Task Manager System

Functions and Stored Procedures

Dr. Sara Nagy

Team ID : 78

Ahmed Mohamed Hussein	7
Hossam Shaaban Ahmed	6
Karim Atef Mostafa	12
Mahmoud Emad	15
Nader Waled Mohamed	18

1. Procedure to insert in employee table:

```
create procedure insert_employee
@name nvarchar(50), @gender nvarchar(10),@password nvarchar(40),
@adress text,@phone nvarchar(40),@mail nvarchar(40),
@joindate date,@department nvarchar(40),@rank nvarchar(40),
@admin_id int,@manegar_id int,@salary_per_hour int,@hours int
AS
insert into
employee(employee_name,employee_gender,employee_password,employee_address,
employee_phone,employee_mail,employee_join_date,employee_department,admin_id,
manager_id,employee_rank,salary_per_hour,hours)
values(@name,@gender,@password,@adress,@phone,@mail,@joindate,@department,
@rank,@admin_id,@manegar_id,@salary_per_hour,@hours)
```

2. Procedure to insert in project table:

```
create procedure insert_project
@name nvarchar(50),@description text,
@start_time date,@end_time date,
@client_id int,@status nvarchar(40),@proiorty int
AS
 insert into project(project_name,project_description,project_start_time,project_end_time,
project_client_id,project_status,project_priority)
values(@name,@description,@start_time,@end_time,@client_id,@status,@proiorty)
```

3. Procedure to delete from task table:

```
create procedure delete_task
@id int
AS
DELETE FROM task
WHERE task_id=@id
```

4. Function to login Admin:

```
create function fn_login_admin

(@username nvarchar(50),@pass nvarchar(50))

returns table

AS

return

(select * from admin

where admin.user_name=@username and admin.password=@pass)
```

5. Procedure to insert in task table:

```
create procedure insert_task

@name nvarchar(40),@description nvarchar(40),

@start_date date,@end_date date,

@project_id int,@employee_id int,

@status nvarchar(50),@priority int

AS

insert into task
(task_name,task_description,task_start_date,task_end_date,task_project_id,
task_employee_id,task_status,task_priority )

values
(@name,@description,@start_date,@end_date,@priority,@employee_id,@status,
@priority )
```

6. Procedure to insert in client table:

```
create procedure insert_client

@name nvarchar(40),

@phone nvarchar(40),

@mail nvarchar(40)

AS

insert into client(client_name,client_phone,client_mail)
values (@name,@phone,@mail)
```

7. Procedure to update project data:

```
 create procedure edit_project

@id int,

@name nvarchar(60),

@description text,

@start_time date,

@end_time date,

@client_id int,

@status nvarchar(40),

@proiorty int

AS

update project set project_name=@name , project_description=@description ,
project_start_time=@start_time,project_end_time=@end_time,
project_client_id=@client_id ,  project_status=@status,project_priority=@proiorty

where project_id=@id
```

8. Procedure to update task data:

```
create procedure edit_task

@id int,@name nvarchar(60),@description text,

@start_date date,@end_date date,

@project_id int,

@employee_id int,

@status nvarchar(40),

@priority int

AS

update task set task_name=@name , task_description=@description ,

task_start_date=@start_date,task_end_date=@end_date,

task_project_id=@project_id,task_employee_id=@employee_id , task_status=@status ,

task_priority=@priority where task_id=@id
```

9. Procedure to update employee data:

```
create procedure edit_employee

@id int,@name nvarchar(50),@gender nvarchar(50),

@passowrd nvarchar(50),@adress text,

@phone nvarchar(50),

@mail nvarchar(50),

@join_date date,

@department nvarchar(50),

@rank nvarchar(50),

@Manager_id int,

@salary_per_hour int,

@hours int

AS

update employee set employee_name=@name ,
employee_gender=@gender,employee_password=@passowrd,
employee_address=@adress, employee_phone=@phone, employee_mail=@mail,
employee_join_date=@join_date,employee_department=@department,
employee_rank=@rank,manager_id=@Manager_id,
salary_per_hour=@salary_per_hour,hours=@hours where employee_id =@id
```

10. Procedure to delete from employee table:

```
create procedure delete_employee

@id int

AS

delete from employee

where employee_id=@id
```

11. Function to login employee:

```
create function fn_login_employee

(@username nvarchar(50),@pass nvarchar(50))

returns table

AS

return

( select * from employee

where employee.employee_name=@username and employee.employee_password=@pass)
```

12. Function to search for ID of manager by name :

```
create function fn_search_id_manager

(@manegar_name nvarchar(50))

returns table AS

return

( select employee_id from employee

where employee.employee_name=@manegar_name )
```

13. Function to view manager’s employee:

```
create function fn_view_employee

(@manegar_id int)

returns table AS

return

( select employee_name,employee_gender,employee_address,employee_phone,
employee_mail,employee_join_date,employee_department,employee_rank,
salary_per_hour,hours

 from employee

 where employee.manager_id=@manegar_id )
```

14. Function to calculate employee’s salary:

```
create function fn_calculate_salary

(@employee_id int)

returns table AS

return

( select employee.salary_per_hour * employee.hours  as salary

 from employee

 where employee.employee_id =@employee_id )
```

15. Function to view manager’s project:

```
create function fn_view_project

(@manegar_id int)

returns table as

return

( select p.project_name,p.project_description,p.project_start_time,p.project_end_time,

p.project_status,p.project_priority,c.client_name,c.client_phone,c.client_mail,t.task_name,

e.employee_name

 from project p LEFT OUTER JOIN task t

 on p.project_id=t.task_project_id,

 project inner join client c

 on project.project_client_id=c.client_id,

 task inner join employee e

 on task.task_employee_id=e.employee_id

 where p.project_id=t.task_project_id and t.task_employee_id=e.employee_id
and e.manager_id=@manegar_id )
```