Q1 a).Create a new table to track the Post_Office location. Post_Office (po_id, po_name, po_address, _city, po_state, po_pincode) po_id is the primary key and should be numeric. po_name, po_address, and po_city is between 1 and 35 characters. – These should not be null. po_state is 2 characters po_pincode is 5 numbers. Check for one of the following pin codes – 75081, 75080, 75082, 75079, 75078

b). Write insert query for the above table (Post_Office). Enter 5 rows in the table.

```
create table Post_Office(
    po_id int primary key,
    po_name varchar(35) not null,
    po_address varchar(35) not null,
    po_city varchar(35) not null,
    po_state varchar(2),
    po_pincode bigint);

-- b)
insert into Post_Office(po_id, po_name, po_address, po_city, po_state, po_pincode)
values(1,'Itwari','Itwari Bazar Road, Nagpur','Nagpur','MH',75081),
(2, 'Nagpur GPO', 'Civil Lines, Nagpur','Nagpur','MH',75080),
(3,'Malkapur','Tahasil Chowk, Malkapur','Malkapur','MP',75082),
(4,'Panji','Panji Road, Goa','Goa','GA',75079),
(5,'Deccan GPO','Deccan Road','Hyderabad','TS',75078);
select * from Post_Office;
```

Result Grid

po_id	po_name	po_address	po_city	po_state	po_pincode
1	Itwari	Itwari Bazar Road, Nagpur	Nagpur	MH	75081
2	Nagpur GPO	Civil Lines, Nagpur	Nagpur	MH	75080
3	Malkapur	Tahasil Chowk, Malkapur	Malkapur	MP	75082
4	Panji	Panji Road, Goa	Goa	GA	75079
5	Deccan GPO	Deccan Road	Hyderabad	TS	75078
NULL	NULL	NULL	NULL	NULL	NULL

Post_Office 2

c). Write a query that will display all the Post Office records of a State. Display the address of Post Office in a same city.

```
-- c)
select po_address,po_city from Post_Office where po_state = 'MH';
```

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

po_address	po_city
Itwari Bazar Road, Nagpur	Nagpur
Civil Lines, Nagpur	Nagpur

Post_Office 4

d).In which city having maximum number of post office,show the pincodes of those cities.

```
-- d)
Select po_pincode,po_city from Post_office where po_city = (SELECT max(distinct po_city)
FROM Post_Office);
```

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

po_pincode	po_city
75081	Nagpur
75080	Nagpur

Post_office 18

Q2. Create a store procedure that receives the first name of the person table as input and the last name as output.

```
create table Person(
    firstname varchar(25),
    lastname varchar(25)
    );
insert into Person( firstname, lastname)
values ("Devashish", "Patil"),
("Kshitij","Kanake"),
("Pranav", "Chavan");
select * from Person;
delimiter //
create PROCEDURE GetLastName(out nme1 varchar(50))
begin
select lastname into nme1 from Person where firstname="Devashish";
end //
delimiter ;
call GetLastName(@d);
```

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

Last_Name
Patil

Q3. Create a query to show the account number and customerid from the customer table for the customer without sales orders.

```
create table customers(
    id int primary key,
    cus_name varchar(50) not null,
    account_no bigint not null
    );
insert into customers(id, cus_name,account_no) values(1, 'Customer1',123), (2, 'Customer2',234), (3,'Customer3',345), (4, 'Customer4',456), (5,'Customer5',567),(6,'Customer6',678),
(7,'Customer7',789), (8,'Customer8',890), (9,'Customer9',901), (10,'Customer10',012), (11,'Customer11',0123), (12,'Customer12',1234), (13,'Customer13',2345);
create table orders(
    oid int,
    id int,
    sales_order bigint default null,
    foreign key (id) references customers(id));
insert into orders(oid,id, sales_order) values (101,1,45),(102,2,56), (103,3,89), (104,4,null ) , (105,5,36), (106,6,null ), (107,7,12), (108, 8, 99), (109,9,16), (110,10,87),
(111,11,65),(112,12,54),(113,13,54);
select distinct customers.account_no, customers.id from customers,orders where (orders.id = customers.id and sales_order is null);
```

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

account_no	id
456	4
678	6

Q4. Create a query to show the top 10 customerIDs of users with more Orders.

```
-- Q4
SELECT id,
       COUNT(DISTINCT sales_order)
FROM orders
GROUP BY 1
ORDER BY 2 DESC
LIMIT 10;
```

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

id	COUNT(DISTINCT sales_order)
5	3
3	2
8	2
1	1
2	1
7	1
9	1

Q5. Creating procedure without parameters

```
-- Q5
delimiter //
create procedure without_parameter()
begin
declare v varchar(25);
set v = 'hello';
select v as result;
end //
delimiter ;
call without_parameter;
```

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

result
hello

Q6..Creating Procedure with (IN/OUT/INOUT) Parameters.

```
-- Q6
delimiter //
create procedure parameter(inout ds varchar(25), in ls varchar(25))
begin
set ds = 'Student x';
set ls = 'Last name';

end //
delimiter ;
call parameter(@s,1);
select @s as Student_Name;
```

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

Student_Name
Student x

Q7.Write a MySQL stored procedure that takes an integer parameter representing a student's score. Based on the score, the procedure should return one of the following grades using IF-ELSE: i) Score>=90:”A” ii) Score>=80:”B” iii) Score>=70:”C” iv) Score>=60:”D” v)Score<60:”Fail”

```
DELIMITER //
CREATE PROCEDURE get_grade(IN score INT)
BEGIN
  DECLARE grade VARCHAR(10);
  IF score >= 90 THEN
    SET grade = 'A';
  ELSEIF score >= 80 THEN
    SET grade = 'B';
  ELSEIF score >= 70 THEN
    SET grade = 'C';
  ELSEIF score >= 60 THEN
    SET grade = 'D';
  ELSE
    SET grade = 'Fail';
  END IF;
  SELECT grade;
END //
DELIMITER ;
call get_grade(85);
```

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

grade
B

Q8. Write a MySQL stored procedure that uses a loop to iterate through a list of numbers from 1 to 20.

```
-- Q8
delimiter \\
create procedure get_number()
begin
    declare num int default 1;
    declare result varchar(100) default '';
    set num = 1;
    set result = '';
    while num <= 20 do
        set result = concat(result,num, ',');
        set num = num+1;
    end while;
    select result;
end \\
delimiter ;
call get_number();
```

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

result
1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,

Q9.Create a stored procedure named CalculateFactorial that accepts a single integer parameter, n.Inside the procedure, use a loop to calculate the factorial of n.

```
delimiter //
create procedure factorial()
begin
    declare n int default 5;
    declare m int default 1;
    declare a int;
    declare str varchar(50);
    set a = 1;
    while m<=n do
    set a = a*m;
    set m = m+1;
    end while;
    select a as result;
end //
delimiter ;
drop procedure factorial;
call factorial();
```

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

result
1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,

Q10. Create a stored procedure named GenerateFibonacciSequence that accepts a single integer parameter, n, representing the number of terms in the Fibonacci sequence.

```
create procedure GenerateFibonacciSequence(in n int,out result varchar(100))
begin
    declare num int default 0;
    declare num1 int default 1;
    declare num2 int default 0;
    declare counter int default 0;
    set result = '';
    repeat
      set  result = concat(result,num,',');
      set num2=num+num1;
      set num=num1;
      set num1=num2;
      set counter=counter+1;
      until counter>n
      end repeat;
end //
delimiter ;
call GenerateFibonacciSequence(10,@result);
select @result;
```

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

@result
0,1,1,2,3,5,8,13,21,34,55,