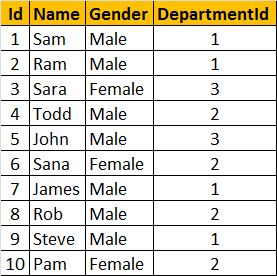
**Stored procedures**

A stored procedure is group of T-SQL (Transact SQL) statements. If you have a situation, where you write the same query over and over again, you can save that specific query as a stored procedure and call it just by it's name.

  
  
  
**Creating a simple stored procedure without any parameters**:

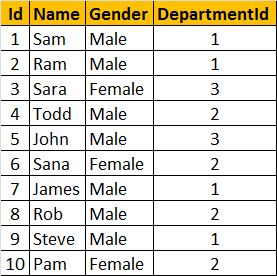
This stored procedure, retrieves Name and Gender of all the employees. To create a stored procedure we use, **CREATE PROCEDURE** or **CREATE PROC** statement.  
  
Create Procedure uspGetEmployees  
as  
Begin  
  Select Name, Gender from tblEmployee  
End  
  
**Note:** When naming user defined stored procedures, Microsoft recommends not to use **"sp\_"** as a prefix. All system stored procedures, are prefixed with **"sp\_"**. This avoids any ambiguity between user defined and system stored procedures and any conflicts, with some future system procedure.  
  
**To execute the stored procedure**, you can just type the procedure name and press F5, or use EXEC or EXECUTE keywords followed by the procedure name as shown below.  
1. uspGetEmployees  
2. EXEC uspGetEmployees  
3. Execute uspGetEmployees  
  
**Note:** You can also right click on the procedure name, in object explorer in SQL Server Management Studio and select EXECUTE STORED PROCEDURE.  
  
**Creating a stored procedure with input parameters:**

This SP, accepts GENDER and DEPARTMENTID parameters. Parameters and variables have an @ prefix in their name.

Create Procedure uspGetEmployeesByGenderAndDepartment   
@Gender nvarchar(50),  
@DepartmentId int  
as  
Begin  
  Select Name, Gender from tblEmployee Where Gender = @Gender and DepartmentId = @DepartmentId  
End  
  
To invoke this procedure, we need to pass the value for @Gender and @DepartmentId parameters. If you don't specify the name of the parameters, you have to first pass value for @Gender parameter and then for @DepartmentId.  
EXECUTE spGetEmployeesByGenderAndDepartment 'Male', 1  
  
On the other hand, if you change the order, you will get an error stating "Error converting data type varchar to int." This is because, the value of **"Male"** is passed into @DepartmentId parameter. Since @DepartmentId is an integer, we get the type conversion error.  
**spGetEmployeesByGenderAndDepartment 1, 'Male'**  
  
When you specify the names of the parameters when executing the stored procedure the order doesn't matter.  
EXECUTE spGetEmployeesByGenderAndDepartment @DepartmentId=1, @Gender = 'Male'  
  
**To view the text, of the stored procedure**  
1. Use system stored procedure sp\_helptext 'SPName'  
OR  
2. Right Click the SP in Object explorer -> Scrip Procedure as -> Create To -> New Query Editor Window  
  
**To change the stored procedure, use ALTER PROCEDURE statement:**

Alter Procedure spGetEmployeesByGenderAndDepartment   
@Gender nvarchar(50),  
@DepartmentId int  
as  
Begin  
  Select Name, Gender from tblEmployee Where Gender = @Gender and DepartmentId = @DepartmentId order by Name  
End  
  
**To encrypt the text of the SP**, use WITH ENCRYPTION option. Once, encrypted, you cannot view the text of the procedure, using sp\_helptext system stored procedure. There are ways to obtain the original text, which we will talk about in a later session.  
Alter Procedure spGetEmployeesByGenderAndDepartment   
@Gender nvarchar(50),  
@DepartmentId int  
WITH ENCRYPTION  
as  
Begin  
  Select Name, Gender from tblEmployee Where Gender = @Gender and DepartmentId = @DepartmentId  
End  
  
To delete the SP, use DROP PROC 'SPName' or DROP PROCEDURE 'SPName'

### Stored procedures with output parameters

  
  
**To create an SP with output parameter**, we use the keywords OUT or OUTPUT. @EmployeeCount is an OUTPUT parameter. Notice, it is specified with OUTPUT keyword.   
Create Procedure spGetEmployeeCountByGender  
@Gender nvarchar(20),  
@EmployeeCount int Output  
as  
Begin  
 Select @EmployeeCount = COUNT(Id)   
 from tblEmployee   
 where Gender = @Gender  
End

**To execute this stored procedure with OUTPUT parameter**  
  
**1.** First initialise a variable of the **same datatype** as that of the **output parameter**. We have declared @EmployeeTotal integer variable.

**2.** Then pass the @EmployeeTotal variable to the SP. You have to specify the **OUTPUT** keyword. If you don't specify the OUTPUT keyword, the variable will be **NULL**.   
**3.** Execute  
  
Declare @EmployeeTotal int  
Execute spGetEmployeeCountByGender 'Female', @EmployeeTotal output  
Print @EmployeeTotal  
  
If you don't specify the OUTPUT keyword, when executing the stored procedure, the @EmployeeTotal variable will be NULL. Here, we have not specified OUTPUT keyword. When you execute, you will see **'@EmployeeTotal is null'** printed.  
  
Declare @EmployeeTotal int  
Execute spGetEmployeeCountByGender 'Female', @EmployeeTotal  
if(@EmployeeTotal is null)  
 Print '@EmployeeTotal is null'  
else  
 Print '@EmployeeTotal is not null'  
  
**You can pass parameters in any order, when you use the parameter names.** Here, we are first passing the OUTPUT parameter and then the input @Gender parameter.  
  
Declare @EmployeeTotal int  
Execute spGetEmployeeCountByGender @EmployeeCount = @EmployeeTotal OUT, @Gender = 'Male'  
Print @EmployeeTotal  
  
**The following system stored procedures, are extremely useful when working procedures.**  
**sp\_help** SP\_Name : View the information about the stored procedure, like parameter names, their datatypes etc. sp\_help can be used with any database object, like tables, views, SP's, triggers etc. Alternatively, you can also press ALT+F1, when the name of the object is highlighted.

**sp\_helptext** SP\_Name : View the Text of the stored procedure  
  
**sp\_depends** SP\_Name : View the dependencies of the stored procedure. This system SP is very useful, especially if you want to check, if there are any stored procedures that are referencing a table that you are abput to drop. sp\_depends can also be used with other database objects like table etc.  
  
**Note:** All parameter and variable names in SQL server, need to have the @symbol.