SQL

TASK 1

PART 2

/\*

1) Clone data from the Shippers table to the NewShippers table.

2) Find the set of products (Product Name) and maximum value of units in stock for each one, which is in the range from 25 to 50. Represent records from the min to max value of units in stock.

3) Get the list of total quantities of ordered products which consists of: total quantity ordered in Germany and the total quantiy\* 0.7% of products ordered in Sweden. (Result should contain 2 rows)

4) Find the list of different countries in Employees and Customers tables.

5) Find the list of the same Postal Codes between Suppliers and Employees tables.

6) Find the top region from which sales specialists were hired.

7) Get two lists of products: with a price < 50.00 with a discountinued flag and < 50 without a discountinued flag.

8) Create new table NewProducts based on the Products table with only discountinued products. Compare data sets between Products and NewProducts tables.

(Check that only discountinued products are inserted).

The 9th query is optional

\* Get the list of orders, where a required date is bigger than the Shipped date ( compare in days) and Ship Region is not specified.

\*/

1. Clone data from the Shippers table to the NewShippers table.

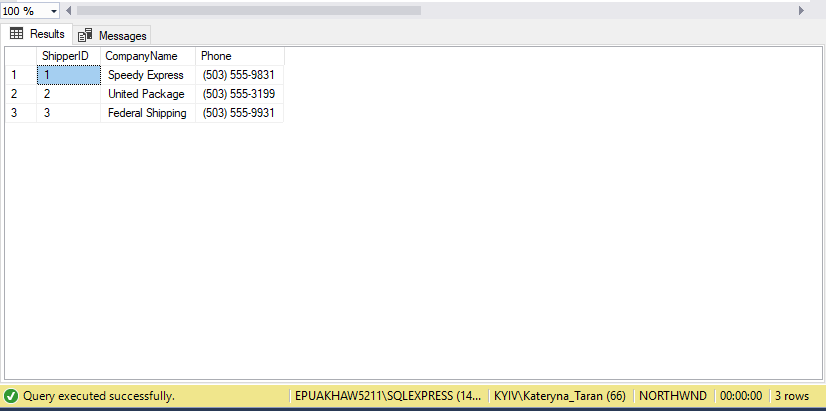
SELECT \*

INTO NewShippers

FROM Shippers

SELECT \*

FROM NewShippers



2) Find the set of products (Product Name) and maximum value of units in stock for each one, which is in the range from 25 to 50.

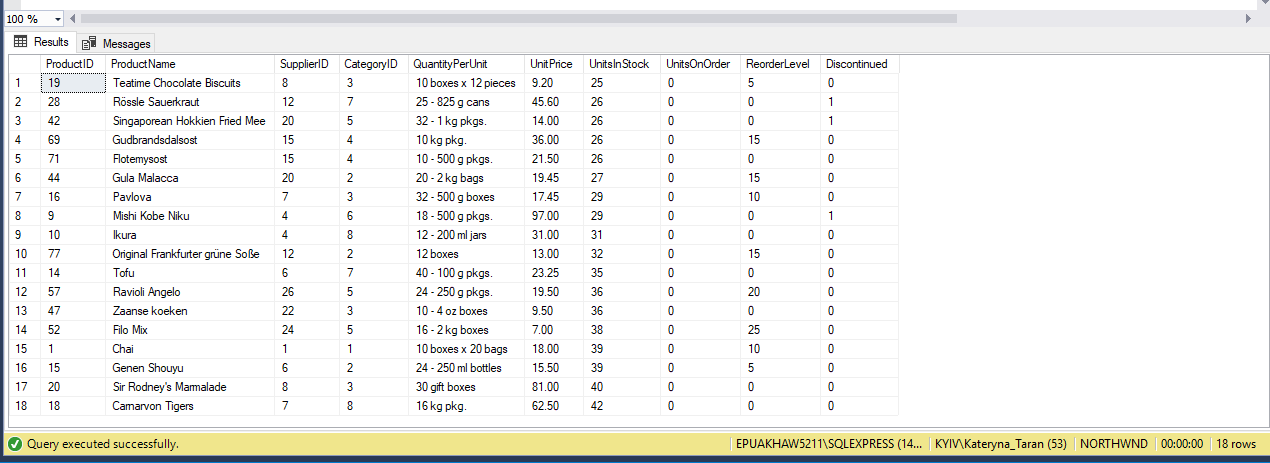
Represent records from the min to max value of units in stock.

SELECT \*

FROM Products

WHERE UnitsInStock BETWEEN 25 AND 50

ORDER BY UnitsInStock ASC



1. Get the list of total quantities of ordered products which consists of: total quantity ordered in Germany and the total quantity\* 0.7% of products ordered in Sweden. (Result should contain 2 rows)

VARIANT 1

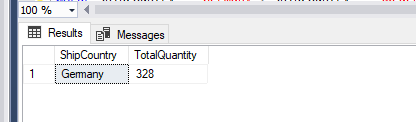
SELECT Orders.ShipCountry, COUNT([Order Details].Quantity) AS TotalQuantity

FROM Orders JOIN [Order Details]

ON Orders.OrderID = [Order Details].OrderID

GROUP BY Orders.ShipCountry

HAVING Orders.ShipCountry = 'Germany'



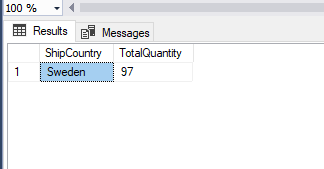
SELECT Orders.ShipCountry, COUNT([Order Details].Quantity) AS TotalQuantity

FROM Orders JOIN [Order Details]

ON Orders.OrderID = [Order Details].OrderID

GROUP BY Orders.ShipCountry

HAVING Orders.ShipCountry = 'Sweden'



SELECT Orders.ShipCountry, COUNT([Order Details].Quantity) AS TotalQuantity, COUNT([Order Details].Quantity) \* 1 AS Result

FROM Orders JOIN [Order Details]

ON Orders.OrderID = [Order Details].OrderID

GROUP BY Orders.ShipCountry

HAVING Orders.ShipCountry = 'Germany'

UNION

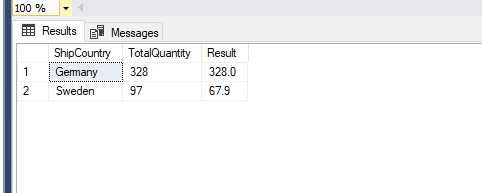
SELECT Orders.ShipCountry, COUNT([Order Details].Quantity) AS TotalQuantity, COUNT([Order Details].Quantity) \*0.7 AS Result

FROM Orders JOIN [Order Details]

ON Orders.OrderID = [Order Details].OrderID

GROUP BY Orders.ShipCountry

HAVING Orders.ShipCountry = 'Sweden'



VARIANT 2

SELECT SUM([Order Details].Quantity) AS TotalQuantity, Customers.Country, SUM([Order Details].Quantity) \*1 AS Result

FROM Customers JOIN Orders

ON Customers.CustomerID = Orders.CustomerID

JOIN [Order Details]

ON [Order Details].OrderID = Orders.OrderID

GROUP BY Customers.Country

HAVING Customers.Country = 'Germany'

UNION

SELECT SUM([Order Details].Quantity) AS TotalQuantity, Customers.Country, SUM([Order Details].Quantity) \* 0.7 AS Result

FROM Customers JOIN Orders

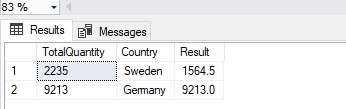
ON Customers.CustomerID = Orders.CustomerID

JOIN [Order Details]

ON [Order Details].OrderID = Orders.OrderID

GROUP BY Customers.Country

HAVING Customers.Country = 'Sweden'



4) Find the list of different countries in Employees and Customers tables.

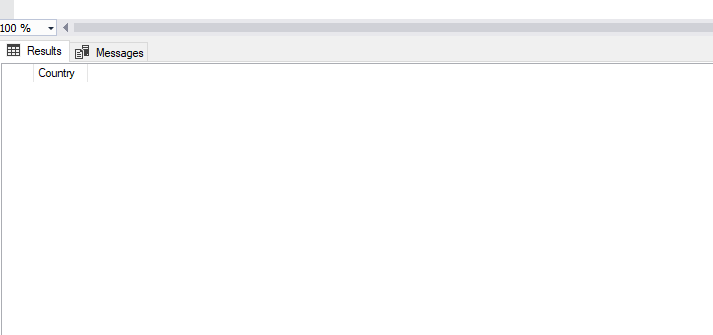
SELECT Country

FROM Employees

EXCEPT

SELECT Country

FROM Customers



5) Find the list of the same Postal Codes between Suppliers and Employees tables.

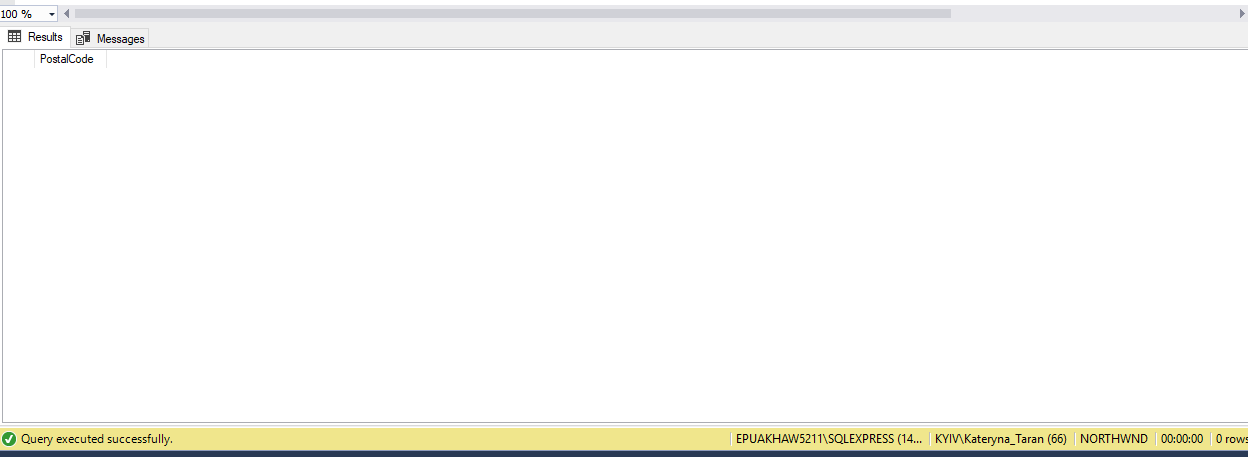
SELECT PostalCode

FROM Suppliers

INTERSECT

SELECT PostalCode

FROM Employees



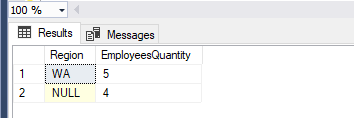
6) Find the top region from which sales specialists were hired.

SELECT Region,COUNT(EmployeeID) AS EmployeesQuantity

From Employees

GROUP BY Region

ORDER BY Region DESC

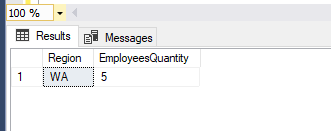


SELECT TOP 1 Region,COUNT(EmployeeID) AS EmployeesQuantity

From Employees

GROUP BY Region

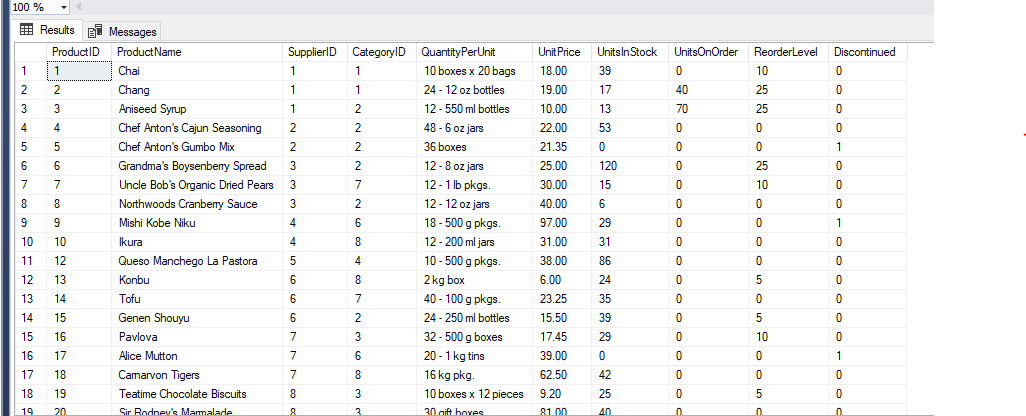
ORDER BY Region DESC



7) Get two lists of products: with a price < 50.00 with a discountinued flag and < 50 without a discountinued flag.

SELECT \*

FROM Products

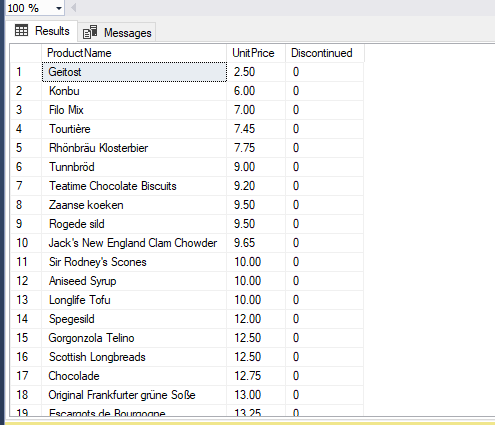


SELECT ProductName, UnitPrice, Discontinued

FROM Products

WHERE UnitPrice < 50.00 and Discontinued <> 1

ORDER BY UnitPrice ASC

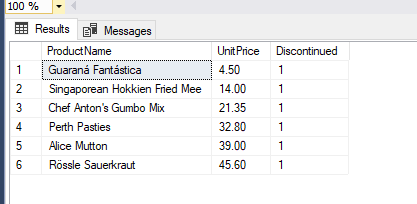


SELECT ProductName, UnitPrice, Discontinued

FROM Products

WHERE UnitPrice < 50.00 AND Discontinued <> 0

ORDER BY UnitPrice ASC



8) Create new table NewProducts based on the Products table with only discountinued products. Compare data sets between Products and NewProducts tables.

(Check that only discountinued products are inserted).

CREATE TABLE NewProducts(

ProductID INT NOT NULL Primary key,

ProductName nvarchar(40) NOT NULL,

SupplierID INT NULL,

FOREIGN key (SupplierID) REFERENCES Suppliers (SupplierID),

CategoryID INT NULL,

FOREIGN key (CategoryID) REFERENCES Categories (CategoryID),

QuantityPerUnit nvarchar(20) NULL,

UnitPrice money NULL,

UnitsInStock smallint NULL,

UnitsOnOrder smallint NULL,

ReorderLevel smallint NULL,

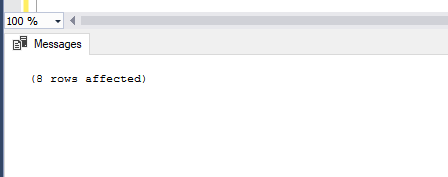
Discontinued bit NOT NULL)

SELECT \*

INTO NewProducts

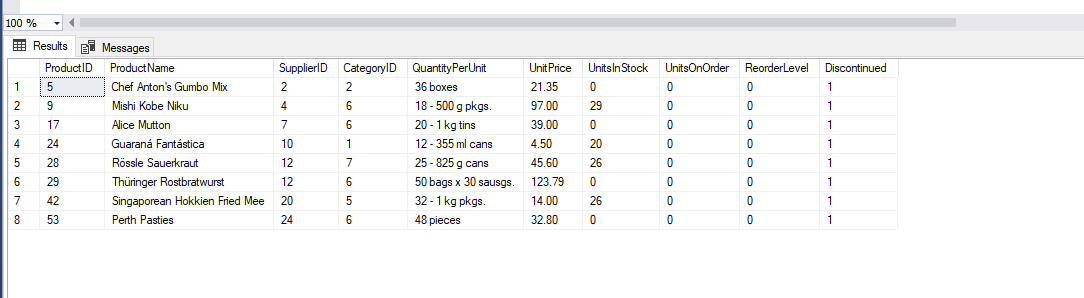
FROM Products

WHERE Discontinued = 1



SELECT \*

FROM NewProducts



The 9th query is optional

\* Get the list of orders, where a required date is bigger than the Shipped date ( compare in days) and Ship Region is not specified.

\*/

SELECT DATEDIFF(DAY,ShippedDate,RequiredDate) AS DifferenceInDays, RequiredDate, ShippedDate, ShipRegion

FROM Orders

WHERE RequiredDate >ShippedDate and shipRegion IS NULL

ORDER BY DifferenceInDays DESC

