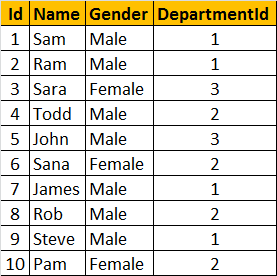
A stored procedure is group of T-SQL (Transact SQL) statements. If you have a situation, where you write the same query over and over again, you can save that specific query as a stored procedure and call it just by it's name.  
  
There are several advantages of using stored procedures, which we will discuss in a later video session. In this session, we will learn how to create, execute, change and delete stored procedures.  
  
   
  
   
  
   
  
  
  
  
**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 spGetEmployees  
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. spGetEmployees  
2. EXEC spGetEmployees  
3. Execute spGetEmployees  
  
**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 spGetEmployeesByGenderAndDepartment   
@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'  
  
**In the next seesion, we will learn creating stored procedures with OUTPUT parameters.**