ORACLE

Introduction to Oracle
: SQL

ORACLE

Chapter8.
테이블 생성 및 관리

- 테이블을 생성하는 방법을 터득한다.
- 컬럼에 제약조건을 부여하여 데이터의 무결성을 유지한다.

※ 데이터베이스 객체

객체	의미
테이블	행과 열로 구성된 기본적인 저장 구조
뷰	하나 이상의 테이블에서 데이터의 부분집합 을 논리적으로 표현
시퀀스	고유한 번호를 자동으로 발생시키는 객체로 주로 PK 값 생성에 사용
인덱스	질의(SELECT) 성능을 향상시키기 위하여 사용하는 물리적인 저장 구조
시노님	객체에 대한 이름을 부여

1. 테이블 생성

(1) 구문

```
CREATE TABLE [schema.]table_name
      ( column        datatype    [DEFAULT expr] [column_constraint],
      . . . . . .
       [table_constraint]);
```

- schema : 테이블의 소유자
- datatype : 열의 자료형
- DEFAULT : INSERT 중 값을 생략 시 입력되는 값 명시
- column_constraint : 컬럼 레벨로 무결성 제약 조건 기술
- table_constraint : 테이블 레벨로 무결성 제약 조건 기술

1. 테이블 생성

(2) 이름 지정 규칙

- 문자로 시작
- 문자의 길이는 1 ~ 300이내를 사용
- 오직 A ~ Z, a ~ z, 0 ~ 9, _, $, # 만 사용 가능
  단, 한글 데이터베이스에서는 한글 사용 가능
- 동일한 사용자가 소유한 객체 이름은 중복 불가
- 예약어는 사용 불가

1. 테이블 생성

(3) 데이터 타입

데이터타입	설 명
VARCHAR2(n)	가변 길이 문자 데이터(1~4000byte)
CHAR(n)	고정 길이 문자 데이터(1~2000byte)
NUMBER(p,s)	전체 p자리 중 소수점 이하 s자리
DATE	날짜 및 시간 데이터
LONG	가변 길이 문자 데이터(1~2Gbyte)
CLOB	단일 바이트 가변 길이 문자 데이터(1~4Gbyte)
RAW(n)	n Byte의 원시 이진 데이터(1~2000)
LONG RAW	가변 길이 원시 이진 데이터(1~2Gbyte)
BLOB	가변 길이 이진 데이터(1~4Gbyte)
BFILE	가변 길이 외부 파일 이진 데이터(1~4Gbyte)

1. 테이블 생성

(4) DEFAULT 옵션

- 데이터 삽입 시 열에 대한 기본값 명시
- 기술 가능한 값은 문자값, 표현식, SQL 함수
- 다른 컬럼 이름이나 의사열(NEXTVAL, CURRVAL등)은 사용 불가
- DEFAULT 값의 데이터 타입은 열의 데이터 타입과 일치

1. 테이블 생성

(5) 제약조건

1) 제약조건의 정의

- 테이블 레벨에서 규칙을 적용
- 종속성이 존재할 경우 테이블 삭제를 방지
- 데이터가 삽입, 갱신, 삭제될 때마다 규칙 적용

1. 테이블 생성

(5) 제약조건

2) 제약조건의 정의 방법

- 컬럼 레벨

```
컬럼명  데이터타입 [CONSTRAINT 제약조건명] 제약조건 유형
```

- 테이블 레벨

```
컬럼명  데이터타입,
[CONSTRAINT 제약조건명] 제약조건 유형 (컬럼명1[,컬럼명2,.....])
```

1. 테이블 생성

(5) 제약조건

3) 제약조건의 종류

제약조건	설 명
PRIMARY KEY (PK)	유일하게 테이블의 각행을 식별 (NOT NULL과 UNIQUE 조건을 만족)
FOREIGN KEY (FK)	다른 컬럼값을 참조하는 관계
UNIQUE (UK)	유일한 값만 입력을 허용 (NULL값 허용)
NOT NULL (NN)	NULL값을 허용하지 않음
CHECK (CK)	컬럼값이 만족해야 하는 조건 지정

1. 테이블 생성

(5) 제약조건

4) 제약조건 부여

- PRIMARY KEY

```
CREATE TABLE test_tab2(
            id        NUMBER(2),
            name      VARCHAR2(10),
            CONSTRAINT id_pk PRIMARY KEY (id) ) ;
```

- FOREIGN KEY

```
CREATE TABLE emp_tab (
            empno         NUMBER(4),
            ename         VARCHAR2(10),
            deptno        NUMBER(2) NOT NULL,
            CONSTRAINT emp_deptno_fk FOREIGN KEY (deptno)
            REFERENCES dept (deptno) ) ;
```

1. 테이블 생성

(5) 제약조건

4) 제약조건 부여

- **UNIQUE**

```
CREATE TABLE uni_tab2 (
            deptno          NUMBER(2),
            dname           CHAR(14),
            CONSTRAINT uni_tab_deptno_uk UNIQUE (deptno) ) ;
```

- **NOT NULL**

```
CREATE TABLE nn_tab1 (
            deptno          NUMBER(2)  NOT NULL ,
            dname           CHAR(14),
            loc             CHAR(13)) ;
```

1. 테이블 생성

(5) 제약조건

4) 제약조건 부여

- **CHECK**

```
CREATE TABLE ck_tab2 (
            deptno          NUMBER(2),
            dname           CHAR(14),
            loc             CHAR(13),
            CONSTRAINT uni_tab_deptno_ck CHECK (deptno IN (10,20,30,40,50))) ;
```

1. 테이블 생성

(5) 제약조건

5) 제약조건 추가

```
ALTER TABLE  테이블명
ADD  [CONSTRAINT 제약조건명]  제약조건 유형 (컬럼명);
```

```
ALTER TABLE emp
ADD CONSTRAINT emp_ename_uk UNIQUE (ename);
```

6) 제약조건 삭제

```
ALTER TABLE  테이블명
DROP  CONSTRAINT  제약조건명 [CASCADE];
```

```
ALTER TABLE emp
DROP CONSTRAINT emp_ename_uk;
```

1. 테이블 생성

(5) 제약조건

7) 제약조건 비활성화

```
ALTER TABLE  테이블명
DISABLE CONSTRAINT 제약조건명 [CASCADE];
```

```
ALTER TABLE emp
DISABLE CONSTRAINT emp_primary_key;
```

8) 제약조건 활성화

```
ALTER TABLE  테이블명
ENABLE CONSTRAINT 제약조건명 ;
```

```
ALTER TABLE emp
ENABLE CONSTRAINT emp_primary_key ;
```

1. 테이블 생성

(5) 제약조건

9) 제약조건 조회

```
SELECT constraint_name,constraint_type,search_condition
FROM user_constraints
WHERE table_name = 'EMP';
```

```
SELECT constraint_name, column_name
FROM user_cons_columns
WHERE table_name = 'EMP';
```

1. 테이블 생성

(6) 테이블 생성

1) 일반적 테이블 생성

```
CREATE TABLE post(
            post1       CHAR(3),
            post2       CHAR(3),
            addr        VARCHAR2(60) CONSTRAINT post_addr_nn NOT NULL,
            CONSTRAINT post_post12_pk PRIMARY KEY (post1,post2));
```

1. 테이블 생성

(6) 테이블 생성

2) 서브쿼리를 사용한 테이블 생성

```
CREATE TABLE 테이블명 [컬럼명1[, 컬럼명2, . . . . . .]]
  AS 서브쿼리
```

```
CREATE TABLE emp_30
            AS
            SELECT employee_id,last_name, salary
            FROM emp
            WHERE department_id = 30 ;
```

2. 테이블 수정

(1) 새로운 컬럼 추가

```
ALTER TABLE  테이블명
ADD  (컬럼명1 데이터타입 [DEFAULT expr]
      [,컬럼명2 데이터타입 [DEFAULT expr] . . . . . .]
```

```
ALTER TABLE  bonus
ADD              (etc VARCHAR2(50));
```

2. 테이블 수정

(2) 컬럼 수정

```
ALTER TABLE  테이블명
MODIFY  (컬럼명1 데이터타입 [DEFAULT expr]
         [, 컬럼명2 데이터타입 [DEFAULT expr] . . . . . .]
```

```
ALTER  TABLE  emp_30
MODIFY        (last_name VARCHAR2(15)) ;
```

- 컬럼의 사이즈 증가 가능
- 열이 모두 NULL이거나 데이터가 없으면 사이즈 감소 가능
- DEFAULT 값을 변경 내용은 이후 INSERT문에만 영향

2. 테이블 수정

(3) 컬럼 삭제

```
ALTER TABLE   테이블명
DROP          (컬럼명) ;
```

```
ALTER TABLE   emp_30
DROP           (last_name) ;
```

3. 테이블 삭제

(1) 테이블 삭제

- 테이블의 모든 구조와 데이터가 삭제
- 완료되지 않은 트랜젝션은 COMMIT
- 모든 인덱스가 삭제
- 수행 후 ROLLBACK 불가

(2) 테이블 삭제 구문

```
DROP TABLE 테이블명 [CASCADE CONSTRAINT]
```

```
DROP TABLE emp_30 ;
```

4. 객체 이름 변경

(1) RENAME

```
RENAME 기존 객체명 TO 바꿀 객체명 ;
```

```
RENAME emp_30 TO emp_temp30;
```

5. TRUNCATE문

(1) TRUNCATE

- 테이블의 모든 행을 삭제
- 테이블의 데이터 저장 영역을 반납
- 수행 후 ROLLBACK 불가

(2) TRUNCATE 구문

```
TRUNCATE TABLE 테이블명 ;
```

```
TRUNCATE TABLE emp_temp30 ;
```

6. 주석

(1) COMMENT 구문

```
COMMENT ON TABLE 테이블명 | COLUMN 테이블명.컬럼명  IS  'text' ;
```

```
COMMENT ON TABLE emp  IS 'Employee Information' ;
```

(2) 주석 확인

- ALL_COL_COMMENTS
- USER_COL_COMMENTS
- ALL_TAB_COMMENTS
- USER_TAB_COMMENTS