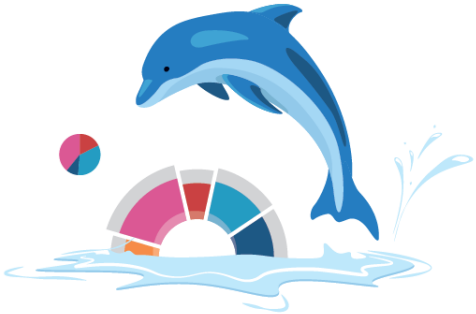
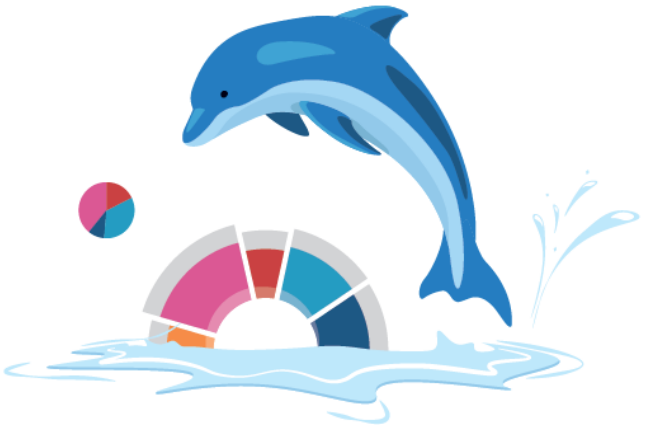
난생처음

MySQL



한빛아카데미
Hanbit Academy, Inc.

Chapter 08
DDL:
데이터 정의어

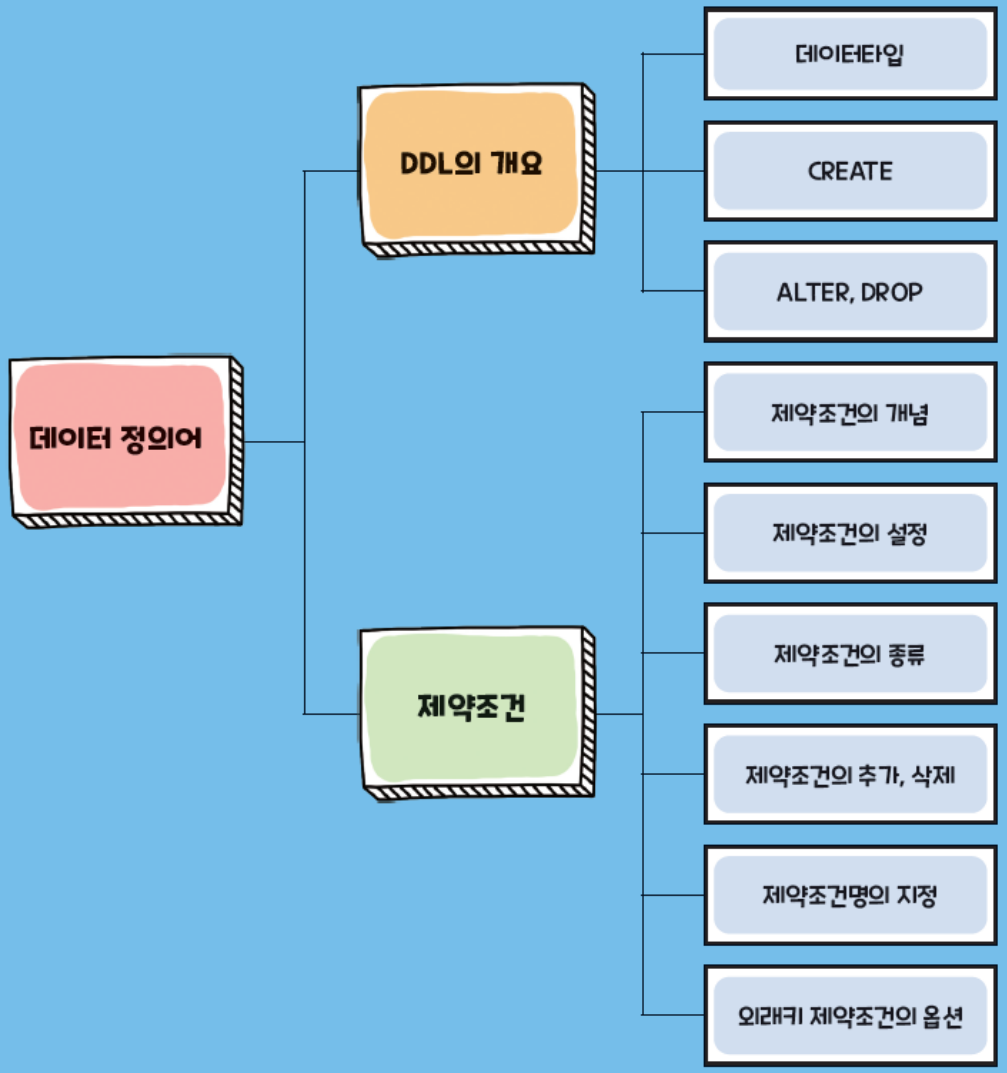


목차

학습목표

- 데이터 정의어를 작성할 수 있습니다.
- 제약조건을 이해하고 적용할 수 있습니다

Preview


데이터 정의어
DDL의 개요
데이터타입
CREATE
ALTER, DROP
제약조건
제약조건의 개념
제약조건의 설정
제약조건의 종류
제약조건의 추가, 삭제
제약조건명의 지정
외래키 제약조건의 옵션

Section 01

DDL의 개요

1. DDL의 개요

■ 데이터 정의어(Data Definition Language, DDL)

- 데이터베이스 내에 테이블이나 인덱스, 뷰 등의 객체를 만들거나 수정, 삭제할 때 사용하는 언어를 의미함
- DDL에는 CREATE, ALTER, DROP이 있음

그림 8-1 CREATE, ALTER, DROP의 개념

Section 02

데이터타입

■ 문자형 데이터타입

- CHAR
  - ✓ 고정길이 문자형 데이터타입
  - ✓ 지정한 길이보다 데이터 길이가 작으면 빈칸만큼 공백(Blank)이 들어감
  - ✓ [예] 고객번호, 주민등록번호
- VARCHAR
  - ✓ 가변길이 문자형 데이터타입
  - ✓ 데이터의 길이만큼의 메모리만 차지함
  - ✓ [예] 고객회사명, 주소
- TEXT
  - ✓ VARCHAR과 달리 길이를 지정하지 않음
  - ✓ 컬럼의 최대 길이를 모르는 경우에 사용하기 적합함

■ 문자형 데이터타입

표 8-1 문자형 데이터타입

데이터타입	설명	최대크기(단위: Bytes)
CHAR(길이)	고정길이 문자형 데이터타입	255
VARCHAR(길이)	가변길이 문자형 데이터타입	65,535
TINYTEXT	문자열 데이터타입	255
TEXT	문자열 데이터타입	65,535
MEDIUMTEXT	문자열 데이터타입	16,777,215
LONGTEXT	문자열 데이터타입	4,294,967,295
JSON	JSON 문자열 데이터타입	1GB

■ 숫자형 데이터타입

- 정수형
  - ✓ 들어갈 데이터의 크기에 따라 TINYINT부터 BIGINT 중에서 선택하여 지정할 수 있음

표 8-2 정수형 데이터타입

데이터타입	최대크기(단위: Bytes)
TINYINT	1
SMALLINT	2
MEDIUMINT	3
INT	4
BIGINT	8

- 실수형
  - ✓ FLOAT형, DOUBLE형, DECIMAL형 중에서 선택하여 지정할 수 있음

표 8-3 실수형 데이터타입

데이터타입	표현 형식	최대크기
FLOAT	부동 소수점 형식	4Bytes 소수점 아래 7자리까지 표현
DOUBLE	부동 소수점 형식	8Bytes 소수점 아래 15자리까지 표현
DECIMAL(M,D) M: 전체 자릿수 D: 소수점 자릿수	고정 소수점 형식 DEC, NUMERIC이라고도 함	전체 자릿수(M)은 65까지 표현 소수점 자릿수(D)는 30까지 표현 예 DECIMAL(4,2): -99.99 ~ 99.99까지 저장

3. 날짜시간형 데이터타입

■ 날짜시간형 데이터타입

- DATE는 날짜, TIME은 시간을 저장하기 위한 데이터타입
- DATETIME과 TIMESTAMP는 날짜와 시간을 함께 저장할 수 있는 데이터타입
  - ✓ DATETIME과 TIMESTAMP의 차이는 시간대(Time Zone)의 적용 여부임
  - ✓ TIMESTAMP는 내부적으로 시간을 가져올 때 시간대를 적용시켜 보여줌
  - ✓ 즉, 데이터베이스가 글로벌 서비스에서 사용된다면 DATETIME 대신 TIMESTAMP를 사용해야 함

표 8-4 날짜시간형 데이터타입

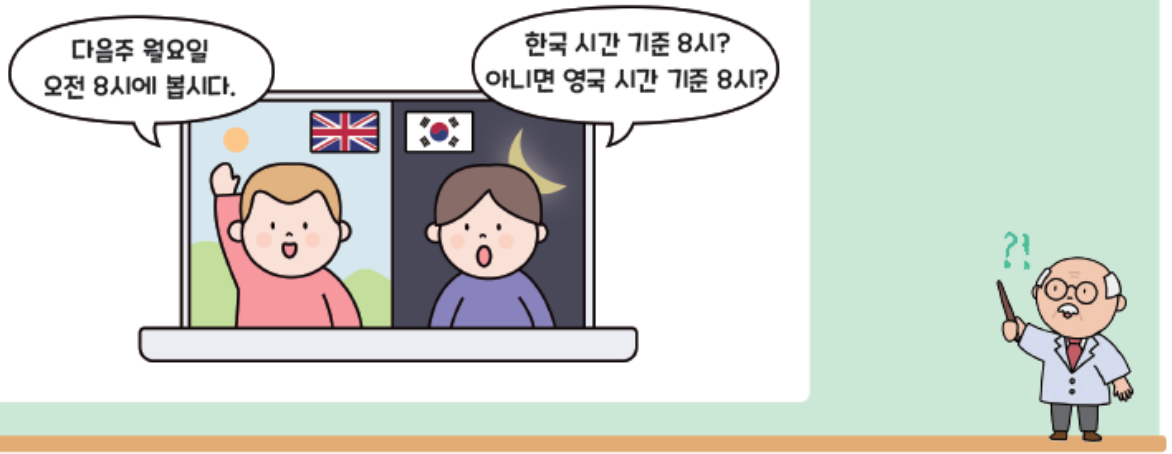
데이터타입	형식	데이터 범위
DATE	YYYY-MM-DD	1000-01-01 ~ 9999-12-31
TIME	HH:MI:SS	-838:59:59 ~ 838:59:59
DATETIME	YYYY-MM-DD HH:MI:SS	1000-01-01 00:00:00 ~ 9999-12-31 23:59:59
TIMESTAMP	YYYY-MM-DD HH:MI:SS	1970-01-01 00:00:01 UTC ~ 2038-01-19 03:14:07 UTC

3. 날짜시간형 데이터타입

하나 더 알기 Y **시간대**

시간대(Time Zone)란 지역 사이에 생기는 낮과 밤의 차이를 인위적으로 조정하기 위해 고안된 시간의 구분선을 의미합니다. 이는 영국의 그리니치 천문대를 기준으로 지역에 따른 시간의 차이를 계산하여 적용하기 때문에 GMT(Greenwich Mean Time)라 지칭합니다.

UTC(Coordinated Universal Time)는 1972년 1월 1일부터 시행된 국제 표준시로 UTC와 GMT는 초의 소수점 단위에서만 차이가 나기 때문에 혼용되어 사용되기도 하지만 기술적인 표기에서는 UTC가 사용됩니다. 우리나라에서는 KST(Korea Standard Time)를 표준시로 사용하는데, 이는 UTC에 9시간을 더한 시간대입니다.



4. 이진형 데이터타입, 공간현 데이터타입

■ 이진형 데이터타입

- BINARY 또는 BLOB(Binary Large OBject) 데이터타입
- BINARY(n)와 BYTE(n)는 CHAR 형태의 이진형 데이터타입
- VARBINARY(n)는 VARCHAR 형태의 이진형 데이터타입
- BLOB도 저장 크기에 따라 TINYBLOB, BLOB, MEDIUMBLOB, LONGBLOB 등의 데이터타입을 사용할 수 있음

■ 공간형 데이터타입

- 공간 데이터를 저장하기 위한 데이터타입
- GEOMETRY, POINT, LINESTRING, POLYGON, MULTIPOINT, MULTILINESTRING, MULTIPOLYGON, GEOMETRYCOLLECTION 등이 있음

Section 03

CREATE

1. 데이터베이스 생성하기

CREATE

- 데이터베이스나 테이블, 뷰, 인덱스 등 객체를 만들 때 사용함
- 형식

```
CREATE DATABASE [IF NOT EXISTS] 데이터베이스명;
```

- MySQL에서 한글을 사용하려면 Character Set 설정

```
CREATE DATABASE 데이터베이스명 [CHAR SET utf8mb4 COLLATE utf8mb4_general_ci];
```

[예제 8-1] 한빛학사 데이터베이스를 생성하시오.

```
CREATE DATABASE 한빛학사;
```

2. 테이블 생성하기

■ 테이블 생성

- 테이블을 생성할 때 모든 컬럼에는 데이터타입을 지정해주어야 함
- 형식

```
CREATE TABLE 테이블명
(
 컬럼1 데이터타입
,컬럼2 데이터타입
);
```

2. 테이블 생성하기

■ [예제 8-2] 다음과 같이 한빛학사 데이터베이스에 학과 테이블을 생성하고, 레코드 3건을 삽입하시오.

■ 학과 테이블 명세서

컬럼명	데이터타입
학과번호	고정문자형 2자
학과명	가변문자형 20자
학과장명	가변문자형 20자

■ 학과 데이터

학과번호	학과명	학과장명
AA	컴퓨터공학과	배경민
BB	소프트웨어학과	김남준
CC	디자인융합학과	박선영

2. 테이블 생성하기

■ [예제 8-2] 다음과 같이 한빛학사 데이터베이스에 학과 테이블을 생성하고, 레코드 3건을 삽입하시오.

- 학과 테이블을 생성

```
CREATE TABLE 학과
(
 학과번호 CHAR(2)
,학과명 VARCHAR(20)
,학과장명 VARCHAR(20)
);
```

- 학과 테이블에 3건의 레코드를 삽입

```
INSERT INTO 학과
VALUES('AA','컴퓨터공학과','배경민')
     ,('BB','소프트웨어학과','김남준')
     ,('CC','디자인융합학과','박선영');
```

▶ 실행결과

학과번호	학과명	학과장명
AA	컴퓨터공학과	배경민
BB	소프트웨어학과	김남준
CC	디자인융합학과	박선영

2. 테이블 생성하기

■ [예제 8-3] 다음과 같이 학생 테이블을 생성하고, 레코드 3건을 삽입하시오.

■ 학생 테이블 명세서

컬럼명	데이터타입
학번	고정문자형 5자
이름	가변문자형 20자
나이	숫자형
연락처	숫자형
학과명	가변문자형 20자

- Q) 학생 테이블 명세서에서 적절하지 않은 컬럼은?
- A) 나이, 연락처, 학과명

2. 테이블 생성하기

■ [예제 8-3] 다음과 같이 학생 테이블을 생성하고, 레코드 3건을 삽입하시오.

■ 변경된 학생 테이블 명세서

컬럼명	데이터타입
학번	고정문자형 5자
이름	가변문자형 20자
생일	날짜형
연락처	가변문자형 20자
학과번호	고정문자형 2자

■ 학생 데이터

학과번호	이름	생일	연락처	학과번호
S0001	이윤주	2020-01-30	01033334444	AA
S0001	이승은	2021-02-23	NULL	AA
S0003	백재용	2018-03-31	01077778888	DD

2. 테이블 생성하기

■ [예제 8-3] 다음과 같이 학생 테이블을 생성하고, 레코드 3건을 삽입하시오.

- 학생 테이블 생성

```
CREATE TABLE 학생
(
 학번 CHAR(5)
,이름 VARCHAR(20)
,생일 DATE
,연락처 VARCHAR(20)
,학과번호 CHAR(2)
);
```

- 학생 데이터 표를 참고하여 3건의 레코드 삽입

```
INSERT INTO 학생
VALUES('S0001','이윤주','2020-01-30','01033334444','AA')
     ,('S0001','이승은','2021-02-23',NULL,'AA')
     ,('S0003','백재용','2018-03-31','01077778888','DD');
```

▶ 실행결과

학번	이름	생일	연락처	학과번호
S0001	이윤주	2020-01-30	01033334444	AA
S0001	이승은	2021-02-23	NULL	AA
S0003	백재용	2018-03-31	01077778888	DD

2. 테이블 생성하기

■ 테이블의 구조와 데이터 복사

- 형식

```
CREATE TABLE 테이블명 AS
    SELECT문;
```

■ [예제 8-4] 학생 테이블을 사용하여 휴학생 테이블을 생성하시오. 이때 데이터는 복사하지 않고, 구조만 복사하시오.

```
CREATE TABLE 휴학생 AS
    SELECT *
    FROM 학생
    WHERE 1 = 2;
```

▶ 실행결과

학번	이름	생일	연락처	학과번호

3. GENERATED 컬럼

■ GENERATED 컬럼

- 테이블에 있는 컬럼을 기반으로 하여 계산된 값을 저장할 수 있는 컬럼
- 형식

```
컬럼 데이터타입 [GENERATED ALWAYS] AS 계산식 [VIRTUAL ¦ STORED]
```

- VIRTUAL 방식
  - ✓ 기본값
  - ✓ 데이터는 저장하지 않고 정의만 데이터 사전에 추가하는 방식
- STORED 방식
  - ✓ 레코드가 삽입되거나 수정될 때마다 값이 계산되어 저장되기 때문에 추가 저장 공간이 필요하지만, SELECT 검색 시에는 계산 없이 바로 값을 읽어올 수 있음

3. GENERATED 컬럼

■ [예제 8-5] 휘트니스센터의 회원을 관리하는 테이블을 만드시오. 이때 체질량 지수가 자동 계산되어 저장되도록 GENERATED 컬럼으로 설정하시오.

■ 회원 테이블 명세서

컬럼명	데이터타입	제약조건
아이디	가변문자형 20자	기본키
회원명	가변문자형 20자	
키	정수형	
몸무게	정수형	
체질량지수	실수형	몸무게(Kg) / POWER(키(cm), 2)

3. GENERATED 컬럼

■ [예제 8-5] 휘트니스센터의 회원을 관리하는 테이블을 만드시오. 이때 체질량 지수가 자동 계산되어 저장되도록 GENERATED 컬럼으로 설정하시오.

- 회원 테이블 생성

```
CREATE TABLE 회원
(
 아이디 VARCHAR(20) PRIMARY KEY
,회원명 VARCHAR(20)
,키 INT
,몸무게 INT
,체질량지수 DECIMAL(4,1) AS (몸무게 / POWER(키 / 100, 2)) STORED
);
```

- 레코드 삽입

```
INSERT INTO 회원(아이디, 회원명, 키, 몸무게)
VALUES('APPLE','김사과',178, 70);
```

▶ 실행결과

아이디	회원명	키	몸무게	체질량지수
APPLE	김사과	178	70	22.1

3. GENERATED 컬럼

확인문제

다음 지문의 내용이 맞으면 ○, 틀리면 ×를 표시하시오.

1. MySQL에서는 컬럼에 있는 값을 기반으로 자동 계산되어지는 컬럼을 생성할 수 있다. ( )

2. CREATE문을 사용하여 기존에 존재하는 테이블의 구조와 데이터를 그대로 복사할 수 있다. ( )

정답

1. ○ 2. ○

Section 04

ALTER, DROP

■ 컬럼 추가

- 형식

```
ALTER TABLE 테이블명 ADD COLUMN new_컬럼명 데이터타입;
```

■ [예제 8-6] 학생 테이블에 성별 컬럼을 추가하시오.

컬럼명	데이터타입
성별	고정문자형 1자

```
ALTER TABLE 학생 ADD 성별 CHAR(1);
```

▶ 실행결과

학번	이름	생일	연락처	학과번호	성별
S0001	이운주	2020-01-30	01033334444	AA	NULL
S0001	이승은	2021-02-23	NULL	AA	NULL
S0003	백재웅	2018-03-31	01077778888	DD	NULL

■ 컬럼 데이터타입 변경

- 형식

```
ALTER TABLE 테이블명 MODIFY COLUMN 컬럼명 new_데이터타입;
```

■ [예제 8-7] 학생 테이블에 성별 컬럼을 추가하시오.

컬럼명	데이터타입
성별	가변문자형 2자

```
ALTER TABLE 학생 MODIFY COLUMN 성별 VARCHAR(2);
```

■ 컬럼명 변경

- 형식

```
ALTER TABLE 테이블명 CHANGE COLUMN old_컬럼명 new_컬럼명 데이터타입;
```

■ [예제 8-8] 학생 테이블에서 연락처 컬럼명을 휴대폰번호로 변경하시오

```
ALTER TABLE 학생 CHANGE COLUMN 연락처 휴대폰번호 VARCHAR(20);
```

▶ 실행결과

학번	이름	생일	휴대폰번호	학과번호	성별
S0001	이윤주	2020-01-30	01033334444	AA	NULL

■ 컬럼 삭제

- 형식

```
ALTER TABLE 테이블명 DROP COLUMN 컬럼명;
```

■ [예제 8-9] 학생 테이블에서 성별 컬럼을 삭제하시오

```
ALTER TABLE 학생 DROP COLUMN 성별;
```

▶ 실행결과

학번	이름	생일	휴대폰번호	학과번호
S0001	이윤주	2020-01-30	01033334444	AA
S0001	이승은	2021-02-23	NULL	AA
S0003	백재용	2018-03-31	01077778888	DD

1. ALTER

■ 테이블명 변경

- 형식

```
ALTER TABLE old_테이블명 RENAME new_테이블명;
```

■ [예제 8-10] 휴학생 테이블명을 졸업생 테이블명으로 변경하시오

```
ALTER TABLE 휴학생 RENAME 졸업생;
```

2. DROP

DROP

- 데이터베이스, 테이블 및 기타 여러 객체를 삭제할 때 사용함
- 형식

```
DROP DATABASE 데이터베이스명;
DROP TABLE 테이블명;
```

[예제 8-11] 학과 테이블과 학생 테이블을 삭제하시오

```
DROP TABLE 학과;
DROP TABLE 학생;
```

2. DROP

확인문제

다음 지문에 따라 명령어가 실행되도록 빈칸을 채우시오.

1. 학생 테이블명을 student로 변경한다.

```
ALTER TABLE 학생 [          ] student;
```

2. 학번 컬럼을 student_id로 변경한다.

```
ALTER TABLE student [          ] COLUMN 학번 student_id CHAR(5);
```

3. student 테이블에 address 컬럼을 추가한다.

```
ALTER TABLE student [          ] COLUMN address VARCHAR(10);
```

4. address 컬럼의 데이터타입 크기를 50으로 변경한다.

```
ALTER TABLE student [          ] COLUMN address VARCHAR(50);
```

5. address 컬럼을 삭제한다.

```
ALTER TABLE student [          ] COLUMN address;
```

정답

1. RENAME 2. CHANGE 3. ADD 4. MODIFY 5. DROP

Section 05

제약조건

1. 제약조건의 개념

■ 제약조건

- 데이터베이스에는 무결한 데이터가 들어가야 함
- [예제 8-3]에서 보았듯이 테이블에 아무런 제약 사항을 두지 않으면 적합하지 않은 데이터가 저장되어 무결성 위배가 발생할 수 있음
- 이를 위해 테이블에 제약조건을 설정하여 데이터 무결성을 유지할 수 있음

■ 제약조건의 특징

- 제약조건은 데이터 사전에 저장됨
- 제약조건은 CREATE문으로 테이블을 생성할 때 지정할 수 있고, ALTER문으로 테이블의 구조를 변경할 때 지정할 수도 있음
- 제약조건은 고유한 이름을 붙여서 식별할 수 있음
- 한 컬럼에 여러 개의 제약조건을 설정할 수 있음

■ 제약조건의 설정

- 형식

```
CREATE TABLE 테이블명
(
 컬럼1 데이터타입 제약조건 ---- ①
,컬럼2 데이터타입 ---- ②
,제약조건(컬럼2) ---- ③
);
```

- 테이블 레벨 제약조건 설정
  - ✓ ① **컬럼 레벨 제약조건 설정**: 컬럼의 데이터타입 바로 다음에 기술함
  - ✓ ②, ③ **테이블 레벨 제약조건 설정**: 컬럼의 정의를 끝낸 후 제약조건을 별도로 지정

하나 더 알기 **복합키와 대리키**

레코드를 식별하기 위해서는 기본키를 설정해야 합니다. 기본키는 한 개의 컬럼 이상으로 구성이 될 수 있으며, 두 개 이상의 컬럼으로 생성된 기본키를 복합키(Composite Key)라고 합니다.
대리키(Surrogate key)는 기본키를 대체하는 키로 인공키(Artificial key)라고도 합니다. 기본키가 여러 개의 컬럼으로 구성되어야 하거나 기본키로 사용할 속성이 없을 경우에 대리키를 사용할 수 있습니다. 또는 데이터 길이가 너무 길거나 보안이 필요한 컬럼을 기본키로 사용해야 하는 경우 기본키를 대체할 수 있도록 일련번호와 같은 컬럼을 생성하여 기본키로 사용할 수 있습니다.

3. 제약조건의 종류

■ PRIMARY KEY 제약

- 기본키를 설정하는 제약조건
- 기본키는 레코드를 대표하는 키로 한 개 이상의 컬럼으로 구성됨
- 기본키는 테이블당 한 개여야 함
- 기본키를 생성하면 자동으로 인덱스가 생성됨
- 기본키는 NOT NULL이어야 하고, 유일한 값을 가져야 함
  - ✓ (NOT NULL 제약조건 + UNIQUE 제약조건)

■ NOT NULL 제약

- NOT NULL 제약조건이 설정된 컬럼에는 반드시 값을 넣어야 함
- NOT NULL 제약조건은 반드시 컬럼 레벨로 설정해야 함

3. 제약조건의 종류

UNIQUE 제약

- UNIQUE 제약조건이 설정된 컬럼에는 반드시 유일한 값을 넣어야 함
- 자동으로 인덱스가 생성됨

CHECK 제약

- CHECK 제약조건이 설정된 컬럼에는 설정된 조건에 맞는 값만 넣어야 함
- 조건으로는 특정 값이나 범위, 특정 패턴의 숫자나 문자 값 등을 설정할 수 있음

DEFAULT 제약

- 값을 넣지 않을 경우 지정한 값이 자동으로 들어감

FOREIGN KEY 제약

- 외래키를 설정하는 제약조건
- 한 테이블의 외래키는 참조하는 테이블의 기본키이거나 NULL이어야 함
- 외래키의 컬럼과 참조하는 테이블의 기본키 컬럼의 데이터타입과 크기는 서로 동일해야 함

3. 제약조건의 종류

하나 더 알기 **개체 무결성과 참조 무결성**

개체 무결성과 참조 무결성은 데이터 무결성을 보장하는 중요한 규칙으로 이를 지키지 않으면 데이터의 정확성과 일관성이 손상될 수 있으므로 무결성 규칙을 엄격하게 준수해야 합니다.

■ **개체 무결성**

개체 무결성(Entity Integrity)은 기본키와 관련이 있습니다. 개체 무결성은 기본키 컬럼에 중복된 값이나 NULL 값을 허용하지 않는 원칙을 나타냅니다.

■ **참조 무결성**

참조 무결성(Referential Integrity)은 외래키와 관련이 있습니다. 특정 테이블의 외래키는 다른 테이블의 기본키를 참조함으로써 관계가 항상 일관성 있게 유지되어야 함을 의미합니다.

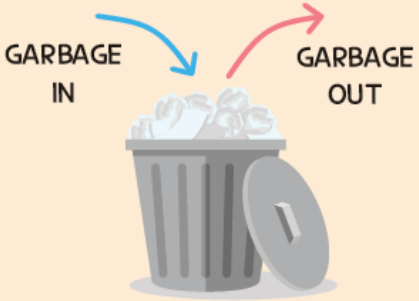
3. 제약조건의 종류

Q

테이블에 기본키를 꼭 지정해야 하나요?

A

기본키는 각 레코드를 고유하게 식별하는 역할을 합니다. 기본키를 지정하면 정확하고 빠르게 데이터를 검색하고 관리할 수 있으며, 레코드 간에 관계를 설정하거나 데이터를 조작할 때도 중요한 역할을 합니다. 또한 기본키는 중복을 방지하고 데이터의 무결성을 보장하는 데에도 도움이 됩니다. 기본키를 지정하지 않으면 데이터베이스 시스템이 중복 레코드를 허용할 수 있는데, 이는 데이터 관리와 정확성에 대해 문제를 초래할 수 있습니다. 따라서 특정 상황을 제외하고는 기본키를 지정하여 데이터의 고유성을 보장하고 데이터베이스의 성능을 향상시키는 것이 좋습니다.



■ [예제 8-12] 제약조건을 추가하여 학과 테이블을 다시 생성하시오

■ 학과 테이블 명세서(제약조건 추가)

컬럼명	데이터타입	제약조건
학과번호	고정문자형 2자	기본키
학과명	가변문자형 20자	필수 입력
학과장명	가변문자형 20자	

[예제 8-12] 제약조건을 추가하여 학과 테이블을 다시 생성하시오

- 방법1: 컬럼 레벨의 제약조건 설정

```
CREATE TABLE 학과
(
 학과번호 CHAR(2) PRIMARY KEY
,학과명 VARCHAR(20) NOT NULL
,학과장명 VARCHAR(20)
);
```

- 방법2: 테이블 레벨로 제약조건 설정

```
CREATE TABLE 학과
(
 학과번호 CHAR(2)
,학과명 VARCHAR(20) NOT NULL ●——② 
,학과장명 VARCHAR(20)
,PRIMARY KEY(학과번호) ●——①
);
```

■ [예제 8-13] 제약조건을 추가하여 학생 테이블을 다시 생성하시오

■ 학생 테이블 명세서(제약조건 추가)

컬럼명	데이터타입	제약조건
학번	고정문자형 5자	기본키
이름	가변문자형 20자	필수 입력
생일	날짜형	필수 입력
연락처	가변문자형 20자	유일한 값만 입력 가능
학과번호	고정문자형 2자	학과 테이블의 학과번호 참조
성별	고정문자형 1자	남 또는 여만 입력 가능
등록일	날짜형	오늘 날짜 자동 입력

[예제 8-13] 제약조건을 추가하여 학생 테이블을 다시 생성하시오

- 방법1: 컬럼 레벨의 제약조건 설정

```
CREATE TABLE 학생
(
 학번 CHAR(5) PRIMARY KEY
,이름 VARCHAR(20) NOT NULL
,생일 DATE NOT NULL
,연락처 VARCHAR(20) UNIQUE
,학과번호 CHAR(2) REFERENCES 학과(학과번호)
,성별 CHAR(1) CHECK(성별 IN ('남','여'))
,등록일 DATE DEFAULT(CURDATE())
,FOREIGN KEY(학과번호) REFERENCES 학과(학과번호)
);
```

[예제 8-13] 제약조건을 추가하여 학생 테이블을 다시 생성하시오

- 방법2: 테이블 레벨로 제약조건 설정

```
CREATE TABLE 학생
(
 학번 CHAR(5)
,이름 VARCHAR(20) NOT NULL
,생일 DATE NOT NULL
,연락처 VARCHAR(20)
,학과번호 CHAR(2)
,성별 CHAR(1)
,등록일 DATE DEFAULT(CURDATE())
,PRIMARY KEY(학번)
,UNIQUE(연락처)
,CHECK(성별 IN ('남','여'))
,FOREIGN KEY (학과번호) REFERENCES 학과(학과번호)
);
```

3. 제약조건의 종류

■ [예제 8-14] 제약조건을 추가하여 과목 테이블을 생성하시오

■ 과목 테이블 명세서

컬럼명	데이터타입	제약조건
과목번호	고정문자형 5자	기본키
과목명	가변문자형 20자	필수 입력
학점	정수형	필수 입력, 2학점부터 4학점까지만 입력 가능
구분	가변문자형 20자	전공, 교양, 일반만 입력 가능

```
CREATE TABLE 과목
(
 과목번호 CHAR(5) PRIMARY KEY
,과목명 VARCHAR(20) NOT NULL
,학점 INT NOT NULL CHECK(학점 BETWEEN 2 AND 4)
,구분 VARCHAR(20) CHECK(구분 IN ('전공','교양','일반'))
);
```

1

3. 제약조건의 종류

■ [예제 8-15] 제약조건을 추가하여 수강_1 테이블을 생성하시오. 기본키는 수강년도, 수강학기, 학번, 과목번호를 모두 포함하시오

■ 수강_1 테이블 명세서

컬럼명	데이터타입	제약조건
수강년도	고정문자형 4자	필수 입력
수강학기	가변문자형 20자	필수 입력. 1학기, 2학기, 여름학기, 겨울학기만 입력 가능
학번	고정문자형 5자	필수 입력. 학생 테이블의 학번 참조
과목번호	고정문자형 5자	필수 입력. 과목 테이블의 과목번호 참조
성적	실수형	0부터 4.5까지 입력 가능

```
CREATE TABLE 수강_1
(
 수강년도 CHAR(4) NOT NULL
,수강학기 VARCHAR(20) NOT NULL CHECK(수강학기 IN ('1학기','2학기','여름학기','겨울학기'))
,학번 CHAR(5) NOT NULL
,과목번호 CHAR(5) NOT NULL
,성적 NUMERIC(3,1) CHECK(성적 BETWEEN 0 AND 4.5)
,PRIMARY KEY(수강년도, 수강학기, 학번, 과목번호)
,FOREIGN KEY (학번) REFERENCES 학생(학번)
,FOREIGN KEY (과목번호) REFERENCES 과목(과목번호)
);
```

1

3. 제약조건의 종류

■ [예제 8-16] 수강번호라는 대리키를 기본키로 하는 제약조건을 추가하여 수강_2 테이블을 생성하시오. 수강번호는 일련번호로 생성하시오.

```
CREATE TABLE 수강_2
(
 수강번호 INT PRIMARY KEY AUTO_INCREMENT
,수강년도 CHAR(4) NOT NULL
,수강학기 VARCHAR(20) NOT NULL CHECK(수강학기 IN ('1학기','2학기','여름학기','겨울학기'))
,학번 CHAR(5) NOT NULL
,과목번호 CHAR(5) NOT NULL
,성적 NUMERIC(3,1) CHECK(성적 BETWEEN 0 AND 4.5)
,FOREIGN KEY (학번) REFERENCES 학생(학번)
,FOREIGN KEY (과목번호) REFERENCES 과목(과목번호)
);
```

3. 제약조건의 종류

■ [예제 8-17] 학과 테이블에 다음과 같이 레코드 3건을 추가하시오. 오류가 발생한다면 이유가 무엇인지 생각해봅니다.

```
INSERT INTO 학과
VALUES ('AA','컴퓨터공학과','배경민');

INSERT INTO 학과
VALUES ('AA','소프트웨어학과','김남준');   ① 오류 발생

INSERT INTO 학과
VALUES ('CC','디자인융합학과','박선영');
```

3. 제약조건의 종류

■ [예제 8-17] 학과 테이블에 다음과 같이 레코드 3건을 추가하시오. 오류가 발생한다면 이유가 무엇인지 생각해봅니다.

```
INSERT INTO 학과
VALUES ('AA','컴퓨터공학과','배경민');

INSERT INTO 학과
VALUES ('AA','소프트웨어학과','김남준');   ① 오류 발생

INSERT INTO 학과
VALUES ('CC','디자인융합학과','박선영');
```

- **문장①**에서 오류 발생

```
Error Code: 1062. Duplicate entry 'AA'for key '학과.PRIMARY'
```

해결

```
INSERT INTO 학과
VALUES ('BB','소프트웨어학과','김남준');
```

▶ 실행결과

학과번호	학과명	학과장명
AA	컴퓨터공학과	배경민
BB	소프트웨어학과	김남준
CC	디자인융합학과	박선영

3. 제약조건의 종류

■ [예제 8-18] 학생 테이블에 다음과 같이 레코드 3건을 추가하시오. 오류가 발생한다면 이유가 무엇인지 생각해봅니다.

```
INSERT INTO 학생(학번, 이름, 생일, 학과번호)
VALUES ('S0001','이윤주','2020-01-30', 'AA');

INSERT INTO 학생(이름, 생일, 학과번호)
VALUES ('이승은','2021-02-23', 'AA');

INSERT INTO 학생(학번, 이름, 생일, 학과번호)
VALUES ('S0003','백재용','2018-03-31','DD');
```

❶ 오류 발생

❷ 오류 발생

3. 제약조건의 종류

■ [예제 8-18] 학생 테이블에 다음과 같이 레코드 3건을 추가하시오. 오류가 발생한다면 이유가 무엇인지 생각해봅니다.

```
INSERT INTO 학생(학번, 이름, 생일, 학과번호)
VALUES ('S0001','이윤주','2020-01-30', 'AA');

INSERT INTO 학생(이름, 생일, 학과번호)
VALUES ('이승은','2021-02-23', 'AA');

INSERT INTO 학생(학번, 이름, 생일, 학과번호)
VALUES ('S0003','백재용','2018-03-31','DD');
```

❶ 오류 발생

❷ 오류 발생

- **문장①**에서 오류 발생

```
Error Code: 1364. Field '학번' doesn't have a default value
```

해결

```
INSERT INTO 학생(학번, 이름, 생일, 학과번호)
VALUES ('S0002', '이승은','2020-01-30', 'AA');
```

3. 제약조건의 종류

■ [예제 8-18] 학생 테이블에 다음과 같이 레코드 3건을 추가하시오. 오류가 발생한다면 이유가 무엇인지 생각해봅니다.

```
INSERT INTO 학생(학번, 이름, 생일, 학과번호)
VALUES ('S0001','이윤주','2020-01-30', 'AA');

INSERT INTO 학생(이름, 생일, 학과번호)
VALUES ('이승은','2021-02-23', 'AA');

INSERT INTO 학생(학번, 이름, 생일, 학과번호)
VALUES ('S0003','백재용','2018-03-31','DD');
```

❶ 오류 발생

❷ 오류 발생

- **문장②**에서 오류 발생

```
Error Code: 1452. Cannot add or update a child row: a foreign key constraint fails('한빛학사','학생',CONSTRAINT '학생_ibfk_1'FOREIGN KEY ('학과번호') REFERENCES '학과' ('학과번호'))
```

해결

```
INSERT INTO 학생(학번, 이름, 생일, 학과번호)
VALUES ('S0003','백재용','2018-03-31','CC');
```

3. 제약조건의 종류

■ [예제 8-19] 과목 테이블에 다음과 같이 레코드 3건을 추가하시오. 오류가 발생한다면 이유가 무엇인지 생각해봅니다.

```
INSERT INTO 과목(과목번호, 과목명, 구분)
VALUES ('C0001','데이터베이스실습', '전공');

INSERT INTO 과목(과목번호, 과목명, 구분, 학점)
VALUES ('C0002','데이터베이스 설계와 구축', '전공', 5);

INSERT INTO 과목(과목번호, 과목명, 구분, 학점)
VALUES ('C0003','데이터 분석', '전공', 3);
```

① 오류 발생

② 오류 발생

3. 제약조건의 종류

■ [예제 8-19] 과목 테이블에 다음과 같이 레코드 3건을 추가하시오. 오류가 발생한다면 이유가 무엇인지 생각해봅니다.

```
INSERT INTO 과목(과목번호, 과목명, 구분)
VALUES ('C0001','데이터베이스실습', '전공');

INSERT INTO 과목(과목번호, 과목명, 구분, 학점)
VALUES ('C0002','데이터베이스 설계와 구축', '전공', 5);

INSERT INTO 과목(과목번호, 과목명, 구분, 학점)
VALUES ('C0003','데이터 분석', '전공', 3);
```

① 오류 발생

② 오류 발생

- **문장①**에서 오류 발생

```
Error Code: 1364. Field '학점' doesn't have a default value
```

해결

```
INSERT INTO 과목(과목번호, 과목명, 구분, 학점)
VALUES ('C0001','데이터베이스실습', '전공', 3);
```

■ [예제 8-19] 과목 테이블에 다음과 같이 레코드 3건을 추가하시오. 오류가 발생한다면 이유가 무엇인지 생각해봅니다.

```
INSERT INTO 과목(과목번호, 과목명, 구분)
VALUES ('C0001','데이터베이스실습', '전공');

INSERT INTO 과목(과목번호, 과목명, 구분, 학점)
VALUES ('C0002','데이터베이스 설계와 구축', '전공', 5);

INSERT INTO 과목(과목번호, 과목명, 구분, 학점)
VALUES ('C0003','데이터 분석', '전공', 3);
```

❶ 오류 발생

❷ 오류 발생

- **문장②**에서 오류 발생

```
Error Code: 3819. Check constraint '과목_chk_1'is violated.
```

해결

```
INSERT INTO 과목(과목번호, 과목명, 구분, 학점)
VALUES ('C0002','데이터베이스 설계와 구축', '전공', 3);
```

▶ 실행결과

과목번호	과목명	학점	구분
C0001	데이터베이스실습	3	전공
C0002	데이터베이스 설계와 구축	3	전공
C0003	데이터 분석	3	전공

3. 제약조건의 종류

■ [예제 8-20] 수강_1 테이블에 다음과 같이 레코드 4건을 추가하시오. 오류가 발생한다면 이유가 무엇인지 생각해봅니다.

```
INSERT INTO 수강_1(수강년도, 수강학기, 학번, 과목번호, 성적)
VALUES('2023','1학기','S0001','C0001',4.3);

INSERT INTO 수강_1(수강년도, 수강학기, 학번, 과목번호, 성적)
VALUES('2023','1학기','S0001','C0001',4.5);

INSERT INTO 수강_1(수강년도, 수강학기, 학번, 과목번호, 성적)
VALUES('2023','1학기','S0001','C0002',4.6);

INSERT INTO 수강_1(수강년도, 수강학기, 학번, 과목번호, 성적)
VALUES('2023','1학기','S0002','C0009',4.3);
```

① 오류 발생

② 오류 발생

③ 오류 발생

3. 제약조건의 종류

■ [예제 8-20] 수강_1 테이블에 다음과 같이 레코드 4건을 추가하시오. 오류가 발생한다면 이유가 무엇인지 생각해봅니다.

- **문장①**에서 오류 발생

```
Error Code: 1062. Duplicate entry '2023-1학기-S0001-C0001'for key '수강_1.PRIMARY'
```

- **문장②**에서 오류 발생

```
Error Code: 3819. Check constraint '수강_1_chk_2'is violated.
```

해결

```
INSERT INTO 수강_1(수강년도, 수강학기, 학번, 과목번호, 성적)
VALUES('2023','1학기','S0001','C0002',4.4);
```

- **문장③**에서 오류 발생

```
Error Code: 1452. Cannot add or update a child row: a foreign key constraint fails (`한빛학사`.`수강_1`, CONSTRAINT `수강_1_ibfk_2` FOREIGN KEY (`과목번호`) REFERENCES `과목` (`과목번호`))
```

해결

```
INSERT INTO 수강_1(수강년도, 수강학기, 학번, 과목번호, 성적)
VALUES('2023','1학기','S0002','C0002',4.3);
```

▶ 실행결과

수강년도	수강학기	학번	과목번호	성적
2023	1학기	S0001	C0001	4.3
2023	1학기	S0001	C0002	4.4
2023	1학기	S0002	C0002	4.3

■ [예제 8-21] 수강_1에서 넣은 것과 동일하게 수강_2 테이블에 2개의 레코드를 추가해보시오.

```
INSERT INTO 수강_2(수강년도, 수강학기, 학번, 과목번호, 성적)
VALUES('2023','1학기','S0001','C0001',4.3);

INSERT INTO 수강_2(수강년도, 수강학기, 학번, 과목번호, 성적)
VALUES('2023','1학기','S0001','C0001',4.5);
```

▶ 실행결과

수강번호	수강년도	수강학기	학번	과목번호	성적
1	2023	1학기	S0001	C0001	4.3
2	2023	1학기	S0001	C0001	4.5

4. 제약조건의 추가, 삭제

■ 제약조건의 추가

- 형식

```
ALTER TABLE 테이블명 ADD CONSTRAINT 제약조건(컬럼명);
```

■ [예제 8-22] 학생 테이블의 학번 컬럼에 CHECK 제약조건을 추가하시오.

- 제약조건: 모든 학번은 'S'로 시작한다.

```
ALTER TABLE 학생 ADD CONSTRAINT CHECK(학번 LIKE 'S%');
```

■ [예제 8-23] 학생 테이블에 설정되어 있는 제약조건 명세를 확인하시오.

```
SELECT *
FROM INFORMATION_SCHEMA.TABLE_CONSTRAINTS
WHERE CONSTRAINT_SCHEMA = '한빛학사'
AND TABLE_NAME = '학생';
```

▶ 실행결과

CONSTRAINT_CATALOG	CONSTRAINT_SCHEMA	CONSTRAINT_NAME	TABLE_SCHEMA	TABLE_NAME	CONSTRAINT_TYPE	ENFORCED
def	한빛학사	PRIMARY	한빛학사	학생	PRIMARY KEY	YES
def	한빛학사	연락처	한빛학사	학생	UNIQUE	YES
def	한빛학사	학생_ibfk_1	한빛학사	학생	FOREIGN KEY	YES
def	한빛학사	학생_chk_1	한빛학사	학생	CHECK	YES
def	한빛학사	학생_chk_2	한빛학사	학생	CHECK	YES

4. 제약조건의 추가, 삭제

■ 제약조건의 삭제

- 형식

```
ALTER TABLE 테이블명 DROP CONSTRAINT 제약조건명;
```

■ [예제 8-24] 학생 테이블에 설정되어 있는 제약조건 중 연락처에 설정되어 있는 제약조건을 삭제하시오.

```
ALTER TABLE 학생 DROP CONSTRAINT 연락처;
```

■ [예제 8-25] 성별 컬럼에 설정되어 있는 CHECK 제약조건을 삭제하시오.

- 학생 테이블에 걸려있는 CHECK 제약조건 2개를 삭제하고 학번에 설정했던 CHECK제약조건은 다시 설정

```
ALTER TABLE 학생 DROP CONSTRAINT 학생_chk_1;

ALTER TABLE 학생 DROP CONSTRAINT 학생_chk_2;

ALTER TABLE 학생 ADD CHECK (학번 LIKE 'S%');
```

▶ 실행결과

CONSTRAINT_CATALOG	CONSTRAINT_SCHEMA	CONSTRAINT_NAME	TABLE_SCHEMA	TABLE_NAME	CONSTRAINT_TYPE	ENFORCED
def	한빛학사	PRIMARY	한빛학사	학생	PRIMARY KEY	YES
def	한빛학사	학생_ibfk_1	한빛학사	학생	FOREIGN KEY	YES
def	한빛학사	학생_chk_1	한빛학사	학생	CHECK	YES

■ 제약조건명의 지정

- 제약조건에는 고유한 이름을 지정하여 관리할 수 있음
- 형식

```
컬럼 레벨:
컬럼명 데이터타입 [CONSTRAINT 제약조건명] 제약조건

테이블 레벨:
[CONSTRAINT 제약조건명] 제약조건
```

5. 제약조건명의 지정

■ [예제 8-26] [예제 8-13]에 있는 학생 테이블의 구조를 사용하여 학생_2 테이블을 생성하시오. 이때 제약조건명을 지정하여 생성하시오.

- 학생_2 테이블 생성

```
CREATE TABLE 학생_2
(
 학번 CHAR(5)
,이름 VARCHAR(20) NOT NULL
,생일 DATE NOT NULL
,연락처 VARCHAR(20)
,학과번호 CHAR(2)
,성별 CHAR(1)
,등록일 DATE DEFAULT(CURDATE())
,PRIMARY KEY(학번)
,CONSTRAINT UK_학생2_연락처 UNIQUE(연락처)
,CONSTRAINT CK_학생2_성별 CHECK(성별 IN ('남','여'))
,CONSTRAINT FK_학생2_학과번호 FOREIGN KEY (학과번호) REFERENCES 학과(학과번호)
);
```

5. 제약조건명의 지정

■ [예제 8-26] [예제 8-13]에 있는 학생 테이블의 구조를 사용하여 학생_2 테이블을 생성하시오. 이때 제약조건명을 지정하여 생성하시오.

- 제약조건 명세 확인

```
SELECT *
FROM INFORMATION_SCHEMA.TABLE_CONSTRAINTS
WHERE CONSTRAINT_SCHEMA = '한빛학사'
AND TABLE_NAME = '학생_2';
```

▶ 실행결과

CONSTRAINT_CATALOG	CONSTRAINT_SCHEMA	CONSTRAINT_NAME	TABLE_SCHEMA	TABLE_NAME	CONSTRAINT_TYPE	ENFORCED
def	한빛학사	PRIMARY	한빛학사	학생_2	PRIMARY KEY	YES
def	한빛학사	UK_학생2_연락처	한빛학사	학생_2	UNIQUE	YES
def	한빛학사	FK_학생2_학과번호	한빛학사	학생_2	FOREIGN KEY	YES
def	한빛학사	CK_학생2_성별	한빛학사	학생_2	CHECK	YES

■ 외래키 제약조건의 옵션

표 8-5 외래키 제약조건 옵션

	ON DELETE	ON UPDATE
CASCADE	부모 레코드 삭제 시 자식 레코드도 연쇄적으로 삭제된다.	부모 레코드의 기본키 수정 시 자식 레코드의 외래키 값도 연쇄적으로 수정된다.
SET NULL	부모 레코드 삭제 시 자식 레코드의 외래키 값이 NULL로 변경된다.	부모 레코드의 기본키 수정 시 자식 레코드의 외래키 값이 NULL로 변경된다.
SET DEFAULT	부모 레코드 삭제 시 자식 레코드의 외래키 값이 기본값으로 변경된다.	부모 레코드의 기본키 수정 시 자식 레코드의 외래키 값이 기본값으로 변경된다.
NO ACTION(기본값)	자식 레코드가 있으면 부모 레코드를 삭제할 수 없다.	자식 레코드가 있으면 부모 레코드의 기본키 값을 수정할 수 없다.

6. 외래키 제약조건의 옵션

■ [예제 8-27] 수강평가 테이블을 생성하시오. 이때 평가순번은 자동번호로 생성합니다. 또한 과목 테이블에서 레코드를 삭제하면 수강평가 테이블에서도 해당 과목에 대한 평가 레코드가 삭제되도록 옵션을 설정하시오.

■ 수강평가 테이블 명세서

컬럼명	데이터타입	제약조건 및 조건
평가순번	정수형	기본키, 자동번호
학번	고정문자형 5자	학생 테이블의 학번 참조
과목번호	고정문자형 5자	과목 테이블의 과목번호 참조
평점	정수형	0부터 5까지 입력 가능
과목평가	가변문자형 500자	
평가일시	날짜시간형	현재 일시 자동 등록

6. 외래키 제약조건의 옵션

■ [예제 8-27] 수강평가 테이블을 생성하시오. 이때 평가순번은 자동번호로 생성합니다. 또한 과목 테이블에서 레코드를 삭제하면 수강평가 테이블에서도 해당 과목에 대한 평가 레코드가 삭제되도록 옵션을 설정하시오.

- 수강평가 테이블 생성

```
CREATE TABLE 수강평가
(
 평가순번 INT PRIMARY KEY AUTO_INCREMENT
,학번 CHAR(5) NOT NULL
,과목번호 CHAR(5) NOT NULL
,평점 INT CHECK(평점 BETWEEN 0 AND 5)
,과목평가 VARCHAR(500)
,평가일시 DATETIME DEFAULT CURRENT_TIMESTAMP
,FOREIGN KEY (학번) REFERENCES 학생(학번)
,FOREIGN KEY (과목번호) REFERENCES 과목(과목번호) ON DELETE CASCADE
);
```

6. 외래키 제약조건의 옵션

■ [예제 8-28] 수강평가 테이블에 4건의 레코드를 추가하시오.

```
INSERT INTO 수강평가(학번, 과목번호, 평점, 과목평가)
VALUES('S0001','C0001',5,'SQL학습에 도움이 되었습니다.')
     ,('S0001','C0003',5,'SQL 활용을 배워서 좋았습니다.')
     ,('S0002','C0003',5,'데이터 분석에 관심이 생겼습니다.')
     ,('S0003','C0003',5,'머신러닝과 시각화 부분이 유용했습니다.');
```

▶ 실행결과

평가순번	학번	과목번호	평점	과목평가	평가일시
1	S0001	C0001	5	SQL학습에 도움이 되었습니다.	2023-12-27 11:25:38
2	S0001	C0003	5	SQL 활용을 배워서 좋았습니다.	2023-12-27 11:25:38
3	S0002	C0003	5	데이터 분석에 관심이 생겼습니다.	2023-12-27 11:25:38
4	S0003	C0003	5	머신러닝과 시각화 부분이 유용했습니다.	2023-12-27 11:25:38

6. 외래키 제약조건의 옵션

■ [예제 8-29] 과목 테이블에서 'C0003' 과목을 삭제하시오. 그리고 과목 테이블과 수강평가 테이블에서 결과를 확인해봅니다.

```
DELETE FROM 과목 WHERE 과목번호 = 'C0003';
```

```
SELECT * FROM 과목;
SELECT * FROM 수강평가;
```

▶ 실행결과

• 과목 테이블

과목번호	과목명	학점	구분
C0001	데이터베이스실습	3	전공
C0002	데이터베이스 설계와 구축	3	전공

• 수강평가 테이블

평가순번	학번	과목번호	평점	과목평가	평가일시
1	S0001	C0001	5	SQL학습에 도움이 되었습니다.	2023-12-27 11:25:38

6. 외래키 제약조건의 옵션

■ [예제 8-30] 과목 테이블에서 'C0001' 과목을 삭제하시오. 오류가 난다면 이유를 생각해봅니다.

```
DELETE FROM 과목 WHERE 과목번호 = 'C0001';
```

```
Error Code: 1451. Cannot delete or update a parent row: a foreign key constraint fails ('한빛학사'.'수강_1', CONSTRAINT '수강_1_ibfk_2' FOREIGN KEY ('과목번호') REFERENCES '과목' ('과목번호'))
```

6. 외래키 제약조건의 옵션

확인문제

게시판 테이블을 생성하려고 합니다. 다음 조건에 맞게 빈칸을 채우시오.

조건

1. 번호는 일련번호가 자동으로 삽입된다.
2. 작성일은 시스템의 현재 날짜와 시간이 자동 등록된다.
3. 작성자는 학생 테이블의 학번 또는 NULL만 허용된다.

```
CREATE TABLE 게시판
(
 번호 INT PRIMARY KEY [ ① ]
,제목 VARCHAR(50) NOT NULL
,내용 VARCHAR(1000)
,작성자 CHAR(5)
,작성일 DATETIME [ ② ] CURRENT_TIMESTAMP
,CONSTRAINT fk_게시판_작성자 FOREIGN KEY(작성자) [ ③ ] 학생(학번)
);
```

정답

① AUTO_INCREMENT ② DEFAULT ③ REFERENCES

점검문제

점검문제

문제 1

제품 테이블의 재고 컬럼에 CHECK 제약조건을 추가하시오.

- 제약조건: 재고는 0보다 크거나 같아야 합니다.

문제 2

제품 테이블에 재고금액 컬럼을 추가하시오. 이때 재고금액은 '단가 * 재고'가 자동 계산되어 저장되도록 합니다.

문제 3

제품 테이블에서 제품 레코드를 삭제하면 주문세부 테이블에 있는 관련 레코드도 함께 삭제되도록 주문세부 테이블의 제품번호 컬럼에 외래키 제약조건과 옵션을 설정하시오.

Thank you!

한빛아카데미 Hanbit Academy, Inc.