ASSIGNMENT- 3 DBMS

-Yash Yadav(107)

1. Create a procedure to reset all employee salaries to 50000.

use pgdac;

drop procedure if exists reset;

DELIMITER //

CREATE PROCEDURE reset( )

BEGIN

SET SQL_SAFE_UPDATES = 0;

update emp set sal='50000';   SET

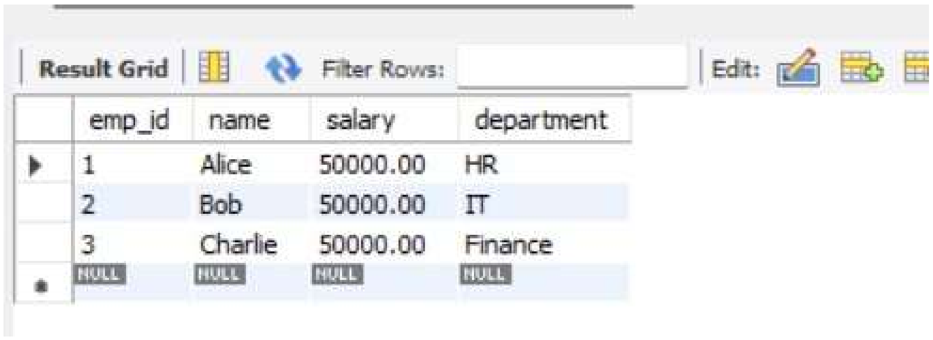
SQL_SAFE_UPDATES = 1;

select*from emp;

END //

DELIMITER //

CALL reset();



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

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

use labdb;

drop procedure if exists deleteHR;

DELIMITER //

CREATE PROCEDURE deleteHR( )

BEGIN

SET SQL_SAFE_UPDATES = 0;

delete from employees where department='HR';

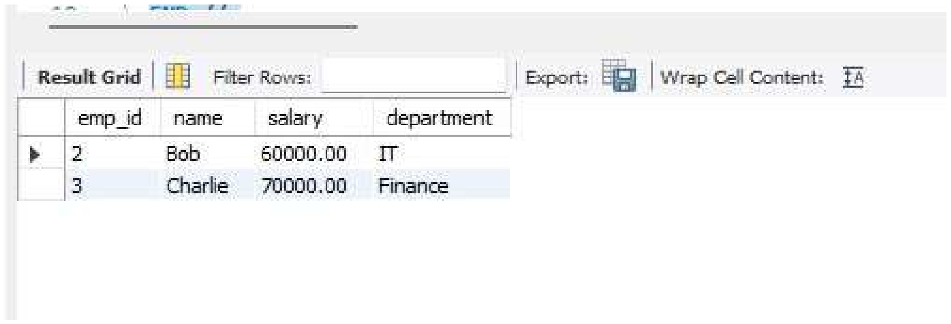
SET SQL_SAFE_UPDATES = 1; select*from

employees;

END //

DELIMITER //

CALL deleteHR();



Result Grid | Filter Rows: | Export: | Wrap Cell Content:

emp_id	name	salary	department
2	Bob	60000.00	IT
3	Charlie	70000.00	Finance

3. Create a procedure to increase all employee salaries by 5%.

use labdb;

drop procedure if exists increaseSal;

DELIMITER //

CREATE PROCEDURE increaseSal( )

BEGIN

SET SQL_SAFE_UPDATES = 0;

update employees set salary=salary+(salary*0.05);

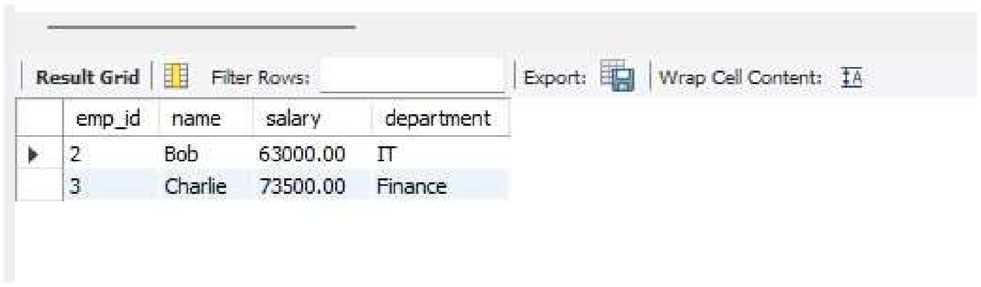
SET SQL_SAFE_UPDATES = 1;

select*from employees;

END //

DELIMITER //

CALL increaseSal();


Result Grid | Filter Rows: | Export: | Wrap Cell Content:

	emp_id	name	salary	department
▶	2	Bob	63000.00	IT
	3	Charlie	73500.00	Finance

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

use labdb;

drop procedure if exists insertEmp;

DELIMITER //

CREATE PROCEDURE insertEmp(in id int,in name varchar(40),in salary decimal(10,2),in department varchar(50))

BEGIN

SET SQL_SAFE_UPDATES = 0;

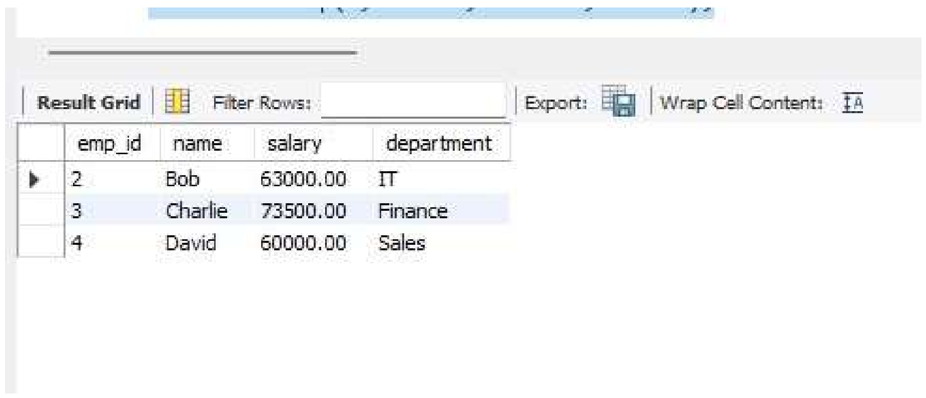
insert into employees values(id,name,salary,department);

SET SQL_SAFE_UPDATES = 1;   select*from employees;

END //

DELIMITER //

CALL insertEmp(4,'David',60000.00,'Sales');

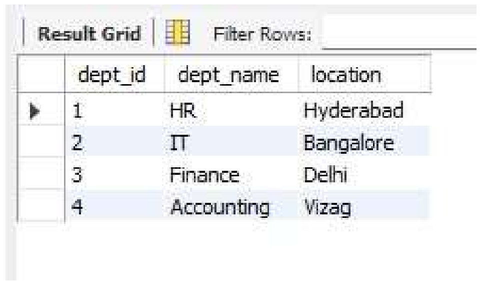


Result Grid | Filter Rows: | Export: | Wrap Cell Content:

emp_id	name	salary	department
2	Bob	63000.00	IT
3	Charlie	73500.00	Finance
4	David	60000.00	Sales

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

```
use labdb;
drop procedure if exists dept_insertion; DELIMITER
//
create procedure dept_insertion(in id int,in dept_name varchar(20),in location
varchar(20)) begin
SET SQL_SAFE_UPDATES = 0;
insert into departments values(id,dept_name,location);
SET SQL_SAFE_UPDATES = 1; select * from
departments; end//
DELIMITER ;
CALL dept_insertion(4,'Accounting','Vizag');
```

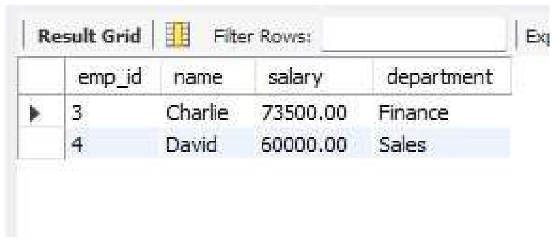


Result Grid | Filter Rows:

dept_id	dept_name	location
1	HR	Hyderabad
2	IT	Bangalore
3	Finance	Delhi
4	Accounting	Vizag

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

```
use labdb;
drop procedure if exists delete_employee; DELIMITER
//
create procedure delete_employee(in del_name varchar(20)) begin
SET SQL_SAFE_UPDATES = 0;
delete from employees where del_name=name;
SET SQL_SAFE_UPDATES = 1; select * from
employees; end//
DELIMITER ;
CALL delete_employee('Bob');
```



emp_id	name	salary	department
3	Charlie	73500.00	Finance
4	David	60000.00	Sales

7. Create a procedure to change an employee’s department (IN parameters).

```
use labdb;
drop procedure if exists change_department; DELIMITER
//
```

create procedure change_department(in p_name varchar(20),in new_dept varchar(20)) begin

SET SQL_SAFE_UPDATES = 0;

update employees set

department=new_dept

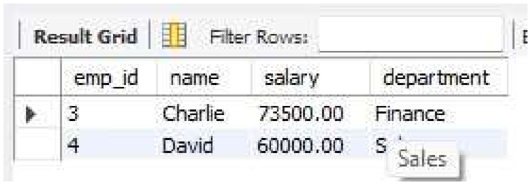
where name=p_name;

select * from employees;

end//

DELIMITER ;

CALL change_department('Alice','HR');



	emp_id	name	salary	department
▶	3	Charlie	73500.00	Finance
	4	David	60000.00	Sales

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

use labdb;

drop procedure if exists highest_salary; delimiter

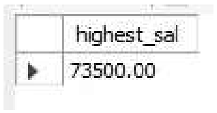
//

create procedure highest_salary(out largest_salary decimal(10,2)) begin

select max(salary) into largest_salary from employees;

end // delimiter ; call highest_salary(@sal); select

@sal as highest_sal;



	highest_sal
▶	73500.00

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

use labdb;

drop procedure if exists average_salary; delimiter

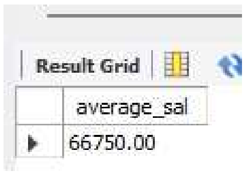
//

create procedure average_salary(out average_salary decimal(10,2)) begin

select avg(salary) into average_salary from employees;

end // delimiter ; call average_salary(@sal);

select @sal as average_sal;



```
average_sal
66750.00
```

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

use labdb;

drop procedure if exists dept_count; delimiter

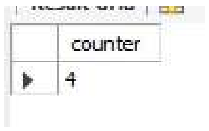
//

create procedure dept_count(out count_dept int) begin

select count(*) into count_dept from departments;

end // delimiter ; call dept_count(@count); select

@count as counter;



	counter
▶	4

11. Create a procedure to get an employee’s name by ID (IN and OUT parameter).

use labdb;

drop procedure if exists fetch_name; delimiter

//

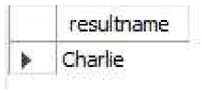
create procedure fetch_name(in id int , out emp_name varchar(20)) begin

select name into emp_name from employees where emp_id=id;

end // delimiter ;

call fetch_name(3,@names); select

@names as resultname;



	resultname
▶	Charlie

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

use labdb;

drop procedure if exists increaseSalary;

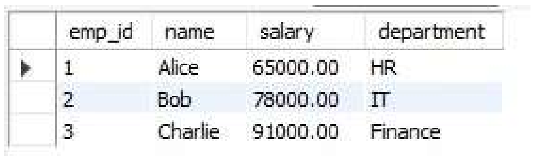
DELIMITER //

CREATE PROCEDURE increaseSalary(in percent decimal(4,2))

BEGIN

SET SQL_SAFE_UPDATES = 0;

```
    update employees set salary=salary+(salary*(percent/100));
SET SQL_SAFE_UPDATES = 1;    select*from employees;
END //
DELIMITER //
CALL increaseSalary(30);
```



	emp_id	name	salary	department
▶	1	Alice	65000.00	HR
	2	Bob	78000.00	IT
	3	Charlie	91000.00	Finance

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

```
use labdb;
drop procedure if exists updateSalary;
DELIMITER //
CREATE PROCEDURE updateSalary(in id int,inout sal decimal(10,2) )
BEGIN

SET SQL_SAFE_UPDATES = 0;
    update employees set salary=salary+sal where emp_id=id;

  SET SQL_SAFE_UPDATES = 1;
  select salary into sal from employees  where emp_id=id;
```

END // DELIMITER

// set

@salary=3000;

CALL

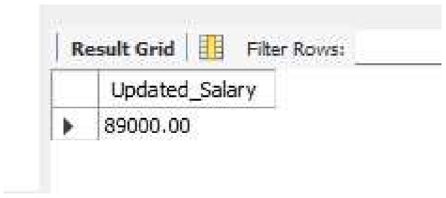
updateSalary(1,@

salary); select

@salary as

Updated_Salary;

/*select * from employees;*/



```
Updated_Salary
89000.00
```

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

use labdb;

drop procedure if exists changeDepartment;

DELIMITER //

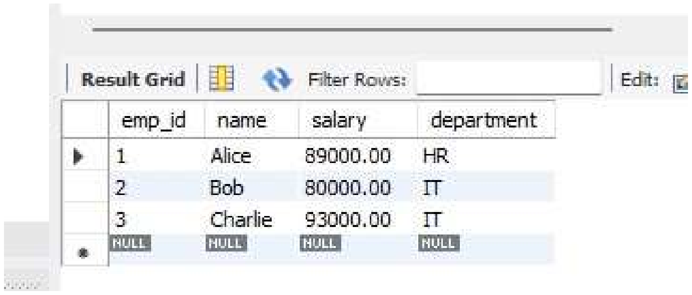
CREATE PROCEDURE changeDepartment(in id int,inout dept varchar(30))

BEGIN

SET SQL_SAFE_UPDATES = 0;

    update employees set department=dept where emp_id=id;

```
  SET SQL_SAFE_UPDATES = 1;
  select department into dept from employees  where emp_id=id;
END //
DELIMITER // set
@dept='IT';
CALL
changeDepartm
ent(3,@dept);
select @dept as
New_Dept;

 select * from employees;
```

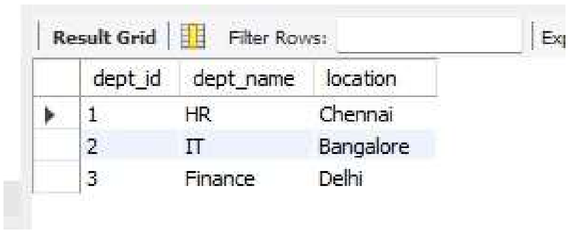


```
   emp_id  name     salary    department
▶  1       Alice    89000.00  HR
   2       Bob      80000.00  IT
   3       Charlie  93000.00  IT
●  NULL    NULL     NULL      NULL
```

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

```
use labdb;
drop procedure if exists changeLocation;
```

```
DELIMITER //
CREATE PROCEDURE changeLocation(in id int,inout loc varchar(30) )
BEGIN


SET SQL_SAFE_UPDATES = 0;
    update departments set location=loc where dept_id=id;


   SET SQL_SAFE_UPDATES = 1;
   select location into loc from departments  where dept_id=id;
END // DELIMITER // set
@loc='Chennai'; CALL
changeLocation(1,@loc);
select @loc as New_loc;


 select * from departments;
```

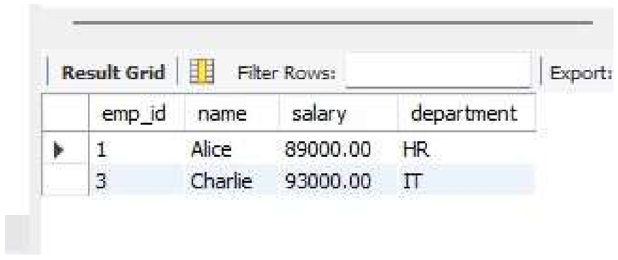


dept_id	dept_name	location
1	HR	Chennai
2	IT	Bangalore
3	Finance	Delhi

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

```
use labdb;
drop procedure if exists empAboveGivenSal;
DELIMITER //
```

```
CREATE PROCEDURE empAboveGivenSal(in sal int )
BEGIN

SET SQL_SAFE_UPDATES = 0;

    select*from employees where salary>sal;
   SET SQL_SAFE_UPDATES = 1;

END //
DELIMITER //
CALL empAboveGivenSal(85000);
```

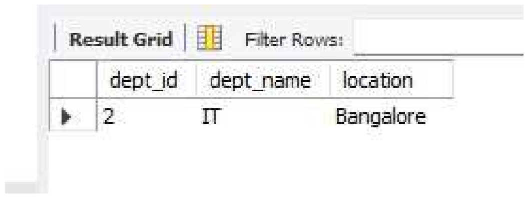


emp_id	name	salary	department
1	Alice	89000.00	HR
3	Charlie	93000.00	IT

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

```
use labdb;
drop procedure if exists deptLoc;
DELIMITER //
CREATE PROCEDURE deptLoc(in loc varchar(20) )
BEGIN
```

```
SET SQL_SAFE_UPDATES = 0;


    select*from departments where location=loc;
   SET SQL_SAFE_UPDATES = 1;


END //
DELIMITER //
CALL deptLoc('Bangalore');
```

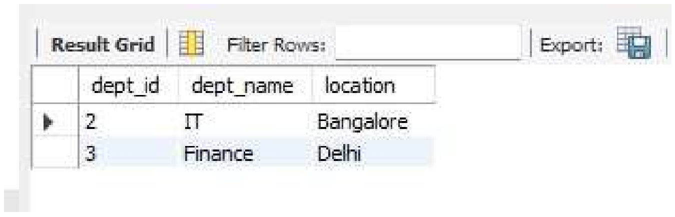


dept_id	dept_name	location
2	IT	Bangalore

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

```
use labdb;
drop procedure if exists deleteDept;
DELIMITER //
CREATE PROCEDURE deleteDept(in name varchar(20) )
BEGIN

SET SQL_SAFE_UPDATES = 0;
```

```
    delete from departments where dept_name=name;
select*from departments;
  SET SQL_SAFE_UPDATES = 1;

END //
DELIMITER //
CALL deleteDept('HR');
```



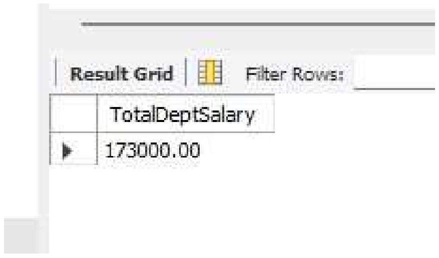
Result Grid | Filter Rows: | Export:

dept_id	dept_name	location
2	IT	Bangalore
3	Finance	Delhi

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

```
use labdb;
drop procedure if exists totalSalDept;
DELIMITER //
CREATE PROCEDURE totalSalDept(in deptName varchar(20),out sal
decimal(10,2) )
BEGIN

SET SQL_SAFE_UPDATES = 0;

    select sum(salary) into sal from employees where department=deptName ;
```

```
  SET SQL_SAFE_UPDATES = 1;

END //
DELIMITER //
CALL totalSalDept('IT',@totalSal); select
@totalSal as TotalDeptSalary;
```



	TotalDeptSalary
▶	173000.00

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

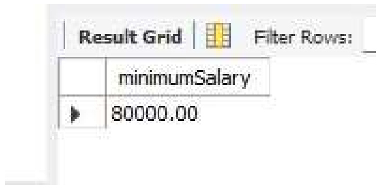
```
use labdb;
drop procedure if exists minSal;
DELIMITER //
CREATE PROCEDURE minSal(out sal decimal(10,2) )
BEGIN

SET SQL_SAFE_UPDATES = 0;

   select min(salary) into sal from employees  ;
  SET SQL_SAFE_UPDATES = 1;
```

END //

DELIMITER // CALL

minSal(@mSal); select @mSal as

minimumSalary;



```
minimumSalary
80000.00
```