**What is a Stored Procedure?**

A stored procedure is a prepared SQL code that you can save, so the code can be reused over and over again.

So if you have an SQL query that you write over and over again, save it as a stored procedure, and then just call it to execute it.

You can also pass parameters to a stored procedure, so that the stored procedure can act based on the parameter value(s) that is passed.

**Stored Procedure Syntax**

CREATE PROCEDURE procedure\_name

AS

sql\_statement

GO;

**Introduction to MySQL CREATE PROCEDURE statement**

To create a stored procedure, you use the CREATE PROCEDURE statement.

Here’s the basic syntax of the CREATE PROCEDURE statement:

CREATE PROCEDURE sp\_name(parameter\_list)

BEGIN

statements;

END;

In this syntax:

First, specify the stored proceed name sp\_name after the CREATE PROCEDURE keywords.

Second, specify the parameter list (parameter\_list) inside the parentheses followed by the stored procedure’s name.

Third, write the stored procedure body that consists of one or more valid SQL statements.

If you create a stored procedure that already exists, MySQL will issue an error.

To prevent the error, you can add an additional clause IF NOT EXISTS after the CREATE PROCEDURE keyword:

CREATE PROCEDURE [IF NOT EXISTS] sp\_name ([proc\_parameter[,...]])

routine\_body;

Code language: CSS (css)

In this case, MySQL will issue a warning if you attempt to create a stored procedure whole name that already exists instead of throwing an error.

Note that the IF NOT EXISTS clause has been available since MySQL 8.0.29.

MySQL CREATE PROCEDURE statement example

We’ll use the products table in the sample database for the demonstration:

The following statements create a new stored procedure called GetAllProducts():

DELIMITER //

CREATE PROCEDURE GetAllProducts()

BEGIN

SELECT \* FROM products;

END //

DELIMITER ;

Code language: SQL (Structured Query Language) (sql)

How it works:

First, change the default delimiter to //:

DELIMITER //

Code language: SQL (Structured Query Language) (sql)

Second, use the CREATE PROCEDURE statement to create a new stored procedure. Since we changed the delimiter to //, we can use the semicolon (;) inside the stored procedure:

CREATE PROCEDURE GetAllProducts()

BEGIN

SELECT \* FROM products;

END //

Code language: SQL (Structured Query Language) (sql)

Third, change the delimiter back to the default delimiter, which is a semicolon (;):

DELIMITER ;

Code language: SQL (Structured Query Language) (sql)

Creating a stored procedure using MySQL client

First, connect to the classicmodels sample database using the mysql client:

C:\>mysql -u root -p classicmodels

Enter password: \*\*\*\*\*\*\*\*

Second, change the delimiter to //:

mysql> DELIMITER //

**Code language: JavaScript (javascript)**

**Third, type the following code to create the stored procedure:**

mysql> CREATE PROCEDURE GetAllProducts()

-> BEGIN

-> SELECT \* FROM products;

-> END //

Query OK, 0 rows affected (0.01 sec)

Code language: JavaScript (javascript)

**Finally, change the delimiter back to a semicolon:**

DELIMITER ;

Query:

-- Create a new database named "SampleDB"

CREATE DATABASE storedDB;

-- Switch to the new database

USE storedDB;

-- Create a new table named "Customers"

CREATE TABLE Customers (

CustomerID INT PRIMARY KEY,

CustomerName VARCHAR(50),

ContactName VARCHAR(50),

Country VARCHAR(50)

);

-- Insert some sample data into the Customers table

INSERT INTO Customers (CustomerID, CustomerName, ContactName, Country)

VALUES (1, 'Shubham', 'Thakur', 'India'),

(2, 'Aman ', 'Chopra', 'Australia'),

(3, 'Naveen', 'Tulasi', 'Sri lanka'),

(4, 'Aditya', 'Arpan', 'Austria'),

(5, 'Nishant. Salchichas S.A.', 'Jain', 'Spain');

-- Create a stored procedure named "GetCustomersByCountry"

CREATE PROCEDURE GetCustomersByCountry

@Country VARCHAR(50)

AS

BEGIN

SELECT CustomerName, ContactName

FROM Customers

WHERE Country = @Country;

END;

-- Execute the stored procedure with parameter "Sri lanka"

EXEC GetCustomersByCountry @Country = 'Sri lanka';

Note: You will need to make sure that the user account has the necessary privileges to create a database. You can try logging in as a different user with administrative privileges or contact the database administrator to grant the necessary privileges to your user account. If you are using a cloud-based database service, make sure that you have correctly configured the user account and its permissions.

Output:

CustomerName Contact Name

Naveen Tulasi