Name: DBarreto

Date: 2/25/2023

Course: Foundation of Database and SQL Programming

GitHub : <https://github.com/daisybt99/Assigment07_DBarreto>

**Assignment 07: User Define Functions (UDF)**

**Introduction**

In this module we explore the different types of functions that can be used in SQL. We explore the complexity of the query and how the data can me manipulated as desired with the help of User Defined Functions (UDF). UDF have many benefits and can be used in many ways depending on the job. There are warnings when working with UDFs as they can get complicated and if not careful, the functions written can have side effects or mutate data in ways that was not intended.

1. ***Explain when you would use a SQL UDF.***

UDF Stands for User Defined Functions and it’s a series of functions that are written by the user to create a desired table from a database. There can be 4 different types of UDFs: Function with no arguments and no return value, function with no arguments and a return value, and function with arguments and no return value.

There is no exact syntax or format on how to write a UDF, as it varies from function to function, but typically all UDFs include: A name to invoke the function, an argument that are the inputs a function takes, an output or return statement and finally, the function body which is the code that the database uses to execute the provided input.

The best time to use a User Define Function is when there are one or more values from different tables in a join operation, where some type of calculation needs to be done and an aggregation returned. The advantages of using the UDFs or apply to your queries is that the define function can be used in a SELECT, WHERE or CASE STATEMENT. They also can be used to create joins. In conclusion, User Defined Functions (UDF) are simpler to invoke than stored procedures from inside another SQL statement.

1. ***Explain are the differences between Scalar, Inline, and Multi-Statement Functions.***

Scalar function, Inline table valued functions and multi-Statement table-value functions are User defined functions.

Scalar Functions: A scalar function accepts any number of parameters and returns one value. The term scalar differentiates a single, "flat" value from more complex structured values, such as arrays or result sets. This pattern is much like that of traditional functions written in common programming language. Some of the most common scalar functions are FORMAT() UCASE(), LCASE(), MID(), LENGTH(), ROUND(), ETC.

Inline Table-Valued Functions: This type of functions returns a result set, much like a view. However, unlike a view, functions can accept parameters. The inline function's syntax is quite simple. In the function definition, the return type is set to a table. A return statement is used with a select query in parenthesis.

Multi-Statement Table-Valued Functions: Multi-Statement functions can be used to do some very unique things outside the context of a standard SELECT statement. This type of function returns a table-type result set, but the table is explicitly constructed in script. This can be used to accomplish one of two things: either to process some very unique business logic by assembling a virtual table on the fly, or to duplicate the functionality of an inline function in a more verbose and compiled way. In sort, if you need to select records from an existing a result set, use an inline table-valued function.

**Conclusion**

UDFs are functions that allow you to create a function by using a SQL expression. All UDFs have an advantage as they make the program easier to understand, easier to maintain and easier to debug. Therefore, this module shows us the complexity of a query and the actions that can be done in one single query.