Clariecia Groves

August 24, 2020

IT FDN 130 A - Foundations of Database Management

Assignment 07

https://github.com/CGroves3-UW/DBFoundations-Module07

Creating Functions in a Database

Introduction

For this assignment, I learned more about how to create databases, joins, views and functions. Joins are helpful for extracting data from multiple tables. Views and functions allow you to save simple and complex select statements so that they can be stored into a database. These options provide an easy way to execute saved code using simple select statements which makes it easier for business users to access various reports and data. Built-in and custom functions are presented in this module along with scalar, inline, and multi-statement functions.

SQL User-Defined Functions

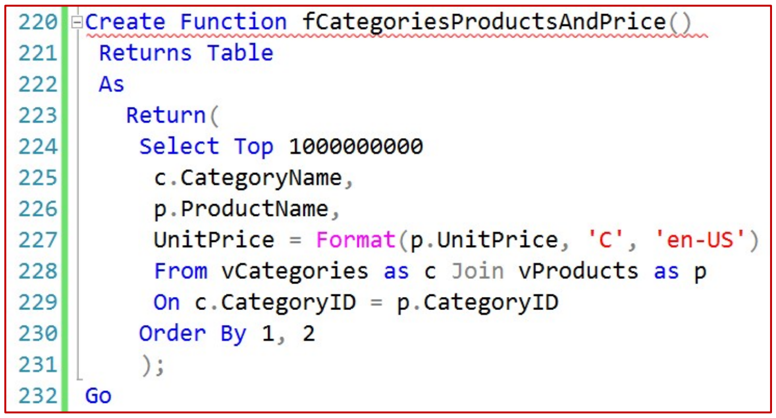
Explain when you would use a SQL UDF.

SQL user-defined functions (UDFs) are blocks of code that perform specific tasks defined by the user. UDFs accept parameters, perform actions, and return the results as a value. The return value can either be a single scalar value or a result set (https://docs.microsoft.com/en-us/sql/relational-databases/user-defined-functions/user-defined-functions?view=sql-server-ver15#:~:text=Like%20functions%20in%20programming%20languages,value%20or%20a%20result%20set., 2020).

The advantages of using UDFs are as follows:

- They allow for faster execution.
- They help to automate repetitive tasks.
- They allow for shorter and simpler written code when performing complex tasks.
- They can be used in a Select, Where or Case statement.

In this assignment we had separate tables for products and categories. The products table contained the ProductID, ProductName, CategoryID, and UnitPrice. The categories table contained the CategoryID and CategoryName. Figure 1 provides example code for creating a function that shows a list of category and product names along with the unit price of each product.



```
Create Function fCategoriesProductsAndPrice()
 Returns Table
 As
   Return(
    Select Top 1000000000
     c.CategoryName,
     p.ProductName,
     UnitPrice = Format(p.UnitPrice, 'C', 'en-US')
     From vCategories as c Join vProducts as p
     On c.CategoryID = p.CategoryID
    Order By 1, 2
    );
 Go
```

Figure 1: SQL UDF with Join on the Categories and Products Views

Scalar, Inline, and Multi-Statement Functions

Explain the differences between Scalar, Inline, and Multi-Statement Functions.

In this section, we will highlight some of the differences between scalar, inline, and multi-statement functions.

- A scalar function returns a single (scalar) value as an expression.
- An inline function returns a result set as opposed to a single scalar value.
- A mult-statement function returns a result set with a more powerful result.

Figure 2 provides a table comparison of the similar features and description of scalar, inline, and multi-statement functions.

Function	Similar Features	Description
Scalar	Similar to Built-In Functions	Returns a single value
Inline	Similar to Views with Parameters	Returns a table as the result of a single Select statement
Multi-Statement	Similar to Stored Procedures	Returns a new table as the result of Insert statements

Table 1: Similar Features and Description of Scalar, Inline, and Multi-Statement Functions (https://slideplayer.com/slide/10623411/, 2020)

Summary

After completing this module which included reading supplemental websites, watching videos, and practicing exercises, I was able to successfully create functions in a database. This written assignment demonstrates my knowledge of joining tables, creating views, and user-defined functions. I am excited to continue learning more about databases in future modules and hope to build my own database in practice.