Name: HuiHui Chen

Date: November 26, 2022

IT FDN 130 A: Foundations Of Databases & SQL Programming

Assignment 07

Jessicachen0418/DBFoundations-Module07: Module 07 Assignments (github.com)

Functions

Introduction

This week, I learned the SQL built-in functions and how to create my own custom functions, and there are two types of basic UDFs (User Defined Functions); functions that return a single value (Scalar-valued functions) and functions that return a table of values (Table-valued functions). For the TVFs, there are also two types: in-line vs. multi-statement TVFs. Below 2 topics cover all the UDFs we have learned for this week.

Topic 1: Explain when you would use a SQL UDF.

- When built-in functions can't meet your needs then we can use UDFs (User Defined Functions) to create our very own custom functions. For example, if we want to perform some complex calculations, then we can place them in a separate method and store it in the database. Whenever we need the calculation, we can call it.
- There are two basic types of UDFs; functions that return a single value (Scalar-valued functions) and functions that return a table of values (Table-valued functions).

Topic 2: Explain the differences between Scalar, Inline, and Multi-Statement Functions.

- Scalar (Single Value) Function allows you to bring back a single value. That contrasts with a table-valued function, which returns a result set in the form of a table.
  - The basic syntax of Scalar Function

```
CREATE FUNCTION Function_Name(@ Paratemeter 1 DataType, @ Paratemeter 2
DataType,… @ Paratemeter N DataType)
RETURNS  Return_Datatype
AS
BEGIN
    --Function Body
    Return  Return_Datatype
```

END

--Syntax for calling a scalar function

Select **dbo**. <Function_Name> (Value)

- Inline Table -Valued Function (TVF) allows you to bring back a result set **in the form of a table**. In contrast with Scalar functions, we specify TABLE as return type, and the function body is NOT enclosed between BEGIN and END block.
  - ➢ The basic syntax of Inline TVF

```
CREATE FUNCTION Function_Name
(
@ Paratemeter 1 DataType
, @ Paratemeter 2 DataType
,…
,@ Paratemeter N DataType
)
RETURNS  TABLE
AS
RETURN (Select_Satement)
```

--Syntax for calling an Inline TVF

Select * From Function_Name (Value)

- The Multi-Statement Table Valued Function in SQL Server is the same as the Inline Table-Valued Function means it is also going to returns a table as an output but with the following differences.
  1. The Multi-Statement Table-Valued Function body can **contain more than one statement**. In Inline Table-Valued Function, it contains only a single Select statement prepared by the return statement.
  2. In Multi-Statement Table-Valued Function, the structure of the table returned from the function is defined by us. But, in Inline Table-Valued Function, the structure of the table is defined by the Select statement that is going to return from the function body. (Above reference Multi-Statement Table Valued Function in SQL Server - Dot Net Tutorials)
     - ➢ The basic syntax of Multi-Statement TVF

```
CREATE FUNCTION Function_Name
(
@ Paratemeter 1 DataType
, @ Paratemeter 2 DataType
,…
,@ Paratemeter N DataType
)
RETURNS  @ TableVarible TABLE (
Column1 DataType
, Column2 DataType
,…
, ColumnN DataType
)
AS
BEGIN
    Function Body
    RETURN
END
```

--Syntax for calling an Inline TVF

Select * From Function_Name (Value)

Summary

SQL Functions are very handled tools to retrieve information from a database, and if we have to repeatedly write intense SQL scripts to perform the same task then we can create a User Defined function that performs that similar task seamlessly.