Name: Rhyss Garvida

Date: 5/23/2022

Course: IT FDN 130 A

Assignment 06

GitHub: https://github.com/rhyssg/DBFoundations

Storing and Simplifying Code

Introduction

The goal for query building is simplification. And often, the more Server Query Language (SQL) code is written, the more difficult it is to maintain. The Create View, Function, and Stored Procedure statements allows for simplification in that it centralizes frequently used code for quick recall, allowing for fast retrieval of stored queries within those statements.

Create View, Function, and Stored Procedure

The Create View statement is “a virtual table based on the result-set of an SQL statement” (W3 Schools, https://www.w3schools.com/sql/sql_view.asp, 2022) (External Site). The query syntax selects from one or more present tables within the database and allows for data view without exposing risks to the data integrity of the database (Figure 1). Simply, the statement “does not hold any data and does not exist physically in the database” (SQL Shack, https://www.sqlshack.com/sql-view-a-complete-introduction-and-walk-through/, 2019) (External Site). It is an abstraction layer that enables the user to retrieve data from the original database tables and showcases complex query reporting i.e., think of it as having a copy of an original (Figure 2). This is advantageous in that it can be used to limit the visibility of data from the public or from certain groups, can aggregate disconnected rows from multiple tables with better detail and also it can simplify reporting only to those specific data retrieved from multiple tables. This would be used in instances when you have a complex select retrieval statement, and rather than repeating it in multiple instances within the code, it can be recalled and reported from a view statement for query use.

```
CREATE VIEW view_name AS
SELECT column1, column2, ...
FROM table_name
WHERE condition;
```

Figure 1: CREATE VIEW Syntax

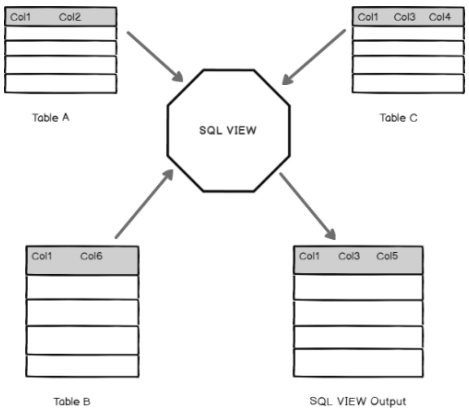

Figure 2: CREATE VIEW logic

In comparison, functions are “a set of SQL statements that perform a specific task” (SQL Shack, https://www.sqlshack.com/use-sql-server-built-functions-create-user-defined-scalar-functions/, 2017) (External Site). Similarly, to the other two storage statements, large repeated SQL scripts that perform the same task can be stored in a function statement. There can be multiple types of function statements where a built-in command or a custom-made function statement can be utilized. Its purpose is to accept “inputs in the form of parameters and returns a value” (SQL Shack, https://www.sqlshack.com/use-sql-server-built-functions-create-user-defined-scalar-functions/, 2017) (External Site). For example, a built-in function such as CONCAT() concatenates a group of strings that outputs “SQL is fun!” (Figure 3). In contrast, complexity can be created in user defined function statements as exemplified by “east_or_west” where it transforms a parameter based on the SQL code written within the function statement (Figure 4). As such, it is useful in instances where the data is transformed to add, subtract, multiply, divide and to perform any other custom-made instructions. Depending on the desired output, function statements can encapsulate queries that transform parameter values to return a one-to-one value.

```
SELECT CONCAT('SQL', ' ', 'is', ' ', 'fun!');
```

Figure 3: SELECT Syntax of built-in function CONCAT()

```
CREATE FUNCTION east_or_west (
    @long DECIMAL(9,6)
)
RETURNS CHAR(4) AS
BEGIN
    DECLARE @return_value CHAR(4);
    SET @return_value = 'same';
    IF (@long > 0.00) SET @return_value = 'east';
    IF (@long < 0.00) SET @return_value = 'west';

    RETURN @return_value
END;
```

Figure 4: Custom-Made User-Defined “east_or_west” Function Statement

Similarly, to the two above, a Stored Procedure “is a prepared SQL code that you can save, so the code can be reused over and over again” (W3 Schools, https://www.w3schools.com/sql/sql_stored_procedures.asp, 2022) (External Site). However, unlike views, it is distinct in that it “accepts the parameters and executes the SQL statements in the procedure, [and] returns the result set if any (SQL Shack, https://www.sqlshack.com/sql-server-stored-procedures-for-beginners/, 2019) (External Site). It is also more flexible than a more rigid structure like a function statement, where it can either return a value for a singular or multiple parameters (Figure 5, 6). Some of the benefits of using a Stored Procedure statement are that it can be easily modified, reduce network traffic, allow for reusability, increase security, and increase performance. It is useful in instances where there are large and repeatable SQL scripts that transforms data sets throughout the application. As such, it allows a user to recall the procedure statement to access or modify data within the database; and return outputs to the application while maintaining readability and upholding data integrity.

```
CREATE PROCEDURE SelectAllCustomers @City nvarchar(30)
AS
SELECT * FROM Customers WHERE City = @City
GO;
```

Figure 5: STORED PROCEDURE with a Singular Parameter

```
CREATE PROCEDURE SelectAllCustomers @City nvarchar(30), @PostalCode nvarchar(10)
AS
SELECT * FROM Customers WHERE City = @City AND PostalCode = @PostalCode
GO;
```

Figure 6: STORED PROCEDURE with Multiple Parameters

Summary

Overall, Create Views, Function, and Stored Procedure statements can be used to store large SQL scripts and minimize repetitions and bulkiness. These can be recalled based on use whether it be for query reporting, returning a transformed value or transforming multiple data within the application. As such, depending on the desired outcome, these statements enable a user to simplify and ease the readability of SQL scripts by calling in the user-defined variables containing instructions to display or transform the data within the database.