

Creating Programmatic SQL Database Objects

Lab 3 – Creating User Defined Functions

# Overview

The existing Marketing application includes some functions. Your manager has requested your assistance in creating a new function for formatting phone numbers. She also needs you to modify an existing function to improve its usability.

Before starting this lab, you should view **Module 3 – Creating User Defined Functions** in the course *Creating Programmatic SQL Database Objects*. Then, if you have not already done so, follow the instructions in the **Getting Started** document for this course to set up the lab environment.

If you find some of the challenges difficult, don’t worry – you can find suggested solutions for all of the challenges in the **Lab Solution** folder for this module.

# What You’ll Need

To complete the labs, you will need the following:

* An Azure SQL Database instance with the AdventureWorksLT sample database. Review the Getting Started document for information about how to provision this.
* The lab files for this course

# Challenge 1: Format Phone Numbers

Your manager has noticed that different users tend to format phone numbers that are entered into the database in different ways. She has asked you to create a function that will be used to format the phone numbers. You need to design, implement, and test the function.

## Review the Design Requirements

Review the following design requirements for your stored procedure:

**Function Name:** FormatPhoneNumber (created in the dbo schema)

**Input Parameter:** PhoneNumberToFormat nvarchar(16)

**Return Value:** nvarchar(16)

**Rules to apply in formatting:**

* Any phone number beginning with the international dialing code (ie: a + sign), should be left unformatted.
* Phone numbers that contain 10 digits should be formatted as: (XXX) XXX-XXXX
* Phone numbers that contain 8 digits should be formatted as: XXXX-XXXX
* Phone numbers that contain 7 digits should be formatted as: XXX-XXXX
* Phone numbers that contain 6 digits should be formatted as: XXX-XXX
* All other characters should be stripped out
* Phone numbers that have different numbers of digits should have only the digits returned ie: (9234) 2345-2342 should be returned as 923423452342.

## Design and Create the Function

Design and create the function for reformatting phone numbers.

## Test the Function

Execute the FormatPhoneNumber function to ensure that the function correctly formats the phone number

## Review the Reports.GetProductsAndModels Stored Procedure Specification

1. Review the following design requirements for your stored procedure:

|  |  |
| --- | --- |
| Stored Procedure: | Reports.GetProductsAndModels |
| Input Parameters: | None |
| Output Parameters: | None |
| Output Columns: | ProductID, Name, ProductNumber, SellStartDate, SellEndDate and Color (from SalesLT.Product), ProductModelID (from SalesLT.ProductModel), Description (from SalesLT.ProductDescription). |
| Output Order: | ProductID, ProductModelID |
| Notes: | For descriptions, return the Description column from the SalesLT.ProductDescription table. |

## Create the Reports.GetProductsandModels Stored Procedure

1. Design, implement, and execute the stored procedure in accordance with the design specifications.

# Challenge 2: Create Parameterized Stored Procedures

In this exercise, you will create a stored procedure to support one of the new reports.

## Review the Reports.GetProductsByColor Stored Procedure specification

1. Review the following design requirements for your stored procedure:

|  |  |
| --- | --- |
| Stored Procedure | Reports.GetProductsByColor |
| Input parameters | @Color (same data type as the Color column in the Production.Product table) |
| Output parameters | None |
| Output columns | ProductID, Name, ListPrice (returned as a column named Price), Color, and Size (from Production.Product) |
| Output order | Name |
| Notes | The procedure should return products that have no Color if the parameter is NULL. |

## Create the Reports.GetProductsByColor Stored Procedure

1. Design and create the Reports.GetProductsByColor stored procedure.
2. Execute the Reports.GetProductsByColor stored procedure with a color of ‘Blue’.
3. Execute the Reports.GetProductsByColor stored procedure with a color of NULL.