Programmers – SQL

Lv. 1

1. 모든 레코드 조회하기

```
select *
from ANIMAL_INS
order by ANIMAL_ID
```

2. 역순 정렬하기

```
select NAME, DATETIME
from ANIMAL_INS
order by ANIMAL_ID   desc
```

3. 아픈 동물 찾기

```
select ANIMAL_ID, Name
from ANIMAL_INS
where INTAKE_CONDITION = 'Sick'
```

4. 어린 동물 찾기

```
select ANIMAL_ID, name
from ANIMAL_INS
where INTAKE_CONDITION != 'Aged'
order by ANIMAL_ID
```

5. 이름이 없는 동물의 아이디

```
SELECT ANIMAL_ID
from ANIMAL_INS
where NAME is Null
order by ANIMAL_ID asc
```

6. 동물의 아이디와 이름

```
select ANIMAL_ID, Name
from ANIMAL_INS
order by ANIMAL_ID
```

7. 여러 기준으로 정렬하기

```
select ANIMAL_ID, Name, DATETIME
from ANIMAL_INS
order by Name, DATETIME desc
```

8. 상위 n개 레코드

```
select name
from ANIMAL_INS
order by DATETIME
limit 1
```

9. 이름이 있는 동물의 아이디

```
SELECT ANIMAL_ID
from ANIMAL_INS
where Name is not null
```

10. 최댓값 구하기

```
select max(DATETIME)
from ANIMAL_INS
```

11. 강원도에 위치한 생산공장 목록 출력하기

```
SELECT FACTORY_ID, FACTORY_name, Address
from FOOD_FACTORY
where ADDRESS like '강원도%'
order by FACTORY_ID asc
```

12. 경기도에 위치한 식품창고 목록 출력하기

```
SELECT WAREHOUSE_ID, WAREHOUSE_NAME, ADDRESS, ifnull(FREEZER_YN, 'N') as Freezer
from FOOD_WAREHOUSE
where ADDRESS like '경기도%'
order by WAREHOUSE_ID
```

13. 나이 정보가 없는 회원 수 구하기

```
SELECT count(*) as users
from USER_INFO
where age is null
```

14. 조건에 맞는 회원수 구하기

```
select count(*) as users_cnt
from USER_INFO
where year(JOINED) = 2021 and age >= 20 and age <= 29
```

15. 가장 비싼 상품 구하기

```
select max(PRICE) as MAX_PRICE
from PRODUCT
```

16. 12세 이하인 여자 환자 목록 출력하기

```
select PT_NAME, PT_NO, GEND_CD, AGE, ifnull(TLNO, 'NONE')
from PATIENT
where AGE <= 12 and GEND_CD = 'W'
order by AGE desc, PT_NAME
```

17. 흉부외과 또는 일반외과 의사 목록 출력하기

```
SELECT DR_NAME, DR_ID, MCDP_CD, TO_CHAR(HIRE_YMD, '%Y-%m-%d') as Hire_YMD
FROM DOCTOR
WHERE MCDP_CD in('CS', 'GS')
ORDER BY HIRE_YMD DESC, DR_NAME;
```

18. 인기있는 아이스크림

```
SELECT FLAVOR
from FIRST_HALF
order by TOTAL_ORDER desc, SHIPMENT_ID
```

19. 과일로 만든 아이스크림 고르기

```
SELECT a.flavor
from FIRST_HALF a left join ICECREAM_INFO b on a.flavor = b.flavor
where a.TOTAL_ORDER > 3000 and b.INGREDIENT_TYPE = 'fruit_based'
order by a.TOTAL_ORDER desc
```

Lv. 2

1. 최솟값 구하기

```
select min(DATETIME)
from ANIMAL_INS
```

2. 고양이와 개는 몇 마리 있을까

```
select ANIMAL_TYPE, count(*) as Cat
from ANIMAL_INS
where ANIMAL_TYPE = 'Cat'

union all

select ANIMAL_TYPE, count(*) as Dog
from ANIMAL_INS
where ANIMAL_TYPE = 'Dog'
```

3. 동명 동물 수 찾기

```
select name, count(name) as cnt
from ANIMAL_INS
where name is not null
group by name
having count(name) > 1
order by name
```

4. 루시와 엘라 찾기

```
SELECT ANIMAL_ID, name, SEX_UPON_INTAKE
from ANIMAL_INS
where name in ('Lucy', 'Ella', 'Pickle', 'Rogan', 'Sabrina', 'Mitty')
```

5. 이름에 el이 들어가는 동물 찾기

```
SELECT ANIMAL_ID, name
from ANIMAL_INS
where ANIMAL_TYPE = 'Dog' and name like '%EL%'
order by name
```

6. 동물 수 구하기

```
select count(ANIMAL_ID) as cnt
from ANIMAL_INS
```

7. 중복 제거하기

```
select count(distinct name) as t_name
from ANIMAL_INS
where name is not null
```

8. 중성화 여부 파악하기

```
SELECT ANIMAL_ID, NAME,
    case when (SEX_UPON_INTAKE like '%Neutered%' or SEX_UPON_INTAKE like '%Spayed%') then 'O'
    else 'X'
    end as '중성화'
from ANIMAL_INS
order by ANIMAL_ID
```

9. NULL 처리하기

```
SELECT ANIMAL_TYPE, ifnull(NAME, 'No name'), SEX_UPON_INTAKE
from ANIMAL_INS
order by ANIMAL_ID
```

10. 입양 시각 구하기(1)

```
select hour(DATETIME) as Hour, count(ANIMAL_ID) as cnt
from ANIMAL_OUTS
where hour(DATETIME) between 9 and 19
group by Hour
order by Hour
```

11. DATETIME에서 DATE로 형 변환

```
SELECT ANIMAL_ID, NAME, date_format(DATETIME, '%Y-%m-%d') as 'date'
from ANIMAL_INS
order by ANIMAL_ID
```

12. 가격이 제일 비싼 식품의 정보 출력하기

```
SELECT PRODUCT_ID, PRODUCT_NAME, PRODUCT_CD, CATEGORY, PRICE
from FOOD_PRODUCT
order by PRICE desc
limit 1
```

13. 3월에 태어난 여성 회원 목록 출력하기

```
SELECT MEMBER_ID, MEMBER_NAME, GENDER, date_format(DATE_OF_BIRTH, '%Y-%m-%d') as Birth
from MEMBER_PROFILE
where month(DATE_OF_BIRTH) = 3 and GENDER = 'W' and TLNO is not null
order by MEMBER_ID
```

14. 카테고리 별 상품 개수 구하기

```
SELECT left(PRODUCT_CODE, 2) as Category, count(PRODUCT_ID) as cnt
from PRODUCT
group by Category
order by Category
```

15. 가격대 별 상품 개수 구하기

```
select (case when price between 0 and 9999 then '0'
            when price between 10000 and 19999 then '10000'
            when price between 20000 and 29999 then '20000'
            when price between 30000 and 39999 then '30000'
            when price between 40000 and 49999 then '40000'
            when price between 50000 and 59999 then '50000'
            when price between 60000 and 69999 then '60000'
            when price between 70000 and 79999 then '70000'
```

```
when price between 80000 and 89999 then '80000' end) as Price_group, count(*) as cnt
from PRODUCT
group by Price_group
order by Price_group
```

16. 상품 별 오프라인 매출 구하기

```
SELECT a.PRODUCT_CODE, sum(a.PRICE * b.SALES_AMOUNT) as sales
from product a join OFFLINE_SALE b on a.PRODUCT_ID = b.PRODUCT_ID
group by a.PRODUCT_CODE
order by sales desc, PRODUCT_CODE
```

17. 재구매가 일어난 상품과 회원 리스트 구하기

```
select USER_ID, PRODUCT_ID
from ONLINE_SALE
group by USER_ID, PRODUCT_ID
having count(*) > 1
order by USER_ID, PRODUCT_ID desc
```

18. 진료과별 총 예약 횟수 출력하기

```
SELECT MCDP_CD, count(*) as cnt
from APPOINTMENT
where month(APNT_YMD) = 5
group by MCDP_CD
order by cnt, MCDP_CD
```

19. 성분으로 구분한 아이스크림 총 주문량

```
select B.INGREDIENT_TYPE, sum(A.TOTAL_ORDER) as TOTAL_ORDER
from FIRST_HALF A join ICECREAM_INFO B on A.FLAVOR = B.FLAVOR
group by B.INGREDIENT_TYPE
order by TOTAL_ORDER
```

Lv. 3

1. 없어진 기록 찾기

입양을 간 기록은 있는데, 보호소에 들어온 기록이 없는 동물의 ID와 이름을 ID 순으로 조회하는 SQL문을 작성해주세요.

```
select B.ANIMAL_ID, B.Name
from ANIMAL_OUTS  B left join ANIMAL_INS A on A.ANIMAL_ID = B.ANIMAL_ID
where A.ANIMAL_ID is null
order by ANIMAL_ID
```

2. 있었는데요 없었습니다

보호 시작일보다 입양일이 더 빠른 동물의 아이디와 이름을 조회하는 SQL문을 작성해주세요.

```
SELECT A.ANIMAL_ID, A.name
from ANIMAL_INS A left join ANIMAL_OUTS B on A.ANIMAL_ID = B.ANIMAL_ID
where A.DATETIME > B.DATETIME
order by A.DATETIME
```

3. 오랜 기간 보호한 동물(1)

아직 입양을 못 간 동물 중, 가장 오래 보호소에 있었던 동물 3마리의 이름과 보호 시작일을 조회하는 SQL문을 작성해주세요.

```
SELECT A.name, A.DATETIME
from ANIMAL_INS A left join ANIMAL_OUTS B on A.ANIMAL_ID = B.ANIMAL_ID
where B.ANIMAL_ID is null
order by A.DATETIME
limit 3
```

4. 오랜 기간 보호한 동물(2)

입양을 간 동물 중, 보호 기간이 가장 길었던 동물 두 마리의 아이디와 이름을 조회하는 SQL문을 작성해주세요.

```
SELECT B.ANIMAL_ID, B.NAME
from ANIMAL_INS A inner join ANIMAL_OUTS B
                    on A.ANIMAL_ID = B.ANIMAL_ID
order by   datediff(B.DATETIME, A.DATETIME) desc
limit 2
##########################################
SELECT B.ANIMAL_ID, B.NAME
from ANIMAL_INS A join ANIMAL_OUTS B
                    on A.ANIMAL_ID = B.ANIMAL_ID
order by   datediff(B.DATETIME, A.DATETIME) desc
limit 2
```

5. 헤비 유저가 소유한 장소

공간을 둘 이상 등록한 사람을 "헤비 유저"라고 부릅니다. 헤비 유저가 등록한 공간의 정보를 아이디 순으로 조회하는 SQL문을 작성해주세요.

```
SELECT *
FROM PLACES
WHERE HOST_ID IN(
    SELECT HOST_ID
    FROM PLACES
    GROUP BY HOST_ID
    HAVING COUNT(ID) >= 2
)
ORDER BY ID
```

6. 조건별로 분류하여 주문상태 출력하기

FOOD_ORDER 테이블에서 5월 1일을 기준으로 주문 ID, 제품 ID, 출고일자, 출고여부를 조회하는 SQL문을 작성해주세요. 출고여부는 5월 1일까지 출고완료로 이후 날짜는 출고 대기로 미정이면 출고미정으로 출력해주시고, 결과는 주문 ID를 기준으로 오름차순 정렬해주세요.

```
select ORDER_ID, PRODUCT_ID, date_format(OUT_DATE, '%Y-%m-%d') as out_date, (
    case when datediff(out_date, '2022-05-01') <= 0 then '출고완료'
        when datediff(out_date, '2022-05-01') > 0 then '출고대기'
```

```
        else '출고미정' end
) as 출고여부
from FOOD_ORDER
order by ORDER_ID
```

7. 즐겨찾기가 가장 많은 식당 정보 출력하기

REST_INFO 테이블에서 음식종류별로 즐겨찾기수가 가장 많은 식당의 음식 종류, ID, 식당 이름, 즐겨찾기수를 조회하는 SQL문을 작성해주세요.

```
select A.FOOD_TYPE, A.REST_ID, A.REST_NAME, A.FAVORITES
from REST_INFO A join(
    select FOOD_TYPE, max(FAVORITES) as FAVORITES
    from REST_INFO
    group by FOOD_TYPE) B
    on A.FOOD_TYPE = B.FOOD_TYPE
        and A.FAVORITES = B.FAVORITES
order by A.FOOD_TYPE desc
```

※ *Samchully Test 01*

CART_PRODUCTS 테이블은 장바구니에 담긴 상품 정보를 담은 테이블입니다. CART_PRODUCTS 테이블의 구조는 다음과 같으며, ID, CART_ID, NAME, PRICE는 각각 테이블의 아이디, 장바구니의 아이디, 상품 종류, 가격을 나타냅니다.

장바구니에 담긴 상품 가격 합이 5만 미만이면 배송료로 3000원을 더 내야 합니다. 각 장바구니의 아이디와 결제 금액은 얼마인지 조회하는 SQL 문을 작성해주세요. 결과는 장바구니의 아이디 순으로 나와야 합니다.

```
SELECT CART_ID,
(
CASE WHEN SUM (PRICE) < 50000 THEN SUM(PRICE)+3000
ELSE SUM(PRICE)
END
)
as
'결제액'
FROM CART_PRODUCTS
GROUP BY CART_ID
```

Lv. 4,5

1. 보호소에서 중성화한 동물

보호소에 들어올 당시에는 중성화1되지 않았지만, 보호소를 나갈 당시에는 중성화된 동물의 아이디와 생물종, 이름을 조회하는 아이디 순으로 조회하는 SQL 문을 작성해주세요.

```
select A.ANIMAL_ID, A.ANIMAL_TYPE, A.Name
from ANIMAL_INS A join ANIMAL_OUTS B on A.ANIMAL_ID = B.ANIMAL_ID
where A.SEX_UPON_INTAKE like "%Intact%"
    and (B.SEX_UPON_OUTCOME like "%Spayed%" or B.SEX_UPON_OUTCOME like "%Neutered%")
order by ANIMAL_ID
```

2. 입양 시각 구하기(2)

0시부터 23시까지, 각 시간대별로 입양이 몇 건이나 발생했는지 조회하는 SQL문을 작성해주세요.

```
SET @HOUR = -1;

SELECT (@HOUR := @HOUR +1) AS HOUR,
    (SELECT COUNT(HOUR(DATETIME))
    FROM ANIMAL_OUTS
    WHERE HOUR(DATETIME)=@HOUR) AS COUNT
FROM ANIMAL_OUTS
WHERE @HOUR < 23;
```

3. 우유와 요거트가 담긴 장바구니

우유와 요거트를 동시에 구입한 장바구니의 아이디를 조회하는 SQL 문을 작성해주세요.

```
SELECT CART_ID
FROM CART_PRODUCTS
WHERE NAME IN ('Milk','Yogurt')
GROUP BY CART_ID
HAVING COUNT(DISTINCT NAME) = 2
```

4. 식품분류별 가장 비싼 식품의 정보 조회하기

식품분류별로 가격이 제일 비싼 식품의 분류, 가격, 이름을 조회하는 SQL문을 작성해주세요. 이때 식품분류가 '과자', '국', '김치', '식용유'인 경우만 출력

```
select CATEGORY, PRICE as max_price, PRODUCT_NAME
from FOOD_PRODUCT
where PRICE in (
    select max(PRICE)
    from FOOD_PRODUCT
    group by CATEGORY
)
    and
    CATEGORY in ('과자', '국', '김치', '식용유')
order by max_price desc
```

5. 5월 식품들의 총매출 조회하기

FOOD_PRODUCT와 FOOD_ORDER 테이블에서 생산일자가 2022년 5월인 식품들의 식품 ID, 식품 이름, 총 매출을 조회하는 SQL문을 작성해주세요.

```
SELECT A.PRODUCT_ID, B.PRODUCT_NAME, (SUM(A.AMOUNT) * B.PRICE) AS TOTAL_SALES
FROM FOOD_ORDER A
JOIN FOOD_PRODUCT B ON A.PRODUCT_ID = B.PRODUCT_ID
WHERE YEAR(PRODUCE_DATE) = 2022 AND MONTH(PRODUCE_DATE) = 5
GROUP BY A.PRODUCT_ID
ORDER BY TOTAL_SALES DESC, A.PRODUCT_ID
```

6. 서울에 위치한 식당 목록 출력하기

REST_INFO와 REST_REVIEW 테이블에서 서울에 위치한 식당들의 식당 ID, 식당 이름, 음식 종류, 즐겨찾기수, 주소, 리뷰 평균 점수를 조회하는 SQL문을 작성해주세요. 이때 리뷰 평균점수는 소수점 세 번째 자리에서

반올림 해주시고 결과는 평균점수를 기준으로 내림차순 정렬해주시고, 평균점수가 같다면 즐겨찾기수를 기준으로 내림차순 정렬해주세요.

```
SELECT A.REST_ID, B.REST_NAME, B.FOOD_TYPE, B.FAVORITES, B.ADDRESS, ROUND(AVG(A.REVIEW_SCORE),2) AS SCORE
FROM REST_REVIEW A
JOIN REST_INFO B ON A.REST_ID = B.REST_ID
GROUP BY A.REST_ID
HAVING B.ADDRESS LIKE '서울%'
ORDER BY SCORE DESC, B.FAVORITES DESC
```

7. 그룹별 조건에 맞는 식당 목록 출력하기

MEMBER_PROFILE와 REST_REVIEW 테이블에서 리뷰를 가장 많이 작성한 회원의 리뷰들을 조회하는 SQL문을 작성해주세요. 회원 이름, 리뷰 텍스트, 리뷰 작성일이 출력되도록 작성해주시고, 결과는 리뷰 작성일을 기준으로 오름차순, 리뷰 작성일이 같다면 리뷰 텍스트를 기준으로 오름차순 정렬해주세요.

```
SELECT A.MEMBER_NAME,
    B.REVIEW_TEXT,
    DATE_FORMAT(B.REVIEW_DATE, '%Y-%m-%d') AS REVIEW_DATE
FROM MEMBER_PROFILE A
    JOIN REST_REVIEW B ON A.MEMBER_ID = B.MEMBER_ID
WHERE A.MEMBER_ID IN (
    SELECT MEMBER_ID
    FROM REST_REVIEW
    GROUP BY MEMBER_ID
    HAVING count(member_id) = (
        SELECT COUNT(REVIEW_TEXT) AS 'NUM' FROM REST_REVIEW
        GROUP BY MEMBER_ID
        ORDER BY NUM DESC
        LIMIT 1)
)
ORDER BY B.REVIEW_DATE, B.REVIEW_TEXT
```

8. 년, 월, 성별 별 상품 구매 회원 수 구하기

USER_INFO 테이블과 ONLINE_SALE 테이블에서 년, 월, 성별 별로 상품을 구매한 회원수를 집계하는 SQL문을 작성해주세요.

```
SELECT YEAR(B.SALES_DATE) AS YEAR,
        MONTH(B.SALES_DATE) AS MONTH,
    A.GENDER AS GENDER,
    COUNT(DISTINCT A.USER_ID) AS USERS
FROM USER_INFO A JOIN ONLINE_SALE B ON A.USER_ID = B.USER_ID
GROUP BY YEAR, MONTH, GENDER
HAVING GENDER IS NOT NULL
ORDER BY YEAR, MONTH, GENDER
```

9. 오프라인/온라인 판매 데이터 통합하기

ONLINE_SALE 테이블과 OFFLINE_SALE 테이블에서 2022년 3월의 오프라인/온라인 상품 판매 데이터의 판매 날짜, 상품ID, 유저ID, 판매량을 출력하는 SQL문

을 작성해주세요.

```
select date_format(SALES_DATE, '%Y-%m-%d') as Date,
PRODUCT_ID, USER_ID, SALES_AMOUNT
from ONLINE_SALE
where month(SALES_DATE) = 3

union all

select date_format(SALES_DATE, '%Y-%m-%d') as Date,
PRODUCT_ID, null as USER_ID, SALES_AMOUNT
from OFFLINE_SALE
where month(SALES_DATE) = 3

order by Date, PRODUCT_ID, USER_ID
```

10. 취소되지 않은 진료 예약 조회하기

PATIENT, DOCTOR 그리고 APPOINTMENT 테이블에서 2022년 4월 13일 취소되지 않은 흉부외과(CS) 진료 예약 내역을 조회하는 SQL문을 작성해주세요.

```
select  c.APNT_NO,
        a.PT_NAME,
        c.PT_NO,
        c.MCDP_CD,
        b.DR_NAME,
        c.APNT_YMD
from APPOINTMENT c
    join (
        select PT_NAME, PT_NO
        from PATIENT) a
    join (
        select MCDP_CD, DR_NAME, DR_ID
        from DOCTOR ) b
    on a.PT_NO = c.PT_NO and b.DR_ID = c.MDDR_ID
where c.MCDP_CD = 'CS'
    and month(c.APNT_YMD) = '04'
    and c.APNT_CNCL_YN = 'N'
group by PT_NAME
order by c.APNT_YMD
```

11. 주문량이 많은 아이스크림들 조회하기

7월 아이스크림 총 주문량과 상반기의 아이스크림 총 주문량을 더한 값이 큰 순서대로 상위 3개의 맛을 조회하는 SQL 문을 작성해주세요.

```
SELECT F.FLAVOR
FROM FIRST_HALF F JOIN (
    SELECT FLAVOR, SUM(TOTAL_ORDER) AS
TOTAL_ORDER
    FROM JULY
    GROUP BY FLAVOR) J
    ON F.FLAVOR = J.FLAVOR
ORDER BY (F.TOTAL_ORDER + J.TOTAL_ORDER) DESC
LIMIT 3
```

12. Lv.5 상품을 구매한 회원 비율 구하기

USER_INFO 테이블과 ONLINE_SALE 테이블에서 2021년에 가입한 전체 회원들 중 상품을 구매한 회원수와 상품을 구매한 회원의 비율(=2021년에 가입한 회원 중 상품을 구매한 회원수 / 2021년에 가입한 전체 회원 수)을 년, 월 별로 출력하는 SQL문을 작성해주세요. 상품을 구매한 회원의 비율은 소수점 두번째자리에서 반올림하고, 전체 결과는 년을 기준으로 오름차순 정렬해주시고 년이 같다면 월을 기준으로 오름차순 정렬해주세요.

```
SELECT  YEAR, MONTH,
        COUNT(*) AS PUCHASED_USERS,
            ROUND((COUNT(*)/ (SELECT COUNT(*)

FROM USER_INFO
                        WHERE  YEAR(JOINED)  =
2021)), 1) AS PUCHASED_RATIO
FROM (
    SELECT DISTINCT YEAR(S.SALES_DATE) AS YEAR,
                    MONTH(S.SALES_DATE)      AS
MONTH, U.USER_ID
    FROM ONLINE_SALE S
    JOIN USER_INFO U
        ON S.USER_ID = U.USER_ID AND YEAR(JOINED)
= 2021
    ) A
GROUP BY YEAR, MONTH
ORDER BY YEAR, MONTH
```