Assignment 6: SQL Views
Safaa Darwish
11/28/2021
IT FDN 130 A Au 21: Foundations Of Databases & SQL Programming
Instructor: Randal Root
GitHub Code for Module 07 (Link to External Site): https://github.com/Makhlutha/Module07

SQL Functions

I. Introduction

This article explores SQL Functions. Functions are powerful queries that help execute code in an efficient manner, specifically cutting back on re-writing code, creating concise queries that either return a table of values or a single value[1]. User Defined Functions (UDFs) is generally what we call a custom Function[2]. Functions use parameters to alter the query[3], depending on what we want to achieve -this can be very useful when you know that you may need to alter code in future iterations of your program. This article also explores Scalar, Inline and, MultiStatment Functions.

II. SQL User Defined Functions

User Defined Functions (UDFs) are custom created Functions -that is, the functions that we write ourselves. We create these functions ourselves and then call them when we need them to execute a task in our code. When using Tabular UDFs (such as, Inline and MultiStatement) in MySQL, it is important to use a two part name in order to make sure that you are pointing to the correct table (i.e. using 'dbo.' Before the name of the function)[4]. For tabular functions, you do not need to use 'Begin' and 'End' in your script while in a Scalar, you do need to use these statements[5].

III. Function Types: Scalar, Inline, MultiStatement

A Scalar Function returns a single value as an expression[6]. Unlike a Function that returns a table, using parameters on Scalars is very simple[7] and allows us to update and run the code when we need. Inline Functions are basically Single Statement functions, and Inlines contain a single statement. Multistatement UDFs contain multiple statements. Both Inline and MultiStatement Functions return tables. Unlike the Scalar value where you simply call the function, you must treat the Inline and MultiStatement UDFs as tables in order to call them. In both instances, you create a table. In an Inline UDF, you would have a single statement altering the table. In a MultiStatement UDF, you would create the table and then have multiple statements setting parameters for how the table can be interacted with and altered.

[1] Root, Randal, *Module 06 Notes,* 11*,* Module 6, IT FDN 130 A Au 21: Foundations Of Databases & SQL Programming, University of Washington 2021
[2] Id.
[3] Id. at 12
[4] Root, Randal, *Functions-02 User Defined Functions UDFs*, at 00:02:06, IT FDN 130 A Au 21: Foundations Of Databases & SQL Programming
University of Washington 2021, Link to External Website: https://youtu.be/XEiQ3M2LhU4.
[5] Id. at 00:02:59
[6] Root, Randal, *Module 06 Notes,* 13*,* Module 6, IT FDN 130 A Au 21: Foundations Of Databases & SQL Programming, University of Washington 2021
[7] Id.

Assignment 6: SQL Views
Safaa Darwish
11/28/2021
IT FDN 130 A Au 21: Foundations Of Databases & SQL Programming
Instructor: Randal Root
GitHub Code for Module 07 (Link to External Site): https://github.com/Makhlutha/Module07

IV. Summary

UDFs are very powerful and if used correctly, can help immensely in creating concise code and faster programs. If the code within the functions is sound, there will be far less work to do in writing new script every time you want a task to be accomplished.

Fig. 1. This is some person's imagining of what the Improbability Drive might look like. The Improbability Drive probably does not look like this. This image is being used to experiment with Markdown language in Github. *Anonymous*, drulpaldays.org, Date accessed 11/30/2021, Webpage, Link to External Website: http://szeged2014.drupaldays.org/sites/default/files/styles/large/public/def_d_infinite_drive.png?itok=7AcRrFqR