DBMS LAB
Lab Assignment number 09

Name: Aamir Ansari **Batch:** A **Roll no.** 01

Aim:- Experiment to study Stored Procedures and functions.

Theory:-
PL/SQL:- (Procedural SQL)
PL/SQL is extension of SQL.
SQL with Programming Capability.

Variable:- Is an object that can hold data value.
Local variable:-
Local variable name should begin with @ sign in SQL Server.
Local variables are used for passing values to SQL statement.
Local variables are used to store value temporary
eg. @empid

System Global variable:-
1. Global variable names begins with @@ sign.
2. eg. @@version

Declaring variable:-
General syntax:
Declare vname as datatype
Or
Declare vname datatype
Eg. Declare @eid as integer
Every variable should be defined for its datatype separately.
Begin……..end: -
Block of code is identified by using begin….end keyword.
Scope of the variable is identified by using begin and end keyword.
Eg. Begin
Declare @a as float
End
Assigning Value to the variable:-
Set or select statement is used to assign value to the variable.
Set keyword:-
Set @vname=10
Select @vname=columnname

Eg.

```
Begin
Declare @s as integer,@name as varchar(20)
Set @s=101
Select @name=ename from emp where ssn=@s
   End
```

```
select @@version          --Global Variable
--output
Microsoft SQL Server  2000
```

Control structures:-

-- If Control structure

```
begin
declare @a int,@b int,@c int General Syntax:-
set @a=10 if (condition)
set @b=20 set @c=15  begin
if(@a>@b)and(@a>@c) stmts
   print @a end
else if(@b>@c) else
     print @b begin
else print @c stmts
end end
--output 20
```

-- While loop

```
 begin
 declare @i int   General syntax:-
 set @i=1 while(cond)
 while(@i<10) begin
 begin stmts
    print 'i='+cast(@i as varchar(20))            end
--print @i
set @i=@i+1
 end
end
--output
i=1
i=2
i=3
i=4
```

i=5
i=6
i=7
i=8
i=9

-- Case Control Structures

```
begin
declare @t as varchar(20)
declare @s varchar(20) General Syntax:-
set @t=’o’ case
set @s=case when condn then exp
      when @t='o' then 'one' when condn then exp
       when @t=’t’ then ’two’ else exp
      else ‘greater than two’ end
end
end
```

Stored Procedures:-

Stored procedures are precompiled database queries that improves the security, efficiency and usability of code.

Stored procedures are extremely similar to the constructs seen in other programming languages.

They accept data in the form of input parameters that are specified at execution time. These input parameters (if implemented) are utilized in the execution of a series of statements that produce some result.

This result is returned to the calling environment through the use of a recordset, output parameters and a return code.

Stored procedures can have upto 1024 parameter.

General syntax:-

```
Create procedure procname (@v as datatype in|out)
As
Begin
   Stmts
 End
```

Calling Procedure:-

Execute procname value

```
create procedure getmonth
as
begin
select month(getdate())
end
```

```
execute getmonth
```

--Procedure without output parameter

```
-- create procedure which displays salary of emp when we pass  ssn as parameter to the procedure
create procedure listsal(@s int)
as
begin
   declare @sal int
   select @sal=emp_sal from empl1 where emp_id=@s
   print @sal
end
```

--Executing Procedure without parameter

```
execute listsal 1
```

--Procedure with output parameter

```
create procedure listsal1(@e int,@sal int output)
as
begin
      select @sal=emp_sal from empl1 where emp_id=@e
end
--Execution
begin
declare @s int
execute listsal1 2,@s output
print @s
end
```

```
--Nesting of procedure
create procedure grosssal(@e int,@d int,@hra int)
as
begin
  declare @g int,@s int
  execute listsal1 @e, @s output
   set @g=@s+(@s*(@d/100))+(@s*(@hra/100))
print 'gross sal'+ cast(@g as varchar(20))
end
```

```
execute grosssal 1,50,20
```

Advantages of Stored procedure:-

Precompiled execution. SQL Server compiles each stored procedure once and then reutilizes the execution plan. Execution speed increases.

Reduced client/server traffic.

Efficient reuse of code and programming abstraction. Stored procedures can be used by multiple users and client programs.

Enhanced security controls. You can grant users permission to execute a stored procedure independently of underlying table permissions.

Function:-

Function is precompiled set of statements which returns value explicitly to the caller of function.

```
Create funcname fname (@v as datatype) returns datatype
As
Begin
   Stmts
Return value
 End
```

Execution :-

```
Print username.functionname(value)
create function avgsal1(@d int)
returns int as
begin
  declare @avgsal int
  select @avgsal=avg(emp_sal) from empl1 where do=@d
  return @avgsal
end
```

--Execution

```
print jayshree.avgsal1(1)
select *from empl1 where emp_sal>jayshree.avgsal1(1)
```

Procedures

1. Printing numbers from 1 to 10

```
begin
        declare @a as int;
        set @a=1;
        while(@a<10)
        begin
                print 'a='+cast(@a as varchar(20))
                set @a=@a+1;
        end
end
```

```
a=1
a=2
a=3
a=4
a=5
a=6
a=7
a=8
a=9
```

2. Print the greatest of 3 numbers

```
begin
        declare @a as int,@b as int,@c as int;
        set @a=15;
        Set @b=20
        set @c=10;
        print 'The largest number is'
        if(@a>@b)and(@a>@c)
                print @a
        else if(@b>@c)
                print @b
        else
                print @c
end
```

```
The largest number is
20
```

3. Printing the table of 5

```
begin
```

```
declare @i as int;
set @i=1
declare @m as int;
set @m=5
while(@i<=10)
begin
    print cast(@m as varchar(20))+'x'+cast(@i as varchar(20))+'='+cast(@m*@i as
    varchar(20))
    set @i=@i+1;
end
end
```

```
5x1=5
5x2=10
5x3=15
5x4=20
5x5=25
5x6=30
5x7=35
5x8=40
5x9=45
5x10=50
```

Stored Procedures:

1. Without Parameters: Create a procedure which displays details of all employees

```
CREATE PROCEDURE EmployeeSelect
AS
SELECT * FROM Employee
GO;
EXEC EmployeeSelect;
```

	f_name	m_name	l_name	ssn	dob	addr	sex	salary	super_ssn	d_no
1	John	B	Smith	123456789	1965-01-09	731 Fondren, Houston, TX	M	15000.00	333445555	5
2	Franklin	T	Wong	333445555	1955-12-08	638 Voss, Houston, TX	M	40000.00	888665555	5
3	Joyce	A	English	453453453	1972-07-31	5361 Rice, Houston, TX	F	25000.00	333445555	5
4	Ramesh	K	Narayan	666884444	1962-09-15	975 Fire Oak, Humble, TX	M	38000.00	333445555	5
5	James	E	Borg	888665555	1937-11-10	450 Stone, Houston, TX	M	55000.00	NULL	1
6	Jennifer	S	Wallace	987654321	1941-06-20	291 Berrym, Bellaire, TX	F	43000.00	888665555	4
7	Ahmad	V	Jabbar	987987987	1969-03-29	980 Dallas, Houston, TX	M	25000.00	987654321	4
8	Alicia	J	Zelaya	999887777	1968-01-19	3321 Castle, Spring, TX	F	25000.00	987654321	4

2. With Parameters

a. A procedure which displays details of Employee with name 'James'

```
CREATE PROCEDURE EmpSelect @Fname varchar(30)
AS
SELECT * FROM Employee WHERE Fname=@Fname
```

```
GO
EXEC EmpSelect @Fname='James';
```

	f_name	m_name	l_name	ssn	dob	addr	sex	salary	super_ssn	d_no
1	James	E	Borg	888665555	1937-11-10	450 Stone, Houston, TX	M	55000.00	NULL	1

b. A procedure which displays details of Employee with name 'Ramesh' and salary=38000

```
CREATE PROCEDURE selectCondition @Fname varchar(30), @Salary money
AS
SELECT * FROM Employee WHERE Fname=@Fname AND Salary=@Salary
GO
EXEC selectCondition @Fname='Ramesh',@Salary=38000;
```

	f_name	m_name	l_name	ssn	dob	addr	sex	salary	super_ssn	d_no
1	Ramesh	K	Narayan	666884444	1962-09-15	975 Fire Oak, Humble, TX	M	38000.00	333445555	5

Conclusion: Hence we have successfully studied and implemented stored Procedures in DBMS