Name: Stan Godyn

Date: November 25, 2025

Course: Foundations of Databases & SQL Programming

https://github.com/godstaUW/DBFoundations

Assignment 7 – Functions

Introduction

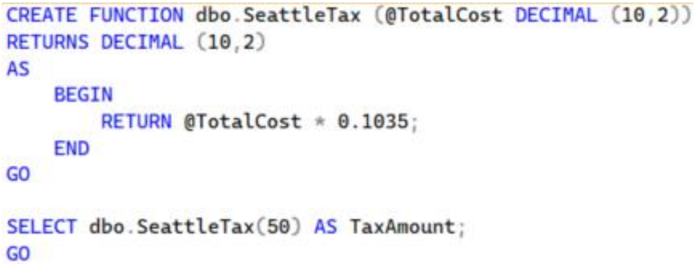
User-Defined Functions (UDFs) let users create custom, reusable SQL functions for specific operations. The three types—scalar, inline, and multi-statement table functions—will be discussed, compared, and contrasted.

User-Defined Function (UDF)

A user employs a UDF when built-in functions lack the specific operation or calculation needed to be performed. The UDF contributes additional capabilities to help simplify queries, reduce code repetition, and make complex logic easier to manage.

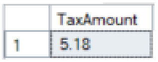
Scalar, Inline, & Multi-Statement Table Functions

There are three types of UDFs. **Scalar** UDFs return a *single value*, and are beneficial for its consistency, reusability and simplification. An example involves calculating the sales tax for a $50 purchase from Pike’s Place Market in Seattle.



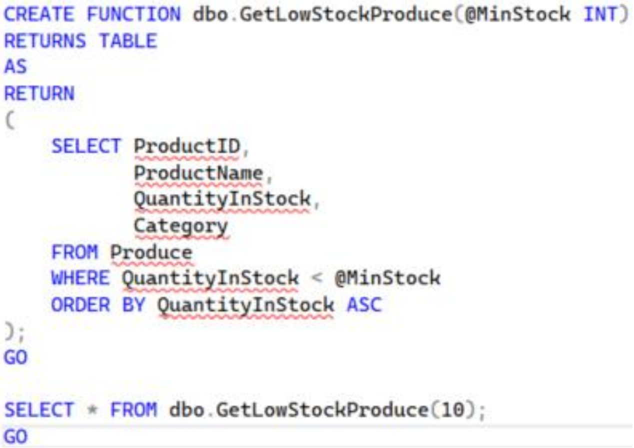
```
CREATE FUNCTION dbo.SeattleTax (@TotalCost DECIMAL (10,2))
RETURNS DECIMAL (10,2)
AS
    BEGIN
        RETURN @TotalCost * 0.1035;
    END
GO

SELECT dbo.SeattleTax(50) AS TaxAmount;
GO
```

When $50 is spent shopping for gifts, the function call will return the amount paid as tax (5.18).



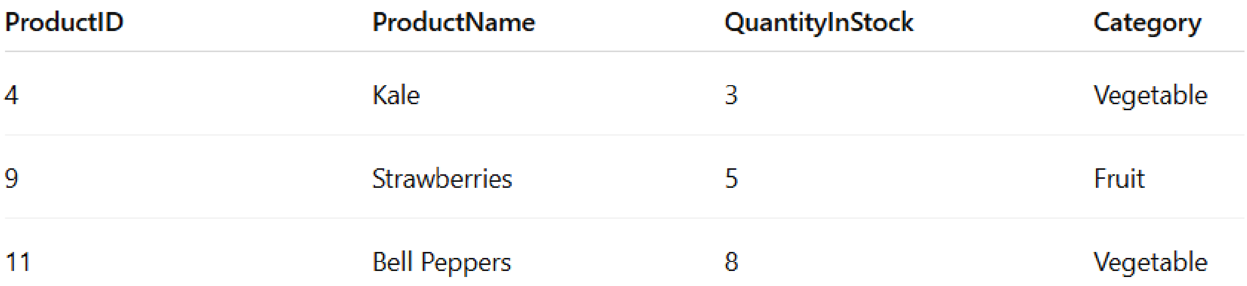
	TaxAmount
1	5.18

Inline UDFs return a *table* using a single SELECT statement, so it acts like a view to quickly produce results. Like the scalar UDF, it is reusable and simple to maintain. In this example inline UDF, a list of products is returned when a minimum quantity threshold is breached, flagging that replenishment is needed:



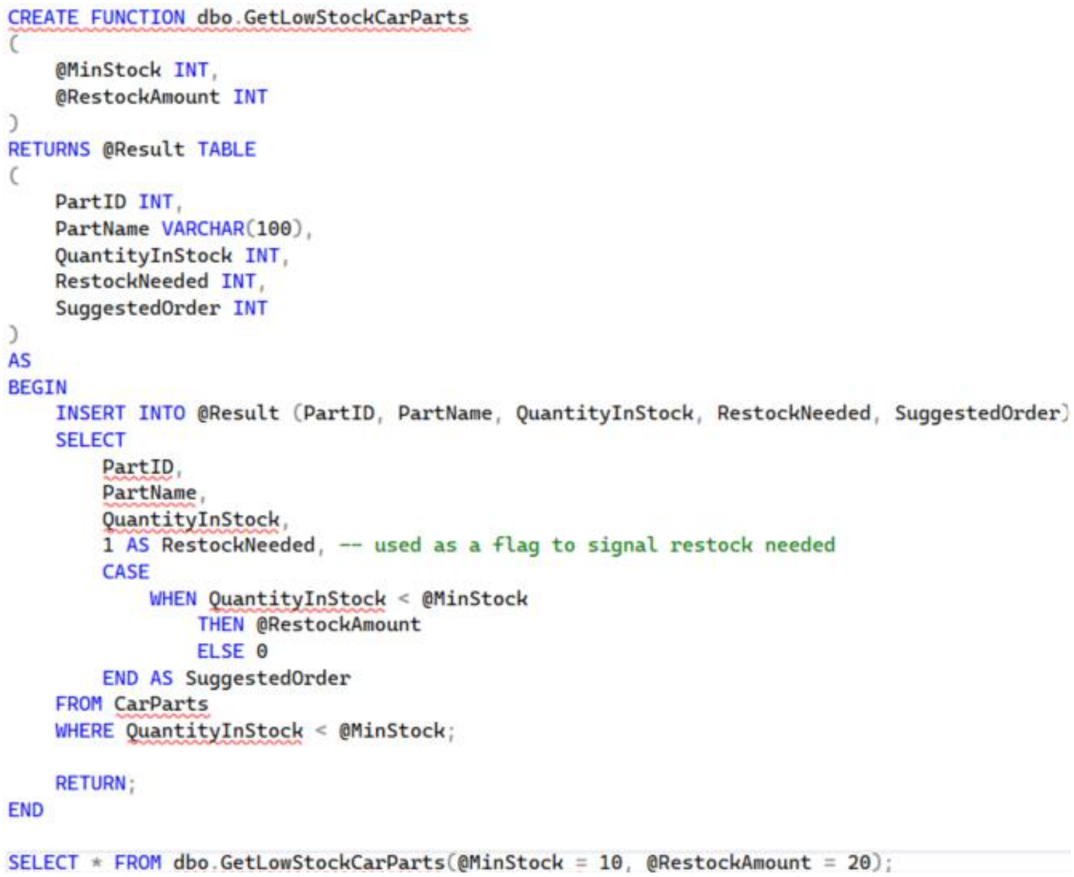
```sql
CREATE FUNCTION dbo.GetLowStockProduce(@MinStock INT)
RETURNS TABLE
AS
RETURN
(
    SELECT ProductID,
           ProductName,
           QuantityInStock,
           Category
    FROM Produce
    WHERE QuantityInStock < @MinStock
    ORDER BY QuantityInStock ASC
);
GO

SELECT * FROM dbo.GetLowStockProduce(10);
GO
```

A sample output could look like the following for a produce stand:



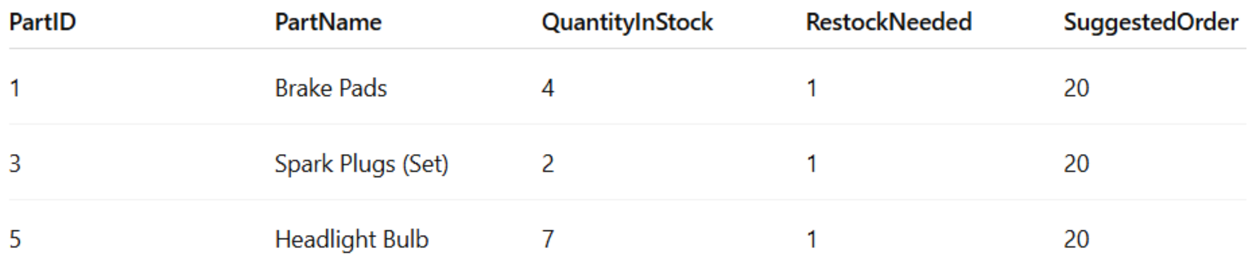
ProductID	ProductName	QuantityInStock	Category
4	Kale	3	Vegetable
9	Strawberries	5	Fruit
11	Bell Peppers	8	Vegetable

A **Multi-Statement Table-Valued Function** (MSTVF) returns tables that are more complex, such as involving multiple SQL statements involving inserts, calculations, or conditional statements. It can help when needing to perform multiple transformative steps before producing a table to return. Following the previous example, using car parts instead of produce, this MSTVF flags when parts fall below a minimum threshold level.



```
CREATE FUNCTION dbo.GetLowStockCarParts
(
    @MinStock INT,
    @RestockAmount INT
)
RETURNS @Result TABLE
(
    PartID INT,
    PartName VARCHAR(100),
    QuantityInStock INT,
    RestockNeeded INT,
    SuggestedOrder INT
)
AS
BEGIN
    INSERT INTO @Result (PartID, PartName, QuantityInStock, RestockNeeded, SuggestedOrder)
    SELECT
        PartID,
        PartName,
        QuantityInStock,
        1 AS RestockNeeded, -- used as a flag to signal restock needed
        CASE
            WHEN QuantityInStock < @MinStock
                THEN @RestockAmount
                ELSE 0
        END AS SuggestedOrder
    FROM CarParts
    WHERE QuantityInStock < @MinStock;

    RETURN;
END

SELECT * FROM dbo.GetLowStockCarParts(@MinStock = 10, @RestockAmount = 20);
```

If we suppose we have 4 Brake Pads, 11 Air Filters, 2 sets of Spark Plugs, 18 Oil Filters and 7 Headlight Bulbs, and our minimum stock quantity is to be 10, the following example produces a table for parts that need to be reordered.



PartID	PartName	QuantityInStock	RestockNeeded	SuggestedOrder
1	Brake Pads	4	1	20
3	Spark Plugs (Set)	2	1	20
5	Headlight Bulb	7	1	20

Summary

UDFs enable users to define their own reusable SQL functions for particular operations. This includes scalar functions, inline table-valued functions, and multi-statement table-valued functions. Each function were described for their appropriate applications and the benefits for their usage.