

6주차 3차시 SQL개요와 데이터 정의(1/3)

【학습목표】

1. SQL의 역할을 이해하고, 기능을 설명할 수 있다.
2. SQL을 사용한 데이터 정의 방법 중 속성 정의 방법을 설명할 수 있다.

학습내용1 : SQL 개요

1. SQL (Structured Query Language)

- 데이터베이스에 데이터 삽입/삭제/수정/검색 방법
 - 관계 대수나 관계 해석을 사용 ==> 일반사용자가 사용하기 어렵다.
 - 대안 : SQL 사용
- SQL
 - 관계 데이터베이스를 위한 표준 질의어
 - 사용자가 처리를 원하는 데이터가 무엇인지만 제시
 - 데이터를 얻는 방법에 대하여 언급하지 않음

==> 비절차적 데이터 언어의 특성

2. SQL의 역사

- 1974년 SEQUEL (Structured English QUery Language)에서 유래
 - IBM 연구소의 연구용 관계 데이터베이스 관리시스템 (System R)을 위한 언어
- 이후 개발된 관계 데이터베이스 관리 시스템은 다른 질의어를 사용
- 1986년 미국 표준 연구소인 ANSI와 국제 표준화 기구인 ISO에서 SQL을 관계 데이터베이스의 표준 질의어로 채택하고 표준화 작업 진행
 - 1986년 표준 SQL : SQL-86 또는 SQL1
 - 1992년 표준 SQL : SQL-92 또는 SQL2
 - 1999년 표준 SQL : SQL-99 또는 SQL3

3. SQL 사용 방법

- DBMS 내에서 대화식으로 질의
- C, C++, Java와 같은 언어로 작성한 프로그램에 삽입하여 사용

4. SQL의 분류

- 데이터정의어 (DDL, Data Definition Language)
 - . 테이블을 생성, 변경, 삭제하는 기능
- 데이터조작어 (DML, Data Manipulation Language)
 - . 테이블에 데이터를 삽입, 수정, 삭제, 검색하는 기능
- 데이터제어어 (DCL, Data Control Language)
 - . 보안을 위해 데이터에 대한 접근 및 사용 권한을 사용자별로 부여, 취소하는 기능
 - . 데이터베이스 관리자가 주로 사용

5. SQL의 기능

- 주기능 : 검색을 위한 질의 작성용 비절차적 데이터 조작어
- 보조기능 : 데이터 정의 및 제어

학습내용2 : 데이터의 정의

1. 실습용 데이터베이스

A. Customer 테이블

Customer

account	name	grade	credit	address
bank	홍길동	Gold	30000	서울 종로구
apple	이남이	VIP	5000	경기 용인시
pencil	김돌	Silver	350	경기 수원
eagle	박세째	Gold	2450	부산 남구
bird	구선두	New	0	충남 천안시
king	오나라	Gold	15000	서울 성북구

그림. 실습 질의용 Customer 릴레이션

B. Items 테이블

Items

code	name	stocks	price
E01	김	30	1500
E02	단무지	45	3250
E03	햄	15	7000
E04	우유	25	4560
E05	주스	20	2300
E06	라면	30	2560

그림. 실습 질의용 Items 릴레이션

C. Orders 테이블

Orders

num	customer	item	qty	date	saddr
00001	apple	E06	15	2015-06-01	경기 용인시
00002	apple	E05	2	2015-06-01	경기 용인시
00003	eagle	E01	3	2015-06-03	부산 남구
00004	bank	E04	5	2015-06-03	서울 종로구
00005	bird	E02	10	2015-06-04	충남 천안시
00006	bird	E03	1	2015-06-04	충남 천안시

그림. 실습 질의용 Orders 릴레이션

2. 테이블 생성 - 속성 정의

표. SQL의 데이터 정의 기능

데이터 정의	명령어
테이블 생성	CREATE TABLE
테이블 변경	ALTER TABLE
테이블 삭제	DROP TABLE

A. 테이블 생성시 정의할 사항

- ① 테이블 이름
- ② 속성 이름 및 데이터 타입, 제약 사항
- ③ 기본키, 대체키, 외래키의 정의
- ④ 데이터 무결성을 위한 제약조건 정의

표. CREATE TABLE 기본 형식

```
CREATE TABLE 테이블이름 (
  ② 속성이름 데이터타입 [NOT NULL] [DEFAULT 기본값],
  ③ [, PRIMARY KEY (속성리스트)]
  ③ [, UNIQUE (속성리스트)]
  ③ [, FOREIGN KEY (속성리스트) REFERENCES 테이블이름(속성리스트)]
  ④ [, ON DELETE 옵션]
  ④ [, ON UPDATE 옵션]
  ④ [, CONSTRAINT 이름]
  ④ [, CHECK (조건)]
);
```

* [] : 생략 가능

- SQL 질의문
 - . 세미콜론(;)으로 문장의 끝을 표시
 - . 대소문자 구분하지 않음

B. 속성 정의

- 테이블을 구성하는 속성 정의
 - . 데이터 타입 정의
 - . NULL 값 허용 여부 정의
 - . 기본 값 필요 여부

C. 속성의 데이터 타입

표. 데이터 타입 (문자)

데이터 타입	설명
CHAR(size)	고정길이 (size)의 문자열 저장(255문자까지 저장)
VARCHAR(size)	최대길이가 n인 가변길이 문자열 저장 (255문자까지 저장, 255보다 클 경우, TEXT로 변환됨)
TINYTEXT	최대 255문자까지 문자열 저장
TEXT	최대 65,535문자까지 문자열 저장
BLOB	BLOB(Binary Large Objects). 65,535 바이트의 데이터를 저장
MEDIUMTEXT	최대 16,777,215 (16M)문자까지 문자열 저장
MEDIUMBLOB	최대 16,777,215 (16M)바이트의 데이터를 저장
LONGTEXT	최대 4,294,967,295 (4G) 문자까지 문자열 저장
LOBLOB	최대 4,294,967,295 (4G) 바이트까지 데이터를 저장
ENUM(x,y,z,etc)	65,535개의 가능한 값을 가진 리스트. List에 없는 값을 삽입시, 공백값이 삽입됨
SET	64개 리스트 항목을 포함

표. 데이터 타입 (숫자)

데이터 타입	설명	크기
TINYINT(size)	-128 ~ 127 (부호없는 정수* : 0 ~ 255) 최대 숫자를 괄호안에 지정할 수 있다.	2 ⁸
SMALLINT(size)	-32768 ~ 32767 (부호없는 정수* : 0 ~ 65535)	2 ¹⁶
MEDIUMINT(size)	-8388608 ~ 8388607 (부호없는 정수* : 0 ~ 16777215)	2 ²⁴
INT(size)	-2147483648 ~ 2147483647 (부호없는 정수* : 0 ~ 4294967295)	2 ³²
BIGINT(size)	-9223372036854775808 to 9223372036854775807 (부호없는 정수* : 0 to 18446744073709551615)	2 ⁶⁴
FLOAT(size,d)	실수 (size : 최대 자릿수, d : 소수점의 오른쪽 자릿수)	
DOUBLE(size, d)	실수 (size : 최대 자릿수, d : 소수점의 오른쪽 자릿수)	
DECIMAL(size,d)	고정 소수점으로 표현된 실수의 문자열	

* 부호없는 정수 : UNSIGNED 옵션으로 정수 타입을 지정

표. 데이터 타입 (날짜)

데이터 타입	설명
DATE()	YYYY-MM-DD 형식의 날짜 (1000-01-01 ~ 9999-12-31)
DATETIME(*)	YYYY-MM-DD HH:MI:SS 형식의 날짜와 시간 (1000-01-01 00:00:00 ~ 9999-12-31 23:59:59)
TIMESTAMP(*)	타임스탬프, 유닉스 기점 (1970-01-01 00:00:00 UTC) 이후의 초 범위 : 1970-01-01 00:00:01 UTC ~ 2038-01-09 03:14:07 UTC INSERT와 UPDATE query에서 자동으로 현재 날짜와 시간으로 설정된다. 여러가지 형식 제공 : YYYYMMDDHHMISS, YYMMDDHHMISS, YYYYMMDD, YYMMDD
TIME()	HH:MI:SS 형식의 시간
YEAR()	2자리 또는 4자리의 년도 (4자리 범위 : 1901 ~ 2155, 2자리 범위 : 70 ~ 69 = 1970 ~ 2069)

* DATETIME과 TIMESTAMP가 같은 형식을 반환하지만, 다르게 동작됨.

D. NOT NULL

- 속성의 NULL 값을 허용하지 않음을 의미
- 반드시 값이 입력되어야 하는 속성에 사용
- 인터넷 상의 회원가입시 필수항목
- 예 : account VARCHAR(20) NOT NULL

E. DEFAULT

- 속성의 기본 값을 지정
- DEFAULT를 지정하지 않고, 값이 입력되지 않은 경우 NULL값으로 자동 저장
- 예 : credit INT DEFAULT 0
- 예 : grade VARCHAR(10) DEFAULT 'NEW'

[TIP]

- 문자열, 날짜 데이터 : 작은 따옴표로 묶어야 함
- 작은 따옴표 내의 문자열 : 대소문자 구분됨
예 : grade VARCHAR(10) DEFAULT 'NEW'
grade VARCHAR(10) DEFAULT 'New'

F. AUTO_INCREMENT

- 테이블에 새로운 튜플이 삽입될 때, 자동으로 DBMS에서 유일한 번호를 생성
- AUTO_INCREMENT인 속성은 기본키로 지정되어야 함.
- 1부터 시작하며, 새로운 튜플(레코드) 추가시 1씩 증가
- 형식 : 속성이름 INT NOT NULL AUTO_INCREMENT
- 다른 값부터 시작하도록 설정할 수 있다.
 . 형식 : ALTER TABLE 테이블명 AUTO_INCREMENT=초기값
- 새로운 레코드를 삽입할 때, AUTO_INCREMENT로 지정된 속성값은 입력하지 않아도 됨
- AUTO_INCREMENT로 지정된 속성의 현재 값은 변경할 수 없음
- 레코드 삽입시 SQL 구조
 - * INSERT INTO 테이블명 (속성1, ... 속성N) values (속성값1, ... 속성값N)
 - * AUTO_INCREMENT로 지정된 속성은 제외할 수 있음

G. 실습용 데이터베이스 데이터 정의

- Customer 테이블

3. 실습용 데이터베이스 데이터 정의 (Customer 테이블)

- * 기본 테이블 생성

표. CREATE TABLE 예 : Customer

```
CREATE TABLE Customer (
  account  varchar(10) NOT NULL,
  name     varchar(10) NOT NULL,
  grade    varchar(6) NOT NULL DEFAULT 'New',
  credit   int DEFAULT 0,
  address  varchar(100),
  PRIMARY KEY (account)
);
```

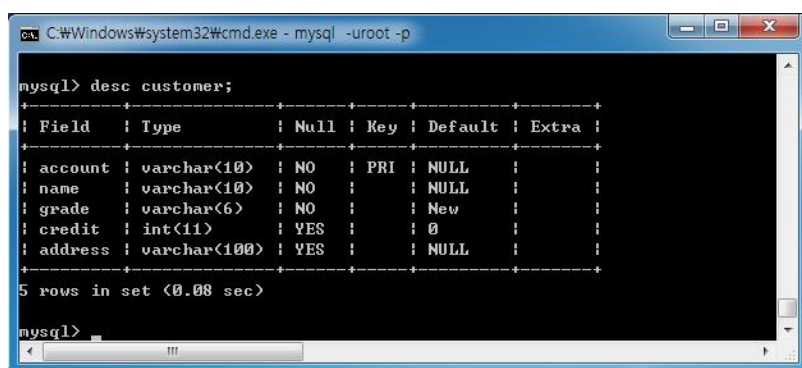


그림. MySQL 적용 후 Customer 릴레이션 구조

* 데이터 입력

표. customer 릴레이션 데이터 입력

```

insert into customer values ( 'bank', '홍길동', 'Gold', 30000, '서울 종로구');
insert into customer values ( 'apple', '이남이', 'VIP', 5000, '경기 용인시');
insert into customer values ( 'pencil', '김돌', 'Silver', 350, '경기 수원');
insert into customer values ( 'eagle', '박세째', 'Gold', 2450, '부산 남구');
insert into customer values ( 'bird', '구선두', '', 0, '충남 천안시');
insert into customer values ( 'king', '오나라', 'Gold', 15000, '서울 성북구');

```

```

C:\Windows\system32\cmd.exe - mysql -uroot -p
mysql> select * from customer;
+-----+-----+-----+-----+-----+
| account | name  | grade | credit | address |
+-----+-----+-----+-----+-----+
| apple  | 이남이 | VIP   | 5000   | 경기 용인시 |
| bank   | 홍길동 | Gold  | 30000  | 서울 종로구 |
| bird   | 구선두 |      | 0      | 충남 천안시 |
| eagle  | 박세째 | Gold  | 2450   | 부산 남구   |
| king   | 오나라 | Gold  | 15000  | 서울 성북구 |
| pencil | 김돌   | Silver | 350    | 경기 수원   |
+-----+-----+-----+-----+-----+
6 rows in set (0.00 sec)

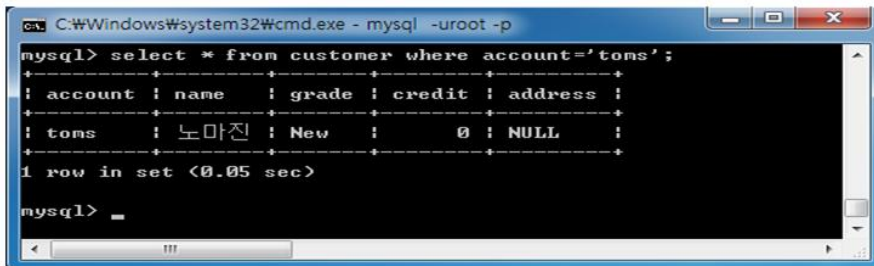
mysql> _

```

그림. customer 릴레이션 데이터 확인

* 데이터입력 : NOT NULL 속성 데이터만 입력시

```
insert into customer ( account, name ) values ( 'toms', '노마진');
```



```

C:\Windows\system32\cmd.exe - mysql -uroot -p
mysql> select * from customer where account='toms';
+-----+-----+-----+-----+-----+
| account | name   | grade | credit | address |
+-----+-----+-----+-----+-----+
| toms    | 노마진 | New   | 0      | NULL    |
+-----+-----+-----+-----+-----+
1 row in set (0.05 sec)

mysql>
  
```

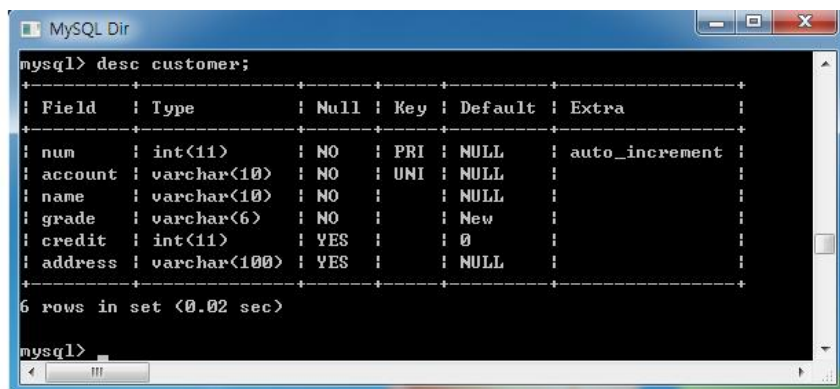
그림.customer 릴레이션 데이터 입력 (DEFAULT)

* AUTO_INCREMENT 적용시

표. CREATE TABLE 예 : Customer

```

CREATE TABLE Customer (
  num          int NOT NULL AUTO_INCREMENT,
  account      varchar(10) NOT NULL,
  name         varchar(10) NOT NULL,
  grade        varchar(6) NOT NULL DEFAULT 'New',
  credit       int DEFAULT 0,
  address      varchar(100),
  PRIMARY KEY ( num ),
  UNIQUE ( account )
);
  
```



```

MySQL Dir
mysql> desc customer;
+-----+-----+-----+-----+-----+-----+
| Field | Type          | Null | Key | Default | Extra           |
+-----+-----+-----+-----+-----+-----+
| num   | int(11)       | NO   | PRI | NULL    | auto_increment |
| account | varchar(10)  | NO   | UNI | NULL    |                 |
| name  | varchar(10)   | NO   |     | NULL    |                 |
| grade | varchar(6)    | NO   |     | New     |                 |
| credit | int(11)       | YES  |     | 0       |                 |
| address | varchar(100) | YES  |     | NULL    |                 |
+-----+-----+-----+-----+-----+-----+
6 rows in set (0.02 sec)

mysql>
  
```

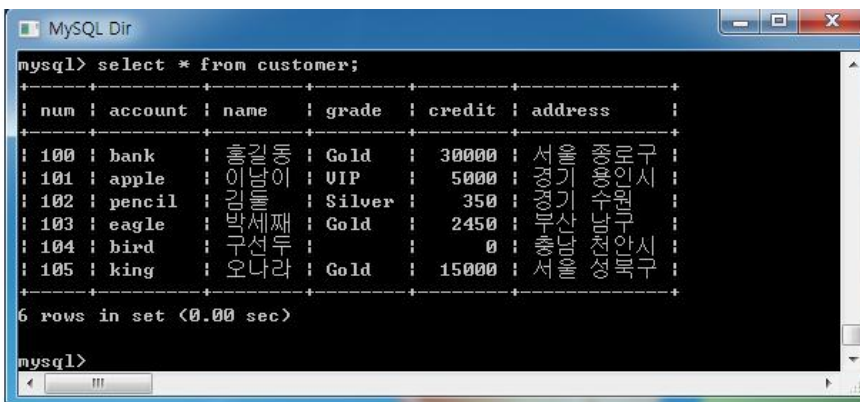
그림. MySQL 적용 후 Customer 릴레이션 구조

- * AUTO_INCREMENT 의 초기값을 100으로 설정하기
 - 테이블 생성 직후에 설정

표. CREATE TABLE : AUTO_INCREMENT의 초기값 설정

```
ALTER TABLE customer AUTO_INCREMENT=100;
```

```
insert into customer (account, name, grade, credit, address )
values ( 'bank', '홍길동', 'Gold', 30000, '서울 종로구');
insert into customer (account, name, grade, credit, address )
values ( 'apple', '이남이', 'VIP', 5000, '경기 용인시');
insert into customer (account, name, grade, credit, address )
values ( 'pencil', '김둘', 'Silver', 350, '경기 수원');
insert into customer (account, name, grade, credit, address )
values ( 'eagle', '박세째', 'Gold', 2450, '부산 남구');
insert into customer (account, name, grade, credit, address )
values ( 'bird', '구선두', '', 0, '충남 천안시');
insert into customer (account, name, grade, credit, address )
values ( 'king', '오나라', 'Gold', 15000, '서울 성북구');
```



```
mysql> select * from customer;
```

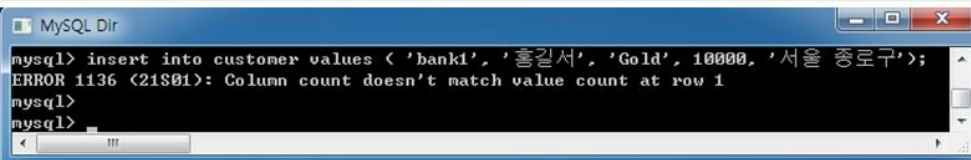
num	account	name	grade	credit	address
100	bank	홍길동	Gold	30000	서울 종로구
101	apple	이남이	VIP	5000	경기 용인시
102	pencil	김둘	Silver	350	경기 수원
103	eagle	박세째	Gold	2450	부산 남구
104	bird	구선두		0	충남 천안시
105	king	오나라	Gold	15000	서울 성북구

6 rows in set (0.00 sec)

그림. MySQL 적용 후 Customer 릴레이션 데이터

- * AUTO_INCREMENT 적용시 INSERT 오류

```
insert into customer values ( 'bank1', '홍길서', 'Gold', 10000, '서울 종로구');
```



```
mysql> insert into customer values ( 'bank1', '홍길서', 'Gold', 10000, '서울 종로구');
ERROR 1136 (21S01): Column count doesn't match value count at row 1
mysql>
```

그림. MySQL 적용 시 튜플 삽입오류 : 속성과 속성값의 개수가 다름

- * AUTO_INCREMENT : 속성이름 리스트를 사용하지 않고 입력하기
 - AUTO_INCREMENT 속성 값 = 숫자 이외의 모든 문자열이나 NULL을 속성 값으로 입력하면 DBMS는 자동으로 다음 숫자를 사용하여 튜플을 삽입함
 - AUTO_INCREMENT 속성 값 = 숫자를 사용하면, 입력한 숫자로 값이 지정됨

```
insert into customer values ( NULL, 'bank1', '홍길서', 'Gold', 10000, '서울 종로구');
```

```
mysql> insert into customer values ( NULL, 'bank1', '홍길서', 'Gold', 10000, '서울 종로구' );
Query OK, 1 row affected (0.05 sec)

mysql> select * from customer;
```

num	account	name	grade	credit	address
2	apple	이남이	VIP	5000	경기 용인시
3	pencil	김들	Silver	350	경기 수원시
4	eagle	박세재	Gold	2450	부산 남구
5	bird	구선두		0	충남 천안시
6	king	오나라	Gold	15000	서울 성동구
103	bank1	홍길서	Gold	10000	서울 종로구

```
6 rows in set (0.00 sec)
```

그림. MySQL 적용 : AUTO_INCREMENT 속성 = NULL

```
insert into customer values ( '', 'bank2', '홍길동', 'Silver', 2000, '서울 종로구');
```

```
mysql> insert into customer values ( '', 'bank2', '홍길동', 'Silver', 2000, '서울 종로구' );
Query OK, 1 row affected, 1 warning (0.08 sec)

mysql> select * from customer;
```

num	account	name	grade	credit	address
2	apple	이남이	VIP	5000	경기 용인시
3	pencil	김들	Silver	350	경기 수원시
4	eagle	박세재	Gold	2450	부산 남구
5	bird	구선두		0	충남 천안시
6	king	오나라	Gold	15000	서울 성동구
103	bank1	홍길서	Gold	10000	서울 종로구
104	bank2	홍길동	Silver	2000	서울 종로구

```
7 rows in set (0.00 sec)
```

그림. MySQL 적용 : AUTO_INCREMENT 속성 = ''

insert into customer values (200, 'oil', '오오구', 'New', 0, '충남 서산');

```
mysql> insert into customer values ( 200, 'oil', '오오구', 'New', 0, '충남 서산' );
Query OK, 1 row affected (0.05 sec)

mysql> select * from customer;
+----+-----+-----+-----+-----+-----+
| num | account | name | grade | credit | address |
+----+-----+-----+-----+-----+-----+
| 2 | apple | 이 | VIP | 5000 | 경기 용인시 |
| 3 | pencil | 이 | Silver | 350 | 경기 수원시 |
| 4 | eagle | 이 | Gold | 2450 | 경기 동탄 |
| 5 | bird | 이 | Gold | 0 | 경기 안성 |
| 6 | king | 이 | Gold | 15000 | 경기 안성 |
| 103 | bank1 | 이 | Gold | 10000 | 경기 안성 |
| 104 | bank2 | 이 | Silver | 2000 | 경기 안성 |
| 200 | oil | 이 | New | 0 | 충남 서산 |
+----+-----+-----+-----+-----+-----+
8 rows in set (0.00 sec)

mysql>
```

그림. MySQL 적용 : AUTO_INCREMENT 속성의 값 = 임의의 숫자

4. 실습용 데이터베이스 데이터 정의 (Items 테이블)

표. CREATE TABLE 예 : Items

```
CREATE TABLE Items (
  code      char(3) NOT NULL,
  name      varchar(20) NOT NULL,
  stocks    int DEFAULT 0,
  price     int DEFAULT 0,
  PRIMARY KEY (code )
);
```

```
mysql> desc items;
+----+-----+-----+-----+-----+-----+
| Field | Type | Null | Key | Default | Extra |
+----+-----+-----+-----+-----+-----+
| code | char(3) | NO | PRI | NULL | |
| name | varchar(20) | NO | | NULL | |
| stocks | int(11) | YES | | 0 | |
| price | int(11) | YES | | 0 | |
+----+-----+-----+-----+-----+-----+
4 rows in set (0.02 sec)

mysql>
```

그림. MySQL 적용 후 Items 릴레이션 구조

표. items 릴레이션 데이터 입력

```
insert into items values ('E01', '김', 30, 1500 );
insert into items values ('E02', '단무지', 45, 3250 );
insert into items values ('E03', '햄', 15, 7000 );
insert into items values ('E04', '우유', 25, 4560 );
insert into items values ('E05', '주스', 20, 2300 );
insert into items values ('E06', '라면', 30, 2560 );
```

```

C:\Windows\system32\cmd.exe - mysql -uroot -p
mysql> select * from items;
+----+-----+-----+-----+
| code | name  | stocks | price |
+----+-----+-----+-----+
| E01  | 김치  | 30     | 1500  |
| E02  | 된장무지 | 45     | 3250  |
| E03  | 행어  | 15     | 7000  |
| E04  | 우유  | 25     | 4560  |
| E05  | 주스  | 20     | 2300  |
| E06  | 라면  | 30     | 2560  |
+----+-----+-----+-----+
6 rows in set (0.01 sec)

mysql>

```

그림. items 릴레이션 데이터 확인

- Orders 테이블

표. CREATE TABLE 예 : Orders

```

CREATE TABLE Orders (
  num      int NOT NULL,
  customer varchar(10) NOT NULL,
  item     char(3),
  qty      int NOT NULL DEFAULT 1,
  date     date,
  saddr    varchar(100),
  PRIMARY KEY ( num )
);

```

```

C:\Windows\system32\cmd.exe - mysql -uroot -p
mysql> desc orders;
+----+-----+-----+-----+-----+-----+
| Field | Type      | Null | Key | Default | Extra |
+----+-----+-----+-----+-----+-----+
| num   | int(11)   | NO   | PRI | NULL    |       |
| customer | varchar(10) | NO   |     | NULL    |       |
| item  | char(3)   | YES  |     | NULL    |       |
| qty   | int(11)   | NO   |     | 1       |       |
| date  | date      | YES  |     | NULL    |       |
| saddr | varchar(100) | YES  |     | NULL    |       |
+----+-----+-----+-----+-----+-----+
6 rows in set (0.05 sec)

mysql>

```

그림. MySQL 적용 후 Orders 릴레이션 구조

【학습정리】

1. SQL을 사용하여 데이터베이스를 조작할 수 있습니다. SQL은 데이터정의어, 데이터조작어, 데이터제어어로 분류할 수 있다.
2. SQL을 사용하여 테이블을 생성할 때, 테이블 이름과 속성이름, 데이터 타입, 제약사항, 키정의 및 데이터 무결성을 위한 제약 조건을 정의하여야 한다.
3. 데이터 타입은 속성의 도메인을 정의하기 위한 방법 중에서 가장 보편적인 것이며, 각각의 속성 데이터에 적합한 타입과 크기를 가지고 정의하여야 한다.