

# 13강

## 기본키와 고유키 그리고 외래키





## 1. 기본키(Primary Key) 더 알기

- 중복되지 않는 고유값만 허용
- **NULL** 값 허용하지 않음
- 테이블당 **하나의 기본키만** 지정 가능

➤ 기본키 설정 방법1

```
CREATE TABLE people (
    first_name CHAR(2) PRIMARY KEY,
    last_name CHAR(3),
    nickname VARCHAR(10)
);
```

➤ 기본키 설정 방법2

```
CREATE TABLE people (
    first_name CHAR(2),
    last_name CHAR(3),
    nickname VARCHAR(10),
    PRIMARY KEY (first_name)
);
```



## ➤ 기본 키 변경하기

## ALTER TABLE [테이블명] DROP PRIMARY KEY

```
ALTER TABLE people DROP PRIMARY KEY;
```

**ALTER TABLE [테이블명] ADD PRIMARY KEY (Key 설정할 컬럼 명1, Key 설정할 컬럼 명 2, ...)**

```
ALTER TABLE people ADD PRIMARY KEY (last_name);
```



## ➤ 다중 기본키

- primary Key가 여러 개가 아니라 지정한 필드명의 조합이 하나이면 된다는 의미임
- first\_name 과 last\_name의 조합을 고유값으로 행들을 구분해준다는 의미

```
CREATE TABLE people (
    first_name CHAR(2),
    last_name CHAR(3),
    nickname VARCHAR(10),
    PRIMARY KEY (first_name, last_name)
);
```

```
INSERT INTO PEOPLE VALUES('홍', '길동', '별명');
INSERT INTO PEOPLE VALUES('전', '우치', '별명');
INSERT INTO PEOPLE VALUES('전', '길동', '별명');
INSERT INTO PEOPLE VALUES('홍', '우치', '별명');
```

	first_name	last_name	nickname
▶	전	길동	별명
	전	우치	별명
	홍	길동	별명
*	홍	우치	별명
	NULL	NULL	NULL

First\_name과 last\_name이 각각 primary key가 아니라 둘의 조합이 하나의 primary key가 된다.



```
INSERT INTO PEOPLE VALUES('홍', '길동', '별명');
```



First\_name과 last\_name 의 조합으로 이미 '홍길동' 이 입력되어 있으므로  
Duplicate entry error가 출력됨을 주의하자

The screenshot shows a MySQL Workbench interface. In the SQL tab, there is a single query:

```
1 INSERT INTO PEOPLE VALUES('홍', '길동', '별명');  
2
```

In the Output tab, the results of the query are shown in a table:

Action Output		
#	Time	Action
1	20:34:37	INSERT INTO PEOPLE VALUES('홍', '길동', '별명')

To the right of the table, the message area displays:

Message  
Error Code: 1062. Duplicate entry '홍-길동' for key 'people.PRIMARY'

## ➤ 테이블 구조 확인 명령어

```
desc menus;
```

	Field	Type	Null	Key	Default	Extra
▶	menu_id	int	NO	PRI	NULL	auto_increment
	fk_business_id	int	NO	MUL	NULL	
	menu_name	varchar(20)	NO		NULL	
	kilocalories	decimal(7,2)	NO		NULL	
	price	int	NO		NULL	
	likes	int	NO		0	



## 2. 고유키(Unique) 더 알기

- 중복 제한, NULL 값 가능

-- 고유키 넣는 방법 1

```
CREATE TABLE people (
    person_id INT AUTO_INCREMENT PRIMARY KEY,
    first_name CHAR(2) UNIQUE,
    last_name CHAR(3)
);
```

-- 고유키 넣는 방법 2

```
CREATE TABLE people (
    person_id INT AUTO_INCREMENT PRIMARY KEY,
    first_name CHAR(2),
    last_name CHAR(3),
    UNIQUE (first_name)
);
```

-- 다중 고유키

```
CREATE TABLE people (
    person_id INT AUTO_INCREMENT PRIMARY KEY,
    first_name CHAR(2),
    last_name CHAR(3),
    UNIQUE (first_name, last_name)
);
```



### 3. 외래키(Foreign Key)

- 외래키(Foreign Key)는 두 테이블을 서로 연결하는 데 사용되는 키이다.
- 외래키(Foreign Key)가 포함된 테이블을 **자식 테이블**이라고 하고 외래키 값을 제공하는 테이블을 **부모 테이블**이라한다.

#### ➤ 외래키 사용시 주의 사항

- 외래키 값은 **NULL**이거나 **부모 테이블의 기본키 값**과 동일해야한다. (참조 무결성 제약조건)
- 외래키로 지정할 두 테이블의 필드는 **같은 데이터 타입**이어야 한다.
- 부모 테이블의 기본키, 고유키를 외래키로 지정할 수 있다.
- 부모 테이블의 기본키, 고유키가 여러개의 컬럼으로 이루어져 있다면 부모가 가진 기본키, 고유키 컬럼을 원하는 개수만큼 묶어서 외래키로 지정할 수 있다.

#### ➤ 외래 키 추가하기

```
ALTER TABLE _자식테이블명  
ADD CONSTRAINT _제약명  
FOREIGN KEY ( _자식테이블외래키 )  
REFERENCES 부모테이블명 ( _부모테이블기본키 )  
-- ON DELETE _삭제시제약  
-- ON UPDATE _수정시제약
```

# 명령어를 이용한 Foreign Key 지정방법





➤ businesses에 sections에 대한 외래키를 설정

```
alter table businesses
add constraint
foreign key (fk_section_id)
references sections (section_id);
```

자식테이블 외래키  
부모 테이블 기본키



```
INSERT INTO businesses
(fk_section_id, business_name, status, can_takeout)
VALUES ('8', '섹션에없는식당', 'OPN', '1');
```



Sections은 부모 테이블이고 businesses는 자식 테이블로 외래키 제약조건이 지정되어 있으므로 sections 테이블의 section\_id에 존재하지 않는 데이터를 businesses 자식테이블에 입력할 수 없다. 고로 외래키 제약조건 실패라는 error가 출력된다.

#	Time	Action	Message
1	21:24:58	INSERT INTO businesses (fk_section_id, business_name, status, can_takeout) VALUES ('8', '섹션에없는식당', 'OPN', '1')	Error Code: 1452. Cannot add or update a child row: a foreign key constraint fails ('mydatabase'.businesses', CONSTRAINT 'businesses_ibfk_1' FOREIGN KEY (fk_section_id) REFERENCES sections (section_id))

# 메뉴를 이용한 Foreign Key 지정방법



## 메뉴로 외래키(Foreign Key) 설정



- businesses 테이블에 fk\_section\_id를 sections테이블에 section\_id 컬럼을 참조하는 외래키를 메뉴를 이용하여 설정한다.

먼저 sections 테이블의 section\_id 컬럼과 businesses 테이블에 fk\_section\_id컬럼이 데이터타입이 동일한지 확인한다.

## 메뉴로 외래키(Foreign Key) 설정



- Foreign Keys 탭 메뉴를 클릭하고 이름을 지어준다.

The screenshot shows the 'businesses' table configuration in MySQL Workbench. The 'Foreign Key Name' field contains 'businesses\_section\_id', which is highlighted with a red box. The 'Referenced Table' dropdown menu is open, showing several options including 'mydb.sections'. The 'Column' dropdown shows 'business\_id' and 'section\_id'. The 'Referenced Column' dropdown shows 'business\_name' and 'takeout'. The 'Foreign Key Options' section includes 'On Update' and 'On Delete' dropdowns, both set to 'NO ACTION'. A checkbox for 'Skip in SQL generation' is also present.

- 참조할 테이블을 고른다.

This screenshot continues the process of setting up the foreign key. The 'Referenced Table' dropdown has been selected, and the 'mydb.sections' option is highlighted with a blue box. The other listed tables are 'businesses', 'menus', 'people', 'ratings', 'register', and 'takeout'. The rest of the interface remains consistent with the previous screenshot, showing the table name 'businesses', schema 'mydb', and various configuration options for the foreign key.

## 메뉴로 외래키(Foreign Key) 설정



- ▶ 참조할 컬럼을 고른다 (businesses 테이블의 fk\_section\_id 컬럼은 sections 테이블의 section\_id 컬럼을 참조한다.)

The screenshot shows the 'Create Table' dialog in MySQL Workbench. The 'Table Name' is set to 'businesses', 'Schema' to 'mydb', 'Charset/Collation' to 'utf8mb4', and 'Engine' to 'InnoDB'. The 'Comments' field is empty. In the 'Foreign Key Name' section, 'businesses\_section\_id' is selected, and the 'Referenced Table' is set to 'mydb`.`sections'.

Foreign Key Name	Referenced Table	Column	Referenced Column	Foreign Key Options
businesses_section_id	'mydb`.`sections'	<input type="checkbox"/> business_id <input checked="" type="checkbox"/> fk_section_id <input type="checkbox"/> business_name <input type="checkbox"/> status <input type="checkbox"/> can_takeout	section_id section_id section_name floor Specify Column...	On Update: NO ACTION On Delete: NO ACTION <input type="checkbox"/> Skip in SQL generation

- ▶ Apply 를 눌러주면 foreign key 설정 완료



- 외래키를 지정하는 목적은 테이블끼리 join을 자유자재로 사용하기 위함이다.

```
select s.section_name, b.business_name, b.status from sections s  
join businesses b on b.fk_section_id = s.section_id where section_name = '한식';
```



	section_name	business_name	status
▶	한식	북극냉면	OPN
	한식	보쌈마니아	OPN
	한식	할매장국	CLS

```
select s.section_name, b.business_name, b.status from sections s  
join businesses b on b.fk_section_id = s.section_id where section_name = '중식';
```



	section_name	business_name	status
▶	중식	화룡각	OPN
	중식	린민짬뽕	CLS

# 메뉴를 이용한 Foreign Key 삭제방법



## 메뉴로 외래키(Foreign Key) 설정



- Foreign Keys 탭 메뉴를 클릭하고 Foreign Key Name 위에서 오른쪽 버튼 -> Delete selected 클릭한다.

Table Name: businesses Schema: mydb  
Charset/Collation: utf8mb4 Engine: InnoDB  
Comments:

Foreign Key Name	Referenced Table
businesses_section_id	'mydb'.`sections`

**Delete selected**

Column	Referenced Column
<input type="checkbox"/> business_id	
<input checked="" type="checkbox"/> fk_section_id	section_id
<input type="checkbox"/> business_name	
<input type="checkbox"/> status	
<input type="checkbox"/> can_takeout	

Foreign Key Options  
On Update: RESTRICT  
On Delete: RESTRICT  
 Skip in SQL generation

- Apply 를 눌러주면 foreign key 설정이 삭제된다.

Table Name: businesses Schema: mydb  
Charset/Collation: utf8mb4 Engine: InnoDB  
Comments:

Foreign Key Name	Referenced Table

Column	Referenced Column

Foreign Key Options  
On Update:   
On Delete:

# 명령어 이용한 Foreign Key 삭제방법





## ➤ 외래 키 삭제하기

```
ALTER TABLE _자식테이블명 DROP FOREIGN KEY 제약조건명
```

## ➤ 외래 키 삭제하기 전 제약조건 확인하는 방법

```
select * from information_schema.table_constraints where table_name = 'businesses';
```

	CONSTRAINT_CATALOG	CONSTRAINT_SCHEMA	CONSTRAINT_NAME	TABLE_SCHEMA	TABLE_NAME	CONSTRAINT_TYPE	ENFORCED
▶ def		mydatabase	PRIMARY	mydatabase	businesses	PRIMARY KEY	YES
def		mydatabase	businesses_ibfk_1	mydatabase	businesses	FOREIGN KEY	YES

```
alter table businesses drop foreign key businesses_ibfk_1;
```

## ➤ 외래 키 제약

제약	설명	비고
<b>RESTRICT</b>	자식 테이블이 참조하고 있을 경우, 데이터 삭제 불가	
<b>NO ACTION</b>	Restrict와 동일, 옵션을 지정하지 않았을 경우 자동으로 선택된다.	
<b>CASCADE</b>	부모 데이터 삭제/수정 시 자식 데이터도 삭제/수정	
<b>SET NULL</b>	부모 데이터 삭제/수정 시 자식 테이블의 참조 컬럼을 Null로 업데이트	자식 외래키가 NOT NULL일 시 설정 불가
<b>SET DEFAULT</b>	부모 데이터 삭제/수정 시 자식 테이블의 참조 컬럼을 Default 값으로 업데이트	InnoDB 엔진에서 사용 불가



## InnoDB는 MySQL을 위한 데이터베이스 엔진이다.

### ➤ 데이터베이스 엔진이란?

- 데이터베이스 엔진 (또는 스토리지 엔진)은 DBMS가 데이터베이스에 대해 데이터를 삽입, 추출, 업데이트 및 삭제 (CRUD 참조)하는 데 사용하는 기본 소프트웨어 **컴포넌트(Component : 재사용이 가능한 각각의 독립된 모듈)**
- 데이터베이스 엔진을 조작할 때 DBMS 고유의 사용자 인터페이스를 이용하는 방법과 포트 번호를 이용하는 방법이 있음
- 대부분 DBMS는 고유의 사용자 인터페이스를 통하지 않고, 사용자가 내장된 엔진과 상호작용 할 수 있는 자신만의 API를 포함

### ➤ InnoDB

- MySQL 5.5 버전 이후 기본적으로 MyISAM 대신 InnoDB 사용
- MySQL 바이너리에 내장되어 있음
- 트랜잭션(Transaction) 지원
- 동시성 제어 가능
- 데이터 무결성 보장
- 제약조건, 외래 키 지원
- MVCC(Multi Versioning Concurrency Control) 구조로 동작
- Row-level Lock (행 단위 Lock) 사용으로 변경 작업(INSERT, UPDATE, DELETE)의 속도가 빠름

# 명령어 이용한 Foreign Key 제약하기





- 아래처럼 외래키 및 제약조건을 지정한다.

```
alter table menus
add constraint
foreign key (fk_business_id)
references businesses (business_id)
on delete restrict -- 자식테이블 참조시 삭제 불가
on update cascade; -- 부모테이블의 데이터 수정시 자식테이블의 데이터도 수정
```



```
delete from businesses where business_name='화룡각';
```



Businesses는 부모 테이블이고, menus는 자식테이블로 외래키를 지정하였으며  
삭제시 제약조건으로 **restrict(얽매이다)** 지정하였기 때문에 데이터를 삭제 할 수  
없으므로 error 출력

Message	Duration / Fetch
Error Code: 1451. Cannot delete or update a parent row: a foreign key constraint fails ('mydatabase'.'menus', CONSTRAINT 'menus_ibfk_1' FOREIGN KE... 0.016 sec	

# 메뉴를 이용한 Foreign Key 제약하기



## 메뉴로 외래키(Foreign Key) 설정



- Foreign Keys 탭 메뉴를 클릭하고 On Update: CASCADE, On Delete: RESTRICT 제약조건 지정한다.

The screenshot shows the MySQL Workbench interface for managing a foreign key named 'businesses\_section\_id' in the 'businesses' table. The 'Referenced Table' is 'mydb`.`sections' and the 'Referenced Column' is 'section\_id'. In the 'Foreign Key Options' section, the 'On Update' dropdown is set to 'CASCADE' and the 'On Delete' dropdown is set to 'RESTRICT'. A red box highlights the 'On Delete' dropdown menu, which includes options: RESTRICT, CASCADE, SET NULL, and NO ACTION. The 'CASCADE' option is selected.

Foreign Key Name	Referenced Table
businesses_section_id	'mydb`.`sections'

Column	Referenced Column
<input type="checkbox"/> business_id	
<input checked="" type="checkbox"/> fk_section_id	section_id
<input type="checkbox"/> business_name	
<input type="checkbox"/> status	
<input type="checkbox"/> can_takeout	

Foreign Key Options	
On Update:	CASCADE
On Delete:	RESTRICT
	RESTRICT
	CASCADE
<input type="checkbox"/> Skip in	SET NULL
	NO ACTION

- Apply 를 눌러주면 foreign key 제약조건이 설정이 된다.



```
update businesses set business_id = 80 where business_name='화룡각';
```



business_id	fk_section_id	business_name	status	can_takeout
5	1	북극냉면	OPN	0
6	1	보쌈마니아	OPN	1
7	5	에그사라다	VCT	1
8	6	달다방	OPN	1
9	7	마카오마카롱	OPN	1
10	2	김밥마라	OPN	1
11	7	소소스윗	OPN	1
12	4	사사서셔소소스시	VCT	1
13	3	린민짬뽕	CLS	1
14	7	파시조아	OPN	1
15	1	할매장국	CLS	0
16	5	노선이탈리아	OPN	1
17	6	커피앤햄드	OPN	1
18	2	신림동백순대	VCT	1
80	3	화룡각	OPN	1

Businesses는 부모 테이블이고, menus는 자식테이블로 외래키를 지정하였으며 수정 시 제약조건으로 cascade(종속)로 지정하였기 때문에 부모 테이블인 businesses의 데이터를 수정하면 자식 테이블인 menus의 데이터도 같이 수정된다.

menu_id	fk_business_id	menu_name	kilocalories	price	likes
12	80	간짜장	682.48	7000	3
13	9	뚱카롱	247.62	2000	8
14	5	전통 비빔냉면	563.45	8000	4
15	10	참치김밥	532.39	3000	0
16	2	치즈떡볶이	638.42	5000	15
17	11	플레인와플	299.31	6500	2
18	2	찹쌀순대	312.76	3000	-4
19	15	전통 육개장	423.18	8500	2
20	4	국물떡볶이	483.29	4500	1
21	10	돈가스김밥	562.72	4000	0
22	80	삼선짬뽕	787.58	8000	32
23	11	수풀레팬케익	452.37	9500	5
24	4	라볶이	423.16	5500	0
25	8	모카프라푸치노	216.39	6000	8
26	14	옛날팥빙수	382.35	8000	2

