*on*

Function is a database object in Sql Server. Basically it is a set of sql statements that accepts only input parameters, perform actions and return the result. Function can return only single value or a table. We can’t use function to Insert, Update, Delete records in the database table(s). For more about stored procedure and function refer the article[Difference between Stored Procedure and Function](http://www.dotnet-tricks.com/Tutorial/sqlserver/7EDL150912-Difference-between-Stored-Procedure-and-Function-in-SQL-Server.html)

Types of Function

1. System Defined Function

These functions are defined by Sql Server for different purpose. We have two types of system defined function in Sql Server

* 1. Scalar Function

Scalar functions operates on a single value and returns a single value. Below is the list of some useful Sql Server Scalar functions.

System Scalar Function

Scalar Function

Description

abs(-10.67)

This returns absolute number of the given number means 10.67.

rand(10)

This will generate random number of 10 characters.

round(17.56719,3)

This will round off the given number to 3 places of decimal means 17.567

upper('dotnet')

This will returns upper case of given string means 'DOTNET'

lower('DOTNET')

This will returns lower case of given string means 'dotnet'

ltrim(' dotnet')

This will remove the spaces from left hand side of 'dotnet' string.

convert(int, 15.56)

This will convert the given float value to integer means 15.

* 1. Aggregate Function

Aggregate functions operates on a collection of values and returns a single value. Below is the list of some useful Sql Server Aggregate functions.

System Aggregate Function

Aggregate Function

Description

max()

This returns maximum value from a collection of values.

min()

This returns minimum value from a collection of values.

avg()

This returns average of all values in a collection.

count()

This returns no of counts from a collection of values.

1. User Defined Function

These functions are created by user in system database or in user defined database. We three types of user defined functions.

* 1. Scalar Function

User defined scalar function also returns single value as a result of actions perform by function. We return any datatype value from function.

* + 1. ***--Create a table***
    2. **CREATE TABLE Employee**
    3. **(**
    4. **EmpID int PRIMARY KEY,**
    5. **FirstName varchar(50) NULL,**
    6. **LastName varchar(50) NULL,**
    7. **Salary int NULL,**
    8. **Address varchar(100) NULL,**
    9. **)**
    10. ***--Insert Data***
    11. **Insert into Employee(EmpID,FirstName,LastName,Salary,Address) Values(1,'Mohan','Chauahn',22000,'Delhi');**
    12. **Insert into Employee(EmpID,FirstName,LastName,Salary,Address) Values(2,'Asif','Khan',15000,'Delhi');**
    13. **Insert into Employee(EmpID,FirstName,LastName,Salary,Address) Values(3,'Bhuvnesh','Shakya',19000,'Noida');**
    14. **Insert into Employee(EmpID,FirstName,LastName,Salary,Address) Values(4,'Deepak','Kumar',19000,'Noida');**
    15. ***--See created table***
    16. **Select \* from Employee**
    17. ***--Create function to get emp full name***
    18. **Create function fnGetEmpFullName**
    19. **(**
    20. **@FirstName varchar(50),**
    21. **@LastName varchar(50)**
    22. **)**
    23. **returns varchar(101)**
    24. **As**
    25. **Begin return (Select @FirstName + ' '+ @LastName);**
    26. **end**
    27. ***--Calling the above created function***
    28. **Select dbo.fnGetEmpFullName(FirstName,LastName) as Name, Salary from Employee**
  1. Inline Table-Valued Function

User defined inline table-valued function returns a table variable as a result of actions perform by function. The value of table variable should be derived from a single SELECT statement.

* + 1. ***--Create function to get employees***
    2. **Create function fnGetEmployee()**
    3. **returns Table**
    4. **As**
    5. **return (Select \* from Employee)**
    6. ***--Now call the above created function***
    7. **Select \* from fnGetEmployee()**
  1. Multi-Statement Table-Valued Function

User defined multi-statement table-valued function returns a table variable as a result of actions perform by function. In this a table variable must be explicitly declared and defined whose value can be derived from a multiple sql statements.

* + 1. ***--Create function for EmpID,FirstName and Salary of Employee***
    2. **Create function fnGetMulEmployee()**
    3. **returns @Emp Table**
    4. **(**
    5. **EmpID int,**
    6. **FirstName varchar(50),**
    7. **Salary int**
    8. **)**
    9. **As**
    10. **begin**
    11. **Insert into @Emp Select e.EmpID,e.FirstName,e.Salary from Employee e;**
    12. ***--Now update salary of first employee***
    13. **update @Emp set Salary=25000 where EmpID=1;**
    14. ***--It will update only in @Emp table not in Original Employee table***
    15. **return**
    16. **end**
    17. ***--Now call the above created function***
    18. **Select \* from fnGetMulEmployee()**
    19. ***--Now see the original table. This is not affected by above function update command***
    20. **Select \* from Employee**

Note

1. Unlike Stored Procedure, Function returns only single value.
2. Unlike Stored Procedure, Function accepts only input parameters.
3. Unlike Stored Procedure, Function is not used to Insert, Update, Delete data in database table(s).
4. Like Stored Procedure, Function can be nested up to 32 level.
5. User Defined Function can have upto 1023 input parameters while a Stored Procedure can have upto 2100 input parameters.
6. User Defined Function can't returns XML Data Type.
7. User Defined Function doesn't support Exception handling.
8. User Defined Function can call only Extended Stored Procedure.
9. User Defined Function doesn't support set options like set ROWCOUNT etc.

Summary

In this article I try to explain the types of function in sql server with example. I hope after reading this article you will be aware about function in Sql Server. I would like to have feedback from my blog readers. Please post your feedback, question, or comments about this article.