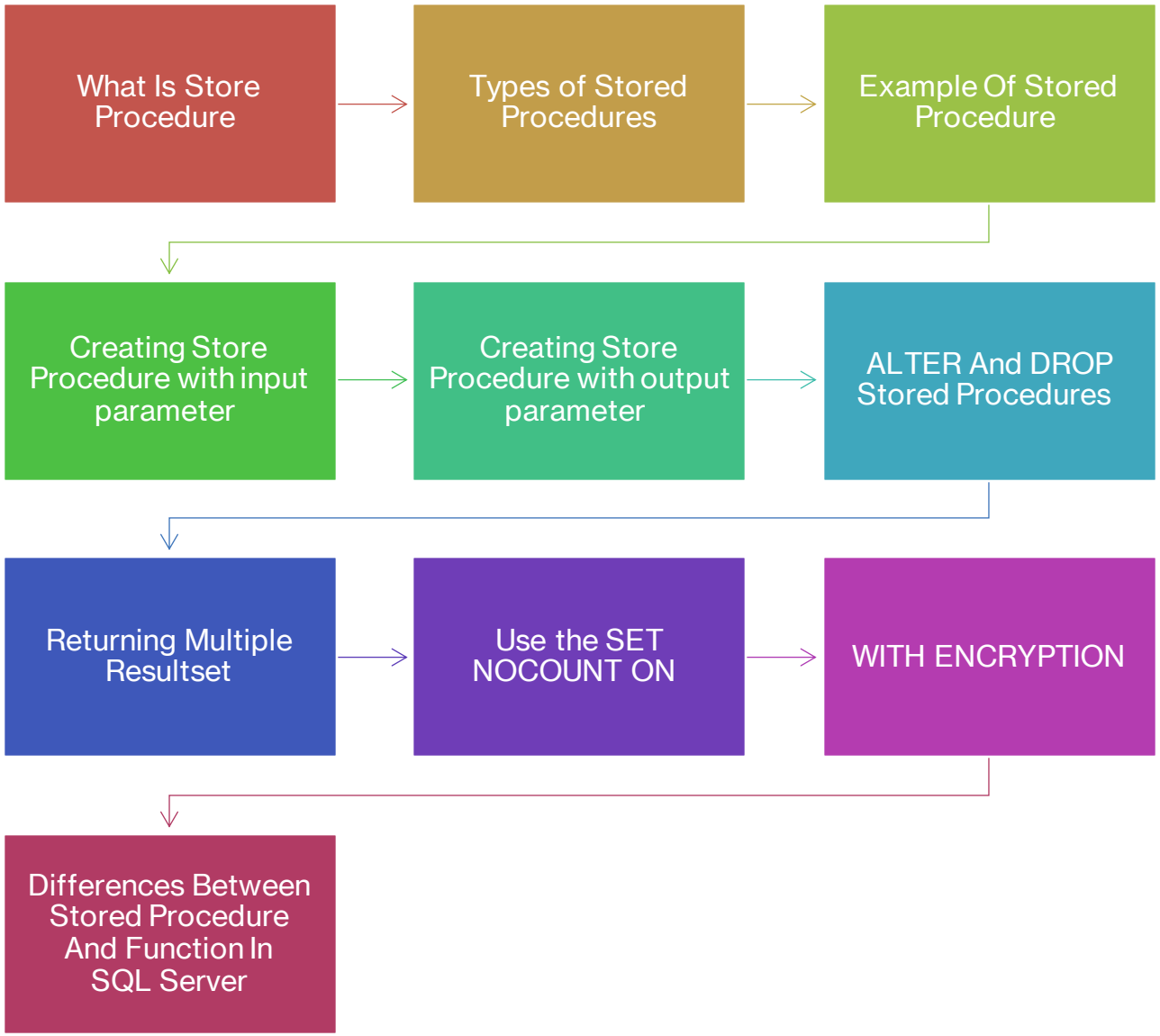
STORE PROCEDURE

By Deep Parmar

What to Learn



What Is Store 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 values that is passed.

Types of Stored Procedures

1. User-defined Stored Procedures

- Database Developers Or Database Administrators Build User-defined Stored Procedures.These Procedures Provide One Or More SQL Statements For Selecting, Updating, Or Removing Data From Database Tables. A Stored Procedure Specified By The User Accepts Input Parameters And Returns Output Parameters. DDL And DML Commands Are Used Together In A User-defined Procedure.

2. System Stored Procedures

- The system stored procedures prevent the administrator from querying or modifying the system and database catalog tables directly.

Example Of Stored Procedure

- CREATE PROCEDURE [schema_name].procedure_name
- @parameter_name data_type,
- ....
- parameter_name data_type
- AS
- BEGIN
- -- SQL statements
- -- SELECT, INSERT, UPDATE, or DELETE statement
- END

```
------------WITHOUT parameter--------------------
CREATE PROCEDURE spEmployeeDepartmentGet
AS
BEGIN
SELECT * FROM Employees AS EMP
JOIN departments AS DEPT ON DEPT.DEPARTMENT_ID=EMP.DepartmentID
END

----------Executing Store Procedure----------------------
spEmployeeDepartmentGet
EXEC spEmployeeDepartmentGet
EXECUTE spEmployeeDepartmentGet
```

Results | Messages | Client Statistics

	EmployeeID	FirstName	LastName	Email	PhoneNumber	HireDate	JobId	Salary	CommissionPct	ManagerID	DepartmentID	DEPARTMENT_ID	DEPARTMENT_NAME	MANAGER_ID	LOCATION_ID
1	100	Steven	King	SKING	515.123.4567	1987-06-17	AD_PRES	24000.00	0.00	0	90	90	Executive	100	1700
2	101	Neena	Kochhar	NKOCHHAR	515.123.4568	1987-06-18	AD_VP	17000.00	0.00	100	90	90	Executive	100	1700
3	102	Lex	De Haan	LDEHAAN	515.123.4569	1987-06-19	AD_VP	17000.00	0.00	100	90	90	Executive	100	1700
4	103	Alexander	Hunold	AHUNOLD	590.423.4567	1987-06-20	IT_PROG	9000.00	0.00	102	60	60	IT	103	1400

Creating Store Procedure with input parameter

- we will extend the stored procedure which allows you to pass one or more values to it. The result of the stored procedure will change based on the values of the parameters.

```
-------------Creating Store Procedure with input parameter--------------

CREATE PROCEDURE spEmployeeGetName
@Job_Id varchar(10),
@Salary DECIMAL(8,2)
AS
BEGIN

    SELECT Employee_Name=(FirstName+SPACE(1)+LastName) FROM Employees
    WHERE JobId=@Job_Id AND Salary > @Salary
END

EXECUTE spEmployeeGetName 'SH_CLERK',3000
```

117 %

Results | Messages | Client Statistics

	Employee_Name
1	Winston Taylor
2	Jean Fleaur
3	Nandita Sarchand
4	Alexis Bull
5	Julia Dellinger
6	Kelly Chung
7	Jennifer Dilly
8	Sarah Bell
9	Britney Everett
10	Samuel McCain
11	Alana Walsh

Creating Store Procedure with output parameter

- A stored procedure can have many output parameters. In addition, the output parameters can be in any valid data type e.g., int, date, and varying character .

```
----------------Creating Store Procedure with output para
CREATE PROCEDURE spEmployeeGetCount
@Department_Name VARCHAR(15),
@Count INT OUTPUT
AS
BEGIN

    SELECT @Count=COUNT(EMP.EmployeeID) FROM Employees AS
    JOIN departments AS DEPT ON EMP.DepartmentID=DEPT.DEP
    WHERE DEPT.DEPARTMENT_NAME=@Department_Name
END

DECLARE @Count_No INT
EXECUTE spEmployeeGetCount 'IT',@Count_No OUTPUT
PRINT 'Total Employee In IT Department is: '+CAST(@Count_
```

17 %

Messages | Client Statistics

```
Total Employee In IT Department is: 5

Completion time: 2021-09-06T19:45:45.3559424+05:30
```

ALTER And DROP Stored Procedures

- To modify an existing stored procedure, you use the ALTER PROCEDURE statement.

- To delete a stored procedure, you use the DROP PROCEDURE or DROP PROC statement:
- DROP PROCEDURE sp_name

```
----------------ALTER Store Pr
ALTER PROCEDURE spEmployeeDep
AS
BEGIN
SELECT Employee_Name=(FirstNa
JOIN departments AS DEPT ON D
END


----------------DROP Store Pro
DROP PROCEDURE spEmployeeDepa
```

```
----------Returning multiple resultset----------
CREATE PROCEDURE spMultipleResultset
AS
BEGIN
SELECT * FROM Employees
SELECT * FROM departments
END

EXEC spMultipleResultset
```

117 %

Results | Messages | Client Statistics

```
(107 rows affected)

(27 rows affected)

Completion time: 2021-09-06T20:47:35.6795659+05:30
```

Returning Multiple Resultset

```
---------Return JSON output from Store Procedure----------
CREATE PROCEDURE spEmployeeDepartmentjson
@Json NVARCHAR(MAX) OUTPUT
AS
BEGIN
SET @Json=(SELECT TOP (10) Employee_Name=(FirstName+SPACE(1)+LastName),DEPT.DEPARTMENT_NAME,Salary
        JOIN departments AS DEPT ON DEPT.DEPARTMENT_ID=EMP.Department_ID
        FOR JSON PATH,WITHOUT_ARRAY_WRAPPER)
END

DECLARE @json NVARCHAR(MAX)
EXEC spEmployeeDepartmentjson @json OUTPUT
PRINT(@json)
```

117 %

Messages | Client Statistics

```
{"Employee_Name":"Steven King","DEPARTMENT_NAME":"Executive","Salary":24000.00},{"Employee_Name":"Neena Kochhar

Completion time: 2021-09-06T19:43:11.5916190+05:30
```

Return JSON output from Store Procedure

Use the SET NOCOUNT ON

- SET NOCOUNT ON: This prevents the message from showing which contains the number of affected rows.

SET NOCOUNT OFF: This shows the number of affected rows in a message window.

```
----------Use the SET NOCOUNT ON----------
CREATE PROCEDURE spEmployeeDetailsUsingCity
@CITY VARCHAR(10)
AS
BEGIN
    SET NOCOUNT ON
    SELECT Employee_Name=(FirstName+SPACE(1)+La
    JOIN departments AS DEPT ON DEPT.DEPARTMENT_
    JOIN Locations AS LOC ON DEPT.LOCATION_ID=L
    WHERE LOC.City=@CITY
END

EXECUTE spEmployeeDetailsUsingCity 'Seattle'
```

17 %

Results | Messages | Client Statistics

```
Commands completed successfully.

Completion time: 2021-09-06T20:39:30.5008386+05:30
```

WITH ENCRYPTION

- That means if this With Encryption attribute is used while creating the stored procedure, then the text or content of the stored procedure is encrypted and will not be stored in the text column of the syscomments system table. As a result, we cannot view the text of the stored procedure.

```
-----------------WITH ENCRYPTION----------------------------------
CREATE PROCEDURE spEmployeeDetailsUsingCityDept
@CITY VARCHAR(10),
@DEPARTMENT_Name VARCHAR(10)
WITH ENCRYPTION
AS
BEGIN
    SET NOCOUNT ON
    SELECT Employee_Name=(FirstName+SPACE(1)+LastName),DEPT.DEPARTI
    JOIN departments AS DEPT ON DEPT.DEPARTMENT_ID=EMP.DepartmentI
    JOIN Locations AS LOC ON DEPT.LOCATION_ID=LOC.LocationID
    WHERE LOC.City=@CITY
END

EXECUTE spEmployeeDetailsUsingCityDept 'Seattle','Finance'

sp_helptext spEmployeeDetailsUsingCityDept
```

117 %

Messages

```
The text for object 'spEmployeeDetailsUsingCityDept' is encrypted.

Completion time: 2021-09-06T19:37:55.4453791+05:30
```

Differences Between Stored Procedure And Function In SQL Server

The Function Must Return A Value But In stored Procedure it Is Optional. Even A Procedure Can Return Zero Or N Values.

Functions Can Have Only Input Parameters For It Whereas Procedures Can Have Input Or Output Parameters.

Functions Can Be Called From Procedure Whereas Procedures Cannot Be Called From A Function.

The procedure Allows select As Well As dml(insert/Update/Delete) statement In It Whereas Function Allows Only SELECT Statement In It.