```
create database class_3;
use class_3;

--- PATTERN MATCHING
--- Question 1: How can you select all employees whose names start with the letter 'A'?
--- Ans1
select * from employees where name like 'A%';

/* Question 2: How do you find all products whose names contain the word 'phone'
regardless of case? */
--- Ans2
select * from products where name like '%phone%';

--- Question 3: How can you retrieve all email addresses from a table that end with '.com'?
--- Ans3
select Email_Address from Table_Name where Email_Address regexp '.com$' ;

--- Question 4: How do you find all phone numbers that start with the area code '555'?
--- Ans4
select Phone_numbers from Table_Name where Phone_numbers regexp '^555';

/* Question 5: How can you select all cities that start with 'New' followed by any
characters? */
--- Ans5
select City from Table_Name where City regexp '^New';

/* Question 6: How do you find all records where the value in the 'description' column
contains either 'apple' or 'orange'? */
--- Ans6
select * from Table_Name where description regexp 'apple|orange';
```

/* Question 7:How can you retrieve all email addresses that follow the pattern of "user@domain.com"? */

--- Ans7

```
select Email_address from table_name where email_address like '_%@_%.com';
```

/* Question 8:How do you find all records where the 'product_code' is exactly four characters long and consists of letters and digits? */

--- Ans8

```
select * from table_name where product_code like '____' and
product_code not regexp  '[^A-Za-z0-9]' and
product_code regexp '[a-z][0-9]|[0-9][a-z]';
```

/* Question 9:How can you retrieve all phone numbers that match the pattern '###-###-####'? */

--- Ans9

```
select Phone_numbers from Table_name where Phone_numbers like '___-___-____' and
Phone_numbers regexp '[0-9][0-9][0-9]-[0-9][0-9][0-9]-[0-9][0-9][0-9][0-9]';
```

/* Question 10:How do you find all records where the 'text' column contains two consecutive digits? */

--- Ans10

```
select * from Table_name where text regexp '[0-9][0-9]';
```

--- NULL VALUES:

--- Question1: Find all employees whose birthdates are not recorded (NULL).

--- Ans1

```
select employees from table_name where birthdates is null;
```

--- Question2: List all orders that dont have a customer assigned (NULL customerID).

--- Ans2

```
select orders from tables_name where customer is null;
```

--- FUNCTIONS:

/* ● Consider a table named Sales with the following columns:

SaleID (integer): The unique identifier for each sale.

Product (string): The name of the product sold.

Quantity (integer): The quantity of the product sold.

Price (decimal): The price per unit of the product. */

```
create table Sales
(
        SaleID int unique,
   Product varchar(40),
   Quantity int,
   Price double
   );

insert into Sales values
(1,'Ball',20,10.00),
(2,'Bat',10,100.00),
(3,'Book',25,50.25),
(4,'Bat',100,100.00),
(5,'Ball',150,10.00),
(6,'Book',35,50.25);
```

--- Question: Find the total quantity sold for each product.

--- Ans

```
select product, sum(quantity) from Sales group by product;
```

/* Question: Calculate the total revenue generated from each product (Total Revenue =

Quantity * Price). */

--- Ans

```
select product, sum(Quantity*Price) as Total_Revenue from sales group by product;


--- Question: Determine the average price of each product.

--- Ans

select product , avg(price) as avg_price from sales group by product;


--- Question: Find the product with the highest total revenue (Quantity * Price)

--- Ans

select product, sum(Quantity*Price) as total_revenue from sales group by product order by
total_revenue desc limit 1;


--- Question: Calculate the total quantity sold across all products.

--- Ans

select sum(quantity) as total_quantity from sales;


--- Question: Determine the average price of all products.

--- Ans

select avg(price) as avg_price_of_all_products from sales;


/* ● Consider a table named Products with the following columns:

ProductID (integer): The unique identifier for each product.

ProductName (string): The name of the product.

Price (decimal): The price of the product. */


create table Products

(

        ProductID int unique,

   ProductName varchar (30),

   Price double

   );
```

```
insert into products values
(1,'Ball',10.00),
(2,'Bat',100.00),
(3,'Book',50.25),
(4,'Bat',100.00),
(5,'Ball',10.00),
(6,'Book',50.25);

--- Question: Determine the square root of the price for each product.
--- Ans
select productName , sqrt(price) as sqrt_price from products;

--- Question: Find the ceiling (smallest integer greater than or equal to) of the prices.
select productname , ceil(price) as ceil_price from products;

--- Question: Calculate the floor (largest integer less than or equal to) of the prices.
select productname , floor(price) as floor_price from products;

/* ● Consider a table named Orders with the following columns:
OrderID (integer): The unique identifier for each order.
OrderDate (datetime): The date and time when the order was placed.
DeliveryDate (datetime): The date and time when the order was delivered. */

create table orders
(
        OrderID int,
   OrderDate datetime,
   DeliveryDate datetime
   );

insert into orders values
```

(100, '2023-01-01', '2023-01-10'),

(101, '2023-01-02', '2023-01-11'),

(101, '2023-01-03', '2023-01-14'),

(102, '2023-01-05', '2023-01-20'),

(100, '2023-01-04', '2023-01-11'),

(102, '2023-01-04', '2023-01-17'),

(102, '2023-01-04', '2023-01-04 10:10:10');

/* Question:Find the difference in days between the order date and delivery date for each

order. */

--- Ans

select orderid , abs(datediff(deliverydate,orderdate)) as days_diff from orders;

--- Question:Calculate the total delivery time in hours for all orders.

--- Ans

SELECT SUM(TIMESTAMPDIFF(HOUR, OrderDate, DeliveryDate)) AS DeliveryHours FROM Orders;

--- Question:Determine the day of the week when each order was placed.

--- Ans

select orderID , day(OrderDate) as DayOfWeek from orders;

--- Question:Find the orders that were placed on a Saturday (DayOfWeek = 7).

--- Ans

select orderId from orders where day(Orderdate) = 7;

--- Question: Calculate the average delivery time in days for all orders.

--- Ans

select avg(datediff(deliverydate,orderdate)) as Avg_Dlvry_time from orders;

--- Question: Find the orders that were delivered on the same day they were placed.

--- Ans

```
select * from orders where date(orderdate) = date(deliverydate);

--- GROUP BY, HAVING, ORDER BY
/* Question 1: Consider a table named Orders with the following columns:

OrderID (integer): The unique identifier for each order.
CustomerID (integer): The unique identifier for each customer.
TotalAmount (decimal): The total amount of the order.

Write an SQL query to find the customer IDs of customers who have placed orders with
a total amount greater than $1,000. */

create table orders
(
        OrderID int,
   CustomerID int,
   TotalAmount decimal
   );
insert into orders values
(1,1,100),
(2,2,150),
(3,1,200),
(4,1,200),
(5,1,600),
(6,1,1600);

--- Ans1
select CustomerID, sum(totalamount) as sumamount from orders group by customerID having
sumamount > 1000;

/* Question 2:Consider a table named Sales with the following columns:
```

ProductID (integer): The unique identifier for each product.

SaleDate (date): The date of the sale.

QuantitySold (integer): The quantity of the product sold on that date.

Write an SQL query to find the product IDs of products that have been sold in quantities greater than 100 on at least three different sale dates. */

```
create table sales
(
        ProductID int,
   SaleDate date,
   QuantitySold int
   );
insert into sales values
(1,'2023-01-01',20),
(1,'2023-01-03',40),
(1,'2023-01-02',50),
(2,'2023-01-10',20),
(2,'2023-01-11',20),
(2,'2023-01-11',70),
(3,'2023-01-20',20),
(3,'2023-01-21',50),
(3,'2023-01-22',20),
(3,'2023-01-23',20);

--- Ans2
select productID , sum(QuantitySold)  as SumQntySold , count(distinct saledate) as datecount
from sales group by productID
having SumQntySold>100 and datecount>=3;
```

/* Question 3:Consider a table named Employees with the following columns:

EmployeeID (integer): The unique identifier for each employee.
Department (string): The department in which the employee works.
Salary (decimal): The salary of the employee.

Write an SQL query to find the average salary of employees in each department, but only for departments where the average salary is greater than $60,000. */

```
create table employee
(
        EmployeeID int,
   Department varchar(30),
   Salary decimal
   );

insert into employee values
(1,'HR', 70000),
(2,'MKT', 50000),
(3,'HR', 60000),
(4,'MKT', 50000);

--- Ans3
select department, avg(salary) as Avg_Salary from employee
group by department having Avg_salary>60000;
```

/* Question 4:Consider a table named Students with the following columns:

StudentID (integer): The unique identifier for each student.
Course (string): The course name.
Score (integer): The score obtained by the student in the course.

Write an SQL query to find the course names in which the average score of all students is greater than or equal to 80. */

```
create table if not exists students
(
        StudentID int,
   Course varchar(30),
   Score int);
insert into students values
(1,'Engg',80),
(2,'Engg',80),
(3,'ART',100),
(4,'ART',20);

--- Ans4
select course , avg(score) as avg_score from students
group by course having avg_score>=80;
```

/* Question 5: Consider a table named Employees with the following columns:

EmployeeID (integer): The unique identifier for each employee.
Department (string): The department in which the employee works.
Salary (decimal): The salary of the employee.

Write an SQL query to find the department with the highest average salary. */

```
create table if not exists employees
(
        EmployeeID int,
   Department varchar(30),
```

```
   Salary decimal
   );
 insert into employees values
(1,'HR', 70000),
(2,'MKT', 50000),
(3,'HR', 60000),
(4,'MKT', 50000);

--- Ans5
select department, avg(salary) as avg_salary from employees
group by department order by avg_salary desc limit 1;

/* Question 6: Consider a table named Sales with the following columns:

ProductID (integer): The unique identifier for each product.
SaleDate (date): The date of the sale.
QuantitySold (integer): The quantity of the product sold on that date.

Write an SQL query to find the product with the highest total quantity sold. */
--- Ans6
select productID, sum(QuantitySold) as Total_soldQnty from sales
group by productID order by Total_soldQnty desc limit 1;

/* Question 7: Consider a table named Students with the following columns:

StudentID (integer): The unique identifier for each student.
Course (string): The course name.
Score (integer): The score obtained by the student in the course.

Write an SQL query to find the top three students with the highest average score across
all courses */
```

--- Ans 7

```
select studentID, avg(score) as avg_score from students
group by studentID order by avg_score desc limit 3;
```

/* Question 8:Consider a table named Orders with the following columns:

OrderID (integer): The unique identifier for each order.
CustomerID (integer): The unique identifier for each customer.
OrderDate (date): The date of the order.
TotalAmount (decimal): The total amount of the order.

Write an SQL query to find the total amount of orders placed by each customer, ordered in descending order of total amount. */

--- Ans 8

```
select customerID, sum(totalamount) as total_amountorders from orders
group by customerID order by total_amountorders desc;
```

/* Question 9: Consider a table named Books with the following columns:

BookID (integer): The unique identifier for each book.
Author (string): The author of the book.
PublicationYear (integer): The year the book was published.

Write an SQL query to find the number of books published by each author in descending order of the count. */

```
create table books
(
        BookID int,
   Author varchar (10),
   PublicationYear int);
insert into books values
```

```
(1,'Hi',2020),
(2,'Hi',2021),
(3,'Bi',2022);
--- Ans9
select author, count(bookID) as books_published from books
group by author order by books_published desc;

/* Question 10: Consider a table named Orders with the following columns:

OrderID (integer): The unique identifier for each order.
CustomerID (integer): The unique identifier for each customer.
OrderDate (date): The date of the order.
TotalAmount (decimal): The total amount of the order.

Write an SQL query to find the customer IDs of customers who have placed orders with
a total amount greater than $1,000 and have placed at least two orders. */
--- Ans10
select customerid, sum(totalamount) as sumOfTotalAmount, count(orderID) as NoOfOrders
from orders group by customerid having sumOfTotalAmount>1000 and NoOfOrders>=2;

/* Question 11: Consider a table named Products with the following columns:

ProductID (integer): The unique identifier for each product.
Category (string): The category of the product.
Price (decimal): The price of the product.

Write an SQL query to find the average price of products in each category, ordered by
category name in ascending order. */
--- Ans11
select category, avg(price) as avg_price from products
group by category order by category;
```

/* Question 12: Consider a table named Employees with the following columns:

EmployeeID (integer): The unique identifier for each employee.
Department (string): The department in which the employee works.
Salary (decimal): The salary of the employee.

Write an SQL query to find the department(s) with the lowest average salary. */

--- Ans12

```
select department, avg(salary) as avg_salary from employees
group by department order by avg_salary limit 1;
```

/* Question 13: Consider a table named Orders with the following columns:

OrderID (integer): The unique identifier for each order.
CustomerID (integer): The unique identifier for each customer.
OrderDate (date): The date of the order.
TotalAmount (decimal): The total amount of the order.

Write an SQL query to find the customer IDs of customers who have placed orders with a total amount greater than $500 in the year 2023, ordered by customer ID in ascending order. */

```
select customerid, sum(totalamount) as sum_amount from orders
where year(orderdate) = 2023
group by customerid
having sum_amount>500
order by customerid;
```

/* Question 14: Consider a table named Students with the following columns:

StudentID (integer): The unique identifier for each student.

Course (string): The course name.

Score (integer): The score obtained by the student in the course.

Write an SQL query to find the course names in which the highest score achieved by any student is greater than or equal to 90, ordered by course name in ascending order. */

--- Ans14

```
select course , score from students where score >= 90 order by course asc;
```

/* Question 15: Consider a table named Orders with the following columns:

OrderID (integer): The unique identifier for each order.

CustomerID (integer): The unique identifier for each customer.

OrderDate (date): The date of the order.

TotalAmount (decimal): The total amount of the order.

Write an SQL query to find the customer IDs of customers who have placed orders with a total amount greater than $500 in the year 2023 and have placed at least two orders in that year. */

--- Ans 15

```
select customerid from orders where year(orderdate) = 2023
group by customerid having sum(totalamount)>500
and count(orderid)>=2;
```

/* Question 16: Consider a table named Students with the following columns:

StudentID (integer): The unique identifier for each student.

Course (string): The course name.

Score (integer): The score obtained by the student in the course.

Write an SQL query to find the course names where the average score of students who scored less than 70 in at least one course is greater than or equal to 80. */

```
--- Ans16
select course, avg(score) from students group by course
having avg(score) >= 80 and
count(case
        when score<70
                        then 1
        else
                null
                        end) >=1;

/* Question 17:Consider a table named Employees with the following columns:

EmployeeID (integer): The unique identifier for each employee.
Department (string): The department in which the employee works.
Salary (decimal): The salary of the employee.

Write an SQL query to find the departments where the highest salary is greater than
$80,000 and the total number of employees in that department is at least 5. */
--- Ans17
select department from employees
group by department
having max(salary)> 80000 and count(employeeid)>=5;

/* Question 18: Consider a table named Products with the following columns:

ProductID (integer): The unique identifier for each product.
Category (string): The category of the product.
Price (decimal): The price of the product.

Write an SQL query to find the categories where the average price of products is
greater than or equal to $50, and the maximum price within that category is greater than
```

$100. */

--- Ans18

```
select category from products
group by category having avg(price)>=50 and max(price) >100;
```

/* Question 19: Consider a table named Orders with the following columns:

OrderID (integer): The unique identifier for each order.

CustomerID (integer): The unique identifier for each customer.

OrderDate (date): The date of the order.

TotalAmount (decimal): The total amount of the order.

Write an SQL query to find the customer IDs of customers who have placed orders with a total amount greater than $1,000 in any single order and have placed orders on at least three different dates. */

--- Ans19

```
select customerID, max(totalamount) from orders
group by customerid
having count(distinct orderdate)>=3
and max(totalamount)>1000;
```