Aleksei Konorev

August 25, 2021

Foundations of Databases & SQL Programming

Assignment 07

<https://github.com/konorevalexey/DBFoundationsCourse>

# Assignment 07

## Introduction

In this module we got acquainted with SQL Functions. Functions is a powerful tool which allows us to apply DRY (don’t repeat yourself) principle in coding, provides us more flexibility in writing queries, and also enables some SQL trick like dynamic/parameterized table constraints.

## When I would use SQL UDF

UDF is basically a script which allows you to automate a routine and/or simplify your code. As in other languages, functions in SQL are used for repeatable actions, or just to make code more readable (and also easier to write code). I would use UDF as a building block for my query (table or scalar), OR when I want to apply a function to each row in the table (scalar). Table-type UDFs are similar to views, but there a few differences. Here are advantages of UDFs:

* Complex code is stored in one structure. You can later look at that structure as on the black box, where you’re only interested in passing appropriate values as parameters and the function will do the rest
* You can much easier test input parameters using IF or CASE, and even use loops in the functions. This is sometimes very hard (sometimes impossible) to simulate directly in SELECT statements
* Once you create a function, and after it’s properly tested, you don’t have to bother later is it working as expected and you’re avoiding a possibility to make an error because you’re not rewriting the same code over and over again (not to mention that you’ll use less time when not rewriting the same code)
* If you need to make changes to your code, you’ll do it in one place and it will reflect at every place this function is used (<https://www.sqlshack.com/learn-sql-user-defined-functions/>, 2021)

There are many great applications which would be hard/impossible to achieve without UDFs, for instance, you can use UDF as a default value or check constraint in the table as was shown in lecture notes.

## Differences between Scalar, Inline and Multi-Statement Functions

Scalar function will return just one value (scalar) given the input parameters. The biggest advantage of scalar functions is that it can be easily applied for each row in a table to generate powerful calculations. Inline (table-valued) functions or iTVF are technically similar to views and return a table based on a parameter value and the SELECT statement written in the function. Multi-Statement functions also return a table, but also has the following advantages:

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.

(<https://dotnettutorials.net/lesson/multi-statement-table-valued-function-in-sql-server/> , 2021)

## Summary

We took our first stab at SQL functions. There are much more to learn. Functions make our coding easier and faster (DRY principle), and also allow more powerful applications like MTVFs.