Lauren Ferrier
February 28, 2023
IT Foundations of Database Management
Git Hub Link
Assignment 07

SQL Functions

Introduction to This Week's Topics

Last week we explored creating Views within our database, which permitted us to understand how our databases will continue to scale from not only its stored data, but also from its number of users. This week we are taking this theme further by diving into Functions. As the data and questions that prompt us to review our data grow in complexity, the necessity of Functions increases. Similarly, as our workload continues forward, our need to speedily call down and store these Functions additionally grows. This why understanding all of SQL's various Function types and the ability to implement User Defined Functions are key components of this week's module, which we will explore further below.

Documenting Module Knowledge

Questions for our consideration:

1. ***Explain when you would use a SQL UDF.***
   Firstly, diving into the basics of Functions in SQL, Functions are predefined formulas that combine one or more arguments as an input that is processed for a returned output. In other words, a user is often looking to preform a more complex calculation that is transforming data saved within their database (MOD07Notes.DOCX 2023, PG 1). Naturally, depending on the scope of work a user is undertaking, these functions can become complex and numerous, which is when a User Defined Function (UDF) can become handy. A UDF often is several functions stored as a single defined function, and it has a variety of benefits, such as: preventing a user from rewriting the same logic numerous times, increasing query speed against your database, and separates more complex calculations from regular queries for easier understanding and troubleshooting (MOD07Notes.DOCX 2023, PG 17 -19).

2. ***Explain are the differences between Scalar, Inline, and Multi-Statement Functions.***
   There are three main types of User Defined Functions, which are as follows:
   - *Scalar*: This returns a single value. For example, a function that processes two integer values and then subsequently returns the difference of the two (MOD07Notes.DOCX 2023, PG 17).

- *Inline*: Also known as table-valued functions, this returns a table data type. For instance, if a function processes one datapoint to then return all the subsequently associated datapoints in its table (MOD07Notes.DOCX 2023, PG 17 - 18). This is one step beyond a Scalar function since we are returning not only the calculated data, but additionally associated datapoints as well.

- *Multi-Statement*: While similar to Inline Functions since a table of data will be returned, Multi-Statement Functions can contain more than one statement while an Inline Function cannot. Additionally, the user defines the structure of the returned table while an Inline Function has its table determined the by the Select statement returned from the function body (Dot Net Tutorials, 2022).

Summary

This week's module had similar themes to modules four and five where we were processing and joining data to uncover questions, patterns, and similarities within the tables of data in our databases. However, we are taking this further by escalating our scope of work. Now we are looking to answer questions that involve more directly transforming data via calculations. This is where functions come into play, and as these functions grow in complexity, the need to use multiple at time while staying organized becomes a necessity. Subsequently, User Defined Functions come into play as a method of achieving this streamlined approach to our SQL queries as our workload advances and we additionally become more mature database administrators!