Dylan Schechtel
01 March 2022
Foundations of Databases and SQL Programming
Assignment 07
https://github.com/dschechtel-uw/DBFoundations-Module07

Functions

Introduction

This writeup will include a general overview of UDF use cases and the differences between scalar, inline, and multi-statement SQL functions. SQL functions can be placed into two broad categories – built-in and custom-built. Built-in functions are those common, frequently utilized actions like GetDate(), Count(), and Sum() that are delivered in a database out of the box. Database users also have the option to build one-off functions that serve their own unique needs and processes. These functions are referred to as User Defined Functions (UDFs) and are stored in the database in the same way that views and stored procedures are. Functions are also characterized into sub-categories based on more granular details related to the results they produce and their component parts.

SQL UDF Use Cases

There are a number of different instances in which using a UDF is necessary or appropriate. The overarching reasoning behind the use of UDFs is similar to that behind the use of stored procedures. If you know you will need to perform the same action or evaluate the same expression repeatedly in your work, it is much more efficient to build a UDF to invoke rather than rewrite complicated SQL every time the action is needed.

UDFs are also utilized in ETL Processing, which is described in the Functions-04 video as "the process of extracting data from a source, transforming it, and then loading it into a destination" (https://www.youtube.com/watch?v=URSjRbN1rCQ&list=PLfycUyp06LG9wAGPKBZ7poKBcbDZrmXpi&index=3&ab_channel=RandalRoot) (External site). UDFs can be leveraged to alter elements of data when it is received from a source to make it more functional for end use. In an example scenario, customer name and phone number data is received from a source and is organized into two columns. To make this data more useable in a normalized database table, a UDF could be used to separate customer first and last names into two columns and to isolate the area codes from the rest of the phone number by dividing the source value into two separate values. These changes are necessary to meet the needs of end users and are simplified with UDFs.

UDFs can also be used as part of check constraints in a database table. The function serves to define the parameters that will be evaluated by the check constraint whenever data is inserted into or updated within a table. A UDF could be leveraged in a check constraint to allow or disallow an action in a table based on parameters defined in the function. An example of this is highlighted in the Functions-05 video when a UDF is leveraged to set a constraint that disallows the action of registering for a meeting after the meeting’s start time has passed.

Scalar, Inline, & Multi-Statement Functions

SQL functions can be classified as scalar, inline, or multi-statement based on both their code architecture and their output. A scalar function is a function that takes one or more parameters as input, evaluates an expression or statement using the given parameters, and returns one value per evaluation. These functions can be used to perform arithmetic or basic comparisons, but they can also be leveraged to perform more complex evaluations. Regardless of their complexity, however, scalar functions always return a single value when invoked.

An inline function is a type of simple table-valued function. Unlike a scalar function that returns a single value, table-valued functions return a set of rows in a table format after evaluating the parameters passed into the function. A multi-statement function is a more complex table-valued function that can contain multiple SQL statements in addition to conditional logic that determines the function’s output. Like inline functions, the results of multi-statement functions are returned in row format as if they were part of a temporary table.

Conclusion

SQL functions are valuable tools to have at your disposal when working with data and databases. User defined functions are particularly useful in that they allow database users to build custom function to fit their own needs and processes. In particular, the unique structure and characteristics of scalar, inline, and multi-statement functions increase the type of evaluations and actions that can be performed easily and repeatedly by those interacting with a database at any capacity.