Task#1.

Write a query that returns the maximum value in the orderdate column for each customer.

Output:

CustomerID MaxOrderDate

----------- -----------------------

Task#2.

Using the query from Exercise #1 return orders with the maximum order date for each customer.

Output:

SalesOrderID CustomerID OrderDate SalesPersonID

------------ ---------- ----------- -------------

Task#3.

Write a query that calculates a row number for each order based on orderdate, SalesOrderID ordering.

Output:

SalesOrderID OrderDate CustomerID rownum

------------ --------- ---------- ------

Task#4.

Return rows with row numbers 11 through 100 based on the row number definition in Exercise #3 using CTE.

Output:

SalesOrderID OrderDate CustomerID rownum

------------ ---------- ---------- ------

Task#5.

Return the number of distinct customers handled in each order year

Output:

OrderYear NumberOfCustomers

---------- -----------------

Task#6.

Return order years and the number of customers handled in each year only for years in which more than 10000 customers were handled

Output:

OrderYear NumberOfCustomers

---------- -----------------

Task#7.

Calculate the difference between the number of customers handled in the current and previous years.

Output:

OrderYear CurNumCusts PrvNumCusts growth

---------- ----------- ----------- -----------

Task#8.

Create a view named vCustOrders in Sales schema. It should return the total quantity for each customer and year.

When running "SELECT \* FROM Sales.vCustOrders" the output should be:

CustomerID OrderYear OrderQty

---------- --------- -----------

Task#9.

Write a query against Sales.vCustOrders that returns the running total quantity for each customer and year.

Output:

CustomerID OrderYear OrderQty runqty

----------- ---------- --------- -----------

Task#10.

Create an inline UDF Sales.MostRecentOrders that accepts as input a customer ID and a requested number

of orders (@n AS INT). The function should return @n most recent orders for specified customer ID.

Function should return: SalesOrderID,OrderDate,SalesPersonID

Task#11.

Using the function you created in Exercise #10, return, for each customer,the three most recent orders.

Output:

CustomerID SalesOrderID OrderDate SalesPersonID

---------- ------------ ----------- -------------

Task#12.

--Run the following script to add and populate ManagerID column in HumanResources.Employee table:

--------------------------------------------------------------------------------------------USE AdventureWorks2012;

GO

IF COLUMNPROPERTY(OBJECT\_ID('HumanResources.Employee'),'ManagerID','AllowsNull') IS NOT NULL

ALTER TABLE HumanResources.Employee DROP COLUMN ManagerID;

GO

ALTER TABLE HumanResources.Employee ADD ManagerID INT;

GO

UPDATE E

SET ManagerID = M.BusinessEntityID

FROM HumanResources.Employee E

JOIN HumanResources.Employee M

ON M.OrganizationNode = E.OrganizationNode.GetAncestor(1);

GO

--------------------------------------------------------------------------------------------

Create an inline UDF that will take as input @BusinessEntityID and return employee and all his subordinates.

For @BusinessEntityID = NULL all org structure should be returned.

Output of "select \* from HumanResources.GetSubordinates(2)":

EmployeeBusinessEntityID EmployeeJobTitle EmployeeFirstName EmployeeLastName ManagerID ManagerFirstName ManagerLastName Level

------------------------ ----------------------------- ----------------- ---------------- --------- ---------------- --------------- -----

2 Vice President of Engineering Terri Duffy 1 Ken Sanchez 1

3 Engineering Manager Roberto Tamburello 2 Terri Duffy 2

........