

cmd에서 sql 접속

mysql -u root -p

-> 나와도 이어쓰기 가능

\*\*대소문자 모두 동일처리\*\*

DB 목록 보기

show databases;

DB 생성

create database [DB\_name];

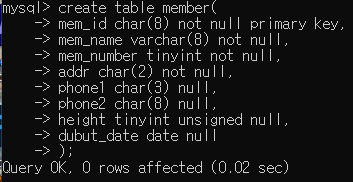
DB 삭제

drop database [DB\_name];

DB 사용(선택)

use [DB\_name];

table 생성



CREATE TABLE member(  
mem\_id CHAR(8) NOT NULL PRIMARY KEY,  
mem\_name VARCHAR(10) NOT NULL,   
mem\_number TINYINT NOT NULL,   
addr CHAR(2) NOT NULL,  
phone1 CHAR(3) NULL,  
phone2 CHAR(8) NULL,  
height TINYINT UNSIGNED NULL,   
debut\_date DATE NULL  
);

tinyint : 자바에서 byte와 유사, -128 ~ 127

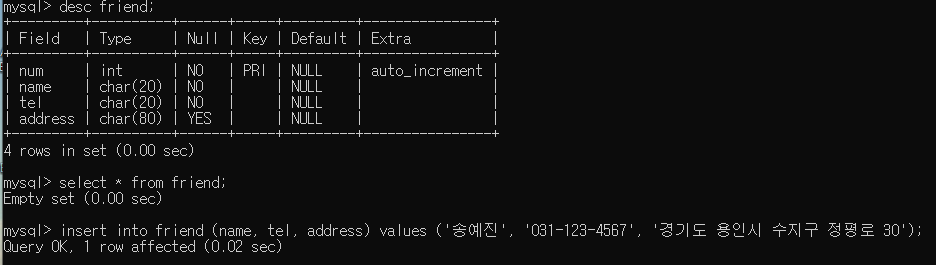
tinyint unsigned : 0~255, 음수x 음수만큼 양수 범위가 늘어남

table구조 보기

desc [table\_name];

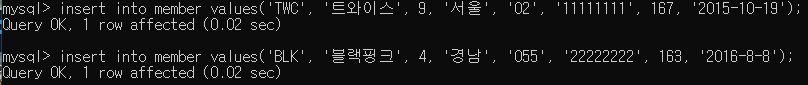
내용 추가

insert into [table\_name] (필드 나열) values (내용 작성);



나열한 필드만큼 내용 작성

insert into [table\_name] values (내용 작성);



모든 필드 수 만큼 내용을 작성

auto\_increment와 같이 직접 입력해주지 않아도 되는 값에 대해서는 null 넣기

table 지우기

drop table [table\_name];

table 내용 보기

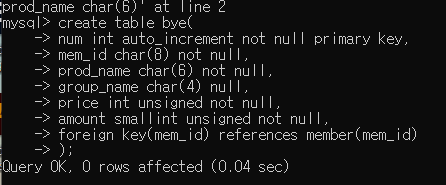
select \* from [table\_name];

field 지정해서 보기

select [field1], [field2] from [table\_name]

table 이름 변경

rename table [기존이름] to [새이름]



맨 뒤에 field 추가

alter table [table\_name] add [field\_name] [field\_type];

field 지우기

alter table [table\_name] drop [field\_name];

위치 지정해서 필드 추가

\*(field\_name0) 뒤에 추가

alter table [table\_name] add [field\_name] [field\_type] after [field\_name0]

field 수정

alter table [table\_name] change [old\_field\_name] [new\_field\_name] [new\_field\_type]

field type 수정

alter table [table\_name] modify [field\_name] [field\_new\_type]

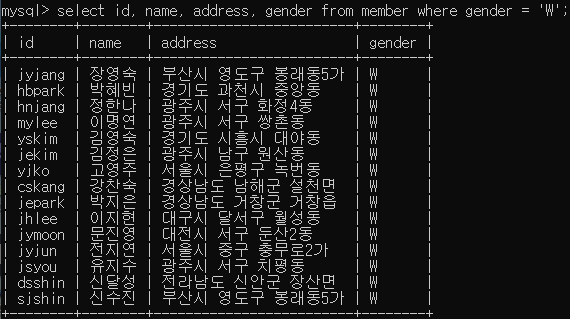
table 삭제

drop table [table\_name];

조건 검색

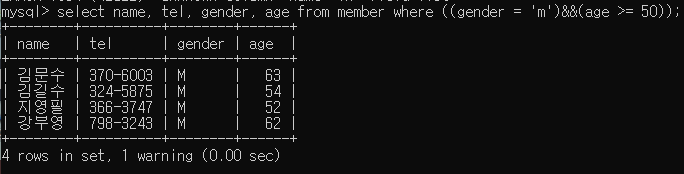
select [field1], [field2] from [table\_name] where [조건]

\*\*여성회원의 아이디, 이름, 주소, 성별 검색

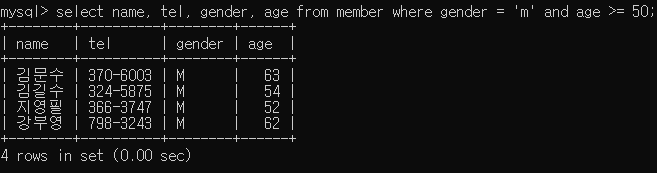


((조건1) && (조건2))

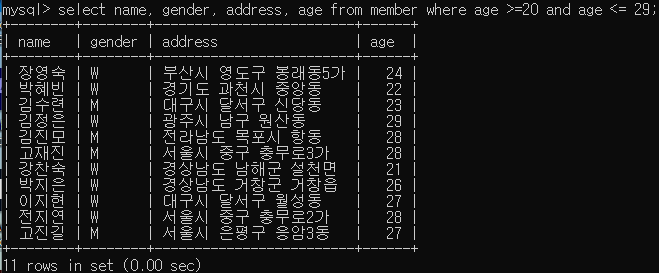
\*\*50세 이상인 남성 회원의 이름, 전화번호, 성별, 나이 검색



조건1 and 조건2

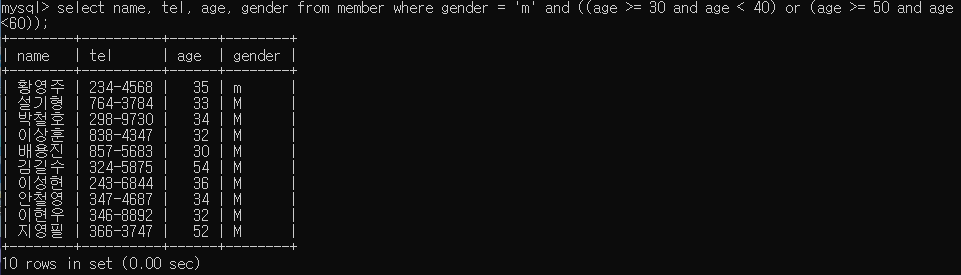


\*\*20대 회원의 이름 성별, 주소, 나이 검색

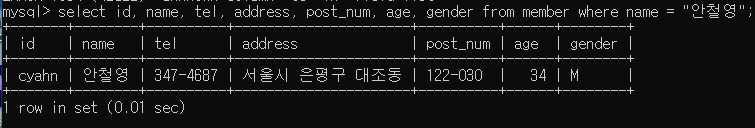


\*\*30대 또는 50대 남자 회원이 이름, 전화번호, 나이, 성별 검색

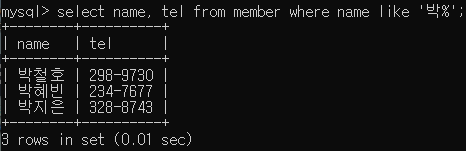
select name, tel, age, gender from member where gender = 'm' and ((age >= 30 and age < 40) or (age >= 50 and age <60));



\*\* “안철영”의 일련번호, 이름, 전화번호, 주소, 우편번호, 나이, 성별 검색



\*\* 박씨 이름과 전화번호 검색

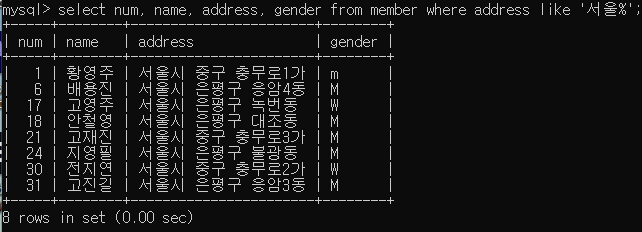


like ‘%A’ : A로 끝나는

like ‘A%’ : A로 시작하는

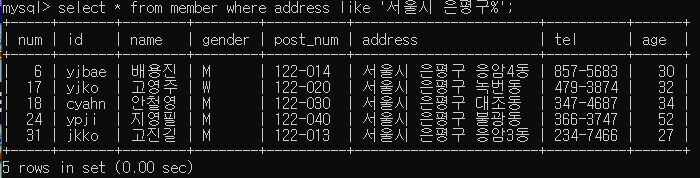
like ‘%A%’ : 중간에 A가 포함된

\*\*서울에 거주하는 회원들의 일련번호, 이름, 주소, 성별 검색

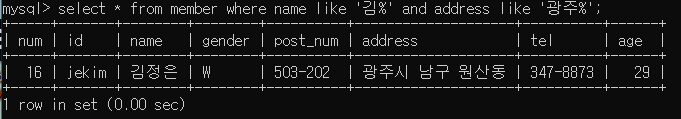


\*\* 은평구에 거주하는 회원의 모든 정보 검색

like ‘%은평구%’



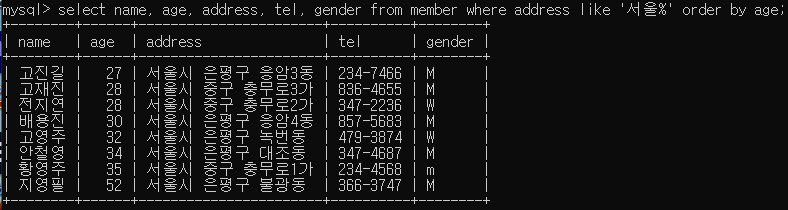
\*\* 광주에 거주하는 김씨 회원들의 모든 정보 검색



레코드 정렬

select [field\_name1], [field\_name2] from [table\_name] order by [기준이 되는 field\_name]

\*\* 서울에 사는 회원을 나이 오름차순으로 정렬한 뒤, 이름, 나이, 주소, 전화번호, 성별 검색



\*\* 부산에 사는 회원을 나이 내림차순으로 정렬한 뒤, 이름, 나이, 주소, 전화번호 검색

select name, age, address, tel from member where address like '부산%' order by age desc;

desc : 내림차순(descending)

레코드 수정

update 테이블명 set 필드명=필드값 where 조건식

\*\*고재진의 전화번호를 123-4567로 변경

update member set tel = ‘123-4567’ where name = “고재진”;

\*\* 정한나의 나이를 37로 변경

update member set age = 37 where name = “정한나”;

레코드 삭제

delete from 테이블명 where 조건식

\*\*김수련의 레코드 삭제

delete from member where name = ‘김수련’;

\*\* 40대 회원의 레코드를 삭제하고 35~53세 회원의 이름, 나이, 성별을 나이순으로 출력

delete from member where age >= 40 and age < 50;

select name, age, gender from member age >= 35 and age <= 53 order by age;

delete from 테이블명;

테이블 전체 레코드 삭제

조건(where)을 생략하면 테이블의 전체 레코드를 삭제할 수 있으니 주의

table 출력

show tables;

열 추가

alter table [table\_name] add column [name] [type];

열 이름 변경

alter table [table\_name] change column [name] [new\_name] [type];

열 타입 변경

alter table [table\_name] modify column [name] [new\_type];

열 삭제

alter table [table\_name] drop column [name];

.sql 파일로 저장

\*\*\*\*CMD에서

mysqldump -uroot -p1234 [table\_name] >[filename.sql]



A as B, A(value)의 제목을 B로 표현하겠다.

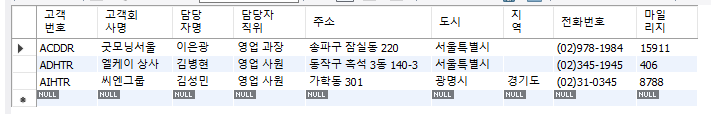




limit : 개수제한

고객 테이블에서 1행부터 시작하여 3개의 고객정보를 검색



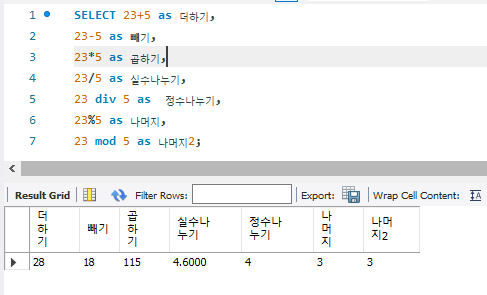


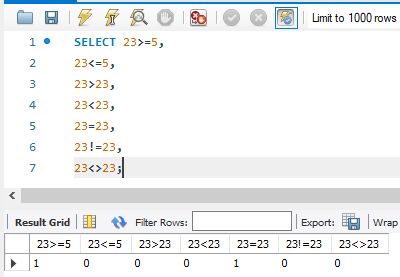


limit n, k; n행부터 k개



distinct 중복데이터 제거





!=, <> 둘 다 “다르다”는 의미를 가짐

지역 값에 들어있지 않는 고객의 정보 검색

select \* from 고객 where 지역 is null;

CSV 파일에서 테이블을 읽어올 때, 값이 들어있지 않는 셀은 “”(빈문자열)

select \* from 고객 where 지역 = “”;

그래서 빈 문자열을 null로 변경

update 고객 set 지역 = null where 지역 = “”; (x : safemode는 전체 테이블에서 업데이트 방지)

* SET SQL\_SAFE\_UPDATES = 0; 을 앞에 선언
* 이후 SET SQL\_SAFE\_UPDATES = 1;로 safemode 복구

Edit> Preferences > SQL Editor > Safe Updates 옵션 설정 후 재부팅 해도 됨.

null에 대한 접근은 = 나 !=는 사용하지 않고, is나 is not을 통해 접근한다

select 고객번호, 담당자명, 담당자직위 from 고객 where 담당자직위 = '영업 과장' or 담당자직위 = '마케팅 과장';

select 고객번호, 담당자명, 담당자직위 from 고객 where 담당자직위 **in** ('영업 과장', '마케팅 과장');

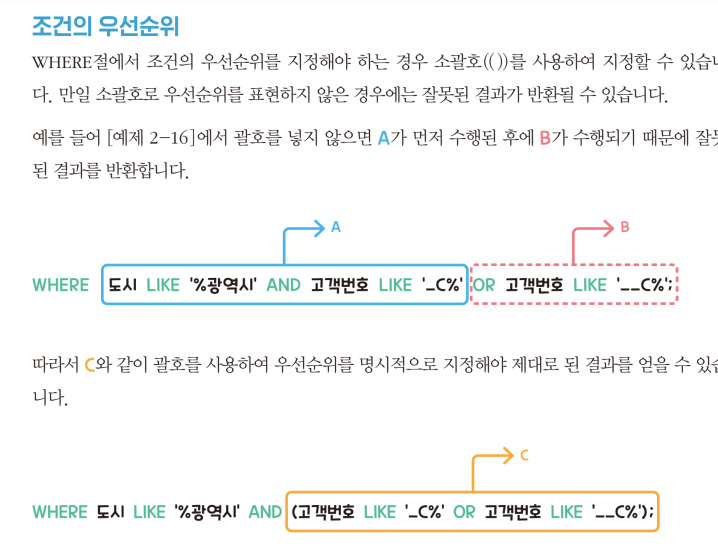
select 담당자명, 마일리지 from 고객 where 100000 <= 마일리지 && 마일리지 <= 200000;

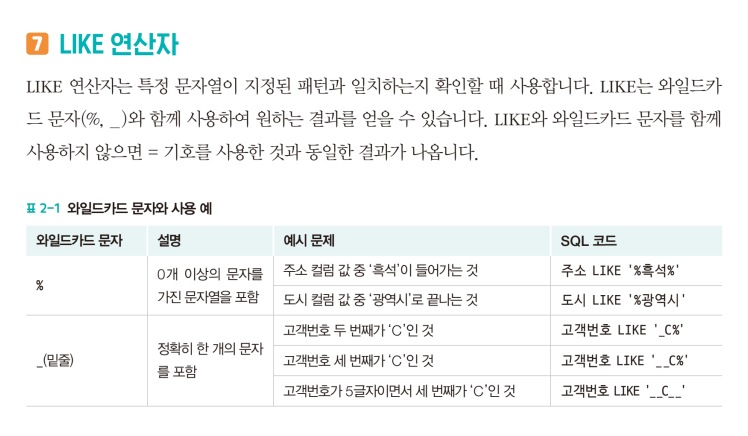
select 담당자명, 마일리지 from 고객 where 마일리지 **between** 100000 and 200000;

SELECT \* FROM 고객 WHERE 도시 LIKE "%광역시" AND (고객번호 LIKE '\_C%' OR 고객번호 LIKE '\_\_C%');

\_C : 두번째 글자가 C

\_\_C : 세번째 글자가 C





글자의 길이

select char\_length('hello'), // 영문자 개수 반환 5

length('hello'), // 영문자 바이트 수 5

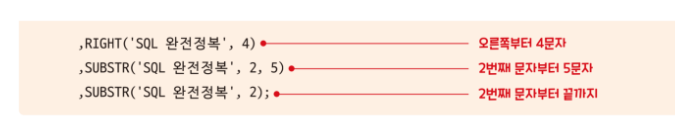
char\_length('안녕'), // 문자 개수2

length('안녕'); // utf-8 바이트 수 6

select concat('dreams', 'come', 'true'), // dreamscometrue

concat\_ws('-','2025','04','10'); // 2025-04-10

SELECT LEFT("SQL 완전정복", 3) -> 왼쪽부터 지정한 길이만큼의 문자열 반환



(문자열, 구분할 문자, 개수)

SELECT SUBSTRING\_INDEX('서울시 동작구 흑석로', ' ', 2)

,SUBSTRING\_INDEX('서울시 동작구 흑석로', ' ', -2);

(문자열, 개수, 채울 문자)

SELECT lpad('SQL', 10, '#'), // 문자열 앞에 전체길이가 될 때까지 문자를 채운다.

rpad('SQL', 5, '\*'); // 문자열 뒤에 전체길이가 딜 때까지 문자를 채운다

글자 지우기(TRIM) -> 글자수(LENGTH)

SELECT LENGTH(LTRIM(' SQL '))

,LENGTH(RTRIM(' SQL '))

,LENGTH(TRIM(' SQL '));

SELECT TRIM(BOTH 'abc' FROM 'abcSQLabcabc') // 양쪽 모든 abc 제거

,TRIM(LEADING 'abc' FROM 'abcSQLabcabc') // 왼쪽 모든 abc 제거

,TRIM(TRAILING 'abc' FROM 'abcSQLabcabc'); // 오른쪽 모든 abc 제거

SELECT repeat('\*', 5); // \*\*\*\*\*

SELECT replace('010.1234.5678', '.', '-'); .을 -로 바꾸기

SELECT CEILING(123.56), // 올림

FLOOR(123.56), // 버림

ROUND(123.56), // 반올림

ROUND(123.56, 1), // 소수 둘째자리 반올림, 첫째자리 표기

TRUNCATE(123.56, 1); // 소수 한자리까지 남김, 둘째자리부터 버림

나머지 구하기

SELECT MOD(203, 4), 203 % 4, 203 MOD 4;

SELECT POWER(2,3), // 2^3

SQRT(16), // 루트 16

RAND(), // 0~1 무작위수

RAND(100), // SEED 설정

round(RAND()\*100); // 0~100

SELECT NOW(), sysdate(), // 현재시간 년-월-일 시:분:초

curdate(), // 현재 날짜만 년-월-일

curtime(); // 현재 시간만 시:분:초

SELECT NOW()

,YEAR(NOW())

,QUARTER(NOW()) // 분기

,MONTH(NOW())

,DAY(NOW())

,HOUR(NOW())

,MINUTE(NOW())

,SECOND(NOW());

SELECT NOW()

,DATEDIFF('2026-12-20', NOW()) // 앞 – 뒤

,DATEDIFF(NOW(), '2026-12-20') // 뒤 – 앞

,timestampdiff(YEAR ,NOW(), '2026-12-20') // 년 차이

,timestampdiff(MONTH ,NOW(), '2026-12-20') // 월 차이

,timestampdiff(DAY ,NOW(), '2026-12-20'); // 일 차이

SELECT NOW()

,adddate(NOW(), 50)

,adddate(now(), interval 50 DAY) // 50일 후

,adddate(now(), interval 50 MONTH) // 50 개월 후

,subdate(now(), interval 50 YEAR); // 50년 전

SELECT NOW()

,LAST\_DAY(NOW()) // 마지막 날자

,dayofyear(NOW()) // 1년중 몇번째 일인지

,DAYOFMONTH(NOW())

,WEEKDAY(now());

형변환

SELECT CAST('1' AS UNSIGNED INTEGER),

CAST('2' AS CHAR(1)),

CONVERT('1', UNSIGNED INTEGER),

CONVERT('2', CHAR(1));

SELECT IF(12500 \* 450 > 5000000, '초과달성', '미달성');

(조건, TRUE, FALSE);

IF

SELECT CASE

WHEN 12500 \* 450 > 5000000 THEN '초과달성'

WHEN 2500 \* 450 > 4000000 THEN '달성'

ELSE '미달성'

END;

select 고객회사명,

concat("\*\*", SUBSTR(고객회사명, 3)) AS 고객회사2,

전화번호,

replace(substr(전화번호, 2), ')', '-') AS 전화번호2

FROM 고객;



SELECT \*,

단가 \* 주문수량 AS 주문금액,

TRUNCATE(단가 \* 주문수량 \* 할인율, -1) AS 할인금액,

단가 \* 주문수량 - TRUNCATE(단가 \* 주문수량 \* 할인율, -1) AS 실주문금액

FROM 주문세부;



SELECT 담당자명

,고객회사명

,도시

,IF(도시 LIKE '%특별시' OR 도시 LIKE '%광역시', '대도시', '도시') AS 도시구분

,마일리지

,CASE WHEN 마일리지 >= 100000 THEN 'VVIP고객'

WHEN 마일리지 >= 10000 THEN 'VIP고객'

ELSE '일반고객'

END AS 마일리지구분

FROM 고객



SELECT 주문번호, 고객번호, 주문일

, YEAR(주문일) AS '주문년도'

, quarter(주문일) AS '주문분기'

, month(주문일) AS '주문월'

, day(주문일) AS '주문일'

, weekday(주문일) AS '주문요일'

, CASE weekday(주문일) WHEN 0 THEN '월요일'

WHEN 1 THEN '화요일'

WHEN 2 THEN '수요일'

WHEN 3 THEN '목요일'

WHEN 4 THEN '금요일'

WHEN 5 THEN '토요일'

ELSE '일요일'

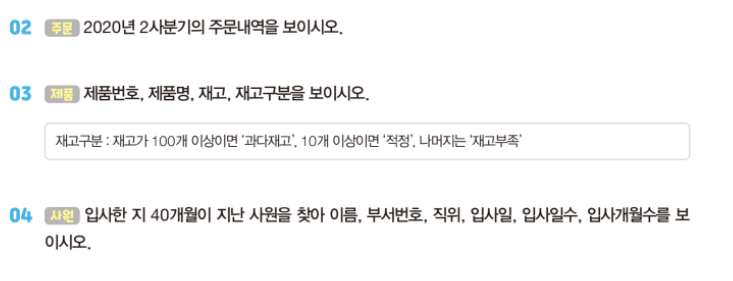
END AS '한글요일'

from 주문;



SELECT \*, datediff(발송일, 요청일) AS 지연일수

from 주문 WHERE datediff(발송일, 요청일) >= 7;



SELECT \* from 주문 WHERE (YEAR(주문일) = 2020) AND (quarter(주문일) = 2);

SELECT 제품번호, 제품명, 재고,

CASE 재고 WHEN 재고 > 100 THEN '과다재고'

WHEN 재고 >= 10 THEN '적정'

ELSE '재고부족'

END AS 재고구분 FROM 제품;

SELECT 이름, 부서번호, 직위, 입사일,

datediff(NOW(), 입사일) AS 입사일수,

TIMESTAMPDIFF(MONTH, 입사일, NOW()) AS 입사개월수

FROM 사원 WHERE TIMESTAMPDIFF(MONTH, 입사일, NOW()) >= 40;

root 진입

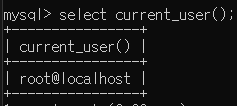
mysql -u root -p

mysql 사용

use mysql

현재 로그인한 사용자

select current\_user();



새로운 user 생성

CREATE USER [name]@localhost IDENTIFIED BY ‘[PASSWORD]’;

특정 DB에 권한주기

GRANT ALL ON [DB\_NAME].\* TO [name]@localhost;

flush privileges; 설정(변경사항) 저장

모든 DB에 대한 권한

GRANT ALL ON \*.\* TO [name]@localhost;

GRANT ALL ON \*.\* TO [name]@%localhost;

SELECT, INSERT 권한만 주기

GRANT SELECT, INSERT ON [DB\_NAME].\* TO [name]@localhost;

특정 DB에 포함된 TABLE에 권한 주기

GRANT ALL ON [DB.TABLE] TO [name]@localhost

ex) GRANT ALL ON test.frient TO [name]@localhost

쓰기, 수정 권한 삭제

REVOKE INSERT, UPDATE ON [DB].\* FROM [name];

전체 권한 삭제

REVOKE ALL ON [DB].\* FROM [name];

사용자 전체 쓰기 권한 삭제

REVOKE INSERT ON \*.\* TO [name]

사용자 삭제

DROP USER [name]@localhost

localhost 와 % 차이

localhost 로컬 컴퓨터(자기 자신)에서만 접속 허용

% 어디서든 접속 허용(외부 ip에서도 => 보안 위험, 방화벽 필수)

CREATE USER [user@192.168.1.10](mailto:user@192.168.1.10) IDENTIFIED BY ‘1234’;

특정 IP에서만 접근 가능하도록

집계 함수의 개념

SELECT [집계함수] FORM [TABLE] WHERE [조건];



EX) 대전에 있는 모든 고객의 마일리지 합

SELECT SUM(마일리지) AS 마일리지합 FROM 고객 WHERE 도시 LIKE "대전%";



SELECT count(\*) // NULL과 상관없이 모든 CELL의 개수를 셈.

, COUNT(고객번호) , COUNT(도시) , COUNT(지역)

FROM 고객;

합, 평균, 최소값, 최대값

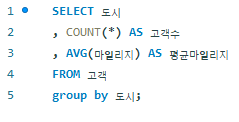
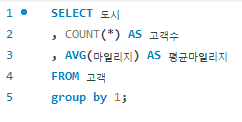
SELECT SUM(마일리지), AVG(마일리지), MIN(마일리지), MAX(마일리지)

FROM 고객;

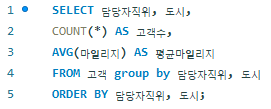
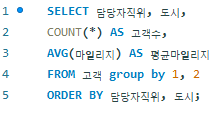
SELECT SUM(마일리지), AVG(마일리지), MIN(마일리지), MAX(마일리지)

FROM 고객 WHERE 도시 = "서울특별시"

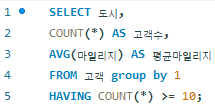
도시 별 고객의 수와 해당 도시 고객들의 평균 마일리지

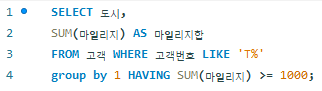
SELECT와 GROUP BY에 동일한 COLUMN 이름 혹은 순번을 넣어줘야함

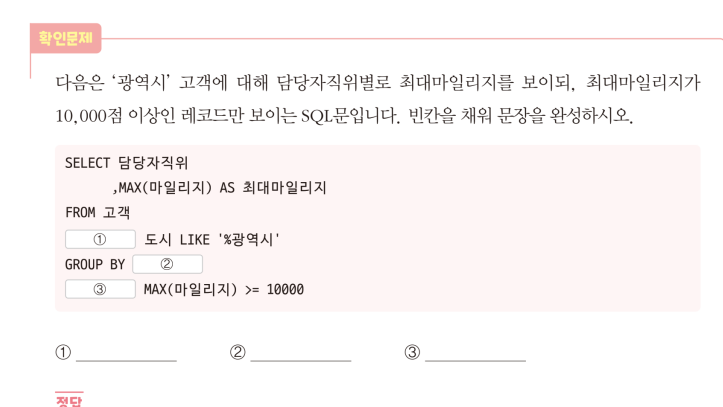
 

예제 4-6  
고객 테이블에서 도시 별 그룹으로 묶어 고객수와 평균 마일리지  
이 중 고객수가 10 이상인 레코드 보기  
GROUP BY에 대한 조건을 걸어줄 때 : HAVING



고객번호가 T로 시작하는 고객에 대해 도시별로 묶어 고객의 마일리지 합 구하고, 마일리지 합이 1000 이상인 레코드 보기





SELECT 담당자직위

,MAX(마일리지) AS 최대마일리지

FROM 고객

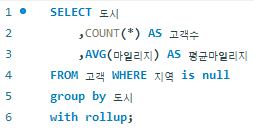
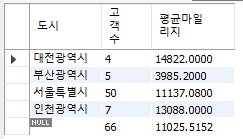
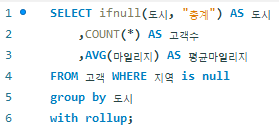
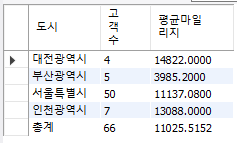
WHERE 도시 LIKE '%광역시'

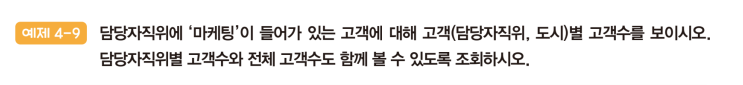
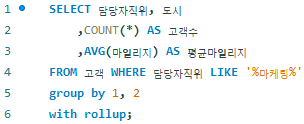
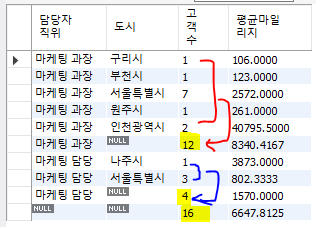
GROUP BY 1

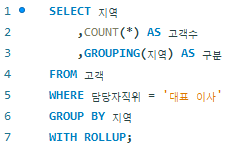
HAVING MAX(마일리지) >= 10000;

WITH ROLLUP

그룹별 소개와 전체 총계 구하기

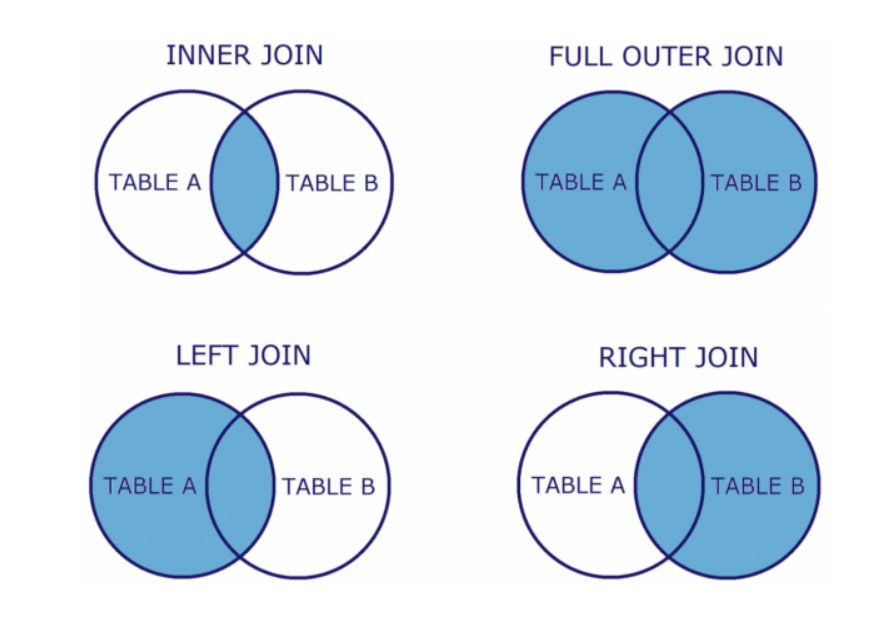
GROUPING : ROLL UP 결과로 나온 NULL에 대해 1 나머지는 0

GROUP\_CONCAT() 모든 COLUMN 값을 출력(중복 상관 없이 모두)

JOIN

: 2개 이상의 테이블을 연결하여 DATA를 검색하는 방법  
서로 다른 테이블에서 저장된 관련된 데이터를 함께 가져와 하나의 결과로 표시  
검색하고 싶은 COLUMN이 다른 테이블에 있을 때 사용,  
여러 개의 테이블을 마치 하나의 테이블인 것처럼 쓸 수 있다.



snl\_show에 host,로 출연한 celeb을 기준으로, celeb table과 snl\_show table을 inner join하여 각 table의 id와 이름 조회

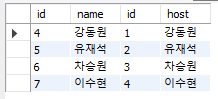
SELECT celeb.id, celeb.name, snl\_show.id, snl\_show.host

from celeb

inner join snl\_show

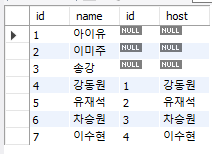
on celeb.name = snl\_show.host;

inner

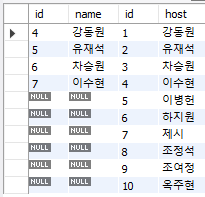


left

두 테이블에서 공동 영역을 포함하고, 왼쪽(먼저 나온)테이블에 다른 데이터를 포함하는 join 방식  
아이유, 이미주, 송강은 host에 없어 null



right  
host 중 celeb에는 없어서 null



full outer join

두개의 테이블에서 양쪽 테이블을 모든 영역을 포함  
full join은 sql에서 지원하지 않는 방식.

* left join과 right join을 unio(합집합)을 통해 full outer join으로 만듦

select celeb.id, celeb.name, snl\_show.id, snl\_show.host

from celeb

left join snl\_show

on celeb.name = snl\_show.host

union

select celeb.id, celeb.name, snl\_show.id, snl\_show.host

from celeb

right join snl\_show

on celeb.name = snl\_show.host;

NON.ANSI JOIN

inner join과 유사하고, table 간의 공통된 데이터를 다루는데 가장 자주 사용

SELECT celeb.id, celeb.name, snl\_show.id, snl\_show.host

from celeb, snl\_show

where celeb.name = snl\_show.host;

inner join -> , / on -> where

문제1) celeb 중 snl\_show에 출연하고, 소속사가 안테나인 셀럽의 이름과 직업

inner join(ANSI) 먼저

ex) select celeb.name, celeb.job\_title

from celeb

inner join snl\_show

on celeb.name = snl\_show.host

where celeb.agency = '안테나';

NON.ANSI 방법

select celeb.name, celeb.job\_title

from celeb, snl\_show

where celeb.name = snl\_show.host and celeb.agency = '안테나';

UNION

여러 개의 SQL문을 합쳐 하나의 SQL문으로 만들어주는 방법, (주의 – CLOUMN 개수가 같아야함)

test1에 1, 2, 3, test2에 5, 6, 3일 때 union을 하면 1, 2, 3, 5, 6이 나옴

성별이 여자이고, 쿼리와 소속사가 YG인 데이터를 UNION으로 조회

SELECT \* FROM celeb WHERE SEX = 'f'

UNION (ALL) # ALL은 중복 포함

SELECT \* FROM celeb WHERE AGENCY LIKE 'YG%';

데이터 베이스 테이블을 스캔할 때 중복된 행을 하나만 반환할 가능성이 높음

SELECT DISTINCT \* FROM celeb WHERE SEX = 'f' or AGENCY LIKE 'YG%';

문제 5-2

‘이소미’ 사원의 사원번호, 직위, 부서번호, 부서명을 보이기  
사원 테이블에서 부서테이블을 inner join하고 NONANSI join도 써보기

SELECT 사원.사원번호, 사원.직위, 사원.부서번호, 부서.부서명

FROM 사원 inner join 부서

on 사원.이름 = '이소미' and 사원.부서번호 = 부서.부서번호;

SELECT 사원.사원번호, 사원.직위, 사원.부서번호, 부서.부서명

FROM 사원, 부서

where 이름 = '이소미' and 사원.부서번호 = 부서.부서번호;

고객과 주문 join  
고객 회사들이 주문한 주문건수를 주문건수가 많은 순서대로 보이고,  
고객회사의 정보로는 고객번호, 담당자명, 고객회사명을 보이기

SELECT 고객.고객번호, 고객.담당자명, 고객.고객회사명, count(\*) as 고객건수

FROM 고객 inner join 주문

on 고객.고객번호 = 주문.고객번호

group by 1, 2, 3

order by 고객건수 desc;

SELECT 고객.고객번호, 고객.담당자명, 고객.고객회사명, count(\*) as 고객건수

FROM 고객, 주문

where 고객.고객번호 = 주문.고객번호

group by 1, 2, 3

order by 고객건수 desc;

고객 테이블과 마일리지 등급 join

고객 테이블에서 담당자가 ‘이은광’인 고객 번호, 고객회사명, 담당자명, 마일리지와 마일리지 등급 보이기

SELECT 고객.고객번호,고객.고객회사명, 고객.담당자명, 고객.마일리지, 마일리지등급.등급명

FROM 고객 inner join 마일리지등급

on 마일리지 between 하한마일리지 and 상한마일리지

where 담당자명 = '이은광';

SELECT 고객.고객번호,고객.고객회사명, 고객.담당자명, 고객.마일리지, 마일리지등급.등급명

FROM 고객, 마일리지등급

where (마일리지 between 하한마일리지 and 상한마일리지)

and 담당자명 = '이은광';

Inner Join 연습 문제 5개

1⃣ 학생 이름과 그 학생이 신청한 과목명을 모두 출력하세요.

select \* from 학생

inner join 수강신청

on 수강신청.학번 = 학생.학번

2⃣ 전공이 \*\*'컴퓨터공학'\*\*인 학생들의 이름과 수강한 과목명을 출력하세요.

select 학생.이름, 수강신청.과목명

from 학생 inner join 수강신청

on 수강신청.학번 = 학생.학번

where 전공 = '컴퓨터공학';

3⃣ 성적이 \*\*'A+'\*\*인 학생 이름과 과목명을 출력하세요.

select 학생.이름, 수강신청.과목명

from 학생 inner join 수강신청

on 수강신청.학번 = 학생.학번

where 성적 = 'A+';

4⃣ 학생 이름, 전공, 신청한 과목명, 성적을 전부 출력하고 성적순으로 내림차순 정렬하세요.

select 학생.이름, 학생.전공, 수강신청.과목명, 수강신청.성적

from 학생 inner join 수강신청

on 수강신청.학번 = 학생.학번

order by 수강신청.성적 desc;

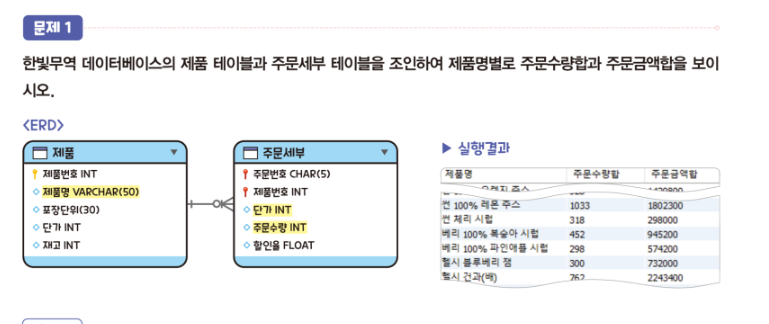
5⃣ 학생 이름별 신청한 과목 수를 출력하세요. (이름, 과목 수)

select 학생.이름, count(\*) as 과목수

from 학생 inner join 수강신청

on 수강신청.학번 = 학생.학번

group by 1;



SELECT 제품.제품명, sum(주문수량) as 주문수량합, sum(주문세부.단가 \* 주문세부.주문수량) as 주문금액합

FROM 제품 inner join 주문세부

on 제품.제품번호 = 주문세부.제품번호

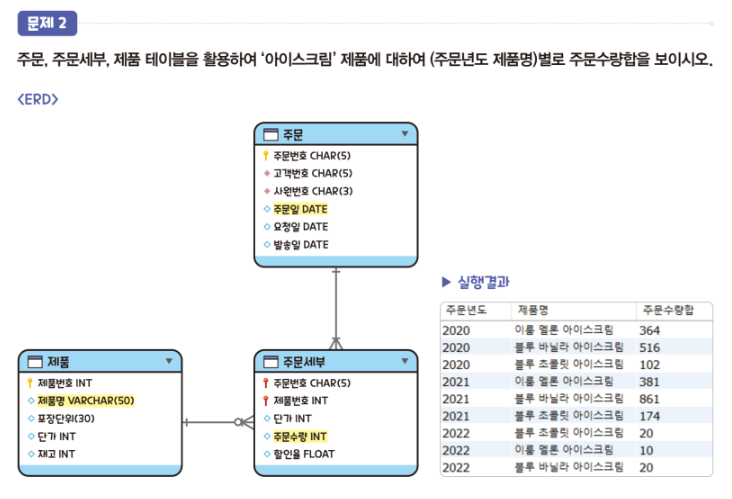
group by 1;

SELECT 제품.제품명, sum(주문수량) as 주문수량합, sum(주문세부.단가 \* 주문세부.주문수량) as 주문금액합

FROM 제품, 주문세부

where 제품.제품번호 = 주문세부.제품번호

group by 1;



SELECT YEAR(주문.주문일) AS 주문년도, 제품명, SUM(주문수량) AS 주문수량합

FROM 제품 INNER JOIN 주문세부

ON 제품.제품번호 = 주문세부.제품번호

INNER JOIN 주문

ON 주문.주문번호 = 주문세부.주문번호

WHERE 제품명 LIKE '%아이스크림'

group by 1, 2;

SELECT YEAR(주문.주문일) AS 주문년도, 제품명, SUM(주문수량) AS 주문수량합

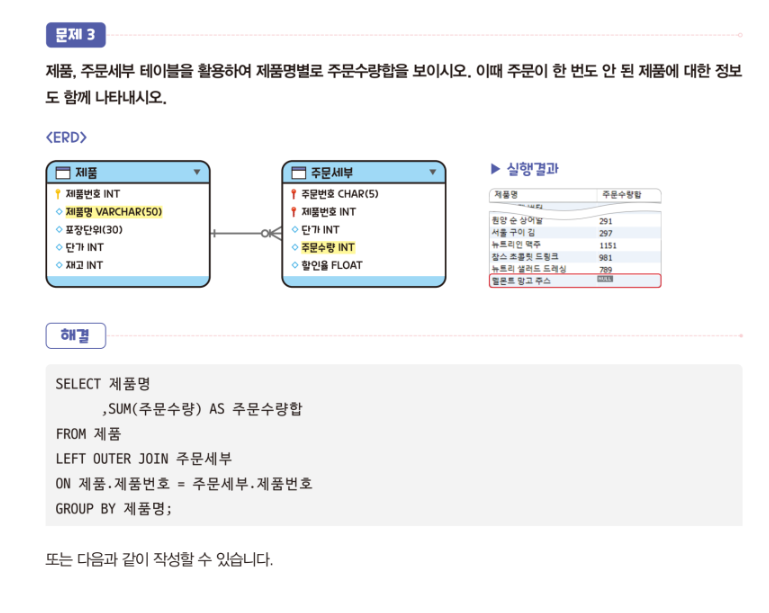
FROM 제품, 주문세부, 주문

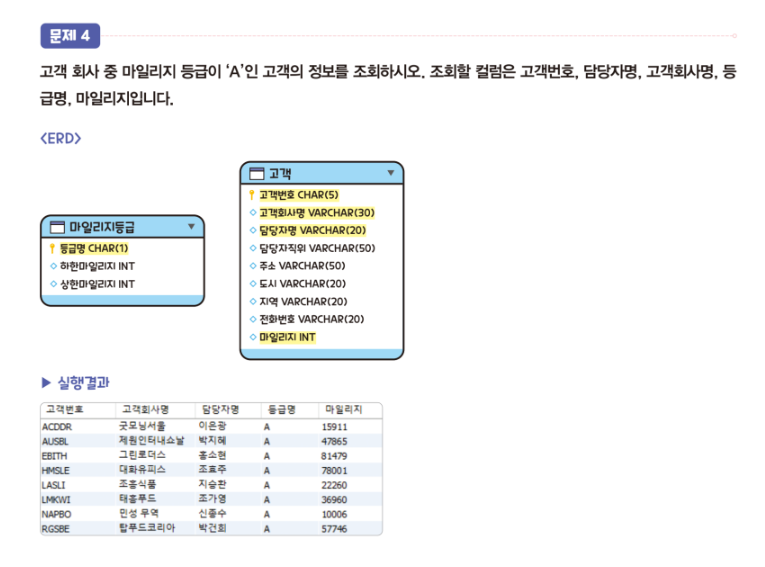
WHERE 제품.제품번호 = 주문세부.제품번호

AND 주문.주문번호 = 주문세부.주문번호

AND 제품명 LIKE '%아이스크림'

group by 1, 2;





SELECT 고객번호, 고객회사명, 담당자명, 마일리지등급.등급명, 마일리지

FROM 고객 INNER JOIN 마일리지등급

ON 마일리지 BETWEEN 하한마일리지 AND 상한마일리지

WHERE 마일리지등급.등급명 = 'A';

문제 1:

고객의 이름과 해당 고객이 주문한 주문 ID를 함께 조회하세요.

SELECT 이름, 주문.주문ID

FROM 고객 INNER JOIN 주문

ON 고객.고객ID = 주문.고객ID;

\*\* 고객을 c로 주문을 o로 지정 \*\*

SELECT c.이름, o.주문ID

FROM 고객 c

INNER JOIN 주문 o

ON c.고객ID = o.고객ID;

문제 2:

주문한 고객의 이름과 주문일자, 총금액을 조회하세요.

SELECT 이름, 주문.주문일자, 주문.총금액

FROM 고객 INNER JOIN 주문

ON 고객.고객ID = 주문.고객ID;

문제 3:

VIP 등급 고객의 이름과 그 고객의 모든 주문 ID를 조회하세요.

SELECT 이름, 주문.주문ID

FROM 고객 INNER JOIN 주문

ON 고객.고객ID = 주문.고객ID

WHERE 고객.등급 = "VIP"

문제 4:

총금액이 10만원 이상인 주문을 한 고객의 이름과 주문 ID, 총금액을 조회하세요.

SELECT 이름, 주문.주문ID, 주문.총금액

FROM 고객 INNER JOIN 주문

ON 고객.고객ID = 주문.고객ID

WHERE 주문.총금액 >= 100000;

문제 5:

각 고객별로 주문 건수를 조회하고, 고객 이름과 주문 건수를 함께 출력하세요. (힌트: GROUP BY 절과 COUNT() 함수를 사용하세요.)

SELECT 이름, COUNT(주문.주문ID) AS 주문건수

FROM 고객 INNER JOIN 주문

ON 고객.고객ID = 주문.고객ID

GROUP BY 1;

① 제품 테이블에 새로운 제품 ('태블릿', 450000, 1, 25)을 삽입하는 명령을 쓰시오.

insert into 제품 (제품명, 가격, 카테고리ID, 재고수량) values ('태블릿', 450000, 1, 25);

② 제품 테이블에서 제품명이 '마우스'인 제품의 가격을 30000으로 수정하는 명령을 쓰시오.

update 제품 set 가격 = 30000 where 제품명 = '마우스';

③제품 테이블에서 재고 수량이 10개 미만인 제품의 제품명과 재고 수량을 조회하는 명령을 쓰시오.

SELECT 제품명, 재고수량

FROM test.제품

WHERE 재고수량 <= 10;

④ 제품 테이블에서 카테고리 ID가 NULL인 제품을 삭제하는 명령을 쓰시오.

DELETE FROM 제품 WHERE 카테고리ID IS NULL;

⑤ 카테고리 테이블에 새로운 카테고리 ('사무용품')을 삽입하는 명령을 쓰시오.

INSERT INTO 카테고리 (카테고리명) VALUES ('사무용품');

⑥ 카테고리 테이블에서 카테고리명이 '도서'인 레코드를 삭제하는 명령을 쓰시오

DELETE FROM 카테고리 WHERE 카테고리명 = '도서';

JOIN 문제:

⑦ 제품 테이블과 카테고리 테이블을 조인하여 제품명과 해당 제품의 카테고리명을 함께 조회하세요.

SELECT 제품.제품명, 카테고리.카테고리명

FROM 제품 INNER JOIN 카테고리

ON 제품.카테고리ID = 카테고리.카테고리ID;

⑧ 가격이 10만원 이상인 제품의 제품명과 카테고리명을 조회하세요.

SELECT 제품.제품명, 카테고리.카테고리명

FROM 제품 INNER JOIN 카테고리

ON 제품.카테고리ID = 카테고리.카테고리ID

WHERE 가격 >= 100000;

⑨ 카테고리명이 '컴퓨터 주변기기'인 모든 제품의 제품명과 가격을 조회하세요.

SELECT 제품.제품명, 카테고리.카테고리명

FROM 제품 INNER JOIN 카테고리

ON 제품.카테고리ID = 카테고리.카테고리ID

WHERE 카테고리명 = '컴퓨터 주변기기';

10.각 카테고리별로 포함된 제품의 수를 조회하고, 카테고리명과 제품 수를 함께 출력하세요.

SELECT 카테고리.카테고리명, COUNT(제품.재고수량) AS 제품수

FROM 제품 INNER JOIN 카테고리

group by 1;

서브 쿼리

SQL 서브 쿼리는 메인 쿼리 안에 또 다른 쿼리(서브 쿼리)를 사용하여 데이터를 추출하거나 처리

데이터 필터링

집계 및 계산

비교 및 검증

중첩된 데이터 추출

EXISTS: 특정 조건을 만족하는 레코드 존재 여부

[6-1] 최고 마일리지를 보유한 고객 정보 보기

SUM(), COUNT()와 같은 집계함수는 WHERE에 직접 사용할 수 없어서  
서브 쿼리를 통해 결과를 받아와야함.

SELECT 고객번호, 고객회사명, 담당자명, 마일리지

FROM 고객

WHERE 마일리지 = (select MAX(마일리지) FROM 고객);

SELECT 고객번호, 고객회사명, 담당자명, 마일리지

FROM 고객

order by 마일리지 DESC LIMIT 1;

주문번호 H0250을 주문한 고객에 대해 고객 회사명, 담당자명 INNER JOIN과 서브 쿼리를 구해보기

SELECT 고객회사명, 담당자명

FROM 고객 INNER JOIN 주문

ON 고객.고객번호 = 주문.고객번호

WHERE 주문.주문번호 = 'H0250';

SELECT 고객회사명, 담당자명

FROM 고객

WHERE 고객번호 = (SELECT 고객번호 FROM 주문 WHERE 주문번호 = 'H0250');-

[6-3] 부산광역시에서 고객의 최소 마일리지보다 더 큰 마일리지를 가진 고객 정보 보기

SELECT 담당자명, 고객회사명, 마일리지

FROM 고객

WHERE 마일리지 > (SELECT MIN(마일리지) FROM 고객 WHERE 도시 = '부산광역시');

복수행 서브 쿼리

서브 쿼리의 결과가 여러 행이 나오는 쿼리

(IN, ALL, ANY, SOME, EXISTS)와 같은 복수행 비교연산자를 사용하여, 메인 쿼리와 서브 쿼리를 연결.

IN : 서브 쿼리 결과에 해당 값이 포함되어 있는지

ALL: 서브 쿼리의 모든 결과 값과 비교해서 조건을 만족하는지

ANY/SOME: 서브 쿼리의 결과 값이 어느 하나라도 조건을 만족하는지

EXISTS: 서브 쿼리의 결과가 존재하는지, 보통 WHERE 내부에서 존재여부 확인.

‘부산광역시’ 고객이 주문한 주문건수 보기

SELECT COUNT(\*) AS 주문건수

FROM 주문

WHERE 고객번호 IN (SELECT 고객번호 FROM 고객 WHERE 도시 = '부산광역시');

[6-5] 부산광역시 전체 고객의 마일리지보다 많은 마일리지를 가진 고객의 정보



SELECT 담당자명, 고객회사명, 마일리지

FROM 고객

WHERE 마일리지 > ANY(SELECT 마일리지 FROM 고객 WHERE 도시 = '부산광역시');

ANY와 ALL의 차이

ANY) 비교의 조건 중 하나라도 맞으면 TRUE

ALL) 비교의 모든 조건이 맞아야 TRUE

[6-6] 각 지역의 평균 마일리지 이상의 마일리지를 가진 고객의 정보 보기

SELECT 담당자명, 고객회사명, 마일리지

FROM 고객

WHERE 마일리지 > ALL(SELECT AVG(마일리지) FROM 고객 GROUP BY 지역);

모든 지역 평균 마일리지보다 높은 고객을 찾기 위해 GROUP BY 사용

[6-7] 한번이리도 주문한 적이 있는 고객의 정보 보기

SELECT 고객번호, 고객회사명

FROM 고객

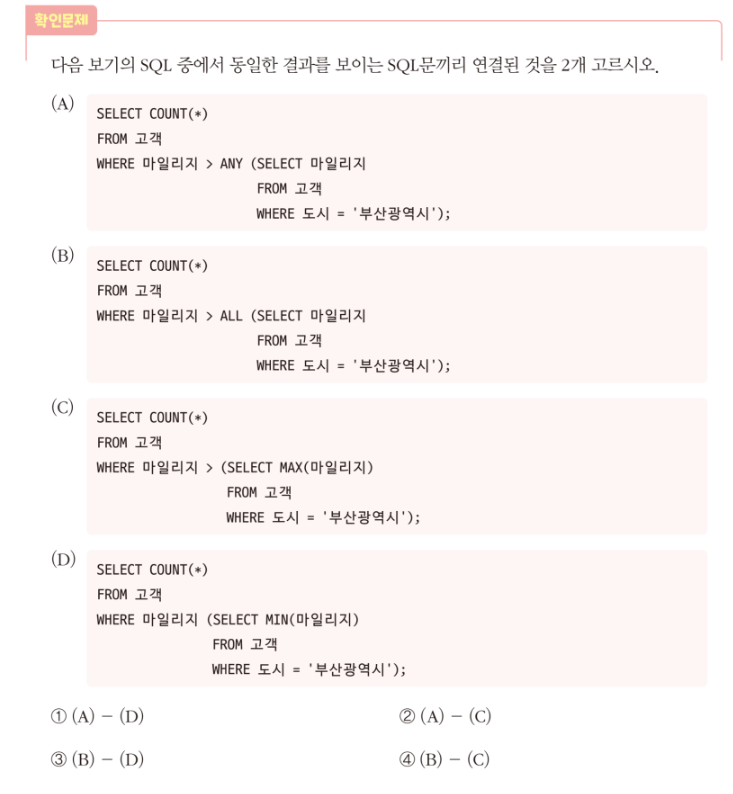
WHERE EXISTS (SELECT \* FROM 주문 WHERE 고객번호 = 고객.고객번호);

SELECT DISTINCT 고객.고객번호, 고객.고객회사명

FROM 고객 INNER JOIN 주문

ON 고객.고객번호 = 주문.고객번호;

EXISTS 쿼리는 INNER JOIN + DISTINCT로 동일한 결과를 얻을 수 있음



1, 4

HAVING: GROUP BY 이후 조건을 걺, 집계 값 필터링

WHERE: GROUP BY 전 조건을 걺, 일반 행 필터링

[6-8] 고객 전체의 평균 마일리지보다 평균 마일리지가 큰 도시의 도시명과 평균 마일리지

SELECT 도시, AVG(마일리지) AS 평균마일리지

FROM 고객

GROUP BY 1

HAVING AVG(마일리지) > (SELECT AVG(마일리지) FROM 고객);

AVG(마일리지) : 도시가 GROUP BY로 묶여 있어 도시 별 마일리지 평균,

서브 쿼리 내 AVG(마일리지) : 전체 마일리지 평균

FROM에서 사용하는 서브 쿼리

FROM에서도 서브 쿼리를 사용할 수 있다. = 인라인 뷰  
인라인 뷰에는 반드시 별명을 지정해줘야 하고, 이는 테이블명처럼 사용 가능

[6-9] 담당자명, 고객회사명, 마일리지, 도시, 해당 도시 평균 마일리지  
그리고 고객이 위치하는 도시의 평균 마일리지와 고객 마일리지의 차이 보기

SELECT 담당자명, 고객회사명, 마일리지, 고객.도시, 도시\_평균마일리지, 도시\_평균마일리지-마일리지 AS 차이

FROM 고객

,(select 도시, AVG(마일리지) AS 도시\_평균마일리지 FROM 고객 group by 도시) AS 도시별요약

WHERE 고객.도시 = 도시별요약.도시;

SELECT 담당자명, 고객회사명, 마일리지, 고객.도시, 도시\_평균마일리지, 도시\_평균마일리지-마일리지 AS 차이

FROM 고객

INNER JOIN

(select 도시, AVG(마일리지) AS 도시\_평균마일리지 FROM 고객 group by 도시) AS 도시별요약

ON 고객.도시 = 도시별요약.도시;

SELECT에서 사용하는 서브 쿼리 = 스칼라 서브 쿼리

정확히 한 개의 행에 대해 한 개의 값을 반환.

[6-10] 고객번호, 담당자명, 고객의 최종 주문일 보기

SELECT 고객번호, 담당자명, (SELECT MAX(주문일) FROM 주문 WHERE 주문.고객번호 = 고객.고객번호) AS 최종주문일

FROM 고객;

SELECT 고객.고객번호, 고객.담당자명, MAX(주문.주문일) AS 최종주문일

FROM 고객 LEFT JOIN 주문

ON 고객.고객번호 = 주문.고객번호

GROUP BY 1, 2;

CTE는 뷰와 마찬가지로 파생 테이블처럼 사용 가능.

WITH [CTE] AS (

SELECT ~~~

)

재활용, 간결함, 가독성

SELECT 담당자명, 고객회사명, 마일리지, 고객.도시, 도시\_평균마일리지, 도시\_평균마일리지-마일리지 AS 차이

FROM 고객

,(select 도시, AVG(마일리지) AS 도시\_평균마일리지 FROM 고객 group by 도시) AS 도시별요약

WHERE 고객.도시 = 도시별요약.도시;

=>

WITH 도시별요약 AS (

SELECT 도시, AVG(마일리지) AS 도시\_평균마일리지

FROM 고객

GROUP BY 도시

)

SELECT 담당자명, 고객회사명, 마일리지, 고객.도시, 도시별요약.도시\_평균마일리지, 도시별요약.도시\_평균마일리지 - 마일리지 AS 차이

FROM 고객, 도시별요약

WHERE 고객.도시 = 도시별요약.도시;

상관 서브 쿼리

메인 쿼리와 서브 쿼리 간의 상관관계를 포함하는 형태의 쿼리

사원 테이블에서 사원번호, 사원이름, 상사번호, 상사이름 보기

SELECT 사원번호, 이름, 상사번호, (SELECT 이름 FROM 사원 AS 상사 WHERE 상사.사원번호 = 사원.상사번호) AS 상사이름

FROM 사원;

SELF JOIN: 테이블이 자기 자신과 JOIN하는 경우

(사원 테이블이 자기 자신과 JOIN할 때, 상사라는 별명을 줌,  
LEFT JOIN으로 상사번호가 없는 사원도 함께 표시,  
ON 조건을 통해 상사의 이름을 가져오기)

SELECT 사원.사원번호, 사원.이름, 사원.상사번호, 상사.이름 AS 상사이름

FROM 사원 LEFT JOIN 사원 AS 상사

ON 사원.상사번호 = 상사.사원번호;

다중 COLUMN 서브 쿼리

여러 개의 COLUMN을 사용하여 다중 비교

각 도시마다 최고 마일리지를 보유한 고객의 정보 보기

SELECT 도시, 담당자명, 고객회사명, 마일리지

FROM 고객

WHERE (도시, 마일리지) IN (SELECT 도시, MAX(마일리지) FROM 고객 group by 도시);