2일차(SQL) | 생각하기(T) : SQL문 연습하기

질의 내용	아래 순서대로 복사하여 붙여넣기 하면 됩니다.

```
-- Drop the tables if they exist
DROP TABLE IF EXISTS t_sales;
DROP TABLE IF EXISTS t_customer;
DROP TABLE IF EXISTS t_product;
DROP TABLE IF EXISTS t_region;

-- Create the t_region table
CREATE TABLE t_region (
    region_code varchar(3) not null,
    region_name varchar(10) not null,
    primary key(region_code)
);

-- Create the t_customer table
CREATE TABLE t_customer (
     id int not null auto_increment,
     customer_name varchar(10) not null,
     phone varchar(20) not null unique,
     email varchar(50) not null unique,
     address varchar(100) not null,
     regist_date datetime default now(),
     region_code varchar(3) not null,
     primary key(id)
);

-- Create the t_product table
CREATE TABLE t_product(
  id int not null auto_increment,
  product_code varchar(12) not null unique,
  product_name varchar(50) not null,
  price int,
  primary key(id)
);
```

```
-- Create the t_sales table
CREATE TABLE t_sales (
    id int not null auto_increment,
    customer_id int not null,
    product_code varchar(12) not null,
    qty int not null,
    sales_date datetime default now(),
    primary key(id)
);
-- Add foreign key constraint to t_customer table
ALTER TABLE t_customer
ADD CONSTRAINT fk_region_code FOREIGN KEY (region_code)
REFERENCES t_region(region_code);

ALTER TABLE t_sales
ADD CONSTRAINT fk_customer_id FOREIGN KEY (customer_id) REFERENCES t_customer (id),
ADD CONSTRAINT fk_product_code FOREIGN KEY (product_code) REFERENCES t_product (product_code);

-- t_region 테이블에 데이터 추가
INSERT INTO t_region (region_code, region_name) VALUES
('02', '서울특별시'),
('031', '경기도'),
('032', '인천광역시'),
('033', '강원특별자치도'),
('041', '충청남도'),
('042', '대전광역시'),
('043', '충청북도'),
('044', '세종특별자치시'),
('051', '부산광역시'),
('052', '울산광역시'),
('053', '대구광역시'),
('054', '경상북도'),
('055', '경상남도'),
('061', '전라남도'),
```

```
('062', '광주광역시'),
('063', '전라북도'),
('064', '제주특별자치도');

-- t_customer 테이블에 데이터 추가
INSERT INTO t_customer (customer_name, phone, email, address, region_code)
VALUES
('홍길동', '010-1234-5678', 'hong@example.com', '서울시 강남구', '02'),
('김철수', '010-9876-5432', 'kim@example.com', '경기도 수원시', '031'),
('이영희', '010-1111-2222', 'lee@example.com', '인천시 남구', '032'),
('박민지', '010-5555-7777', 'park@example.com', '강원도 춘천시', '033'),
('정기호', '010-9999-8888', 'jung@example.com', '대전시 중구', '042');

-- t_product 테이블에 데이터 추가
INSERT INTO t_product (product_code, product_name, price)
VALUES
('P001', '노트북', 1500000),
('P002', '스마트폰', 1000000),
('P003', '키보드', 50000),
('P004', '마우스', 30000),
('P005', '이어폰', 70000);

-- t_sales 테이블에 데이터 추가
INSERT INTO t_sales (customer_id, product_code, qty)
VALUES
(1, 'P001', 2),
(2, 'P002', 1),
(3, 'P003', 5),
(4, 'P004', 3),
(5, 'P005', 2),
(1, 'P002', 3),
(3, 'P001', 1),
(2, 'P004', 2),
(4, 'P003', 4),
(5, 'P005', 1);
```

id	customer_name	phone	email	address	regist_date	region_code
1	홍길동	010-1234-5678	hong@example.com	서울시 강남구	2023-07-07 13:42:24	02
2	김철수	010-9876-5432	kim@example.com	경기도 수원시	2023-07-07 13:42:24	031
3	이영희	010-1111-2222	lee@example.com	인천시 남구	2023-07-07 13:42:24	032
4	박민지	010-5555-7777	park@example.com	강원도 춘천시	2023-07-07 13:42:24	033
5	정기호	010-9999-8888	jung@example.com	대전시 중구	2023-07-07 13:42:24	042

id	product_code	product_name	price
1	P001	노트북	1500000
2	P002	스마트폰	1000000
3	P003	키보드	50000
4	P004	마우스	30000
5	P005	이어폰	70000

id	customer_id	product_code	qty	sales_date
1	1	P001	2	2023-07-07 13:42:24
2	2	P002	1	2023-07-07 13:42:24
3	3	P003	5	2023-07-07 13:42:24
4	4	P004	3	2023-07-07 13:42:24
5	5	P005	2	2023-07-07 13:42:24
6	1	P002	3	2023-07-07 13:42:24
7	3	P001	1	2023-07-07 13:42:24
8	2	P004	2	2023-07-07 13:42:24
9	4	P003	4	2023-07-07 13:42:24
10	5	P005	1	2023-07-07 13:42:24

region_code	region_name
02	서울특별시
031	경기도
032	인천광역시
033	강원특별자치도
041	충청남도
042	대전광역시
043	충청북도
044	세종특별자치시
051	부산광역시
052	울산광역시
053	대구광역시
054	경상북도
055	경상남도
061	전라남도
062	광주광역시
063	전라북도
064	제주특별자치도

Q1. 모든 고객의 이름과 이메일을 가져오는 질의

```
SELECT customer_name, email
FROM t_customer;
```

Q2. 특정 지역(예: '서울특별시')에 사는 고객의 이름과 전화번호를 가져오는 질의:

```
SELECT customer_name, phone
FROM t_customer
WHERE region_code = '02';
```

Q3. 특정 제품의 판매량과 판매 일자를 가져오는 질의:

```
SELECT qty, sales_date
FROM t_sales
WHERE product_code = 'P00110';
```

Q4. 제품별로 판매된 총 수량과 가격을 계산하는 질의:

```
SELECT product_name, SUM(qty) AS total_quantity, SUM(qty * price) AS total_price
FROM t_sales
JOIN t_product ON t_sales.product_code = t_product.product_code
GROUP BY product_name;
```

Q5. 특정 기간 동안의 판매량을 계산하는 질의

```
SELECT DATE(sales_date) AS sales_date, SUM(qty) AS total_quantity
FROM t_sales
WHERE sales_date BETWEEN '2023-01-01' AND '2023-07-30'
GROUP BY DATE(sales_date);
```

Q6. 가장 최근에 등록된 고객의 정보를 가져오는 질의

```
SELECT *
FROM t_customer
ORDER BY regist_date DESC
LIMIT 1;
```

Q7. 각 지역별로 고객 수를 계산하는 질의:

```
SELECT t_region.region_name, COUNT(t_customer.id) AS customer_count
FROM t_region LEFT JOIN t_customer
ON t_region.region_code = t_customer.region_code
GROUP BY t_region.region_name;
```

FROM: FROM 절에서는 t_region 테이블과 t_customer 테이블을 사용합니다. t_region 테이블은 왼쪽 테이블로 설정되고, t_customer 테이블은 조인 조건에 따라 오른쪽 테이블로 설정됩니다.

LEFT JOIN: LEFT JOIN은 t_region 테이블을 기준으로 왼쪽 테이블의 모든 레코드를 선택하고, 조인 조건에 맞는 t_customer 테이블의 레코드를 연결합니다. 이 때, t_region.region_code와 t_customer.region_code가 일치하는 조인 조건을 사용합니다. LEFT JOIN을 통해 모든 지역에 대한 고객 수를 계산할 수 있습니다.

GROUP BY: GROUP BY 절에서는 t_region.region_name을 기준으로 그룹화합니다. 이렇게 함으로써 지역별로 고객 수를 계산할 수 있습니다.

SELECT: SELECT 절에서는 t_region.region_name과 COUNT(t_customer.id)를 선택합니다. COUNT(t_customer.id)는 각 지역에 속하는 고객 수를 계산하는 집계 함수입니다.

Q8. 제품별로 판매된 총 가격과 평균 가격을 계산하는 질의

```
SELECT t_product.product_name, SUM(t_sales.qty * t_product.price) AS total_price,
AVG(t_product.price) AS average_price
FROM t_sales
JOIN t_product ON t_sales.product_code = t_product.product_code
GROUP BY t_product.product_name;
```

Q9. 특정 고객이 구매한 제품의 목록을 가져오는 질의:

```
SELECT t_product.product_name
FROM t_sales
JOIN t_product ON t_sales.product_code = t_product.product_code
WHERE t_sales.customer_id = 3;
```

Q10. 고객이 속한 지역의 이름과 해당 지역의 판매량을 가져오는 질의:

```
SELECT t_region.region_name, SUM(t_sales.qty) AS total_quantity
FROM t_customer
JOIN t_region ON t_customer.region_code = t_region.region_code
JOIN t_sales ON t_customer.id = t_sales.customer_id
GROUP BY t_region.region_name;
```

Q11. 고객이 구매한 제품의 평균 가격보다 높은 가격으로 판매된 제품의 목록을 가져오는 질의:

```
SELECT t_product.product_name, t_product.price
FROM t_product
WHERE t_product.price > (SELECT AVG(price) FROM t_product)
```

Q12. 고객 당 총 판매 금액:

```
SELECT t1.customer_name, SUM(t3.qty * t2.price) AS total_sales_amount
FROM t_customer t1
JOIN t_sales t3 ON t1.id = t3.customer_id
JOIN t_product t2 ON t2.product_code = t3.product_code
GROUP BY t1.customer_name;
```

Q13. 최고 판매 제품:

```
SELECT t2.product_name, SUM(t3.qty) AS total_quantity_sold
FROM t_product t2
JOIN t_sales t3 ON t2.product_code = t3.product_code
GROUP BY t2.product_name
ORDER BY total_quantity_sold DESC
LIMIT 5;
```

Q14. 월간 판매 추이:

```
SELECT DATE_FORMAT(t3.sales_date, '%Y-%m') AS sales_month, SUM(t3.qty * t2.price) AS total_sales_amount
FROM t_sales t3
JOIN t_product t2 ON t2.product_code = t3.product_code
GROUP BY sales_month
ORDER BY sales_month;
```

Q15. 판매당 평균 수량:

```
SELECT AVG(t3.qty) AS average_quantity
FROM t_sales t3;
```

Q16. 복수 구매 고객:

```
SELECT t1.customer_name, COUNT(DISTINCT t3.product_code) AS num_products_purchased
FROM t_customer t1
JOIN t_sales t3 ON t1.id = t3.customer_id
GROUP BY t1.customer_name
HAVING COUNT(DISTINCT t3.product_code) > 1;
```

Q17. 판매 테이블에서 고객의 ID, 이름, 주소와 함께 제품의 코드, 이름, 판매 날짜, 수량, 가격 및 계산된 총액이 포함되도록 질의:

```
SELECT t1.id, t1.customer_name, t1.address,
       t2.product_code, t2.product_name,
       t3.sales_date, t3.qty, t2.price, t3.qty * t2.price as total_amount
FROM t_customer t1
JOIN t_sales t3 ON t1.id = t3.customer_id
JOIN t_product t2 ON t2.product_code = t3.product_code
WHERE t1.id = 3;
```

Q18. 고객의 총 판매 금액과 함께 고객 정보를 검색하는 쿼리

```
SELECT
  c.customer_name,
  c.email,
  (
    SELECT SUM(s.qty * p.price)
    FROM t_sales s
    INNER JOIN t_product p ON s.product_code = p.product_code
    WHERE s.customer_id = c.id
  ) AS total_sales_amount
FROM t_customer c;
```

Q19. 구매한 총 제품 수와 함께 고객 이름을 검색

```
SELECT c.customer_name,
    (
        SELECT COUNT(*)
        FROM t_sales s
        WHERE s.customer_id = c.id
    ) AS total_products_purchased
FROM t_customer c;
```

Q20.