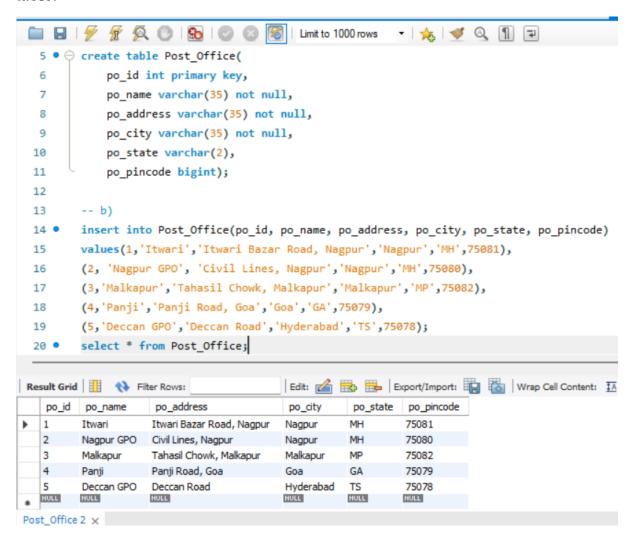
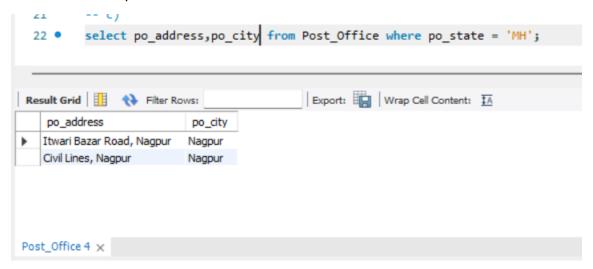
Q1 a).Create a new table to track the Post_Office location. Post_Office (po_id, po_name, po_address, _city, po_state, po_pincode) po_id is the primary key and should be numeric. po_name, po_address, and po_city is between 1 and 35 characters. – These should not be null. po_state is 2 characters po_pincode 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.



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.



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



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

```
30 • ⊝ create table Person(
            firstname varchar(25),
  31
  32
            lastname varchar(25)
  33
  34 •
       insert into Person( firstname, lastname)
        values ("Devashish", "Patil"),
  35
         ("Kshitij", "Kanake"),
  36
         ("Pranav", "Chavan");
  37
         select * from Person;
  38 •
         delimiter //
  39
         create PROCEDURE GetLastName(out nme1 varchar(50))
  40 •
  41
  42
         select lastname into nme1 from Person where firstname="Devashish";
  43
         end //
 44
         delimiter;
  45 •
         call GetLastName(@d);
                                       Export: Wrap Cell Content: IA
Last Name
Patil
```

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

```
51 • ⊖ create table customers(
         id int primary key,
53
          cus name varchar(50) not null,
54
          account_no bigint not null
56 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);
58 • ⊖ create table orders(
59
          oid int,
60
          id int.
61
         sales_order bigint default null,
62
          foreign key (id) references customers(id));
63 • 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),
64 (111,11,65),(112,12,54),(113,13,54);
65 • select distinct customers.account_no, customers.id from customers, orders where (orders.id = customers.id and sales_order is null);
Export: Wrap Cell Content: IA
                                                                                                                                                                        account_no id
 678 6
```

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

Q5. Creating procedure without parameters

```
75
        -- Q5
         delimiter //
 76
         create procedure without_parameter()
 78

    ⇒ begin

         declare v varchar(25);
 79
         set v = 'hello';
 80
         select v as result;
 81
       end //
 82
         delimiter ;
 83
         call without_parameter;
                                      Export: Wrap Cell Content: IA
Result Grid Filter Rows:
   result
▶ hello
```

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

```
-- Q6
 86
        delimiter //
        create procedure parameter(inout ds varchar(25), in ls varchar(25))
 88 •

    ⇒ begin

 89
        set ds = 'Student x';
 90
        set ls = 'Last name';
 91
 92
 93
        end //
        delimiter;
 94
 95 •
        call parameter(@s,1);
        select @s as Student_Name;
                                      Export: Wrap Cell Content: IA
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 //
 99
         CREATE PROCEDURE get_grade(IN score INT)
100 •

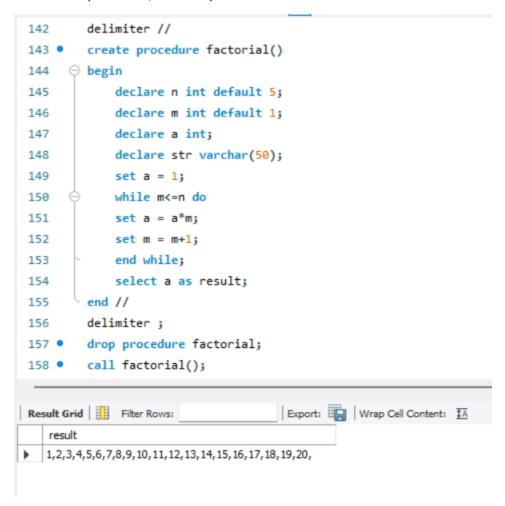
→ BEGIN

101
102
           DECLARE grade VARCHAR(10);
           IF score >= 90 THEN
103
             SET grade = 'A';
104
           ELSEIF score >= 80 THEN
105
             SET grade = 'B';
106
107
           ELSEIF score >= 70 THEN
             SET grade = 'C';
108
           ELSEIF score >= 60 THEN
109
             SET grade = 'D';
110
           ELSE
111
112
             SET grade = 'Fail';
           END IF;
113
           SELECT grade;
114
115
         END //
         DELIMITER;
116
 117 •
         call get_grade(85);
                                      Export: Wrap Cell Content: IA
Result Grid Filter Rows:
    grade
▶ B
```

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

```
123
        -- Q8
124
        delimiter \\
125 •
         create procedure get number()
      ⊖ begin
126
            declare num int default 1;
127
            declare result varchar(100) default '';
128
129
             set num = 1;
            set result = '';
130
131
            while num <= 20 do
                set result = concat(result, num, ',');
132
                set num = num+1;
133
134
            end while;
             select result;
135
       end \\
136
        delimiter;
137
138 • call get_number();
                                     Export: Wrap Cell Content: 1A
Result Grid Filter Rows:
   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.



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

```
163 •
         create procedure GenerateFibonacciSequence(in n int,out result varchar(100))
164

    ⇒ begin

165
            declare num int default 0;
            declare num1 int default 1;
166
167
            declare num2 int default 0;
            declare counter int default 0;
168
            set result = '';
169
170
            repeat
              set result = concat(result, num, ', ');
171
              set num2=num+num1;
172
173
              set num=num1;
174
              set num1=num2;
175
              set counter=counter+1;
               until counter>n
176
177
               end repeat;
        end //
178
         delimiter;
179
180 •
         call GenerateFibonacciSequence(10,@result);
         select Mresult:
 181 •
Export: Wrap Cell Content: IA
    @result
0,1,1,2,3,5,8,13,21,34,55,
```