

3day-MYSQL

# ALTER TABLE

테이블명 변경 :

alter table 이전테이블명 rename  
새테이블명;

```
mysql> show tables; 테이블명 변경
+-----+
| Tables_in_com |
+-----+
| car            |
| salesman      |
+-----+
2 rows in set (0.00 sec)

mysql> alter table salesman rename employee;
Query OK, 0 rows affected (0.00 sec)

mysql> show tables;
+-----+
| Tables_in_com |
+-----+
| car            |
| employee      |
+-----+
2 rows in set (0.00 sec)
```

필드 삭제

alter table 테이블명 drop column  
필드명 ;

```
mysql> desc car; alter 필드 삭제
+-----+-----+-----+-----+-----+-----+
| Field | Type          | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+-----+
| cno   | char(2)       |      | PRI |          |       |
| cname | varchar(10)   |      |     |          |       |
| cc    | smallint(6)   | YES  |     | NULL     |       |
| weight | smallint(6)   | YES  |     | NULL     |       |
| outlet | varchar(10)   | YES  |     | NULL     |       |
| vendor | varchar(20)   | YES  |     | NULL     |       |
+-----+-----+-----+-----+-----+-----+
6 rows in set (0.00 sec)

mysql> alter table car drop column vendor;
Query OK, 0 rows affected (0.02 sec)
Records: 0 Duplicates: 0 Warnings: 0

mysql> desc car;
+-----+-----+-----+-----+-----+-----+
| Field | Type          | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+-----+
| cno   | char(2)       |      | PRI |          |       |
| cname | varchar(10)   |      |     |          |       |
| cc    | smallint(6)   | YES  |     | NULL     |       |
| weight | smallint(6)   | YES  |     | NULL     |       |
| outlet | varchar(10)   | YES  |     | NULL     |       |
+-----+-----+-----+-----+-----+-----+
5 rows in set (0.00 sec)
```

# ALTER TABLE

## 필드명 변경

`alter table 테이블명 change 이전필드명 새필드명 데이터형 ;`

```
mysql> alter table test change phone phones char(12) not null;  
Query OK, 0 rows affected (0.02 sec)  
Records: 0 Duplicates: 0 Warnings: 0
```

```
mysql> desc test;
```

Field	Type	Null	Key	Default	Extra
id	int	NO	PRI	NULL	
name	char(10)	NO		NULL	
phones	char(12)	NO		NULL	

## 필드 타입 변경

`alter table 테이블명 change 필드명 필드명 데이터형 ;`

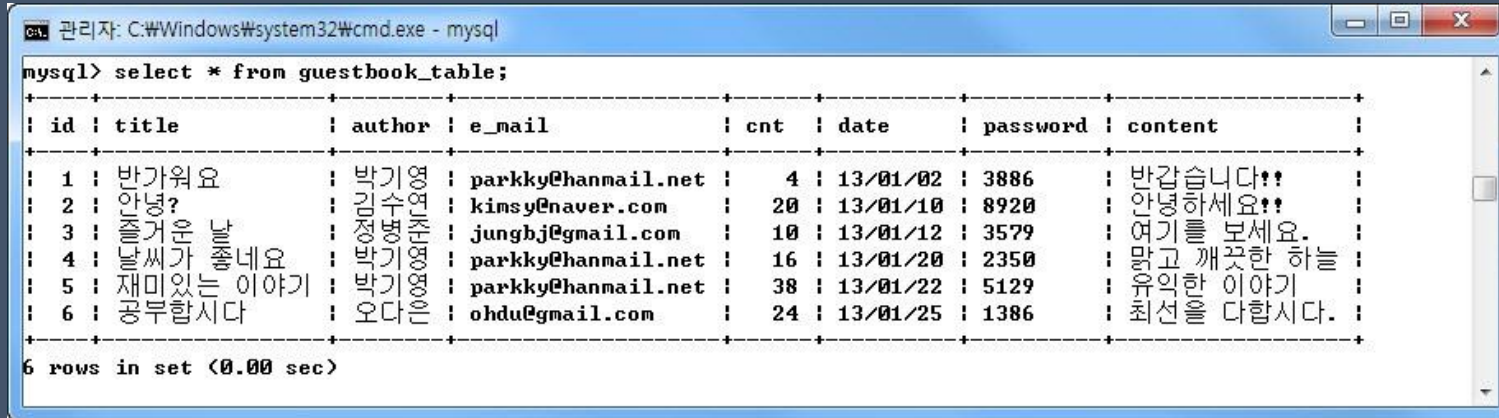
```
mysql> alter table test change phones phones char(15);  
Query OK, 0 rows affected (0.05 sec)  
Records: 0 Duplicates: 0 Warnings: 0
```

```
mysql> desc test;
```

Field	Type	Null	Key	Default	Extra
id	int	NO	PRI	NULL	
name	char(10)	NO		NULL	
phones	char(15)	YES		NULL	

# 데이터 조작

- select 문
  - 모든 데이터 검색



```
관리자: C:\Windows\system32\cmd.exe - mysql
mysql> select * from guestbook_table;
```

id	title	author	e_mail	cnt	date	password	content
1	반가워요	박기영	parkky@hanmail.net	4	13/01/02	3886	반갑습니다!!
2	안녕?	김수민	kimsy@naver.com	20	13/01/10	8920	안녕하세요!!
3	즐거운 날	정영준	jungbj@gmail.com	10	13/01/12	3579	여기를 보세요.
4	날씨가 좋네요	박기영	parkky@hanmail.net	16	13/01/20	2350	맑고 깨끗한 하늘
5	재미있는 이야기	박기영	parkky@hanmail.net	38	13/01/22	5129	유익한 이야기
6	공부합시다	오다은	ohdu@gmail.com	24	13/01/25	1386	최선을 다합니다.

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

## 형식

```
select [ all(기본) | distinct ] 열_이름_리스트
from 테이블_이름
[where 조건식]
[group by 열_이름_리스트 ]
[order by 열_이름_리스트 [ asc(기본) | desc ]];
```

# 데이터 조작

- select 문
  - 열들 중 일부 검색

```
관리자: C:\Windows\system32\cmd.exe - mysql
mysql> select id, title, author as 글쓴이, date, content
-> from guestbook_table;
```

id	title	글쓴이	date	content
1	반가워요	박기영	13/01/02	반갑습니다!!
2	안녕?	김수연	13/01/10	안녕하세요!!
3	즐거운 날	정병준	13/01/12	여기를 보세요.
4	날씨가 좋네요	박기영	13/01/20	맑고 깨끗한 하늘
5	재미있는 이야기	박기영	13/01/22	유익한 이야기
6	공부합시다	오다은	13/01/25	최선을 다합니다.

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

- distinct를 이용한 검색

```
관리자: C:\Windows\system32\cmd.exe - mysql
mysql> select author
-> from guestbook_table;
```

author
박기영
김수연
정병준
박기영
박기영
오다은

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

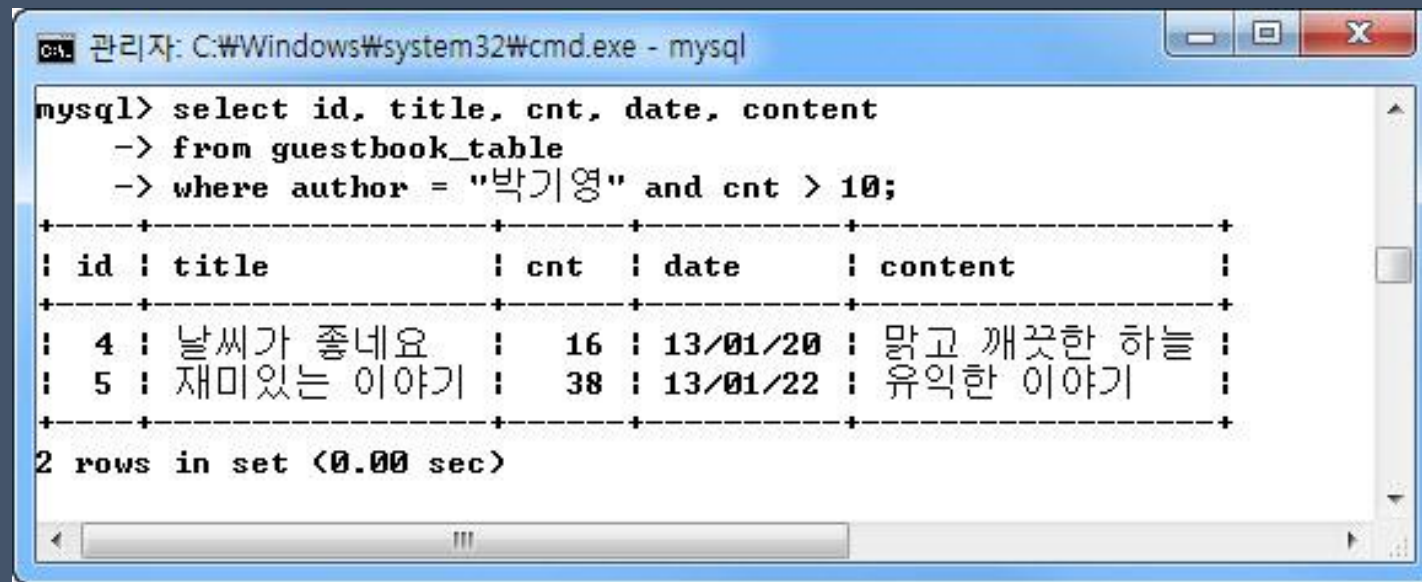
```
관리자: C:\Windows\system32\cmd.exe - mysql
mysql> select distinct author
-> from guestbook_table;
```

author
박기영
김수연
정병준
오다은

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

# 데이터 조작

- select 문(계속)
  - where 절을 이용한
  - 조건 검색



The screenshot shows a Windows command prompt window titled "관리자: C:\Windows\system32\cmd.exe - mysql". Inside the window, a MySQL prompt "mysql>" is followed by the SQL query: "select id, title, cnt, date, content -> from guestbook\_table -> where author = '박기영' and cnt > 10;". The results are displayed in a table format with columns: id, title, cnt, date, and content. There are two rows of data. Below the table, it says "2 rows in set (0.00 sec)".

```
mysql> select id, title, cnt, date, content
-> from guestbook_table
-> where author = "박기영" and cnt > 10;
```

id	title	cnt	date	content
4	날씨가 좋네요	16	13/01/20	맑고 깨끗한 하늘
5	재미있는 이야기	38	13/01/22	유익한 이야기

2 rows in set (0.00 sec)

# MySQL – 2개의 이상의 테이블 사용하기(JOIN)

- Join 개요

- ✓ 하나의 웹 서비스에는 여러 테이블 사용
- ✓ 회원정보, 게시물, 댓글, 구매상품 정보, 로그인 기록, 공지사항 등등의 테이블이 존재
- ✓ 2개 이상의 테이블을 서로 결합시켜 필요한 정보 얻는 경우가 많음

리뷰에 관한 테이블 생성 - prodReview

```
create table prodReview (  
  prodReviewID int unsigned auto_increment comment '리뷰의 고유 번호',  
  myMemberID int unsigned comment '리뷰를 작성한 회원번호',  
  content tinytext comment '리뷰 내용',  
  regTime datetime not null comment '리뷰 작성 날짜',  
  primary key(prodReviewID))  
charset=utf8 comment='상품 리뷰';
```



# MySQL – 2개의 이상의 테이블 사용하기(JOIN)

- 리뷰에 관한 테이블 생성 - prodReview

```
INSERT INTO prodReview(myMemberID, content, regTime)
VALUES(2, '이 신발 너무 편하고 좋습니다.', now( ));
```

```
INSERT INTO prodReview(myMemberID, content, regTime)
VALUES(3, '실제로 보니 더 예쁩니다 ', now( ));
```

```
INSERT INTO prodReview(myMemberID, content, regTime)
VALUES(4, '신발이 너무 예쁘긴 하나 너무 비쌉니다.', now( ));
```

- 리뷰내용

- 6개 리뷰
- 작성자 이름이 없음
- 고객의 이름을 join을 통해 얻을 수 있음

```
INSERT INTO prodReview(myMemberID, content, regTime)
VALUES(5, '너무 좋아서 친구에게도 추천했습니다.', now( ));
```

```
INSERT INTO prodReview(myMemberID, content, regTime)
VALUES(6, '가격이 너무 비싸지만 예쁘긴 합니다 ', now( ));
```

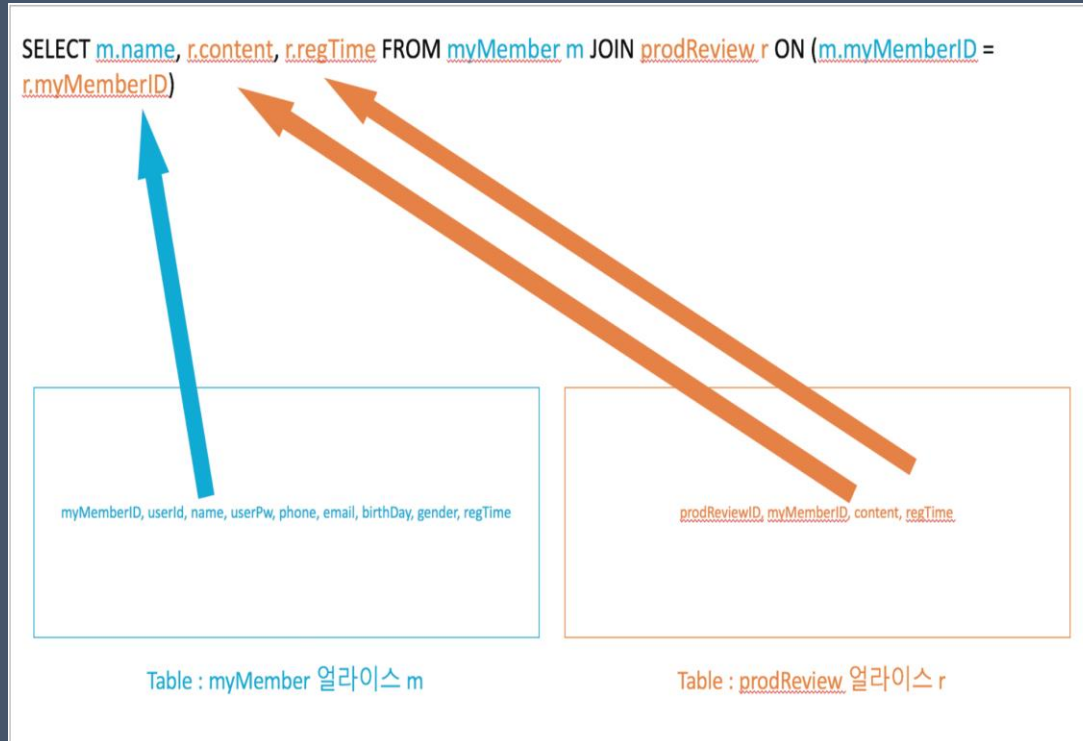
```
INSERT INTO prodReview(myMemberID, content, regTime)
VALUES(7, '너무 좋아서 한개 더 주문했습니다', now( ));
```

```
SELECT 필드명 FROM 테이블명 앨리어스 JOIN 연결할 테이블명 앨리어스 ON(두 테이블의 연결고리 역할을 할 필드 조건문)
```



# MySQL – 2개의 이상의 테이블 사용하기(JOIN)

SELECT 필드명 FROM 테이블명 앨리어스 JOIN 연결할 테이블명 앨리어스 ON(두 테이블의 연결고리 역할을 할 필드 조건문)



JOIN문의 예

- 1 SELECT m.name, r.content FROM myMember m JOIN prodReview r ON(m.myMemberID = r.myMemberID);

# MySQL – 2개의 이상의 테이블 사용하기(JOIN)

- join 결과

```
mysql> SELECT m.name, r.content, r.regTime FROM myMember m JOIN prodReview r ON (m.myMemberID = r.myMemberID);
```

name	content	regTime
김태현	이 신발 너무 편하고 좋습니다.	2016-11-15 12:01:11
이지형	실제로 보니 더 예쁩니다	2016-11-15 12:01:11
전윤동	신발이 너무 예쁘긴 하나 너무 비쌌습니다.	2016-11-15 12:01:11
노진우	너무 좋아서 친구에게도 추천했습니다.	2016-11-15 12:01:11
한정은	가격이 너무 비싸지만 예쁘긴 합니다	2016-11-15 12:01:11
이정미	너무 좋아서 한개 더 주문했습니다	2016-11-15 12:01:12

6 rows in set (0.00 sec)

- 필드명을 변경해 보고 싶을 때

SELECT 필드명 **AS** 기존 필드명 대신 출력할 필드명 FROM 테이블명

```
SELECT m.name, r.content, r.regTime AS reviewRegTime FROM myMember m JOIN  
prodReview r ON (m.myMemberID = r.myMemberID);
```

# MySQL – 집계함수

집계함수를 사용하면 레코드의 수, 값들의 합계, 평균, 최댓값, 최솟값등을 구할 수 있다.

## 집계함수의 종류

종류	뜻
count(필드명)	레코드의 개수를 표시(값이 null인 경우 포함하지 않음)
count(*)	레코드의 개수를 표시(null을 포함)
sum(필드명)	필드의 값의 합계를 표시
avg(필드명)	필드의 값의 평균을 표시
max(필드명)	필드의 값의 최댓값을 표시
min(필드명)	필드의 값의 최솟값을 표시

# MySQL – 집계 함수

## 사용될 테이블 만들기

▶ 생성할 테이블 정보	
테이블 이름	schoolRecord
필요한 필드	학생 번호 myMemberID와 값이 일치하게 만들
	클래스(소속 반)
	영어 점수
	수학 점수
	과학 점수
	일본어 점수
	코딩 점수

```
CREATE TABLE schoolRecord (  
  studentID int(10) unsigned NOT NULL AUTO_INCREMENT COMMENT '학생 번호',  
  class tinyint unsigned comment '소속 클래스(반)',  
  english tinyint unsigned NOT NULL comment '영어 점수',  
  math tinyint unsigned NOT NULL comment '수학 점수',  
  science tinyint unsigned NOT NULL comment '과학 점수',  
  japanese tinyint unsigned NOT NULL comment '일본어 점수',  
  coding tinyint unsigned NOT NULL comment '코딩 점수',  
  PRIMARY KEY (studentID)) CHARSET=utf8 comment='성적 정보';
```

# MySQL – 집계함수

---

## 테이블에 내용 입력하기

```
INSERT INTO schoolRecord(english, math, science, japanese, coding) VALUES(67, 4, 20, 100, 100);
INSERT INTO schoolRecord(class, english, math, science, japanese, coding) VALUES(1, 55, 60, 80, 50, 100);
INSERT INTO schoolRecord(class, english, math, science, japanese, coding) VALUES(2, 100, 90, 100, 50, 100);
INSERT INTO schoolRecord(class, english, math, science, japanese, coding) VALUES(2, 90, 86, 100, 70, 100);
INSERT INTO schoolRecord(class, english, math, science, japanese, coding) VALUES(3, 80, 50, 60, 20, 100);
INSERT INTO schoolRecord(class, english, math, science, japanese, coding) VALUES(3, 100, 80, 70, 20, 80);
INSERT INTO schoolRecord(class, english, math, science, japanese, coding) VALUES(4, 100, 100, 100, 30, 40);
```

## MySQL – 개수, 합계, 평균, 최대, 최소값

---

### 레코드 개수 구하기

```
SELECT count(*) FROM schoolRecord;
```

### 모든 학생의 영어점수 합계 구하기

```
SELECT sum(english) FROM schoolRecord;
```

### 1~5번 학생의 수학점수 구하기

```
SELECT avg(math) FROM schoolRecord WHERE studentID >= 1 AND studentID <= 5;
```

### 가장 높은 일본어 점수 구하기

```
SELECT max(japanese) FROM schoolRecord;
```

### 가장 낮은 수학 점수 구하기

```
SELECT min(math) FROM schoolRecord;
```

### 반별 영어점수 평균 구하기

```
SELECT class, avg(english) FROM schoolRecord WHERE class IN(1,2,3,4) GROUP BY class;
```

### 반별 수학점수 합계 구하기

```
SELECT class, sum(math) FROM schoolRecord WHERE class IN(1,2,3,4) GROUP BY class;
```

반별로 수학점수 합계가 150 이상인 반 구하기  
group by를 사용할 때 조건은 having 절 다음에 쓴다~~~

```
SELECT class, sum(math) FROM schoolRecord WHERE class IN(1,2,3,4) GROUP BY class HAVING  
sum(math) >= 150;
```

### 가장 높은 일본어 점수 구하기

```
SELECT max(japanese) FROM schoolRecord;
```

### 가장 낮은 수학 점수 구하기

```
SELECT min(math) FROM schoolRecord;
```



## MySQL – 출력 결과의 정렬

특정 데이터를 오름차순 또는 내림차순으로 정렬하는 명령어는 ORDER BY

### ORDER BY 명령문 사용방법

```
1  SELECT 필드명 FROM 테이블명 ORDER BY 정렬 기준이 될 필드명 DESC 또는  
   ASC
```

ORDER BY의 옵션에 쓰이는  
DESC는 큰값에서 작은값으로 정렬  
ASC는 작은값에서 큰값으로 정렬

영어점수를 높은순에서 낮은순으로 정렬하는 예제

```
1  SELECT * FROM schoolRecord ORDER BY english DESC;
```

```
SELECT studentID, english FROM schoolRecord ORDER BY english DESC;
```

집계함수도 정렬 가능

```
SELECT class, sum(math) FROM schoolRecord WHERE class IN(1,2,3,4) GROUP BY class ORDER  
BY sum(math) DESC;
```

## MySQL – 불러올 레코드 수 지정하기

- 게시판 같은 곳에서 한 페이지에 불러올 수가 일정하게 정해져 있음
- 불러올 레코드의 수를 지정하는 명령어는 LIMIT

### LIMIT 명령문 사용방법

1 **SELECT** 필드명 **FROM** 테이블명 **LIMIT** 불러올 수

LIMIT 다음에는 불러올 수를 정수로 작성.

다음은 3개의 레코드를 불러오는 예제

1 **SELECT \* FROM member LIMIT 3;**

LIMIT의 값으로 1개를 사용하면 불러올 개수

LIMIT의 값으로 2개를 사용하면 첫번째 값은 불러올 순번, 두번째 값은 불러올 개수

다음은 3번째 레코드 부터 3개를 불러오는 예제

1 **SELECT \* FROM member LIMIT 3, 3;**

### LIMIT의 값이 1개일 때와 2개 일때의 차이 그림설명

#### LIMIT의 값이 1개일 때는

불러오는 레코드의 수를 의미

```
SELECT fieldName FROM tableName LIMIT 3
```

위의 쿼리문은 3개의 레코드를 불러온다.

myMemberID	name
1	김 태 영
2	김 태 현
3	이 지 형
4	전 윤 동
5	노 진 우
6	한 정 은
7	이 정 미

3개를 출력

#### LIMIT의 값이 1개일 때는

첫 번째 값은 불러오는 레코드의 순서를 의미

순서는 0번부터 시작

두 번째 값은 불러오는 레코드의 수를 의미

```
SELECT fieldName FROM tableName LIMIT 3, 3
```

위의 쿼리문은 3번째 쿼리부터 3개의 레코드를 불러온다.

myMemberID	name
1	김 태 영
2	김 태 현
3	이 지 형
4	전 윤 동
5	노 진 우
6	한 정 은
7	이 정 미

3번째 레코드부터 3개를 출력

## MySQL – 2개 이상의 테이블을 묶어 사용하기

- join : 일치하는 특정 필드를 기준으로 테이블을 연결함
- UNION : 2개 이상의 테이블을 묶어서 사용 할 때 사용하는 명령어는 UNION
- UNION은 2개 이상의 테이블에 있는 레코드를 한번에 보는 기능.
- 사용조건으로 불러오는 필드의 수가 같아야 함.

### UNION 실습 위한 테이블 생성

```
create table dropOutOld(  
dropOutOldID int unsigned not null auto_increment comment '예전 탈퇴회원 번호',  
name varchar(10) not null comment '탈퇴 회원 이름',  
email varchar(30) not null comment '탈퇴 회원 이메일',  
primary key(dropOutOldID))charset=utf8 comment '2016년 탈퇴 회원';
```

```
create table dropOutNew(  
dropOutNewID int unsigned not null auto_increment comment '예전 탈퇴회원 번호',  
name varchar(10) not null comment '탈퇴 회원 이름',  
email varchar(30) not null comment '탈퇴 회원 이메일',  
primary key(dropOutNewID))charset=utf8 comment '2017년 탈퇴 회원';
```

## MySQL – 2개 이상의 테이블을 묶어 사용하기

- join : 일치하는 특정 필드를 기준으로 테이블을 연결함
- UNION : 2개 이상의 테이블을 묶어서 사용 할 때 사용하는 명령어는 UNION
- UNION은 2개 이상의 테이블에 있는 레코드를 한번에 보는 기능.
- 사용조건으로 불러오는 필드의 수가 같아야 함.

### UNION 실습 위한 테이블 생성

```
INSERT INTO dropOutOld(name,email) VALUES('김태영', 'ever@everdevel.com');
INSERT INTO dropOutOld(name,email) VALUES('김태현', 'jake@everdevel.com');
INSERT INTO dropOutOld(name,email) VALUES('이지형', 'tigger@everdevel.com');
INSERT INTO dropOutOld(name,email) VALUES('전윤동', 'nelly@everdevel.com');
INSERT INTO dropOutOld(name,email) VALUES('노진우', 'cooper@everdevel.com');
INSERT INTO dropOutOld(name,email) VALUES('최기욱', 'btm@everdevel.com');
INSERT INTO dropOutNew(name,email) VALUES('최기욱', 'btm@everdevel.com');
INSERT INTO dropOutNew(name,email) VALUES('한정은', 'hanje@everdevel.com');
INSERT INTO dropOutNew(name,email) VALUES('이정미', 'stella@everdevel.com');
```

## MySQL – 2개 이상의 테이블을 묶어 사용하기

- join : 일치하는 특정 필드를 기준으로 테이블을 연결함
- UNION : 2개 이상의 테이블을 묶어서 사용 할 때 사용하는 명령어는 UNION
- UNION은 2개 이상의 테이블에 있는 레코드를 한번에 보는 기능.
- 사용조건으로 불러오는 필드의 수가 같아야 함.

### UNION 명령문 사용방법

```
1 (첫 번째 테이블의 SELECT문) UNION (두 번째 테이블의 SELECT문)
```

dropOutOld 테이블과 dropOutNew 테이블이 있다고 가정한 UNION 명령어 사용 방법

```
1 (SELECT name FROM dropOutOld) UNION (SELECT name FROM dropOutNew);
```

다음과 같이 2개의 SELECT문의 필드수가 같지 않으면 오류 발생

```
1 (SELECT name, email FROM dropOutOld) UNION (SELECT name FROM dropOutNew);
```

```
1 (SELECT name, email FROM dropOutOld) UNION (SELECT name, email FROM dropOutNew);
```

쿼리문안에 또 하나의 쿼리문 작성이 가능.

점수 데이터가 있는 테이블에서 가장 높은 점수를 출력시에 서브쿼리문을 사용하여 가장 높은 점수를 산출 한 후  
조건문에 대입

1. 가장 높은 영어 점수를 가진 레코드를 구하기 위해 먼저 가장 높은 영어 점수를 구한다.

SELECT max(english) FROM schoolRecord;  
(가장 높은 점수인 100을 표시)

2. 영어 점수를 조건으로 하는 레코드를 구하는 쿼리문을 작성한다.

SELECT \* FROM schoolRecord WHERE english = 영어 점수

3. 2번 쿼리문의 조건에 대입할 값으로 1번 쿼리문을 대입한다.

SELECT \* FROM schoolRecord WHERE english = (SELECT max(english) FROM schoolRecord);



## MySQL – 쿼리문 안의 쿼리문 서브쿼리

---

서브쿼리를 이용해 가장 높은 영어점수를 가진 레코드 구하기

```
SELECT * FROM schoolRecord WHERE english = (SELECT max(english)
FROM schoolRecord);
```

서브쿼리를 이용해 영어평균을 구하고 전체 필드와, 평균필드를 포함해 출력하시오.

```
SELECT *, (SELECT avg(English) FROM schoolRecord) as englishAVG
FROM schoolRecord;
```

## MySQL – 특정 필드에 같은 값을 넣지 않는 방법

회원 정보를 테이블에 입력시에 중복이 되면 안되는 값인 아이디, 이메일 주소, 전화번호와 같은 정보는 테이블 생성시에 중복값을 받지 않도록 생성하여 원천적으로 차단

이를 구현하기 위해 사용하는 옵션은 UNIQUE

다음은 email 필드에 UNIQUE를 적용하는 예제

```
1 ALTER TABLE myMember MODIFY email varchar(30) NOT NULL UNIQUE;
```

email필드에는 원천적으로 중복값이 입력되지 않게 처리.

```
bin — mysql • sudo — 148x21
mysql> ALTER TABLE myMember modify email varchar(30) NOT NULL UNIQUE COMMENT '고객의 이메일 주소';
Query OK, 0 rows affected (0.02 sec)
Records: 0 Duplicates: 0 Warnings: 0

mysql> DESC myMember;
```

Field	Type	Null	Key	Default	Extra
myMemberID	int(10) unsigned	NO	PRI	NULL	auto_increment
userId	varchar(15)	NO		NULL	
name	varchar(10)	NO		NULL	
password	varchar(30)	NO		NULL	
phone	varchar(13)	NO		NULL	
email	varchar(30)	NO	UNI	NULL	
birthDay	char(10)	NO		NULL	
gender	enum('m','w','x')	YES		x	
regTime	datetime	NO		NULL	

```
9 rows in set (0.01 sec)

mysql>
```

## MySQL – 서로 다른 필드의 값을 합쳐서 출력하기

테이블의 서로 다른 필드에 있는 값을 합쳐서 출력하는 명령문은 concat

concat명령문 사용 방법

1 **SELECT concat(합칠 문자열 또는 필드, 합칠 문자열 또는 필드) FROM 테이블명**

다음은 '누구의 이메일 주소는 무엇입니다.'라는 문구를 출력하는 예제

**SELECT concat(name, '의 이메일 주소는', email, '입니다.') FROM myMember WHERE email != '';**

```
bin — mysql — sudo — 148x15
mysql> SELECT concat(name, '의 이메일 주소는 ', email, ' 입니다.') FROM myMember;
+-----+
| concat(name, '의 이메일 주소는 ', email, ' 입니다.') |
+-----+
| 김태영의 이메일 주소는 everdevel@everdevel.com 입니다. |
| 김태현의 이메일 주소는 입니다. |
| 이지형의 이메일 주소는 tigger@everdevel.com 입니다. |
| 전윤동의 이메일 주소는 nelly@everdevel.com 입니다. |
| 노진우의 이메일 주소는 cooper@everdevel.com 입니다. |
| 한정은의 이메일 주소는 hanje@everdevel.com 입니다. |
| 이정미의 이메일 주소는 stella@everdevel.com 입니다. |
+-----+
7 rows in set (0.00 sec)
```

기존의 이메일 주소를 넣으면 에러가 발생~

```
mysql> INSERT INTO myMember(userId, name, password, phone, email, birthDay, gender, regTime)
-> VALUES('pepper', '페퍼', 'vpvjeptm', '010-1234-5678', 'everdevel@everdevel.com', '2015-11-18', 'm', now());
ERROR 1062 (23000): Duplicate entry 'everdevel@everdevel.com' for key 'email'
```

## MySQL – 검색을 더욱 빠르게 하는 인덱스

---

데이터베이스의 레코드를 더욱 빠른 속도로 불러오게 하려면 인덱스를 사용

인덱스 적용 방법

```
1 INDEX(필드명)
```

다음은 name 필드에 인덱스를 추가하는 예제

```
1 ALTER TABLE myMember ADD INDEX(name);
```

레코드의 수가 적으면 인덱스를 사용한것과 사용하지 않는것의 차이는 별로 없으나 레코드의 수가 많은 경우 유용.