Rahul Rathod
SUID: 303187206

11/03/2018

IST 659 LAB Assignment NO -7

1) Add an employee using your own name and create a project assignment for yourself using existing project id.

Query:

```
/*1)    Add an employee using your own name */
INSERT INTO employee VALUES (11118, 'Rahuk', 'Rathod', 100000);

/* updating the employeeFName from Rahuk to Rahul */
update employee
set employeeFName= 'Rahul'
where employeeID=11118

/*create a project assignment for yourself using existing project id.*/
INSERT INTO projectAssignment VALUES (11118, 1003, 'manager');
```

Result:

```
/*1)    Add an employee using your own name */
INSERT INTO employee VALUES (11118, 'Rahuk', 'Rathod', 100000);

/* updating the employeeFName from Rahuk to Rahul */
update employee
set employeeFName= 'Rahul'
where employeeID=11118
select * from employee
```

100 %

Results | Messages

	employeeID	employeeFName	employeeLName	employeeSalary
1	11111	James	Smith	30000
2	11112	Ada	Zack	40000
3	11113	Ben	Yale	50000
4	11114	Callen	Wales	60000
5	11115	Dale	Veller	70000
6	11116	Ethan	Miller	80000
7	11117	Fanny	Niel	90000
8	11118	Rahul	Rathod	100000

```
/*create a project assignment for yourself using existing project id.*/
INSERT INTO projectAssignment VALUES (11118, 1003, 'manager');
select * from projectAssignment
```

100 %

Results | Messages

	employeeID	projectID	projectRole
1	11111	1001	manager
2	11112	1001	PHP programmer
3	11113	1001	Oracle DBA
4	11113	1002	test engineer
5	11114	1001	quality assurance
6	11114	1002	quality assurance
7	11115	1001	test engineer
8	11115	1002	manager
9	11116	1002	test engineer
10	11117	1001	Oracle DBA
11	11117	1002	test engineer
12	11118	1003	manager

1) Write a scalar function that returns the average salary of the Employees

Query:

```
/* scalar function that returns the average salary of the Employees*/
create function averageSalary(@i int)
returns decimal(10,1)
as
begin
declare @ret INT;
select @ret= avg(employeeSalary)
from employee
return @ret;
end;

/* call the function averageSalary*/
select dbo.averageSalary(8) as averageSalary
```

Result:

```
/* scalar function that returns the average salary of the Employees*/
create function averageSalary(@i int)
returns decimal(10,1)
as
begin
declare @ret INT;
select @ret= avg(employeeSalary)
from employee
return @ret;
end;

/* call the function averageSalary*/
select dbo.averageSalary(8) as averageSalary

/* create table-valued function that returns the Projects given an EmployeeID as a paramete*/
```

100 %

Results | Messages

	averageSalary
1	65000.0

3. Write a table-valued function that returns the Projects given an EmployeeID as a parameter and
   a. Show the function created
   b. return the results for your own project

Query:

```
/* create table-valued function that returns the Projects given an EmployeeID as a paramete*/
create function employeeProject(@employeeID int)
returns table
as
return
(
select pa.projectId as employeeProject
from projectAssignment pa
Inner Join employee e on pa.employeeID = e.employeeID
Inner Join project p on pa.projectID = p.projectID
where e.employeeID = @employeeID
)
drop function dbo.employeeProject

/* call the function employeeProject*/
select * from employeeProject(11118);
```

Result:

```
/* create table-valued function that returns the Projects given an EmployeeID as a paramete*/
create function employeeProject(@employeeID int)
returns table
as
return
(
select pa.projectId as employeeProject
from projectAssignment pa
Inner Join employee e on pa.employeeID = e.employeeID
Inner Join project p on pa.projectID = p.projectID
where e.employeeID = @employeeID
)

/* call the function employeeProject*/
select * from employeeProject(11118);
```

100 %

Results | Messages

	employeeProject
1	1003

4. Alter the Employee table to add a new column called 'Num of Projects' which can be **INTEGER** data type. Write a procedure that updates employee table with the total projects assigned to each employee (to the newly created column)

Query:

```
/*Alter the Employee table to add a new column called 'Num of Projects' */
alter table employee add NumOfProjects int;
select * from employee

CREATE PROCEDURE TOTAL_PROJ
AS
BEGIN
UPDATE EMPLOYEE
SET NumOfProjects = projCount.total_count
FROM
(
SELECT employeeID,count(projectID) AS total_count
FROM projectAssignment
GROUP BY employeeID
) projCount
WHERE employee.employeeID=projCount.employeeID
END;
```

EXEC TOTAL_PROJ;

Result:

```
/*Alter the Employee table to add a new column called 'Num of Projects' */
alter table employee add NumOfProjects int;
select * from employee
```

100 %

Results | Messages

	employeeID	employeeFName	employeeLName	employeeSalary	NumOfProjects
1	11111	James	Smith	30000	NULL
2	11112	Ada	Zack	40000	NULL
3	11113	Ben	Yale	50000	NULL
4	11114	Callen	Wales	60000	NULL
5	11115	Dale	Veller	70000	NULL
6	11116	Ethan	Miller	80000	NULL
7	11117	Fanny	Niel	90000	NULL
8	11118	Rahul	Rathod	100000	NULL

```
CREATE PROCEDURE TOTAL_PROJ
AS
BEGIN
UPDATE EMPLOYEE
SET NumOfProjects = projCount.total_count
FROM
(
SELECT employeeID,count(projectID) AS total_count
FROM projectAssignment
GROUP BY employeeID
) projCount
WHERE employee.employeeID=projCount.employeeID
END;

EXEC TOTAL_PROJ;
select * from employee
```

100 %

Results | Messages

	employeeID	employeeFName	employeeLName	employeeSalary	NumOfProjects
1	11111	James	Smith	30000	1
2	11112	Ada	Zack	40000	1
3	11113	Ben	Yale	50000	2
4	11114	Callen	Wales	60000	2
5	11115	Dale	Veller	70000	2
6	11116	Ethan	Miller	80000	1
7	11117	Fanny	Niel	90000	2
8	11118	Rahul	Rathod	100000	1

5. Create a trigger that can update the num of projects whenever a new project is assigned to an employee.
   Test the trigger with the below insert
   a. INSERT INTO projectAssignment VALUES (11114, 1003, 'quality assurance');
   b. INSERT INTO projectAssignment VALUES (11115, 1003,'test engineer');

Query:

```
CREATE TRIGGER NumofProjectsTrigger
ON projectAssignment
FOR INSERT,UPDATE
AS
IF @@ROWCOUNT >= 1
BEGIN
```

```
UPDATE EMPLOYEE
SET NumOfProjects = projCount.total_count
FROM
(
SELECT employeeID,count(projectID) AS total_count
FROM projectAssignment
GROUP BY employeeID
) projCount
WHERE employee.employeeID=projCount.employeeID
END;


INSERT INTO projectAssignment VALUES (11114, 1003, 'quality assurance');
INSERT INTO projectAssignment VALUES (11115, 1003,'test engineer');
```

Result:

```
CREATE TRIGGER NumofProjectsTrigger
ON projectAssignment
FOR INSERT,UPDATE
AS
IF @@ROWCOUNT >= 1
BEGIN
UPDATE EMPLOYEE
SET NumOfProjects = projCount.total_count
FROM
(
SELECT employeeID,count(projectID) AS total_count
FROM projectAssignment
GROUP BY employeeID
) projCount
WHERE employee.employeeID=projCount.employeeID
END;
```

100 %

Results | Messages

	employeeID	employeeFName	employeeLName	employeeSalary	NumOfProjects
1	11111	James	Smith	30000	1
2	11112	Ada	Zack	40000	1
3	11113	Ben	Yale	50000	2
4	11114	Callen	Wales	60000	3
5	11115	Dale	Veller	70000	3
6	11116	Ethan	Miller	80000	1
7	11117	Fanny	Niel	90000	2
8	11118	Rahul	Rathod	100000	1