SQL SERVER – FUNCTIONS, STORED PROCEDURE AND TRIGGERS

```sql
--user defined function
--Scalar Valued function - returns single value
create function GetCountEmployee()
returns int
as
begin
   declare @noofempl int;--scalar variable
   select @noofempl=count(*) from employee;
   return @noofempl;
end;

--use function - depends on select in order to call it
select dbo.GetCountEmployee();

alter function GetCountEmployee(@city varchar(20))
returns int
as
begin
   declare @noofempl int;--scalar variable
   select @noofempl=count(*) from employee where city=@city;
   return @noofempl;
end;

select dbo.GetCountEmployee('chennai')


--Table valued function
create or alter function
GetEmployeeSal(@minsalary decimal(8,2),@maxsalary decimal(8,2))
returns table
as
return
(
   select * from Employee where salary between @minsalary and @maxsalary
)

select * from dbo.GetEmployeeSal(65000,70000)

create or alter function eastorwest(@logitude int)
returns char(4)
as
begin
   declare @res char(4)
   set @res='Same'
   if(@logitude > 0.0)
     set @res='East'
   else
         set @res='West'
       return @res
end

select dbo.eastorwest(-12)


--Stored Procedure - query to be executed repeatedly - put the query in sp
create or alter procedure sp_emp
as
begin
  select * from employee;
end
```

sp_emp

```
create or alter procedure sp_emp(@minsal decimal(8,2),@maxsal decimal(8,2))
as
begin
  select * from dbo.GetEmployeeSal(@minsal,@maxsal);
end

sp_emp 65000,80000;
```

Triggers

```
-- Create the trigger
-- Create the managers table
CREATE TABLE managers (
    id INT IDENTITY(1,1) PRIMARY KEY,
    name VARCHAR(50) NOT NULL,
    salary DECIMAL(10,2) NOT NULL
);

-- Create the trigger to modify salary before insert, by using instead of
CREATE TRIGGER update_salary_trigger
ON managers
INSTEAD OF INSERT
AS
BEGIN
    INSERT INTO managers (name, salary)
    SELECT name, salary + 1000
    FROM inserted;
END;

-- Insert data into managers table
INSERT INTO managers (name, salary) VALUES ('Riya', 15000);
-- before insert,it calls the trigger that executes function which gets incremented by 1000
--when perform select, the salary will be incremented to 13000

-- Verify the inserted data
SELECT * FROM managers;

-- Create log table
CREATE TABLE managers_log (
    log_id INT IDENTITY(1,1) PRIMARY KEY,
    name VARCHAR(50),
    salary DECIMAL(10,2),
    inserted_at DATETIME DEFAULT GETDATE()
);

-- Create trigger to log insertions
CREATE TRIGGER log_insert_trigger
ON managers
AFTER INSERT
AS
BEGIN
    INSERT INTO managers_log (name, salary)
    SELECT name, salary FROM inserted
END;

select * from managers_log
```

```
-- Create Trigger: Insert into Managers when an Employee is added
CREATE TRIGGER insert_trigger_employee_to_manager
ON Employee
AFTER INSERT
AS
BEGIN
    INSERT INTO Managers (name, salary)
    SELECT empname, salary FROM inserted;
END;

insert into Employee(empname,city,salary,dept,joiningdate)
       values('Piyush','Hyderabad',75000,'Developer','2024-10-20')

select * from Managers
```