**SUB PROGRAMS:** A sub program is a named block of code that is

directly saved on the server and it can be executed when and where it is required.

We have two types of sub programs in SQL server.

* Stored Procedures/Procedure
* Stored Functions/Functions

**Stored Procedures/Procedure:** A stored procedure is a database

object which contains precompiled queries. Stored Procedures are a block of code

designed to perform a task whenever we called.

**Why we need stored procedure**: Whenever we want to execute a

SQL query from an application the SQL query will be first parsed (i.e. complied)

for execution where the process of parsing is time consuming because parsing

occurs each and every time we execute the query or statement.

To overcome the above problem we write SQL statements or query under stored

procedure and execute, because a stored procedure is a pre complied block of code

without parsing the statements gets executed whenever the procedures are called

which can increase the performance of an application.

**Advantages of Stored Procedure:**

* As there is no unnecessary compilation of queries, this will reduce burden on database.
* Application performance will be improved
* User will get quick response
* Code reusability facility

How to Create Stored Procedures/Procedure (Without parameter):

Syntax: Create Procedures <Procedures Name>

As

Begin

<Statements>

End

Once the Procedure is created it is physically saved on the server as a Database

Object which can be called whenever we required to the user.

We can call the above procedure from anywhere and from any application that is

developed using JAVA (or) .NET languages

How to Call a Stored Procedures/Procedure:

Syntax: Exec <Procedure name>

Examples on without parameters Procedures:

1) Write a simple procedure program (with out parameters) to print WELCOME

statement on the query window.

create procedure Test1

as

begin

print 'WELCOME TO STOREDPROCEDURES'

end

Passing Parameters to Procedures: If we want to pass

parameters to procedures then we are using the below syntax.

Syntax: Create Procedures <Procedures Name>

(Passing parameters)

As

Begin

<Statements>

End

Examples on Parameter Procedures:

1) Write a program to add the two values with Parameters Procedure.

create procedure test2(@a int,@b int)

as

begin

declare @c int

set @c=@a+@b;

print 'Addition of two variables are:-'+cast(@c as varchar);

end

2) Write a program to perform arithmetic operations of two values with Parameters

Procedure.

create procedure test3(@a int,@b int)

as

begin

declare @x int,@y int,@z int,@s int

set @x=@a+@b;

set @y=@a-@b;

set @z=@a\*@b;

set @s=@a/@b;

print 'Add of two variables are:-'+cast(@x as varchar);

print 'Sub of two variables are:-'+cast(@y as varchar);

print 'Mul of two variables are:-'+cast(@z as varchar);

print 'Div of two variables are:-'+cast(@s as varchar);

end

3)create a procedure to display employee details to the user

create procedure spselect

as

begin

select \* from Employee

end

Output: execute spselect

4)create a procedure to accept employee ID and delete the record from employee

table.

create procedure spdel

@eid int

as

begin

delete from Employee where EmpID=@eid

end

Output: exec spdel 4

5)create a procedure to accept employee ID and update the employee details from

employee table.

create procedure spupdate

@eid int,@ename varchar(max),@salary money,@address char(30)

as

begin

update Employee set

EmpName=@ename,Salary=@salary,Address=@address where

EmpID=@eid

end

Output: exec spupdate 1,'kamal',88000,'vizag'

6)create a procedure to add records in employee table.

create procedure spinst

@eid int,@ename varchar(50),@salary money,@address varchar(50)

as

begin

insert into Employee values(@eid,@ename,@salary,@address)

end

Output: exec spinst 6 ,'Suman' ,41000 ,'chennai'

7)create a procedure to insert records in two tables.

create procedure spinserttwotables

@eid int,@ename varchar(50),@salary money,@Address

char(40),@Deptno int,@Dname char(30),@Loc char(20)

as

begin

insert into Employee values(@eid,@ename,@salary,@Address,@Deptno)

insert into Dept values(@Deptno,@Dname,@Loc)

end

Output: exec spinsert 7,'mohan',62000,'mumbai',10,'dotnet','hyd'

How To Drop Stored Procedure:

* Drop Procedure <Procedure Name>

Ex: Drop Procedure SP1

**Stored Functions/Functions:** Function is a block of code similar to a

stored procedure which is also used to perform an action and returns result as a

value. Function can be divided into two types, these are

1)Scalar-Valued Function: In this case we can return a attribute

datatype as an output from the function.

Syntax: Create Function <Function Name> (@parameter <Data Type> [size])

Returns <return attribute data type>

As

Begin

<Function Body>

Return <return attribute name>

End

How to Call Scalar valued Functions:

Syntax: Select <dbo>.<Function Name> (column name)

1) Create a function to return the cube of the given value.

create function fcube (@x int)

returns int

as

begin

return @x\*@x\*@x

end

Output:select dbo.fcube(3)

2) Create a function that takes an employee id and returns the salary of that

employee.

create function fsal(@eid int)

returns int

as

begin

declare @sal money

select @sal=salary from employee where empid=@eid

return @sal

end

Output:select dbo.fsal(1)

2)Table-Valued function : In this case we can return a table as an output

from the function.

Syntax:

Create Function <Function Name> (@parameter <Data Type> [size])

Returns <Table>

As

Return <return select statement>

How to Call a Table-Valued Function:

Syntax: select \* from functionname(value)

Ex: Create a function that accept the Address and returns the list of employee

working in given address from the table.

create function ft1(@add varchar(50))

returns table

as

return(select \* from employee where address=@add)

Output:select \* from ft1('hyd')

Ex:Create a function to get the deptno and return list of employee working in EMP

and DEPT tables.

create function saidata(@deptno int)

returns table

as

return(select e.eid,e.ename,e.salary,d.deptno,d.dname,d.location from

emp e inner join dept d on e.deptno=d.deptno where

e.deptno=@deptno)

Output: Select \* from saidata(10)

How To Drop Functions:

 Drop Function <Function Name>

Ex: Drop Function Saidata

Difference between Function And Procedure:

* A function must return a value where as procedure never returns a value.
* A procedure can have parameters of both input (with parameters) and output (without parameters) where as a function can have only input (with

parameters) parameters only.

* In procedure we can perform select, insert, update and delete operation

where as function can used only to perform select. Cannot be used to

perform insert, update and delete operations.

* A procedure provides the option for to perform transaction management

where as these operations are not permitted in a function.

* We call a procedure using execute command where as function are called by

using select command only