[테이블생성]

create table 테이블명(컬럼명1 컬럼타입 [제약조건],
컬럼명2 컬럼타입 [제약조건],.....);

- 문자로 시작(30자이내) : 영문 대소문자, 숫자, 특수문자(_ , $, #), 한글
- 중복되는 이름은 사용 안됨
- 예약어(create, table, column등)은 사용할 수 없다

- 자료형
  number : number(전체자리, 소수이하), number → 숫자형(가변형)
  int : 정수형 숫자(고정형)
  varchar/varchar2 : 문자, 문자열(가변형) → 최대 4000byte
  char : 문자, 문자열(고정형) → 2000byte
  date : 날짜형
  clob : 문자열 → 최대4GB
  blob : 바이너리형(그림, 음악, 동영상..) → 최대4GB

- 제약조건
  not null : 해당 컬럼에 NULL을 포함되지 않도록 함 (컬럼)
  unique : 해당 컬럼 또는 컬럼 조합 값이 유일하도록 함 (컬럼, 테이블)
  primary key : 각 행을 유일하게 식별할 수 있도록 함 (컬럼, 테이블)
  references table(column) :
  해당 컬럼이 참조하고 있는 (부모)테이블의 특정 (컬럼, 테이블) 컬럼 값들과 일치하거나 또는 NULL이 되도록 보장함

  check : 해당 컬럼에 특정 조건을 항상 만족시키도록 함 (컬럼, 테이블)
  [참고] primary key = unique + not null

ex)	idx	일련번호	primary key
	id	아이디	unique
	name	이름	not null
	phone	전화번호	
	address	주소	
	score	점수	check
	subject_code	과목코드	
	hire_date	입학일	기본값 (오늘날짜)
	marriage	결혼	check

- 제약조건 확인
  constraint_name:이름
  constraint_type:유형
    P : primary key
    U : unique
    R : reference
    C : check, not null

  search_condition : check조건 내용
  r_constraint_name : 참조테이블의 primary key 이름
  delete_rule : 참조테이블의 primary key 컬럼이 삭제될 때 적용되는 규칙
    (no action, set null, cascade등)

- 삭제 RULE
on delete cascade
: 대상 데이터를 삭제하고, 해당 데이터를 참조하는 데이터도 삭제
on delete set null
: 대상 데이터를 삭제하고, 해당 데이터를 참조하는 데이터는 NULL로 바꿈
on delete restricted
: 삭제대상 데이터를 참조하는 데이터가 존재하면 삭제할 수 없음 (기본값)

- 수정 RULE
on update cascade
: 대상 데이터를 수정하면, 해당 데이터를 참조하는 데이터도 수정

[테이블수정]

- 구문 -
alter table 테이블명
  add 컬럼명 데이터타입 [제약조건]
  add constraint 제약조건명 제약조건타입(컬럼명)
  modify 컬럼명 데이터타입
  drop column 컬럼명 [cascade constraints]
  drop primary key [cascade] | union (컬럼명,.....) [cascade] .... | constraint 제약조건명 [cascade]

- 이름 변경 -
alter table 기존테이블명 rename to 새테이블명
alter table 테이블명 rename column 기존컬럼명 to 새컬럼명
alter table 테이블명 rename constraint 기존제약조건명 to 새제약조건명

[테이블복사]

- 서브쿼리를 이용한 테이블 생성 및 행(레코드) 복사
- 서브쿼리를 이용해서 복사한 경우 not null을 제외한 제약조건은 복사 안됨
  (not null제약조건도 sys_*****로 복사됨)

- 구문 -

create table 테이블명([컬럼명1,컬럼명2.....]) as 서브쿼리

- 구조만 복사 -

create table 테이블명1 as select * from 테이블명2 where 1=0

[시퀀스]

: 순차적으로 정수값을 자동으로 생성하는 객체

create sequence 시퀀스명
[increment by 증가값] [start with 시작값]
[maxvalue 최대값 | minvalue 최소값]
[cycle | nocycle]
[cache | nocache]

- increment by 증가값 : 증가/감소 간격(기본값 : 1)
- start with : 시작번호(기본값 : 1)
- maxvalue / minvalue : 시퀀스의 최대/최소값지정
- cycle/nocycle : 최대/최소값에 도달 시 반복여부결정
- cache / nocache : 지정한 수량만큼 메모리 생성여부결정 (최소값:2, 기본값:20)

[insert]

: 테이블에 데이터(새로운 행)추가

insert into 테이블명 [(column1, column2, ...)] values (value1,value2,.....)

- column과 values의 순서일치
- column과 values의 개수 일치

[update]

: 테이블에 포함된 기존 데이터수정
전체 데이터 건수(행수)는 달라지지 않음
조건에 맞는 행(또는 열)의 컬럼값을 갱신할 수 있다

update 테이블명 set 컬럼명1=value1, 컬럼명2=value2 ..... [where 조건절]

- where 이 생략이 되면 전체행이 갱신
- set절은 서브쿼리 사용가능, default옵션 사용가능

[delete]

: 테이블에 포함된 기존데이터를 삭제
  행 단위로 삭제되므로 전체행수가 달라짐

delete [from] 테이블명 [where 조건절];
- where을 생략하면 전체행이 삭제됨
- 데이터는 삭제되고 테이블 구조는 유지됨

[truncate]

: 테이블의 데이터를 전부 삭제하고 사용하고 있던 공간을 반납
  해당 테이블의 데이터가 모두 삭제되지만 테이블 자체가 지워지는 것은 아님
  해당 테이블에 생성되어 있던 인덱스도 함께 truncate 됨

TRUNCATE TABLE 테이블명;

DELETE 후에는 데이터만 지워지고 쓰고 있던 디스크 상의 공간은 그대로 가지고 있다.

TRUNCATE 작업은 최초 테이블이 만들어졌던 상태, 즉 데이터가 1건도 없는 상태로 모든 데이터 삭제, 칼럼 값만 남아 있다.
그리고 용량도 줄고 인덱스 등도 모두 삭제된다.

→ DELETE보다 TRUNCATE가 더 좋아 보이나 DELETE는 where를 이용하여 원하는 데이터만 골라서 삭제가 가능하나 TRUNCATE는 조건을 사용할 수 없다.
또 DDL이기 때문에 사용권한 문제도 있다.

DROP 명령어는 데이터와 테이블 전체를 삭제하게 되고 사용하고 있던 공간도 모두 반납하고 인덱스나 제약조건 등 오브젝트로 삭제된다.

[transaction처리]

: 일의 시작과 끝이 완벽하게 마무리(commit)
  처리도중 인터럽트(interrupt:장애)가 발생하면 되돌아옴(rollback)

ex1) 테이블 : test

```
create table test(
id number(5),
name char(10),
address varchar2(50));
```

ex2) 테이블 : user1

```
create table user1(
idx       number   primary key,
id        varchar2(10) unique,
name      varchar2(10) not null,
phone     varchar2(15),
address varchar2(50),
score     number(6,2)   check(score >= 0 and score <= 100),
subject_code   number(5),
hire_date   date default sysdate,
marriage    char(1)   default 'N'   check(marriage in('Y', 'N')));
```

ex3) 제약조건 확인

```
select constraint_name, constraint_type
from user_constraints
where table_name='USER1';

select *
from user_constraints
where table_name='USER1';
```

ex4) 테이블

```
create table user2(
idx       number          constraint PKIDX primary key,
id        varchar2(10)   constraint UNID unique,
name      varchar2(10)   constraint NOTNAME not null,
phone     varchar2(15),
address varchar2(50),
score     number(6,2)    constraint CKSCORE check(score >= 0 and score <= 100),
subject_code   number(5),
hire_date   date default sysdate,
marriage    char(1)   default 'N' constraint CKMARR check(marriage in('Y','N')));
```

ex5) 제약조건 확인

```
select constraint_name, constraint_type
from user_constraints
where table_name='USER2';

select *
from user_constraints
where table_name='USER2';
```

ex6) 추가

```
insert into user1(idx,id,name,phone,address,score,subject_code,hire_date,marriage)
values(1,'aaa','kim','010-000-0000','서울',75,100,'2010-08-01','Y');

insert into user1(idx,id,name,phone,address,score,subject_code,hire_date,marriage)
values(1,'aaa','kim','010-000-0000','서울',75,100,'2010-08-01','Y');
```

→ 무결성제약조건에 위배(이유: idx 1 중복)

```
insert into user1(idx,id,name,phone,address,score,subject_code,hire_date,marriage)
values(2,'aaa','kim','010-000-0000','서울',75,100,'2010-08-01','Y');
```

→ 무결성제약조건에 위배(이유: id aaa 중복)

```
insert into user1(idx,id,name,phone,address,score,subject_code,hire_date,marriage)
values(2,'bbb','','010-000-0000','서울',75,100,'2010-08-01','Y');
```

→ NULL을 ("HR"."USER1"."NAME") 안에 삽입할 수 없습니다

```
insert into user1(idx,id,name,phone,address,score,subject_code,hire_date,marriage)
values(2,'bbb','lee','010-000-0000','서울',120,100,'2010-08-01','Y');
```

→ 체크 제약조건에 위배되었습니다(이유: score가 0~100사이의 수 이어야함)

```
insert into user1(idx,id,name,phone,address,score,subject_code,hire_date,marriage)
values(2,'bbb','lee','010-000-0000','서울',75,100,'2010-08-01','K');
```

→ 체크 제약조건에 위배되었습니다(이유 : marriage가 Y 또는 N 이어야함)

```
insert into user1(idx,id,name,phone,address,score,subject_code,hire_date,marriage)
values(2,'bbb','lee','010-000-0000','서울',75,100,'2010-08-01','N');
```

ex7) 테이블 목록 확인

```
select * from tab;
```

ex8) 테이블의 레코드(내용) 확인

```
select * from user1;
select * from user2;
```

ex9) 테이블의 구조 확인

```
desc user1; (= describe user1)
```

ex10) 시퀀스 목록 확인

```
select * from user_sequences;
```

ex11) 테이블명 변경 : test → user3

```
alter table test rename to user3;
```

ex12) 컬럼 추가 : user3 → phone varchar2(15)

```
alter table user3 add phone varchar2(15);
desc user3;
```

ex13) 제약조건 추가 : user3 → id에 unique, 제약조건명 UID2

```
alter table user3 add constraint UID2 unique(id);

select constraint_name, constraint_type
from user_constraints
where table_name='USER3';
```

ex14) 제약조건 삭제 - alter table 테이블명 drop constraint 제약조건명;

```
alter table user3 drop constraint UID2;
alter table user3 DROP constraint SYS_C007693;

select *
from user_constraints
where table_name='USER3';
```

ex15) 컬럼 추가 : user3 → no number (PK 설정)

```
alter table user3 add no number primary key;
desc user3;
```

ex16) 구조 변경 : user3 → name char(10) 를 varchar2(10)로 바꿈

```
alter table user3 modify name varchar2(10);
desc user3;
```

ex17) 컬럼 삭제 : user3 → address

```
alter table user3 drop column address;
desc user3;
```

ex18) 테이블삭제 / 휴지통비우기: user3

```
drop table user3;
select * from tab;
purge recyclebin; -- 휴지통 비우기

drop table user1 purge; -- 휴지통에 넣지 않고 바로 삭제
select * from tab;

drop table user2;
select * from tab;

show recyclebin; -- 휴지통 보기

flashback table user2 to before drop; -- 휴지통에서 되살리기
flashback table "BIN$cEf2dC1fRhilpiULWNRf3A==$0" to before drop;

select * from recyclebin; -- 휴지통에 테이블 정보 검색
```

ex19) 시퀀스 생성 / 삭제

```
create sequence idx_sql increment by 2 start with 1 maxvalue 9 cycle nocache;

select idx_sql.nextval from dual; -- 다음 시퀀스값 표시(nextval)
select idx_sql.currval from dual; -- 현재 시퀀스값 표시(currtval)

select * from user_sequences;

drop sequence idx_sql;
select * from user_sequences;
```

ex20) 테이블 생성과 시퀀스 적용

```
create  table  book(
no  number primary key,
subject  varchar2(50),
price number,
year date);

create sequence no_seq  increment by 1 start with 1 nocycle nocache;
```

```
insert into book(no, subject, price, year)
values(no_seq.nextval, '오라클 무작정 따라하기', 10000, sysdate);

insert into book(no, subject, price, year)
values(no_seq.nextval, '자바 3일 완성', 15000, '2007-03-01');

insert into book values(no_seq.nextval, 'JSP 달인 되기', 18000, '2010-01-01');

select * from book;
```

ex21) 테이블 구조만 복사

```
create table user3 as select * from user2 where 1=0;
desc user3;
```

USER2에는 제약조건이 5개가 보인다

```
select constraint_name, constraint_type, search_condition
from user_constraints
where table_name='USER2';

select constraint_name, constraint_type, search_condition
from user_constraints
where table_name='USER3';
```

← not null을 제외하고는 제약조건이 복사 안됨

← not null 제약조건도 sys_*****로 복사됨 (제약조건명 그대로 복사가 안된다)

ex22) 테이블(idx → bunho, name → irum, address → juso) 을 복사하고 id가 bbb인 레코드를 복사하시오

원본테이블 : user1 / 사본테이블 : user4

```
create table user4(bunho, irum, juso)
as select idx, name, address from user1 where id='bbb';

select * from user1;
select * from user4;
```

ex23) 테이블 생성 후 행 추가

테이블명 : dept

deptno number → 기본키, 제약조건명(DNO)

dname varcahr2(30) → 널 허용 안됨, 제약조건명(DNAME)

테이블명 : emp
empno number → 기본키, 제약조건명(ENO)
ename varchar2(30) → 널 허용 안됨, 제약조건명(ENAME)
deptno number → 외래키, 제약조건명(FKNO),
대상데이터를 삭제하고 참조하는 데이터는 NULL로 바꿈

```
create table dept(
deptno number constraint DNO primary key,
dname varchar2(30) constraint DNAME not null);

create table emp(
empno number constraint ENO primary key,
ename varchar2(30) constraint ENAME not null,
deptno number,
constraint FKNO foreign key(deptno) references dept on delete set null);

insert into dept(deptno, dname) values(10, '개발부');
insert into dept(deptno, dname) values(20, '영업부');
insert into dept(deptno, dname) values(30, '관리부');
insert into dept(dname) values(40, '경리부'); → ORA-00913: 값의 수가 너무 많습니다.
insert into dept(deptno, dname) values(40, '경리부');

insert into emp(empno, ename, deptno) values(100, '홍길동', 10);
insert into emp(empno, ename, deptno) values(101, '라이언', 20);
insert into emp(empno, ename, deptno) values(102, '튜브', 50);
→ 50번부서 없음(무결성제약조건위배) - 부모키가 없습니다

insert into emp(empno, ename, deptno) values(103, '어피치', 40);
insert into emp(empno, ename) values(105, '프로도');
insert into emp(ename, deptno) values('콘', 10); → primary key는 NULL허용 안함

commit;
```

ex24) 삭제

```
delete from dept;
select * from dept;
rollback;
select * from dept;

delete from dept where deptno=40;
select * from dept; -- 40번 부서 삭제
```

```
select * from emp; -- 40번 부서 컬럼에 (null)이 들어간다.
```

ex25) 수정(update) - emp테이블 장동건 사원의 부서번호를 30으로 수정하시오

```
update emp set deptno=30 where ename='프로도';
select * from emp;
commit;
```