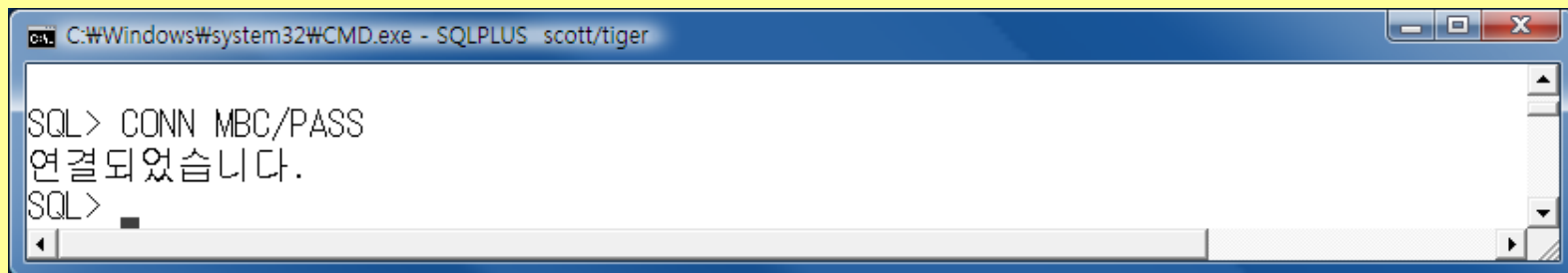
GRANT CREATE SESSION TO USER01;

3. USER01 사용자에게 데이터베이스에 연결할 수 있는 권한인 CREATE SESSION이 성공적으로 부여되었기에 USER01 사용자로 데이터베이스에 접속을 시도하면 성공적으로 접속됩니다.

CONN USER02/TIGER;

<탄탄히 다지기>

1. MBC 라는 사용자를 생성하고 기본적인 권한 부여를 하지 않으면 데이터베이스에 로그인 불가능하므로 CONNECT 과 RESOURCE 권한을 MBC 사용자에게 부여하시오.



```
C:\Windows\system32\CMD.exe - SQLPLUS scott/tiger

SQL> CONN MBC/PASS
연결되었습니다.
SQL>
```

3.1 WITH ADMIN OPTION 옵션

- ❖ **사용자에게 시스템 권한을 WITH ADMIN OPTION과 함께 부여하면 그 사용자는 데이터베이스 관리자가 아닌데도 불구하고 부여받은 시스템 권한을 다른 사용자에게 부여할 수 있는 권한도 함께 부여 받게 됩니다.**
- ❖ **데이터베이스 관리자로 로그인해서 사용자 USER02 와 USER03 를 생성합니다.**
- ❖ **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 것입니다.**

3.1 WITH ADMIN OPTION 옵션

- ❖ **USER02** 사용자로 로그인하면 **CREATE SESSION** 권한을 **USER01** 사용자에게 부여할 수 있습니다.
- ❖ 이것이 가능해진 것은 **USER02**에게 **WITH GRANT OPTION**을 사용하여 **CREATE SESSION** 권한을 부여하여 그 권한을 다른 사용자에게도 부여할 수 있도록 허용하였기 때문입니다.
- ❖ **USER03** 사용자는 단순히 **CREATE SESSION** 권한만을 부여받았으므로 그 권한을 다른 사용자에게 부여할 수 없습니다.

〈실습하기〉 WITH ADMIN OPTION을 지정하여 권한 부여

WITH ADMIN OPTION과 함께 권한을 부여하여 그 권한을 다른 사용자에게 부여할 수 있도록 합니다.

1. DBA 권한을 가진 system 사용자로 접속합니다.
2. 사용자명은 USER02 암호는 TIGER로 사용자를 생성해봅시다. 사용자를 생성하기 위해서는 CREATE USER 명령어를 사용합니다.

```
CREATE USER USER02 IDENTIFIED BY TIGER;
```

3. USER02에게 CREATE SESSION 권한을 WITH ADMIN OPTION을 지정하여 부여합니다.

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

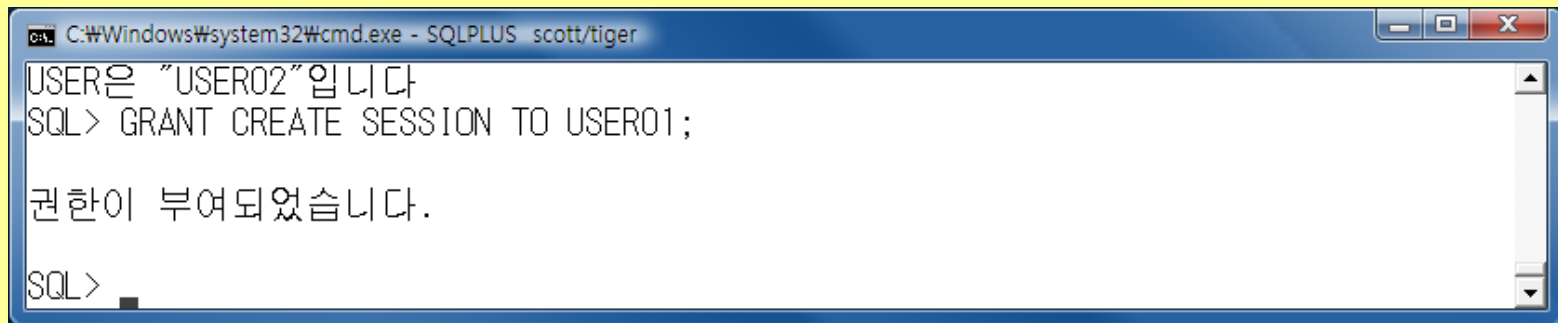
3. USER02 사용자로 접속합니다.

```
CONN USER02/TIGER;
```

〈실습하기〉 CREATE SESSION 권한 부여하기

4. USER02 사용자는 자기가 받은 권한을 다른 사용자에게 부여할 수 있습니다.

GRANT CREATE SESSION TO USER01;



```
C:\Windows\system32\cmd.exe - SQLPLUS scott/tiger
USER은 "USER02"입니다
SQL> GRANT CREATE SESSION TO USER01;

권한이 부여되었습니다.

SQL> _
```


〈실습하기〉 WITH ADMIN OPTION을 지정하지 않고 권한 부여

WITH ADMIN OPTION을 지정하지 않으면 부여 받은 권한을 다른 사용자에게 부여할 수 없음을 확인해 봅시다.

1. 사용자명은 USER03 암호는 TIGER로 사용자를 생성해봅시다. 사용자를 생성하기 위해서는 CREATE USER 명령어를 사용합니다.
2. USER03에게 WITH ADMIN OPTION을 지정하지 않고 CREATE SESSION 권한을 부여합니다.

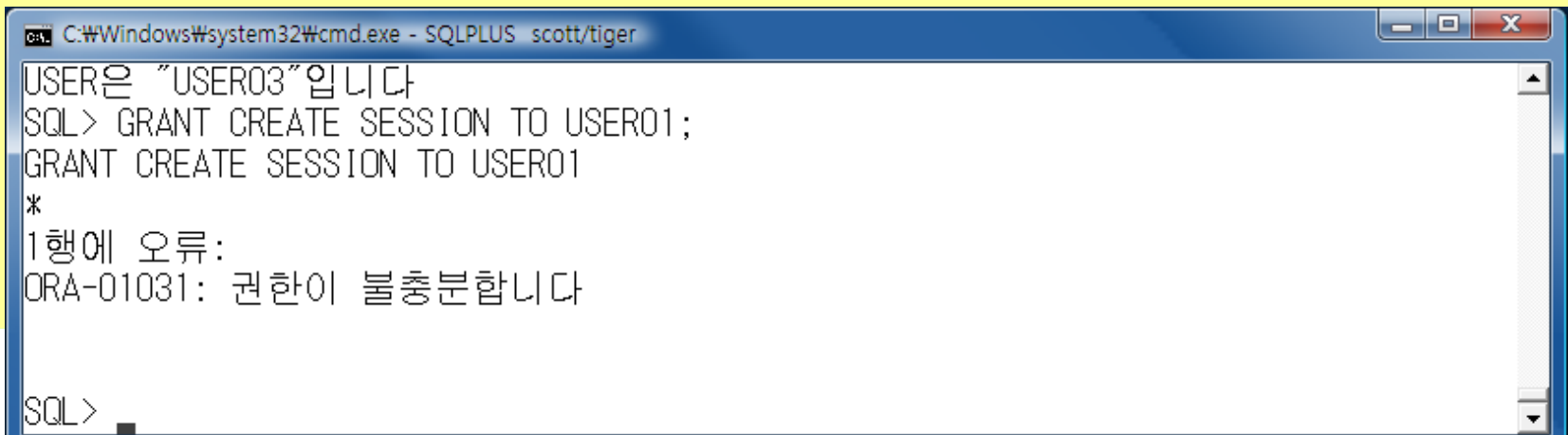
GRANT CREATE SESSION TO USER03;

3. USER03 사용자로 접속합니다.

CONN USER03/TIGER;

4. USER03 사용자는 자기가 받은 권한을 다른 사용자에게 부여할 수 없습니다

GRANT CREATE SESSION TO USER01;



```
C:\Windows\system32\cmd.exe - SQLPLUS scott/tiger
USER은 "USER03"입니다
SQL> GRANT CREATE SESSION TO USER01;
GRANT CREATE SESSION TO USER01
*
1행에 오류:
ORA-01031: 권한이 불충분합니다
SQL>
```

04. 객체 권한

- ❖ 객체 권한은 특정 객체에 조작을 할 수 있는 권한입니다. 객체의 소유자는 객체에 대한 모든 권한을 가집니다.
- ❖ 다음은 객체와 권한 설정할 수 있는 명령어를 매핑시켜 놓은 표입니다.

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		

04. 객체 권한

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

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

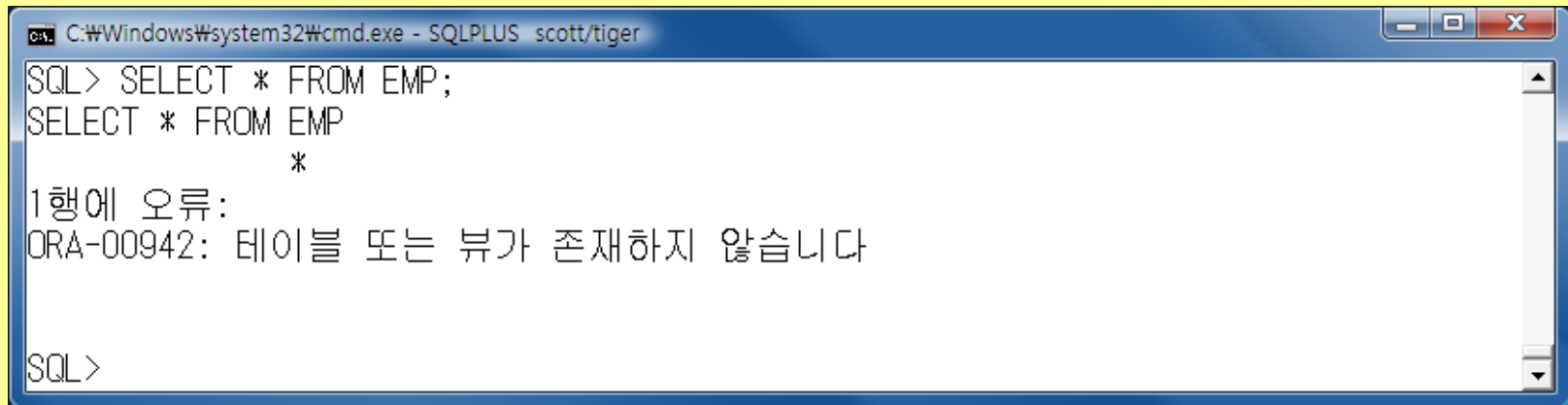
- ❖ GRANT 명령어의 형식은 어떤 객체(②)에 어떠한 권한(①)을 어느 사용자(③)에게 부여하는가를 설정합니다. 시스템 권한과 차이점이 있다면 ON 옵션이 추가된다는 점입니다. ON 다음에 테이블 객체나 뷰 객체 등을 기술합니다.

〈실습하기〉 다른 유저의 객체 접근하기

객체 권한을 부여해야 하는 필요성을 살펴보기 위해서 새롭게 생성한 USER01 객체로 EMP 테이블의 내용을 조회해 봅시다.

1. USER01로 접속합니다.
2. EMP 테이블을 조회합니다.

```
SELECT * FROM EMP;
```



The screenshot shows a Windows command prompt window titled "C:\Windows\system32\cmd.exe - SQLPLUS scott/tiger". The user has entered the SQL command "SELECT * FROM EMP;". The prompt shows the command being entered twice, with a cursor at the end of the second line. Below the command, the output shows an error: "1행에 오류:" followed by "ORA-00942: 테이블 또는 뷰가 존재하지 않습니다". The prompt "SQL>" is visible at the bottom.

```
C:\Windows\system32\cmd.exe - SQLPLUS scott/tiger
SQL> SELECT * FROM EMP;
SELECT * FROM EMP
      *
1행에 오류:
ORA-00942: 테이블 또는 뷰가 존재하지 않습니다

SQL>
```

04. 객체 권한

- ❖ 특정 객체에 대한 권한은 그 객체를 만든 사용자에게만 기본적으로 주어집니다.
- ❖ 우리가 지금까지 사용했던 EMP 테이블은 SCOTT 사용자 소유의 테이블입니다.
- ❖ 그러므로 다음과 같이 SCOTT 사용자로 로그인해서 USER01 사용자가 테이블 객체 EMP를 조회할 수 있도록 권한 부여를 해야 합니다.

〈실습하기〉 테이블 객체에 대한 SELECT 권한 부여하기

특정 객체에 대한 권한은 그 객체를 만든 사용자에게만 기본적으로 주어집니다. 우리가 지금까지 사용했던 EMP 테이블은 SCOTT 사용자 소유의 테이블입니다. 그러므로 SCOTT 사용자로 로그인해서 USER01 사용자가 테이블 객체 EMP를 조회할 수 있도록 권한 부여를 해야 합니다.

1. SCOTT으로 접속합니다.

```
CONN scott/tiger
```

2. SCOTT 사용자 소유의 EMP 테이블을 조회(SELECT)할 수 있는 권한을 USER01이란 사용자에게 부여합니다.

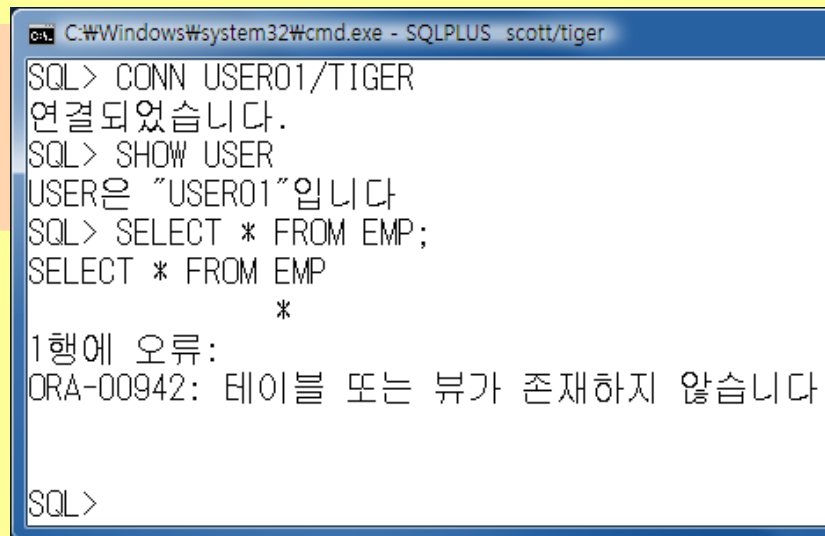
```
GRANT SELECT ON EMP TO USER01;
```

.

<실습하기> 테이블 객체에 대한 SELECT 권한 부여하기

3. 권한 부여가 되었다면 다시 USER01로 로그인하여 EMP 테이블에 접속해 봅시다.

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



```
C:\Windows\system32\cmd.exe - SQLPLUS scott/tiger
SQL> CONN USER01/TIGER
연결되었습니다.
SQL> SHOW USER
USER은 "USER01"입니다
SQL> SELECT * FROM EMP;
SELECT * FROM EMP
                *

1행에 오류:
ORA-00942: 테이블 또는 뷰가 존재하지 않습니다

SQL>
```

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

4.1 스키마

- ❖ 스키마(SCHEMA)란 객체를 소유한 사용자명을 의미합니다. 객체명 앞에 소속 사용자명을 기술합니다.

```
SELECT * FROM SCOTT.EMP;
```

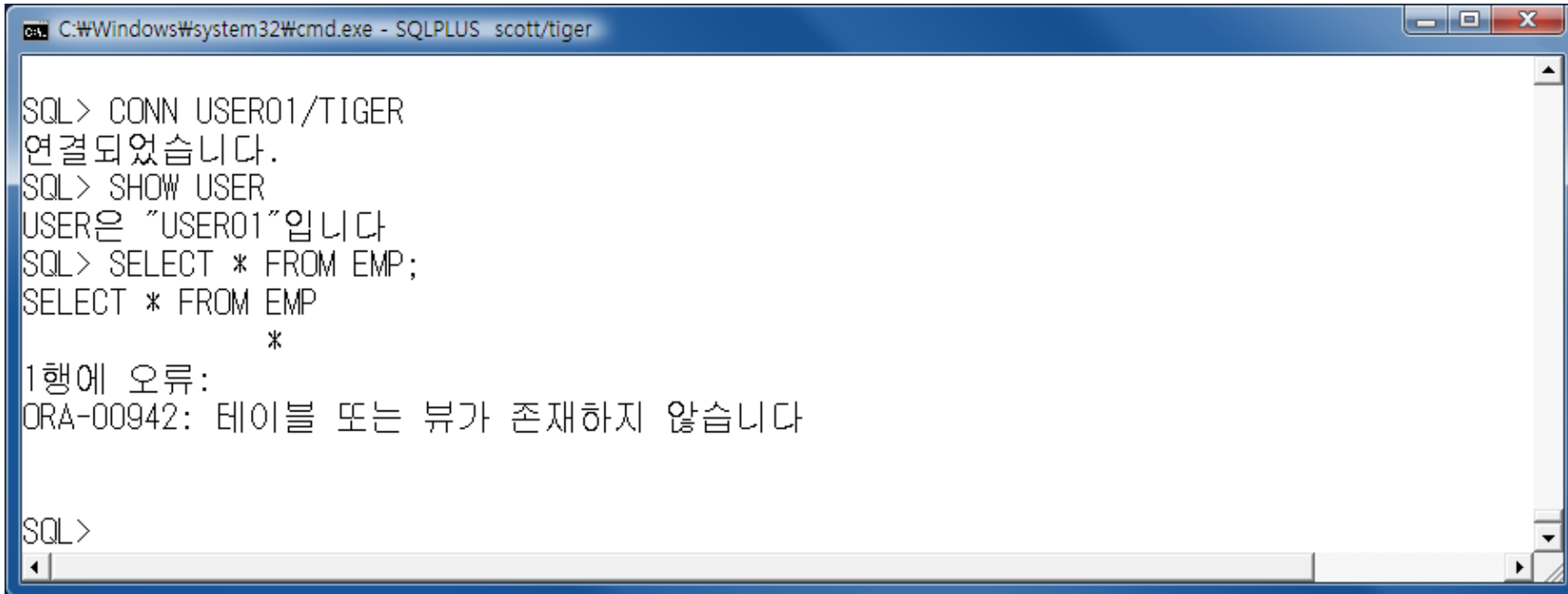
- ❖ EMP 앞에 SCOTT은 EMP 테이블 객체를 소유한 사용자명입니다. 지금까지 객체를 기술할 때 다음과 같이 기술했습니다.

```
SELECT * FROM EMP;
```

- ❖ 왜냐하면 현재 사용자가 SCOTT일 경우 자신이 소유한 객체를 언급할 때 객체명 앞에 스키마를 생략할 수 있기 때문입니다.
- ❖ 하지만 USER01 계정에서 다음과 같이 기술하면 자기 자신인 USER01 사용자가 소유한 EMP 테이블을 조회하려고 하다가 USER01 사용자는 EMP 테이블을 소유하지 않았기 때문에 다음과 같은 에러가 발생합니다.

4.1 스키마

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



```
C:\Windows\system32\cmd.exe - SQLPLUS scott/tiger

SQL> CONN USER01/TIGER
연결되었습니다.
SQL> SHOW USER
USER은 "USER01"입니다
SQL> SELECT * FROM EMP;
SELECT * FROM EMP
          *

1행에 오류:
ORA-00942: 테이블 또는 뷰가 존재하지 않습니다

SQL>
```

4.1 스키마

- ❖ 다음과 같이 자신이 소유한 객체가 아닌 경우에는 그 객체를 소유한 사용자명을 반드시 기술해야 합니다.

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

```
SQL> CONN USER01/TIGER
연결되었습니다.
SQL> SHOW USER
USER은 "USER01"입니다
SQL> SELECT * FROM SCOTT.EMP;
```

EMPNO	ENAME	JOB	MGR	HIREDATE	SAL	COMM	DEPTNO
7369	SMITH	CLERK	7902	80/12/17	800		20
7499	ALLEN	SALESMAN	7698	81/02/20	1600	300	30
7521	WARD	SALESMAN	7698	81/02/22	1250	500	30
7566	JONES	MANAGER	7839	81/04/02	2975		20
7654	MARTIN	SALESMAN	7698	81/09/28	1250	1400	30
7698	BLAKE	MANAGER	7839	81/05/01	2850		30
7782	CLARK	MANAGER	7839	81/06/09	2450		10
7788	SCOTT	ANALYST	7566	87/04/19	3000		20
7839	KING	PRESIDENT		81/11/17	5000		10
7844	TURNER	SALESMAN	7698	81/09/08	1500	0	30
7876	ADAMS	CLERK	7788	87/05/23	1100		20
7900	JAMES	CLERK	7698	81/12/03	950		30
7902	FORD	ANALYST	7566	81/12/03	3000		20
7934	MILLER	CLERK	7782	82/01/23	1300		10

4.2 사용자에게 부여된 권한 조회

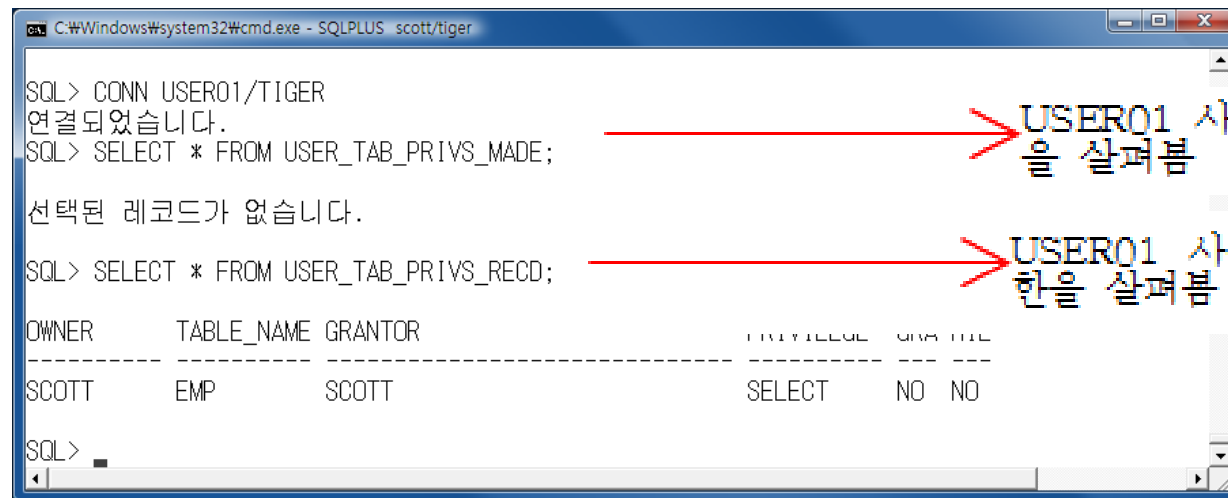
- ❖ 현재 사용자와 관련된 권한을 조회해보도록 합니다.
- ❖ 사용자 권한과 관련된 데이터 디렉터리 중에서 `USER_TAB_PRIVS_MADE` 데이터 디렉터리는 현재 사용자가 다른 사용자에게 부여한 권한 정보를 알려줍니다.
- ❖ 만일 자신에게 부여된 사용자 권한을 알고 싶을 때에는 `USER_TAB_PRIVS_RECD` 데이터 디렉터리를 조회하면 됩니다. `USER01`과 `SCOTT` 사용자가 부여한 권한과 부여된 권한을 살펴봅시다.

4.2 사용자에게 부여된 권한 조회

CONN USER01/TIGER

SELECT * FROM USER_TAB_PRIVS_MADE;

SELECT * FROM USER_TAB_PRIVS_RECD;



```
C:\Windows\system32\cmd.exe - SQLPLUS scott/tiger
SQL> CONN USER01/TIGER
연결되었습니다.
SQL> SELECT * FROM USER_TAB_PRIVS_MADE;

선택된 레코드가 없습니다.

SQL> SELECT * FROM USER_TAB_PRIVS_RECD;

OWNER      TABLE_NAME GRANTOR      PRIVILEGE      GRANTABLE
-----
SCOTT      EMP          SCOTT         SELECT          NO          NO

SQL>
```

→ USER01 사용자가 부여한 권한
을 살펴봄

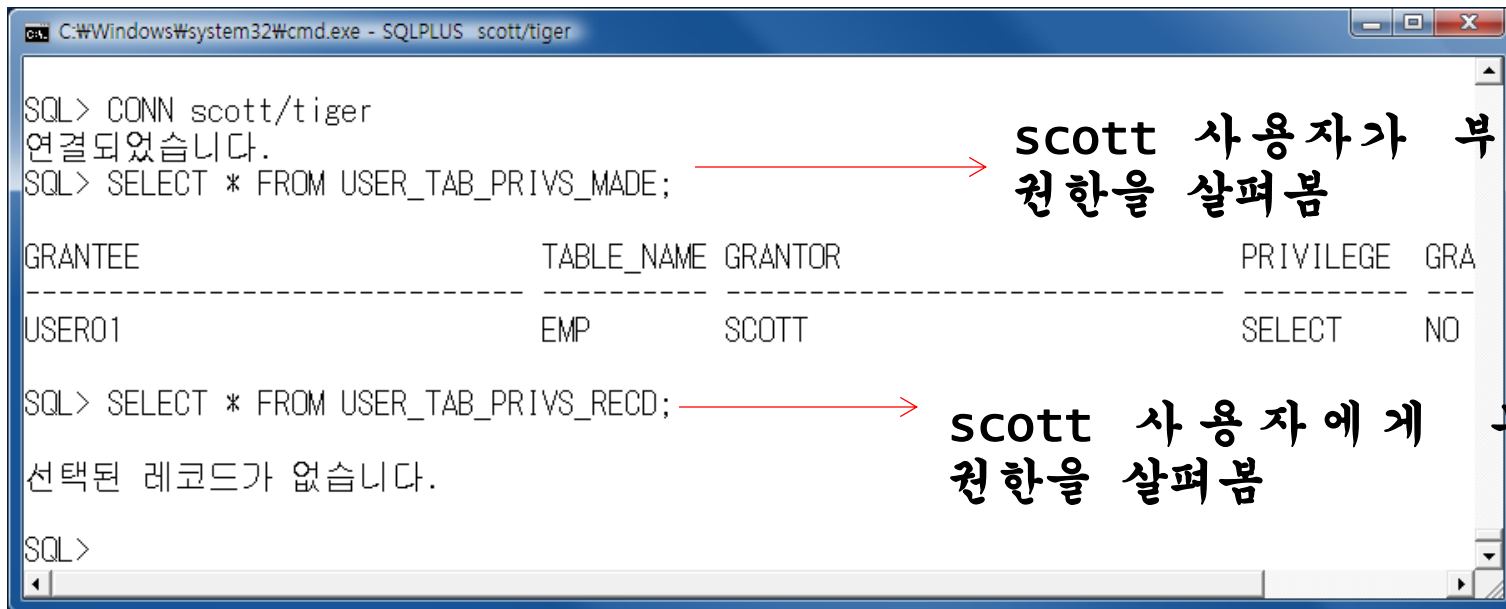
→ USER01 사용자에게 부여된 권
한을 살펴봄

4.2 사용자에게 부여된 권한 조회

CONN scott/tiger

SELECT * FROM USER_TAB_PRIVS_MADE;

SELECT * FROM USER_TAB_PRIVS_RECD;



The screenshot shows a Windows command prompt window titled "C:\Windows\system32\cmd.exe - SQLPLUS scott/tiger". The user has entered the following commands:

```
SQL> CONN scott/tiger
연결되었습니다.
SQL> SELECT * FROM USER_TAB_PRIVS_MADE;
```

The output of the first query is a table with the following columns: GRANTEE, TABLE_NAME, GRANTOR, PRIVILEGE, and GRA. The table contains one row of data:

GRANTEE	TABLE_NAME	GRANTOR	PRIVILEGE	GRA
USER01	EMP	SCOTT	SELECT	NO

The user then enters the second query:

```
SQL> SELECT * FROM USER_TAB_PRIVS_RECD;
```

The output of the second query is the message: "선택된 레코드가 없습니다." (No records selected).

scott 사용자가 부여한
권한을 살펴봄

scott 사용자에게 부여된
권한을 살펴봄

4.3 사용자에게서 권한을 뺏기 위한 REVOKE 명령어

- ❖ 사용자에게 부여한 객체 권한을 데이터베이스 관리자나 객체 소유자로부터 철회하기 위해서는 REVOKE 명령어를 사용합니다. 다음은 REVOKE 명령어의 형식입니다.

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

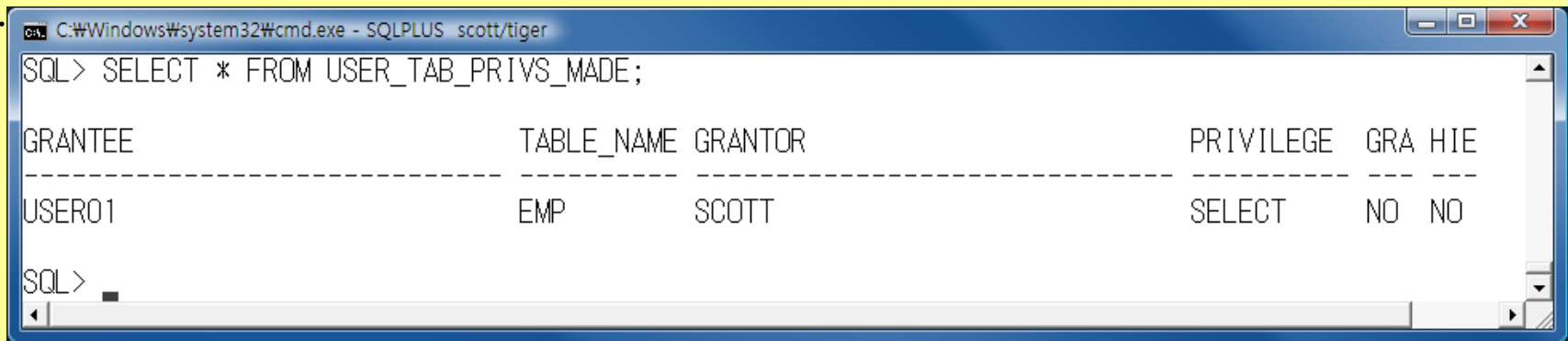
- ❖ REVOKE 명령어 다음에는 철회하고자하는 객체 권한을 기술하고 ON 다음에는 어떤 테이블에 부여된 권한인지 해당 테이블명을 기술하고 FROM 다음에는 어떤 사용자에게 부여한 권한인지 사용자명을 기술합니다.

〈실습하기〉 객체 권한 제거하기

SELECT 권한을 철회해 봅시다.

1. SCOTT 계정으로 로그인합니다.
2. SELECT 권한을 철회하기 전에 SCOTT 계정에 설정된 권한을 살펴봅시다.

```
SELECT * FROM USER_TAB_PRIVS_MADE;
```



The screenshot shows a Windows command prompt window titled "C:\Windows\system32\cmd.exe - SQLPLUS scott/tiger". The user has entered the command "SQL> SELECT * FROM USER_TAB_PRIVS_MADE;". The output is a table with five columns: GRANTEE, TABLE_NAME, GRANTOR, PRIVILEGE, and GRA HIE. The table shows that the user USER01 has been granted SELECT privilege on the EMP table by SCOTT.

GRANTEE	TABLE_NAME	GRANTOR	PRIVILEGE	GRA HIE
USER01	EMP	SCOTT	SELECT	NO NO

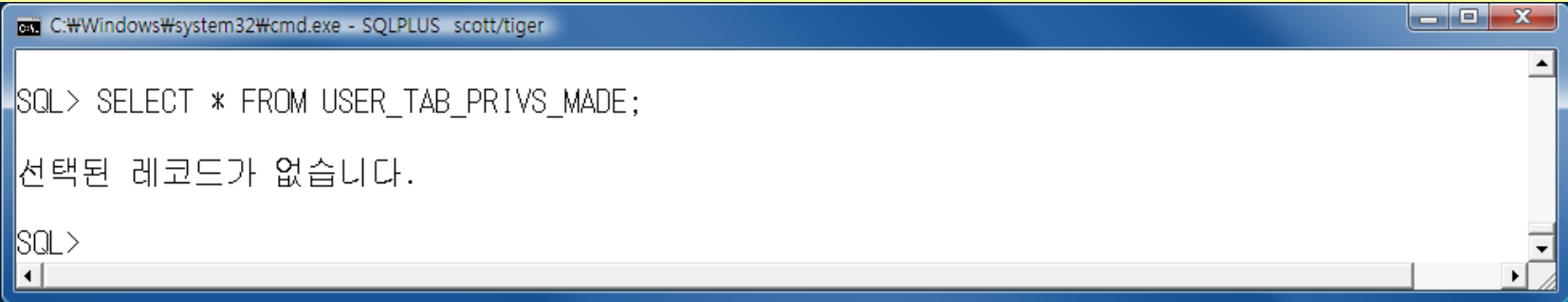
〈실습하기〉 객체 권한 제거하기

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

REVOKE SELECT ON EMP FROM USER01;

4. 권한이 철회되고 나면 데이터 디렉터리에 객체 권한에 대한 정보도 함께 사라집니다.

SELECT * FROM USER_TAB_PRIVS_MADE;



```
C:\Windows\system32\cmd.exe - SQLPLUS scott/tiger

SQL> SELECT * FROM USER_TAB_PRIVS_MADE;

선택된 레코드가 없습니다.

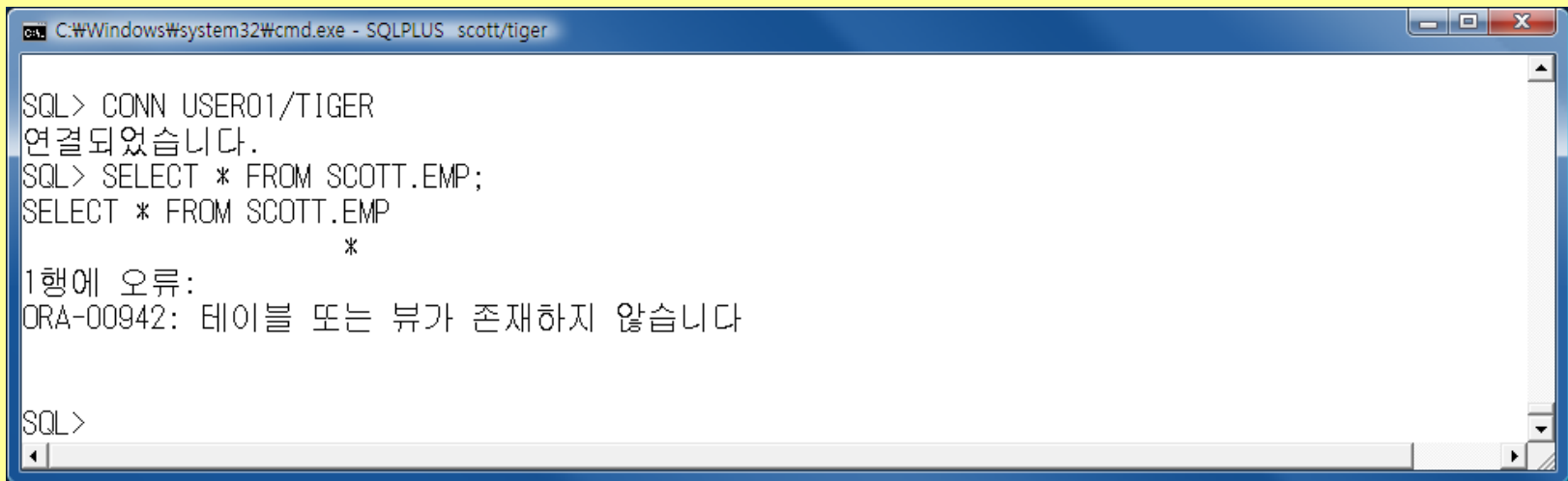
SQL>
```


〈실습하기〉 객체 권한 제거하기

5. USER01 사용자에게 부여된 EMP 테이블에 대한 SELECT 권한을 철회하였기에 USER01 사용자 계정으로 로그인해서 SCOTT 사용자의 EMP 테이블을 사용할 수 없습니다.

CONN USER01/TIGER

SELECT * FROM SCOTT.EMP;



```
C:\Windows\system32\cmd.exe - SQLPLUS scott/tiger

SQL> CONN USER01/TIGER
연결되었습니다.
SQL> SELECT * FROM SCOTT.EMP;
SELECT * FROM SCOTT.EMP
                *
1행에 오류:
ORA-00942: 테이블 또는 뷰가 존재하지 않습니다

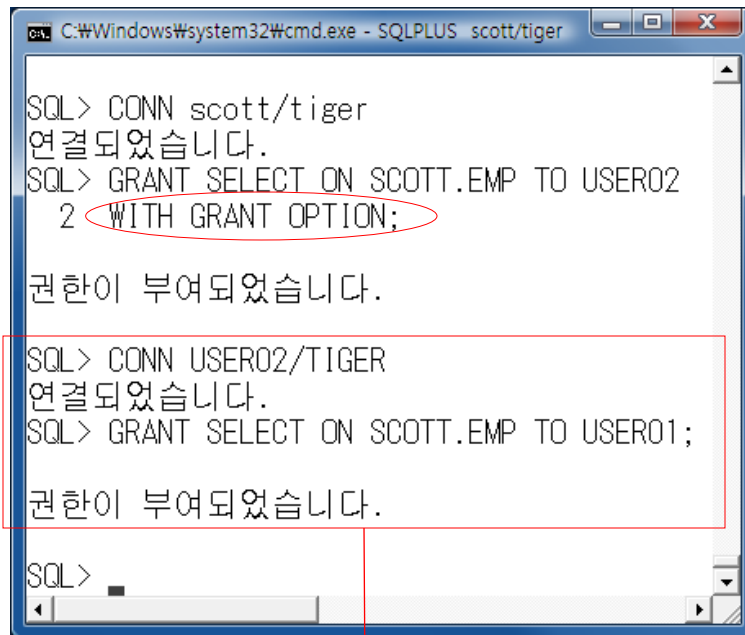
SQL>
```

4.4 WITH GRANT OPTION

- ❖ 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 부여받게 됩니다.
- ❖ 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 확인해 봅시다.
- ❖ USER02는 WITH GRANT OPTION을 지정하였기에 USER02로 로그인해서 객체권한을 또 다른 사용자에게 부여할 수 있습니다.

4.4 WITH GRANT OPTION

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



```
C:\Windows\system32\cmd.exe - SQLPLUS scott/tiger

SQL> CONN scott/tiger
연결되었습니다.
SQL> GRANT SELECT ON SCOTT.EMP TO USER02
2 WITH GRANT OPTION;

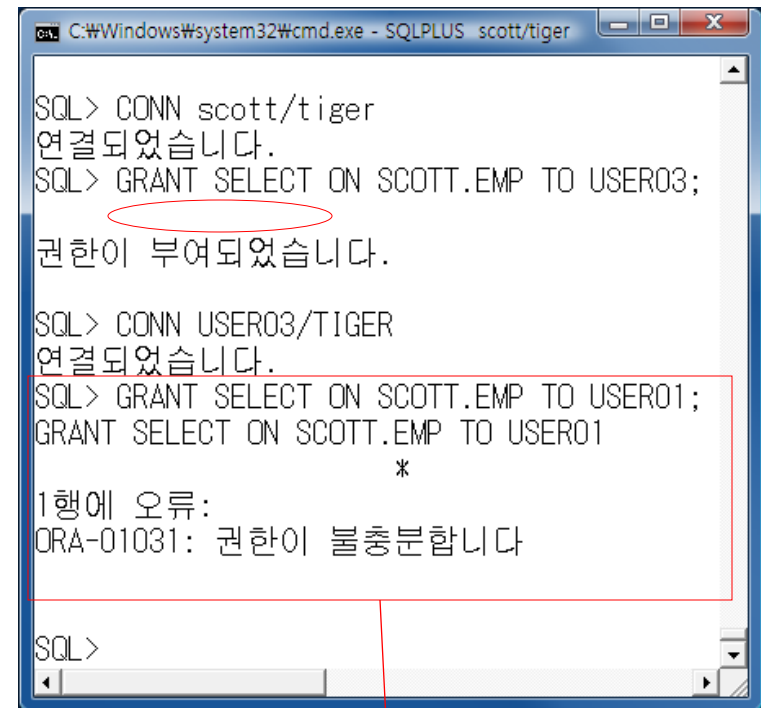
권한이 부여되었습니다.

SQL> CONN USER02/TIGER
연결되었습니다.
SQL> GRANT SELECT ON SCOTT.EMP TO USER01;

권한이 부여되었습니다.

SQL>
```

USER02는 WITH GRANT OPTION을 지정하였기에 USER02로 로그인 해서 객체 권한을 또 다른 사용자에게 부여할 수 있다.



```
C:\Windows\system32\cmd.exe - SQLPLUS scott/tiger

SQL> CONN scott/tiger
연결되었습니다.
SQL> GRANT SELECT ON SCOTT.EMP TO USER03;

권한이 부여되었습니다.

SQL> CONN USER03/TIGER
연결되었습니다.
SQL> GRANT SELECT ON SCOTT.EMP TO USER01;
GRANT SELECT ON SCOTT.EMP TO USER01
*
1행에 오류:
ORA-01031: 권한이 불충분합니다

SQL>
```

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

〈실습하기〉 WITH GRANT OPTION을 지정하여 객체 권한 부여하기

WITH GRANT OPTION을 지정하였기에 다른 사용자에게 객체권한을 부여할 수 있습니다.

1. SCOTT 사용자로 접속합니다.
2. USER02에게 SCOTT.EMP를 SELECT하는 권한을 WITH GRANT OPTION으로 부여합니다.

```
GRANT SELECT ON SCOTT.EMP TO USER02  
WITH GRANT OPTION;
```

3. USER02 사용자로 접속합니다.

```
CONN USER02/TIGER;
```

4. USER02 사용자가 자기가 받은 권한을 다른 사용자에게 부여할 수 있습니다

```
GRANT SELECT ON SCOTT.EMP TO USER01;
```

예제의 마지막 부분을 보면 USER02 사용자로 로그인하여 SCOTT.EMP 테이블을 SELECT 할 수 있는 권한을 USER01 사용자에게 부여하고 있습니다. 이것이 가능해진 것은 USER02에게 WITH GRANT OPTION을 사용하여 SCOTT.EMP 테이블 객체를 SELECT 할 수 있는 권한뿐만 아니라 그 권한을 다른 사용자에게도 부여할 수 있도록 허용하였기 때문입니다..

〈실습하기〉 WITH GRANT OPTION을 지정하지 않고 객체 권한 부여하기

WITH GRANT OPTION을 지정하지 않고 USER03 사용자가 단순히 SCOTT.EMP에 SELECT 할 수 있는 권한만을 부여하면 그 권한을 다른 사용자에게 부여할 수 없음을 확인해 봅시다.

1. SCOTT 사용자로 접속합니다.
2. 단순히 SCOTT.EMP에 SELECT 할 수 있는 권한만을 부여받습니다.

```
GRANT SELECT ON SCOTT.EMP TO USER03;
```

3. USER03 사용자로 접속합니다.

```
CONN USER03/TIGER;
```

4. USER02 사용자가 자기가 받은 권한을 다른 사용자에게 부여할 수 없습니다

```
GRANT SELECT ON SCOTT.EMP TO USER01;
```

〈탄탄히 다지기〉

사용자에게 시스템 권한을 ① _____과 함께 부여하면 그 사용자는 데이터베이스 관리자가 아닌데도 불구하고 부여받은 시스템 권한을 다른 사용자에게 부여할 수 있는 권한도 함께 부여 받게 됩니다.

사용자에게 객체 권한을 ② _____과 함께 부여하면 그 사용자는 그 객체를 접근할 권한을 부여 받으면서 그 권한을 다른 사용자에게 부여할 수 있는 권한도 함께 부여받게 됩니다.

Thank You !