General

SQL 기초 & 자주쓰는 쿼리문 정리

365kim 하루 2021. 3. 29. 00:33

SQL을 배워야하는 이유

데이터베이스와 SQL

우리는 일상 속에서 'DB' 또는 '데이터베이스'라는 단어를 어렵지 않게 접한다. 보통 '관리 목적으로 데이터를 모아놓은 것'을 의미할 때 사용한다. 이 '데이터베이스'는 언제부터 우리의 일상 속에 녹아들었을까?

1970년으로 거슬러 올라가 보자. 당시 대기업과 엔지니어들은 누구나 쉽게 데이터를 정리 정돈할 수 있는 전문적인 소프트웨어가 필요로 했다. 이러한 수요에 발맞춰, 영국의 컴퓨터 과학자, 에드거 테드(1923~2003)는 IBM에서 일하는 동안 데이터베이스 관리를 위한 모델을 만들었다. 그리고 그의 이론에 기반해서 여러 가지 관계형 데이터베이스가 등장하기 시작했다. 데이터베이스의 가장 핵심적인 기능은 'CRUD'이다. 각각 생성(Create), 조회(Read), 갱신(Update), 삭제(Delete)를 의미한다.

'데이터베이스'는 데이터를 '표'로 표현해준다는 점에서 '엑셀 시트'와 비슷하다. '엑셀 시트'에서 마우스 클릭을 통해 데이터를 제어하듯이, '데이터베이스'에는 SQL로 데이터를 제어할 수 있다. 즉, SQL은 '데이터베이스'에서 자료를 다룰 때 사용하는 언어라고 할 수 있다.

구조화된 질의 언어, SQL

SQL은 'Structured Query Language'의 약자이다. 이때 'Structured'는 데이터가 표로 정리되어 구조화되어있다는 뜻이고, 'Query'는 사용자가 데이터베이스에게 '데이터를 넣어줘, 삭제해줘, 수정해줘, 읽어줘' 등의 요청을 할 수 있다는 뜻이며, 'Language'는 데이터베이스도 이해할 수 있고 사용자도 이해할 수 있는 언어로 요청한다는 뜻이다.

TMI: SQL은 HTML과 함께 배우기 쉬운 언어로 꼽힌다.

그래서 SQL은 왜 배워야 할까?

SQL은 관계형 데이터베이스를 다루는데 쓰이는 '표준화된' 언어이다. 표준화되었다는 말은 여기저기서 쓰임이 많다는 뜻이기도 하다. 실제로 많은 프로젝트에서, 앱의 주요 데이터를 저장하는 DBMS(Database Management System)를 SQL 언어를 사용해서 다룬다. 따라서, 개발자라면 분야를 막론하고 기본적인 SQL문을 익혀놓는 것이 좋다. SQL문을 자유롭게 구사할 수 있다면 데이터를 더 편하게 다룰 수 있을 것이다.

📌 SQL 자주 쓰는 쿼리문

쿼리문은 어떤 것을 제어하느냐에 따라, 데이터베이스 제어문, 테이블 제어문, 데이터 제어문으로 나눌 수 있다. 자주 쓰는 구문만 간단하게 살펴보자.

데이터베이스 다루기

```
-- 모든 데이터베이스 조회
SHOW DATABASES;

-- 데이터베이스 생성
CREATE DATABASE practice DEFAULT CHARACTER SET utf8 COLLATE utf8_general_ci;

-- 데이터베이스 삭제
DROP DATABASE IF EXISTS practice;
```

테이블 다루기


```
-- 테이블 조회하기
use mysql;
SHOW TABLES;

-- 테이블 구조 확인하기
DESC [테이블명];

-- 테이블 생성하기
CREATE DATABASE practice DEFAULT CHARACTER SET utf8 COLLATE utf8_general_ci;

-- 조회 결과로 테이블 생성하기
CREATE TABLE [생성할 테이블명] AS (SELECT * FROM [기존 테이블명]);

-- 테이블 삭제하기
DROP DATABASE IF EXISTS practice;
```

데이터 다루기

쿼리문의 실행 순서

데이터를 다루기 앞서 쿼리문의 실행 순서를 살펴보자.

```
-- 쿼리 처리순서 알아보기
SELECT city AS 도시, COUNT(city) AS 집계
  FROM user
    WHERE user.age >= 18
      GROUP BY city
        HAVING city >= 'b'
          ORDER BY city
```

위의 예시와 같이 SELECT - FROM - WHERE - GROUP BY - HAVING - ORDER BY 순서로 작성했을 때, 쿼리문은 왼쪽부터 순서대로 처리되지 않는다. 실행 순서는 FROM - WHERE - GROUP BY - HAVING - SELECT - ORDER BY 순이다. 어떤 과정을 거쳐 실행되는지 이해하기 위해 아래 설명을 참고해보자.

1. 우선 FROM에서 테이블을 선택하여 데이터 집합을 만든다.
2. 그리고 WHERE으로 FROM에서 만든 데이터 집합을 조건에 맞게 일부를 선택한다.
3. GROUP BY는 WHERE에서 선택된 데이터를 그룹화한다.
4. HAVING은 GROUP BY에서 그룹핑한 데이터 집합을 다시 조건에 맞게 필터링한다.
5. SELECT는 그룹화 및 필터링된 데이터 집합을 집계한다.
6. ORDER BY는 집계한 데이터 집합을 최종적으로 정렬한다.

데이터 다루기 - 기본

```
-- 기본 데이터 조회하기
SELECT * FROM [테이블명];

-- 별칭 사용하기 (ORDER BY에 사용가능, WHERE에 사용불가)
SELECT CustomerName AS 고객, Address AS 주소, PostalCode AS 우편번호 FROM Cust
omers;

-- 특정 행 특정 열 조회하기
SELECT [열1, 열2, ...] FROM [테이블명] WHERE [행 선택 조건식]
SELECT * FROM Customers WHERE Country = 'Germany';
SELECT * FROM Orders WHERE ShipperID <> 2;
SELECT * FROM OrderDetails WHERE Quantity > 100;
SELECT * FROM Employees WHERE FirstName >= 'O';
SELECT * FROM Employees WHERE BirthDate <= '1950-01-01';

-- 특정 패턴 조회하기
SELECT * FROM [테이블명] WHERE text LIKE '%우아한%';
SELECT * FROM [테이블명] WHERE number BETWEEN 1 and 3;
SELECT * FROM [테이블명] WHERE text IN (1, 2, 3);
SELECT * FROM [테이블명] WHERE text IS NULL;

-- 검색결과 일부 출력 후 정렬하기 (내림차순 DESC)
SELECT [열1, 열2, ...] FROM [테이블명] ORDER BY [열1, 열2, ...] ASC LIMIT 5;

-- 데이터 가공하기
SELECT 1 - 2 + 2 * 3;
SELECT MOD(10, 3)
SELECT ROUND(30.60, 1)
SELECT CONCAT('우아한', '형제들')
SELECT SUBSTRING('20190422', 1, 4)
SELECT CURDATE();
SELECT CURTIME();

-- 데이터 집계하기
```

```
SELECT COUNT(*) FROM [테이블명]
SELECT DISTINCT [열명] FROM [테이블명]
SELECT SUM([열명]) FROM [테이블명]
```


데이터 다루기 - 심화

```
-- 쿼리 중첩하기
SELECT MIN(a) FROM sample;
DELETE FROM sample WHERE a = (SELECT MIN(a) FROM sample);
SELECT (SELECT COUNT(*) FROM [Customers] WHERE Country = 'Germany') AS German
yCount,
       (SELECT COUNT(*) FROM [Customers] WHERE Country = 'Mexico') AS MexicoC
ount;
SELECT * FROM (SELECT FirstName, LastName FROM [Employees] WHERE EmployeeID <
10);
SELECT * FROM [OrderDetails] WHERE Quantity = (SELECT MAX(Quantity) FROM [Ord
erDetails]);

-- 서브쿼리가 부모쿼리와 연관된 경우
DELETE FROM [Customers] WHERE EXISTS (SELECT * FROM [Orders] WHERE OrderDate
 >= "1996-07-08");

-- 여러테이블 다루기: 합집합
SELECT Country FROM [Customers] UNION SELECT Country FROM [Suppliers] ORDER B
Y Country;
SELECT Country FROM [Customers] UNION ALL SELECT Country FROM [Suppliers] ORD
ER BY Country;

-- 여러테이블 다루기: 내부결합
SELECT * FROM [Products], [Employees] WHERE [Employees].EmployeeID = [Product
s].SupplierID;
SELECT * FROM [Products] INNER JOIN [OrderDetails] ON Products.ProductID = Or
derDetails.ProductID;

-- 여러테이블 다루기: 외부결합
SELECT * FROM [Products] LEFT JOIN [Employees] ON [Employees].EmployeeID = [P
roducts].SupplierID;
SELECT * FROM [Employees] RIGHT JOIN [Products] ON [Employees].EmployeeID =
 [Products].SupplierID;
```

🚀 SQL 쿼리문 적용해보기

기본 쿼리문을 살펴보았으니 이제 미션에 적용해볼 시간이다. 코치 CU의 미션은 주어진 3가지 과제를 해결할 수 있는 SQL문을 작성하는 것이었다. **실습 사이트**에는 아래와 같은 데이터베이스가 준비되어 있다. 각 테이블의 이름을 클릭하면 'SELECT * FROM [테이블명];' 쿼리문이 자동 실행된다. 이제 한 문제씩 해결해보자!

MySQL Database:

Tablenames	Records
Customers	91
Categories	8
Employees	9
OrderDetails	2155
Orders	830
Products	77
Shippers	3
Suppliers	29

첫 번째 미션 해결! ☝

> 미션 1. 200개 이상 팔린 상품명과 그 수량을 수량 기준 내림차순으로 보여주세요.

첫 번째 미션을 위해 필요한 항목은 '상품명(ProductName)', '판매수량(Quantity)'이고, 각각 'Products'테이블과, 'OrderDetails'테이블에서 구할 수 있다.

Number of Records: 77

ProductID	ProductName	SupplierID	CategoryID	Unit	Price
1	Chais	1	1	10 boxes x 20 bags	18.00
2	Chang	1	1	24 - 12 oz bottles	19.00
3	Aniseed Syrup	1	2	12 - 550 ml bottles	10.00
4	Chef Anton's Cajun Seasoning	2	2	48 - 6 oz jars	22.00
5	Chef Anton's Gumbo Mix	2	2	36 boxes	21.35
6	Grandma's Boysenberry Spread	3	2	12 - 8 oz jars	25.00
7	Uncle Bob's Organic Dried Pears	3	7	12 - 1 lb pkgs.	30.00
8	Northwoods Cranberry Sauce	3	2	12 - 12 oz jars	40.00

Number of Records: 2155

OrderDetailID	OrderID	ProductID	Quantity
1	10248	11	12
2	10248	42	10
3	10248	72	5
4	10249	14	9
5	10249	51	40
6	10250	41	10
7	10250	51	35
8	10250	65	15

(왼쪽그림) Products 테이블 / (오른쪽그림) OrderDetails 테이블

다음과 같이 순차적으로 접근하면 원하는 테이블을 구성할 수 있다. 2개의 테이블 연결할 때에 INNER JOIN [테이블2] ON 조건 방식을 사용하였다.

365kim 구독하기

```sql
-- 1. 두 테이블 ProductID 로 서로 연결하기
SELECT Products.ProductName, OrderDetails.Quantity
  FROM Products
    INNER JOIN OrderDetails ON Products.ProductID = OrderDetails.ProductID;


-- 2. 판매수량을 ProductName 별로 합치기
SELECT Products.ProductName, SUM(OrderDetails.Quantity)
  FROM Products
    INNER JOIN OrderDetails ON Products.ProductID = OrderDetails.ProductID
      GROUP BY Products.ProductName;


-- 3. 판매수량이 200 이상인 상품만 필터링하기
SELECT Products.ProductName, SUM(OrderDetails.Quantity) AS OrderQuantity
  FROM Products
    INNER JOIN OrderDetails ON Products.ProductID = OrderDetails.ProductID
      GROUP BY Products.ProductName
        HAVING SUM(OrderDetails.Quantity) >= 200


-- 4. 판매수량 내림차순으로 정렬하기
SELECT Products.ProductName, SUM(OrderDetails.Quantity) AS OrderQuantity
  FROM Products
    INNER JOIN OrderDetails ON Products.ProductID = OrderDetails.ProductID
      GROUP BY Products.ProductName
        HAVING SUM(OrderDetails.Quantity) >= 200
          ORDER BY OrderQuantity DESC;
```

Number of Records: 2155

ProductName	Quantity
Queso Cabrales	12
Singaporean Hokkien Fried Mee	10
Mozzarella di Giovanni	5
Tofu	9
Manjimup Dried Apples	40
Jack's New England Clam Chowder	10
Manjimup Dried Apples	35
Louisiana Fiery Hot Pepper Sauce	15

Number of Records: 77

ProductName	SUM(OrderDetails.Quantity)
Chais	828
Chang	1057
Aniseed Syrup	328
Chef Anton's Cajun Seasoning	453
Chef Anton's Gumbo Mix	298
Grandma's Boysenberry Spread	301
Uncle Bob's Organic Dried Pears	763

Number of Records: 72

ProductName	OrderQuantity
Camembert Pierrot	1577
Raclette Courdavault	1496
Gorgonzola Telino	1397
Gnocchi di nonna Alice	1263
Pavlova	1158
Rhönbräu Klosterbier	1155
Guaraná Fantástica	1125
Boston Crab Meat	1103

(왼쪽그림) 1번 실행결과 / (중앙그림) 2번,3번 실행결과 / (오른쪽그림) 4번 실행결과

두 번째 미션 해결! 👌

> 미션 2. 많이 주문한 순으로 고객 리스트(ID, 고객명)를 구해주세요. (고객별 구매한 물품 총 개수)

두 번째 미션을 위해 필요한 항목은 '고객 아이디(CustomerID)', '고객 이름(CustomerName)',) 'Quantity(주문 수량)'이고, 각각 'Customers'테이블과, 'Orders', 'OrderDetails'테이블에서 구할 수 있다.

Number of Records: 91

CustomerID	CustomerName	ContactName	Address	City	PostalCode	Country
1	Alfreds Futterkiste	Maria Anders	Obere Str. 57	Berlin	12209	Germany
2	Ana Trujillo Emparedados y helados	Ana Trujillo	Avda. de la Constitución 2222	México D.F.	05021	Mexico
3	Antonio Moreno Taquería	Antonio Moreno	Mataderos 2312	México D.F.	05023	Mexico
4	Around the Horn	Thomas Hardy	120 Hanover Sq.	London	WA1 1DP	UK
5	Berglunds snabbköp	Christina Berglund	Berguvsvägen 8	Luleå	S-958 22	Sweden
6	Blauer See Delikatessen	Hanna Moos	Forsterstr. 57	Mannheim	68306	Germany
7	Blondel père et fils	Frédérique Citeaux	24, place Kléber	Strasbourg	67000	France
8	Bólido Comidas preparadas	Martín Sommer	C/ Araquil, 67	Madrid	28023	Spain

Number of Records: 830

OrderID	CustomerID	EmployeeID	OrderDate	ShipperID
10248	90	5	1996-07-04	3
10249	81	6	1996-07-05	1
10250	34	4	1996-07-08	2
10251	84	3	1996-07-08	1
10252	76	4	1996-07-09	2
10253	34	3	1996-07-10	2
10254	14	5	1996-07-11	2
10255	68	9	1996-07-12	3

Number of Records: 2155

OrderDetailID	OrderID	ProductID	Quantity
1	10248	11	12
2	10248	42	10
3	10248	72	5
4	10249	14	9
5	10249	51	40
6	10250	41	10
7	10250	51	35
8	10250	65	15

(왼쪽그림) Customers 테이블 / (중앙그림) Orders 테이블 / (오른쪽그림) OrderDetails 테이블

첫 번째 미션에서 활용했던 구문을 그대로 활용해보자. 이번에는 테이블 3개를 연결해야하는데 이때는 INNER JOIN 구문을 한번 더 적어주기만 하면 된다. A - B - C 순으로 연결할 것이라면 FROM 뒤에는 A 테이블을 적어주어야 한다는 점에 유의하자.

```
-- 1. [Orders]-[OrderDetails] 두 테이블 OrderID 로 서로 연결하기
SELECT Orders.CustomerID, OrderDetails.Quantity
  FROM Orders
    INNER JOIN OrderDetails ON Orders.OrderID = OrderDetails.OrderID;


-- 2. 주문수량을 ProductName 별로 합치기
SELECT Orders.CustomerID, SUM(OrderDetails.Quantity) AS OrderQuantity
  FROM Orders
    INNER JOIN OrderDetails ON Orders.OrderID = OrderDetails.OrderID
      GROUP BY Orders.CustomerID;


-- 3. Customers 테이블 연결해서 고객이름 표시하기
SELECT Orders.CustomerID, Customers.CustomerName, SUM(OrderDetails.Quantity)
AS OrderQuantity
  FROM Customers
    INNER JOIN Orders ON Customers.CustomerID = Orders.CustomerID
      INNER JOIN OrderDetails ON Orders.OrderID = OrderDetails.OrderID
        GROUP BY Orders.CustomerID;
```

365kim 구독하기

```
-- 4. 주문수량 내림차순으로 정렬하기
SELECT Orders.CustomerID, Customers.CustomerName, SUM(OrderDetails.Quantity)
AS OrderQuantity
  FROM Customers
    INNER JOIN Orders ON Customers.CustomerID = Orders.CustomerID
      INNER JOIN OrderDetails ON Orders.OrderID = OrderDetails.OrderID
        GROUP BY Orders.CustomerID
          ORDER BY OrderQuantity DESC;
```

Number of Records: 2155

CustomerID	Quantity
90	12
90	10
90	5
81	9
81	40
34	10
34	35
34	15

Number of Records: 89

CustomerID	OrderQuantity
1	174
2	63
3	359
4	650
5	1001
6	140
7	666
8	190

Number of Records: 89

CustomerID	CustomerName	OrderQuantity
71	Save-a-lot Markets	4958
20	Ernst Handel	4543
63	QUICK-Stop	3961
37	Hungry Owl All-Night Grocers	1684
25	Frankenversand	1525
65	Rattlesnake Canyon Grocery	1383
24	Folk och fä HB	1234
35	HILARIÓN-Abastos	1096

(왼쪽그림) 1번 실행결과 / (중앙그림) 2번 실행결과 / (오른쪽그림) 4번 실행결과

세 번째 미션 해결! 🤞

🛟주의: 아래의 해설을 정답이 아닙니다. (서니 제보 감사해요!)
Customer 중에 아예 구매를 하지 않은 2명이 있어서 INNER JOIN이 아닌 LEFT JOIN이나 RIGHT JOIN으로 작성해야 정답 인원 수 91명이 산출됩니다!

> 미션 3. 많은 돈을 지출한 순으로 고객 리스트를 구해주세요.

세 번째 미션을 위해 필요한 항목은 '고객 아이디(CustomerID)', '고객 이름(CustomerName)', 'Quantity(주문 수량)', Price(가격)이고, 각각 'Customers'테이블과, 'Orders', 'OrderDetails'테이블, 'Products'테이블에서 구할 수 있다.

Number of Records: 830

OrderID	CustomerID	EmployeeID	OrderDate	ShipperID
10248	90	5	1996-07-04	3
10249	81	6	1996-07-05	1
10250	34	4	1996-07-08	2
10251	84	3	1996-07-08	1
10252	76	4	1996-07-09	2
10253	34	3	1996-07-10	2
10254	14	5	1996-07-11	2
10255	68	9	1996-07-12	3

Number of Records: 2155

OrderDetailID	OrderID	ProductID	Quantity
1	10248	11	12
2	10248	42	10
3	10248	72	5
4	10249	14	9
5	10249	51	40
6	10250	41	10
7	10250	51	35
8	10250	65	15

Number of Records: 77

ProductID	ProductName	SupplierID	CategoryID	Unit	Price
1	Chais	1	1	10 boxes x 20 bags	18.00
2	Chang	1	1	24 - 12 oz bottles	19.00
3	Aniseed Syrup	1	2	12 - 550 ml bottles	10.00
4	Chef Anton's Cajun Seasoning	2	2	48 - 6 oz jars	22.00
5	Chef Anton's Gumbo Mix	2	2	36 boxes	21.35
6	Grandma's Boysenberry Spread	3	2	12 - 8 oz jars	25.00
7	Uncle Bob's Organic Dried Pears	3	7	12 - 1 lb pkgs.	30.00
8	Northwoods Cranberry Sauce	3	2	12 - 12 oz jars	40.00

(왼쪽그림) Orders 테이블 / (중앙그림) OrderDetails 테이블 / (오른쪽그림) Products 테이블

이번에는 테이블 4개를 연결해야하는데, 역시 INNER JOIN 구문을 이용하니 어렵지 않게 해결할 수 있었다.

```
-- 1. [OrderDetails]-[Products] 두 테이블 OrderID 로 서로 연결하기
SELECT SUM(OrderDetails.Quantity * Products.Price) AS PaymentAmount
  FROM OrderDetails
    INNER JOIN Products ON OrderDetails.ProductID = Products.ProductID;


-- 2. [Orders] 테이블 추가로 연결하기
SELECT Orders.CustomerID, SUM(OrderDetails.Quantity * Products.Price) AS Paym
entAmount
  FROM Orders
    INNER JOIN OrderDetails ON Orders.OrderID = OrderDetails.OrderID
      INNER JOIN Products ON OrderDetails.ProductID = Products.ProductID
        GROUP BY Orders.CustomerID;


-- 3. [Customers] 테이블 연결해서 고객이름 표시하기
SELECT Orders.CustomerID, Customers.CustomerName, SUM(OrderDetails.Quantity *
Products.Price) AS PaymentAmount
  FROM Customers
    INNER JOIN Orders ON Customers.CustomerID = Orders.CustomerID
      INNER JOIN OrderDetails ON Orders.OrderID = OrderDetails.OrderID
        INNER JOIN Products ON OrderDetails.ProductID = Products.ProductID
          GROUP BY Orders.CustomerID;


-- 4. 결제금액 내림차순으로 정렬하기
SELECT Orders.CustomerID, Customers.CustomerName, SUM(OrderDetails.Quantity *
Products.Price) AS PaymentAmount
  FROM Customers
    INNER JOIN Orders ON Customers.CustomerID = Orders.CustomerID
      INNER JOIN OrderDetails ON Orders.OrderID = OrderDetails.OrderID
        INNER JOIN Products ON OrderDetails.ProductID = Products.ProductID
          GROUP BY Orders.CustomerID;
            ORDER BY SUM(OrderDetails.Quantity * Products.Price) DESC;
```

Number of Records: 1

PaymentAmount
1449367.31

Number of Records: 89

CustomerID	PaymentAmount
1	4596.20
2	1425.15
3	7616.15
4	14264.50
5	28355.75
6	3239.80
7	22606.05
8	5543.30

Number of Records: 89

CustomerID	CustomerName	nt
63	QUICK-Stop	
71	Save-a-lot Markets	120718.85
20	Ernst Handel	120390.09
37	Hungry Owl All-Night Grocers	60397.91
65	Rattlesnake Canyon Grocery	58562.42
51	Mère Paillarde	36878.50
34	Hanari Carnes	34916.60
62	Queen Cozinha	34043.90

(왼쪽그림) 1번 실행결과 / (중앙그림) 2번 실행결과 / (오른쪽그림) 4번 실행결과

참고자료

생활코딩 - 데이터베이스 MySQL강의

'**General**' 카테고리의 다른 글

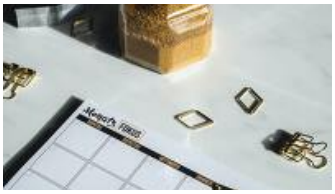
[프로젝트매니징] 이슈 관리 & 일정 추정 도전기 (0) 2021.07.02
SQL 기초 & 자주쓰는 쿼리문 정리 (5) 2021.03.29
Git - 자주 사용하는 명령어 모음 (0) 2021.02.12
[번역] 성급한 추상화에 관하여 - AHA Programming (Kent C. Dodds) (1) 2021.02.11
엑셀 자동화 - 구글 스프레드시트 매크로로 데일리 플래너 만들기 (4) 2021.01.28
VS Code 필수 초기 설정 - 기본설정부터 익스텐션까지 (0) 2021.01.26

태그

#select * from, #sql, #SQL 기초, #데이터베이스

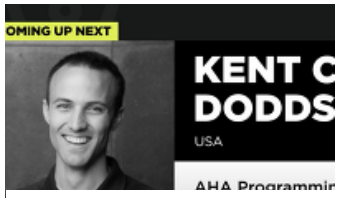
'General' Related Articles



[프로젝트매니징] ...

Git - 자주 사용하는 ...



[번역] 성급한 추상...

엑셀 자동화 - 구글 ...

youngyoungsw2020 2021.05.25 10:21 댓글주소 수정/삭제 댓글쓰기

너무 정리를 잘해두셨네요! 감사합니다

365kim 하루 2021.09.07 19:11 신고 댓글주소 수정/삭제

감사합니다 :)

mwu4ever 2021.09.07 10:53 댓글주소 수정/삭제 댓글쓰기

감사합니다. 이것만 한번 따라치고 두세번 풀어보니 쿼리가 어느정도 쉽게 느껴지네요.
이제 걸음마 뗀 느낌입니다. ㅎㅎ

365kim 하루 2021.09.07 19:11 신고 댓글주소 수정/삭제

도움이 되신 것 같아 기쁘네요 :) 감사합니다.

mwu4ever 2021.09.08 14:04 댓글주소 수정/삭제

한가지 궁금한 것이 있습니다.
2번문제를 보면 주체가 되는 테이블을 FROM 테이블 이라고 쓰는 것 같은데, 제가 Orders테이블을 주가 되는 테이블로 놓아도 결과의 차이가 없어보입니다. 이는 정해진 규칙이 있는 것인가요 아니면 실제 결과 값의 차이가 있을수도 있나요?

이름 암호 Secret

여러분의 소중한 댓글을 입력해주세요.

댓글달기

365kim 구독하기

태그

#리액트 #피그마
#우아한테크코스 테코톡
#사용자경험
#레이캐스팅
#ux디자인
#우아한테크코스 코드리뷰
#42seoul #영어공부
#레이캐스팅 튜토리얼
#cub3d #UI 개발
#cypress
#영어공부 습관
#UI디자인
#우아한테크코스 프론트엔드
#42서울
더보기

최근 포스트

[성능최적화] 2편 - 요청 사이즈 다이어트
웹 성능 핵심개념 - 크리티컬 렌더링 패스
순수(?) 리액트 앱에 webpack 설정하기 (without CRA)
[테코톡 요약] 우아한테크코스 3기 레벨3 테코톡 모아보기
UI 특강 2탄 - 실무에 대하여
UI 특강 1탄 - 궁금한 점 답변드립니다.
[프로젝트매니징] 이슈 관리 & 일정 추정 도전기

검색

검색내용을 입