Viet Nguyen

March 4, 2021

IT FN 130A

**Views**

Introduction:

My analysis in this paper will highlight situations in which one would use User Defined Functions (UDFs), transitioning to explaining the difference between scalar, inline, and multi-statement functions.

**User-Defined Functions**

Microsoft defines User-Defined Functions as “routines that accept parameters, perform an action, such as a complex calculation, and return the result of that action as a value. The return value can either be a single scalar value or a result set.”

A user would use UDFs to encourage consistency; the ability to use a common function to calculate fields such as Extended Price rather than having individuals create the equation themselves. Functions then cut down on errors which subsequently leads to improved data quality in the database. Functions also reduce the need to repeat code, which also leads to reduced errors due to not having to repeat oneself.

Create Database VNguyen01

Use VNguyen01;

go

Create Function dbo.MultiplyValues2(@Value1 Float, @Value2 Float)

Returns Float

As

Begin

Return(Select @Value1 \* @Value2);

End

go

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

Create table dbo.SalesDetails

( SalesId int, SalesLineItemId int

, ProductId int

, SalesPrice money

, SalesQty int,

Primary key(SalesId, SalesLineItemID)

);

go

Insert Into dbo.SalesDetails

(SalesId,SalesLineItemId,ProductId,SalesPrice,SalesQty)

Values

(1,1,1001,$19.99,10)

,(1,2,1002,$10.00,5)

Go

Select

SalesId

,SalesLineItemId

,ProductId

,SalesPrice

,SalesQty

,dbo.MultiplyValues2(SalesPrice,SalesQty) as ExtendedPrice

From dbo.SalesDetails

The above code would produce:



UDFs can also be used to enforce check constraints across multiple tables; something that cannot be done with simple code.

**Scalar, Inline, and Multi-statement Functions**

Scalar functions return a single scalar value as an expression defined in the Return clause.

The function below would return a single value, multiplying Value1 and Value2

Create Function dbo.MultiplyValues2(@Value1 Float, @Value2 Float)

Returns Float

As

Begin

Return(Select @Value1 \* @Value2);

End

go

Inline scalar functions return a scalar value as the result of a single statement.

Multi-statement functions – compared to inline scalar functions – returns a table as an output rather than a single expression.

From our homework assignment – notice the **Returns Table** rather than **Return Float** above:

Create Function dbo.fProductInventoriesWithPreviousMonthCountsWithKPIs (@CountVsPreviousCountKPI int)

Returns Table

As

Return(Select ProductName, InventoryDate, InventoryCount, PreviousMonthCount, CountVsPreviousCountKPI from vProductInventoriesWithPreviousMonthCountsWithKPIs

where CountVsPreviousCountKPI=@CountVsPreviousCountKPI)

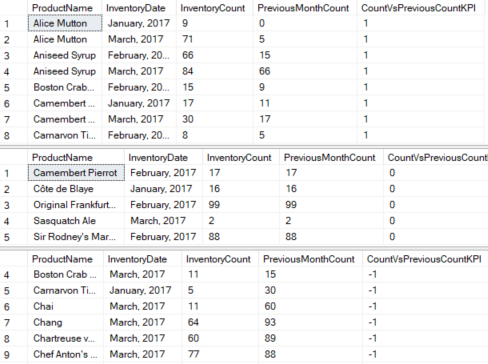
go

Select \* From fProductInventoriesWithPreviousMonthCountsWithKPIs(1);

Select \* From fProductInventoriesWithPreviousMonthCountsWithKPIs(0);

Select \* From fProductInventoriesWithPreviousMonthCountsWithKPIs(-1);

Go



**GitHub Link to Database**

<https://github.com/nguyvie92/DBFoundations-Module07>

**Bibliography:**

* “User-Defined Functions - SQL Server.” SQL Server | Microsoft Docs, docs.microsoft.com/en-us/sql/relational-databases/user-defined-functions/user-defined-functions?view=sql-server-ver15.