1.계정 관련

1.1.계정 확인

SQL> SHOW USER;
USER은 "SYS"입니다.

지금 내가 사용한 계정이 뭔지 보여준다.

1.2.모든 계정을 확인

SQL>SELECT * FROM all_users;

1.3.SYS 계정으로 들어가기

SQL>SYS as sysdba
비밀번호 입력 : (그냥 엔터)

비밀번호가 필요없는 SYS 계정이다.

만약 DB가 여러개라서 다른 DB의 SYS 계정으로 접속하려고 하면 @다른DB_SID 를 추가해준다.
SQL>SYS@coreDB /as sysdba
비밀번호 입력 : (그냥 엔터)

1.4.계정 생성

SQL>CREATE USER scott IDENTIFIED BY tiger;

새로운 사용자인 scott를 생성한다.
비밀 번호는 tiger로 세팅한다. 따옴표(")는 꼭 붙여주자.
없어도 생성은 되는데, 나중에 대소문자 문제로 에러가 발생하는 경우도 있다.

그리고 계정 생성 후 바로 그 계정을 사용하려고 하면 없다고 나온다.
권한을 주고 사용.

1.5.다른 계정으로 넘어가기

SQL>conn scott/tiger;

다른 DB의 계정으로 넘어가는 것은 패스워드 뒤에 @다른DB_SID를 붙여준다.

SQL>conn scott/tiger@xe;

conn 대신 connect 로도 대체 가능하다.

1.6.계정 비밀번호 변경

```
SQL>ALTER USER scott IDENTIFIED BY tiger;
```

1.7.계정 삭제

```
SQL>DROP USER scott;
```

삭제
하지만 여기저기 문어발로 걸쳐놓은 것이 있는 한 많은 계정이라면 그냥 사라지지 않는다.
이때는 CASCADE를 사용하여 해당 사용자의 모든 SCHEMA를 삭제한 뒤에, user 를 삭제한다.

```
SQL>DROP USER scott CASCADE;
```

이렇게 꼼꼼하게 없애준다.

2.권한과 롤

(SYSTEM 계정으로만 가능)

2.1.접속 권한 주기

```
SQL>GRANT CONNECT, RESOURCE TO scott;
```

testuser에게 접속 권한을 준다. 물론 이런 중요한 역할은 SYSTEM 계정이 담당한다.
CONNECT와 RESOURCE에 어떤 권한이 걸려 있는지 확인하려면,

```
SQL>SELECT * DBA_SYS_PRIVS WHERE GRANTEE = 'CONNECT';
```

2.2.사용자에게 권한 주기

```
SQL>GRANT DELETE, INSERT, SELECT, UPDATE ON testdb TO scott;
```

scottr라는 사용자에게 testdb라는 데이터베이스의 권한을 준다.
이때, 권한의 종류 DELETE, SELECT 등은 관리자 마음대로 선택하여 줄 수 있다.

```
SQL>GRANT SELECT ON testdb TO testuser;
```

이런 식으로 testuser에서 testdb에서 권한을 select만 줄 수 있다.

이때 testuser는 select 이외의 create, delete 등의 동작은 할 수 없다.
우리가 월급 받은 만큼만 일하듯, 얘네도 권한을 받은 만큼 일을 한다.

2.3.DBA 권한 주기

```
SQL>GRANT DBA TO scott;
```

testuser에게 DBA 권한을 준다.

2.4.권한 취소

```
SQL>REVOKE CONNECT, RESOURCE FROM scott;
```

Connect와 resource 자리는 select 라든지의 값으로 대체 가능하다.

3.테이블 관련

3.1.모든 테이블 보기

```
SQL>SELECT * FROM TAB;
```

3.2.테이블 구조 확인하기

```
SQL>DESC testtable;
```

testtable의 구조를 확인할 수 있다.

3.3.테이블 생성

```
SQL>CREATE TABLE testtable (
    test_id NUMBER(10) NOT NULL,
    test_name VARCHAR2(10) NOT NULL,
    test_date DATE NOT NULL);
```

MySQL에서 ORACLE로 변경시,
int형은 number로,
varchar형은 varchar2로,
datetime형은 date로 변경한다.

3.4.코멘트 추가

```
SQL>COMMENT ON TABLE testtable IS 'This table is test table';
```

```
SQL>COMMENT ON COLUMN testtable.testfield IS 'This column is test column of testtabe';
```

table과 column일때 각각 코멘트를 추가할 수 있다. 코멘트 내용은 "로 묶는다.

3.5.테이블 변경 - 컬럼 추가

```
SQL>ALTER TABLE testtable ADD(testcolumn VARCHAR2(10));
```

컬럼(testcolumn)을 추가한다.

3.6.테이블 변경 - 컬럼 타입 변경

```
SQL>ALTER TABLE testtable MODIFY(column001 NUMBER);
```

column001 컬럼의 타입을 number로 변경한다.

3.7.테이블 변경 - 컬럼 삭제

```
SQL>ALTER TABLE testtable DROP(column001);
```

column001 컬럼을 삭제한다.

3.8.테이블 삭제

```
SQL>DROP TABLE testtable;
```

4.값 입출력

4.1.레코드 삽입

```
SQL>INSERT INTO testtable(test_id, test_name, test_date)
   VALUES(100, 'honggildong', sysdate);
```

4.2.레코드 수정

```
SQL>UPDATE testtable SET test_name = 'test001' WHERE test_id = '001';
```

4.3.레코드 확인

```
SQL>SELECT * FROM testtable;
```

*는 모든 열을 보여달라는 것이며, 몇몇개를 집어서 보여달라고 하고 싶으면,

```
SQL>SELECT test_id, test_name FROM testtable;

이라고 열 이름을 명시한다.
그리고 다른 조건을 추가하고 싶으면,

SQL>SELECT test_id FROM testtable WHERE test_name='test001';

이렇게 WHERE로 묶어준다.
```

4.4.레코드 삭제

```
SQL>DELETE FROM testtable WHERE test_name = 'test001';
```