Jesse Ewang

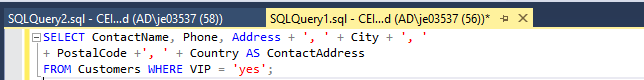
Questions

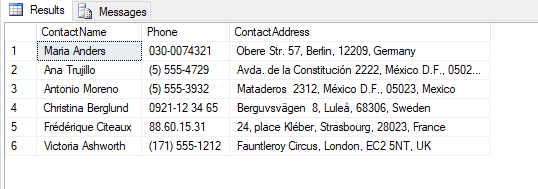
1. Write a SELECT statement that lists out the FullName, Phone and Address [Street,City, Postal Code and Country) as contact information for all customers that are VIPs.

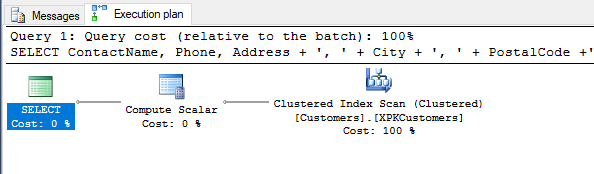
SELECT ContactName, Phone, Address + ', ' + City + ', '

+ PostalCode +', ' + Country AS ContactAddress

FROM Customers WHERE VIP = 'yes';







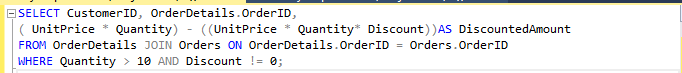
1. Write a SELECT to calculate the discount given to customers who have ordered above 10 items with discount that is not zero and also return the customer associated the order, as well as the orderID.

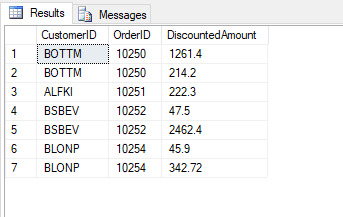
SELECT CustomerID, OrderDetails.OrderID,

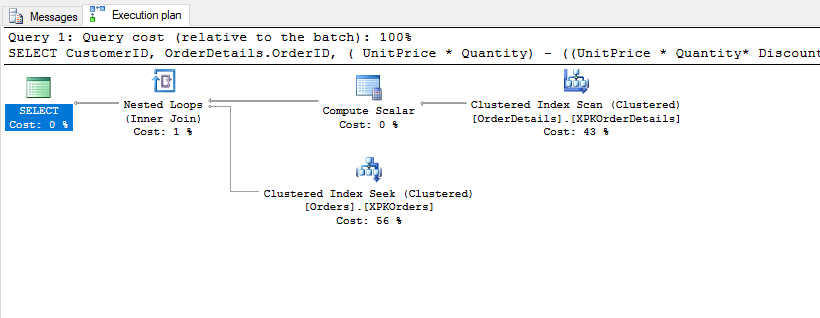
( UnitPrice \* Quantity) - ((UnitPrice \* Quantity\* Discount))AS DiscountedAmount

FROM OrderDetails JOIN Orders ON OrderDetails.OrderID = Orders.OrderID

WHERE Quantity > 10 AND Discount != 0;







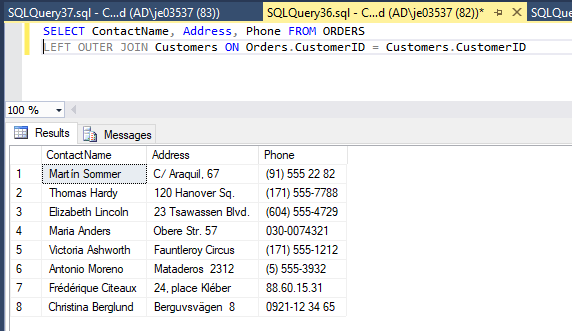
1. Write a SELECT statement that returns a row for each customer that has orders with these columns:

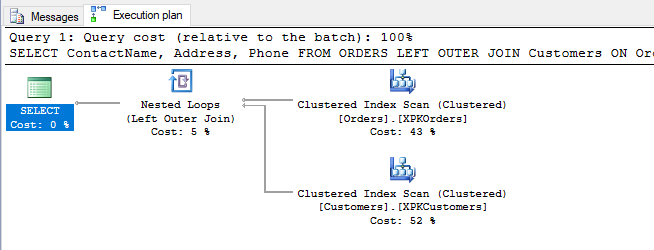
Name

Address

Phone number

SELECT ContactName, Address, Phone FROM ORDERS LEFT OUTER JOIN Customers ON Orders.CustomerID = Customers.CustomerID

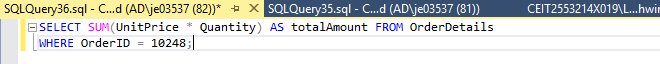




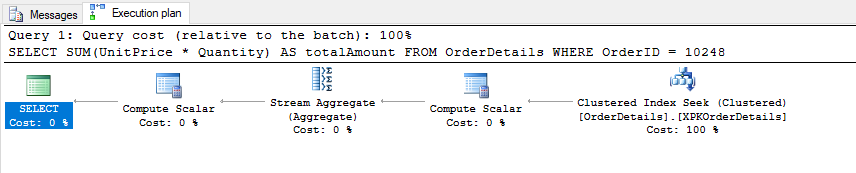
1. Write a SELECT statement that returns the total amount obtained for a single order of ID = 10248

SELECT SUM(UnitPrice \* Quantity) AS totalAmount FROM OrderDetails

WHERE OrderID = 10248;







1. Write a SELECT statement that returns a list of each customer’s information and returns the columns:

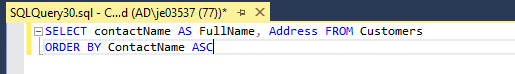
Fullname: which joins LastName and FirstName

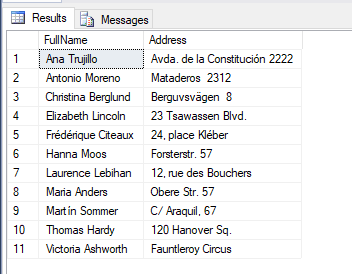
Address

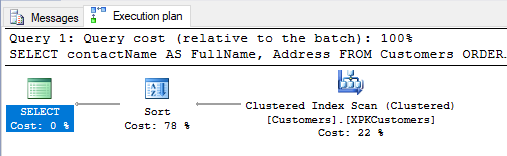
And sorts them in ascending sequence (A-Z)

SELECT contactName AS FullName, Address FROM Customers

ORDER BY ContactName ASC







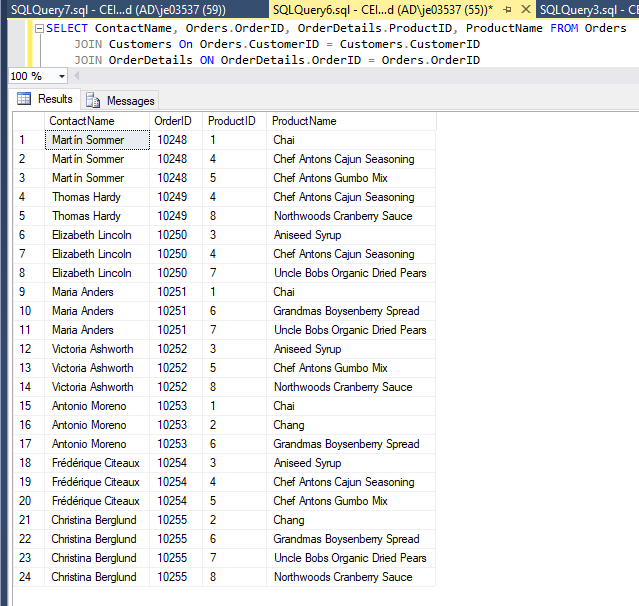
1. Write a SELECT statement that joins the customers and orders tables and return these and gives the names of all customers with orders

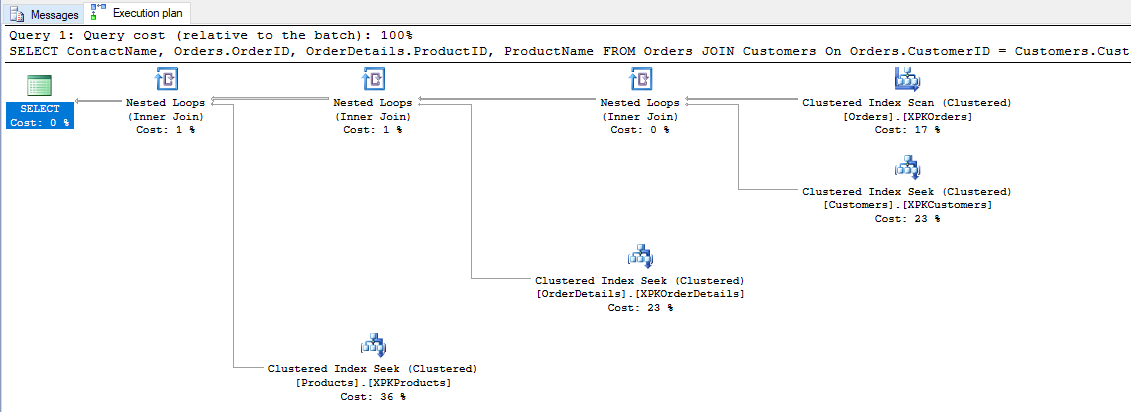
SELECT ContactName, Orders.OrderID, OrderDetails.ProductID, ProductName FROM Orders

JOIN Customers On Orders.CustomerID = Customers.CustomerID

JOIN OrderDetails ON OrderDetails.OrderID = Orders.OrderID

JOIN Products ON Products.ProductID = OrderDetails.ProductID;

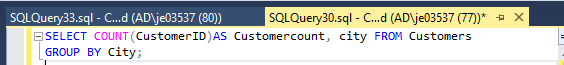


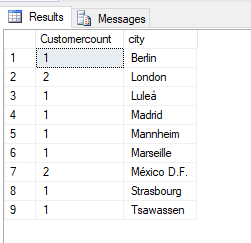


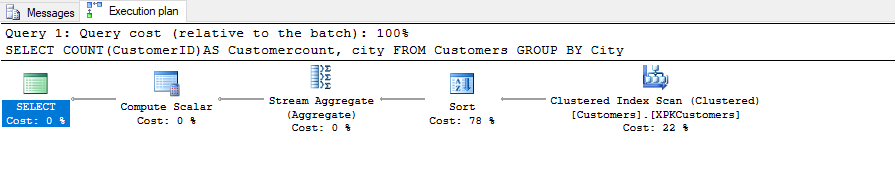
1. A SELECT statement that returns a count of the customers in each city

SELECT COUNT(CustomerID)AS Customercount, city FROM Customers

GROUP BY City;





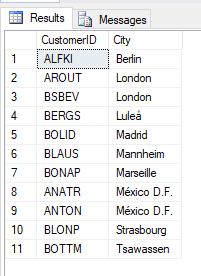


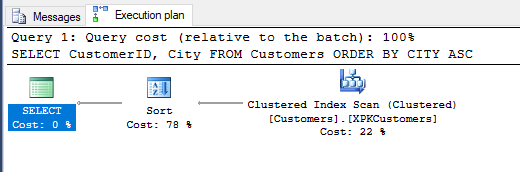
1. Write a SELECT Statement that lists out all the customers them based on their City to create a table for mapping in ascending order.

SELECT CustomerID, City FROM Customers

ORDER BY CITY ASC;

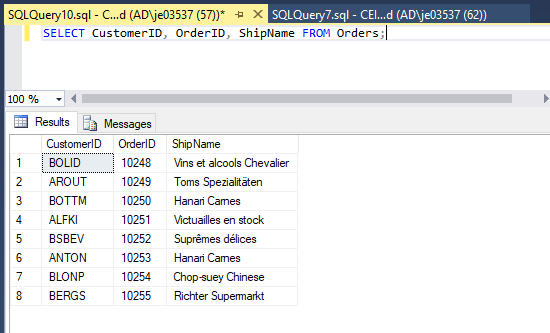


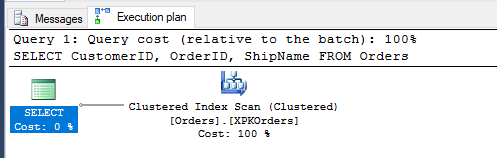




1. Write a SELECT statement that determines what ship was used for each customers order.

SELECT CustomerID, OrderID, ShipName FROM Orders;

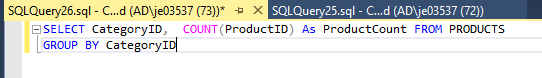


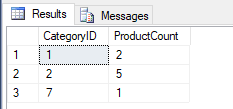


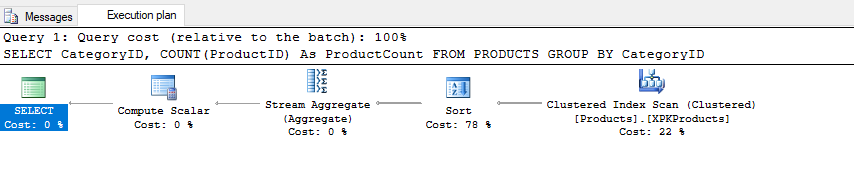
1. Write a SELECT statement that lists out products and groups them into their different categories

SELECT CategoryID, COUNT(ProductID) As ProductCount FROM PRODUCTS

GROUP BY CategoryID







Michael Carter

1. For every product currently on order that we are lower than some threshold on stock, group the

products by supplier with their contact information

select Products.\* from Products,

(Select ProductID from [Order Details],

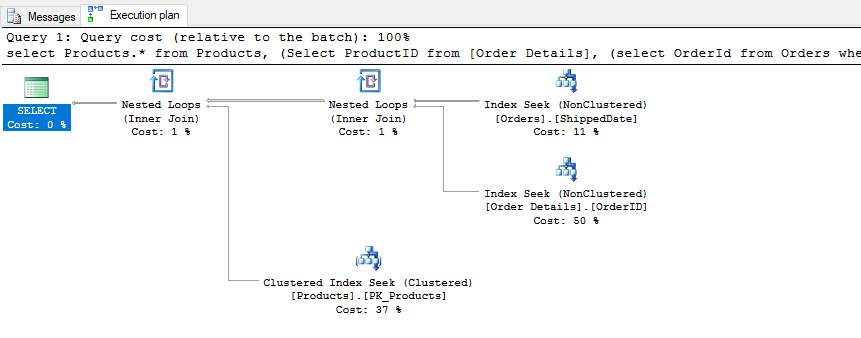
(select OrderId from Orders where shippedDate is null) as ActiveOrders

where [Order Details].OrderID = activeOrders.OrderID) as activeProducts

where Products.ProductID = activeProducts.ProductID

and Products.UnitsInStock+Products.UnitsOnOrder < Products.ReOrderLevel





Justin Frazey

Yellow team group project 10 questions.

Personal Scope: Employees, Employee Territories, Territories, Regions

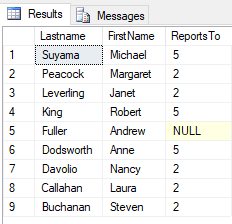
1. A SELECT statement that returns four columns from the employees table: LastName, FirstName,

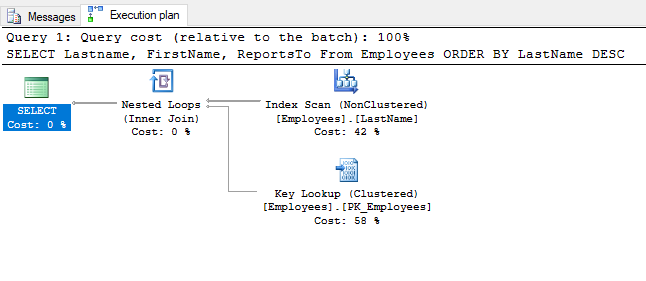
ReportsTo. Order by LastName desc.

SELECT Lastname, FirstName, ReportsTo From Employees

ORDER BY LastName DESC





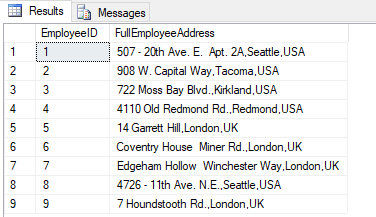


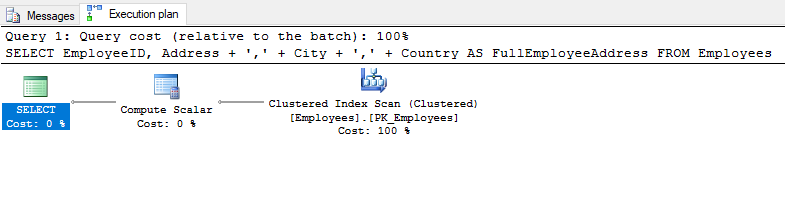
2. A SELECT statement that returns two columns from the employees table; the employeeID, and

one named FullEmployeeAddress that joins the Country, City, and Address columns.

SELECT EmployeeID, Address + ',' + City + ',' + Country AS FullEmployeeAddress FROM Employees







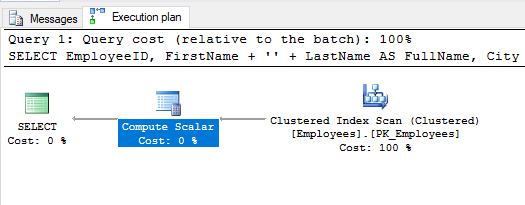
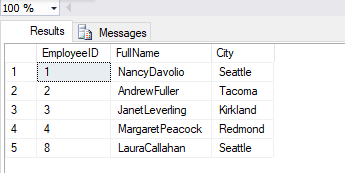
3. A SELECT statement that returns three columns for every employee in a given region.

EmployeeID, FullName column that joins FirstName and LastName, and City

SELECT EmployeeID, FirstName + '' + LastName AS FullName, City FROM Employees

WHERE Region = 'WA';





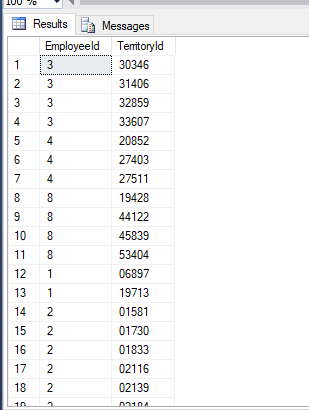
4. A SELECT statement that joins Employees and Territories. Return these

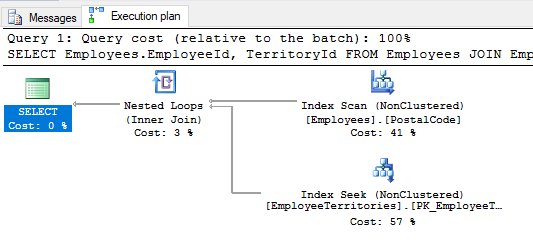
columns: EmployeeID and TerritoryID

SELECT Employees.EmployeeId, TerritoryId FROM Employees

JOIN EmployeeTerritories ON Employees.EmployeeID = EmployeeTerritories.EmployeeID;







Sort the results by RegionID descending.

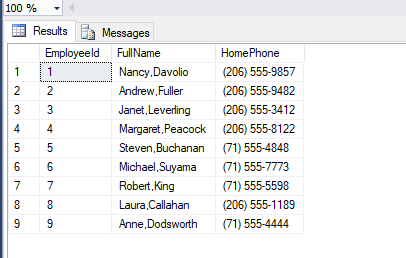
5. A SELECT statement that returns three columns; EmployeeID, A FullName joining FIrstName and

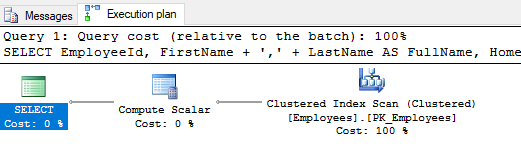
LastName, and HomePhone. Only return results where HomePhone is not NULL.

SELECT EmployeeId, FirstName + ',' + LastName AS FullName, HomePhone

FROM Employees WHERE HomePhone is NOT NULL;





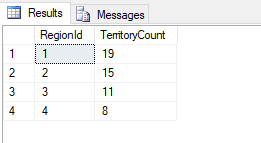


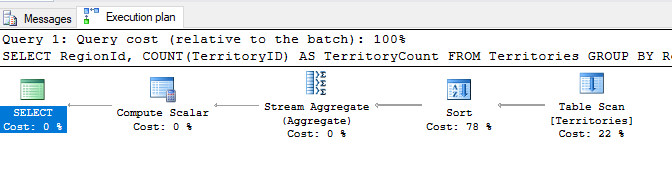
6. A SELECT Statement that returns a count of territories in each region

SELECT RegionId, COUNT(TerritoryID) AS TerritoryCount FROM Territories

GROUP BY RegionID







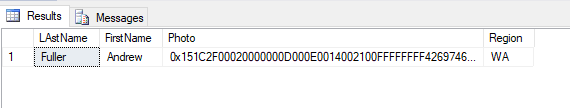
7. A SELECT statement that returns a row for each employee that ReportsTo is null; each row

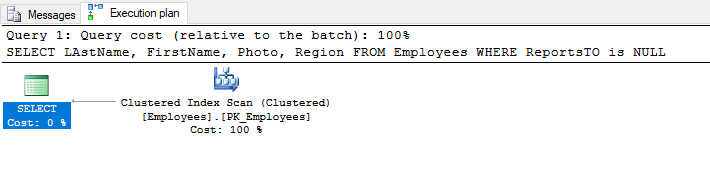
should have a column for Title, LastName, FirstName, Photo, Region.

SELECT LAstName, FirstName, Photo, Region FROM Employees

WHERE ReportsTO is NULL





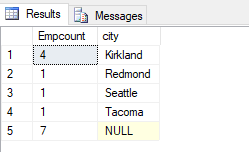


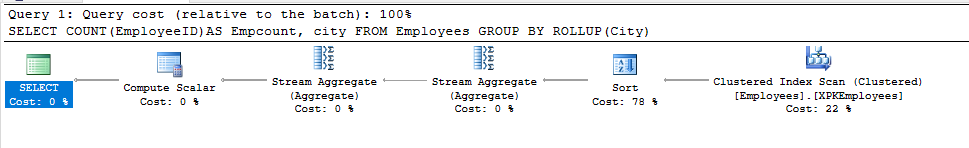
8. A SELECT statement that returns a count of the employees in each city with a ROLLUP.

SELECT COUNT(EmployeeID)AS Empcount, city FROM Employees

GROUP BY ROLLUP(City);





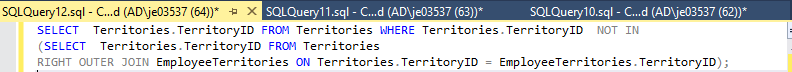


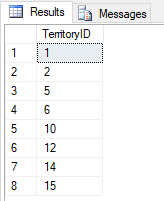
9. A SELECT statement that returns a list of Territories where Employees do not exist.

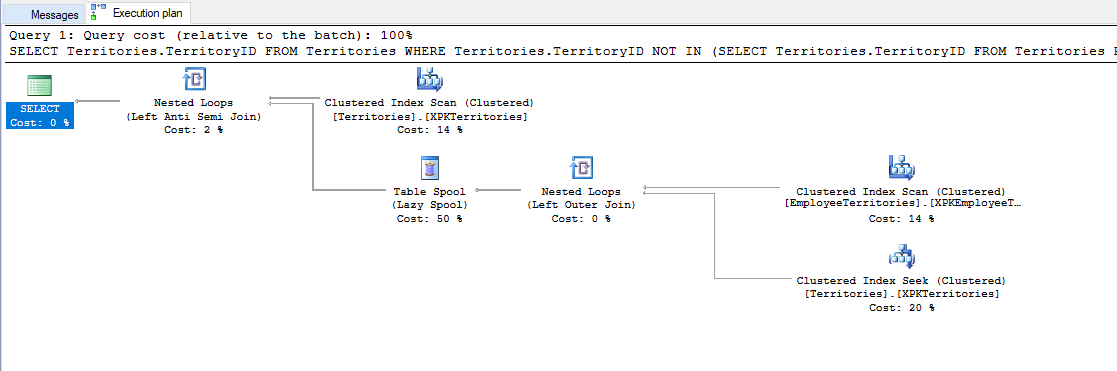
SELECT Territories.TerritoryID FROM Territories WHERE Territories.TerritoryID NOT IN

(SELECT Territories.TerritoryID FROM Territories

RIGHT OUTER JOIN EmployeeTerritories ON Territories.TerritoryID = EmployeeTerritories.TerritoryID);







10. What is the highest total amount by a single order?

SELECT MAX(UnitPrice \* Quantity) AS MAXTOTAL

FROM OrderDetails





