Jackson Bauer

May 30th, 2022

Foundations Of Databases & SQL Programming

Assignment 07

https://github.com/jgcbUW/DBFoundations-Module07/blob/main/docs/index.md

SQL Functions

Introduction

Functions in SQL come in all shapes and sizes. There are functions that can calculate or extrapolate important information, ones that deliver a table with specific requests, and others that can create a fresh (and I believe temporary) table to present data in a specific package. Creating and using functions is an important steppingstone to mastering SQL.

The usefulness of UDFs

For those that aren't aware, UDFs are defined as User Defined Functions. Luckily, these are rather self-descriptive. UDFs can be created for many sorts of purposes, but they aren't without their limitations. When working with code over and over, it is always worthwhile to save that code for replicability rather than typing it over and over. However, functions are a good choice if you need to retrieve specific information mid-statement, return a table with specific parameters, and many more. I particularly liked the example discussed in WiseOwl's discussion on the topic of inline functions.

```
-- show all of the films lasting more than 3 hour 10 minutes
SELECT * from dbo.fnFilmsByDuration(190)
```

Figure 1 - WiseOwl Function https://www.wiseowl.co.uk/blog/s347/in-line.htm

This code will simply retrieve a table that has values with these specific restrictions! Very nifty for ease-of-use coding.

Functions in all their variety

There are three main types of functions in SQL: Scalar functions, inline functions, and table valued functions. Each one of these gets progressively more complicated, but nothing too hard to understand. Scalar functions are capable of using a large number of parameters and return only a single value of any data type. This does not include sets or tables, hence the name. These functions are useful to change datatypes, crunch calculations, aggregate data to one metric, etc. Inline functions are inherently similar, but they can return tables and arrays instead. So, functionally, these mirror views in their capabilities but they can use parameters as well. Lastly, table valued functions (TBFs) are the most complex. They essentially create a temporary table to return. Instead of simply using select statements to return a table as inline functions would, TBFs create a table and insert values into them. I believe this table is a temporary table that sends nearly identically to how a select statement would send a table, but that is a theory I hop to test another day!

Summary

As with any good language, modularity is a very important concept to master. Reusing code is part of what programmers do on a regular basis, so being capable of defining functions, views, and soon stored procedures is one steppingstone toward fluidity in this area.