ALTER
&
고급 SELECT 문

Head First SQL 캐툴즈 6번째 시간

김지현

Index

ALTER

Head First SQL

ALTER TABLE 변경할 테이블

테이블 이름 변경	RENAME TO 바뀐 테이블 이름	
열 추가	ADD COLUMN 추가 열 이름	자료형 추가하고 싶은 위치
열 변경	CHANGE COLUMN 기존 열 이름 바뀐 열 이름	바뀐 자료형 NOT NULL AUTO-INCREMENT
열 수정	MODIFY COLUMN 열 이름	바뀐 자료형 추가하고 싶은 위치
열 제거	DROP COLUMN 열 이름	

ALTER TABLE 변경할 테이블

열 추가 → ADD COLUMN 추가 열 이름 | 자료형 추가하고 싶은 위치

- 비워두기, LAST ⇨ 맨 마지막 열로 추가
- FIRST ⇨ 맨 처음 열로 추가(PRIMARY KEY 설정)
- AFTER 특정 열 ⇨ 특정 열 뒤로 추가
- BEFORE 특정 열 ⇨ 특정 열 전으로 추가
- SECOND, THIRD 등등 ⇨ 두번째, 세번째 등등으로 추가

ALTER TABLE 변경할 테이블

열 변경

CHANGE COLUMN 원래 열 이름 바뀐 열 이름 | 바뀐 자료형 NOT NULL AUTO-INCREMENT

CHANGE ⇨ 열의 이름, 데이터 타입

열 수정

MODIFY COLUMN 열 이름 | 바뀐 자료형 추가하고 싶은 위치

MODIFY ⇨ 위치, 데이터 타입

테이블 변경 예제 242p

projekts

number	descriptionofproj	contractoronjob
1	outside house painting	Murphy
2	kitchen remodel	Valdez
3	wood floor installation	Keller
3	roofing	Jackson

- ❖ 테이블의 이름 수정
- ❖ 열의 이름이 내용과 맞지 않음
  =>프로젝트 번호, 계약회사의 이름, 기타 설명으로 열의 이름 변경
- ❖ 시작일, 예상금액, 전화번호 열 추가

Project_list

proj_id	proj_desc	con_name
1	outside house painting	Murphy
2	kitchen remodel	Valdez
3	wood floor installation	Keller
4	roofing	Jackson

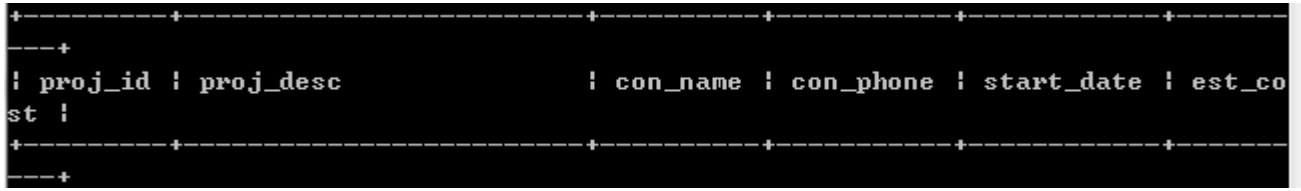
❖ 테이블의 이름 수정

```
ALTER TABLE projekts
RENAME TO project_list;
```

❖ 열의 이름이 내용과 맞지 않음
=>프로젝트 번호, 계약회사의 이름, 기타 설명으로 열의 이름 변경

```
ALTER TABLE project_list
CHANGE COLUMN number proj_id INT NOT NULL AUTO_INCREMENT,
CHANGE COLUMN descriptionofproj proj_desc VARCHAR(100),
CHANGE COLUMN contractoronjob con_name VARCHAR(30),
ADD PRIMARY KEY(proj_id);
```

Project_list



proj_id	proj_desc	con_name	con_phone	start_date	est_cost

❖ 시작일, 예상금액, 전화번호 열 추가

```
ALTER TABLE project_list
ADD COLUMN con_phone VARCHAR(10),
ADD COLUMN start_date DATE,
ADD COLUMN est_cost DECIMAL(7,2);
```

SELECT RIGHT (열 이름, 숫자) FROM 테이블 명

RIGHT (열 이름, 숫자)	열의 오른쪽에서부터 일부 문자들을 선택
LEFT (열 이름, 숫자)	열의 왼쪽에서부터 일부 문자들을 선택
SUBSTRING_INDEX (열 이름, '특정 문자', 숫자)	열 안의 작은 따옴표 안의 문자열을 찾아 그 앞의 모든 문자열을 반환 (숫자는 특정 문자의 순서를 뜻함)
SUBSTRING ('문자열', 숫자, 길이)	작은 따옴표 안의 문자열에서 숫자만큼의 순서인 문자에서부터 시작해 길이만큼 문자열을 반환

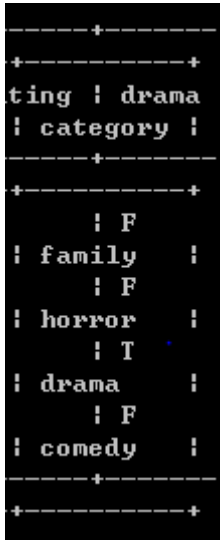
UPPER (‘문자열’)	문자열 모두를 대문자로 변환
LOWER (‘문자열’)	문자열 모두를 소문자로 변환
REVERSE (‘문자열’)	문자열의 순서를 역순으로!
LTRIM(‘ 문자열’)	문자열의 앞(왼쪽부분)에 있는 공백 문자들을 제거
RTRIM(‘문자열 ’)	문자열의 뒤(오른쪽부분)에 있는 공백 문자들을 제거
LENGTH (‘문자열’)	문자열의 문자 수를 반환

고급 SELECT문

Head First SQL

Movie_table

movie_id	title	rating	drama	comedy	action	gore	scifi	for_kids	cartoon	pruchased
1	Monsters,Inc.	G	F	T	F	F	F	T	T	2002-03-06
2	The Godfather	R	F	F	T	T	F	F	F	2002-02-05
3	Gone with the wind	G	T	F	F	F	F	F	F	1999-11-20
4	American Pie	R	F	T	F	F	F	F	F	2003-04-19



방법1

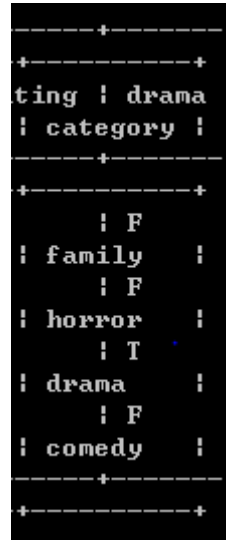
방법2

```
ALTER TABLE movie_table
ADD COLUMN category VARCHAR(10);
UPDATE movie_table SET category = 'drama' where drama ='T';
UPDATE movie_table SET category = 'comedy' where comedy ='T';
UPDATE movie_table SET category = 'action' where action ='T';
UPDATE movie_table SET category = 'horror' where gore ='T';
UPDATE movie_table SET category = 'scifi' where scifi ='T';
UPDATE movie_table SET category = 'family' where for_kids ='T';
UPDATE movie_table SET category = 'family' where cartoon ='T' AND rating='G';
UPDATE movie_table SET category = 'misc' where cartoon ='T' AND rating <>'G';
```

순서가 분류할 때 영향을 미친다!

Movie_table

movie_id	title	rating	pruchased	category	scif
1	Monsters,Inc.	G	2002-03-06	comedy	F
2	The Godfather	R	2002-02-05	drama	
3	Gone with the wind	G	1999-11-20	drama	F
4	American Pie	R	2003-04-19	comedy	F



방법1

방법2

```
UPDATE movie_table
SET category=
CASE
  WHEN drama ='T' THEN 'drama'
  WHEN comedy ='T' THEN 'comedy'
  WHEN action ='T' THEN 'drama'
  WHEN gore ='T' THEN 'gore'
  WHEN scifi ='T' THEN 'scifi'
  WHEN for_kids ='T' THEN 'family'
  WHEN  cartoon ='T' AND rating = 'G' THEN 'family'
  ELSE 'MISC'
END;
```

Movie_table

movie_id	title	rating	pruchased	category
1	Monsters,Inc.	G	2002-03-06	comedy
2	The Godfather	R	2002-02-05	drama
3	Gone with the wind	G	1999-11-20	drama
4	American Pie	R	2003-04-19	comedy

드라마에 속하는 영화를 찾을 때

```
SELECT title, category
FROM movie_table
WHERE
category='drama'
ORDER BY title;
```

title	category
Gone with the wind	drama
The Godfather	drama

내림차순- DESC
오름차순- ASC(기본값)

Cookie_sales

id	first_name	sales	sale_date
1	Lindsay	32	2007-03-06
2	Paris	27	2007-03-06
3	Britney	11	2007-03-06
4	Nicole	19	2007-03-06
5	Lindsay	9	2007-03-07
6	Paris	2	2007-03-07
7	Britney	43	2007-03-07
8	Nicole	8	2007-03-07
9	Lindsay	18	2007-03-08
10	Paris	24	2007-03-08
12	Britney	3	2007-03-08

first_name	SUM(sales)	AVG(sales)
Lindsay	59	19.6667
Britney	57	19.0000

합계와 평균 | 최대와 최소 | 중복제거

```
SELECT first_name, SUM(sales), AVG(sales) ,
COUNT(sales)
FROM cookie_sales
GROUP BY first_name
ORDER BY 2 DESC
LIMIT 2;
```

Cookie_sales

id	first_name	sales	sale_date
1	Lindsay	32	2007-03-06
2	Paris	27	2007-03-06
3	Britney	11	2007-03-06
4	Nicole	19	2007-03-06
5	Lindsay	9	2007-03-07
6	Paris	2	2007-03-07
7	Britney	43	2007-03-07
8	Nicole	8	2007-03-07
9	Lindsay	18	2007-03-08
10	Paris	24	2007-03-08
12	Britney	3	2007-03-08

first_name	MAX(sales)	MIN(sales)
Britney	43	3
Lindsay	32	9
Nicole	19	8
Paris	27	2

합계와 평균 | **최대와 최소** | 중복제거

```
SELECT first_name, MAX(sales), MIN(sales)
FROM cookie_sales
GROUP BY first_name;
```

```
mysql> SELECT sale_date
    -> FROM cookie_sales
    -> ORDER BY sale_date;
+------------+
| sale_date  |
+------------+
| 2007-03-06 |
| 2007-03-06 |
| 2007-03-06 |
| 2007-03-06 |
| 2007-03-07 |
| 2007-03-07 |
| 2007-03-07 |
| 2007-03-07 |
| 2007-03-08 |
| 2007-03-08 |
| 2007-03-08 |
+------------+
```

DISTINCT 추가

```
mysql> SELECT DISTINCT sale_date
    -> FROM cookie_sales
    -> ORDER BY sale_date;
+------------+
| sale_date  |
+------------+
| 2007-03-06 |
| 2007-03-07 |
| 2007-03-08 |
+------------+
```

합계와 평균 | 최대와 최소 | **중복제거**

```
SELECT DISTINCT sale_date
FROM cookie_sales
ORDER BY sale_date;
```

Thank you