데이터 무결성을 위한 제약 조건

데이터 무결성 제작 조건이란

❖ 데이터 무결성

- 데이터의 정확성을 유지 해야 한다.

- EMPLOYEES 테이블에 부서번호 10이 있다면
  반드시 DEPARTMENTS 테이블에 부서 번호 10에 해당하는 레코드가 있어야 한다.

- EMPLOYEES 테이블에 부서번호 20을 참조하는 레코드가 있는데
  부서 테이블에서 부서번호 20을 삭제한다면?

- 무결성 제약 조건으로 이를 방지
  - 반드시 EMPLOYEES 테이블에서 부서번호 20번 참조가 없는 상태에서(삭제하거나 다른 부서로 변경) DEPARTMENTS 테이블에서 부서번호 20번 삭제

무결한 데이터의 5가지 조건

❖ 무결성 제약 조건

- NOT NULL
  - NULL을 허용하지 않는다.
- UNIQUE
  - 중복된 값을 허용하지 않는다
- PRIMARY KEY
  - NULL을 허용하지 않고 중복된 값을 허용하지 않는다.
  - NOT NULL과 UNIQUE 조건을 결합한 형태
- FOREIGN KEY
  - 참조되는 테이블의 컬럼의 값이 존재하면 허용한다
- CHECK
  - 저장 가능한 데이터 값의 범위나 조건을 지정하여 설정한 값만을 허용한다

제약조건 확인하기

❖ 제약 조건 확인하기

- DESC USER_CONSTRAINTS;
  - OWNER : 제약 조건을 소유한 사용자
  - CONSTRAINT_NAME : 제약 조건명
  - CONSTRAINT_TYPE : 제약 조건 유형

CONSTRAINT_TYPE	의미
P	PRIMARY_KEY
R	FOREIGN_KEY
U	UNIQUE
C	CHECK NOT NULL

제약조건 확인하기

❖ 제약 조건 확인하기

```
SELECT CONSTRAINT_NAME, CONSTRAINT_TYPE, TABLE_NAME
FROM USER_CONSTRAINTS;
```

EMP_LAST_NAME_NN	C	EMPLOYEES
EMP_EMAIL_NN	C	EMPLOYEES
EMP_HIRE_DATE_NN	C	EMPLOYEES
EMP_JOB_NN	C	EMPLOYEES
EMP_SALARY_MIN	C	EMPLOYEES
EMP_EMAIL_UK	U	EMPLOYEES
EMP_EMP_ID_PK	P	EMPLOYEES
EMP_DEPT_FK	R	EMPLOYEES
EMP_JOB_FK	R	EMPLOYEES

- 어떤 컬럼에 제약조건이 있는지는 알 수 없음

제약조건 확인하기

❖ 제약조건 컬럼 조회

- USER_CONS_COLUMNS 데이터 딕셔너리 뷰

```
SELECT * FROM USER_CONS_COLUMNS
```

OWNER	CONSTRAINT_NAME	TABLE_NAME	COLUMN_NAME
HR	EMP_EMP_ID_PK	EMPLOYEES	EMPLOYEE_ID
HR	EMP_LAST_NAME_NN	EMPLOYEES	LAST_NAME
HR	EMP_EMAIL_NN	EMPLOYEES	EMAIL
HR	EMP_EMAIL_UK	EMPLOYEES	EMAIL
HR	EMP_HIRE_DATE_NN	EMPLOYEES	HIRE_DATE
HR	EMP_JOB_NN	EMPLOYEES	JOB_ID
HR	EMP_JOB_FK	EMPLOYEES	JOB_ID
HR	EMP_SALARY_MIN	EMPLOYEES	SALARY
HR	EMP_MANAGER_FK	EMPLOYEES	MANAGER_ID
HR	EMP_DEPT_FK	EMPLOYEES	DEPARTMENT_ID

필수 입력을 위한 NOT NULL 제약 조건

❖ NOT NULL 제약 조건

- 해당 컬럼의 값이 반드시 존재해야 하는 경우

```
CREATE TABLE EMP04(
      EMPNO NUMBER(4) NOT NULL,
      ENAME VARCHAR2(10) NOT NULL,
      JOB VARCHAR2(9),
      DEPTNO NUMBER(2));

INSERT INTO EMP04
VALUES(NULL, NULL, 'SALESMAN', 10);
```

유일한 값만 허용하는 UNIQUE 제약조건

❖ UNIQUE 제약조건

- 특정 컬럼에 자료가 중복되지 않게 하는 것
- 지정 컬럼에는 유일한 값이 수록

```
INSERT INTO EMP03
VALUES(7449, 'Jones', 'CLERK', 20);

INSERT INTO EMP03
VALUES(7450, 'Jones', 'CLERK', 20);

INSERT INTO EMP03
VALUES(NULL, 'Jones', 'MANAGER', 20);

INSERT INTO EMP03
VALUES(NULL, 'Jones', 'SALESMAN', 10);
```

컬럼 레벨로 제약 조건명을 명시하여 제약조건 설정하기

❖ 제약 조건 설정시

- 오라클은 SYS_ 다음에 숫자를 나영하여 제약 조건명을 자동 부여
- 제약 조건 위배시 제약 조건명 출력
- 어떤 제약 조건인지 알려면 USER_CONSTRAINTS 데이터 딕셔너리 조회

❖ CONSTRAINT 키워드

- 사용자가 직접 제약 조건 명을 설정
- 의미있게 이름을 부여하여 제약 조건 위배시 쉽게 파악
- 형식

  컬럼이름 데이터타입 CONTRAINT 제약조건명 제약조건타입

- 제약조건명 형식

  테이블명_컬럼명_제약조건유형

  EMP04_EMPNO_UK

컬럼 레벨로 제약 조건명을 명시하여 제약조건 설정하기

❖ 컬럼 레벨로 제약 조건명 명시하기

```
CREATE TABLE EMP04(
      EMPNO NUMBER(4) CONSTRAINT EMP04_EMPNO_UK UNIQUE,
      ENAME VARCHAR2(10) CONSTRAINT EMP04_ENMAE_NN NOT NULL,
      JOB VARCHAR2(9),
      DEPTNO NUMBER(2));


SELECT TABLE_NAME, CONSTRAINT_NAME
FROM USER_CONSTRAINTS
WHERE TABLE_NAME IN('EMP04')
```

컬럼 레벨로 제약 조건명을 명시하여 제약조건 설정하기

❖ 컬럼 레벨로 제약 조건명 명시하기

```
INSERT INTO EMP04
VALUES(7499, 'ALLEN', 'SALESMAN', 30);

INSERT INTO EMP04
VALUES(7499, 'JONES', 'MANAGER', 20);
```

SQL 오류: ORA-00001: unique constraint (**HR.EMP04_EMPNO_UK**) violated

데이터 구분을 위한 PRIMARY KEY 제약 조건

❖ PRIMARY KEY 제약 조건

- 테이블에 저장된 데이터를 유일하게 구분하기 위해 사용
- UNIQUE + NOT NULL의 특성

```
DROP TABLE EMP05

CREATE TABLE EMP05 (
      EMPNO NUMBER(4) CONSTRAINT EMP05_EMPNO_PK PRIMARY KEY,
      ENAME VARCHAR2(10) CONSTRAINT EMP05_ENMAE_NN NOT NULL,
      JOB VARCHAR2(9),
      DEPTNO NUMBER(2)
);
```

데이터 구분을 위한 PRIMARY KEY 제약 조건

❖ PRIMARY KEY 제약 조건

```
INSERT INTO EMP05
VALUES(7499, 'ALLEN', 'SALESMAN', 30);

INSERT INTO EMP05
VALUES(7499, 'JONES', 'MANAGER', 20);

→ SQL 오류: ORA-00001: unique constraint (HR.EMP05_EMPNO_PK)
violated

INSERT INTO EMP05
VALUES(NULL, 'JONES', 'MANAGER', 20);
→ SQL 오류: ORA-01400: cannot insert NULL into
("HR"."EMP05"."EMPNO
```

참조 무결성을 위한 FOREIGN KEY 제약 조건

❖ 참조 무결성

- 테이블 사이의 관계에서 발생
- EMPLOYEES 테이블의 DEPARTMENT_ID는 반드시 DEPARTMENT 테이블에 존재해야 한다.

- 사원은 회사 내에 존재하는 부서에 소속되어야 한다.

참조 무결성을 위한 FOREIGN KEY 제약 조건

❖ 참조의 무결성을 위한 부모와 자식 테이블의 관계

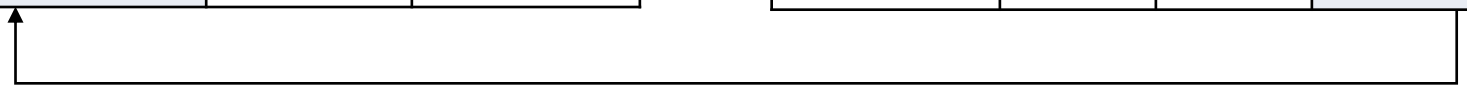
- 부모 테이블-자식 테이블/부모 키 – 자식 키
  - 주체가 되는 테이블 : 부모 테이블(DEPARTMENTS)
  - 종속 되는 테이블 : 자식 테이블(EMPLOYEES)

부모 테이블

Primary Key		

자식 테이블

			Foreign Key



- 부모 키 컬럼
  - 부모 테이블의 기본 키(PRIMARY KEY)이거나 유일 키(UNIQUE)

참조 무결성을 위한 FOREIGN KEY 제약 조건

❖ 외래 키(FOREIN KEY) 제약조건

- 자식 테이블의 Foreign key 컬럼(EMPLOYEES의 DEPARTMENT_ID)을 부모 테이블의 부모 키(DEPARTMENTS의 DEPARTMENT_ID)로 설정하는 것

부모 테이블(DEPARTMENTS)

DEPARTMENT_ID		
20		
10		

자식 테이블(EMPLOYEES)

			DEPARTMENT_ID
			10
			20

Primary Key ↑ **Foreign Key**

참조 무결성을 위한 FOREIGN KEY 제약 조건

❖ 부서테 이블과 사원 테이블간의 외래키 제약 조건 확인

```
SELECT TABLE_NAME, CONSTRAINT_TYPE,
CONSTRAINT_NAME, R_CONSTRAINT_NAME
FROM USER_CONSTRAINTS
WHERE TABLE_NAME IN('DEPTMENTS', 'EMPLOYEES');
```

DEPARTMENTS	P	DEPT_ID_PK	
EMPLOYEES	R	EMP_DEPT_FK	DEPT_ID_PK

참조 무결성을 위한 FOREIGN KEY 제약 조건

❖ **외래키 제약 조건 설정**

```
CREATE TABLE EMP06(
            …
            DEPTNO NUMBER(2) REFERENCES   DEPT (DEPTNO)
);
```

참조 무결성을 위한 FOREIGN KEY 제약 조건

❖ 실습 – 외래키 제약 조건

- 부모 테이블 준비

```
DROP TABLE DEPT06;

CREATE TABLE DEPT06 (
        DEPTNO NUMBER(4) PRIMARY KEY,
        DNAME VARCHAR2(20) NOT NULL,
        LOC   VARCHAR2(20));

INSERT INTO DEPT06 VALUES( 10, 'SALES', 'SEATTLE');
INSERT INTO DEPT06 VALUES( 20, 'ACCOUNTING', 'DALAS');
INSERT INTO DEPT06 VALUES( 30, 'MARKETING', 'NEW YORK');
```

❖ 실습 – 외래키 제약 조건

- 부모 테이블 자식

```
DROP TABLE EMP06;

CREATE TABLE EMP06 (
        EMPNO NUMBER(4) CONSTRAINT EMP06_EMPNO_PK PRIMARY KEY,
        ENAME VARCHAR2(10) CONSTRAINT EMP06_ENMAE_NN NOT NULL,
        JOB VARCHAR2(9),
        DEPTNO NUMBER(2) CONSTRAINT EMP06_DEPTNO_FK
                        REFERENCES DEPT06(DEPTNO)
  );
```

참조 무결성을 위한 FOREIGN KEY 제약 조건

❖ 실습 – 외래키 제약 조건

```
INSERT INTO EMP06
VALUES(7566, 'JONES', 'MANAGER', 50);

→ SQL 오류: ORA-02291: integrity constraint (HR.EMP06_DEPTNO_FK)
violated - parent key not found

INSERT INTO EMP06
VALUES(7566, 'JONES', 'MANAGER', 20);

DELETE  DEPT06
WHERE DEPTNO = 20;
→ SQL 오류: ORA-02292: integrity constraint (HR.EMP06_DEPTNO_FK)
violated - child record found
```

참조 무결성을 위한 FOREIGN KEY 제약 조건

❖ **실습 – 외래키 제약 조건**

- 제약 조건 확인하기

```
SELECT CONSTRAINT_NAME, CONSTRAINT_TYPE, R_CONSTRAINT_NAME
FROM USER_CONSTRAINTS
WHERE TABLE_NAME='EMP06';

→
EMP06_ENMAE_NN  C
EMP06_EMPNO_PK  P
EMP06_DEPTNO_FK R       SYS_C007057
```

CHECK와 DEFAULT의 제약 조건

❖ CHECK 제약조건

- 입력되는 값을 체크하여 설정된 값 이외의 값이 들어오면 오류 메시지와 함께 명령이 수행되지 못하게 한다.
- 조건으로 데이터의 범위나 특정 패턴의 숫자나 문자값을 설정

❖ 디폴트 제약 조건

- 아무런 값을 입력하지 않았을 때 디폴트 제약 조건의 값이 입력된다.

CHECK와 DEFAULT의 제약 조건

❖ CHECK 제약 조건 설정하기

- 사원 테이블에 급여 컬럼을 생성하되 컬럼 값은 500에서 5000사이의 값만 저장할 수 있게 한다.
- 성별을 저장하는 컬럼으로 GENDER를 정의하고 남자는 M, 여자는 F 둘 중의 하나만 저장할 수 있도록 한다

```
DROP EMP07;

CREATE TABLE EMP07 (
        EMPNO NUMBER(4) CONSTRAINT EMP07_EMPNO_PK PRIMARY KEY,
        ENAME VARCHAR2(10) CONSTRAINT EMP07_ENMAE_NN NOT NULL,
        SAL NUMBER(7,2) CONSTRAINT EMP07_SAL_CK
                        CHECK(SAL BETWEEN 500 AND 5000),
        GENDER CHAR(1) CONSTRAINT EMP07_GENDER_CK
                        CHECK(GENDER IN('M','F'))
);
```

CHECK와 DEFAULT의 제약 조건

❖ CHECK 제약 조건 설정하기

```
INSERT INTO EMP07
VALUES(7499, 'ALLEN', 200, 'M');
→ ORA-02290: check constraint (HR.EMP07_SAL_CK) violated

INSERT INTO EMP07
VALUES(7499, 'ALLEN', 4000, 'A');
→ ORA-02290: check constraint (HR.EMP07_GENDER_CK) violated
```

- 제약 조건 확인

CHECK와 DEFAULT의 제약 조건

❖ CHECK 제약 조건 설정하기

- 제약 조건 확인

```
SELECT TABLE_NAME, CONSTRAINT_TYPE, CONSTRAINT_NAME, SEARCH_CONDITION
FROM USER_CONSTRAINTS
WHERE TABLE_NAME='EMP07';
```

```
EMP07   C       EMP07_ENMAE_NN  "ENAME" IS NOT NULL
EMP07   C       EMP07_SAL_CK    SAL BETWEEN 500 AND 5000
EMP07   C       EMP07_GENDER_CK GENDER IN('M','F')
EMP07   P       EMP07_EMPNO_PK
```

CHECK와 DEFAULT의 제약 조건

❖ DEFAULT 제약 조건 설정하기

```
DROP TABLE DEPT07;

CREATE TABLE DEPT07(
        DEPTNO NUMBER(2) PRIMARY KEY,
        DNAME VARCHAR2(14),
        LOC VARCHAR2(20) DEFAULT 'SEOUL');

INSERT INTO DEPT07 (DEPTNO, DNAME)
VALUES(10, 'ACCOUNTING');

SELECT * FROM DEPT07;
→
10      ACCOUNTING      SEOUL
```

테이블 레벨 방식으로 제약 조건 지정하기

❖ 테이블 레벨 방식 제약 조건 지정

- 복합키로 기본 키를 지정할 경우
- ALTER TABLE로 제약 조건을 추가할 때

- 형식

```
CREATE TABLE 테이블명(
        컬럼명1  데이터타입1,
        컬럼명2  데이터타입2,
        …
        [CONSTRAINT 제약조건명] 제약조건타입(컬럼명)
);
```

테이블 레벨 방식으로 제약 조건 지정하기

❖ 컬럼 레벨 제약 조건 설정과 테이블 레벨 제약 조건 설정하기

- 테이블 레벨 제약 조건 설정

```
DROP TABLE EMP06;

CREATE TABLE EMP06(
        EMPNO NUMBER(4),
        ENAME VARCHAR2(10),
        JOB VARCHAR2(9),
        DEPTNO NUMBER(4),
        PRIMARY KEY(EMPNO),
        UNIQUE(JOB),
        FOREIGN KEY(DEPTNO) REFERENCES DEPT06(DEPTNO)
);
```

테이블 레벨 방식으로 제약 조건 지정하기

❖ 컬럼 레벨 제약 조건 설정과 테이블 레벨 제약 조건 설정하기

- 컬럼 레벨 제약 조건 설정

```
DROP TABLE EMP06;

CREATE TABLE EMP06(
        EMPNO NUMBER(4) PRIMARY KEY,
        ENAME VARCHAR2(10) NOT NULL,
        JOB VARCHAR2(9) UNIQUE,
        DEPTNO NUMBER(4) REFERENCES DEPT06(DEPTNO)
);
```

테이블 레벨 방식으로 제약 조건 지정하기

❖ 복합키를 기본 키로 지정하는 방법

```
CREATE TABLE MEMBER01(
NAME VARCHAR2(10),
ADDRESS VARCHAR2(30),
HPHONE VARCHAR2(16),
CONSTRAINT MAMEBER01_COMBO_PK PRIMARY KEY(NAME, HPHONE));

SELECT CONSTRAINT_NAME, CONSTRAINT_TYPE
FROM USER_CONSTRAINTS
WHERE TABLE_NAME='MEMBER01';
→
MAMEBER01_COMBO_PK      P

SELECT *
FROM USER_CONS_COLUMNS
WHERE TABLE_NAME='MEMBER01';
→
HR      MAMEBER01_COMBO_PK      MEMBER01        NAME    1
HR      MAMEBER01_COMBO_PK      MEMBER01        HPHONE  2
```

제약조건 변경하기

❖ 제약 조건 추가하기

ㅇ 형식

```
ALTER 테이블명
ADD [CONSTRAINT  제약조건명] 제약조건_타입(컬럼이름)
```

```
DROP TABLE EMP01;

CREATE TABLE EMP01(
        EMPNO NUMBER(4),
        ENAME VARCHAR2(10),
        JOB VARCHAR2(9),
        DEPTNO NUMBER(4));

SELECT CONSTRAINT_NAME, CONSTRAINT_TYPE, R_CONSTRAINT_NAME
FROM USER_CONSTRAINTS
WHERE TABLE_NAME='EMP01';
```

제약조건 변경하기

❖ 제약 조건 추가하기

```
ALTER TABLE EMP01
ADD CONSTRAINT EMP01_EMPNO_PK PRIMARY KEY(EMPNO);

ALTER TABLE EMP01
ADD CONSTRAINT EMP01_DEPTNO_FK
    FOREIGN KEY(DEPTNO) REFERENCES DEPT06(DEPTNO);

SELECT CONSTRAINT_NAME, CONSTRAINT_TYPE, R_CONSTRAINT_NAME
FROM USER_CONSTRAINTS
WHERE TABLE_NAME='EMP01';
→
EMP01_EMPNO_PK          P
EMP01_DEPTNO_FK         R       PK_DEPT
```

제약조건 변경하기

❖ MODIFY로 NOT NULL 제약 조건 추가하기

- NOT NULL 제약조건은 다른 제약 조건과 달리 ADD CONSTRAINT를 사용하지 못함
- MODIFY CONSTRAINT 명령어 사용
- 형식

```
ALTER TABLE 테이블명
MODIFY 컬럼명 CONSTRAINT 제약조건명 NOT NULL;
```

```
ALTER TABLE EMP01
MODIFY ENAME CONSTRAINT EMP01_ENAME_NN NOT NULL;

SELECT CONSTRAINT_NAME, CONSTRAINT_TYPE, R_CONSTRAINT_NAME
FROM USER_CONSTRAINTS
WHERE TABLE_NAME='EMP01';
```

제약조건 변경하기

❖ 제약 조건 제거하기

- 형식

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

```
ALTER TABLE EMP01
DROP PRIMARY KEY;

ALTER TABLE EMP01
DROP CONSTRAINT EMP01_ENAME_NN;
```