



Instructor:

- Mr. Nazeef Ul Haq (Lab)

Learning Objectives:

- Understanding of Stored Procedure and Views.

Helping Material:

A SQL stored procedure (SP) is a collection SQL statements and sql command logic, which is compiled and stored on the database. Stored procedures in SQL allows us to create SQL queries to be stored and executed on the server. Stored procedures can also be cached and reused. The main purpose of stored procedures to hide direct SQL queries from the code and improve performance of database operations such as select, update, and delete data.

- **Types of stored procedures**

There are two types of stored procedures available in SQL Server:

- I. User defined stored procedures
- II. System stored procedures

- **User defined stored procedures**

User defined stored procedures are created by database developers or database administrators. These SPs contains one more more SQL statements to select, update, or delete records from database tables. User defined stored procedure can take input parameters and return output parameters. User defined stored procedure is mixture of DDL (Data Definition Language) and DML (Data Manipulation Language) commands.

User defined SPs are further classified into two types:

T-SQL stored procedures: T-SQL (Transact SQL) SPs receive and returns parameters. These SPs process the Insert, Update and Delete queries with or without parameters and return data of rows as output. This is one of the most common ways to write SPs in SQL Server.

CLR stored procedures: CLR (Common Language Runtime) SPs are written in a CLR based programming language such as C# or VB.NET and are executed by the .NET Framework.

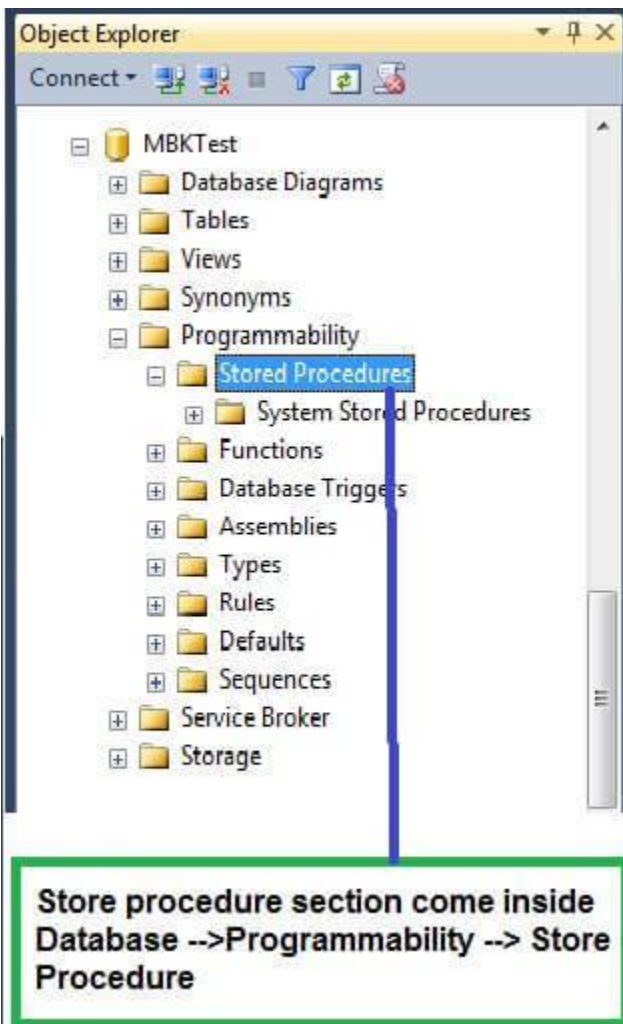
- **System stored procedures**

System stored procedures are created and executed by SQL Server for the server administrative activities. Developers usually don't interfere with system SPs.

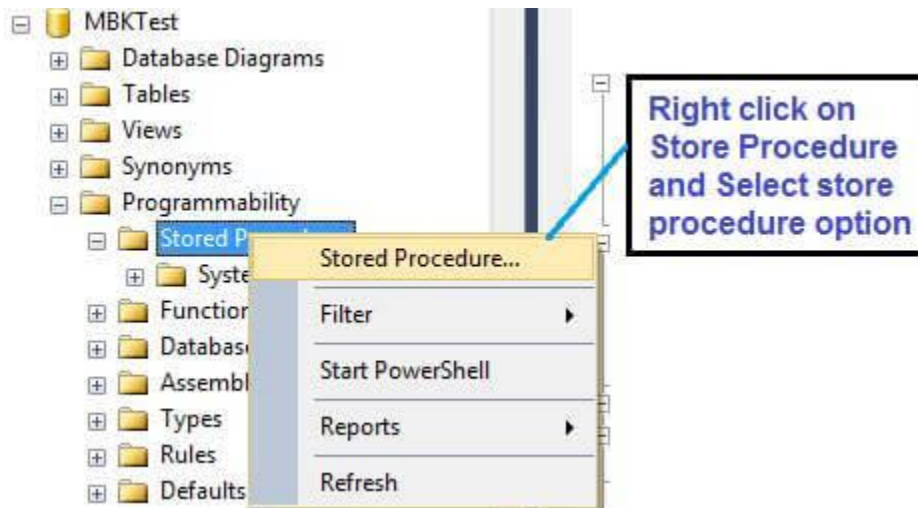
Let's login to our SQL Server database, so we can achieve the following:

- How to create a SELECT QUERY based stored procedure which return all records?
- How to create a PARAMETER based SELECT QUERY stored procedure which return records based on parameters?
- How to create an INSERT query based stored procedure?
- How to create an UPDATE query based stored procedure?
- How to create a DELETE query based stored procedure?

Switch to your database. My database name is MBKTest.



Empty stored procedure will be created using the following:



The empty template created by SQL Server for a SP looks like the following. The **CREATE PROCEDURE** SQL command is used to create a procedure, followed by a SP name and its parameters. The **BEGIN** and **END** area is used to define the query for the operation. This is where you will write a select, update, insert, or delete queries.

```
1. -- =====
2. -- Template generated from Template Explorer using:
3. -- Create Procedure (New Menu).SQL
4. --
5. -- Use the Specify Values for Template Parameters
6. -- command (Ctrl-Shift-M) to fill in the parameter
7. -- values below.
8. --
9. -- This block of comments will not be included in
10. -- the definition of the procedure.
11. -- =====
12. SET ANSI_NULLS ON
13. GO
14. SET QUOTED_IDENTIFIER ON
15. GO
16. -- =====
17. -- Author:    <Author,,Name>
18. -- Create date: <Create Date,,>
19. -- Description: <Description,,>
20. -- =====
21. CREATE PROCEDURE <Procedure_Name, sysname, ProcedureName>
22. -- Add the parameters for the stored procedure here
23.    <@Param1, sysname, @p1> <Datatype_For_Param1, , int> = <Default_Value_For_Param1, , 0>,
24.    <@Param2, sysname, @p2> <Datatype_For_Param2, , int> = <Default_Value_For_Param2, , 0>
```

```

25. AS
26. BEGIN
27.  -- SET NOCOUNT ON added to prevent extra result sets from
28.  -- interfering with SELECT statements.
29.  SET NOCOUNT ON;
30.
31.  -- Insert statements for procedure here
32.  SELECT <@Param1, sysname, @p1>, <@Param2, sysname, @p2>
33. END
34. GO

```

What is the naming convention for stored procedures?

We must follow standard naming conventions which may also depend on your project and coding policies.

For user defined stored procedure naming conventions, my suggestions are to add one of the following prefixes to your SP names.

1. sp
2. stp
3. stp_
4. udstp
5. udstp_

Naming conventions are just to identify objects. By adding these prefixes in the name, we can clearly identify that this object is a stored procedure.

Create a database table

Before, we can create and execute any SPs, we need a database table. I create a database table named, “tblMembers” using the following SQL query and execute it on the server. As you can see, my table has 4 column where the first column is an identity column. Once the table is created, open table in your SSMS and add some data by manually entering data to the table.

```

1. USE [MBKTest]
2. GO
3.
4. /***** Object: Table [dbo].[tblMembers] Script Date: 18-Nov-17,Sat 6:47:55 PM *****/
5. SET ANSI_NULLS ON
6. GO
7.
8. SET QUOTED_IDENTIFIER ON
9. GO
10.
11. SET ANSI_PADDING ON
12. GO
13.

```

```

14. CREATE TABLE [dbo].[tblMembers](
15.     [MemberID] [int] IDENTITY(1,1) NOT NULL,
16.     [MemberName] [varchar](50) NULL,
17.     [MemberCity] [varchar](25) NULL,
18.     [MemberPhone] [varchar](15) NULL
19. )
20.
21. GO
22.
23. SET ANSI_PADDING OFF
24. GO

```

How to create a SELECT stored procedure?

Click on your Database and expand “Programmability” item and right click on “Stored Procedures” or press CTRL + N to get new query window. In the query area between BEGIN and END, type your SELECT statement to select records from the table. See the Select statement in the below code.

```

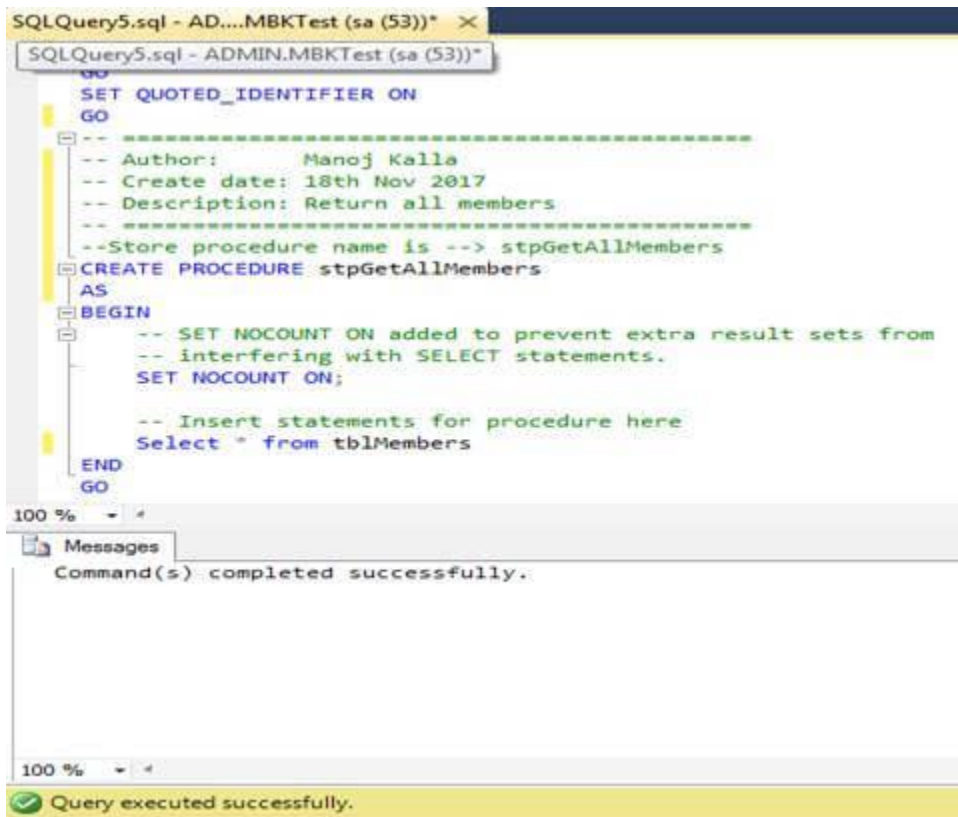
1. SET ANSI_NULLS ON
2. GO
3. SET QUOTED_IDENTIFIER ON
4. GO
5. -- =====
6. -- Author:    Manoj Kalla
7. -- Create date: 18th Nov 2017
8. -- Description: Return all members
9. -- =====
10. --Store procedure name is --> stpGetAllMembers
11. CREATE PROCEDURE stpGetAllMembers
12. AS
13. BEGIN
14.     -- SET NOCOUNT ON added to prevent extra result sets from
15.     -- interfering with SELECT statements.
16.     SET NOCOUNT ON;
17.
18.     -- Select statements for procedure here
19.     Select * from tblMembers
20. END
21. GO

```

Now, press F5 or click on Execute button to execute the SP.



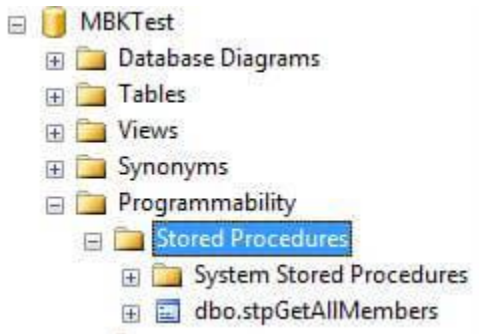
Press F5 or click on Execute button to save the store procedure.



You should see a message, “Command(s) completed successfully.”

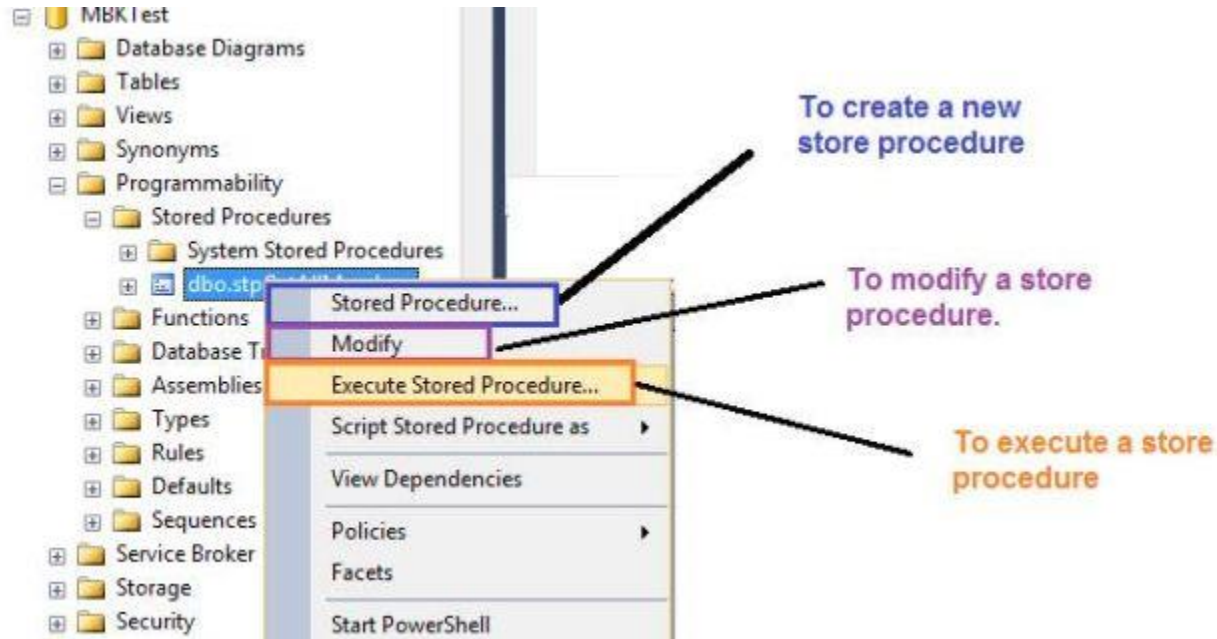
Now go to Programmability -->Stored Procedures, Right Click and select Refresh.

You can see in the following image, the new SP called stpGetAllMembers is created.



Execute stored procedures in SQL Server

In below UI, right click on the SP name and select Execute Stored Procedure... to execute a SP. From here, you can also modify an existing SP.



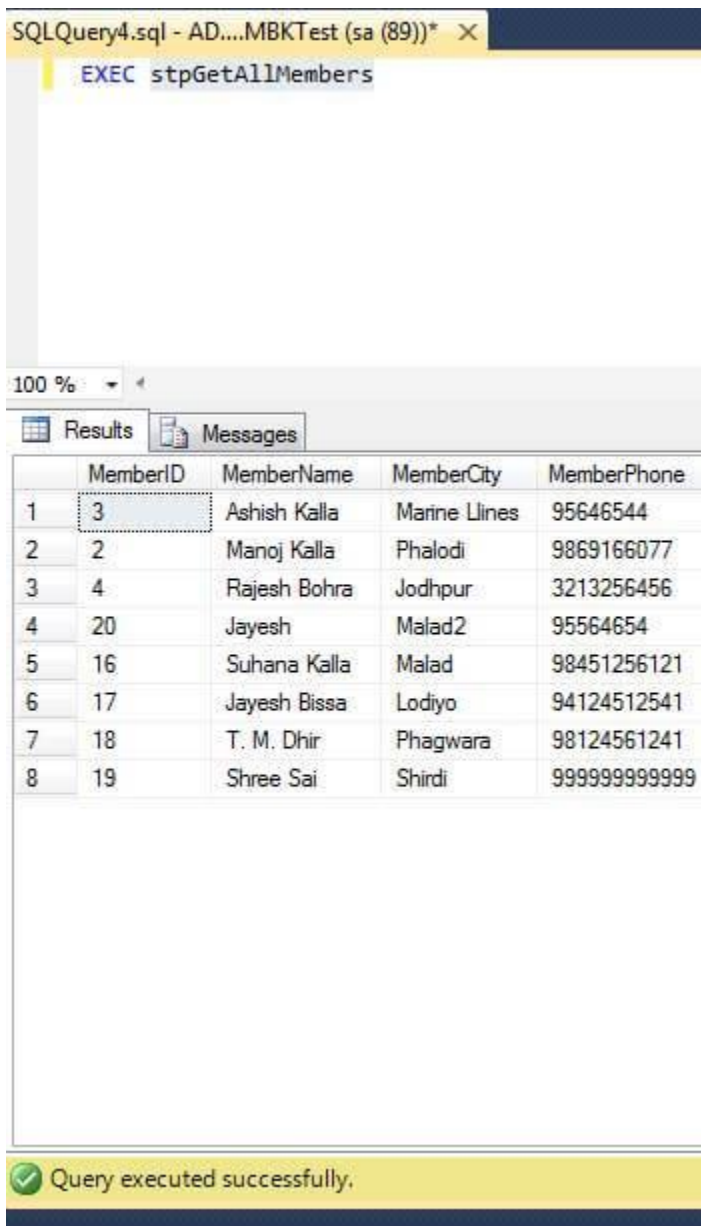
Alternatively, you can also execute a SP from the Query window.

To run stored procedure in SQL Server Management Studio, switch to Query window or CTRL +N to open an new query window and type the following command.

- Syntax - EXEC <stored procedure name>
- Example - EXEC stpGetAllMembers

Now, we run our stored procedure called stpGetAllMembers. The output looks like the following:

OUTPUT



The screenshot shows a SQL Server Enterprise Manager window titled 'SQLQuery4.sql - AD...MBKTest (sa (89))'. The 'EXEC stpGetAllMembers' command is entered in the query editor. Below the editor, the 'Results' tab is active, displaying a table with 8 rows and 5 columns: MemberID, MemberName, MemberCity, and MemberPhone. The first row is highlighted. At the bottom, a yellow status bar indicates 'Query executed successfully.'

	MemberID	MemberName	MemberCity	MemberPhone
1	3	Ashish Kalla	Marine Llines	95646544
2	2	Manoj Kalla	Phalodi	9869166077
3	4	Rajesh Bohra	Jodhpur	3213256456
4	20	Jayesh	Malad2	95564654
5	16	Suhana Kalla	Malad	98451256121
6	17	Jayesh Bissa	Lodiyo	94124512541
7	18	T. M. Dhir	Phagwara	98124561241
8	19	Shree Sai	Shirdi	999999999999

What are parameters in stored procedures?

Parameters in SPs are used to pass input values and return output values. There are two types of parameters:

1. Input parameters - Pass values to a stored procedure.
2. Output parameters - Return values from a stored procedure.

How to create a SELECT query SP with parameters?

In the previous steps, we created a simple SP that returned all rows from a table. Now, let's create a new SP that will take a city name as an input parameter and will return all rows where city name matches the input parameter value.

Here is the updated SP with a parameter @CityName.

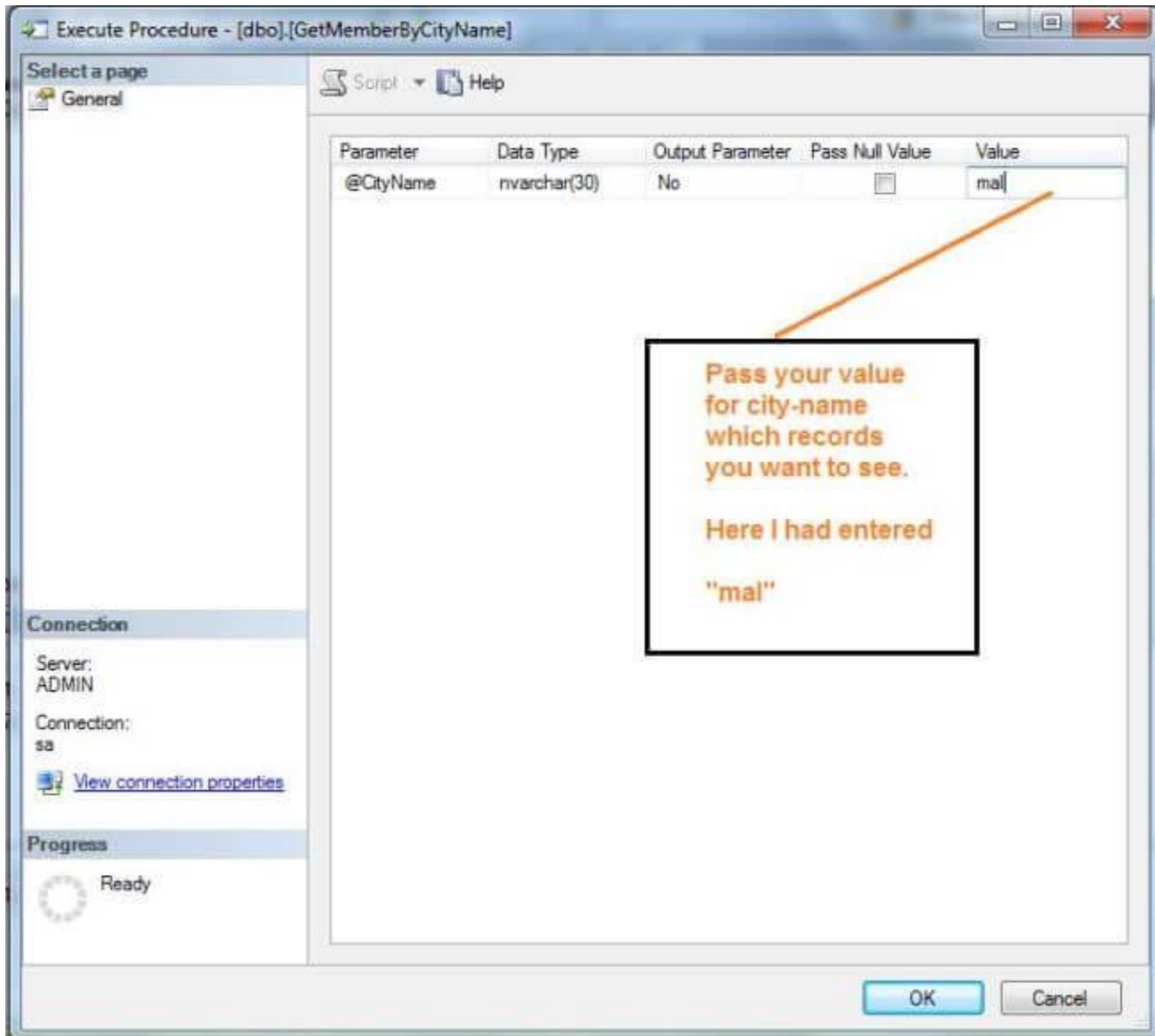
```
1. SET ANSI_NULLS ON
2. GO
3. SET QUOTED_IDENTIFIER ON
4. GO
5. -- =====
6. -- Author:    Manoj Kalla
7. -- Create date: 20-Nov-2017
8. -- Description: Return specific city records
9. -- =====
10. CREATE PROCEDURE stpGetMembersByCityName
11. -- Add the parameters for the stored procedure here
12. @CityName nvarchar(30)
13.
14. AS
15. BEGIN
16. -- SET NOCOUNT ON added to prevent extra result sets from
17. -- interfering with SELECT statements.
18. SET NOCOUNT ON;
19.
20. Select * From tblMembers
21. where MemberCity like '%' + @CityName + '%'
22.
23. END
24. GO
```

Execute it.

To run this SP, type the following command in SQL query tool:

```
EXEC GetMemberByCityName @CityName = 'mal'
```

OR from the UI, run the SP and provide the following input.



The code to execute looks like the following:

```
1. USE [MBKTest]
2. GO
3.
4. DECLARE @return_value int
5.
6. EXEC @return_value = [dbo].[GetMemberByCityName]
7.     @CityName = N'mal'
8.
9. SELECT 'Return Value' = @return_value
10.
11. GO
```

OUTPUT

SQLQuery7.sql - AD....MBKTest (sa (53))* X SQLQuery5.sql - AD....MBKT

EXEC GetMemberByCityName @CityName = 'mal'

100 %

Results Messages

	MemberID	MemberName	MemberCity	MemberPhone
1	20	Jayesh	Malad2	95564654
2	16	Suhana Kalla	Malad	98451256121

How to create a INSERT query based stored procedure?

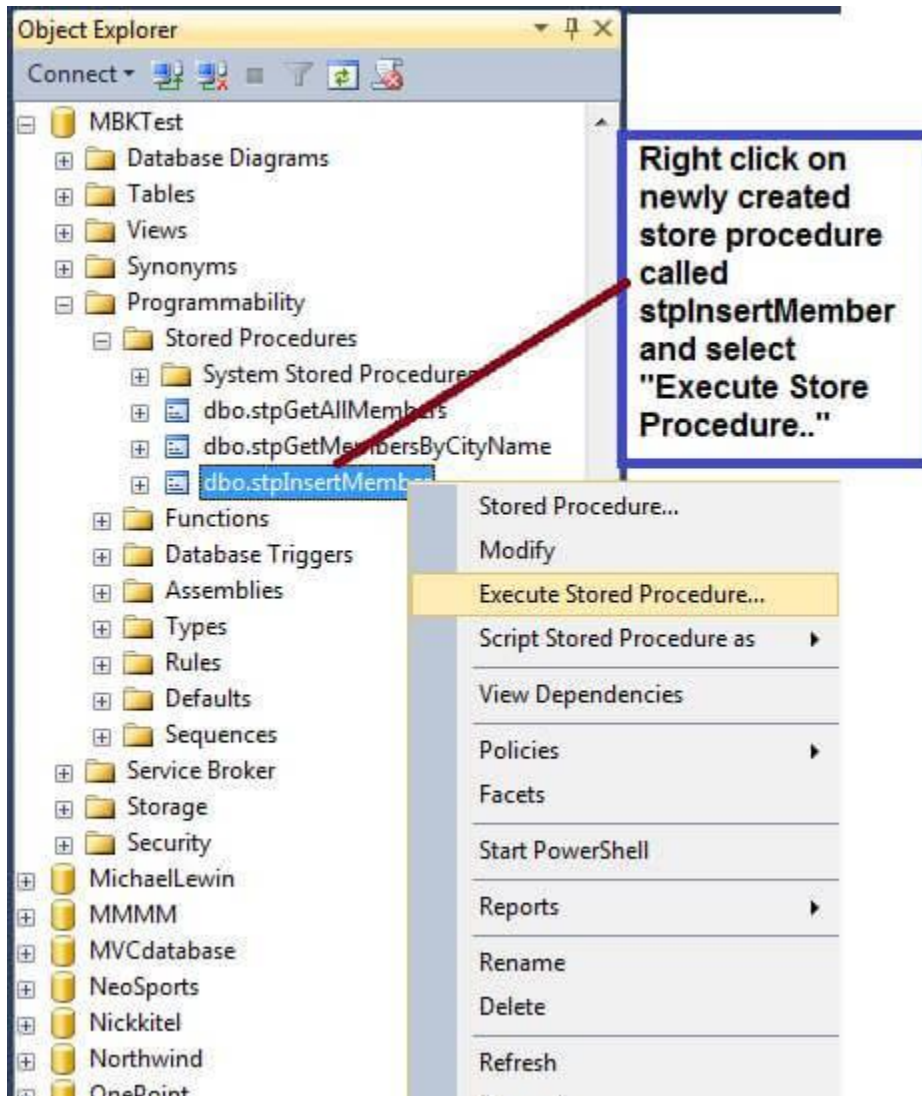
We can use an INSERT INTO SQL query to insert data into a table. The following SQL statement creates an INSERT SP with three parameters.

```

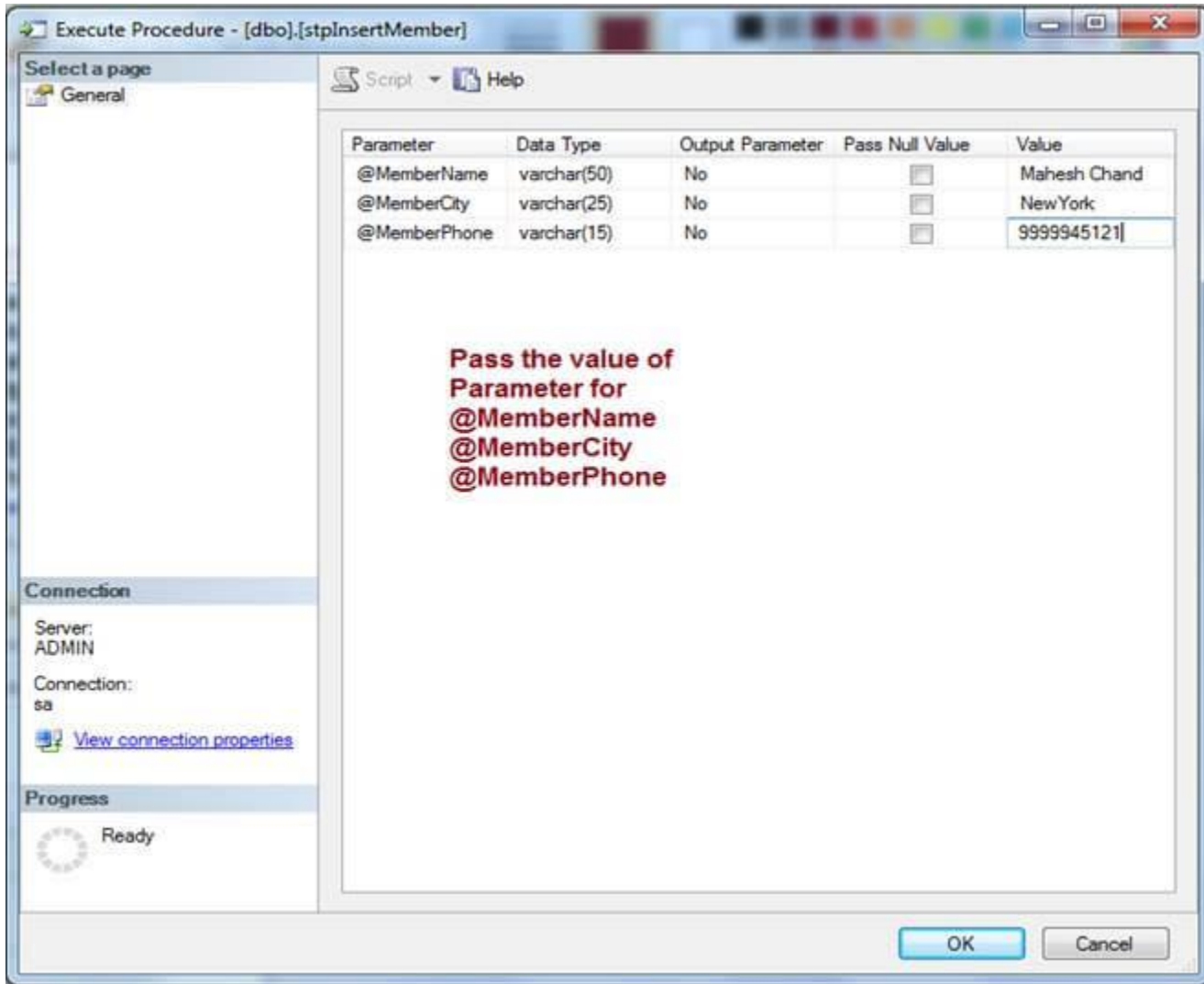
1. SET ANSI_NULLS ON
2. GO
3. SET QUOTED_IDENTIFIER ON
4. GO
5. -- =====
6. -- Author:    Manoj Kalla
7. -- Create date: 20-Nov-2047
8. -- Description: To create a new member
9. -- =====
10. CREATE PROCEDURE stpInsertMember
11. @MemberName varchar(50),
12. @MemberCity varchar(25),
13. @MemberPhone varchar(15)
14.
15. AS
16. BEGIN
17. -- SET NOCOUNT ON added to prevent extra result sets from
18. -- interfering with SELECT statements.
19. SET NOCOUNT ON;
20.
21. Insert into tblMembers (MemberName,MemberCity,MemberPhone)
22. Values (@MemberName,@MemberCity, @MemberPhone)
23.
24. END
25. GO

```

Right click on stored procedure in Object Explorer and select Refresh.



Pass the value of parameter in Execute dialog box. Something like this:

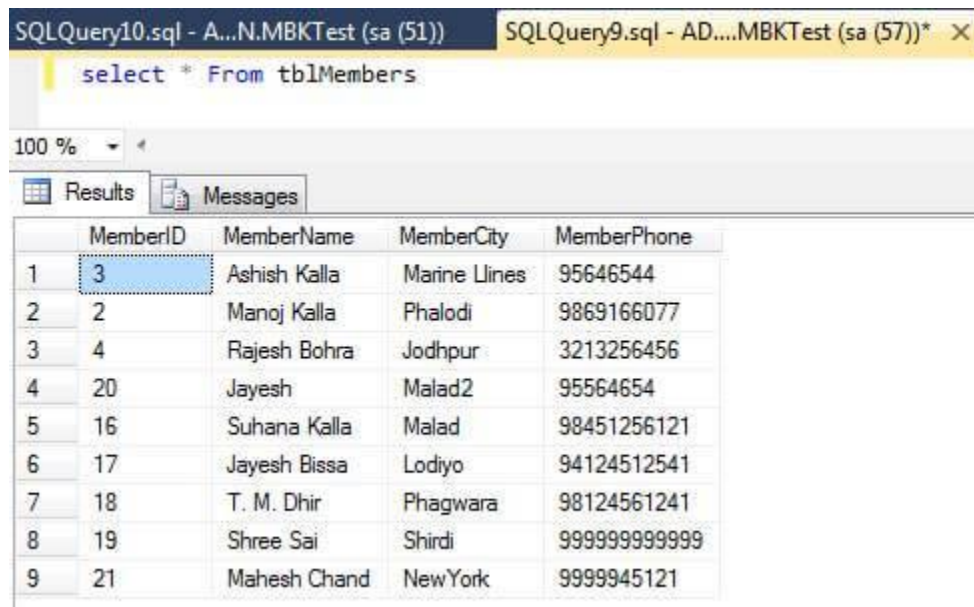


The following code can be used to execute this SP in SSMS.

```
1. USE [MBKTest]
2. GO
3.
4. DECLARE @return_value int
5.
6. EXEC @return_value = [dbo].[stpInsertMember]
7.     @MemberName = N'Mahesh Chand',
8.     @MemberCity = N'NewYork',
9.     @MemberPhone = N'9999945121'
10. SELECT 'Return Value' = @return_value
11. GO
```

OUTPUT

In the query window, you can check if a new record for Member Name 'Mahesh Chand' is added to the table.

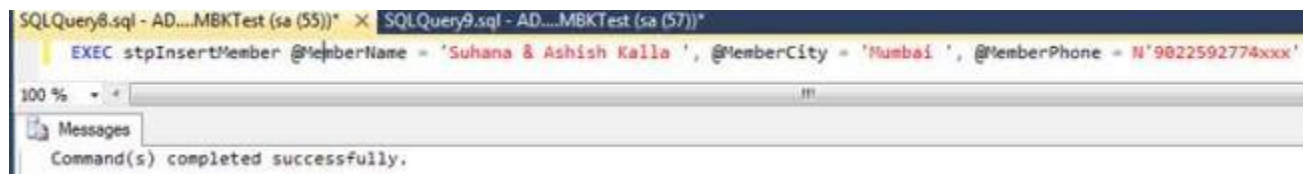


The screenshot shows a SQL query window with the query `select * From tblMembers`. The results are displayed in a table with the following columns: MemberID, MemberName, MemberCity, and MemberPhone. The table contains 9 records, with the 9th record being 'Mahesh Chand' from 'NewYork' with phone number '9999945121'.

	MemberID	MemberName	MemberCity	MemberPhone
1	3	Ashish Kalla	Marine Lines	95646544
2	2	Manoj Kalla	Phalodi	9869166077
3	4	Rajesh Bohra	Jodhpur	3213256456
4	20	Jayesh	Malad2	95564654
5	16	Suhana Kalla	Malad	98451256121
6	17	Jayesh Bissa	Lodiyo	94124512541
7	18	T. M. Dhir	Phagwara	98124561241
8	19	Shree Sai	Shirdi	99999999999
9	21	Mahesh Chand	NewYork	9999945121

You can also run the same SP in code.

*EXEC stpInsertMember @MemberName = 'Suhana & Ashish Kalla ', @MemberCity = 'Mumbai ',
@MemberPhone = N'9022592774xxx'*



OUTPUT

You can check “Suhana & Ashish Kalla” record is added successfully.

SQLQuery8.sql - AD....MBKTest (sa (55))* SQLQuery9.sql - AD....MBKTest (sa (57))*

```
select * From tblMembers
```

100 %

Results Messages

	MemberID	MemberName	MemberCity	MemberPhone
1	3	Ashish Kalla	Marine Lines	95646544
2	2	Manoj Kalla	Phalodi	9869166077
3	4	Rajesh Bohra	Jodhpur	3213256456
4	20	Jayesh	Malad2	95564654
5	16	Suhana Kalla	Malad	98451256121
6	17	Jayesh Bissa	Lodiyo	94124512541
7	18	T. M. Dhir	Phagwara	98124561241
8	19	Shree Sai	Shirdi	999999999999
9	21	Mahesh Chand	NewYork	9999945121
10	22	Suhana & Ashish Kalla	Mumbai	9022592774xxx

How to create an UPDATE quert based stored procedure?

Let's create a new SP that will update a table records based on the Member ID column. The ID is passed as an input parameter. Here is the new SP that uses an UPDATE..SET..WHERE command.

```

1. SET ANSI_NULLS ON
2. GO
3. SET QUOTED_IDENTIFIER ON
4. GO
5. -- =====
6. -- Author:   Manoj Kalla
7. -- Create date: 20-Nov-2017
8. -- Description: Update a member detail by ID
9. -- =====
10. CREATE PROCEDURE stpUpdateMemberByID
11. @MemberID int,
12. @MemberName varchar(50),
13. @MemberCity varchar(25),
14. @MemberPhone varchar(15)
15.
16. AS
17. BEGIN
18. -- SET NOCOUNT ON added to prevent extra result sets from
19. -- interfering with SELECT statements.
20. SET NOCOUNT ON;
21.
22. UPDATE tblMembers
23. Set MemberName = @MemberName,

```



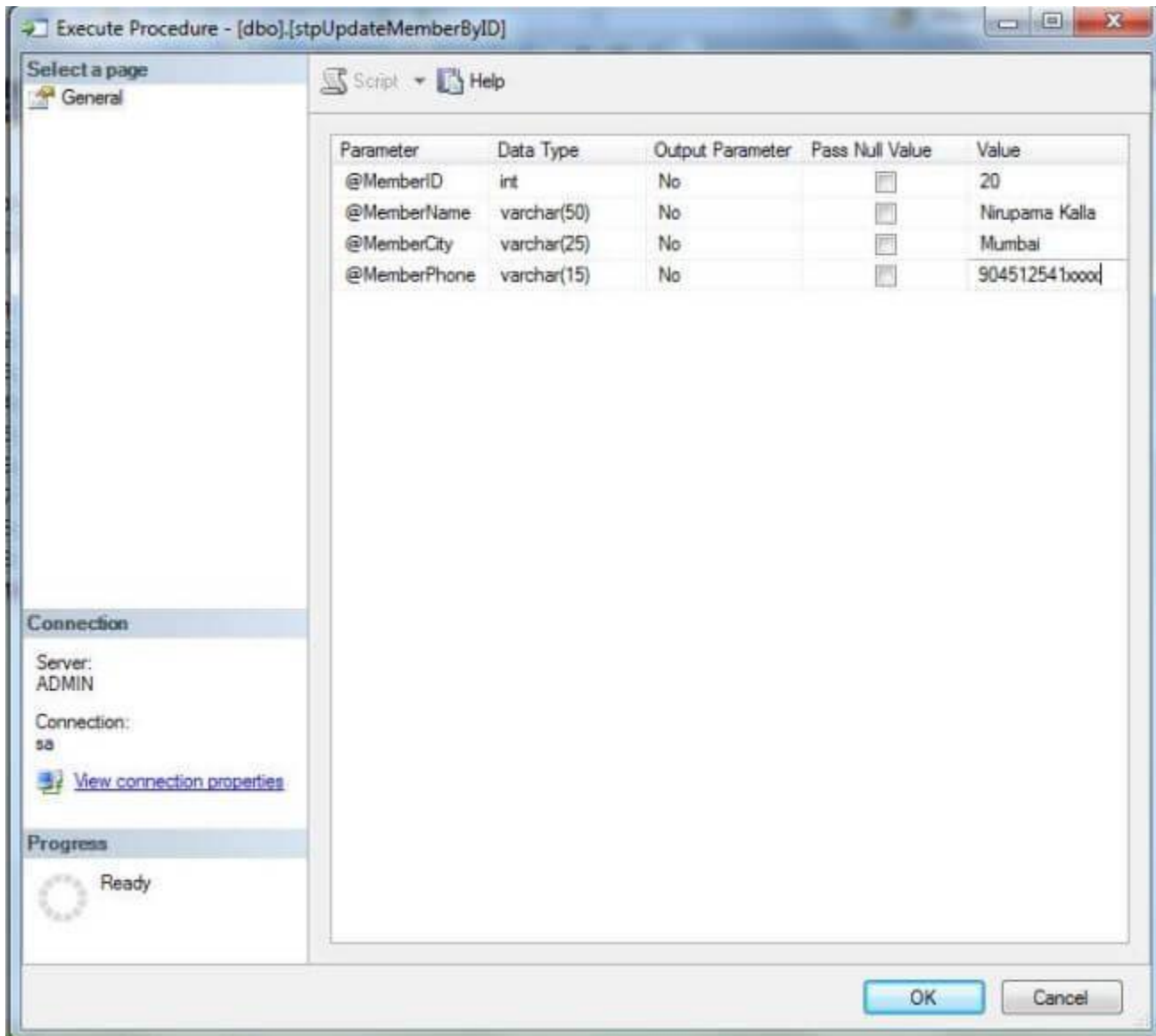
```

24. MemberCity = @MemberCity,
25. MemberPhone = @MemberPhone
26. Where MemberID = @MemberID
27. END
28. GO

```

Right click on stored procedure in the Object Explorer and select Refresh. You will see the SP is created.

Now, Right click on SP name and select Execute stored procedure.... Provide the input values and execute.



We can use the following command in SSMS.

```

1. USE [MBKTest]

```

```

2. GO
3.
4. DECLARE @return_value int
5.
6. EXEC @return_value = [dbo].[stpUpdateMemberByID]
7.     @MemberID = 20,
8.     @MemberName = N'Nirupama Kalla',
9.     @MemberCity = N'Mumbai',
10.    @MemberPhone = N'904512541xxxx'
11.
12. SELECT 'Return Value' = @return_value
13.
14. GO

```

EXEC stpUpdateMemberByID 17,'Gopal Madhavrai','Bikaner','90454564xxx'

The results should show you the updated values.

	MemberID	MemberName	MemberCity	MemberPhone
1	3	Ashish Kalla	Marine Lines	95646544
2	2	Manoj Kalla	Phalodi	9869166077
3	4	Rajesh Bohra	Jodhpur	3213256456
4	20	Nirupama Kalla	Mumbai	904512541xxxx
5	16	Suhana Kalla	Malad	98451256121
6	17	Gopal Madhavrai	Bikaner	90454564xxx
7	18	T. M. Dhir	Phagwara	98124561241
8	19	Shree Sai	Shirdi	99999999999
9	21	Mahesh Chand	NewYork	9999945121
10	22	Suhana & Ashish Kalla	Mumbai	9022592774xxx

**Nirupama Kalla 's
detail updated
successfully.**

**Gopal Madhavrai's detail
updated successfully**

How to create a DELETE query based stored procedure?

Let's create a SP that will delete records. The new SP uses a DELETE command and delete all records that matches provided Member ID.

```

1. SET ANSI_NULLS ON
2. GO
3. SET QUOTED_IDENTIFIER ON
4. GO

```

```

5. -- =====
6. -- Author:    Manoj Kalla
7. -- Create date: 21-Nov-2017
8. -- Description: Delete a Member by Member ID
9. -- =====
10. CREATE PROCEDURE stpDeleteMemberByMemberID
11.    @MemberID int
12. AS
13. BEGIN
14.    -- SET NOCOUNT ON added to prevent extra result sets from
15.    -- interfering with SELECT statements.
16.    SET NOCOUNT ON;
17.
18.    Delete from tblMembers
19.    where MemberId = @MemberID
20.
21. END
22. GO

```

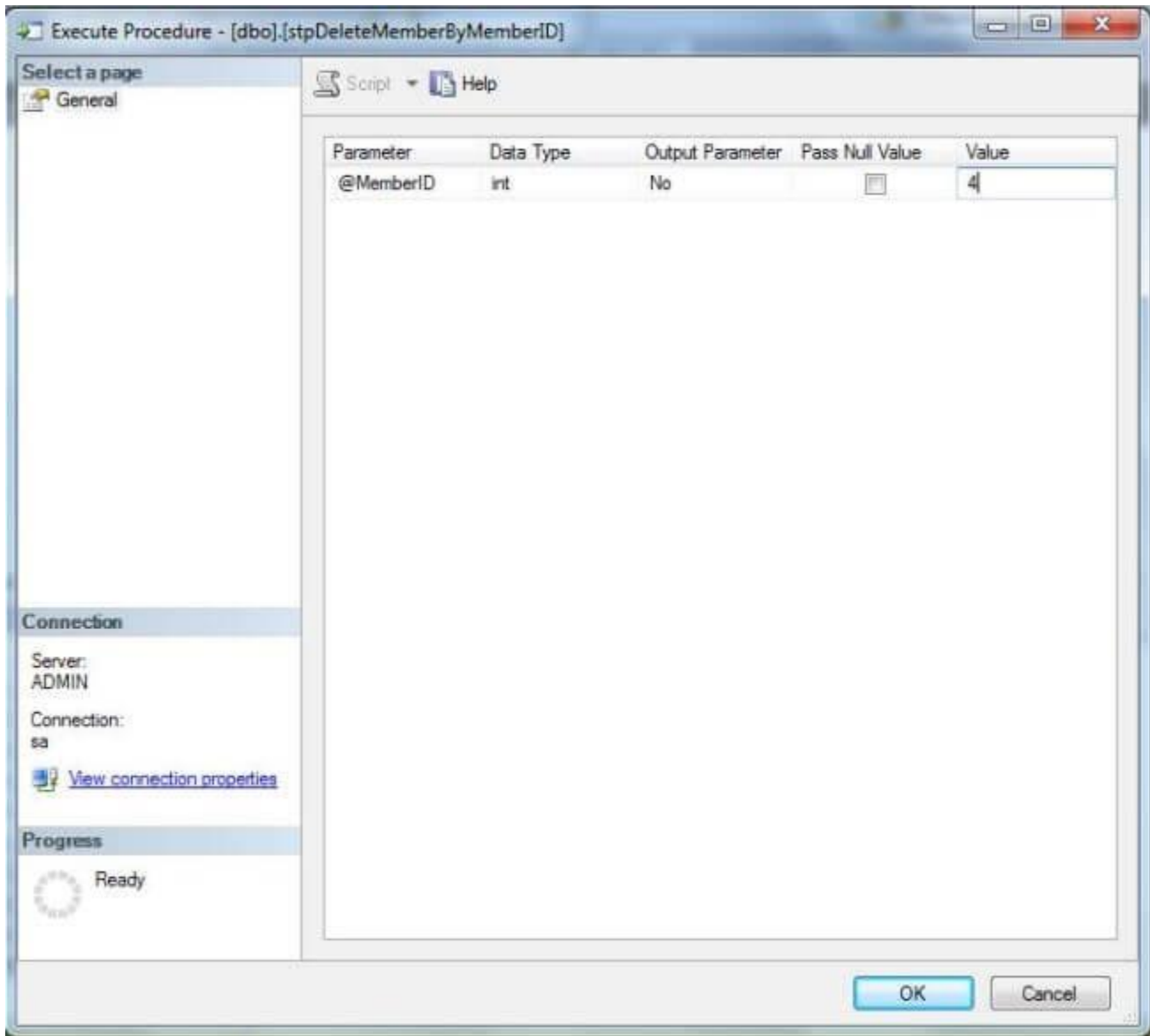
Execute it.

Right click on Stored Procedures in the Object Explorer and select Refresh.

RUN stored procedure BY UI

Now again right click on stored procedure and select Execute stored procedure...

As you can see in the image, I passed @MemberID parameter value = 4.



RUN DELETE stored procedure BY MANUALLY (CODING)

EXEC stpDeleteMemberByMemberID 2

OUTPUT

You can see in image MemberID = 4 record has been deleted successfully.

	MemberID	MemberName	MemberCity	MemberPhone
1	3	Ashish Kalla	Marine Lines	95646544
2	2	Manoj Kalla	Phalodi	9869166077
3	20	Nirupama Kalla	Mumbai	904512541xxxx
4	16	Suhana Kalla	Malad	98451256121
5	17	Gopal Madhavrai	Bikaner	90454564xxx
6	18	T. M. Dhir	Phagwara	98124561241
7	19	Shree Sai	Shirdi	999999999999
8	21	Mahesh Chand	New York	9999945121
9	22	Suhana & Ashish Kalla	Mumbai	9022592774xxx

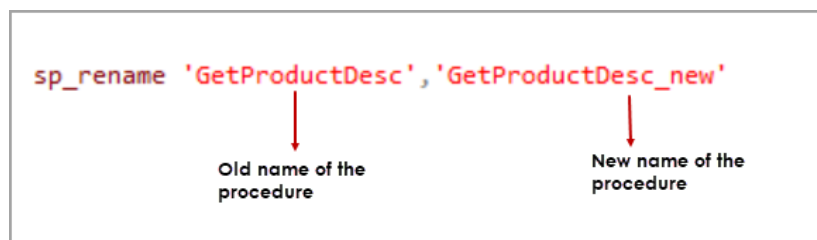
In this article, we saw how to create stored procedures in a SQL Server database for inserting, updating, and deleting records.

Modifying the stored procedure

```
ALTER PROCEDURE GetProductDesc
AS
SELECT P.ProductID,P.ProductName,PD.ProductDescription FROM
Product P
INNER JOIN ProductDescription PD ON P.ProductID=PD.ProductID
END
```

Renaming the stored procedure

```
sp_rename 'GetProductDesc','GetProductDesc_new'
```



SQL Dropping a stored procedure

```
DROP PROCEDURE procedure_name;
```

What to Submit:

You are required to submit the following files.

- Run all the things in any table of Northwind database and create a pdf file similar like this.