1.customer table

create the customer table

CREATE TABLE Customer (

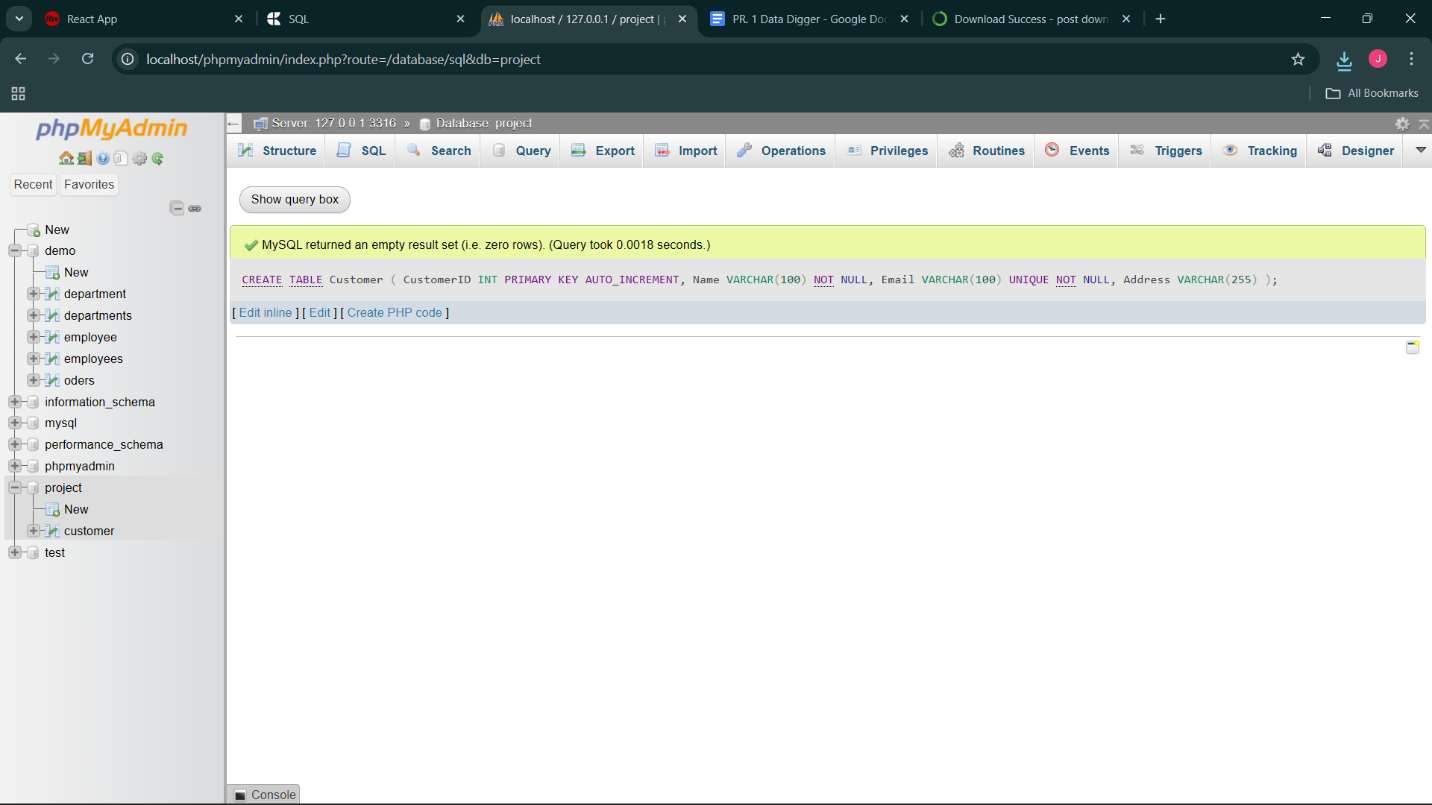
CustomerID INT PRIMARY KEY AUTO\_INCREMENT,

Name VARCHAR(100) NOT NULL,

Email VARCHAR(100) UNIQUE NOT NULL,

Address VARCHAR(255)

);



1.Insert the values in table

INSERT INTO Customer (Name, Email, Address) VALUES

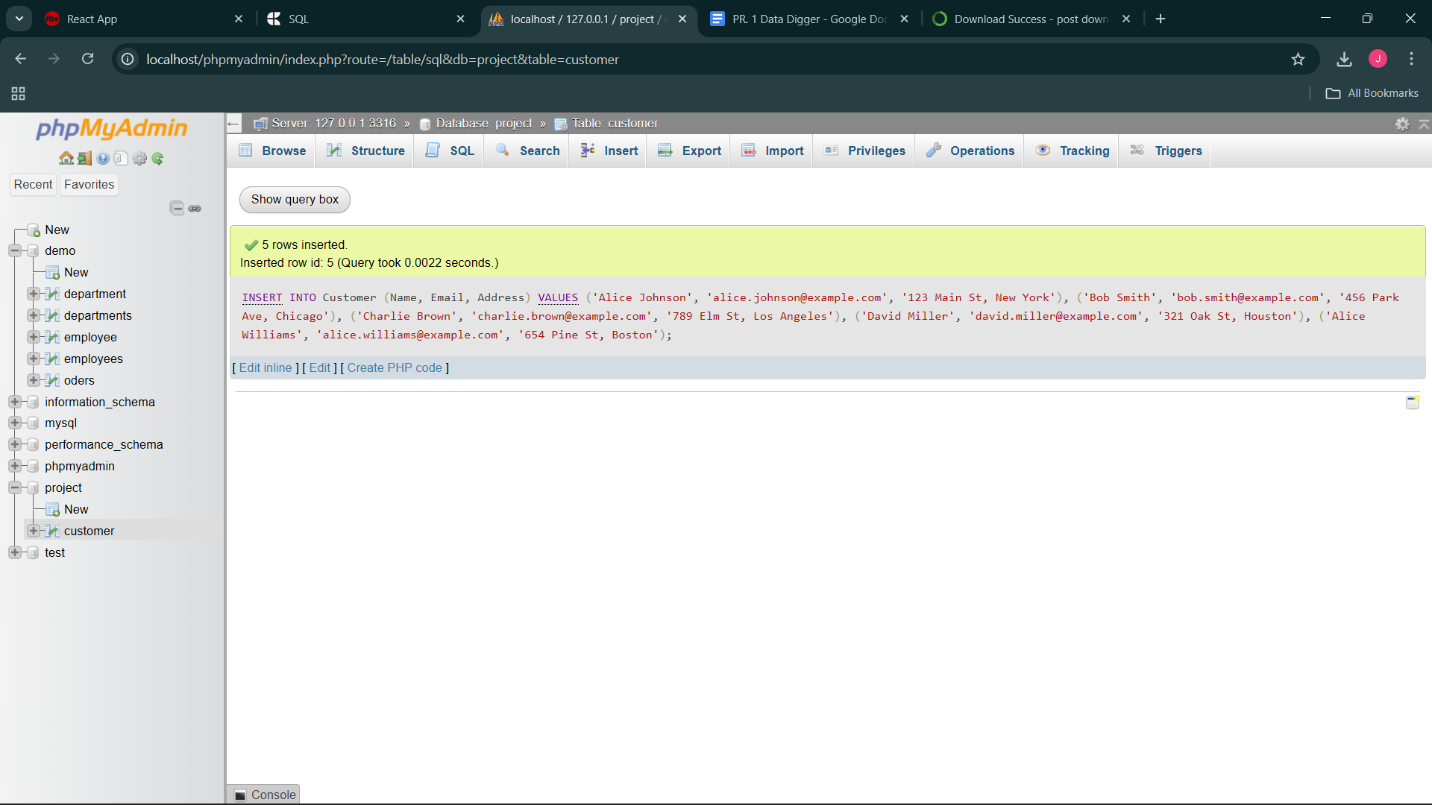
('Alice Johnson', 'alice.johnson@example.com', '123 Main St, New York'),

('Bob Smith', 'bob.smith@example.com', '456 Park Ave, Chicago'),

('Charlie Brown', 'charlie.brown@example.com', '789 Elm St, Los Angeles'),

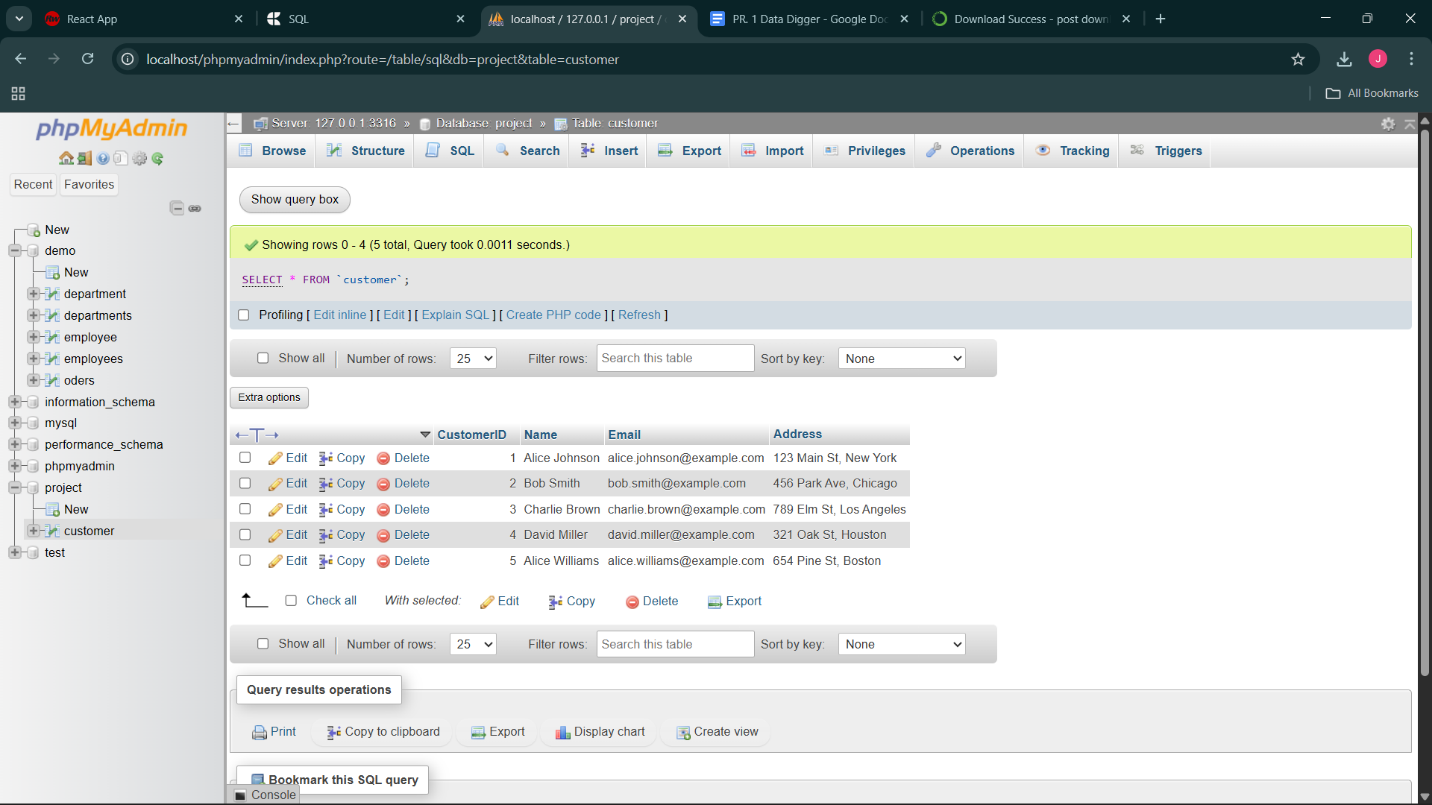
('David Miller', 'david.miller@example.com', '321 Oak St, Houston'),

('Alice Williams', 'alice.williams@example.com', '654 Pine St, Boston');



2.retrive the all records

SELECT \* FROM `customer`

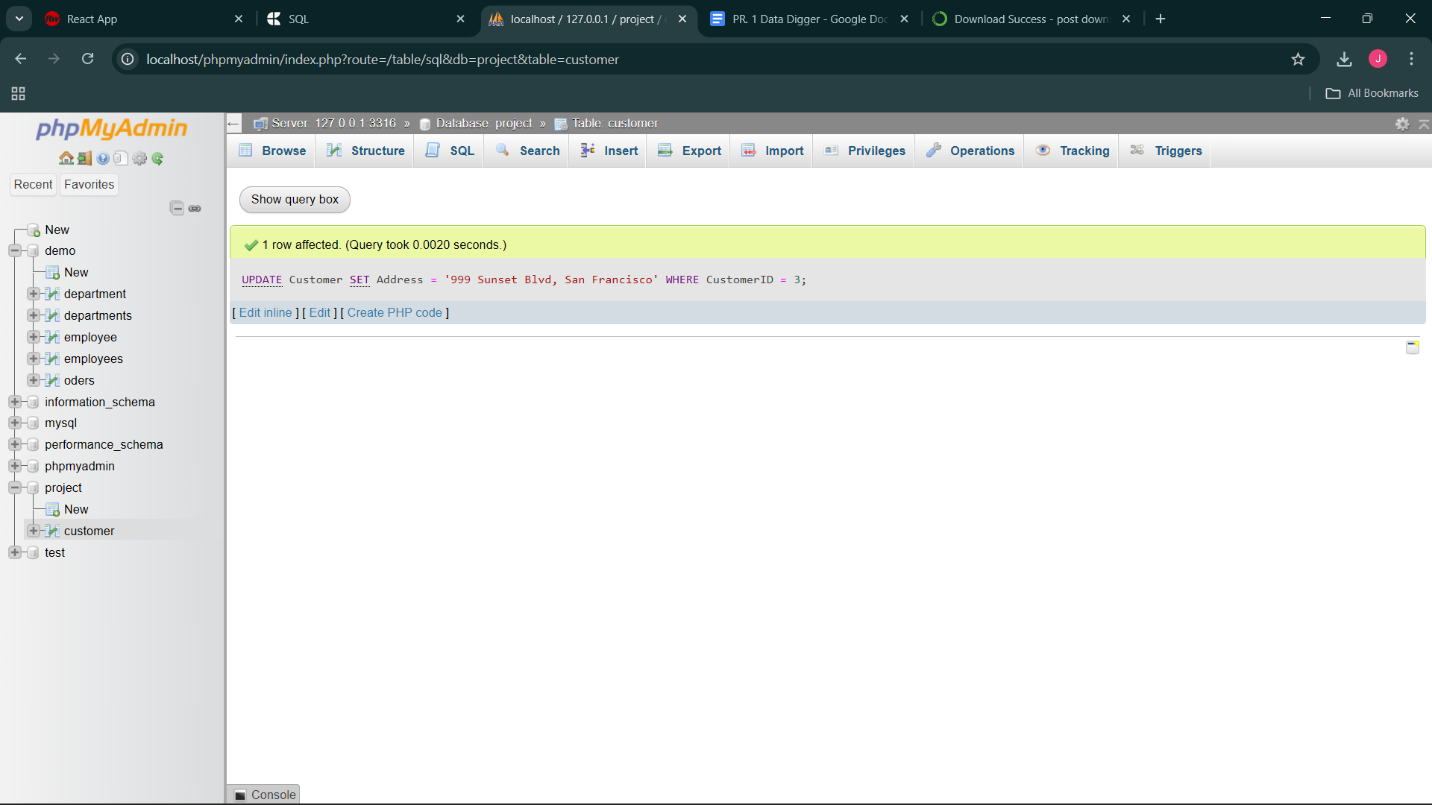


3. update the any customer’s address

UPDATE Customer

SET Address = '999 Sunset Blvd, San Francisco'

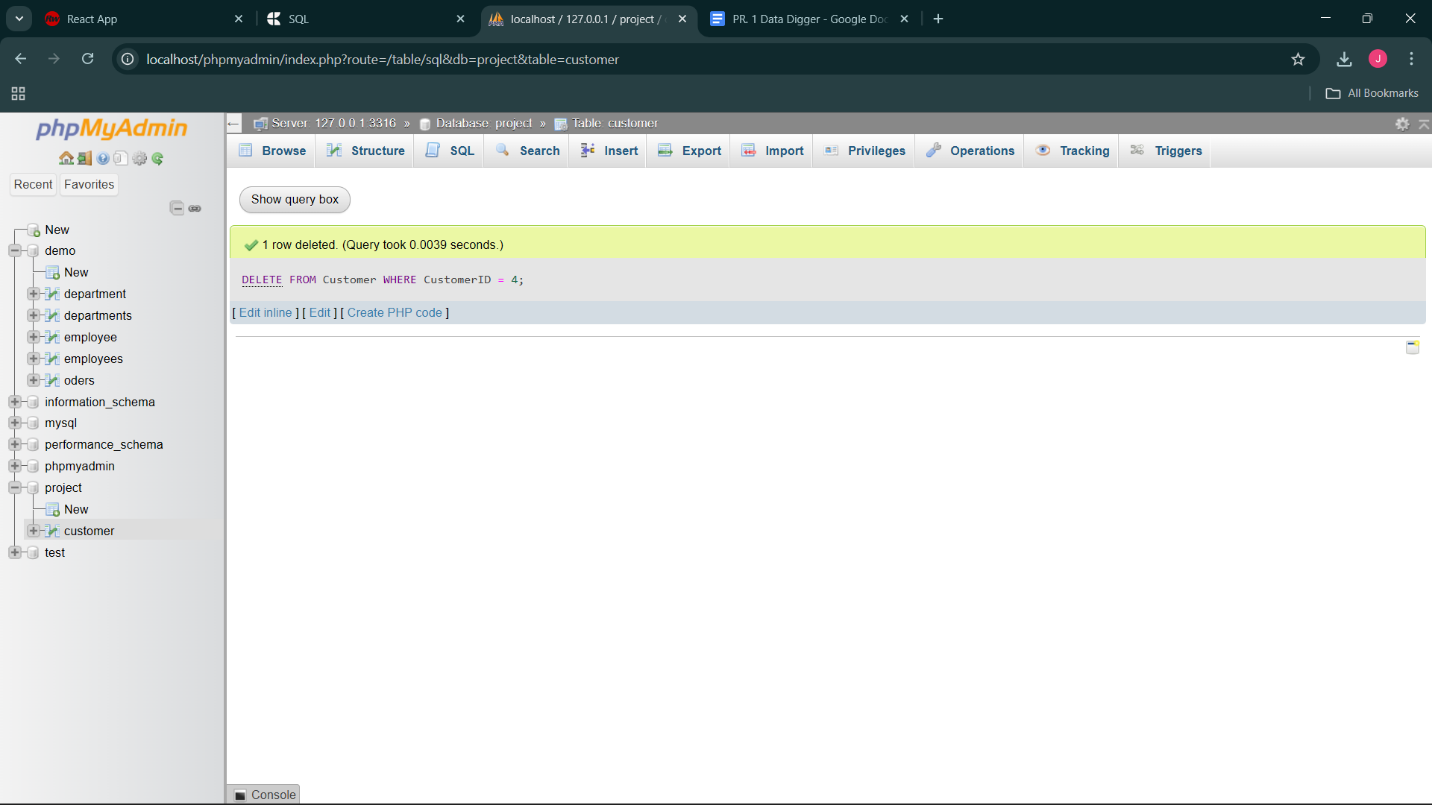
WHERE CustomerID = 3;



4. Delete a customer using their CustomerID

DELETE FROM Customer

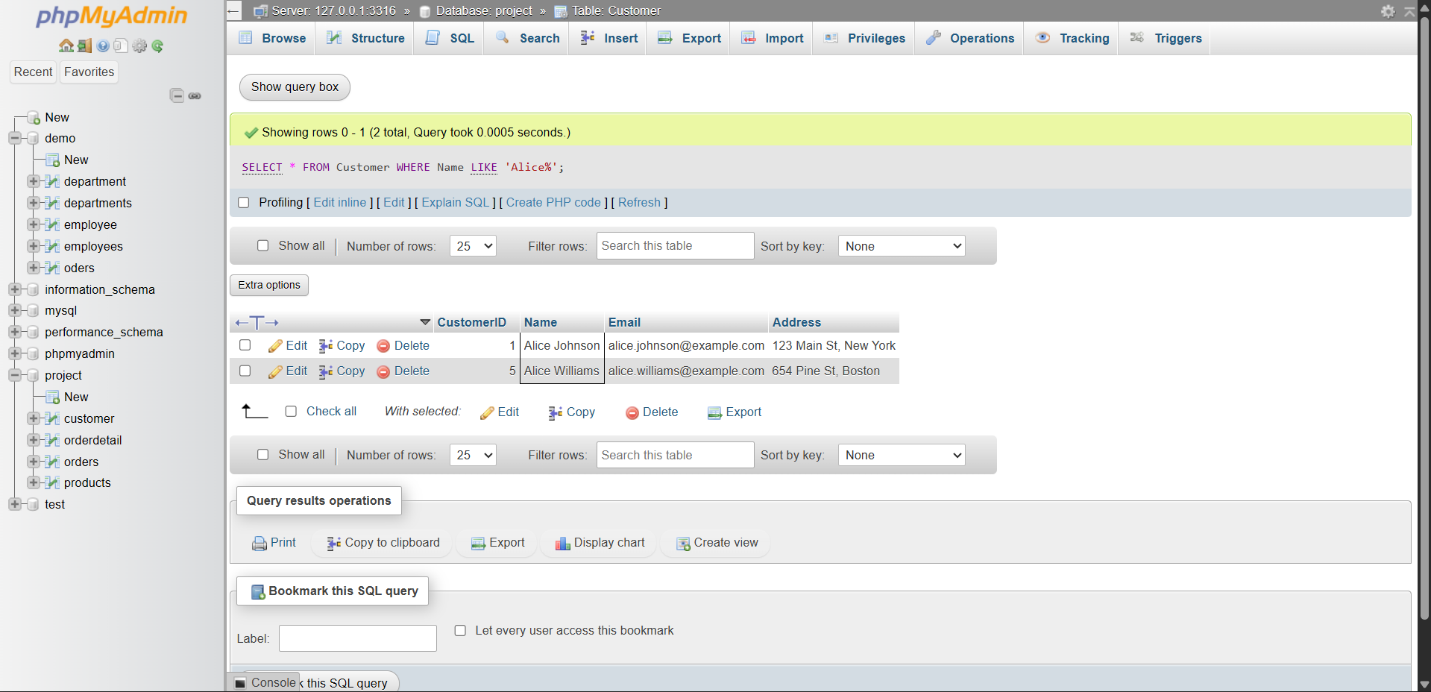
WHERE CustomerID = 4;



5. display all customers whose name is alice

SELECT \* FROM Customer

WHERE Name LIKE 'Alice%';



2. Orders table

create the table

CREATE TABLE Orders (

OrderID INT PRIMARY KEY AUTO\_INCREMENT,

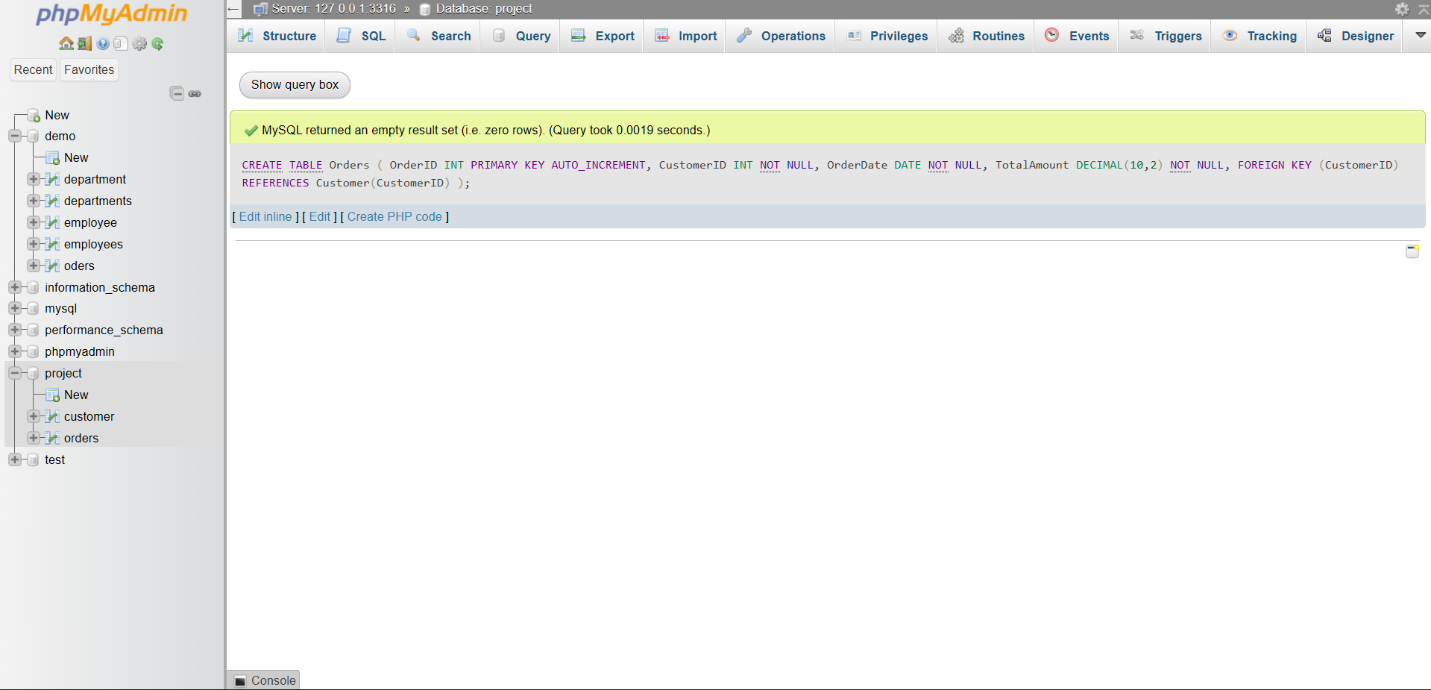
CustomerID INT NOT NULL,

OrderDate DATE NOT NULL,

TotalAmount DECIMAL(10,2) NOT NULL,

FOREIGN KEY (CustomerID) REFERENCES Customer(CustomerID)

);



1. Insert 5 sample orders

INSERT INTO Orders (CustomerID, OrderDate, TotalAmount) VALUES

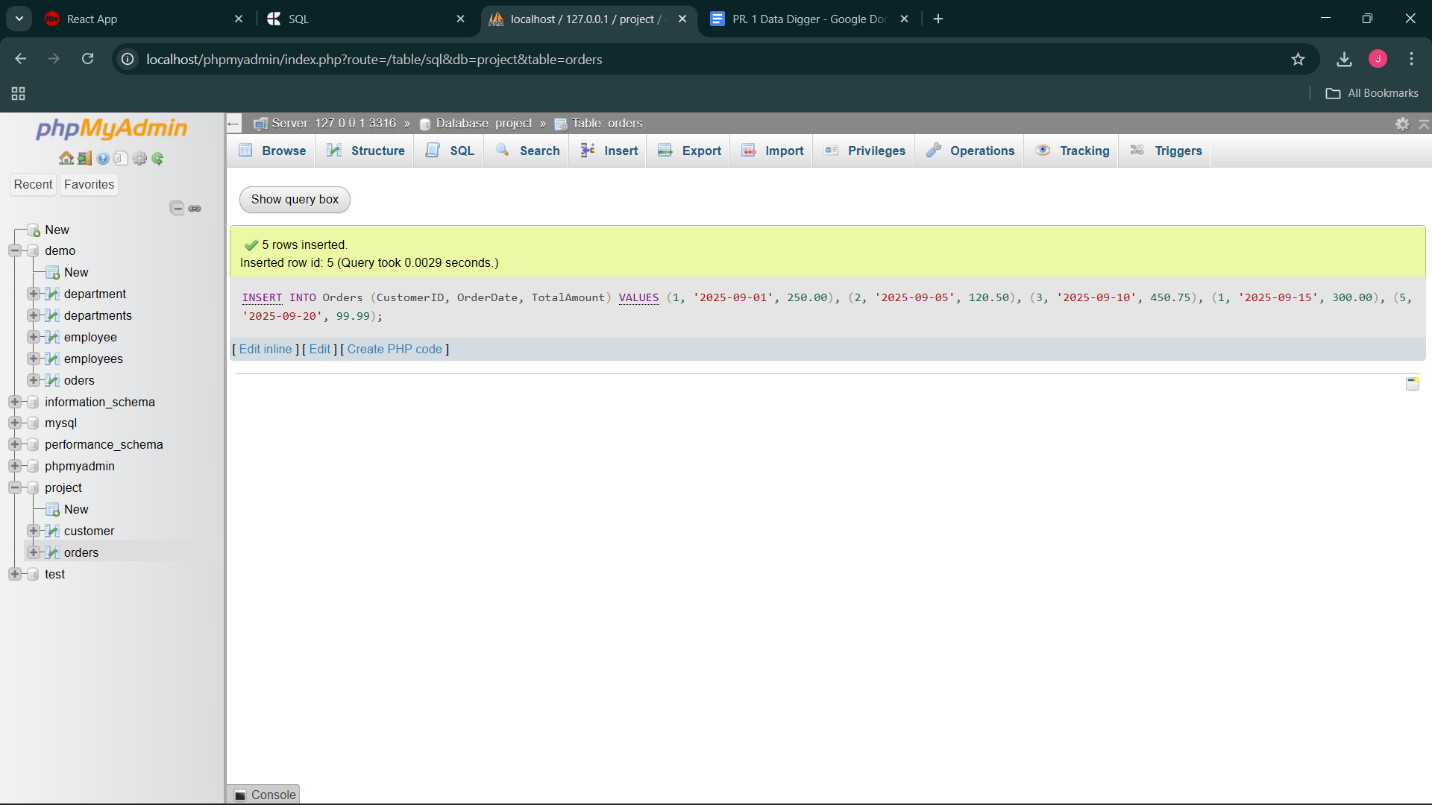
(1, '2025-09-01', 250.00),

(2, '2025-09-05', 120.50),

(3, '2025-09-10', 450.75),

(1, '2025-09-15', 300.00),

(5, '2025-09-20', 99.99);

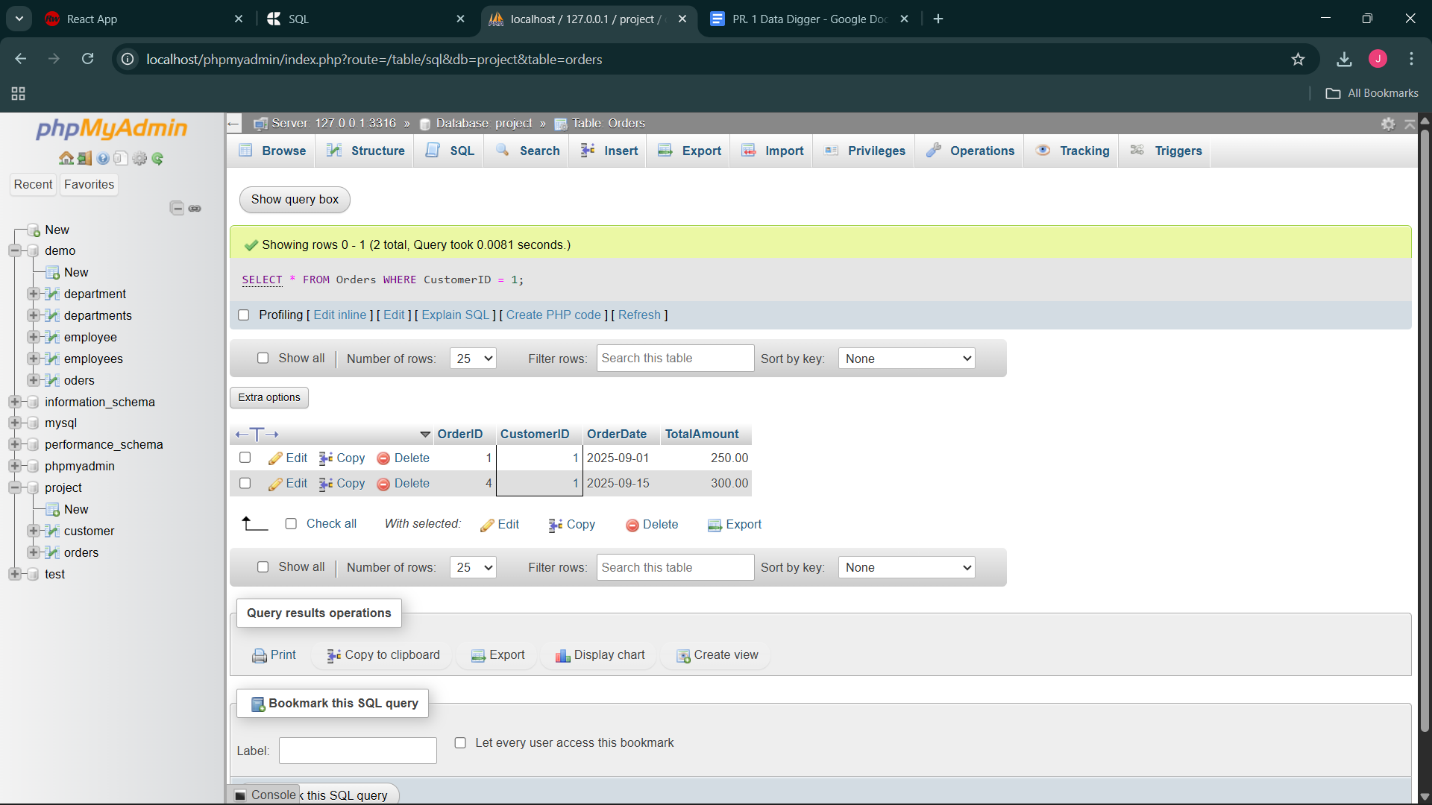


2. Retrieve all orders made by a specific customer

SELECT \*

FROM Orders

WHERE CustomerID = 1;

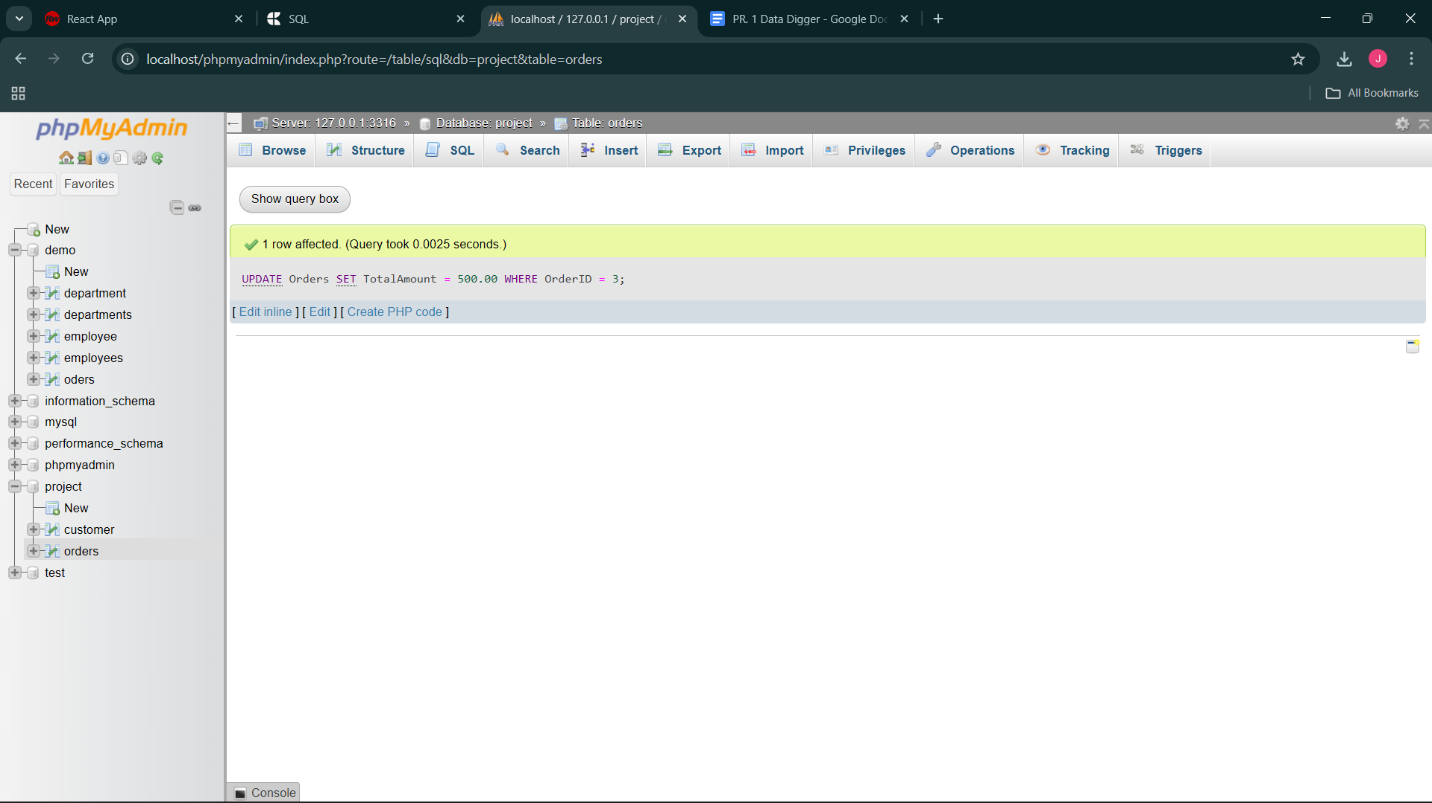


3. Update the order’s total amount

UPDATE Orders

SET TotalAmount = 500.00

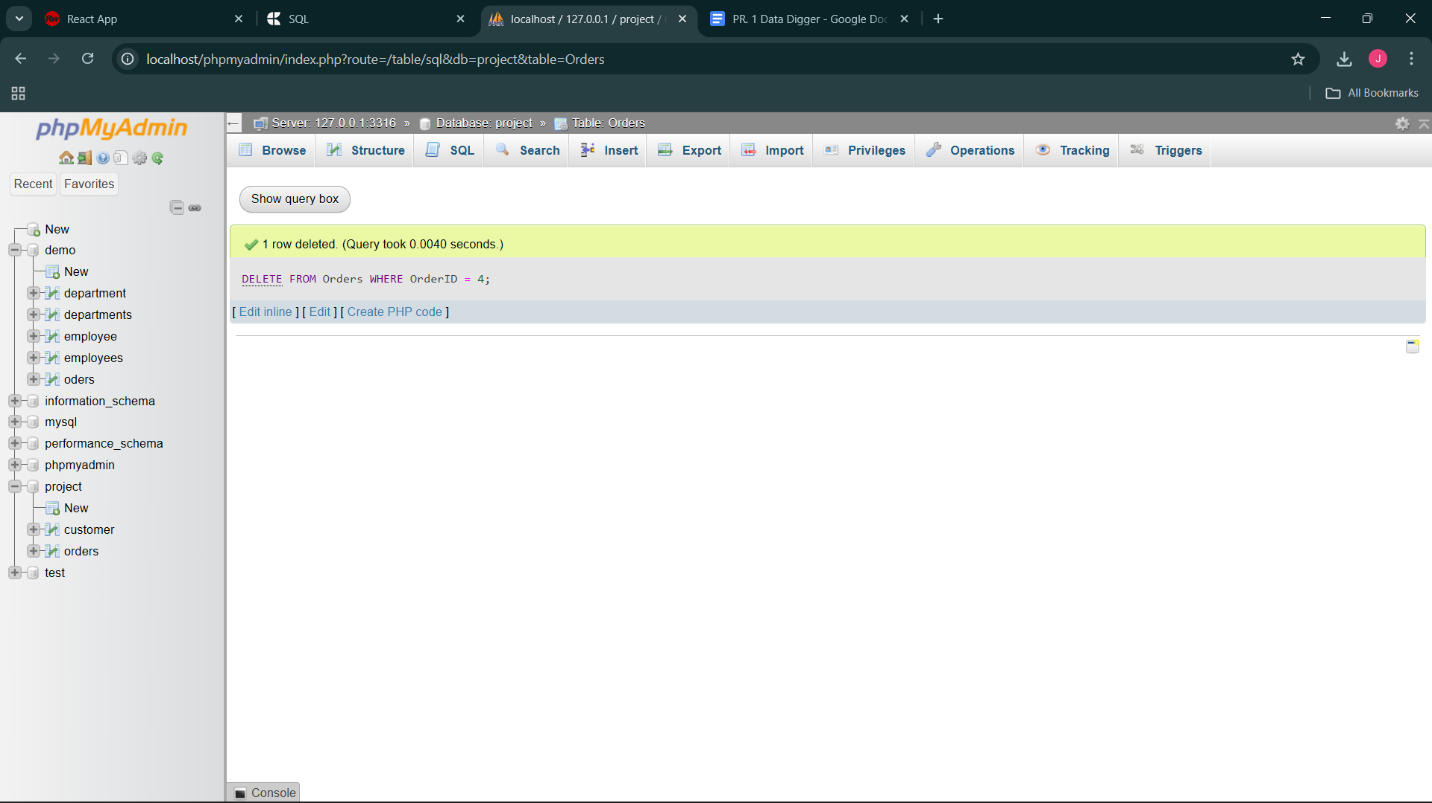
WHERE OrderID = 3;



4. Delete an order using its OrderID

DELETE FROM Orders

WHERE OrderID = 4;

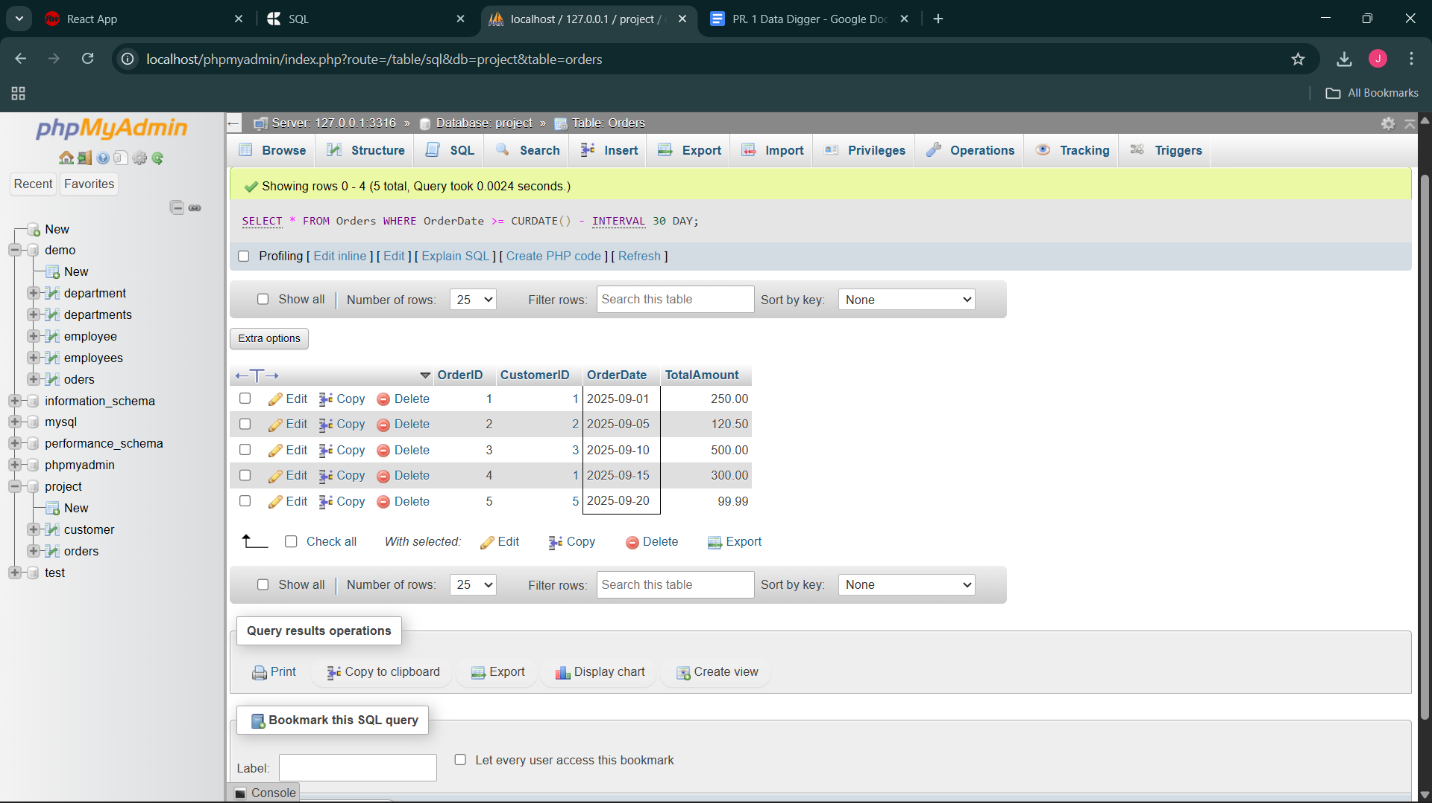


5. Retrieve orders placed in the last 30 days

SELECT \*

FROM Orders

WHERE OrderDate >= CURDATE() - INTERVAL 30 DAY;



6. Retrieve highest, lowest, and average order amount

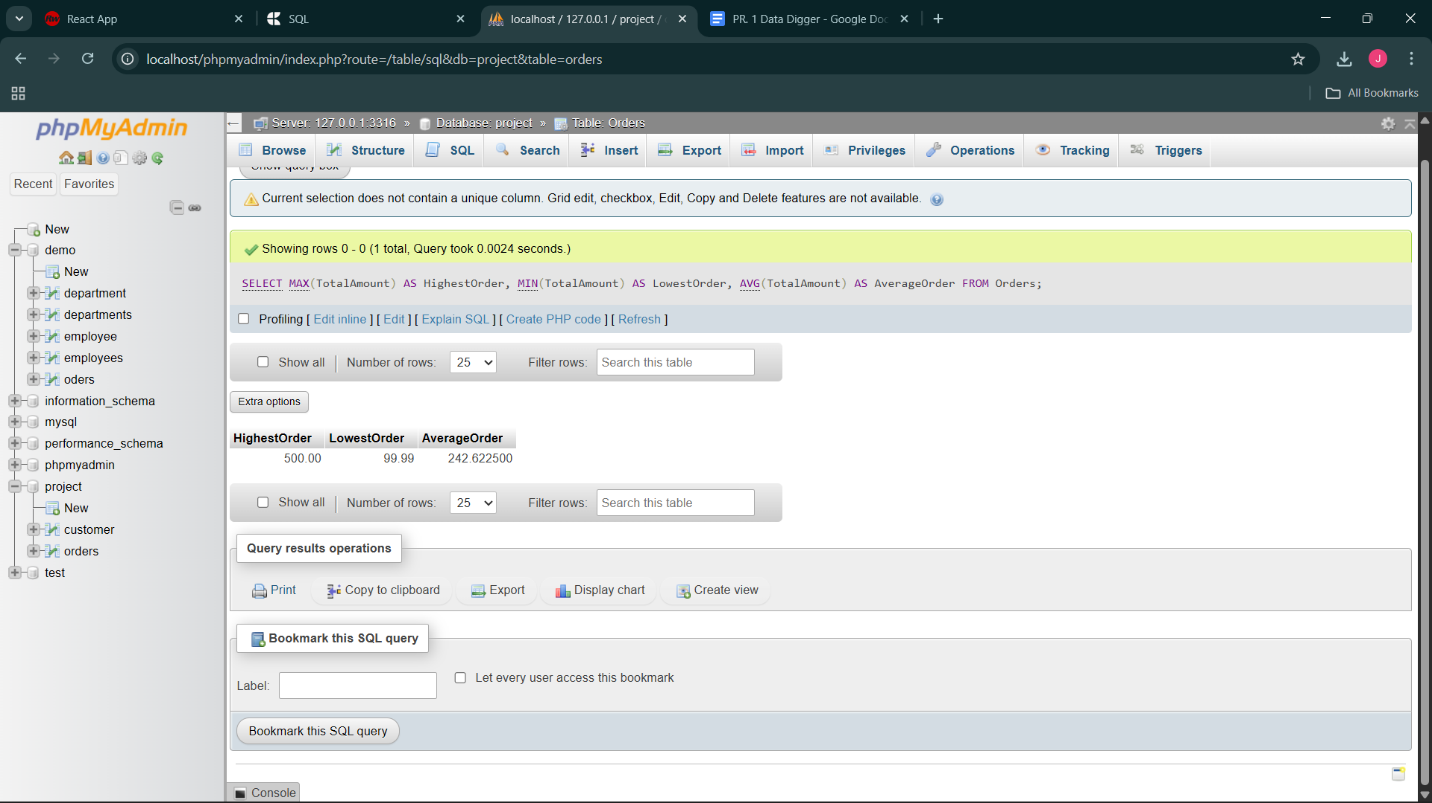
SELECT

MAX(TotalAmount) AS HighestOrder,

MIN(TotalAmount) AS LowestOrder,

AVG(TotalAmount) AS AverageOrder

FROM Orders;



3.Products table

create product table

CREATE TABLE Products (

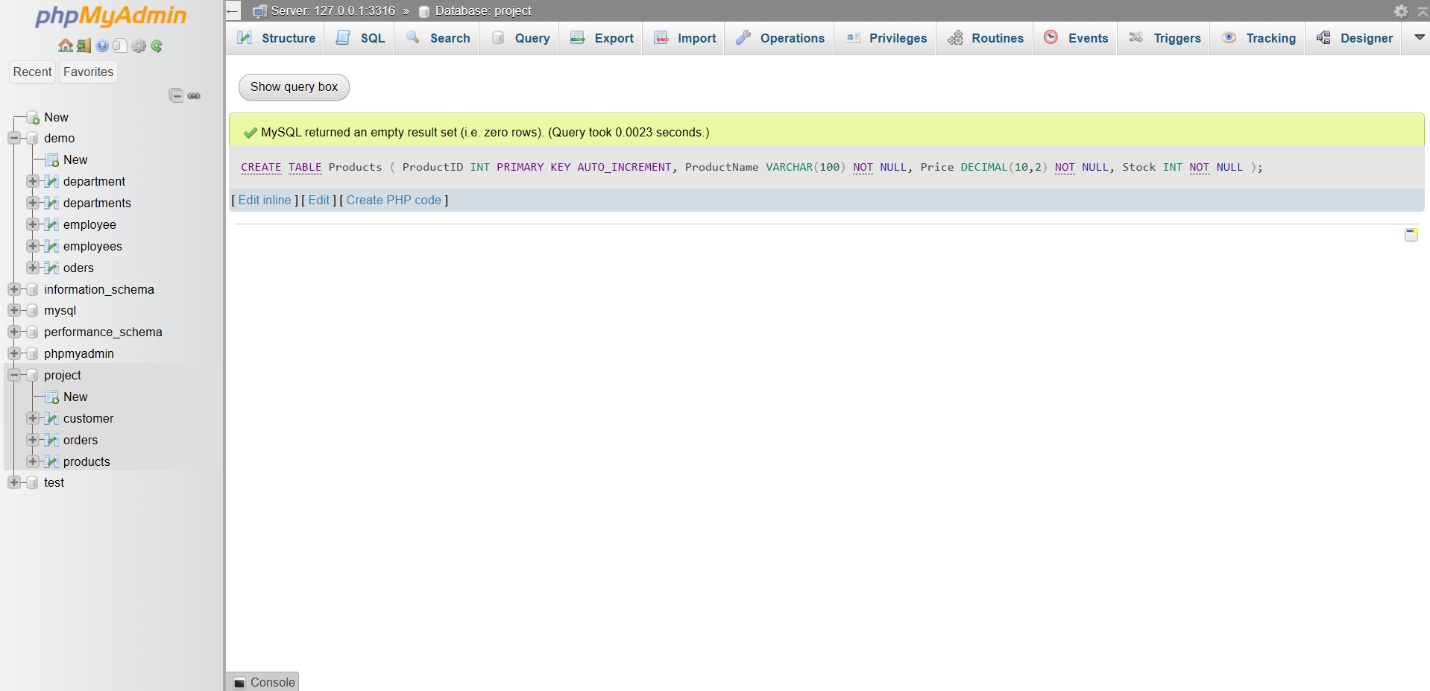
ProductID INT PRIMARY KEY AUTO\_INCREMENT,

ProductName VARCHAR(100) NOT NULL,

Price DECIMAL(10,2) NOT NULL,

Stock INT NOT NULL

);



1. Insert at least 5 sample products

INSERT INTO Products (ProductName, Price, Stock) VALUES

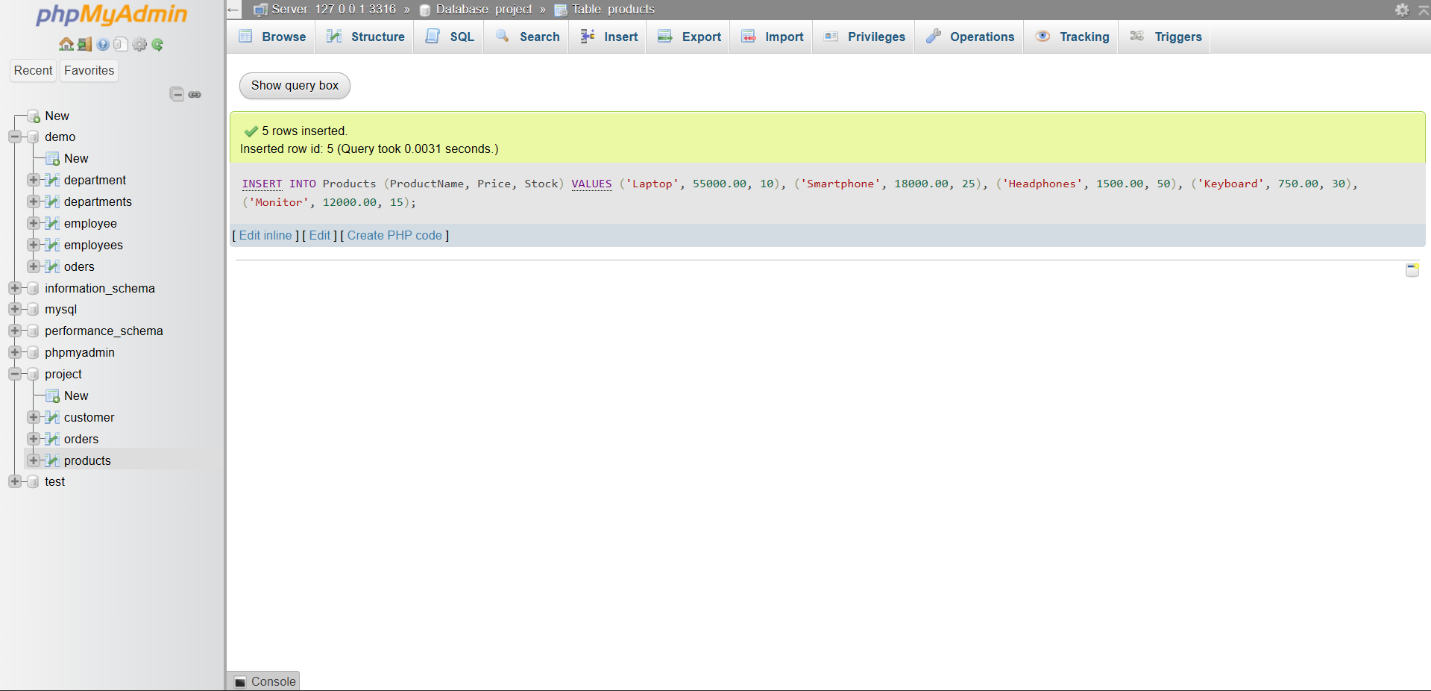
('Laptop', 55000.00, 10),

('Smartphone', 18000.00, 25),

('Headphones', 1500.00, 50),

('Keyboard', 750.00, 30),

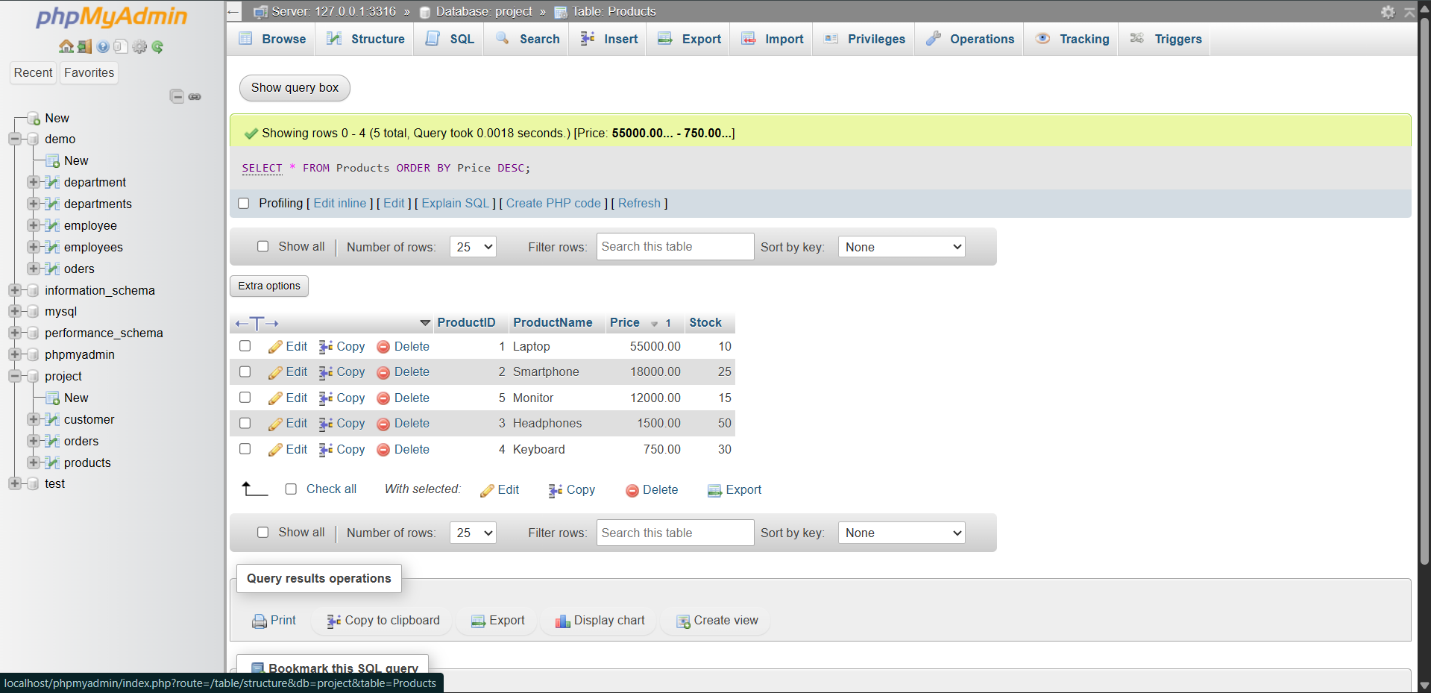
('Monitor', 12000.00, 15);



2. Retrieve all products sorted by price in descending order

SELECT \*

FROM Products

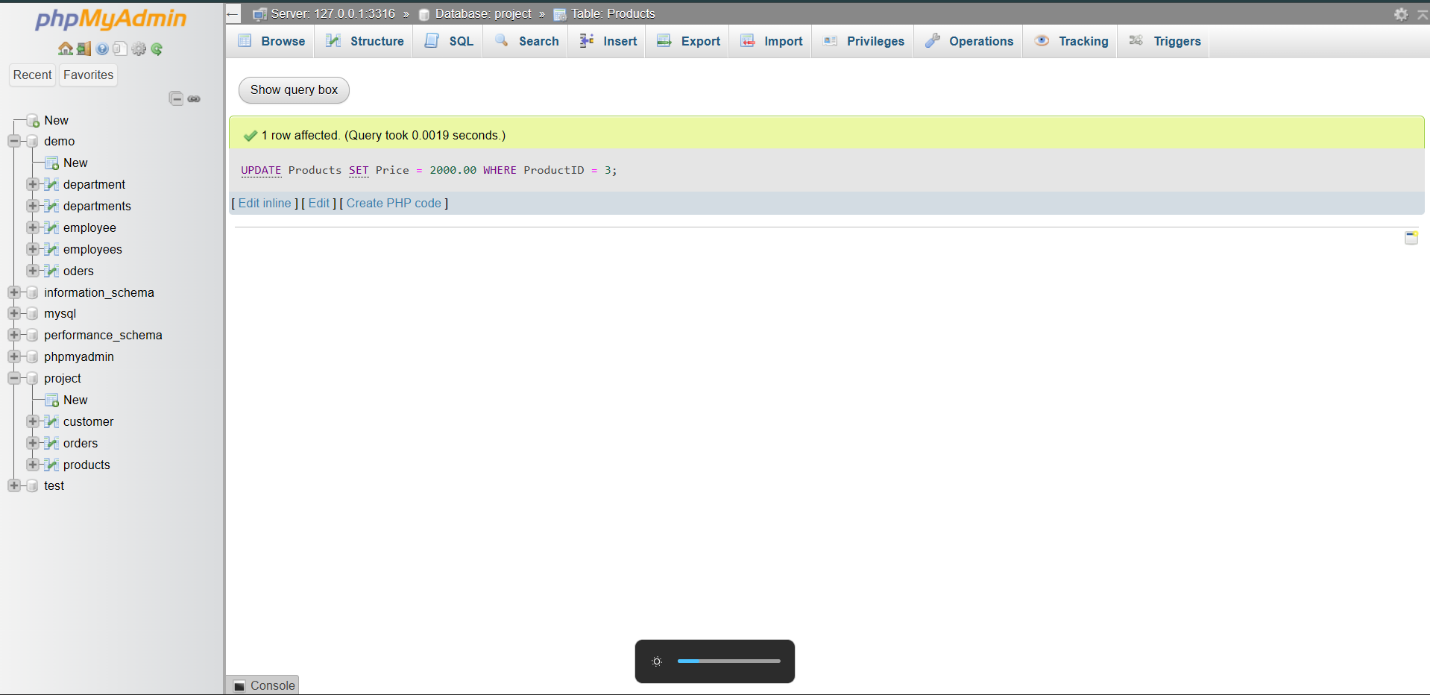
ORDER BY Price DESC;

3. Update the price of a specific product

UPDATE Products

SET Price = 2000.00

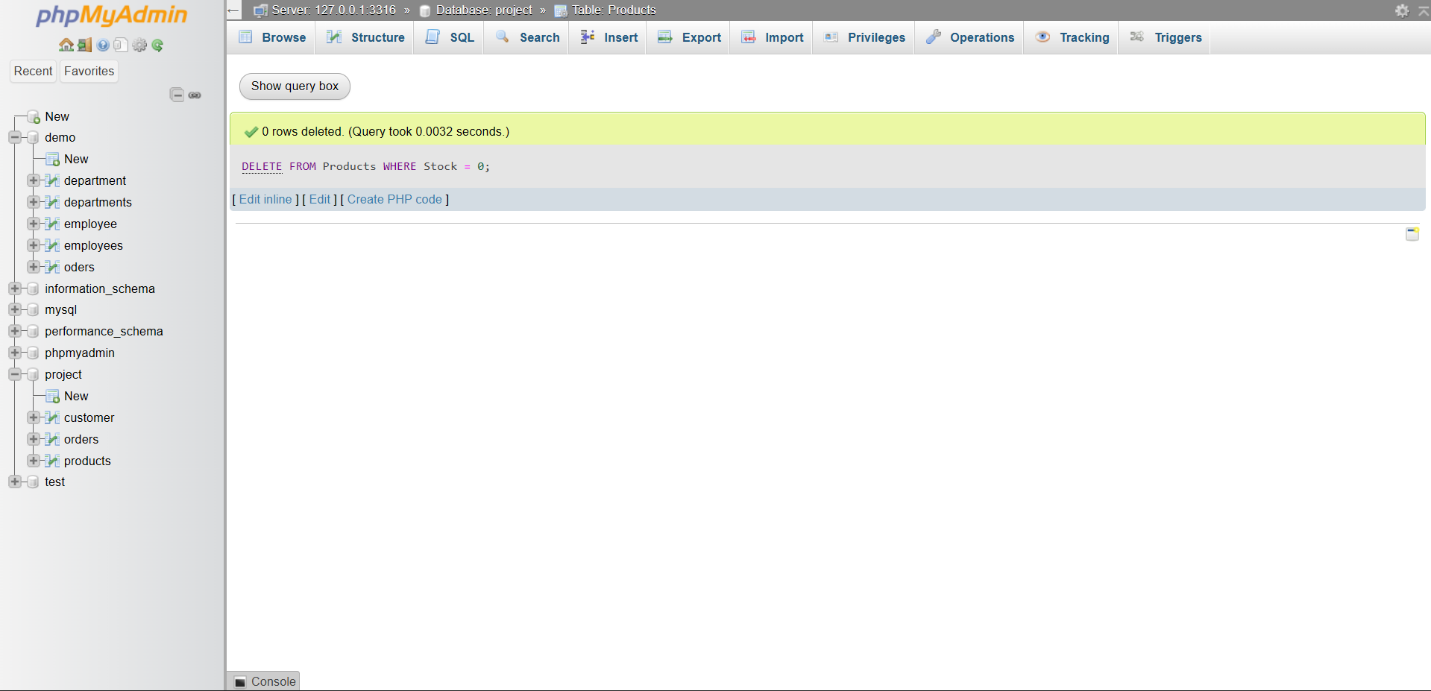
WHERE ProductID = 3;



4. Delete a product if it’s out of stock

DELETE FROM Products

WHERE Stock = 0;

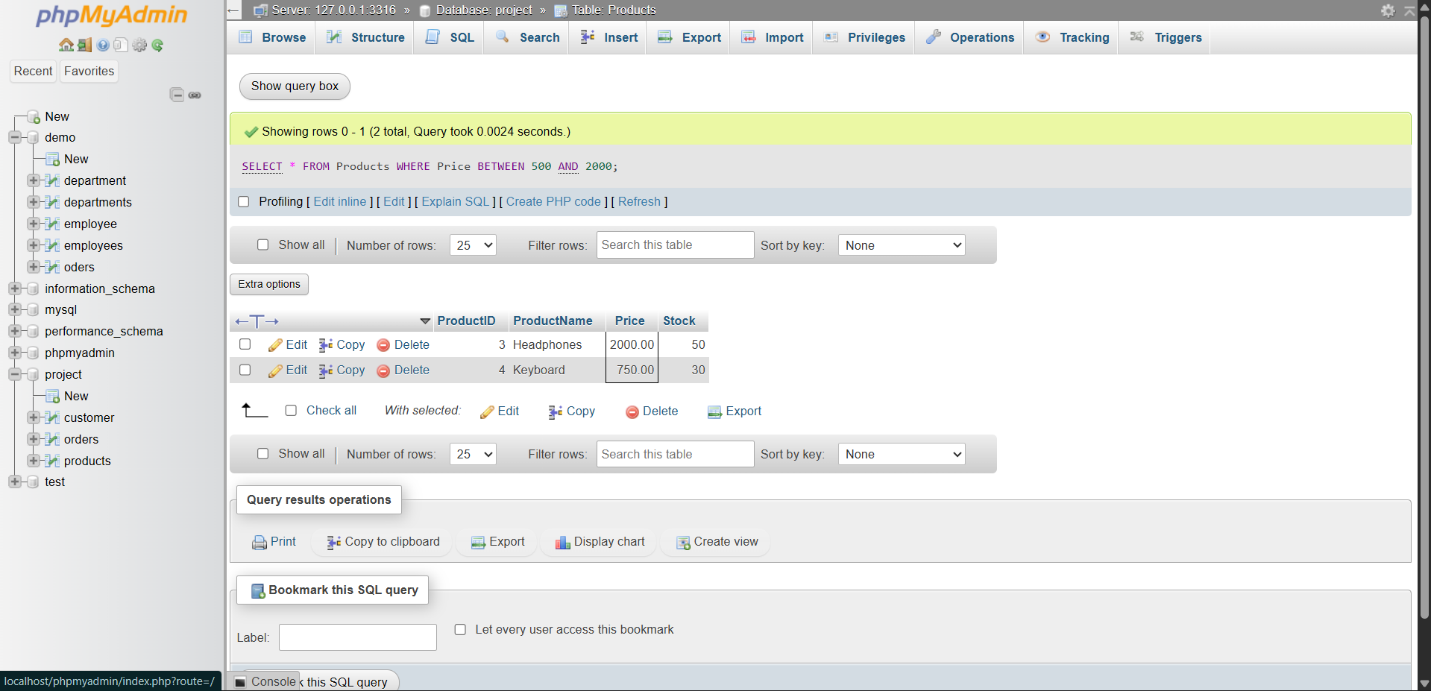


5. Retrieve products whose price is between ₹500 and ₹2000

SELECT \*

FROM Products

WHERE Price BETWEEN 500 AND 2000;



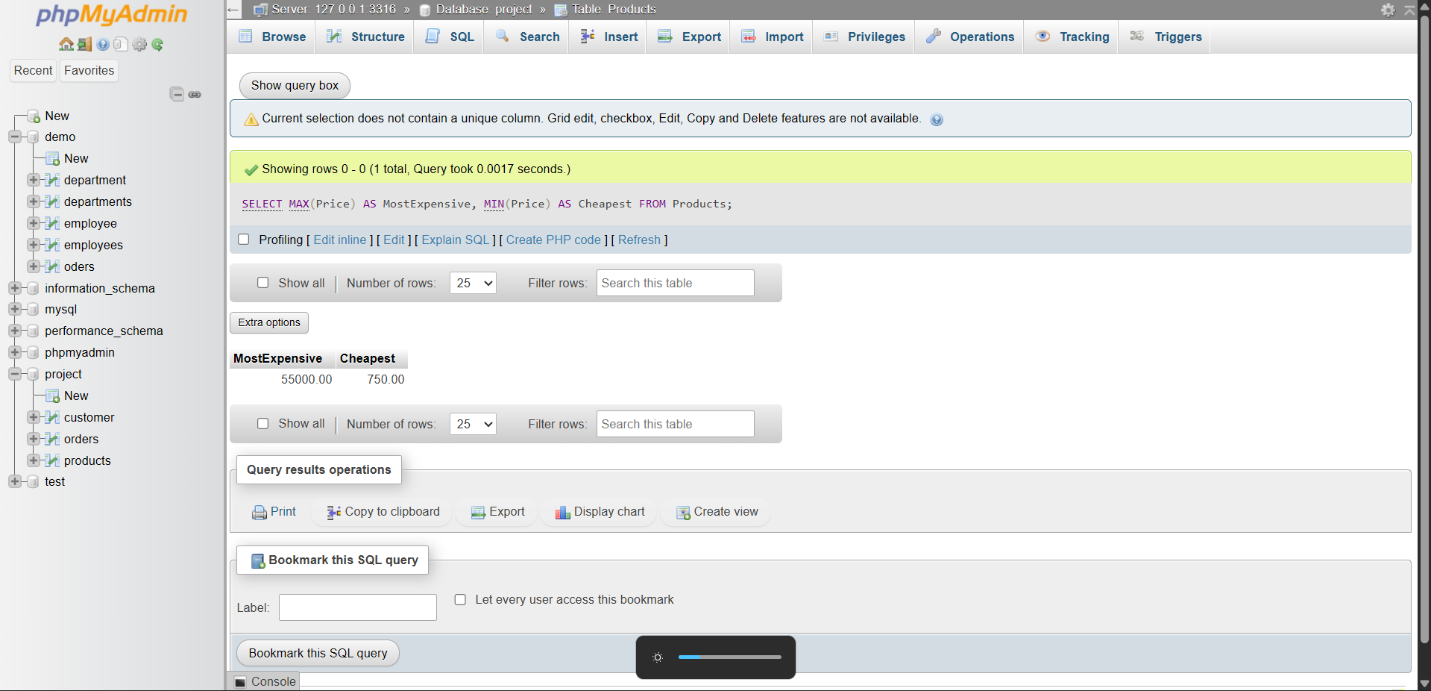
6. Retrieve the most expensive and cheapest product using MAX() and MIN()

SELECT

MAX(Price) AS MostExpensive,

MIN(Price) AS Cheapest

FROM Products;



4.Order details table

create the order detail table

CREATE TABLE OrderDetail (

OrderDetailID INT PRIMARY KEY AUTO\_INCREMENT,

OrderID INT NOT NULL,

ProductID INT NOT NULL,

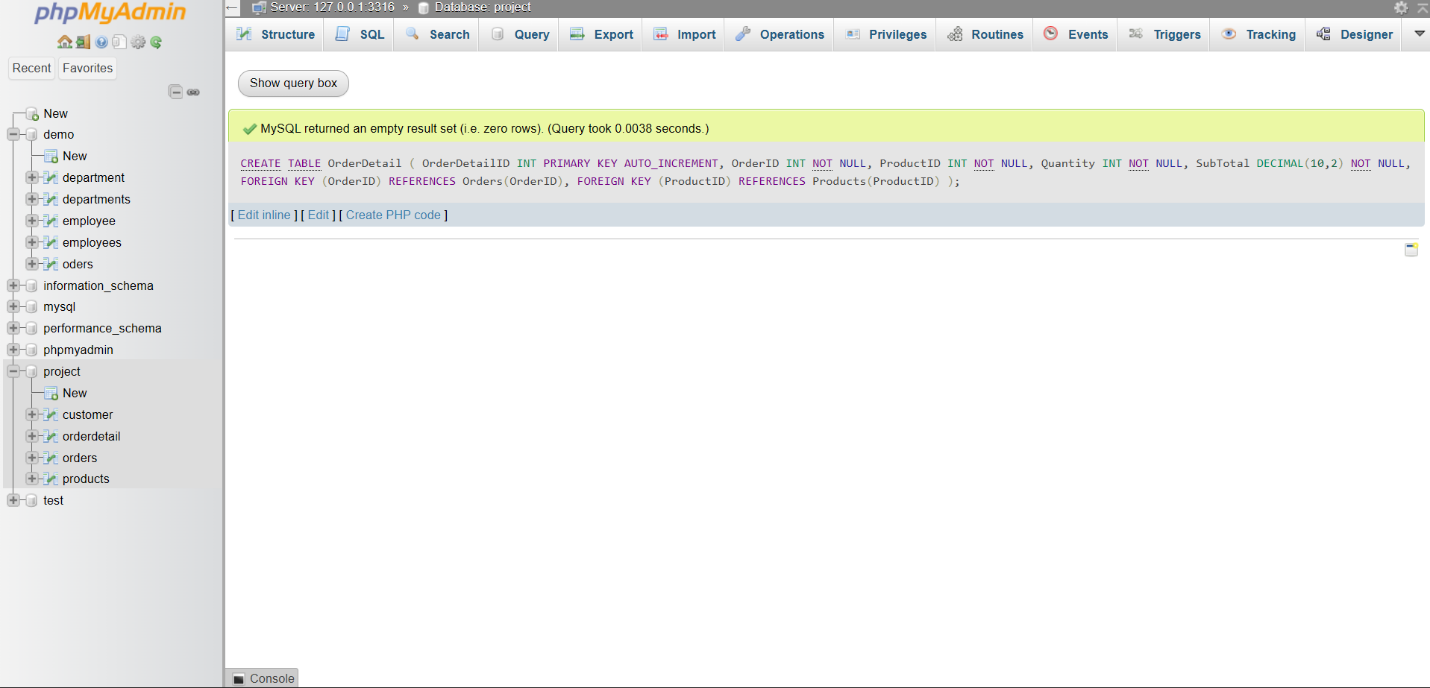
Quantity INT NOT NULL,

SubTotal DECIMAL(10,2) NOT NULL,

FOREIGN KEY (OrderID) REFERENCES Orders(OrderID),

FOREIGN KEY (ProductID) REFERENCES Products(ProductID)

);



1. Insert at least 5 sample records into OrderDetail

INSERT INTO OrderDetail (OrderID, ProductID, Quantity, SubTotal) VALUES

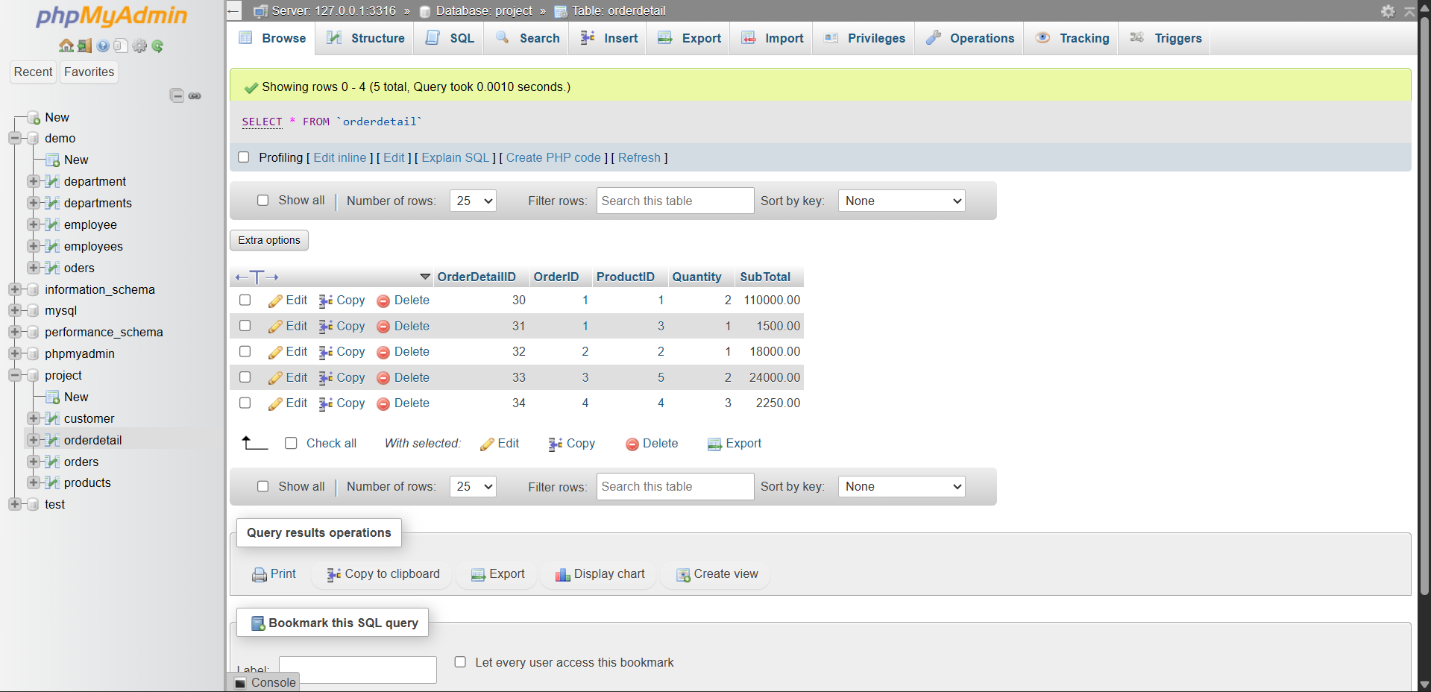
(1, 1, 2, 110000.00),

(1, 3, 1, 1500.00),

(2, 2, 1, 18000.00),

(3, 5, 2, 24000.00),

(4, 4, 3, 2250.00);

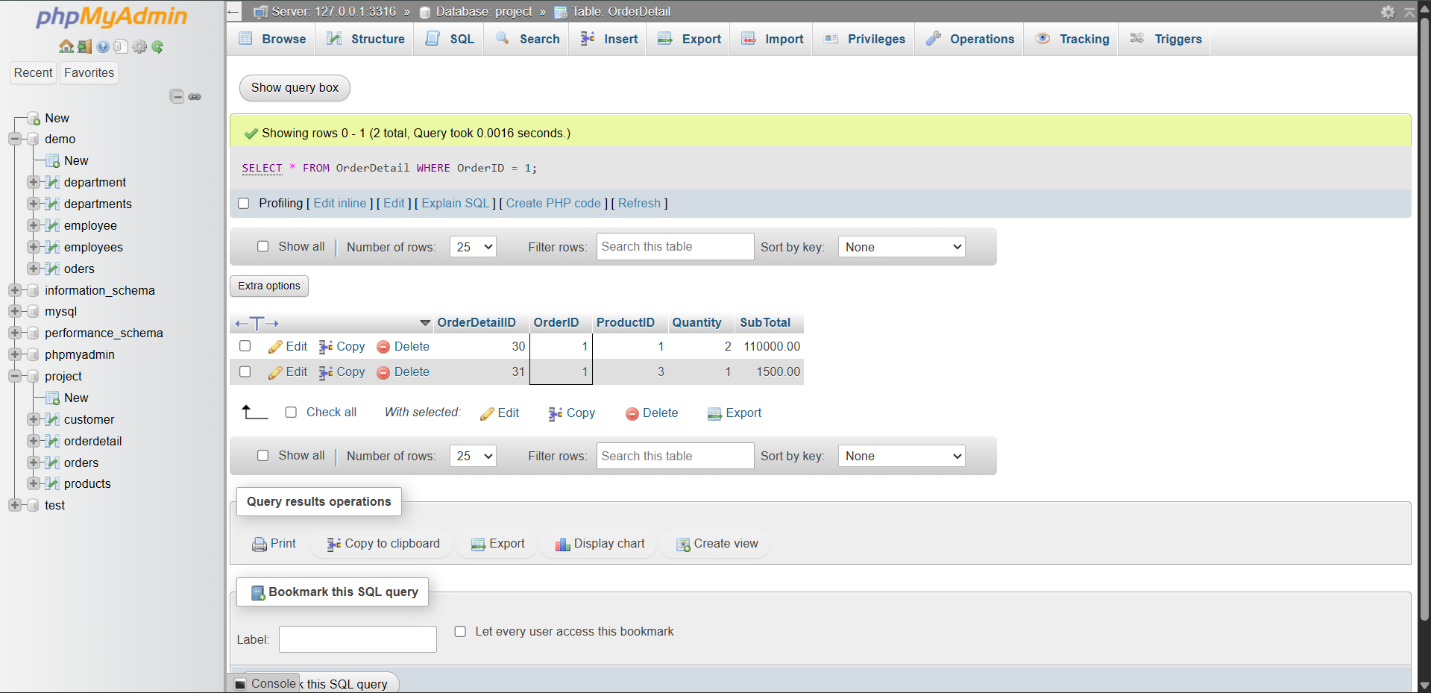


2. Retrieve all order details for a specific order

SELECT \*

FROM OrderDetail

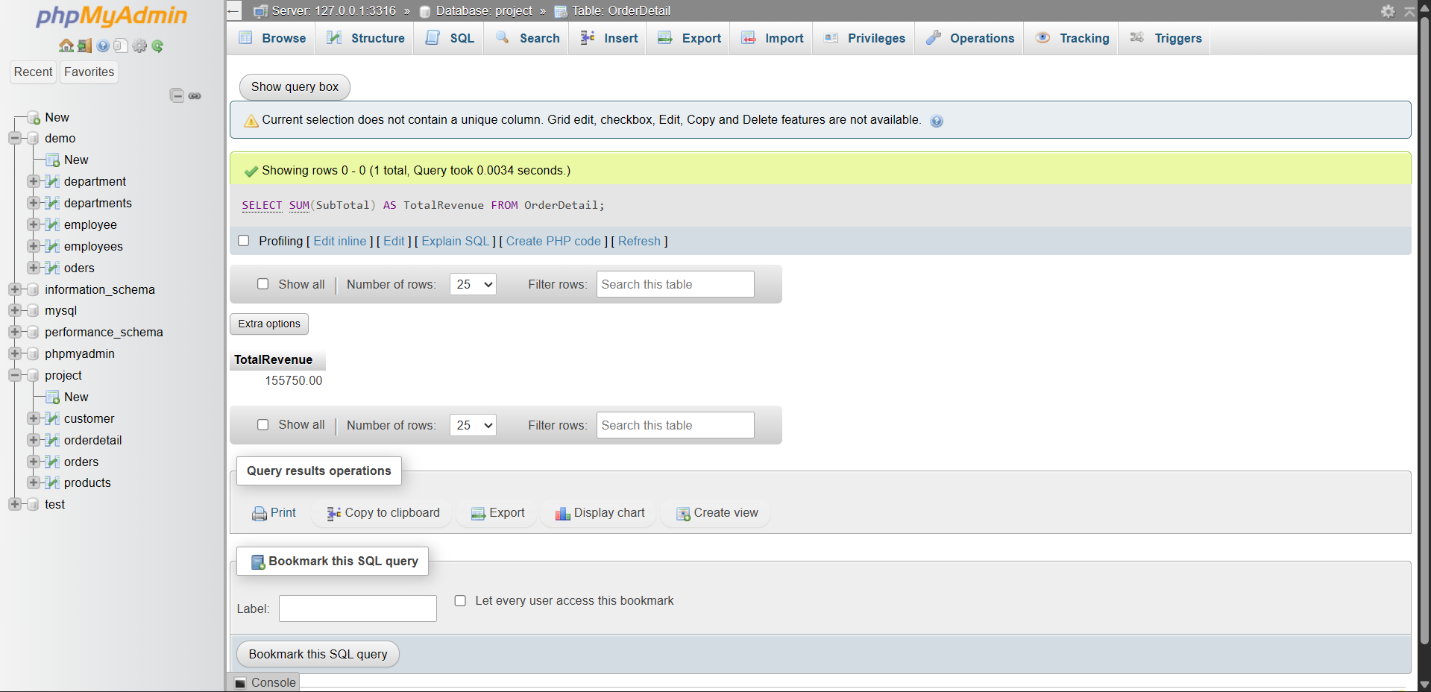
WHERE OrderID = 1;



3. Calculate the total revenue generated from all orders using SUM()

SELECT SUM(SubTotal) AS TotalRevenue

FROM OrderDetail;



4. Retrieve the top 3 most ordered products

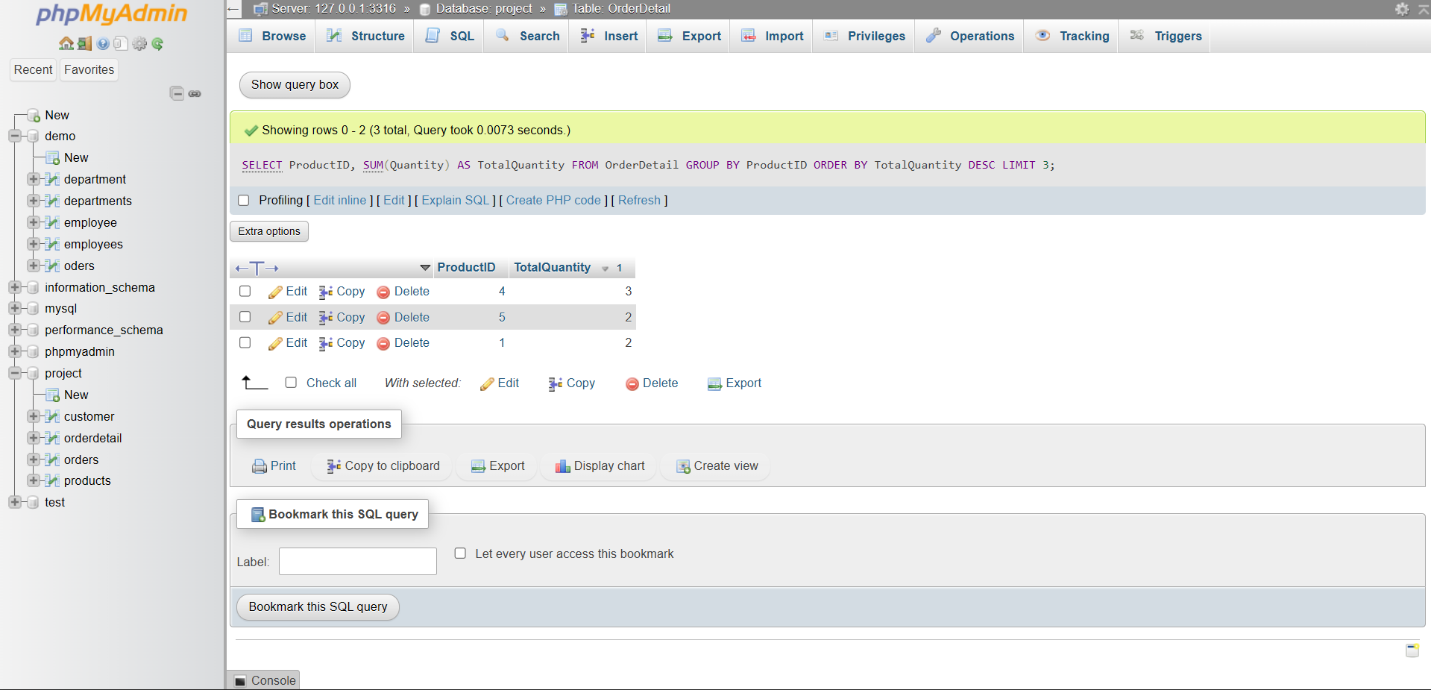
SELECT ProductID, SUM(Quantity) AS TotalQuantity

FROM OrderDetail

GROUP BY ProductID

ORDER BY TotalQuantity DESC

LIMIT 3;



5. Count how many times a specific product has been sold

SELECT COUNT(\*) AS TimesSold

FROM OrderDetail

WHERE ProductID = 1;

