/\*

Simple Queries

\*/

USE DB9\_EmiljanoBodaj47

GO

/\*

Problem 01: Find the Supplier Contact Title, the Product ID, the Product Name, and the Unit Price for

products that are LESS expensive than the average unit price using [Production].[Supplier]

and [Production].[Product]? Order it by Unit Price from least to greatest.

\*/

SELECT PS.[SupplierContactTitle]

,P.[ProductId]

,SUBSTRING(P.ProductName, CHARINDEX(' ', P.ProductName) + 1, LEN(P.ProductName)) AS ProductName

,P.[UnitPrice]

FROM [Production].[Product] AS P

INNER JOIN [Production].[Supplier] AS PS ON P.SupplierId = PS.SupplierId

WHERE UnitPrice < (

SELECT AVG(UnitPrice)

FROM [Production].[Product]

)

ORDER BY UnitPrice ASC

FOR JSON PATH

,ROOT('Supplier INFO');

USE DB9\_EmiljanoBodaj47

GO

/\*

Problem 02: Find the personal information, the Country, Name, Address, Phone #, and Postal Code (in that order) of the suppliers,

with the supplementary Product Name and ID's of all Québecois/Québec-based suppliers?

Concatenate the Supplier's Region and Country for brevity's sake then order by the Company Name.

\*/

SELECT CONCAT (

PS.[SupplierRegion]

,', '

,PS.[SupplierCountry]

) AS SupplierArea

,PS.[SupplierContactName]

,PS.[SupplierPhoneNumber]

,PS.[SupplierAddress]

,PS.[SupplierPostalCode]

,SUBSTRING(P.ProductName, CHARINDEX(' ', P.ProductName) + 1, LEN(P.ProductName)) AS ProductName

,P.[ProductId]

FROM [Production].[Product] AS P

INNER JOIN [Production].[Supplier] AS PS ON P.SupplierId = PS.SupplierId

WHERE PS.SupplierRegion = N'Québec'

ORDER BY SupplierCompanyName

FOR JSON PATH

,ROOT('Supplier Personal INFO');

USE DB9\_EmiljanoBodaj47

GO

/\*

Problem 03: Find the OrderId, CustomerId, CustomerCountry, and CustomerContactTitle using the tables [Sales].[Customer]

and [Sales].[Order]. The people who made NO orders should be displayed at the top of the output. Those who

haven't made an order shouldn't have a CustomerId as well. (NorthWind).

\*/

SELECT O.[OrderId]

,O.CustomerId

,C.CustomerCountry

,C.[CustomerContactTitle]

FROM [Sales].[Order] AS O

RIGHT JOIN [Sales].[Customer] AS C ON O.CustomerId = C.CustomerId

ORDER BY O.OrderId

FOR JSON PATH

,ROOT('Customer and Order INFO');

USE DB9\_EmiljanoBodaj47

GO

/\*

Problem 04: Reduce the query results to the Customers (in [Sales].[Order] and [Sales].[Customer])

who've made absolutely no orders using the EXCEPT clause

\*/

SELECT CustomerId

FROM [Sales].[Customer]

EXCEPT

SELECT CustomerId

FROM [Sales].[Order]

ORDER BY CustomerId DESC

FOR JSON PATH

,ROOT('Customer Orders');

USE DB9\_EmiljanoBodaj47

GO

/\*

Problem 05: Find the ShipperID, complete Shipping Address (Country, City, Region, Postal Code) and Orderdate between 2015 and 2016,

excluding the months of December, January, February, and March using Set Operators. Use Northwind2019TSQLV5

\*/

SELECT [ShipperId]

,CONCAT (

[ShipToCountry]

,', '

,[ShipToCity]

,', '

,[ShipToRegion]

,' '

,[ShipToPostalCode]

) AS ShippingAddress

,[OrderDate]

FROM [Sales].[Order]

WHERE (

YEAR(OrderDate) BETWEEN 2015

AND 2016

)

EXCEPT

SELECT [ShipperId]

,CONCAT (

[ShipToCountry]

,', '

,[ShipToCity]

,', '

,[ShipToRegion]

,' '

,[ShipToPostalCode]

) AS ShippingAddress

,[OrderDate]

FROM [Sales].[Order]

WHERE (

MONTH(OrderDate) IN (

12

,1

,2

,3

)

)

ORDER BY OrderDate

FOR JSON PATH

,ROOT('Specific OrderDates');

/\*

Medium Queries

\*/

USE DB9\_EmiljanoBodaj47

GO

/\*

Problem 06: Find the Full Name (concat), Title of Courtesy, Nationality, Age (datediff()), and EmployeeId and order it by the employee's

surname? Use [HumanResources].[Employee] and [Sales].[Order] in NorthWind

\*/

SELECT CONCAT (

HRE.[EmployeeLastName]

,', '

,HRE.[EmployeeFirstName]

) AS 'Full Name'

,HRE.[EmployeeTitleOfCourtesy] AS 'Title of Courtesy'

,CASE

WHEN HRE.EmployeeCountry = 'USA'

THEN 'American'

WHEN HRE.EmployeeCountry = 'Canada'

THEN 'Canadian'

WHEN HRE.EmployeeCountry IN (

'Germany'

,'UK'

,'Norway'

,'Belgium'

,'Poland'

,'Switzerland'

,'France'

,'Spain'

,'Italy'

,'Portugal'

,'Denmark'

,'Sweden'

,'Finland'

)

THEN 'European'

WHEN HRE.EmployeeCountry IN (

'Brazil'

,'Mexico'

,'Argentina'

,'Venezuela'

)

THEN 'Latin American'

END AS 'Nationality'

,DATEDIFF(year, [BirthDate], GETDATE()) AS Age

,SO.EmployeeId AS ID

FROM [HumanResources].[Employee] AS HRE

INNER JOIN [Sales].[Order] AS SO ON HRE.[EmployeeId] = SO.EmployeeId

GROUP BY SO.EmployeeId

,HRE.EmployeeFirstName

,HRE.EmployeeLastName

,HRE.EmployeeTitleOfCourtesy

,HRE.EmployeeCountry

,HRE.BirthDate

ORDER BY SUBSTRING(HRE.[EmployeeLastName], 1, 1)

FOR JSON PATH

,ROOT('Employee Nationality');

USE DB9\_EmiljanoBodaj47

GO

/\*

Problem 07: Find the CustId, CustomerCompanyName (Without 'Customer'), OrderDate (NULLs must be 'XXXXXXX'), OrderId (NULLs must be 'N/A')

for order that were made at the end of the semester for either the Fall or Spring, with every other date being NULL. Order by Order Date

\*/

SELECT SC.[CustomerId]

,REPLACE(SC.[CustomerCompanyName], 'Customer', '') AS 'Company Name'

,ISNULL(CONVERT(VARCHAR(30), SO.OrderDate, 121), 'XXXXXXXX') AS 'Order Date'

,ISNULL(CONVERT(VARCHAR(30), SO.OrderId, 121), 'N/A') AS 'Order Id'

FROM [Sales].[Customer] AS SC

LEFT OUTER JOIN [Sales].[Order] AS SO ON SC.CustomerId = SO.CustomerId

AND DAY(SO.OrderDate) IN (

21

,20

)

AND MONTH(SO.OrderDate) IN (

12

,5

)

--Group By won't make a difference

ORDER BY SO.OrderDate DESC

FOR JSON PATH

,ROOT('Customer Info');

USE DB9\_EmiljanoBodaj47

GO

/\*

Problem 08: Using the Intersect and Except Set Operations, write a query that'll print out all the Orders made at the END OF THE MONTH between 2015 and 2016. Use [Sales].[Order] and

[Northwinds2019TSQLV5]. Order it in ascending order and don't forget the fact that 2015 was a leap year.

\*/

SELECT EmployeeId

,CustomerId

,OrderDate

FROM [Sales].[Order]

WHERE MONTH(OrderDate) IN (

SELECT MONTH(OrderDate)

FROM [Sales].[Order]

WHERE MONTH(OrderDate) BETWEEN 1

AND 6

)

INTERSECT

SELECT EmployeeId

,CustomerId

,OrderDate

FROM [Sales].[Order]

WHERE YEAR(OrderDate) IN (

SELECT YEAR(OrderDate)

FROM [Sales].[Order]

WHERE YEAR(OrderDate) BETWEEN 2015

AND 2016

)

EXCEPT

SELECT EmployeeId

,CustomerId

,OrderDate

FROM [Sales].[Order]

WHERE DAY(OrderDate) != DAY(EOMONTH(OrderDate))

ORDER BY OrderDate

FOR JSON PATH

,ROOT('2015-2016 EOMONTH Orders');

USE DB9\_EmiljanoBodaj47

GO

/\*

Problem 09: Find the TOP .25% of ProductId, UnitPrice (Use a $ symbol) above $10.00, Discontinued status (Yes or No), Quantity, Discount Percentage,

and CategoryId in the CARTESIAN PRODUCT of [Sales].[OrderDetail] and [Production].[Product] using the NorthWind database. Order it by

Discount Percentage

\*/

SELECT TOP (.25) PERCENT PR.[ProductId]

,CONCAT (

'$'

,PR.[UnitPrice]

) AS 'Unit Price'

,CASE

WHEN PR.[Discontinued] = 1

THEN 'Yes'

WHEN PR.[Discontinued] = 0

THEN 'No'

END AS 'Discontinued?'

,CONCAT (

OD.[Quantity]

,' Items'

) AS Quantity

,CONCAT (

CONVERT(DOUBLE PRECISION, (OD.[DiscountPercentage] \* 100))

,'% '

) AS 'Discount Percentage'

,PR.[CategoryId]

FROM [Sales].[OrderDetail] AS OD

CROSS JOIN [Production].[Product] AS PR

WHERE PR.UnitPrice > 10.00

ORDER BY OD.[DiscountPercentage] DESC

FOR JSON PATH

,ROOT('Top .25% of Discount Percentage');

USE [AdventureWorks2014]

GO

/\*

Problem 10: Use the Full Outer Join to combine the [EmployeeDepartmentHistory] and [BusinessEntityContact] tables and use the EXCEPT operator

to filter out to rows where BusinessEntity is Null and then when EndDate isn't Null ("Ongoing"). Use the AventureWorks2014 database. This problem

will demonstrate understanding of EXCEPT.

\*/

SELECT EDH.[BusinessEntityID]

,EDH.[ShiftID]

,EDH.[DepartmentID]

,EDH.[StartDate]

,ISNULL(CAST(EDH.[EndDate] AS VARCHAR), 'Ongoing') AS 'EndDate'

FROM [HumanResources].[EmployeeDepartmentHistory] AS EDH

FULL OUTER JOIN [Person].[BusinessEntityContact] AS BEC ON EDH.[BusinessEntityID] = BEC.[BusinessEntityID]

EXCEPT

SELECT EDH1.BusinessEntityID

,EDH1.[ShiftID]

,EDH1.[DepartmentID]

,EDH1.[StartDate]

,ISNULL(CAST(EDH1.[EndDate] AS VARCHAR), 'Ongoing') AS 'EndDate'

FROM [HumanResources].[EmployeeDepartmentHistory] AS EDH1

FULL OUTER JOIN [Person].[BusinessEntityContact] AS BEC1 ON EDH1.[BusinessEntityID] = BEC1.[BusinessEntityID]

WHERE EDH1.BusinessEntityID IS NULL

EXCEPT

SELECT EDH2.BusinessEntityID

,EDH2.[ShiftID]

,EDH2.[DepartmentID]

,EDH2.[StartDate]

,ISNULL(CAST(EDH2.[EndDate] AS VARCHAR), 'Ongoing') AS 'EndDate'

FROM [HumanResources].[EmployeeDepartmentHistory] AS EDH2

FULL OUTER JOIN [Person].[BusinessEntityContact] AS BEC2 ON EDH2.[BusinessEntityID] = BEC2.[BusinessEntityID]

WHERE (

YEAR(EDH2.[EndDate]) BETWEEN 2009

AND 2013

)

ORDER BY (EDH.StartDate) DESC

FOR JSON PATH

,ROOT('Customer Info');

USE [AdventureWorks2014]

GO

/\*

Problem 11: Find the Full Names of Employee who are Production Technicians in division WC40, their EmailAddress, JobTitle, the number of years

they've worked at that position, as well as their average Vacation + Sick hours by year (estimated). Sort them

\*/

SELECT CONCAT (

P.[LastName]

,', '

,P.[FirstName]

,' '

,P.[MiddleName]

) AS EmployeeName

,EA.[EmailAddress]

,E.[JobTitle]

,DATEDIFF(YEAR, [HireDate], SYSDATETIME()) AS YearsEmployed

,((VacationHours + SickLeaveHours) / DATEDIFF(YEAR, [HireDate], SYSDATETIME())) AS AverageVacationAndSickHoursbyYear

FROM [Person].[Person] AS P

INNER JOIN [HumanResources].[Employee] AS E ON P.BusinessEntityID = E.BusinessEntityID

INNER JOIN [Person].[EmailAddress] AS EA ON E.[BusinessEntityID] = EA.[BusinessEntityID]

WHERE EXISTS (

SELECT JobTitle

FROM Person.Person

WHERE JobTitle LIKE '%Production Technician%'

AND JobTitle LIKE '%WC40%'

)

ORDER BY AverageVacationAndSickHoursbyYear DESC

FOR JSON PATH

,ROOT('Average Unworked Hours');

USE [AdventureWorks2014]

GO

/\*

Problem 12: Find the BusinessEntityID and OrgnizationNode of those whose BusinessID fall between 20 and 60 and

havv a log-in ID that ends with 0. Use HR.Employee and HR.EmployeeDepartmentHistory. Use Format() to format

the HireDate by MM/dd/yyy.

\*/

SELECT E.[BusinessEntityID]

,E.[OrganizationNode]

,E.[LoginID]

,FORMAT(E.[HireDate], 'MM/dd/yyyy') AS HireDate

FROM [HumanResources].[Employee] AS E

WHERE (

E.BusinessEntityID BETWEEN 20

AND 60

)

AND EXISTS (

SELECT [BusinessEntityID]

FROM [HumanResources].[EmployeeDepartmentHistory] AS EA

WHERE E.BusinessEntityID = EA.BusinessEntityID

)

AND E.LoginID LIKE '%0'

ORDER BY YEAR(E.HireDate) DESC

FOR JSON PATH

,ROOT('Specific Business Entity IDs');

/\*

Complex Queries

\*/

USE [AdventureWorks2014]

GO

/\*

Problem 13: Find the grouped ProductID's and the Total Quantity, Total Receieved Quantity, Total Rejected Quantity by ProductID

for ProductID 317.

\*/

SELECT [ProductID]

,FORMAT(CAST(SUM(UnitPrice) AS MONEY), '$#####.####') AS TotalUnitPriceByProductID

,SUM([OrderQty]) AS TotalQuantityByProductID

,SUM([ReceivedQty]) AS TotalReceivedQuantity

,SUM([RejectedQty]) AS TotalRejectedQuantity

,FORMAT(CAST(SUM([TaxAmt]) AS MONEY), '$######.####') AS TaxAmount

FROM [Purchasing].[PurchaseOrderDetail] AS POD

FULL OUTER JOIN [Purchasing].[PurchaseOrderHeader] AS POR ON POD.PurchaseOrderID = POR.PurchaseOrderID

WHERE POD.ProductID = (

SELECT ProductID

FROM [Purchasing].[PurchaseOrderDetail]

WHERE ProductID = 317

GROUP BY ProductID

)

GROUP BY ProductID

FOR JSON PATH

,ROOT('ProductID 317');

USE [DB9\_EmiljanoBodaj47]

GO

/\*

Problem 14: Find the top 50 Order Id's with the greatest Total Values (Quantity + UnitPrice), including the decimal Discount Percentage, the Total Value

BEFORE the Discount Percentage, as well as the total freight. Use NorthWind with the [Sales].[Customer], [Sales].[Order], and [Sales].[OrderDetail] tables.

\*/

SELECT TOP 50 OD.OrderId

,SUM([dbo].[qtyTimesPrice](OD.Quantity, OD.[UnitPrice])) AS 'Total Value'

,CONCAT (

CONVERT(DOUBLE PRECISION, SUM(OD.[DiscountPercentage] \* 100))

,'% '

) AS 'Discount Percentage'

,CONVERT(DOUBLE PRECISION, ((SUM(OD.Quantity \* OD.[UnitPrice]) \* SUM(OD.DiscountPercentage)) + SUM(OD.Quantity \* OD.[UnitPrice]))) AS 'Price Before Discount'

,SUM(SO.Freight) AS 'Total Freight'

FROM [Sales].[OrderDetail] AS OD

INNER JOIN [Sales].[Order] AS SO ON SO.OrderId = OD.OrderId

INNER JOIN [Sales].[Customer] AS SA ON SO.CustomerId = SA.[CustomerId]

GROUP BY OD.OrderID

,SA.CustomerId

HAVING SUM(OD.DiscountPercentage) > 0.000

ORDER BY SUM(OD.Quantity \* OD.[UnitPrice]) DESC

FOR JSON PATH

,ROOT('Discount by OrderId');

USE [AdventureWorks2014]

GO

/\*

Problem 15: Find the sales made last year by BusinessEntityId. Include Bonuses, Sales Quota, Address (City, Addressline1, and Territory Id), as well as the year of the Modified Date

in descending order. Use tables [Sales].[SalesPerson] and [Sales].[SalesTerritory] in the [AdventureWorks2014] database.

CAST Sales into money so they'll be separated by commas and throw in $.

\*/

SELECT SP.[BusinessEntityID]

,CONCAT (

'$'

,CONVERT(VARCHAR, CAST(SP.[SalesLastYear] AS MONEY), 1)

) AS 'Sales Last Year'

,CONCAT (

'$'

,CONVERT(VARCHAR, CAST(SP.[Bonus] AS MONEY), 1)

) AS 'Total Bonuses'

,CONCAT (

'$'

,ISNULL(CONVERT(VARCHAR, CAST(SP.[SalesQuota] AS MONEY), 1), 0.00)

) AS 'Sales Quota'

,ISNULL(CAST(ST.[TerritoryID] AS VARCHAR), 'N/A') AS 'Territory Id'

,YEAR(ST.[ModifiedDate]) AS 'Business Year'

,CONCAT (

PA.[City]

,', '

,PA.[AddressLine1]

,', '

,BE.[AddressID]

) AS Address

,[dbo].[plusBonus](SP.[SalesLastYear], SP.[Bonus]) AS SalesPlusBonus

FROM [Sales].[SalesPerson] AS SP

INNER JOIN [Person].[BusinessEntityAddress] AS BE ON SP.[BusinessEntityID] = BE.[BusinessEntityID]

INNER JOIN [Sales].[SalesTerritory] AS ST ON SP.[TerritoryID] = ST.[TerritoryID]

INNER JOIN [Person].[Address] AS PA ON BE.[AddressID] = PA.[AddressID]

ORDER BY SP.SalesLastYear DESC

FOR JSON PATH

,ROOT('Sales Last Years');

USE [AdventureWorks2014]

GO

/\*

Problem 16: Find the ProductID, Bin, ListPrice , Quantity, ProductReview, and Price of Stock (Qty \* Price) (Scalar)

that are in Shelves E and C with the tables [Production].[ProductInventory], [Production].[ProductCostHistory],

and [Production].[ProductReview]. Use group by to remove exact duplicates.

\*/

SELECT I.[ProductID]

,I.[Shelf]

,I.[Bin]

,PLPH.[ListPrice]

,I.[Quantity]

,ISNULL(CAST(PR.[ProductReviewID] AS NVARCHAR), 'N/A') AS ProductRating

,[dbo].[getStockPrice](I.Quantity, PLPH.ListPrice) AS PriceOfStock

FROM [Production].[ProductInventory] AS I

RIGHT JOIN [Production].[ProductCostHistory] AS CH ON I.ProductID = CH.ProductID

AND (I.Shelf != 'N/A')

LEFT OUTER JOIN [Production].[ProductReview] AS PR ON I.ProductID = PR.ProductID

INNER JOIN [Production].[ProductListPriceHistory] AS PLPH ON I.[ProductID] = PLPH.[ProductID]

WHERE (I.ProductID IS NOT NULL)

AND (I.Shelf IN(N'E', N'C'))

GROUP BY I.[ProductID]

,I.[Shelf]

,I.[Bin]

,PLPH.[ListPrice]

,I.[Quantity]

,PR.[ProductReviewID]

ORDER BY PR.ProductReviewID DESC

FOR JSON PATH

,ROOT('Product Info');

USE [DB9\_EmiljanoBodaj47]

GO

/\*

Problem 17: Find the top 100 Order Id's with the smallest Total Values (Quantity + UnitPrice), including the decimal Discount Percentage, the Total Value

BEFORE the Discount Percentage, as well as the total freight. Use NorthWind with the [Sales].[Customer], [Sales].[Order], and [Sales].[OrderDetail] tables.

\*/

SELECT TOP 100 OD.OrderId

,SUM([dbo].[qtyTimesPrice](OD.Quantity, OD.[UnitPrice])) AS 'Total Value'

,CONCAT (

CONVERT(DOUBLE PRECISION, SUM(OD.[DiscountPercentage] \* 100))

,'% '

) AS 'Discount Percentage'

,SUM([dbo].[percentGet1](OD.Quantity, OD.UnitPrice, OD.DiscountPercentage)) AS 'Price Before Discount'

,SUM(SO.Freight) AS 'Total Freight'

FROM [Sales].[OrderDetail] AS OD

INNER JOIN [Sales].[Order] AS SO ON SO.OrderId = OD.OrderId

INNER JOIN [Sales].[Customer] AS SA ON SO.CustomerId = SA.[CustomerId]

GROUP BY OD.OrderID

,SA.CustomerId

HAVING SUM(OD.DiscountPercentage) > 0.000

ORDER BY SUM([dbo].[qtyTimesPrice](OD.Quantity, OD.[UnitPrice])) ASC

FOR JSON PATH

,ROOT('Discount by OrderId');

USE [DB9\_EmiljanoBodaj47]

GO

/\*

Problem 18: Find the Order Id's with the Discount Percentages between 50 and 75, Total Values (Quantity + UnitPrice), the Total Value

BEFORE the Discount Percentage, as well as the total freight. Use NorthWind with the [Sales].[Customer], [Sales].[Order], and [Sales].[OrderDetail] tables.

\*/

SELECT OD.OrderId

,SUM([dbo].[qtyTimesPrice](OD.Quantity, OD.[UnitPrice])) AS 'Total Value'

,CONCAT (

CONVERT(DOUBLE PRECISION, SUM(OD.[DiscountPercentage] \* 100))

,'% '

) AS 'Discount Percentage'

,CONVERT(DOUBLE PRECISION, ((SUM(OD.Quantity \* OD.[UnitPrice]) \* SUM(OD.DiscountPercentage)) + SUM(OD.Quantity \* OD.[UnitPrice]))) AS 'Price Before Discount'

,SUM(SO.Freight) AS 'Total Freight'

FROM [Sales].[OrderDetail] AS OD

INNER JOIN [Sales].[Order] AS SO ON SO.OrderId = OD.OrderId

INNER JOIN [Sales].[Customer] AS SA ON SO.CustomerId = SA.[CustomerId]

GROUP BY OD.OrderID

,SA.CustomerId

HAVING SUM(OD.DiscountPercentage) > 0.000 AND SUM(OD.[DiscountPercentage] \* 100) BETWEEN 50 AND 75

ORDER BY SUM(OD.[DiscountPercentage]) DESC

FOR JSON PATH

,ROOT('Discount by OrderId');

USE DB9\_EmiljanoBodaj47

GO

/\*

Problem 19: Output the Employee Name (Title + Surname), Title of Courtesy, Nationality, Age (Use Scalar Function),

EmployeeId, as well as the Years Hired using [HumanResources].[Employee] and [Sales].[Order] in NorthWind.

The query should only display the employees that have been working at that company for the MAX Years Hire ordered by Age

CROSS JOIN this gargatuan query with dbo.Nums and set N to represent the MAX years an Employee has worked

at this location.

\*/

SELECT N.n AS SeniorityYear, [dbo].[getAge]([HireDate]) AS 'Years Working At Company'

,[dbo].[getAge]([BirthDate]) AS Age, CONCAT (

HRE.[EmployeeTitleOfCourtesy]

,' '

,HRE.[EmployeeLastName]

) AS 'Employee Name'

,CASE

WHEN HRE.EmployeeCountry = 'USA'

THEN 'American'

WHEN HRE.EmployeeCountry = 'Canada'

THEN 'Canadian'

WHEN HRE.EmployeeCountry IN (

'Germany'

,'UK'

,'Norway'

,'Belgium'

,'Poland'

,'Switzerland'

,'France'

,'Spain'

,'Italy'

,'Portugal'

,'Denmark'

,'Sweden'

,'Finland'

)

THEN 'European'

WHEN HRE.EmployeeCountry IN (

'Brazil'

,'Mexico'

,'Argentina'

,'Venezuela'

)

THEN 'Latin American'

END AS 'Nationality'

,SO.EmployeeId AS ID

FROM [HumanResources].[Employee] AS HRE

INNER JOIN [Sales].[Order] AS SO ON HRE.[EmployeeId] = SO.EmployeeId

CROSS JOIN dbo.Nums AS N

WHERE [dbo].[getAge]([HireDate]) >= 7 AND N.n = (SELECT MAX([dbo].[getAge]([HireDate])) FROM HumanResources.Employee)

GROUP BY SO.EmployeeId

,HRE.EmployeeFirstName

,HRE.EmployeeLastName

,HRE.EmployeeTitleOfCourtesy

,HRE.EmployeeCountry

,HRE.BirthDate

,HRE.HireDate

,N.n

ORDER BY [dbo].[getAge]([BirthDate]) DESC

FOR JSON PATH

,ROOT('Employee Seniority');

USE DB9\_EmiljanoBodaj47

GO

/\*

Problem 20: Output the Employee Name (Title + Surname), Title of Courtesy, Nationality, Age (Use Scalar Function),

EmployeeId, as well as the Years Hired using [HumanResources].[Employee] and [Sales].[Order] in NorthWind.

The query should only display the employees that have been working at that company for the non-MAX Years Hired

CROSS JOIN this gargatuan query with dbo.Nums and set N to represent the MIN years an Employee has worked

at this location.

Order by Youngest Employee to the Oldest

\*/

SELECT N.n AS SeniorityYear,

[dbo].[getAge]([HireDate]) AS 'Years Working At Company'

,[dbo].[getAge]([BirthDate]) AS Age, CONCAT (

HRE.[EmployeeTitleOfCourtesy]

,' '

,HRE.[EmployeeLastName]

) AS 'Employee Name'

,CASE

WHEN HRE.EmployeeCountry = 'USA'

THEN 'American'

WHEN HRE.EmployeeCountry = 'Canada'

THEN 'Canadian'

WHEN HRE.EmployeeCountry IN (

'Germany'

,'UK'

,'Norway'

,'Belgium'

,'Poland'

,'Switzerland'

,'France'

,'Spain'

,'Italy'

,'Portugal'

,'Denmark'

,'Sweden'

,'Finland'

)

THEN 'European'

WHEN HRE.EmployeeCountry IN (

'Brazil'

,'Mexico'

,'Argentina'

,'Venezuela'

)

THEN 'Latin American'

END AS 'Nationality'

,SO.EmployeeId AS ID

FROM [HumanResources].[Employee] AS HRE

INNER JOIN [Sales].[Order] AS SO ON HRE.[EmployeeId] = SO.EmployeeId

CROSS JOIN dbo.Nums AS N

WHERE [dbo].[getAge]([HireDate]) != 7 AND N.n = (SELECT MAX([dbo].[getAge]([HireDate])) FROM HumanResources.Employee)

GROUP BY SO.EmployeeId

,HRE.EmployeeFirstName

,HRE.EmployeeLastName

,HRE.EmployeeTitleOfCourtesy

,HRE.EmployeeCountry

,HRE.BirthDate

,HRE.HireDate

,N.n

ORDER BY [dbo].[getAge]([BirthDate]) ASC

FOR JSON PATH

,ROOT('Employee non-Seniority');