

Assignment - 3

DBMS

Name: Kamithkar Vinod

Course: PG DAC AUGUST 2025

PRN: 250850320040

Form No: 250500480

Date: 23-10-2025

Problem 1:

Task: Create a procedure to reset all employee salaries to 50000.

Code: —

```
delimiter //
create procedure ResetAllSalaries()
begin
    update employees
    set salary = 50000
    where empid > 0;
end //
delimiter ;

call ResetAllSalaries();

select * from employees;
```

Output: —

emp_id	name	salary	department
1	Alice	50000.00	HR
2	Bob	50000.00	IT
3	Charlie	50000.00	Finance
NULL	NULL	NULL	NULL

Problem 2:

Task: Create a procedure to delete all employees in the HR department.

Code: —

```
delimiter //
create procedure delete_hr_emp()
begin
    delete from employees
    where department = 'hr';
end //
delimiter ;

call delete_hr_emp();

select * from employees;
```

Output: —

	emp_id	name	salary	department
▶	2	Bob	50000.00	IT
	3	Charlie	50000.00	Finance
✱	NULL	NULL	NULL	NULL

Problem 3:

Task: Create a procedure to increase all employee salaries by 5

Code: —

```
delimiter //
create procedure inc_sal()
begin
    update employees
    set salary = salary + salary * (5/100);
end //
delimiter ;

call inc_sal();

select * from employees;
```

Output: —

	emp_id	name	salary	department
▶	2	Bob	52500.00	IT
	3	Charlie	52500.00	Finance
●	NULL	NULL	NULL	NULL

Problem 4:

Task: Create a procedure to insert a new employee (IN parameters).

Code: —

```

delimiter //
create procedure add_emp(in emp_name varchar(50), in emp_sal
    decimal(10,2), in emp_dept varchar(30))
begin
    insert into employees (name, salary, department)
    values (emp_name, emp_sal, emp_dept);
end //
delimiter ;

call add_emp('sony', 43000, 'IT');

select * from employees;
```

Output: —

	emp_id	name	salary	department
▶	2	Bob	52500.00	IT
	3	Charlie	52500.00	Finance
	4	vinod	52500.00	IT
	5	sony	43000.00	IT
●	NULL	NULL	NULL	NULL

Problem 5:

Task: Create a procedure to insert a new department (IN parameters).

Code: —

```

delimiter //
create procedure addDept(
    in d_name varchar(30),
    in loc varchar(30))
begin
    insert into departments(dept_name, location)
    values (d_name, loc);
```

```

end //
delimiter ;

call addDept('Physics', 'Hyderabad');

select * from departments;

```

Output: —

	dept_id	dept_name	location
▶	1	HR	Hyderabad
	2	IT	Bangalore
	3	Finance	Delhi
	4	Physics	Hyderabad
⬇	NULL	NULL	NULL

Problem 6:

Task: Create a procedure to delete an employee by name (IN parameter).

Code: —

```

delimiter //
create procedure del_emp_name(in n varchar(50))
begin
    delete from employees
    where name = n;
end //
delimiter ;

call del_emp_name('vinod');

select * from employees;

```

Output: —

emp_id	name	salary	department
2	Bob	52500.00	IT
3	Charlie	52500.00	Finance
5	sony	43000.00	IT

Problem 7:

Task: Create a procedure to change an employee's department (IN parameters).

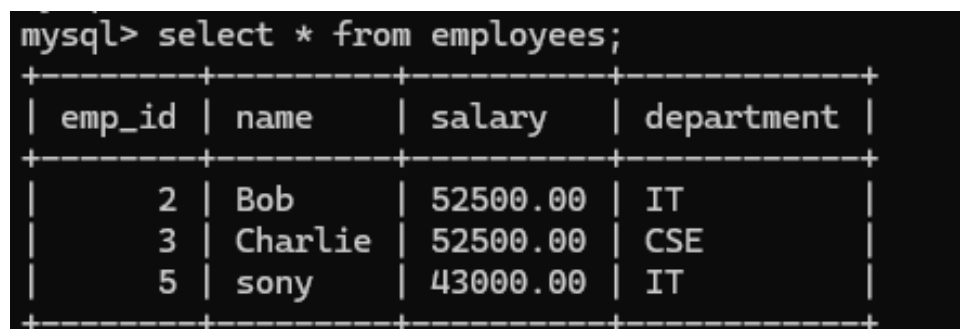
Code: —

```
delimiter //
create procedure empDeptChange(
    in e_id int,
    in dept varchar(30))
begin
    update employees
    set department = dept
    where emp_id = e_id;
end //
delimiter ;

call empDeptChange(3, 'CSE');

select * from employees;
```

Output: —



```
mysql> select * from employees;
```

emp_id	name	salary	department
2	Bob	52500.00	IT
3	Charlie	52500.00	CSE
5	sony	43000.00	IT

Problem 8:

Task: Create a procedure to get the highest salary (OUT parameter).

Code: —

```
delimiter //
create procedure empHighestSalary(out h_sal decimal(10, 2))
begin
    select max(salary) into h_sal from employees;
end //
delimiter ;

call empHighestSalary(@highestSal);

select @highestSal as HighestSalary;
```

Output: —

```
mysql> select @highestSal as HighestSalary;
+-----+
| HighestSalary |
+-----+
|      63000.00 |
+-----+
```

Problem 9:

Task: Create a procedure to get average salary (OUT parameter).

Code: —

```
delimiter //
create procedure empAvgSal(out avgSal decimal(10, 2))
begin
    select avg(salary) into avgSal from employees;
end //
delimiter ;

call empAvgSal(@AvgSal);

select @AvgSal as EmployeesAverageSalary;
```

Output: —

```
mysql> select @AvgSal as EmployeesAverageSalary;
+-----+
| EmployeesAverageSalary |
+-----+
|           52750.00 |
+-----+
```

Problem 10:

Task: Create a procedure to get department count (OUT parameter).

Code: —

```
delimiter //
create procedure deptCount(out d_count int)
begin
    select count(*) into d_count from departments;
end //
delimiter ;

call deptCount(@dept_count);
```

```
select @dept_count as DepartmentCount;  
  
select * from departments;
```

Output: —

```
mysql>  
-> select @dept_count as DepartmentCount;  
+-----+  
| DepartmentCount |  
+-----+  
|                4 |  
+-----+
```

Problem 11:

Task: Create a procedure to get an employee's name by ID (IN and OUT parameter).

Code: —

```
delimiter //  
create procedure getEmpName(in e_id int, out e_name varchar(50))  
begin  
    select name into e_name from employees where emp_id =  
        e_id;  
end //  
delimiter ;  
  
call getEmpName(5, @emp_name);  
  
select @emp_name as EmployeeName;
```

Output: —

```
mysql>  
mysql> select @emp_name as EmployeeName;  
+-----+  
| EmployeeName |  
+-----+  
| sony         |  
+-----+
```

Problem 12:

Task: Create a procedure to increase salary of an employee by a given percentage (IN parameters).

Code: —

```
delimiter //
create procedure incEmpSal(in e_id int, in increase int)
begin
    update employees
    set salary = salary + (salary * (increase/100)) where emp_id
    = e_id;
end //
delimiter ;

call incEmpSal(5, 10);

select * from employees;
```

Output: —

A screenshot of a MySQL terminal window. The prompt is 'mysql> select * from employees;'. The output is a table with 4 columns: emp_id, name, salary, and department. The data rows are: (2, Bob, 52500.00, IT), (3, Charlie, 52500.00, CSE), (5, sony, 47300.00, IT), and (6, bhujanga, 63000.00, IT). The table is displayed with a grid of dashes and plus signs.

emp_id	name	salary	department
2	Bob	52500.00	IT
3	Charlie	52500.00	CSE
5	sony	47300.00	IT
6	bhujanga	63000.00	IT

Problem 13:

Task: Create a procedure to add a bonus to an employee and return updated salary (INOUT parameter).

Code: —

```
delimiter //
create procedure addBonusToEmployee(in e_id int, inout emp_sal
    decimal(10, 2), in bonus decimal(10, 2))
begin
    -- step 1: fetch current salary
    select salary into emp_sal
    from employees where emp_id = e_id;

    -- step 2: add bonus

    set emp_sal = emp_sal + bonus;

    -- step 3: update employee salary
    update employees
    set salary = emp_sal
    where emp_id = e_id;
```



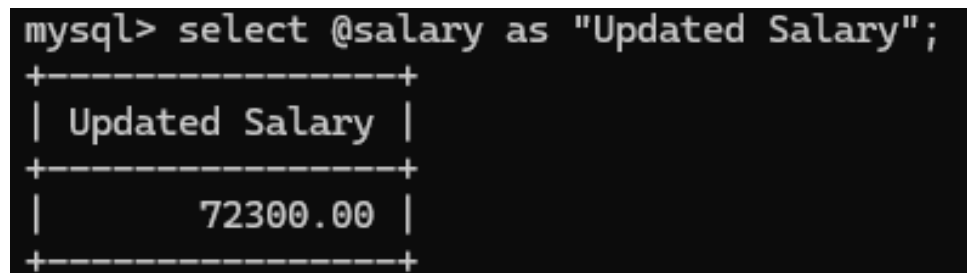
```
end //
delimiter ;

set @salary = 0;

call addBonusToEmployee(5, @salary, 25000);

select @salary as "Updated Salary";
```

Output: —



```
mysql> select @salary as "Updated Salary";
+-----+
| Updated Salary |
+-----+
|          72300.00 |
+-----+
```

Problem 14:

Task: Create a procedure to move an employee to another department and return new department name (INOUT parameter).

Code: —

```
delimiter //
create procedure MoveEmployeeToDept(in e_id int, inout new_dept
    varchar(30))
begin
    update employees
    set department = new_dept
    where emp_id = e_id;
end //
delimiter ;

set @dept = 'dac';
call MoveEmployeeToDept(6, @dept);

select * from employees;
```

Output: —

```
mysql> set @dept = 'dac';
Query OK, 0 rows affected (0.00 sec)

mysql> call MoveEmployeeToDept(6, @dept);
Query OK, 1 row affected (0.01 sec)

mysql> select * from employees;
```

emp_id	name	salary	department
2	Bob	52500.00	IT
3	Charlie	52500.00	CSE
5	sony	72300.00	IT
6	bhujanga	63000.00	dac

Problem 15:

Task: Create a procedure to change department location and return updated location (INOUT parameter).

Code: —

```
delimiter //
create procedure changeDeptLoc(in d_id int, inout d_loc varchar
(30))
begin
    update departments
    set location = d_loc
    where dept_id = d_id;
end //
delimiter ;

set @loc = 'Pune';
call changeDeptLoc(4, @loc);

select @loc as updated_location;

select * from departments;
```

Output: —

```
mysql> set @loc = 'Pune';
Query OK, 0 rows affected (0.00 sec)

mysql> call changeDeptLoc(4, @loc);
Query OK, 1 row affected (0.01 sec)

mysql> select * from departments;
+-----+-----+-----+
| dept_id | dept_name | location |
+-----+-----+-----+
|      1 | HR        | Hyderabad |
|      2 | IT        | Bangalore |
|      3 | Finance   | Delhi     |
|      4 | Physics   | Pune      |
+-----+-----+-----+
```

Problem 16:

Task: Create a procedure to show employees earning above a given salary (IN parameter).

Code: —

```
delimiter //
create procedure empAboveSal(in sal decimal(10, 2))
begin
    select * from employees where salary > sal;
end //
delimiter ;

call empAboveSal(55000);
```

Output: —

```
mysql> call empAboveSal(55000);
+-----+-----+-----+-----+
| emp_id | name     | salary  | department |
+-----+-----+-----+-----+
|      5 | sony     | 72300.00 | IT         |
|      6 | bhujanga | 63000.00 | dac        |
+-----+-----+-----+-----+
```

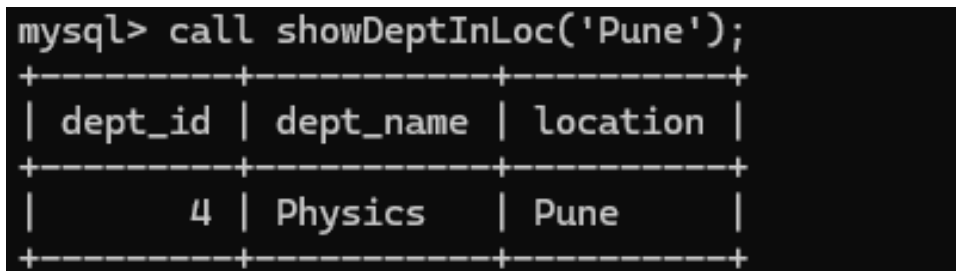
Problem 17:

Task: Create a procedure to show all departments in a specific location (IN parameter).

Code: —

```
delimiter //
create procedure showDeptInLoc(in loc varchar(30))
begin
    select * from departments where location = loc;
end //
delimiter ;

call showDeptInLoc('Pune');
```

Output: —

```
mysql> call showDeptInLoc('Pune');
+-----+-----+-----+
| dept_id | dept_name | location |
+-----+-----+-----+
|      4 | Physics   | Pune     |
+-----+-----+-----+
```

Problem 18:

Task: Create a procedure to delete a department by name (IN parameter).

Code: —

```
delimiter //
create procedure delDeptByName(in d_name varchar(30))
begin
    delete from departments where dept_name = d_name;
end //
delimiter ;

call delDeptByName('Physics');

select * from departments;
```

Output: —

```
mysql> call delDeptByName('Physics');
Query OK, 1 row affected (0.01 sec)

mysql> select * from departments;
+-----+-----+-----+
| dept_id | dept_name | location |
+-----+-----+-----+
|      1 | HR        | Hyderabad |
|      2 | IT        | Bangalore |
|      3 | Finance   | Delhi     |
+-----+-----+-----+
```

Problem 19:

Task: Create a procedure to find the total salary paid in a department (IN and OUT parameter).

Code: —

```
delimiter //
create procedure totalSalDeptWise(in d_name varchar(30), out
    t_sal decimal(10, 2))
begin
    select sum(salary) into t_sal from employees where
        department = d_name;
end //
delimiter ;

call totalSalDeptWise('IT', @totalSal);

select @totalSal as TotalSalary;

select * from employees;
```

Output: —

```
mysql> call totalSalDeptWise('IT', @totalSal);
Query OK, 1 row affected (0.00 sec)

mysql> select @totalSal as TotalSalary;
+-----+
| TotalSalary |
+-----+
| 124800.00 |
+-----+
```

Problem 20:

Task: Create a procedure to find the minimum salary and return it (OUT parameter).

Code: —

```
delimiter //
create procedure MinSal(out sal decimal(10, 2))
begin
    select min(salary) into sal
    from employees;
end //
delimiter ;

call MinSal(@minSalary);

select @minSalary as MinimumSalary;

select * from employees;
```

Output: —

```
mysql> select @minSalary as MinimumSalary;
+-----+
| MinimumSalary |
+-----+
|      52500.00 |
+-----+
```