MySQL Stored Procedure

A procedure (often called a stored procedure) is a **collection of pre-compiled SQL statements** stored inside the database. It is a subroutine or a subprogram in the regular computing language. **A procedure always contains a name, parameter lists, and SQL statements**. We can invoke the procedures by using triggers, other procedures and applications such as [Java](https://www.javatpoint.com/java-tutorial), [Python](https://www.javatpoint.com/python-tutorial), [PHP](https://www.javatpoint.com/php-tutorial), etc. It was first introduced in MySQL **version 5**. Presently, it can be supported by almost all relational database systems.

1. DELIMITER &&
2. **CREATE** **PROCEDURE** procedure\_name [[IN | **OUT** | INOUT] parameter\_name datatype [, parameter datatype]) ]
3. **BEGIN**
4. Declaration\_section
5. Executable\_section
6. **END** &&
7. DELIMITER ;
8. **MySQL procedure parameter has one of three modes:**
9. **IN parameter**
10. It is the default mode. It takes a parameter as input, such as an attribute. When we define it, the calling program has to pass an argument to the stored procedure. This parameter's value is always protected.
11. **OUT parameters**
12. It is used to pass a parameter as output. Its value can be changed inside the stored procedure, and the changed (new) value is passed back to the calling program. It is noted that a procedure cannot access the OUT parameter's initial value when it starts.

**INOUT parameters**

It is a combination of IN and OUT parameters. It means the calling program can pass the argument, and the procedure can modify the INOUT parameter, and then passes the new value back to the calling program.

How to call a stored procedure?

We can use the **CALL statement** to call a stored procedure. This statement returns the values to its caller through its parameters (IN, OUT, or INOUT). The following syntax is used to call the stored procedure in MySQL:

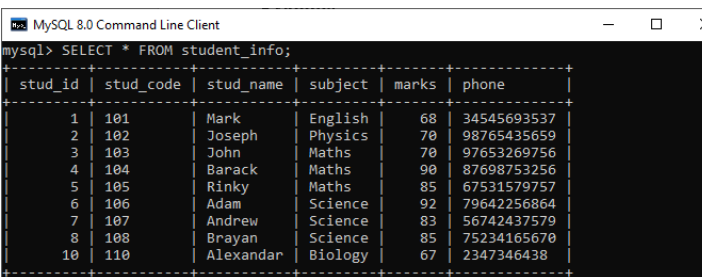
1. CALL procedure\_name ( parameter(s))

Example

Let us understand how to create a procedure in MySQL through example. First, we need to select a database that will store the newly created procedure. We can select the database using the below statement:

1. mysql> USE database\_name;

Suppose this database has a table named **student\_info** that contains the following data:



### **Procedure without Parameter**

Suppose we want **to display all records of this table whose marks are greater than 70** and count all the table rows. The following code creates a procedure named **get\_merit\_students**:

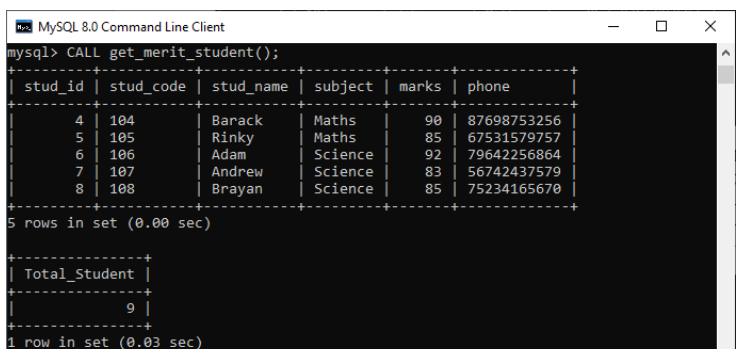
1. DELIMITER &&
2. **CREATE** **PROCEDURE** get\_merit\_student ()
3. **BEGIN**
4. **SELECT** \* **FROM** student\_info **WHERE** marks > 70;
5. **SELECT** COUNT(stud\_code) **AS** Total\_Student **FROM** student\_info;
6. **END** &&
7. DELIMITER ;

If this code executed successfully, we would get the below output:

Let us call the procedure to verify the output:

1. mysql> CALL get\_merit\_student();

It will give the output as follows:



### **Procedures with IN Parameter**

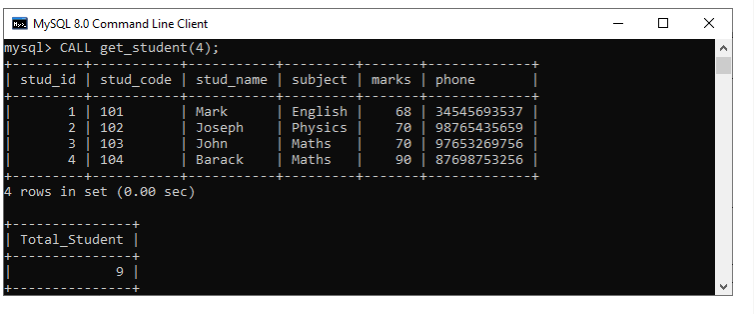
In this procedure, we have used the IN parameter as **'var1**' of integer type to accept a number from users. Its body part fetches the records from the table using a [**SELECT** statement](https://www.javatpoint.com/mysql-select) and returns only those rows that will be supplied by the user. It also returns the total number of rows of the specified table. See the procedure code:

1. DELIMITER &&
2. **CREATE** **PROCEDURE** get\_student (IN var1 **INT**)
3. **BEGIN**
4. **SELECT** \* **FROM** student\_info LIMIT var1;
5. **SELECT** COUNT(stud\_code) **AS** Total\_Student **FROM** student\_info;
6. **END** &&
7. DELIMITER ;

After successful execution, we can call the procedure as follows:

1. mysql> CALL get\_student(4);

We will get the below output:



### **Procedures with OUT Parameter**

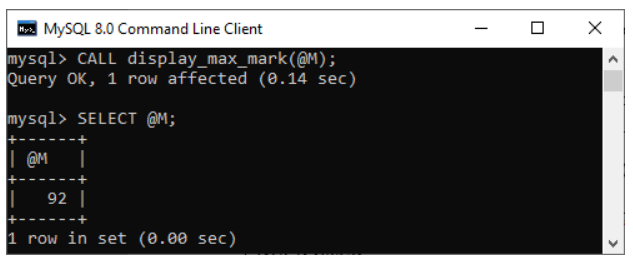
In this procedure, we have used the OUT parameter as the **'highestmark'** of integer type. Its body part fetches the maximum marks from the table using a **MAX() function**. See the procedure code:

1. DELIMITER &&
2. **CREATE** **PROCEDURE** display\_max\_mark (**OUT** highestmark **INT**)
3. **BEGIN**
4. **SELECT** **MAX**(marks) **INTO** highestmark **FROM** student\_info;
5. **END** &&
6. DELIMITER ;

This procedure's parameter will get the highest marks from the **student\_info** table. When we call the procedure, the OUT parameter tells the database systems that its value goes out from the procedures. Now, we will pass its value to a session variable **@M** in the CALL statement as follows:

1. mysql> CALL display\_max\_mark(@M);
2. mysql> **SELECT** @M;

Here is the output:

\

### **Procedures with INOUT Parameter**

In this procedure, we have used the INOUT parameter as **'var1'** of integer type. Its body part first fetches the marks from the table with the specified **id** and then stores it into the same variable var1. The var1 first acts as the IN parameter and then OUT parameter. Therefore, we can call it the INOUT parameter mode. See the procedure code:

1. DELIMITER &&
2. **CREATE** **PROCEDURE** display\_marks (INOUT var1 **INT**)
3. **BEGIN**
4. **SELECT** marks **INTO** var1 **FROM** student\_info **WHERE** stud\_id = var1;
5. **END** &&
6. DELIMITER ;

After successful execution, we can call the procedure as follows:

1. mysql> **SET** @M = '3';
2. mysql> CALL display\_marks(@M);
3. mysql> **SELECT** @M;

We will get the below output:

