

데이터베이스 설계/SQL 단위 평가 시험 문제 구성

문제 1: 데이터베이스 모델링 (25점)

시나리오:

온라인 쇼핑몰을 위한 데이터베이스를 설계하고자 합니다. 이 쇼핑몰에서는 고객, 주문, 제품 정보를 관리해야 합니다. 고객은 여러 개의 주문을 할 수 있으며, 각 주문에는 여러 제품이 포함될 수 있습니다.

문제:

1. 아래의 요구사항을 충족하는 각 테이블의 스키마를 정의하는 SQL DDL(Create Table) 문을 작성하세요. (25점)

- 고객(Customer): 고객 ID, 이름, 이메일, 주소, 전화번호

```
1 • create table Customer(  
2     customer_id int AUTO_INCREMENT PRIMARY KEY  
3     ,customer_name varchar(100)  
4     ,customer_email varchar(100)  
5     ,customer_address varchar(100)  
6     ,customer_call int  
7  
8 )
```

- 제품(Product): 제품 ID, 이름, 가격, 재고

```
1 • create table Product(  
2     product_id int AUTO_INCREMENT PRIMARY KEY  
3     ,product_name varchar(100)  
4     ,product_price int  
5     ,product_inventory int  
6  
7 )
```

- 주문(Order): 주문 ID, 고객 ID, 주문 날짜

```

1 • create table Orders(
2     Orders_id int AUTO_INCREMENT PRIMARY KEY
3     ,customer_id int
4     ,Orders_date Date
5     ,FOREIGN KEY (customer_id)
6     REFERENCES customer(customer_id)
7
8 );

```

- 주문 상세(Order_Detail): 주문 ID, 제품 ID, 수량, 단가

```

1 • create table Orders_Detail(
2     Orders_id int
3     ,product_id int
4     ,Orders_count int
5     ,Orders_price int
6     ,FOREIGN KEY (Orders_id)
7     REFERENCES orders(Orders_id)
8 );

```

문제 2: 데이터 삽입 및 조회 (25점)

시나리오:

위에서 설계한 데이터베이스에 샘플 데이터를 삽입하고, 고객의 주문 내역을 조회하려고 합니다.

문제:

1. 각 테이블(Customer, Product, Order, Order_Detail)에 최소 2개의 샘플 데이터를 삽입하는 SQL INSERT 문을 작성하세요. (15점)

- 예시: 고객 2명, 제품 2개, 주문 2개, 각 주문당 2개의 제품

```

1  INSERT INTO customer
2  (customer_name,customer_email, customer_address, customer_call)
3  VALUES ("sukju","abc@gmail.com","gumi",01012341234)
4  ,("woosuk","aaa@gmail.com","daehu",01011111111);

```

-

- 1 • `INSERT INTO product`
- 2 `(product_name,product_price,product_inventory)`
- 3 `VALUES ("pencil",100,100)`
- 4 `,("apple",1000,10);`
-
- 1 • `INSERT INTO orders`
- 2 `(customer_id,Orders_date)`
- 3 `VALUES (1,'2024-08-05')`
- 4 `, (2,'2024-08-05');`
-
- 1 • `INSERT INTO orders_detail`
- 2 `(Orders_id,product_id,Orders_count,Orders_price)`
- 3 `VALUES (1,1,5,500)`
- 4 `, (1,2,5,5000)`
- 5 `, (2,1,3,300)`
- 6 `, (2,2,3,3000);`

-각 테이블 결과

- 고객은 실수로 결과르 캡처하지 못했음

	product_id	product_name	product_price	product_in
▶	1	pencil	100	100
	2	apple	1000	10
★	NULL	NULL	NULL	NULL

-

	Orders_id	customer_id	Orders_date
▶	1	1	2024-08-05
	2	2	2024-08-05
★	NULL	NULL	NULL

-

	Orders_id	product_id	Orders_count	Orders_price
▶	1	1	5	500
	1	2	5	5000
	2	1	3	300
○	2	2	3	3000

- 2. 특정 고객이 한 주문에 포함된 모든 제품과 그 수량을 조회하는 SQL SELECT 문을 작성하세요. (10점)

문제 3: 데이터 갱신 및 삭제 (20점)

시나리오:

고객의 주소가 변경되었으며, 특정 주문에 포함된 제품을 삭제하려고 합니다.

문제:

- 1. 고객 ID가 1인 고객의 주소를 "New Address, City, Country"로 업데이트하는 SQL UPDATE 문을 작성하세요. (10점)

- 쿼리문

```
1 • update customer
2   set customer_address = "gumi-bongkok-korea"
3   where customer_id = 1;
```

• 결과

	customer_id	customer_name	customer_email	customer_address	customer_call
▶	1	sukju	abc@gmail.com	gumi-bongkok-korea	1012341234
	2	woosuk	aaa@gmail.com	daehu	1011111111
•	NULL	NULL	NULL	NULL	NULL

2. 주문 ID가 1인 주문에서 특정 제품(예: 제품 ID 2)을 삭제하는 SQL DELETE 문을 작성하세요. (10점)

• 쿼리문

```
1 • delete from orders_detail
2   where Orders_id = 1 and product_id = 2 ;
```

• 결과

	Orders_id	product_id	Orders_count	Orders_price
▶	1	1	5	500
	2	1	3	300
•	2	2	3	3000

문제 4: JOIN을 활용한 데이터 조회 (30점)

시나리오:

쇼핑몰 관리자에게 주문에 포함된 모든 제품의 정보를 보여주는 보고서를 생성해야 합니다.

4번 문제 전에 값 및 외래키 수정

• 값수정

```
1 update orders_detail
2   set Orders_price = 100
3   where product_id = 1
```

```
1 update orders_detail
2   set Orders_price = 1000
3   where product_id = 2
```

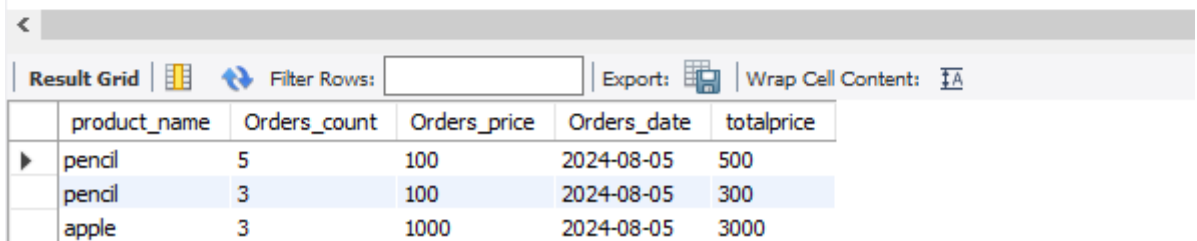
• 외래키 추가

```
1 ALTER TABLE orders_detail
2 ADD CONSTRAINT sameproduct
3 foreign key (product_id) REFERENCES product(product_id);
```

문제:

1. 각 주문에 포함된 제품의 이름, 수량, 단가, 총 금액(수량 * 단가), 그리고 주문 날짜를 조회하는 SQL JOIN 문을 작성하세요. (15점)

```
1 • SELECT
2     p.product_name
3     ,d.Orders_count
4     ,d.Orders_price
5     ,o.Orders_date
6     ,(d.Orders_price * d.Orders_count) as totalprice
7 FROM
8     orders_detail d
9 join
10    orders o
11 join
12    product p
13 ON
14    p.product_id = d.product_id and o.Orders_id =d.Orders_id
15
```





The screenshot shows a database interface with a 'Result Grid' tab. The grid displays the results of the SQL query, with columns: product_name, Orders_count, Orders_price, Orders_date, and totalprice. The data is as follows:

	product_name	Orders_count	Orders_price	Orders_date	totalprice
▶	pencil	5	100	2024-08-05	500
	pencil	3	100	2024-08-05	300
	apple	3	1000	2024-08-05	3000

o

2. 각 고객의 이름과 그들이 총 몇 개의 제품을 구매했는지를 보여주는 SQL JOIN 문을 작성하세요. (15점)

```
1 • SELECT
2     c.customer_name
3     ,SUM(d.Orders_count) AS TotalProducts
4 FROM
5
6     orders o
7 join
8     customer c
9 on o.customer_id = c.customer_id
10
11 join orders_detail d
12 on o.Orders_id = d.Orders_id
13
14 group by
15     c.customer_name;
16
```

<		
Result Grid		
Filter Rows: <input type="text"/>		
Export:  Wrap Cell Content: 		
	customer_name	TotalProducts
▶	sukju	5
	woosuk	6

○

평가 기준

- **데이터베이스 모델링**: 관계형 데이터베이스 구조를 이해하며 적절한 테이블을 설계하는 능력.
- **데이터 삽입 및 조회**: SQL INSERT 및 SELECT 문을 통해 데이터를 정확히 삽입하고, 원하는 데이터를 조회하는 능력.
- **데이터 갱신 및 삭제**: SQL UPDATE 및 DELETE 문을 사용하여 데이터를 갱신하고 삭제하는 능력.
- **JOIN을 활용한 데이터 조회**: SQL JOIN 문을 통해 여러 테이블 간의 관계를 이해하고, 복합적인 데이터를 정확히 조회하는 능력.

시험 시간

1시간