

학습목표

1. 스키마 정의
2. 자료형
3. 제약조건

학습내용

- 스키마 정의를 이해할 수 있습니다.
- 테이블의 각 컬럼의 자료형을 지정할 수 있습니다.
- 테이블의 제약조건을 지정할 수 있습니다.

사전퀴즈

1. 스키마란 테이블이나 DB의 구조와 데이터타입을 정의한 것을 말한다.

정답 : O

해설 : 스키마는 테이블/DB(데이터베이스)의 구조나 데이터타입을 정의한다.

2. AUTO_INCREMENT는 기본키 컬럼에만 허용한다.

정답 : O

해설 : **AUTO_INCREMENT**는 기본키 컬럼에만 사용할 수 있다.

수업

1. 스키마 정의

DDL (Data Definition Language)

- 데이터베이스와 테이블을 CRUD (Create, Retrieve, Update, Delete)
- 테이블에 대한 정보는 메타데이터(Metadata)로 데이터사전(Data Dictionary)에 저장, 관리된다.

데이터베이스(DB) 생성

- CREATE DATABASE 데이터베이스명;

테이블생성

- CREATE TABLE 테이블명(컬럼명1 데이터타입(크기), 컬럼명2 ...);

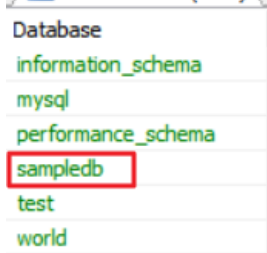
예제 1-1

sampleDB 데이터베이스를 정의하고 생성하시오

```

1 create database sampleDB;
2
3 show databases;
4

```



SCHEMATA (1x6)

Database
information_schema
mysql
performance_schema
sampledb
test
world

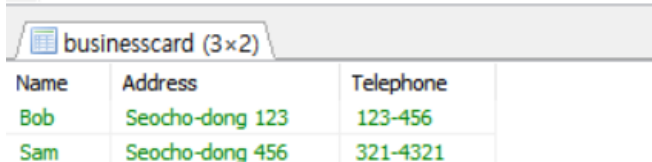
예제1-2

컬럼으로 이름(Name), 주소(Address), 전화번호(Telephone)을 가지는 BusinessCard 테이블을 정의하고 생성하시오.
단, 이름, 주소, 전화번호는 문자열(Varchar) 최대길이 255자로 지정함

```

5 create table BusinessCard(
6     Name varchar(255),
7     Address varchar(255),
8     Telephone varchar(255)
9 )
10 ;
11
12 insert into BusinessCard
13 values('Bob', 'Seocho-dong 123', '123-456')
14 ;
15
16 insert into BusinessCard
17 values('Sam', 'Seocho-dong 456', '321-4321')
18 ;
19
20 select *
21 from BusinessCard
22 ;

```



businesscard (3x2)

Name	Address	Telephone
Bob	Seocho-dong 123	123-456
Sam	Seocho-dong 456	321-4321

2. 자료형

프로그램은 같은 목적이라도 효율적으로 동작하는 것이 더 좋은 프로그램이다.

컴퓨터 알고리즘 학문에서는 이를 평가하기 위해서 시간복잡도와 공간복잡도 라는 개념을 사용한다.

즉, "더 작은 공간을 사용하면서", "더 빠르게 처리할 수 있는" 프로그램을 만들기 위해서 데이터 자료형을 사용하는 것이다.

작은 공간에 넣을수록 연산(SELECT 등의 연산)이 빨라지고 공간도 적게 차지한다.

정수형(부호 있음/부호 없음)

1) TINYINT

- 자료형의 크기 : 1바이트 (1Byte, $2^8 = 8$ bit)
- 범위 : -128 ~ 127 (UNSIGNED일 경우 0 ~ 255)

2) INT

- 자료형의 크기 : 4바이트 (4Byte, $2^{32} = 32$ bit)
- 범위 : -21억 ~ 21억 (UNSIGNED일 경우 0 ~ 43억)

3) BIGINT

- 자료형의 크기 : 8바이트 (8Byte, $2^{64} = 64$ bit)
- 범위 : -9경 ~ 9경 (UNSIGNED일 경우 0 ~ 18경)

실수형(길이, 소수점 이하 자리수)

- 1) FLOAT(size, d)
- 2) DOUBLE(size, d)
- 3) DECIMAL(size, d)

문자열

- 1) CHAR 고정길이 문자열 (최대 255자)
- 2) VARCHAR 가변길이 문자열 (최대 65,535자)

TEXT 문자열

- 1) TEXT (최대 65,535자)
- 2) MEDIUMTEXT (최대 16,777,215자)
- 3) LONGTEXT (최대 4,294,967,295자)

BLOB(Binary Large Object)

이미지나 동영상 파일을 처리하기 위한 데이터 타입

- 1) BLOB (최대 65,535 바이트)
- 2) MEDIUMBLOB (최대 16,777,215 바이트)
- 3) LARGEBLOB (최대 4,294,967,295 바이트)

시간관련

- 1) DATE (YYYY-MM-DD)
- 2) TIME (HH:MI:SS)
- 3) DATETIME (YYYY-MM-DD HH:MI:SS)
- 4) TIMESTAMP (YYYY-MM-DD HH:MI:SS)

MariaDB에서는 1970-01-01 00:00:00 이후(Unix Epoch) 시간을 표현할 수 있다.

참고한 사이트 : <https://blog.lael.be/post/115>

3. 제약조건(Constraint)

입력 데이터의 제약조건을 걸어 해당되지 않는 데이터는 입력되지 않음

NOT NULL	데이터가 NULL 값을 받아들이지 않음
UNIQUE	테이블에 동일한 값이 입력되어 있을 경우 받아들이지 않음
PRIMARY KEY	기본키 제약조건(UNIQUE, NOT NULL 조건)
FOREIGN KEY	외래키 제약조건
CHECK	입력값 체크(예: Age >= 0)
DEFAULT	컬럼값이 입력되지 않으면 기본값을 입력 * 주의 * insert를 할 때, default 값을 넣고 싶은 컬럼의 컬럼명과 값을 제외해야 함 컬럼명을 명시하고 값에 null을 쓰면 default 값이 적용되지 않음

BusinessCard 테이블의 이름(Name)이 NULL이면 안됨

→ Name 컬럼에 not null 조건이 걸려있으므로 Name컬럼에 null 값을 insert 하려고하면 에러가 발생한다.

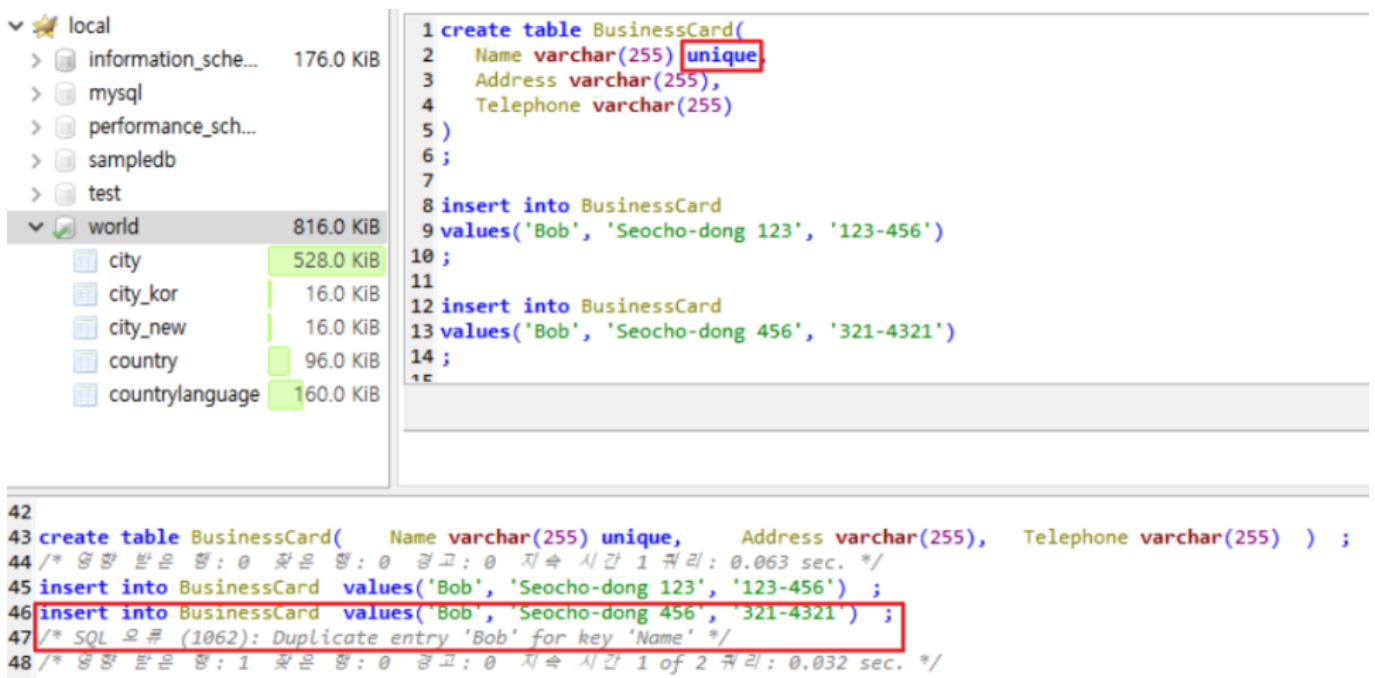


```
1 create table BusinessCard(  
2   Name varchar(255) not null,  
3   Address varchar(255),  
4   Telephone varchar(255)  
5 )  
6 ;  
7  
8 insert into BusinessCard  
9 values('Bob', 'Seocho-dong 123', '123-456')  
10 ;  
11  
12 insert into BusinessCard  
13 values(null, 'Seocho-dong 456', '321-4321')  
14 ;  
15  
29  
30 create table BusinessCard(   Name varchar(255) not null,   Address varchar(255),   Telephone varchar(255) ) ;  
31 /* 실행 받은 행: 0 읽은 행: 0 경고: 0 지속 시간 1 쿼리: 0.047 sec. */  
32 insert into BusinessCard values('Bob', 'Seocho-dong 123', '123-456') ;  
33 /* 실행 받은 행: 1 읽은 행: 0 경고: 0 지속 시간 1 쿼리: 0.188 sec. */  
34 insert into BusinessCard values(null, 'Seocho-dong 456', '321-4321') ;  
35 /* SQL 오류 (1048): Column 'Name' cannot be null */  
36 /* 실행 받은 행: 0 읽은 행: 0 경고: 0 지속 시간 0 of 1 쿼리: 0.000 sec. */
```

예제 3-2

BusinessCard 테이블의 Name이 동일한 값을 허용하지 않도록 지정하시오

→ Name 컬럼에 unique조건이 걸려있으므로 Name컬럼에 같은 값을 insert 하려고하면 에러가 발생한다.



```
1 create table BusinessCard(  
2   Name varchar(255) unique,  
3   Address varchar(255),  
4   Telephone varchar(255)  
5 )  
6 ;  
7  
8 insert into BusinessCard  
9 values('Bob', 'Seocho-dong 123', '123-456')  
10 ;  
11  
12 insert into BusinessCard  
13 values('Bob', 'Seocho-dong 456', '321-4321')  
14 ;  
15  
42  
43 create table BusinessCard(   Name varchar(255) unique,   Address varchar(255),   Telephone varchar(255) ) ;  
44 /* 실행 받은 행: 0 읽은 행: 0 경고: 0 지속 시간 1 쿼리: 0.063 sec. */  
45 insert into BusinessCard values('Bob', 'Seocho-dong 123', '123-456') ;  
46 insert into BusinessCard values('Bob', 'Seocho-dong 456', '321-4321') ;  
47 /* SQL 오류 (1062): Duplicate entry 'Bob' for key 'Name' */  
48 /* 실행 받은 행: 1 읽은 행: 0 경고: 0 지속 시간 1 of 2 쿼리: 0.032 sec. */
```

예제 3-3

BusinessCard 테이블에서 ID컬럼을 기본키로 지정함

```

1 create table BusinessCard(
2     id int primary key,
3     Name varchar(255),
4     Address varchar(255),
5     Telephone varchar(255)
6 )
7 ;
8
9 insert into BusinessCard
10 values(1, 'Bob', 'Seocho-dong 123', '123-456')
11 ;
12
13 insert into BusinessCard
14 values(2, 'Bob', 'Seocho-dong 456', '321-4321')
15 ;
16
17 select *
18 from BusinessCard
19 ;

```

businesscard (4×2)			
id	Name	Address	Telephone
1	Bob	Seocho-dong 123	123-456
2	Bob	Seocho-dong 456	321-4321

예제 3-4

Salary 테이블의 BusinessCard_ID를 외부키로 지정함

* 주의 *

외부키가 설정되어 있는 Salary 테이블에 데이터를 BusinessCard 테이블보다 먼저 넣으면 에러가 발생한다.

```

1 create table BusinessCard(
2     id int,
3     Name varchar(255),
4     Address varchar(255),
5     Telephone varchar(255),
6     primary key(id)
7 )
8 ;
9
10 create table Salary(
11     id int,
12     salary_amount int,
13     business_card_id int not null,
14     primary key(id),
15     foreign key(business_card_id) references BusinessCard(id)
16 )
17 ;
18
19 insert into BusinessCard
20 values(1, 'Bob', 'Seocho-dong 123', '123-456')
21 ;
22
23 insert into Salary
24 values(1, 5000, 1)
25 ;
26
27
28 select *
29 from BusinessCard join Salary
30 on BusinessCard.id = Salary.business_card_id
31 ;
32

```

결과 #1 (7×1)						
id	Name	Address	Telephone	id	salary_amount	business_card_id
1	Bob	Seocho-dong 123	123-456	1	5,000	1

예제 3-5

BusinessCard 테이블의 Age 값은 0이상이어야 함

→ Age 컬럼에 check(Age > 0)조건이 걸려있으므로 Age컬럼에 0보다 같거나 작은 값을 넣으려고 하면 에러가 발생한다.

```
1 create table BusinessCard(
2   Name varchar(255),
3   Address varchar(255),
4   Telephone varchar(255),
5   Age int,
6   check(Age > 0)
7 )
8 ;
9
10 insert into BusinessCard
11 values('Bob', 'Sinsa-dong 345', '010-1111-2222', 0);
12 ;
13
```

```
81
82 create table BusinessCard(   Name varchar(255),   Address varchar(255),   Telephone varchar(255),   Age int,   check(Age > 0)  ) ;
83 /* 실행 받은 행: 0   읽은 행: 0   경고: 0   지속 시간 1 쿼리: 0.031 sec. */
84 insert into BusinessCard values('Bob', 'Sinsa-dong 345', '010-1111-2222', 0) ;
85 /* SQL 오류 (4025): CONSTRAINT `CONSTRAINT 1` failed for `world`.`businesscard` */
86 /* 실행 받은 행: 0   읽은 행: 0   경고: 0   지속 시간 0 of 1 쿼리: 0.000 sec. */
```

예제 3-6

BusinessCard 테이블의 주소(Address)값이 지정되지 않으면 'SEOUL'로 입력되도록 하시오

* 주의 *

Address 값에 null을 넣으면 default값인 SEOUL이 입력되지 않음

default 값이 입력되도록 하려면 컬럼명과 값을 쓰면 안됨

```
1 create table BusinessCard(
2   Name varchar(255),
3   Address varchar(255) default 'SEOUL',
4   Telephone varchar(255)
5 )
6 ;
7
8 insert into BusinessCard
9 values('Bob', null, '010-1111-2222')
10 ;
11
12 insert into BusinessCard(name, telephone)
13 values('Bob2', '010-1111-2222')
14 ,
15
16 select *
17 from BusinessCard
18 ;
19
```

Name	Address	Telephone
Bob	(NULL)	010-1111-2222
Bob2	SEOUL	010-1111-2222

예제 3-7

BusinessCard의 ID값을 자동 증가되도록 지정하시오

* 주의 *

- auto increment (X), auto_increment (O)

- auto_increment는 기본키에만 적용가능

- 테이블에 데이터를 insert할 때

1) auto_increment를 적용한 컬럼을 제외한 컬럼명들을 모두 적어주거나

2) 컬럼명을 모두 적는 대신 테이블명을 적고, 테이블에 있는 모든 컬럼의 순서에 맞게 값을 넣어야 함

3) 테이블명만 적고, auto_increment가 적용된 컬럼에 대한 값을 제외하면 에러발생

4) auto_increment가 적용된 컬럼의 가장 마지막 값에 1이 자동증가되어 값으로 저장됨

```
1 create table BusinessCard(
2   id int auto_increment,
3   Name varchar(255),
4   Address varchar(255),
5   Telephone varchar(255),
6   primary key(id)
7 )
8 ;
9
10 insert into BusinessCard(Name, Address, Telephone)
11 values('Bob', 'SEOUL', '010-1111-2222')
12 ;
13
14 insert into BusinessCard
15 values(3, 'Bob2', 'SEOUL', '010-1111-2222')
16 ;
17
18 insert into BusinessCard
19 values('Bob3', 'SEOUL', '010-1111-2222')
20 ;
21
22 insert into BusinessCard(Name, Address, Telephone)
23 values('Bob4', 'SEOUL', '010-1111-2222')
24 ;
25
26 select *
27 from BusinessCard
28 ;
```

id	Name	Address	Telephone
1	Bob	SEOUL	010-1111-2222
3	Bob2	SEOUL	010-1111-2222
4	Bob4	SEOUL	010-1111-2222

```
144 insert into BusinessCard values(3, 'Bob2', 'SEOUL', '010-1111-2222') ;
145 /* 영향 받은 행: 2  찾아낸 행: 0  경고: 0  지속 시간 3 쿼리: 0.032 sec. */
146 insert into BusinessCard values('Bob3', 'SEOUL', '010-1111-2222') ;
147 /* SQL 오류 (1136): Column count doesn't match value count at row 1 */
148 /* 영향 받은 행: 0  찾아낸 행: 0  경고: 0  지속 시간 0 of 1 쿼리: 0.000 sec. */
149 insert into BusinessCard(Name, Address, Telephone) values('Bob4', 'SEOUL', '010-1111-2222') ;
150 /* 영향 받은 행: 1  찾아낸 행: 0  경고: 0  지속 시간 1 쿼리: 0.016 sec. */
151 select * from BusinessCard ;
152 /* 영향 받은 행: 0  찾아낸 행: 3  경고: 0  지속 시간 1 쿼리: 0.000 sec. */
```

MariaDB의 제약조건(Constraint)중에 자동증가(Auto Increment)와 CoC네이밍과의 관계?

전문가 의견

스프링 프레임워크 등에서 사용하는 CoC(Convention over Configuration : 설정보다 관례) 명명규칙에는 DB 테이블을 정의할 때 몇가지 가이드라인이 있습니다.

클래스명(Sample) → 테이블명(sample"s")

기본키명 id(integer, auto increment)

외래 키명 테이블명_id(sample_id : sample 테이블을 가리키는 외래 키)

관련기술로 객체-관계매핑(ORM : Object relational Mapping) 기술이 있습니다.

CoC에서는 클래스이름만 지정하면 나머지 값들이 자동으로 정해집니다.

특히 테이블 기본키 컬럼을 확인 안해도 되는 편리한 점이 있습니다.