use SQLPractice24th

-- Task: Join the `Orders` and `Customers` tables to find the total order amount per customer and

-- filter out customers who have spent less than $1,000.

Select \* from Orders

Select \* from Customers

Select c.FirstName FROM Customers c join Orders o

on c.CustomerID = o.CustomerID GROUP BY c.FirstName HAVING

SUM(o.TotalAmount) < 1000

-- Task: Create a cumulative sum of the `OrderAmount` for each customer to track

-- the running total of how much each customer has spent.

SELECT CustomerID, TotalAmount, SUM(TotalAmount) OVER

(PARTITION BY CustomerID ORDER BY TotalAmount DESC) AS RunningTotal,

RANK() OVER (ORDER BY TotalAmount DESC) as RankNo

FROM Orders

-- - Task: Rank the customers based on the total amount they have spent, partitioned by city.

CREATE TABLE Customers2 (

CustomerID INT PRIMARY KEY IDENTITY(1,1),

FirstName VARCHAR(50),

LastName VARCHAR(50),

Email VARCHAR(100),

PhoneNumber VARCHAR(15),

City VARCHAR(50)

);

INSERT INTO Customers2 (FirstName, LastName, Email, PhoneNumber, City)

VALUES

('Amit', 'Sharma', 'amit.sharma@example.com', '9876543210','Chennai'),

('Priya', 'Mehta', 'priya.mehta@example.com', '8765432109','Delhi'),

('Rohit', 'Kumar', 'rohit.kumar@example.com', '7654321098','Bangalore'),

('Neha', 'Verma', 'neha.verma@example.com', '6543210987','Chennai'),

('Siddharth', 'Singh', 'siddharth.singh@example.com', '5432109876','Hyderbad'),

('Asha', 'Rao', 'asha.rao@example.com', '4321098765','Pune');

SELECT c.FirstName, c.City, SUM(o.TotalAmount) as TotalSpent,

RANK() OVER (PARTITION BY c.City ORDER BY sum(o.TotalAmount) DESC) as RankNo

from Customers2 c JOIN Orders o on c.CustomerID = o.CustomerID

GROUP BY c.FirstName, c.City ORDER BY c.City

-- Task: Calculate the total amount of all orders (overall total) and the percentage each customer's

-- total spending contributes to the overall total.

WITH OverallTotal AS (

SELECT SUM(o.TotalAmount) AS OverallTotal FROM Orders o

)

SELECT c.CustomerID, c.FirstName,SUM(o.TotalAmount) AS CustomerTotalSpent,

ot.OverallTotal,(SUM(o.TotalAmount) \* 100.0) / ot.OverallTotal AS PercentageContribution

FROM Customers c

JOIN Orders o ON c.CustomerID = o.CustomerID

CROSS JOIN OverallTotal ot

GROUP BY c.CustomerID, c.FirstName, ot.OverallTotal

ORDER BY PercentageContribution DESC;

-- Rank all customers based on the total amount they have spent, without partitioning.

SELECT c.FirstName, c.LastName, SUM(o.TotalAmount) as TotalAmount,

RANK() OVER (ORDER BY SUM(o.TotalAmount) DESC) as RankNo from Customers2 c JOIN

Orders o on c.CustomerID = o.CustomerID GROUP BY c.FirstName, c.LastName

-- Task: Write a query that joins the `Orders` and `Customers` tables, calculates the average order amount for each city, and orders

-- the results by the average amount in descending order.

SELECT c.City, AVG(o.TotalAmount) as AverageAmount from Orders o JOIN Customers2 c

on o.CustomerID = c.CustomerID GROUP BY c.City ORDER BY AverageAmount DESC

-- Task: Write a query to find the top 3 customers who have spent the most, using `ORDER BY` and `LIMIT`.

SELECT c.FirstName, SUM(o.TotalAmount) from Customers2 c JOIN Orders o ON

c.CustomerID = o.CustomerID

GROUP BY c.FirstName ORDER BY SUM(o.TotalAmount) DESC OFFSET 0 ROWS FETCH NEXT 3 ROWS ONLY

-- - Task: Write a query that groups orders by year (using `OrderDate`), calculates the total amount of orders for each year, and orders the results by year.

SELECT YEAR(o.OrderDate) AS OrderYear, SUM(o.TotalAmount) AS TotalOrderAmount

FROM Orders o GROUP BY YEAR(o.OrderDate)

ORDER BY OrderYear;

-- Task: Write a query that ranks customers by their total spending, but only for customers located in "Mumbai". The rank should reset for each customer in "Mumbai".

SELECT c.FirstName, c.City, SUM(o.TotalAmount) as TotalSpent,

RANK() OVER (ORDER BY SUM(o.TotalAmount) DESC) as RankNo FROM Customers2 c

JOIN Orders o ON c.CustomerID = o.CustomerID WHERE c.City = 'Mumbai'

GROUP BY c.FirstName, c.City ORDER BY RankNo

-- Task: Write a query that calculates each customer's total order amount and compares it to the average order amount for all customers.

SELECT c.CustomerID,c.FirstName,SUM(o.TotalAmount) AS CustomerTotalSpent,

AVG(SUM(o.TotalAmount)) OVER () AS AverageOrderAmount

FROM Customers c JOIN Orders o ON c.CustomerID = o.CustomerID

GROUP BY c.CustomerID, c.FirstName ORDER BY CustomerTotalSpent DESC;