How to create a Stored Procedure, View and Function in SQL

by Schlafenhase

Summary: The following document aims to explain as clearly as possible the steps to follow to create a SP, View and Function in the SQL language.

To make the explanation easier to read and understand, both topics will be touched upon separately.

Explanations and steps to follow.

1.1) Stored Procedure: A stored procedure is a block of code written in the query that might need to be used more than once, so SQL offers the option to save and reuse it when necessary.

1.2) Steps to create a Stored Procedure:

- First, you need to open a new query in an existing database, in that query the syntax will be introduced in the way explained below.
- After having created the query, the first thing will be to define the system that we want to create a new Stored Procedure (We will call it SP from now on). For this task we must write "CREATE PROCEDURE" and just after that the name with which we want to name our SP.
  It should be noted that the command can also be shortened as "CREATE PROC" and the name of the SP, in both cases the results are the same.
- After having created our SP we have to declare the variables that are going to be used within it, which will function as the parameters that will be sent to the SP once it is called (In case the SP can work without needing to receive any parameter, skip this step).
  To declare the variables, the syntax "@VariableName" is used and right after that, the typeface of said variable is specified. In the case of more than one, they must be separated with commas.
- After declaring the variables it is necessary to write the reserved command "AS", this reserved word performs the action of a conditioner, conditions can be specified that, if met, would automatically trigger the procedure.

- After this comes the body of the SP, which begins by opening the "BEGIN" command and right after that you can start coding the action or actions you want to carry out said SP, it should be noted that the variables declared at the beginning (that is, the parameters that the SP is supposed to receive once executed) are totally valid to use within the "BEGING" either to make comparisons or declarations.
- After finishing defining everything that is needed from the SP, we proceed to finish with the last two commands, the first being the "END" command which closes the "BEGIN" command. The "END" command is used to ending with ";".
  Right after this it is important to add the command "GO" since without it the code cannot be executed in a fluid way. The function of this command is to "activate" the SP so that it is loaded once the code is parsed and compiled
- To execute an already created SP, you must run the command "EXEC" followed by the name assigned to the SP and the parameters (if they were defended) that it should receive.

2.1) Views: Views are components that allow users to see the information stored in a table, but without giving them access to that table. The amount of information viewed by users can be filtered and managed, allowing them to see only a specific amount of the total table

2.2) Steps to create a View:

- First, you need to open a new query in an existing database, in that query the syntax will be introduced in the way explained below.
- After having created the query, the first thing will be to define the system that we want to create a new View. For this task we must write "CREATE VIEW" and just after that the name with which we want to name our View.
- After creating the view, it is necessary to write the reserved command "AS", this reserved word performs the action of a conditioner, conditions can be specified that, if met, would automatically trigger the procedure.
- After this, a "SELECT" command is usually generated and specifies what data is going to be shown in the view. In other words, the information columns that the user will be able to see are specified, but it is not specified where the information is being obtained from.

- It should be noted that to increase the security of the views, the command "ENCRYPTION" is often added at the beginning, just after the name, to give more security and prevent it from being seen from which table the information is being extracted.

3.1) Functions: Functions are methods created by users to perform a certain action, usually related to data management and operations. Functions can be used to perform calculations between numerical values in a table, standardize columns in some tables, or obtain the percentage among a certain group of values

The main advantage of functions is that they are capable of returning data with certain type of information specified after performing operations or calculations desired by the user, this facilitates the work and administration when handling the information in some tables.

3.2) Steps to create a Function:

- First, you need to open a new query in an existing database, in that query the syntax will be introduced in the way explained below.
- After having created the query, the first thing will be to define the system that we want to create a new Function. For this task we must write "CREATE FUNCTION" and just after that the name with which we want to name our function.
- Having created our Function, open parentheses to declare the variables that are going to be used within it, which will function as the parameters that will be sent to the Function once it is called.
  To declare the variables, the syntax "@VariableName" is used and right after that, the typeface of said variable is specified. In the case of more than one, they must be separated with commas.
- After declaring the variables, we proceed to close the parentheses. Right after we must use the command "RETURN" followed by the type of data that we want our function to return at the end of its execution.
- After declaring the return command, it is necessary to write the reserved command "AS" just after it, this reserved word performs the action of a conditioner, conditions can be specified that, if met, would automatically trigger the procedure.

- After this comes the body of the Function, which begins by opening the "BEGIN" command and right after that you can start coding the action or actions you want to carry out said Function, it should be noted that the variables declared at the beginning (that is, the parameters that the Function is supposed to receive once executed) are totally valid to use within the "BEGIN" either to make comparisons or declarations.
- After finishing defining everything that is needed for the Function, we proceed to finish with the last command, being the "END" command that closes the "START" command. The "END" command is usually terminated with ";".

3.3) Example:

```
CREATE FUNCTION F_Example(
@Var1 float, --might be any type
@Var2 float
)
RETURNS FLOAT
AS
BEGIN
DECLARE @result float
SET @result = (@Var1 + @Var2)/2
RETURN @result
END;
```

4) Bibliography.

[1].SQL Stored Procedures. (2020). Retrieved 3 July 2020, from https://www.w3schools.com/sql/sql_stored_procedures.asp

[2].SQL Server CREATE VIEW - Creating New Views in SQL Server. (2020). Retrieved 3 July 2020, from https://www.sqlservertutorial.net/sql-server-views/sql-server-create-view/

[3] IBM Knowledge Center. (2020). Retrieved 3 July 2020, from https://www.ibm.com/support/knowledgecenter/SS6NHC/com.ibm.swg.im.dashdb.apdv.sqlpl.doc/doc/t0053767.html

schlafenhase