Tasks 1-6 are all in the scripted database Banking

7. Define a function in the database TelerikAcademy that returns all Employee's names (first or middle or last name)

and all town's names that are comprised of given set of letters. Example 'oistmiahf' will return 'Sofia', 'Smith', … but not 'Rob' and 'Guy'.

CREATE FUNCTION CheckIfHasLetters (@word nvarchar(20), @letters nvarchar(20))

RETURNS BIT

AS

BEGIN

DECLARE @lettersLen int = LEN(@letters),

@matches int = 0,

@currentChar nvarchar(1)

WHILE(@lettersLen > 0)

BEGIN

SET @currentChar = SUBSTRING(@letters, @lettersLen, 1)

IF(CHARINDEX(@currentChar, @word, 0) > 0)

BEGIN

SET @matches += 1

SET @lettersLen -= 1

END

ELSE

SET @lettersLen -= 1

END

IF(@matches >= LEN(@word) OR @matches >= LEN(@letters))

RETURN 1

RETURN 0

END

GO

CREATE FUNCTION NamesAndTowns(@letters nvarchar(20))

RETURNS @ResultTable TABLE

(

Name varchar(50) NOT NULL

)

AS

BEGIN

INSERT INTO @ResultTable

SELECT LastName FROM Employees

INSERT INTO @ResultTable

SELECT FirstName FROM Employees

INSERT INTO @ResultTable

SELECT towns.Name FROM Towns towns

DELETE FROM @ResultTable

WHERE dbo.CheckIfHasLetters(Name, @letters) = 0

RETURN

END

GO

SELECT \* FROM dbo.NamesAndTowns('oistmiahf')

8. Using database cursor write a T-SQL script that scans all employees and their addresses and prints all pairs of employees that live in the same town.

CREATE PROCEDURE uspEmployeesInTown @results CURSOR VARYING OUTPUT

AS

BEGIN

SET @results = CURSOR FOR

SELECT emp.LastName, towns.Name FROM Employees emp

JOIN Addresses addr

ON emp.AddressID = addr.AddressID

JOIN Towns towns

ON addr.TownID = towns.TownID

GROUP BY towns.TownID, towns.Name, emp.LastName

OPEN @results

END

GO

DECLARE @employeesInTowns CURSOR

DECLARE @LastName varchar(20), @TownName varchar(20)

EXEC uspEmployeesInTown @results = @employeesInTowns OUTPUT

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

-- If no messages are shown, debug it and click on the messages tab

-- sometimes they get bugged and don't show when code is run normally

-- probably because they are too much.

-- Alternatively replace PRINT with SELECT, but the result won't be pretty.

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

WHILE @@FETCH\_STATUS = 0

BEGIN

PRINT @LastName + ' ' + @TownName

FETCH NEXT FROM @employeesInTowns

INTO @LastName, @TownName

END

CLOSE @employeesInTowns

DEALLOCATE @employeesInTowns

9. \* Write a T-SQL script that shows for each town a list of all employees that live in it. Sample output:

Sofia -> Svetlin Nakov, Martin Kulov, George Denchev

Ottawa -> Jose Saraiva

…

SELECT Towns.Name AS Town,

dbo.STRCONCAT(emp.FirstName + ' ' + emp.LastName) AS Employees

FROM Towns

JOIN Addresses AS addr

ON Towns.TownID = addr.TownID

JOIN Employees AS emp

ON emp.AddressID = addr.AddressID

GROUP BY Towns.Name

ORDER BY Towns.Name

10. Define a .NET aggregate function StrConcat that takes as input a sequence of strings and return a single string that consists of the input strings separated by ','. For example the following SQL statement should return a single string:

SELECT StrConcat(FirstName + ' ' + LastName)

FROM Employees

-- Enable CLR for SQL Server

sp\_configure 'clr enabled', 1

GO

RECONFIGURE

GO

-- Compile the solution in CLRFunctions

-- Go to TelerikAcademy -> Programmability -> Assemblies

-- Right click Assemblies -> New Assembly

-- Browse path to CLRFunctions\ConcatenationFunction\bin\Release and choose the DLL file there

-- this path is in my homework directory, after this is done - proceed with the next batch

CREATE AGGREGATE STRCONCAT (@input nvarchar(200))

RETURNS nvarchar(max)

EXTERNAL NAME ConcatenationFunction.Concat;

GO

SELECT dbo.STRCONCAT(FirstName + ' ' + LastName) FROM Employees