2장 스키마 객체

10. ALTER TABLE 문장

- 다음에 **ALTER TABLE** 문장 사용**.**
  - 새로운 열 추가
  - 기존 열 변경
  - 새로운 열에 대한 디폴트 값 정의

```
ALTER TABLE table
ADD         (column datatype [DEFAULT expr]
             [ , column datatype] …….);
```

```
ALTER TABLE table
MODIFY      (column datatype [DEFAULT expr]
             [ , column datatype] …….);
```

11. 열 추가

- **ADD** 절을 사용하여 열을 추가함.

```
SQL> ALTER  TABLE  datp30
  2   ADD (   job  VARCHAR2(9)  ) ;
Table  altered.
```

- 새로운 열이 마지막 열이 됨.

```
 EMPNO     ENAME        ANNSAL       HIREDATE       JOB
-------  ----------  ----------  -----------------  ---------
  7798    BLAKE          34200     01-MAY-81
  7654    MARTIN         15000     28-SEP-81
  7499    ALLEN          19200     20-FEB-81
  7844    TURNER         18000     08-SEP-81
…
6 rows selected.
```

12. 열 수정

- 열의 데이터형, 크기, 디폴트 값을 변경.

```
ALTER  TABLE  datp30
MODIFY          (ename  VARCHAR2(15));
 Table  altered.
```

- 숫자 열의 정밀도나 폭을 증가시킴.
- 열이 오직 null 값만을 포함하거나 테이블에 행이 없으면 열의 폭을 감소시킴.
- 열이 null 값을 포함하면 데이터형을 변경시킴.
- 열이 null 값을 포함하거나 크기를 변경하지 않으면 CHAR 열을 VARCHAR2 데이터형으로 변경하거나 VARCHAR2 열을 CHAR 데이터형으로 변경.
- 열의 디폴트 값을 변경시키는 것은 오직 테이블에 가해지는 이후의 삽입에만 영향을 미침.

13. 테이블 삭제(1)

- 테이블의 모든 데이터와 구조가 삭제.
- 어떤 결정되지 않은 트랜잭션이 커밋.
- 모든 인덱스가 삭제.
- 이 문장은 롤백할 수 없음.

```
SQL> DROP TABLE datp30;
Table dropped.
```

13. 테이블 삭제(2)

- **TRUNCATE TABLE 문장:**
  - 테이블의 모든 행을 삭제.
  - 해당 테이블에 사용한 기억 공간을 해제.

```
SQL>TRUNCATE  TABLE  dapartment;
Table  truncated.
```

- TRUNCATE를 사용하여 삭제한 행을 롤백 할 수 없음.
- 대안적으로 DELETE 문장을 사용하여 행을 삭제함.

  - DELETE 문장은 테이블의 모든 행을 삭제할 수 있지만, 저장공간을 해제할 수는 없다.

14. 객체 이름 변경

- 테이블 이름, 뷰, 시퀀스 또는 동의어를 변경하기 위해, RENAME 문장을 실행.

```
SQL> RENAME  dept  TO  dapartment;
Table  renamed.
```

- 객체의 소유자 이어야 함.

15. 테이블에 주석문 추가

- COMMENT 문장을 사용하여 테이블이나 열에 주석문을 추가할 수 있음.

```
SQL> COMMENT ON TABLE  emp
  2   IS  'Employee  Information';
Comment  created.
```

- 주석문은 데이터 뷰를 통해서 볼 수 있음.

  - ALL_COL_COMMENTS
  - USER_COL_COMMENTS
  - ALL_TAB_COMMENTS
  - USER_TAB_COMMENTS

제약 조건 추가

```
ALTER TABLE table
ADD [constraint constraint] type (column);
```

- **Add** 절을 가지는 **ALTER TABLE** 문장을 사용하여 기존의 테이블에 대한 제약조건을 추가할 수 있음
- 제약조건의 추가 또는 삭제가능, 수정은 불가능
- 제약조건의 활성화 또는 비활성화
- **MODIFY** 절을 사용하여 **NOT NULL** 제약조건을 추가

제약조건 추가

- **EMP** 테이블 안에 유효한 종업원으로 이미 존재해야 하는 관리자를 나타내는 **EMP** 테이블에 대하여 **FOREIGN key** 제약조건을 추가할 수 있다.

```
SQL>  ALTER TABLE       emp
  2       ADD CONSTRAINT emp_mgr_fk
  3              FOREIGN key(mgr) REPERENCES emp(empno);
Table altered.
```

제약조건 삭제

- 제약조건을 삭제하기 위해서 **USER_CONSTRAINTS** 와 **USER_CONS_COLUMNS** 데이터 사전 뷰에서 제약조건 이름을 식별할 수 있음.
- **DROP** 절과 **ALTER TABLE** 문장을 사용
- **DROP**의 **CASCADE** 옵션은 모든 종속적인 제약조건이 삭제되게 함

```
ALTER   TABLE table
DROP    PRIMARY KEY | UNIQUE (column) |
        CONSTRAINT constraint [CASCADE];
```

- Table : 테이블의 이름 - column : 제약조건에 의해 영향 받은 이름 - constraint : 제약조건의 이름

- 무결성 제약조건을 삭제할 때, 그 제약조건은 더 이상 오라클 서버에 의해 적용되지 않으며 데이터 사전에서 찾을 수 있다.

제약조건 사용불가

- 무결성 제약조건을 비활성화 하기 위해 **ALTER TABLE** 문장의 **DISABLE** 절을 실행
- 종속적인 무결성 제약조건을 비 활성화 하기위해서 **CASCADE** 옵션을 적용함

```
ALTER   TABLE table
DISABLE  CONSTRAINT constraint [CASCADE];
```

- Table : 테이블의 이름 - constraint : 제약조건의 이름

- CREATE TABLE 문장과 ALTER TABLE 문장으로 DISABLE 절을 사용할 수 있다.
- CASCADE 절은 종속적인 무결성 제약조건을 비활성화 함

제약조건의 사용가능

- **ENABLE 절을 가진 ALTER TABLE 문장을 사용하여 삭제 또는 재 생성 없이 제약조건을 활성화 할 수 있음**

```
ALTER   TABLE table
ENABLE    CONSTRAINT constraint [CASCADE];
```

- Table : 테이블의 이름 - constraint : 제약조건의 이름

- 제약조건이 활성화 되면 그 제약조건은 테이블의 모든 데이터에 대해 적용됨
- 테이블의 모든 데이터는 제약조건과 일치해야 함
- UNIQUE key 또는 PRIMARY key 제약조건이 활성화 된다면, UNIQUE 또는 PRIMARY key 인덱스는 자동으로 생성됨.
- CREATE TABLE 문장과 ALTER TABLE 문장으로 ENABLE 절을 사용할 수 있다.

제약조건 보기

- 모든 제약조건의 정의와 이름을 보기위해 USER_CONSTRAINTS 테이블을 질의함

```
SQL>    SELECT          constraint_name, constraint_type,
  2                     search_condition
  3     FROM            user_constraints
  4     WHERE           table_name = 'EMP';
```

```
CONSTRAINT_NAME                C   SEARCH_CONDITION
------------------------------ - ------------------------------
SYS_C00674                     C   EMPNO IS NOT NULL
SYS_C00675                     C   DEPTNO IS NOT NULL
EMP_EMPNO_PK                   P
…
```

제약조건과 연관된 열 보기

- USER_CONS_COLUMNS 뷰에서 제약조건 이름과 관련된 열을 봄

```
SQL>    SELECT          constraint_name, column_name
  2     FROM            user_con_columns
  3     WHERE           table_name = 'EMP';
```

```
CONSTRAINT_NAME                COLUMN_NAME
------------------------------ --------------------------------
EMP_DEPTNO_FK                  DEPTNO
EMP_EMPNO_PK                   EMPNO
EMP_MGR_FK                     MGR
SYS_C00674                     EMPNO
SYS_C00675                     DEPTNO
```

인덱스 제거

- 데이터 사전에서 인덱스를 제거

```
SQL> DROP INDEX index;
```

- 데이터 사전에서 **EMP_ENAME_IDX** 인덱스를 제거

- 인덱스를 제거하기위해, 인덱스의 **DROP ANY INDEX** 권한을 갖고 있어야 함

```
SQL> DROP INDEX emp_ename_idx;
```

외부 테이블

- 외부 테이블이란 읽기 전용 테이블로서 데이터가 데이터베이스가 아닌 외부 파일에 저장되는 것을 말합니다.
- 외부 테이블의 데이터는 **SQL**을 이용하여 조회할 수 있습니다. 그러나 갱신, 삽입, 그리고 삭제는 허용되지 않으며 인덱스 역시 생성할 수 없습니다.
- 외부 테이블에 대한 메타데이터는 **CREATE TABLE** 명령에 의해 생성됩니다.
- 외부 테이블의 정의가 외부 데이터가 데이터베이스에 대해 어떻게 보여지는 지를 결정합니다.

외부 테이블의 이점

- 외부 데이터를 데이터베이스 안으로 적재하지 않더라도 조회하거나 다른 테이블과 죠인할 수 있습니다.
- 데이터 웨어하우스 구축을 위한 별도의 중간 저장소를 데이터베이스 내에 만들 필요가 없습니다.
- 특히 다음의 경우에 유용합니다.
  - **ETL** 과정에서 외부 데이터가 데이터베이스 내의 오브젝트와 죠인된 후 그 결과를 변환하여야 할 때
  - 외부 데이터가 크기가 크지만 자주 참조되는 것은 아닐 때

예: 외부 테이블의 정의

```
CREATE table employees_ext(
  employee_id NUMBER, first_name CHAR(30), ...
)
ORGANIZATION EXTERNAL
TYPE oracle_loader
DEFAULT DIRECTORY delta_dir
ACCESS PARAMETERS(
  RECORDS DELIMITED BY NEWLINE
  FIELDS TERMINATED BY ',')
LOCATION('emp1.txt','emp2.txt')
PARALLEL 5;
```

외부 테이블의 조회

```
SELECT *
  FROM employees_ext;
```