EXPERIMENT NO: 1

AIM:

1. What is DBMS? Explain advantages of DBMS over FPS.

2. List 15 applications of Database. Explain any 2 how Database can be helpful in managing that application?

3. Create the Database for the following:

- **Student Details using Excel.**
- **Employees Details using MS Access**
- **Facebook using Excel**

Introduction/Purpose:

1. DBMS:

Database Management System (DBMS) is a software designed to define, manipulate, retrieve and manage data in a database.

2. Application of DBMS:-

1. Broadcast communications
2. Banking & Insurance
3. Retail Management system
4. Reporting and analysis
5. Railways
6. Universities and Colleges/Schools
7. Credit card exchanges
8. Library Management System
9. Human Resource Department
10. Hospitals store
11. Medical Stores
12. Government Organizations
13. Supply Change Management
14. Manufacturing and Production Control
15. Account
15. E-Commerce (Flipkart, Amazon, eBay etc...)

3. Creating an Employee Details Database using MS Access:

1. Launch Microsoft Access.

2. Choose a database template or create a new blank database.

3. Name your database.

4. Design the Employee table with fields like Employee ID, First Name, Last Name, Date of Birth, Department, Position, Salary, Email, and Phone.

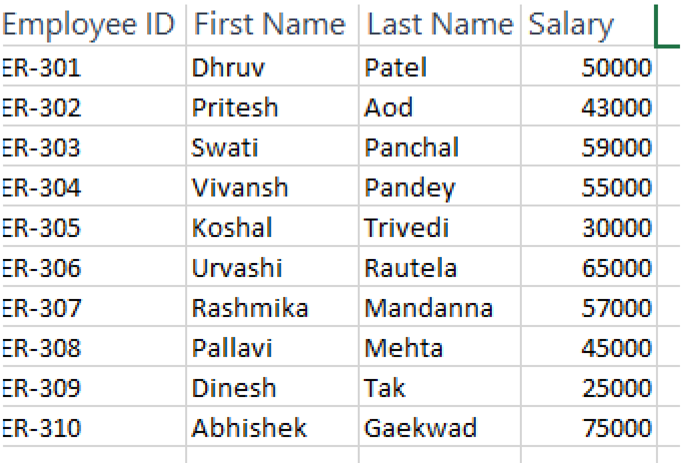
5. Enter employee data into the table.

6. Create additional tables if needed.

7. Define relationships between tables if applicable.

8.Save and backup your database.

OUTPUT:



Employee ID	First Name	Last Name	Salary
ER-301	Dhruv	Patel	50000
ER-302	Pritesh	Aod	43000
ER-303	Swati	Panchal	59000
ER-304	Vivansh	Pandey	55000
ER-305	Koshal	Trivedi	30000
ER-306	Urvashi	Rautela	65000
ER-307	Rashmika	Mandanna	57000
ER-308	Pallavi	Mehta	45000
ER-309	Dinesh	Tak	25000
ER-310	Abhishek	Gaekwad	75000

CONCLUSION:

Here we have studied that,

• Student Details using Excel.

- **Employees Details using MS Access**
- **Facebook using**

Excel done successfully.

EXPERIMENT NO: 2

AIM:-
Create an employee table to apply lowercase & uppercase.

Introduction/Purpose:

SQL UPPER Function:

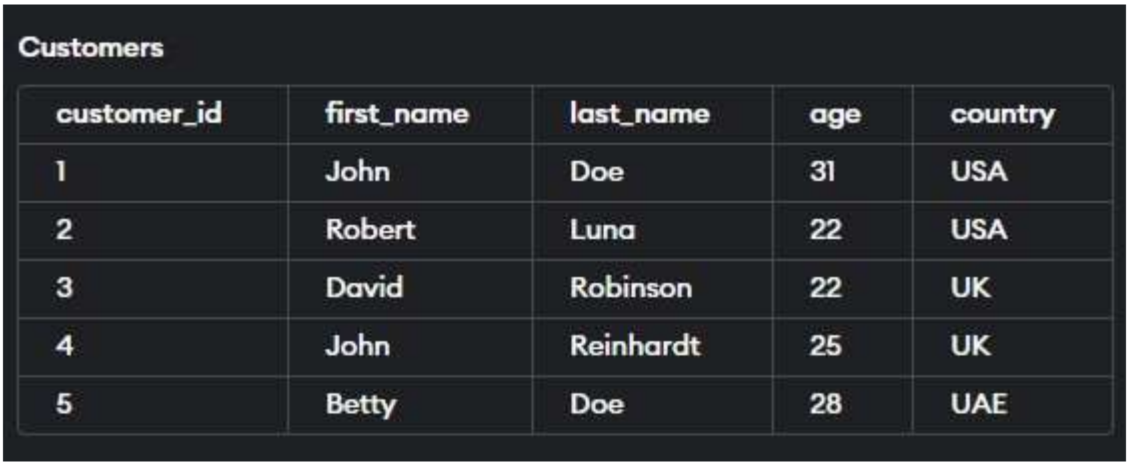
We use SQL UPPER function to convert the character in the expression into uppercase. It converts all the characters into capital letters.

Syntax:

SELECT UPPER (expression) FROM [Source Data]

Example:

Here we have customer table,

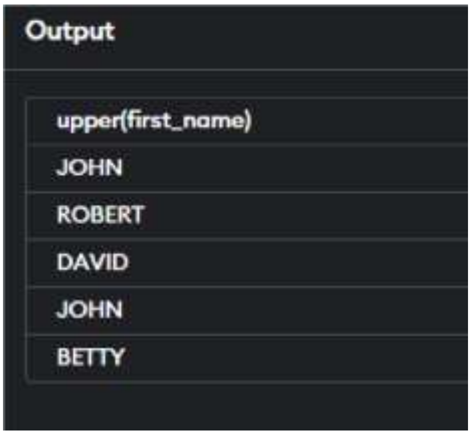


Customers

customer_id	first_name	last_name	age	country
1	John	Doe	31	USA
2	Robert	Luna	22	USA
3	David	Robinson	22	UK
4	John	Reinhardt	25	UK
5	Betty	Doe	28	UAE

Query code:

Select upper (first name) from Customers;
Output:



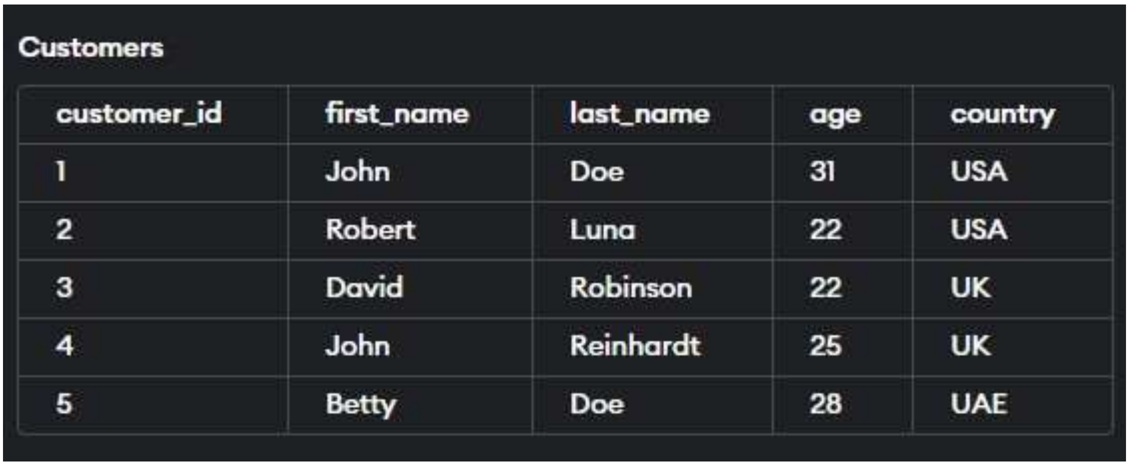
Output

upper(first_name)
JOHN
ROBERT
DAVID
JOHN
BETTY

Syntax:

SELECT UPPER (expression) FROM [Source Data]

Example: Here we have customer table,



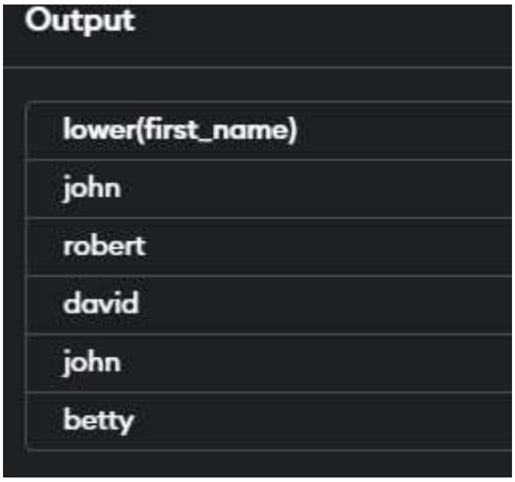
Customers

customer_id	first_name	last_name	age	country
1	John	Doe	31	USA
2	Robert	Luna	22	USA
3	David	Robinson	22	UK
4	John	Reinhardt	25	UK
5	Betty	Doe	28	UAE

Query code:

Select lower (first name) from Customers;

Output:



```
Output

lower(first_name)
john
robert
david
john
betty
```

CONCLUSION:-

Here we have studied that,

Second Practical Aim Create an employee table to apply lowercase & uppercase has done successfully.

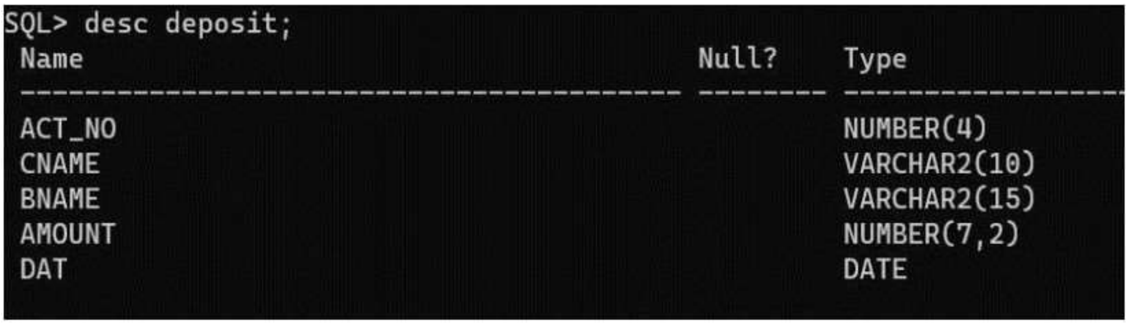
EXPERIMENT NO: 3

AIM:-

Simple Queries:

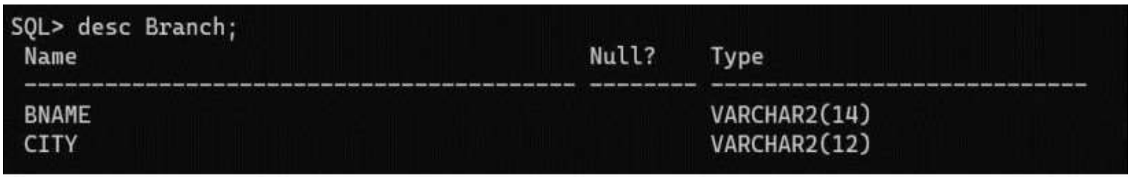
1. Describe deposit, branch.
2. Describe borrow, customers.
3. List all data from table DEPOSIT.
4. List all data from table BORROW.
5. List all data from table CUSTOMERS.
6. List all data from table BRANCH.
7. Give account no and amount of depositors.
8. List all data from SAILORS. 9. List Boat Name and its colour.
10. List Employee name and its city.
11. List all the details of Clients.
12. Describe various products and its price.
13. Describe the sailor's name, age and its rating. 14. Describe the managers of various employees
15. Describe the details of Loan for customers.
16. Describe the date of travel of various sailors.
1. Describe deposit, branch.

Query: desk deposit;



```
SQL> desc deposit;
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 ACT_NO                                             NUMBER(4)
 CNAME                                              VARCHAR2(10)
 BNAME                                              VARCHAR2(15)
 AMOUNT                                             NUMBER(7,2)
 DAT                                                DATE
```

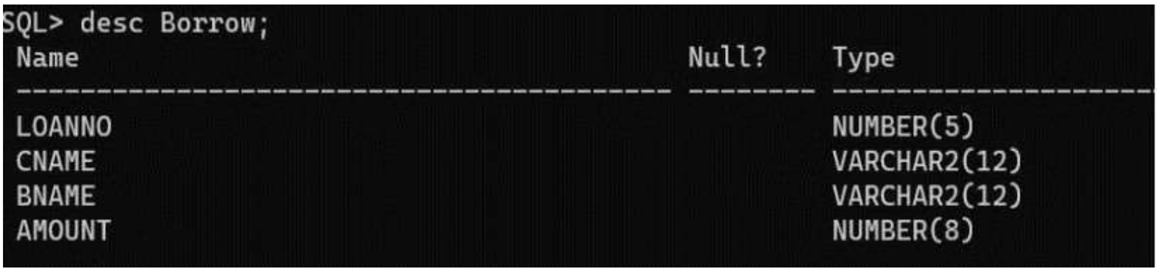
Query:desc Branch;



```
SQL> desc Branch;
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 BNAME                                              VARCHAR2(14)
 CITY                                               VARCHAR2(12)
```

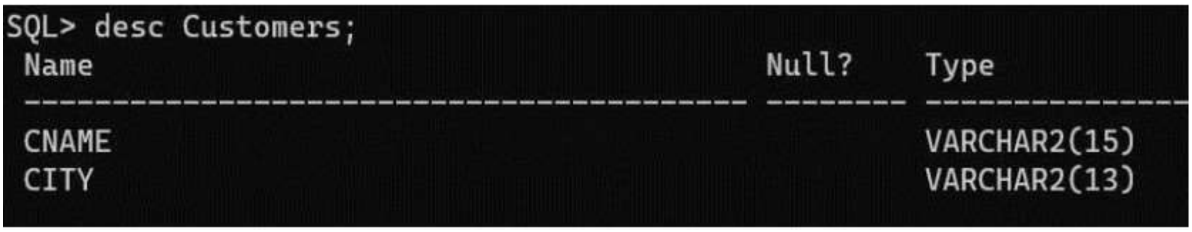
2. **Describe Borrow,Customers.**

Query:desc Borrow;



```
SQL> desc Borrow;
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 LOANNO                                             NUMBER(5)
 CNAME                                              VARCHAR2(12)
 BNAME                                              VARCHAR2(12)
 AMOUNT                                             NUMBER(8)
```

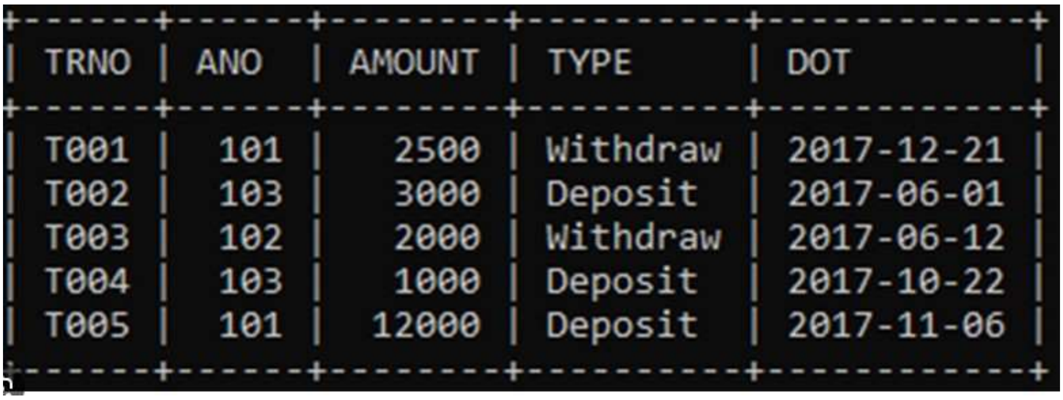
Query: desc Customers;



```
SQL> desc Customers;
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 CNAME                                              VARCHAR2(15)
 CITY                                               VARCHAR2(13)
```

3. **List all data of table Deposit.**

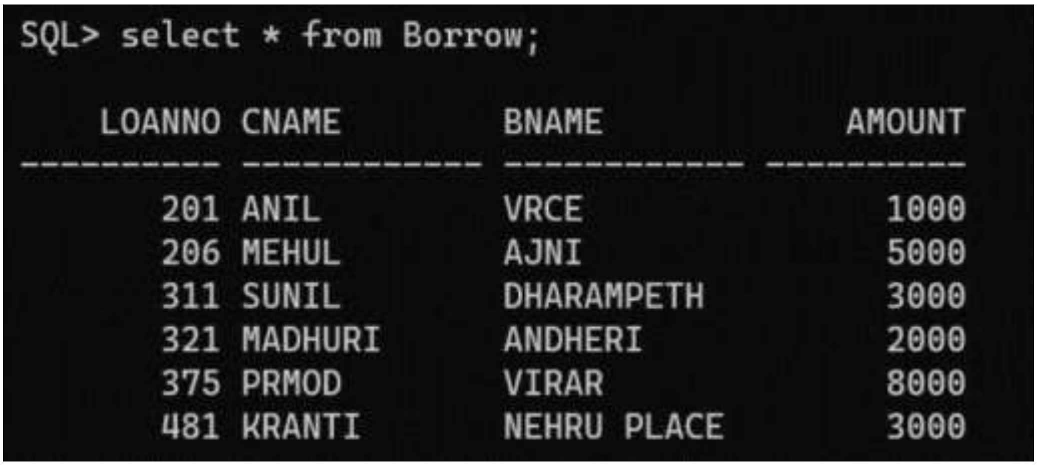
Query:select * from Deposit;



```
+------+------+--------+----------+------------+
| TRNO | ANO  | AMOUNT | TYPE     | DOT        |
+------+------+--------+----------+------------+
| T001 |  101 |   2500 | Withdraw | 2017-12-21 |
| T002 |  103 |   3000 | Deposit  | 2017-06-01 |
| T003 |  102 |   2000 | Withdraw | 2017-06-12 |
| T004 |  103 |   1000 | Deposit  | 2017-10-22 |
| T005 |  101 |  12000 | Deposit  | 2017-11-06 |
+------+------+--------+----------+------------+
```

4. **List all data from table Borrow.**

Query: select * from Borrow;

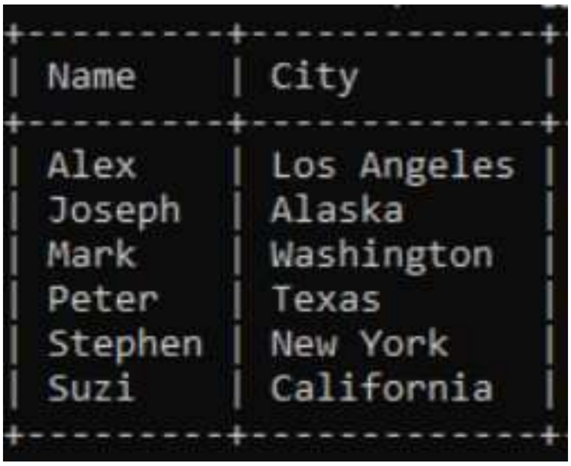


```
SQL> select * from Borrow;

    LOANNO CNAME        BNAME            AMOUNT
---------- ------------ ------------ ----------
       201 ANIL         VRCE               1000
       206 MEHUL        AJNI               5000
       311 SUNIL        DHARAMPETH         3000
       321 MADHURI      ANDHERI            2000
       375 PRMOD        VIRAR              8000
       481 KRANTI       NEHRU PLACE        3000
```

5. List all data from table Customers.

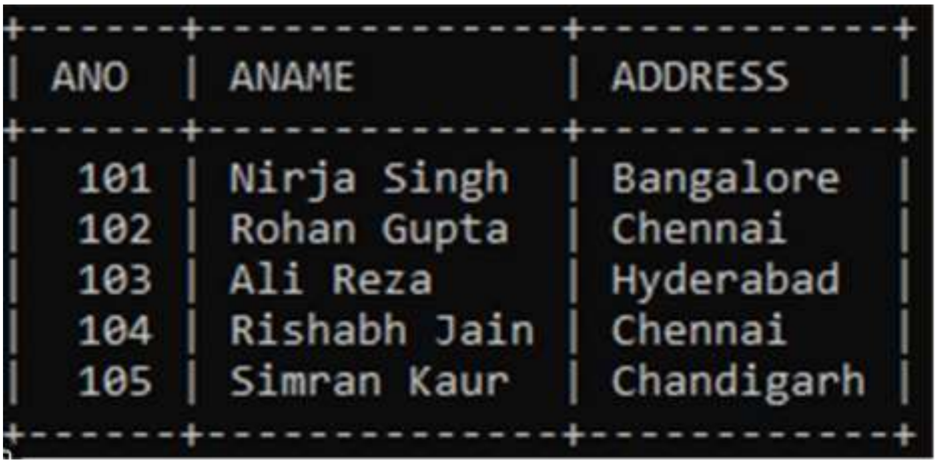
Query:select * from customers;



```
+---------+------------+
| Name    | City       |
+---------+------------+
| Alex    | Los Angeles|
| Joseph  | Alaska     |
| Mark    | Washington |
| Peter   | Texas      |
| Stephen | New York   |
| Suzi    | California |
+---------+------------+
```

6. List all data from table Branch.

Query:select * from Branch;

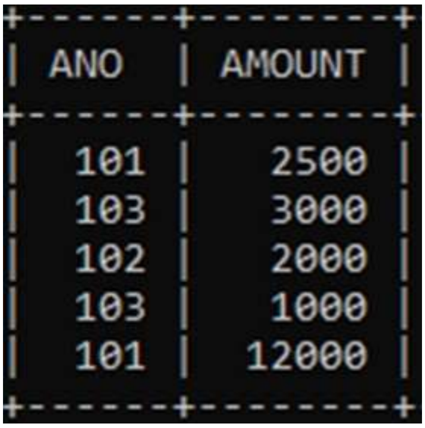


```
+-----+--------------+------------+
| ANO | ANAME        | ADDRESS    |
+-----+--------------+------------+
|  101 | Nirja Singh  | Bangalore  |
|  102 | Rohan Gupta  | Chennai    |
|  103 | Ali Reza     | Hyderabad  |
|  104 | Rishabh Jain | Chennai    |
|  105 | Simran Kaur  | Chandigarh |
+-----+--------------+------------+
```

7. Give Account no. and Amount of Depositor.

Query:select ANO,Amount from Deposit;

Output:-



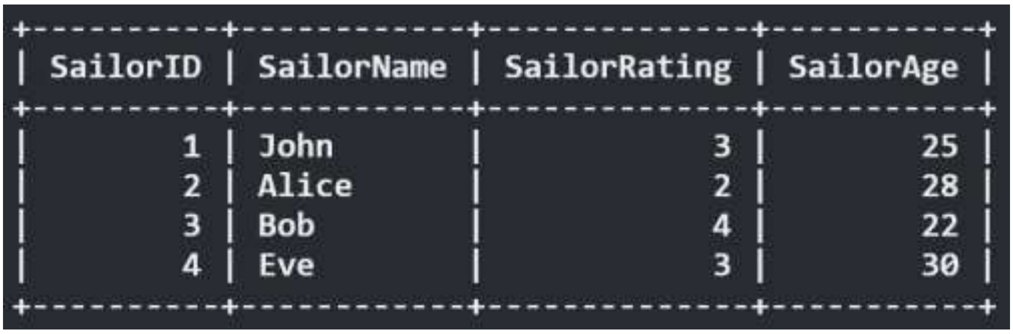
```
+------+--------+
| ANO  | AMOUNT |
+------+--------+
|  101 |   2500 |
|  103 |   3000 |
|  102 |   2000 |
|  103 |   1000 |
|  101 |  12000 |
+------+--------+
```

8. List all data from Sailors.

Query:--

```
CREATE TABLE Sailors (
SailorID INT PRIMARY KEY,
SailorName VARCHAR(255),
SailorRating INT,
SailorAge INT
);
INSERT INTO Sailors (SailorID, SailorName, SailorRating, SailorAge)
VALUES
(1, 'John', 3, 25),
(2, 'Alice', 2, 28),
(3, 'Bob', 4, 22),
(4, 'Eve', 3, 30);
SELECT * FROM Sailors;
```

Output:-



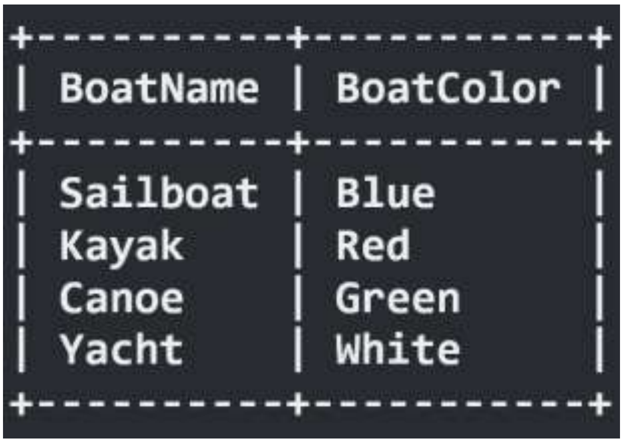
```
+----------+------------+--------------+-----------+
| SailorID | SailorName | SailorRating | SailorAge |
+----------+------------+--------------+-----------+
|        1 | John       |            3 |        25 |
|        2 | Alice      |            2 |        28 |
|        3 | Bob        |            4 |        22 |
|        4 | Eve        |            3 |        30 |
+----------+------------+--------------+-----------+
```

9. List Boat name and its color.

Query:-

```
CREATE TABLE Boats (
BoatID INT PRIMARY KEY,
BoatName VARCHAR(255),
BoatColor VARCHAR(255)
);
INSERT INTO Boats (BoatID, BoatName, BoatColor)
VALUES
(1, 'Sailboat', 'Blue'),
(2, 'Kayak', 'Red'),
(3, 'Canoe', 'Green'),
(4, 'Yacht', 'White');
SELECT BoatName, BoatColor FROM Boats;
```

Output:-



```
+----------+-----------+
| BoatName | BoatColor |
+----------+-----------+
| Sailboat | Blue      |
| Kayak    | Red       |
| Canoe    | Green     |
| Yacht    | White     |
+----------+-----------+
```

10. List Employee name and its city.

Query:-

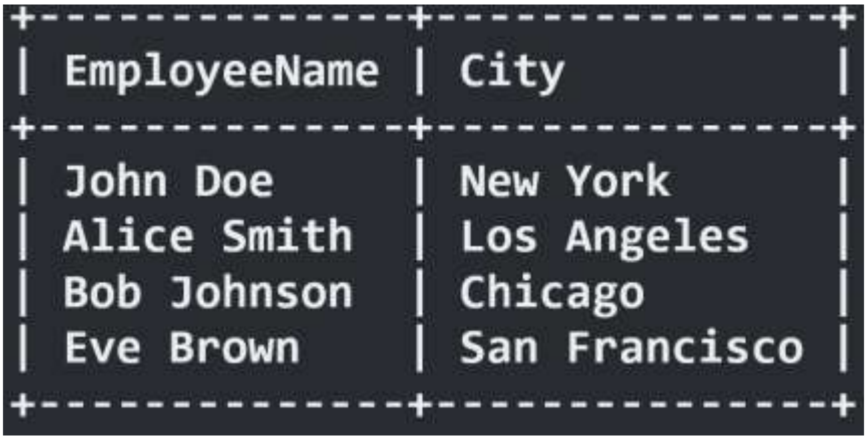
```
CREATE TABLE Employees (

EmployeeID INT PRIMARY KEY,

EmployeeName VARCHAR(255),
City VARCHAR(255)
);
INSERT INTO Employees (EmployeeID, EmployeeName, City)
VALUES
(1, 'John Doe', 'New York'),
(2, 'Alice Smith', 'Los Angeles'),
(3, 'Bob Johnson', 'Chicago'),
(4, 'Eve Brown', 'San Francisco');
SELECT EmployeeName, City FROM Employees;
```

Output:-

Output:-



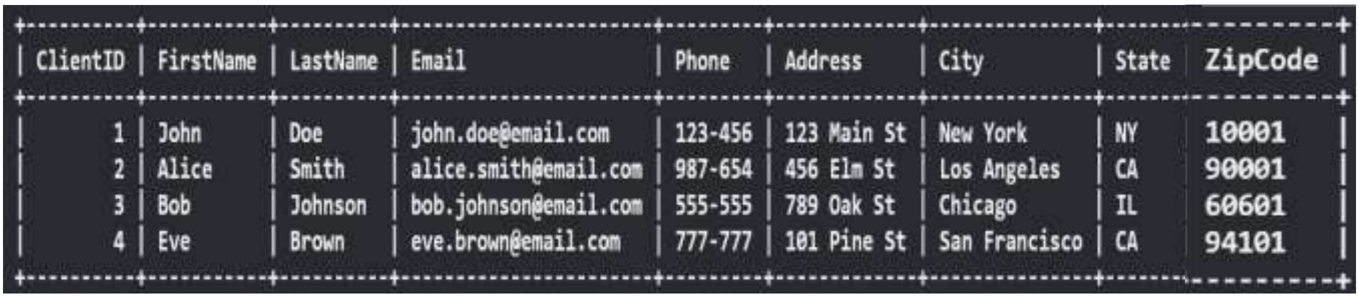
```
+--------------+---------------+
| EmployeeName | City          |
+--------------+---------------+
| John Doe     | New York      |
| Alice Smith  | Los Angeles   |
| Bob Johnson  | Chicago       |
| Eve Brown    | San Francisco |
+--------------+---------------+
```

11. List all the details of Clients.

Query:-

```
CREATE TABLE Clients (
ClientID INT PRIMARY KEY,
FirstName VARCHAR(255),
LastName VARCHAR(255),
Email VARCHAR(255),
Phone VARCHAR(15),
Address VARCHAR(255),
City VARCHAR(255),
State VARCHAR(255),
ZipCode VARCHAR(10)
);
INSERT INTO Clients (ClientID, FirstName, LastName, Email, Phone, Address, City,
State, ZipCode)
VALUES
(1, 'John', 'Doe', 'john.doe@email.com', '123-456', '123 Main St', 'New York', 'NY',
'10001'),
(2, 'Alice', 'Smith', 'alice.smith@email.com', '987-654', '456 Elm St', 'Los Angeles', 'CA',
'90001'),
(3, 'Bob', 'Johnson', 'bob.johnson@email.com', '555-555', '789 Oak St', 'Chicago', 'IL',
'60601'),
(4, 'Eve', 'Brown', 'eve.brown@email.com', '777-777', '101 Pine St', 'San Francisco', 'CA',
'94101');
SELECT * FROM Clients;
```

Output:-



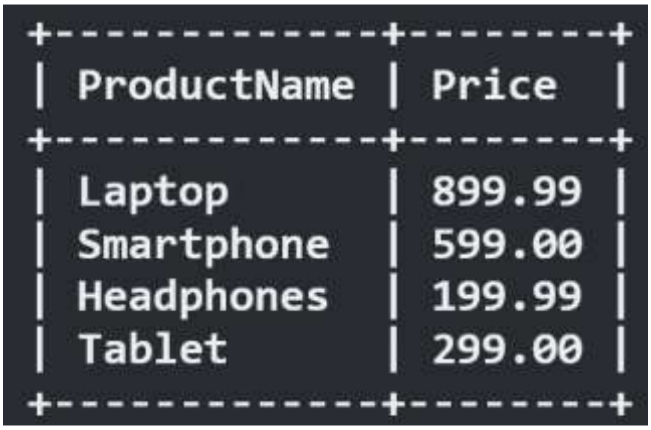
```
+----------+-----------+----------+-------------------------+---------+-------------+---------------+-------+---------+
| ClientID | FirstName | LastName | Email                   | Phone   | Address     | City          | State  ZipCode  |
+----------+-----------+----------+-------------------------+---------+-------------+---------------+-------+---------+
|        1 | John      | Doe      | john.doe@email.com      | 123-456 | 123 Main St | New York      | NY     10001    |
|        2 | Alice     | Smith    | alice.smith@email.com   | 987-654 | 456 Elm St  | Los Angeles   | CA     90001    |
|        3 | Bob       | Johnson  | bob.johnson@email.com   | 555-555 | 789 Oak St  | Chicago       | IL     60601    |
|        4 | Eve       | Brown    | eve.brown@email.com     | 777-777 | 101 Pine St | San Francisco | CA     94101    |
+----------+-----------+----------+-------------------------+---------+-------------+---------------+-------+---------+
```

12. Describe various products and its price.

Query:-

CREATE TABLE Products (ProductID
INT PRIMARY KEY,
ProductName VARCHAR(255),
Description TEXT,
Price DECIMAL(10, 2) -- Price with 2 decimal places
);
INSERT INTO Products (ProductID, ProductName, Description, Price)
VALUES
(1, 'Laptop', 'High-performance laptop with 16GB RAM', 899.99),
(2, 'Smartphone', 'Latest smartphone with 128GB storage', 599.00),
(3, 'Headphones', 'Wireless noise-canceling headphones', 199.99),
(4, 'Tablet', '10-inch tablet with HD display', 299.00); SELECT
ProductName, Price FROM Products;

Output:-



```
+-------------+--------+
| ProductName | Price  |
+-------------+--------+
| Laptop      | 899.99 |
| Smartphone  | 599.00 |
| Headphones  | 199.99 |
| Tablet      | 299.00 |
+-------------+--------+
```

13. Describe the sailor's name, age and its rating.

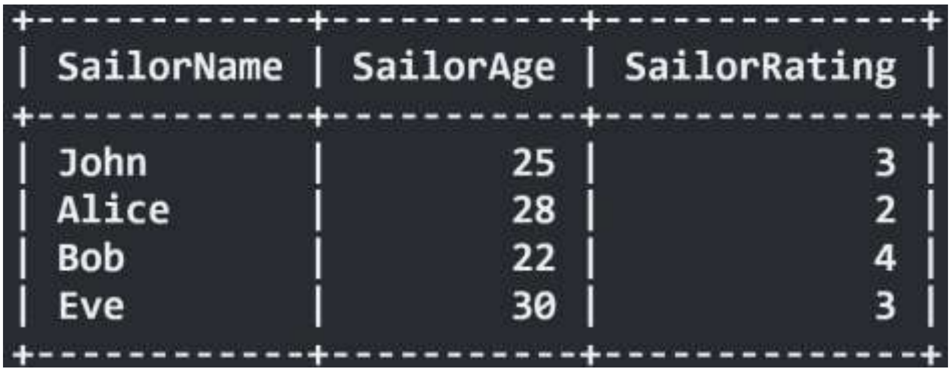
Query:-

CREATE TABLE Sailors (
SailorID INT PRIMARY KEY,
SailorName VARCHAR(255),
SailorAge INT,

Output:-

```
SailorRating INT
);
INSERT INTO Sailors (SailorID, SailorName, SailorAge, SailorRating)
VALUES
(1, 'John', 25, 3),
(2, 'Alice', 28, 2),
(3, 'Bob', 22, 4),
(4, 'Eve', 30, 3);
SELECT SailorName, SailorAge, SailorRating FROM Sailors;
```

Output:-



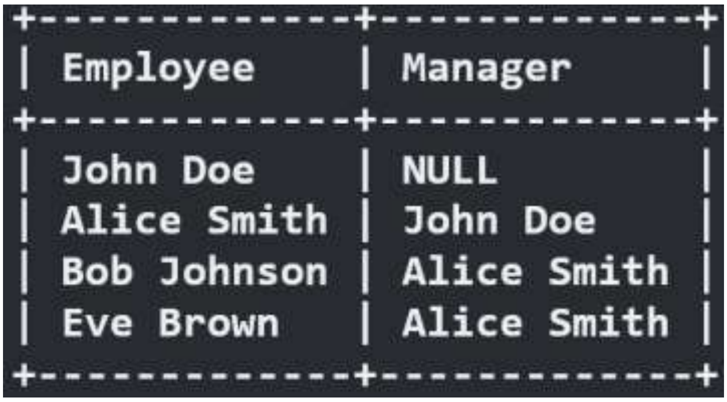
```
+------------+-----------+--------------+
| SailorName | SailorAge | SailorRating |
+------------+-----------+--------------+
| John       |        25 |            3 |
| Alice      |        28 |            2 |
| Bob        |        22 |            4 |
| Eve        |        30 |            3 |
+------------+-----------+--------------+
```

14. Describe the managers of various employees

Query:-

```
        CREATE TABLE Employees ( EmployeeID
INT PRIMARY KEY,
        EmployeeName VARCHAR(255),
        ManagerID INT,
        Position VARCHAR(255)
        );
        INSERT INTO Employees (EmployeeID, EmployeeName, ManagerID, Position)
        VALUES
        (1, 'John Doe', NULL, 'CEO'),
        (2, 'Alice Smith', 1, 'Manager'),
        (3, 'Bob Johnson', 2, 'Team Lead'),
        (4, 'Eve Brown', 2, 'Team Lead');
        SELECT EmployeeName AS Employee,
        (SELECT EmployeeName FROM Employees AS Manager WHERE
        Manager.EmployeeID = E.ManagerID) AS Manager
        FROM Employees AS E;
```

Output:-



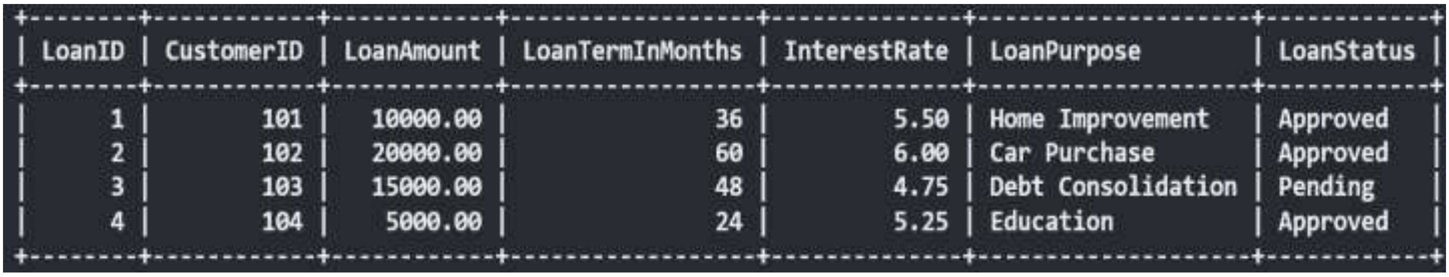
```
+-------------+-------------+
| Employee    | Manager     |
+-------------+-------------+
| John Doe    | NULL        |
| Alice Smith | John Doe    |
| Bob Johnson | Alice Smith |
| Eve Brown   | Alice Smith |
+-------------+-------------+
```

15. Describe the details of Loan for customers.

Query:-

```
CREATE TABLE Loans (
LoanID INT PRIMARY KEY,
CustomerID INT,
LoanAmount DECIMAL(10, 2),
LoanTermInMonths INT,
InterestRate DECIMAL(5, 2),
LoanPurpose VARCHAR(255),
LoanStatus VARCHAR(50)
);
INSERT INTO Loans (LoanID, CustomerID, LoanAmount, LoanTermInMonths,
InterestRate, LoanPurpose, LoanStatus)
VALUES
(1, 101, 10000.00, 36, 5.5, 'Home Improvement', 'Approved'),
(2, 102, 20000.00, 60, 6.0, 'Car Purchase', 'Approved'),
(3, 103, 15000.00, 48, 4.75, 'Debt Consolidation', 'Pending'),
(4, 104, 5000.00, 24, 5.25, 'Education', 'Approved');
SELECT * FROM Loans;
```

Output:-



```
+--------+------------+------------+------------------+--------------+--------------------+------------+
| LoanID | CustomerID | LoanAmount | LoanTermInMonths | InterestRate | LoanPurpose        | LoanStatus |
+--------+------------+------------+------------------+--------------+--------------------+------------+
|      1 |        101 |   10000.00 |               36 |         5.50 | Home Improvement   | Approved   |
|      2 |        102 |   20000.00 |               60 |         6.00 | Car Purchase       | Approved   |
|      3 |        103 |   15000.00 |               48 |         4.75 | Debt Consolidation | Pending    |
|      4 |        104 |    5000.00 |               24 |         5.25 | Education          | Approved   |
+--------+------------+------------+------------------+--------------+--------------------+------------+
```

16. Describe the date of travel of various sailors.

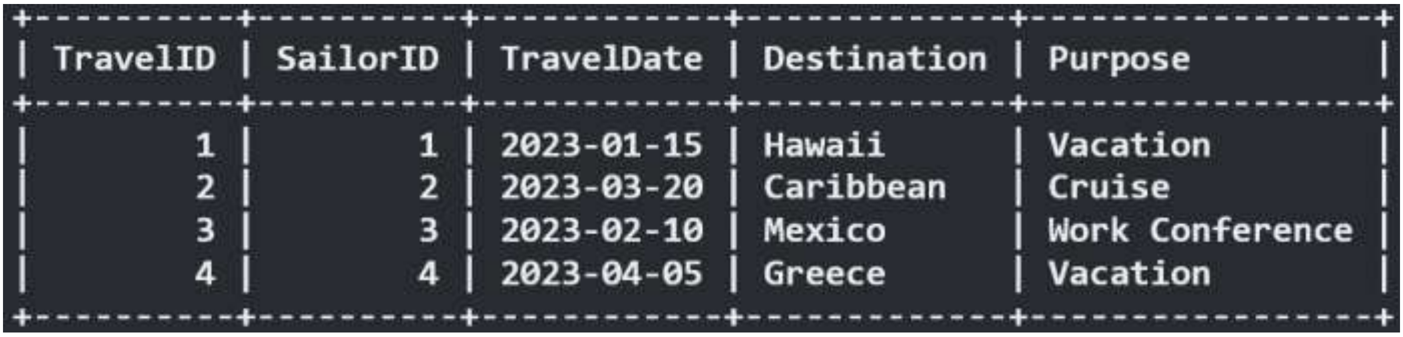
Query:-

```
CREATE TABLE TravelDates (
```

Output:-

```
TravelID INT PRIMARY KEY,
SailorID INT,
TravelDate DATE,
Destination VARCHAR(255),
Purpose VARCHAR(255)
);
    INSERT INTO TravelDates (TravelID, SailorID, TravelDate, Destination, Purpose)
VALUES
(1, 1, '2023-01-15', 'Hawaii', 'Vacation'),
(2, 2, '2023-03-20', 'Caribbean', 'Cruise'),
(3, 3, '2023-02-10', 'Mexico', 'Work Conference'),
(4, 4, '2023-04-05', 'Greece', 'Vacation');
SELECT * FROM TravelDates;
```

Output:-



```
+----------+----------+------------+-------------+-----------------+
| TravelID | SailorID | TravelDate | Destination | Purpose         |
+----------+----------+------------+-------------+-----------------+
|        1 |        1 | 2023-01-15 | Hawaii      | Vacation        |
|        2 |        2 | 2023-03-20 | Caribbean   | Cruise          |
|        3 |        3 | 2023-02-10 | Mexico      | Work Conference |
|        4 |        4 | 2023-04-05 | Greece      | Vacation        |
+----------+----------+------------+-------------+-----------------+
```

EXPERIMENT NO: 4

AIM:-

Simple Queries:-

1. Give names of depositors having amounts greater than 4000.

2. List the employees having a salary less than 22000.

3. List the sailors having age more than 25. 4. List the boats traveling on 10-oct-98

5. List the details of boat Interlake.

6. List the details of the red colored boat.

7. List the details of clients whose city is Mumbai.

8. List Client Name, due balance and city of the clients having balance greater than 1500.

9. Describe the details of products having selling price less than 500.

10. List the products for which quantity ordered is less than 120 and cost price is greater than 250.

11. Display account details having an amount greater than 2200.

12. Display all the customers staying in Nagpur.

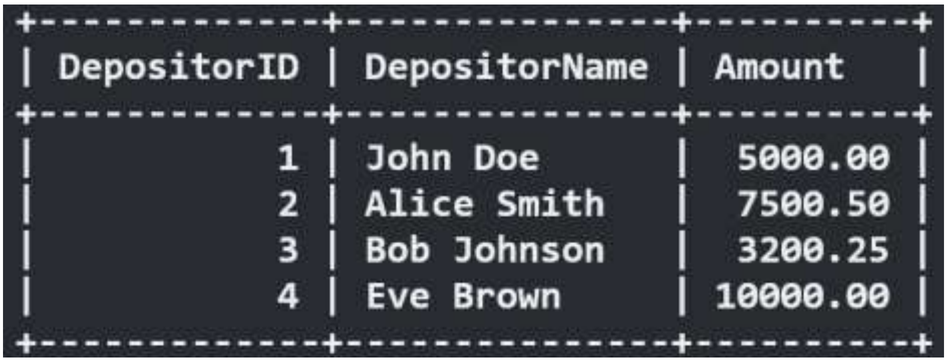
13. Display the names of sailors having ratings greater than 7.

14. Display the orders made in the month of June.

15. List all the accounts created in the month of March

1. Give names of depositors having amounts greater than 4000.

Given table:-



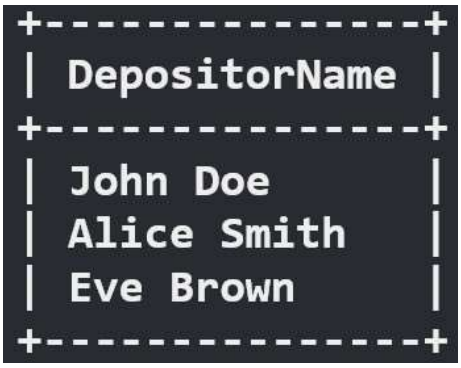
```
+-------------+---------------+----------+
| DepositorID | DepositorName | Amount   |
+-------------+---------------+----------+
|           1 | John Doe      |  5000.00 |
|           2 | Alice Smith   |  7500.50 |
|           3 | Bob Johnson   |  3200.25 |
|           4 | Eve Brown     | 10000.00 |
+-------------+---------------+----------+
```

Query:-

```
SELECT depositor_name
FROM depositors
WHERE amount > 4000;
```

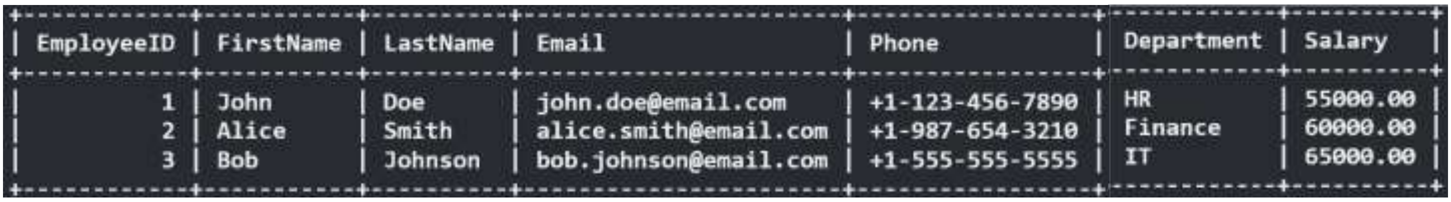
Output:-

Output:-



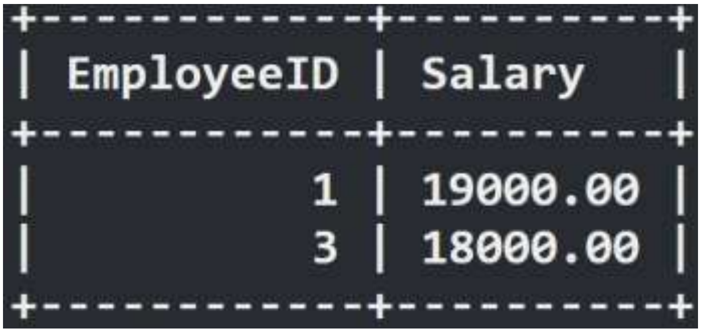
```
+---------------+
| DepositorName |
+---------------+
| John Doe      |
| Alice Smith   |
| Eve Brown     |
+---------------+
```

2. List the employees having a salary less than 22000.
Given table:-



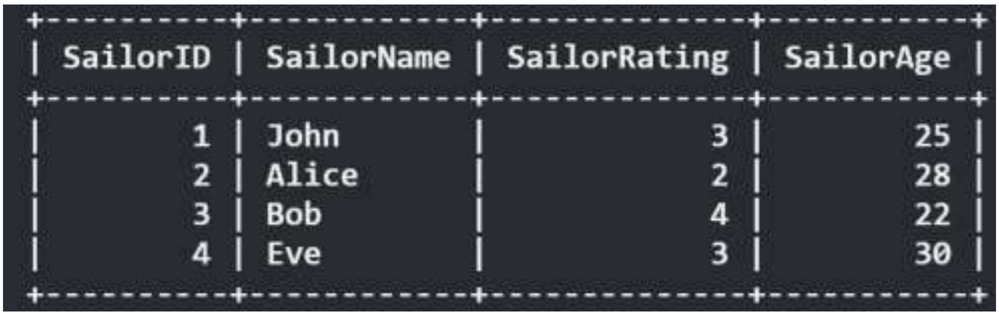
```
+------------+-----------+----------+-----------------------+-----------------+------------+----------+
| EmployeeID | FirstName | LastName | Email                 | Phone           | Department | Salary   |
+------------+-----------+----------+-----------------------+-----------------+------------+----------+
|          1 | John      | Doe      | john.doe@email.com    | +1-123-456-7890 | HR         | 55000.00 |
|          2 | Alice     | Smith    | alice.smith@email.com | +1-987-654-3210 | Finance    | 60000.00 |
|          3 | Bob       | Johnson  | bob.johnson@email.com | +1-555-555-5555 | IT         | 65000.00 |
+------------+-----------+----------+-----------------------+-----------------+------------+----------+
```

Query:SELECT *
FROM employees
WHERE salary < 22000; **Output:-**



```
+------------+----------+
| EmployeeID | Salary   |
+------------+----------+
|          1 | 19000.00 |
|          3 | 18000.00 |
+------------+----------+
```

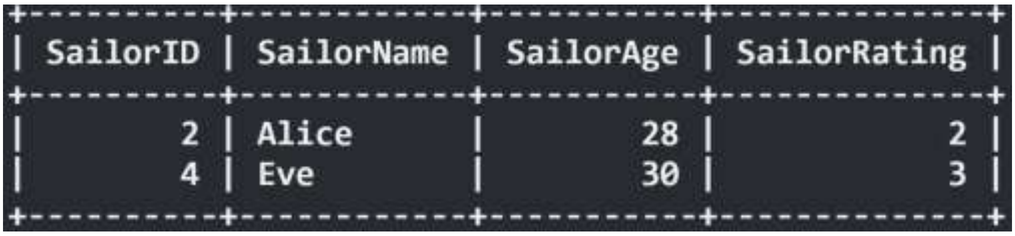
3. List the sailors having age more than 25.
Given table:-



```
+----------+------------+--------------+-----------+
| SailorID | SailorName | SailorRating | SailorAge |
+----------+------------+--------------+-----------+
|        1 | John       |            3 |        25 |
|        2 | Alice      |            2 |        28 |
|        3 | Bob        |            4 |        22 |
|        4 | Eve        |            3 |        30 |
+----------+------------+--------------+-----------+
```

Query:-

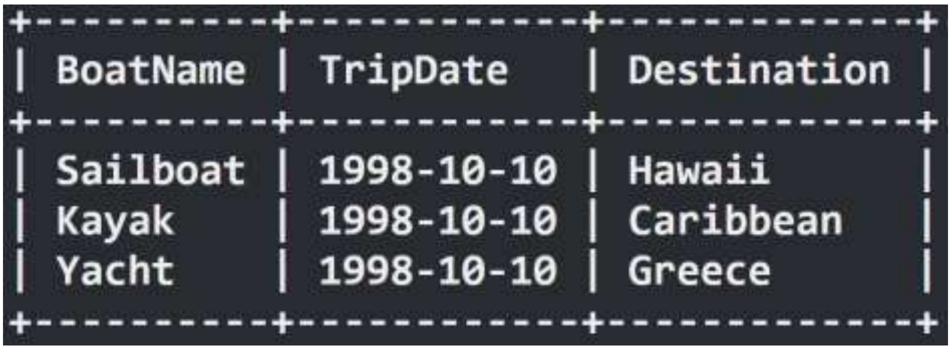
```
                    SELECT * FROM Sailors
                    WHERE SailorAge > 25
```



```
+----------+-----------+-----------+--------------+
| SailorID | SailorName | SailorAge | SailorRating |
+----------+-----------+-----------+--------------+
|        2 | Alice      |        28 |            2 |
|        4 | Eve        |        30 |            3 |
+----------+-----------+-----------+--------------+
```

4. List the boats traveling on 10-oct-98.

Given table:-

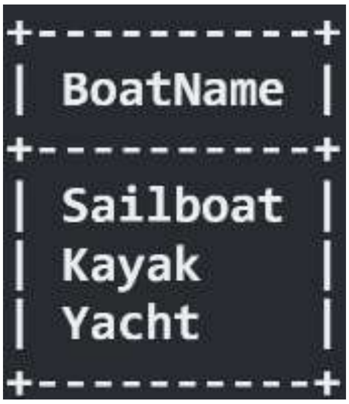


```
+----------+------------+-------------+
| BoatName | TripDate   | Destination |
+----------+------------+-------------+
| Sailboat | 1998-10-10 | Hawaii      |
| Kayak    | 1998-10-10 | Caribbean   |
| Yacht    | 1998-10-10 | Greece      |
+----------+------------+-------------+
```

Query:-

```
                SELECT BoatName
FROM BoatTrips
                  WHERE TripDate = '1998-10-10';
```

Output:-

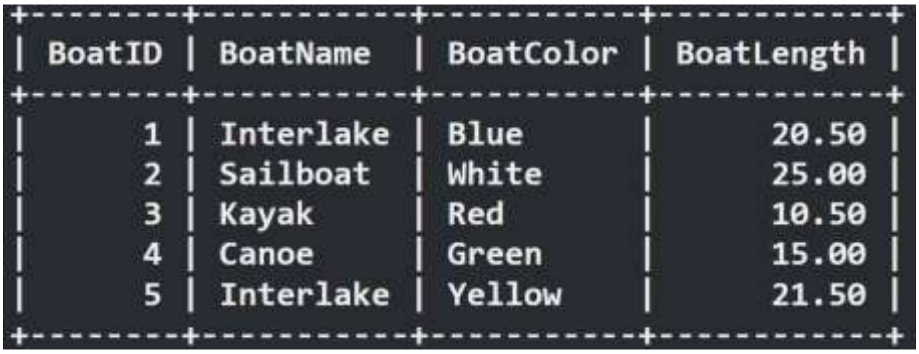


```
+----------+
| BoatName |
+----------+
| Sailboat |
| Kayak    |
| Yacht    |
+----------+
```

5. List the details of boat Interlake.

Given table:-

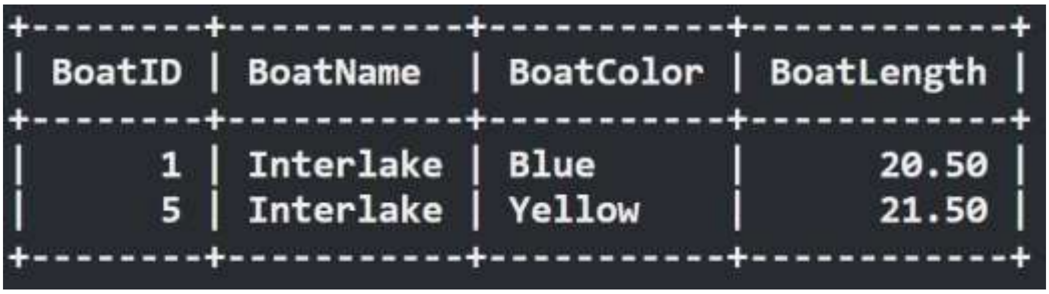
Output:-



```
+--------+-----------+-----------+------------+
| BoatID | BoatName  | BoatColor | BoatLength |
+--------+-----------+-----------+------------+
|      1 | Interlake | Blue      |      20.50 |
|      2 | Sailboat  | White     |      25.00 |
|      3 | Kayak     | Red       |      10.50 |
|      4 | Canoe     | Green     |      15.00 |
|      5 | Interlake | Yellow    |      21.50 |
+--------+-----------+-----------+------------+
```

Query:-

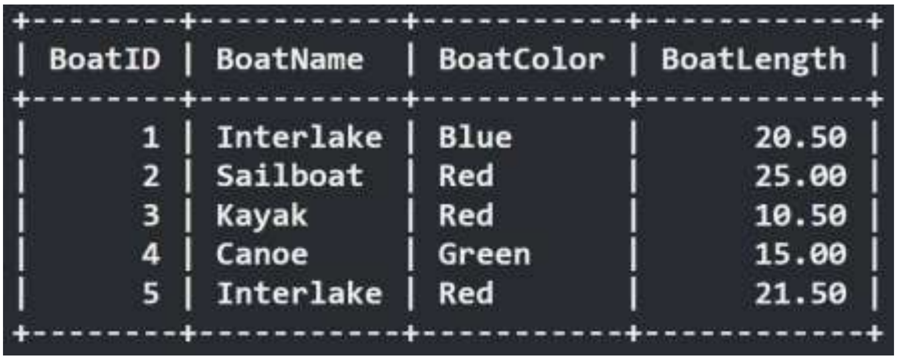
SELECT * FROM Boats
WHERE BoatName = 'Interlake'; **Output:-**



```
+--------+-----------+-----------+------------+
| BoatID | BoatName  | BoatColor | BoatLength |
+--------+-----------+-----------+------------+
|      1 | Interlake | Blue      |      20.50 |
|      5 | Interlake | Yellow    |      21.50 |
+--------+-----------+-----------+------------+
```

6. List the details of the red colored boat.

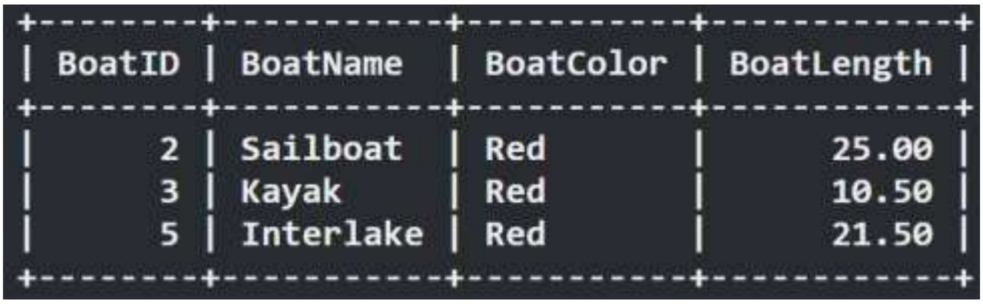
Given table:-



```
+--------+-----------+-----------+------------+
| BoatID | BoatName  | BoatColor | BoatLength |
+--------+-----------+-----------+------------+
|      1 | Interlake | Blue      |      20.50 |
|      2 | Sailboat  | Red       |      25.00 |
|      3 | Kayak     | Red       |      10.50 |
|      4 | Canoe     | Green     |      15.00 |
|      5 | Interlake | Red       |      21.50 |
+--------+-----------+-----------+------------+
```

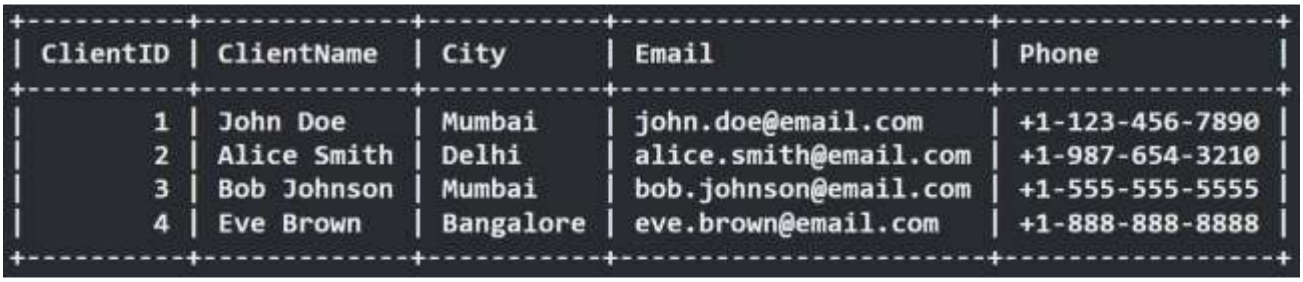
Query:-

SELECT * FROM Boats
WHERE BoatColor = 'Red';



```
+--------+-----------+-----------+------------+
| BoatID | BoatName  | BoatColor | BoatLength |
+--------+-----------+-----------+------------+
|      2 | Sailboat  | Red       |      25.00 |
|      3 | Kayak     | Red       |      10.50 |
|      5 | Interlake | Red       |      21.50 |
+--------+-----------+-----------+------------+
```

7. List the details of clients whose city is Mumbai.
Given table:-

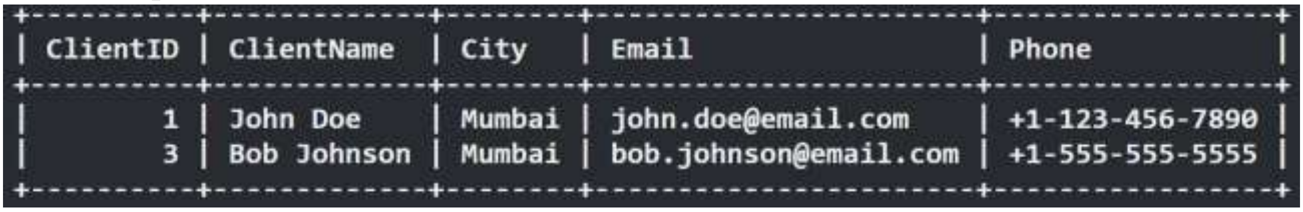


```
+----------+-------------+-----------+-------------------------+-----------------+
| ClientID | ClientName  | City      | Email                   | Phone           |
+----------+-------------+-----------+-------------------------+-----------------+
|        1 | John Doe    | Mumbai    | john.doe@email.com      | +1-123-456-7890 |
|        2 | Alice Smith | Delhi     | alice.smith@email.com   | +1-987-654-3210 |
|        3 | Bob Johnson | Mumbai    | bob.johnson@email.com   | +1-555-555-5555 |
|        4 | Eve Brown   | Bangalore | eve.brown@email.com     | +1-888-888-8888 |
+----------+-------------+-----------+-------------------------+-----------------+
```

Query:-

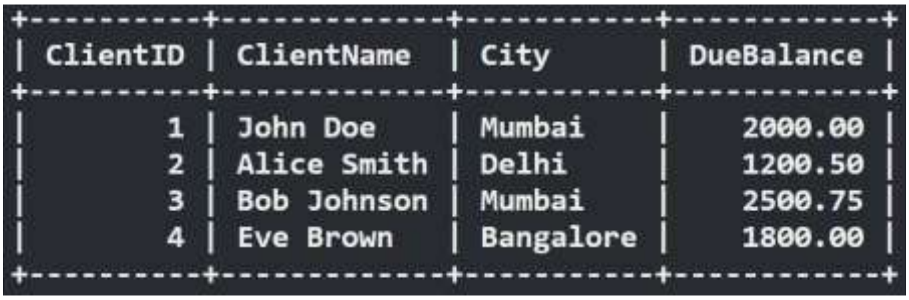
```
SELECT * FROM Clients
WHERE City = 'Mumbai';
```

Output:-



```
+----------+-------------+--------+-----------------------+-----------------+
| ClientID | ClientName  | City   | Email                 | Phone           |
+----------+-------------+--------+-----------------------+-----------------+
|        1 | John Doe    | Mumbai | john.doe@email.com    | +1-123-456-7890 |
|        3 | Bob Johnson | Mumbai | bob.johnson@email.com | +1-555-555-5555 |
+----------+-------------+--------+-----------------------+-----------------+
```

8. List Client Name, due balance and city of the clients having balance greater than 1500. Given table:-

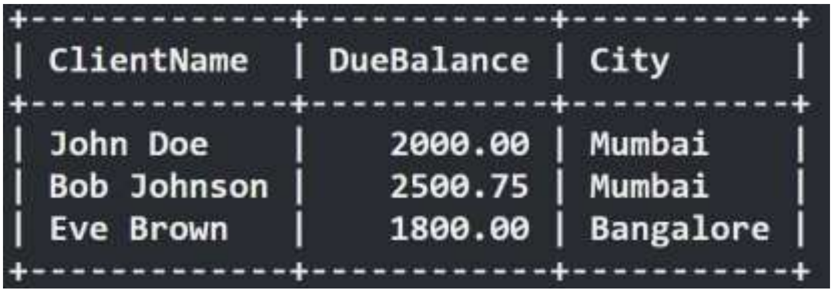


```
+----------+-------------+-----------+------------+
| ClientID | ClientName  | City      | DueBalance |
+----------+-------------+-----------+------------+
|        1 | John Doe    | Mumbai    |    2000.00 |
|        2 | Alice Smith | Delhi     |    1200.50 |
|        3 | Bob Johnson | Mumbai    |    2500.75 |
|        4 | Eve Brown   | Bangalore |    1800.00 |
+----------+-------------+-----------+------------+
```

Query:-

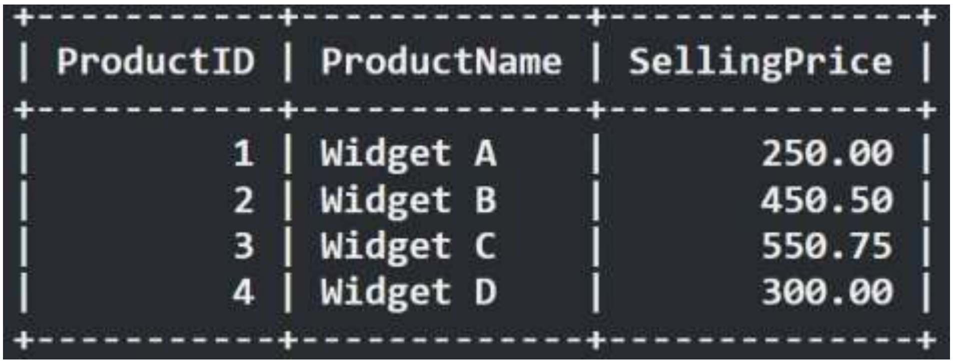
Output:-

SELECT ClientName, DueBalance, City
FROM Clients
WHERE DueBalance > 1500; **Output:-**



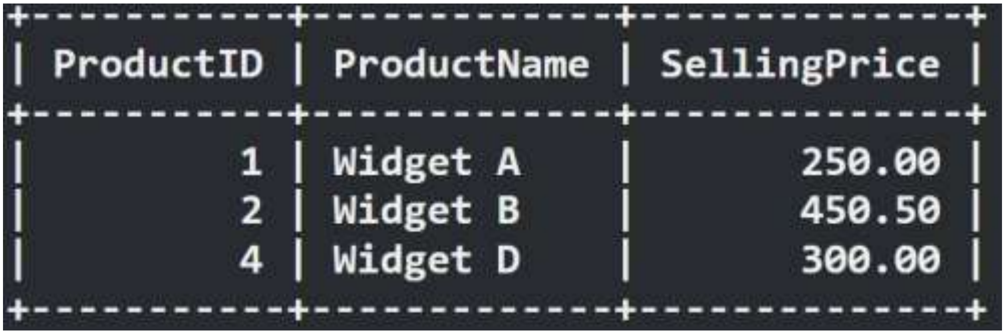
```
+-------------+------------+-----------+
| ClientName  | DueBalance | City      |
+-------------+------------+-----------+
| John Doe    |    2000.00 | Mumbai    |
| Bob Johnson |    2500.75 | Mumbai    |
| Eve Brown   |    1800.00 | Bangalore |
+-------------+------------+-----------+
```

9. Describe the details of products having selling price less than 500.
Given table:-



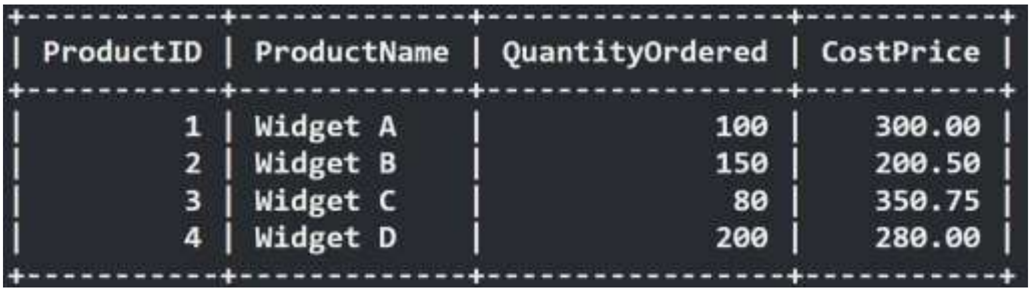
```
+-----------+-------------+--------------+
| ProductID | ProductName | SellingPrice |
+-----------+-------------+--------------+
|         1 | Widget A    |       250.00 |
|         2 | Widget B    |       450.50 |
|         3 | Widget C    |       550.75 |
|         4 | Widget D    |       300.00 |
+-----------+-------------+--------------+
```

Query:-
SELECT * FROM Products
WHERE SellingPrice < 500;



```
+-----------+-------------+--------------+
| ProductID | ProductName | SellingPrice |
+-----------+-------------+--------------+
|         1 | Widget A    |       250.00 |
|         2 | Widget B    |       450.50 |
|         4 | Widget D    |       300.00 |
+-----------+-------------+--------------+
```

10. List the products for which quantity ordered is less than 120 and cost price is greater than 250.
Given table:-

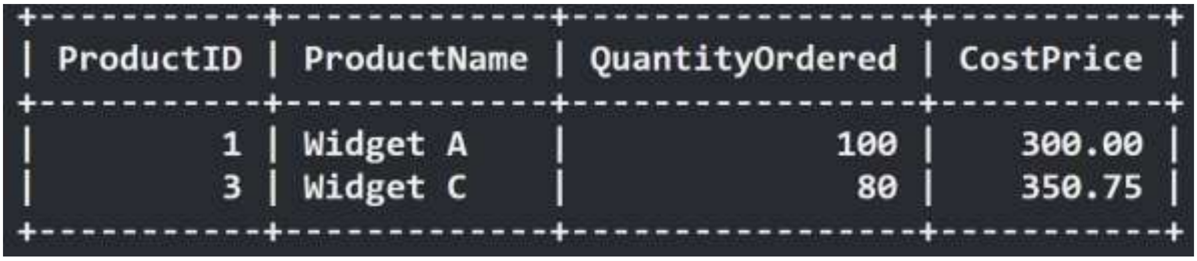


```
+-----------+-------------+-----------------+-----------+
| ProductID | ProductName | QuantityOrdered | CostPrice |
+-----------+-------------+-----------------+-----------+
|         1 | Widget A    |             100 |    300.00 |
|         2 | Widget B    |             150 |    200.50 |
|         3 | Widget C    |              80 |    350.75 |
|         4 | Widget D    |             200 |    280.00 |
+-----------+-------------+-----------------+-----------+
```

Query:-

SELECT *
FROM Products
WHERE QuantityOrdered < 120 AND CostPrice > 250;

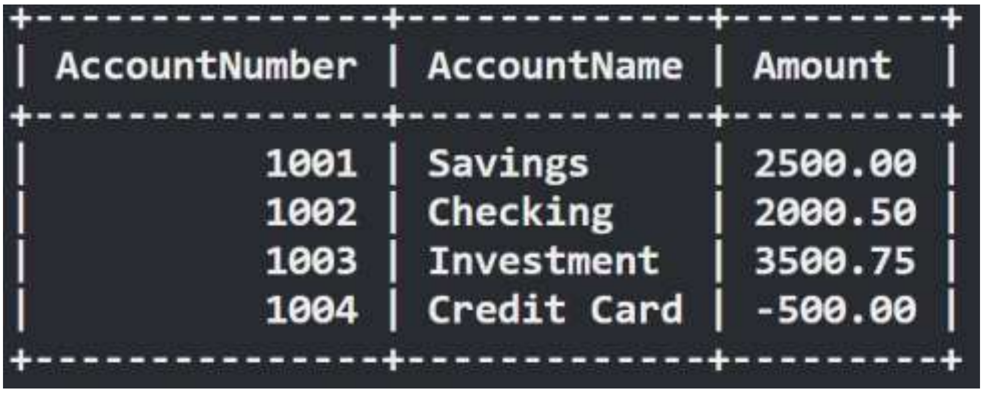
Output:-



```
+-----------+-------------+-----------------+-----------+
| ProductID | ProductName | QuantityOrdered | CostPrice |
+-----------+-------------+-----------------+-----------+
|         1 | Widget A    |             100 |    300.00 |
|         3 | Widget C    |              80 |    350.75 |
+-----------+-------------+-----------------+-----------+
```

11. Display account details having an amount greater than 2200.

Given table:-



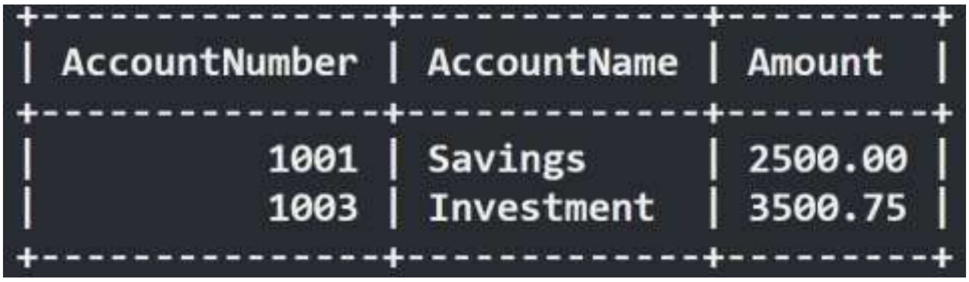
```
+---------------+-------------+---------+
| AccountNumber | AccountName | Amount  |
+---------------+-------------+---------+
|          1001 | Savings     | 2500.00 |
|          1002 | Checking    | 2000.50 |
|          1003 | Investment  | 3500.75 |
|          1004 | Credit Card | -500.00 |
+---------------+-------------+---------+
```

Query:-

SELECT * FROM Accounts
WHERE Amount > 2200;

Output:-

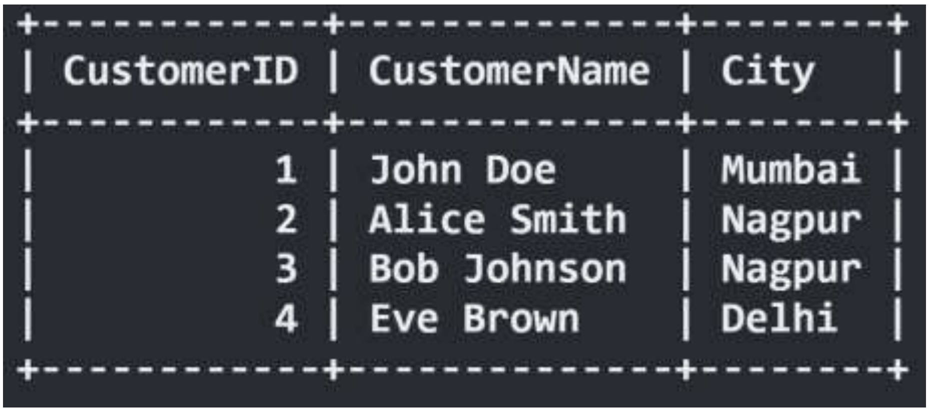
Output:-



```
+---------------+-------------+---------+
| AccountNumber | AccountName | Amount  |
+---------------+-------------+---------+
|          1001 | Savings     | 2500.00 |
|          1003 | Investment  | 3500.75 |
+---------------+-------------+---------+
```

12. Display all the customers staying in Nagpur.

Given table:-

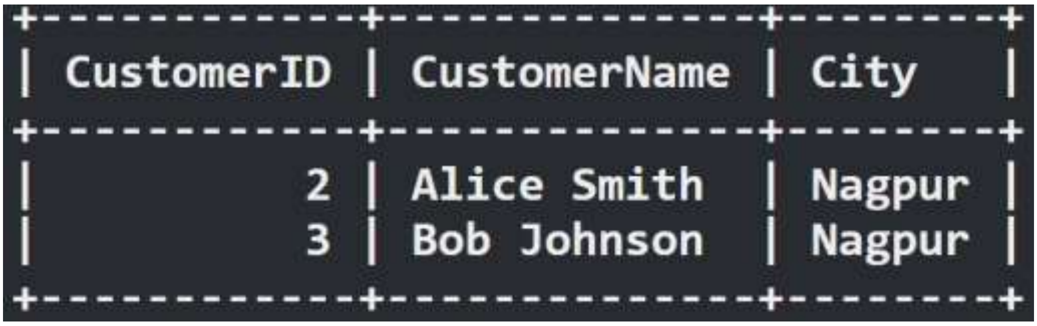


```
+------------+--------------+--------+
| CustomerID | CustomerName | City   |
+------------+--------------+--------+
|          1 | John Doe     | Mumbai |
|          2 | Alice Smith  | Nagpur |
|          3 | Bob Johnson  | Nagpur |
|          4 | Eve Brown    | Delhi  |
+------------+--------------+--------+
```

Query:-

SELECT * FROM Customers
WHERE City = 'Nagpur';

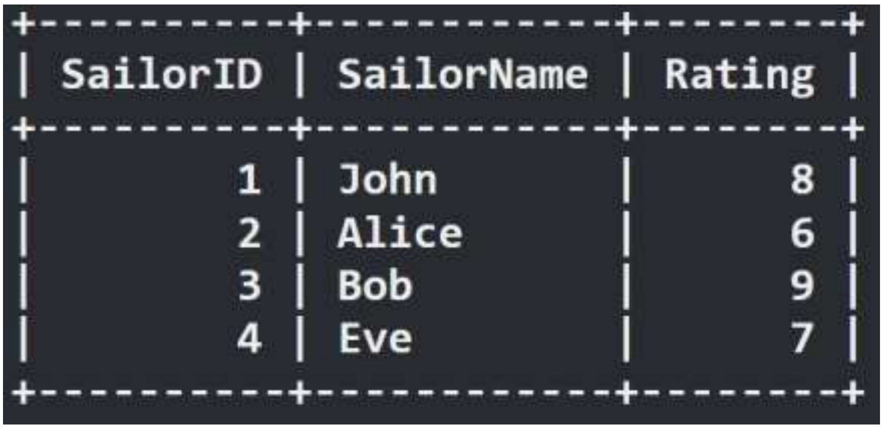
Output:-



```
+------------+--------------+--------+
| CustomerID | CustomerName | City   |
+------------+--------------+--------+
|          2 | Alice Smith  | Nagpur |
|          3 | Bob Johnson  | Nagpur |
+------------+--------------+--------+
```

13. Display the names of sailors having ratings greater than 7.

Given table:-

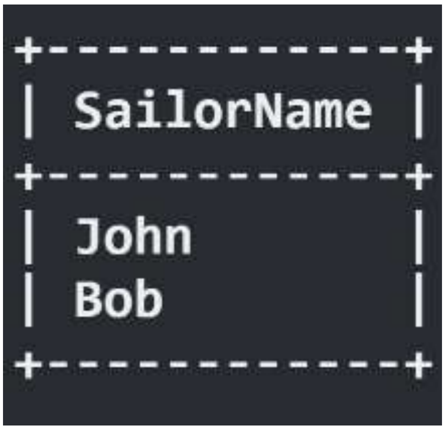


```
+---------+-----------+--------+
| SailorID | SailorName | Rating |
+---------+-----------+--------+
|        1 | John       |      8 |
|        2 | Alice      |      6 |
|        3 | Bob        |      9 |
|        4 | Eve        |      7 |
+---------+-----------+--------+
```

Query:-

```
SELECT SailorName
FROM Sailors
WHERE Rating > 7;
```

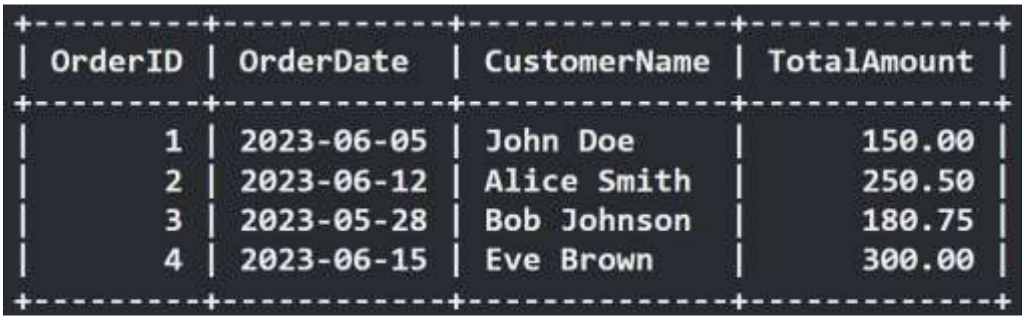
Output:-



```
+------------+
| SailorName |
+------------+
| John       |
| Bob        |
+------------+
```

14. Display the orders made in the month of June.

Given table:-

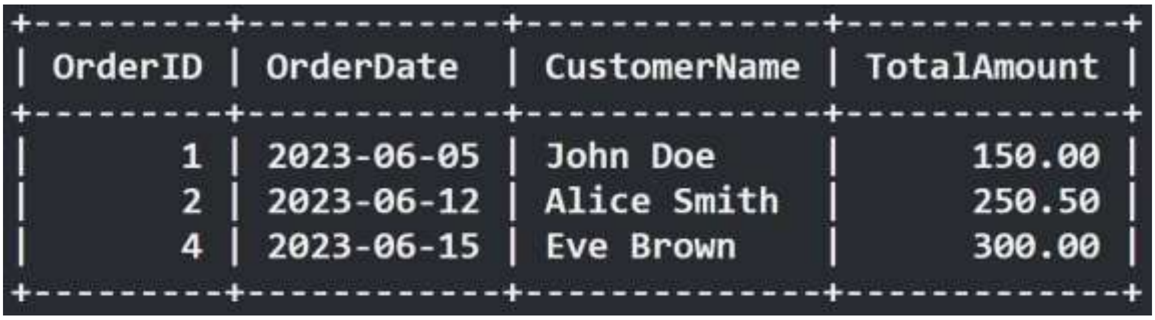


```
+---------+------------+-------------+-------------+
| OrderID | OrderDate  | CustomerName | TotalAmount |
+---------+------------+-------------+-------------+
|       1 | 2023-06-05 | John Doe     |      150.00 |
|       2 | 2023-06-12 | Alice Smith  |      250.50 |
|       3 | 2023-05-28 | Bob Johnson  |      180.75 |
|       4 | 2023-06-15 | Eve Brown    |      300.00 |
+---------+------------+-------------+-------------+
```

Query:-

SELECT * FROM Orders
WHERE MONTH(OrderDate) = 6;

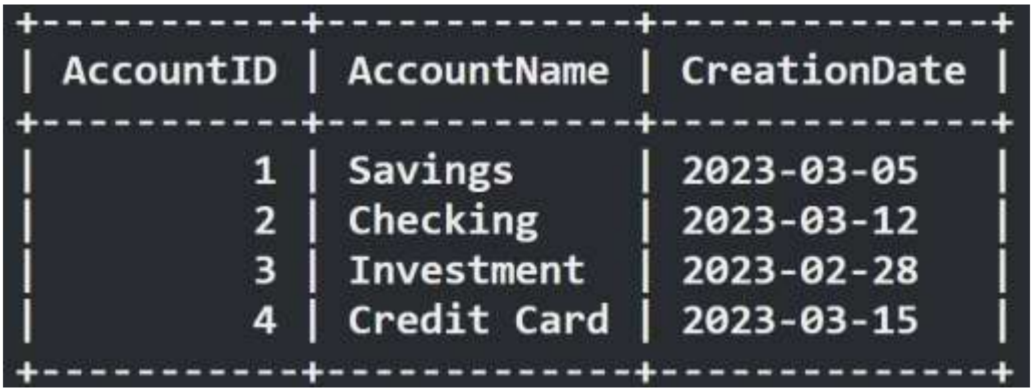
Output:-



```
+---------+------------+--------------+-------------+
| OrderID | OrderDate  | CustomerName | TotalAmount |
+---------+------------+--------------+-------------+
|       1 | 2023-06-05 | John Doe     |      150.00 |
|       2 | 2023-06-12 | Alice Smith  |      250.50 |
|       4 | 2023-06-15 | Eve Brown    |      300.00 |
+---------+------------+--------------+-------------+
```

15. List all the accounts created in the month of March.

Given table:-

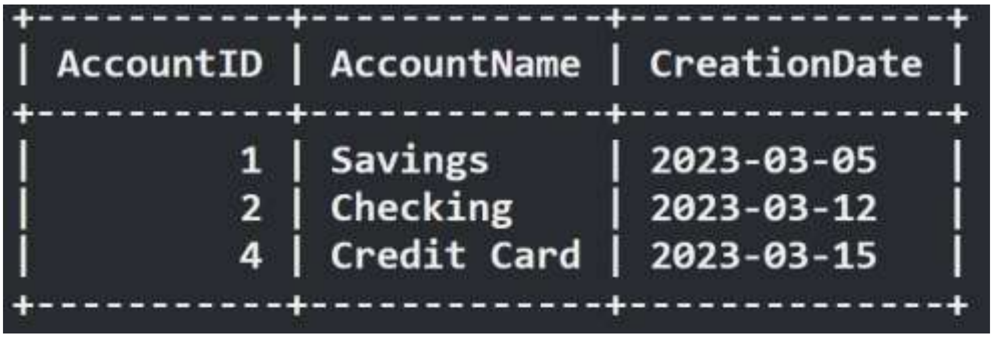


```
+-----------+-------------+--------------+
| AccountID | AccountName | CreationDate |
+-----------+-------------+--------------+
|         1 | Savings     | 2023-03-05   |
|         2 | Checking    | 2023-03-12   |
|         3 | Investment  | 2023-02-28   |
|         4 | Credit Card | 2023-03-15   |
+-----------+-------------+--------------+
```

Query:-

SELECT * FROM Accounts
WHERE MONTH(CreationDate) = 3;

Output:-



```
+-----------+-------------+--------------+
| AccountID | AccountName | CreationDate |
+-----------+-------------+--------------+
|         1 | Savings     | 2023-03-05   |
|         2 | Checking    | 2023-03-12   |
|         4 | Credit Card | 2023-03-15   |
+-----------+-------------+--------------+
```

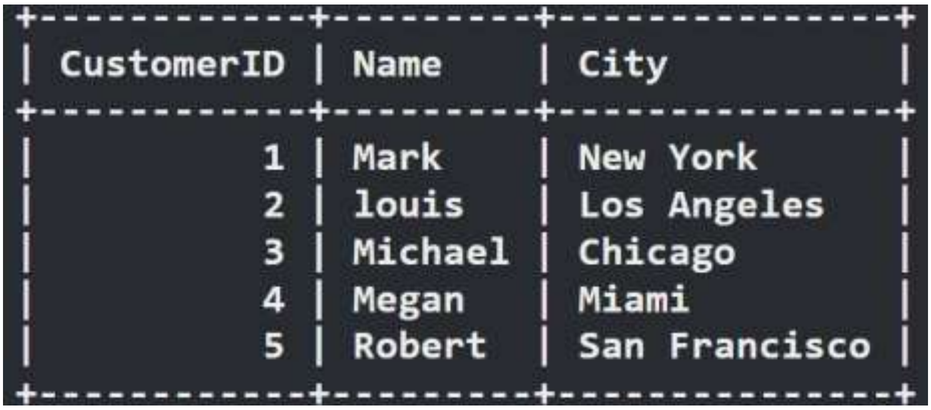
EXPERIMENT NO: 5

AIM:-

To study various options of LIKE predicate LikeQueries:

1. **Display all customers whose name start with ‘M’**
2. **Display all the customers whose name ends with ‘L’**
3. **Display all loan details whose branch starts with ‘A’**
4. **Display the sailors whose name is minimum 6 characters long.**
5. **Display the details of Employees whose address starts with ‘S’**
6. **List the details of the boat ending with ‘e’**
7. **List the details of clients having ‘h’ as a 3rd character in his/her name.**
8. **List Client Name, due balance and city whose pin code starts with 4.**
9. **List all customers whose city contains ‘a’ as a second character.**
10. **List client names and city whose state has ‘a’ as fourth or fifth character.**

- **Given Table of Customers:-**



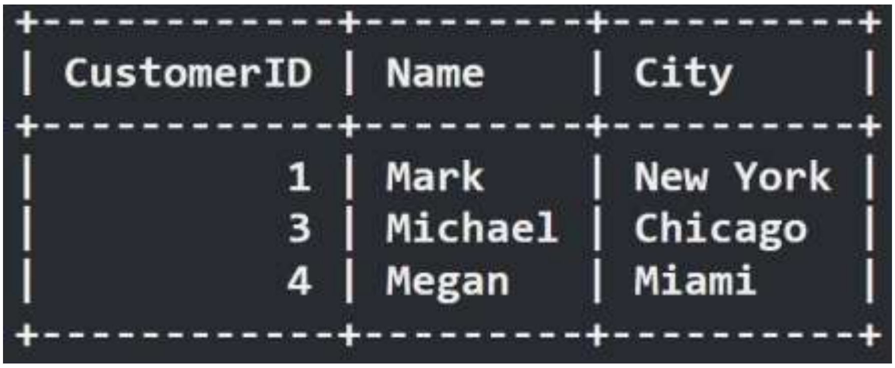
```
+------------+---------+---------------+
| CustomerID | Name    | City          |
+------------+---------+---------------+
|          1 | Mark    | New York      |
|          2 | louis   | Los Angeles   |
|          3 | Michael | Chicago       |
|          4 | Megan   | Miami         |
|          5 | Robert  | San Francisco |
+------------+---------+---------------+
```

1. **Display all customers whose name start with ‘M’**

Query:-

SELECT * FROM Customers WHERE Name LIKE 'M%';

Output:-



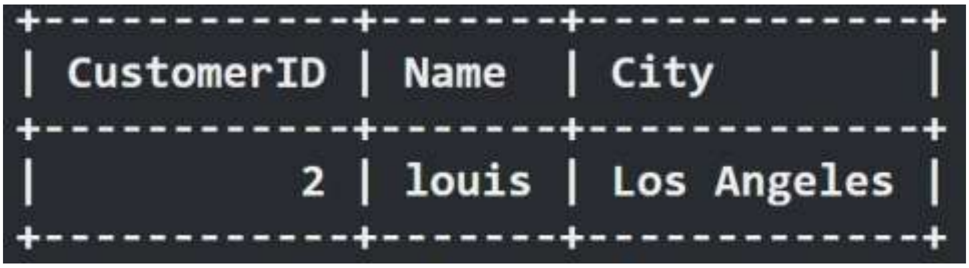
```
+------------+---------+----------+
| CustomerID | Name    | City     |
+------------+---------+----------+
|          1 | Mark    | New York |
|          3 | Michael | Chicago  |
|          4 | Megan   | Miami    |
+------------+---------+----------+
```

2. **Display all the customers whose name ends with 'L'**

Query:-

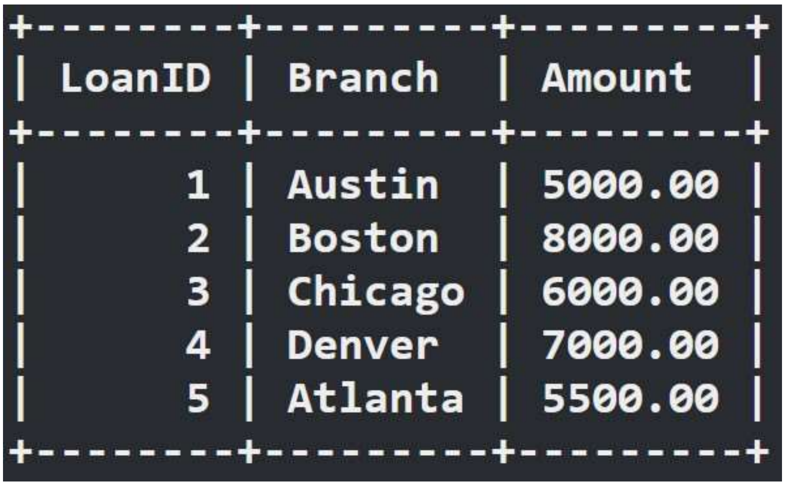
SELECT * FROM Customers WHERE Name LIKE 'L%';

Output:-



```
+------------+-------+-------------+
| CustomerID | Name  | City        |
+------------+-------+-------------+
|          2 | louis | Los Angeles |
+------------+-------+-------------+
```

- **Given Table Loan:-**

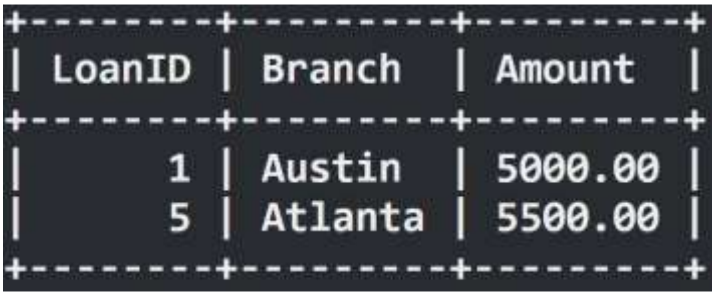


```
+--------+---------+---------+
| LoanID | Branch  | Amount  |
+--------+---------+---------+
|      1 | Austin  | 5000.00 |
|      2 | Boston  | 8000.00 |
|      3 | Chicago | 6000.00 |
|      4 | Denver  | 7000.00 |
|      5 | Atlanta | 5500.00 |
+--------+---------+---------+
```

3. **Display all loan details whose branch starts with 'A' Query:-**

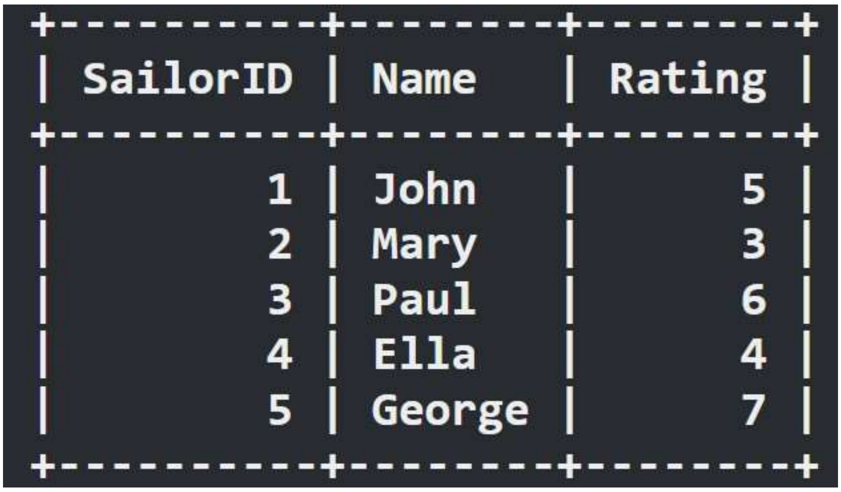
SELECT * FROM Loans WHERE Branch LIKE 'A%';

Output:-



```
+--------+---------+---------+
| LoanID | Branch  | Amount  |
+--------+---------+---------+
|      1 | Austin  | 5000.00 |
|      5 | Atlanta | 5500.00 |
+--------+---------+---------+
```

- **Given Table Sailors:-**



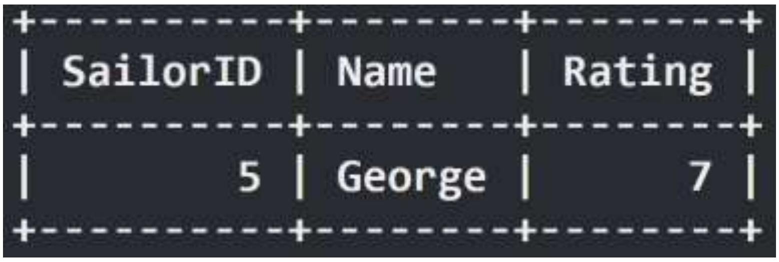
```
+----------+--------+--------+
| SailorID | Name   | Rating |
+----------+--------+--------+
|        1 | John   |      5 |
|        2 | Mary   |      3 |
|        3 | Paul   |      6 |
|        4 | Ella   |      4 |
|        5 | George |      7 |
+----------+--------+--------+
```

4. Display the sailors whose name is minimum 6 characters long.

Query:-

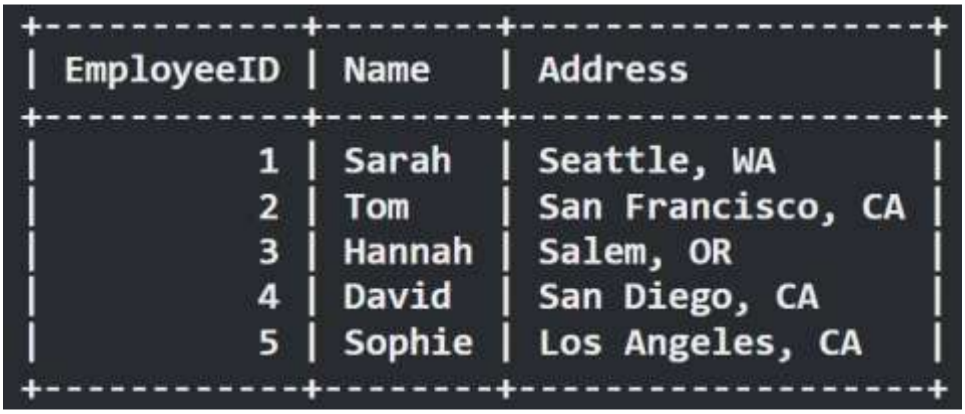
SELECT * FROM Sailors WHERE LENGTH(Name) >= 6;

Output:-



```
+----------+--------+--------+
| SailorID | Name   | Rating |
+----------+--------+--------+
|        5 | George |      7 |
+----------+--------+--------+
```

- **Given Table Employee:-**

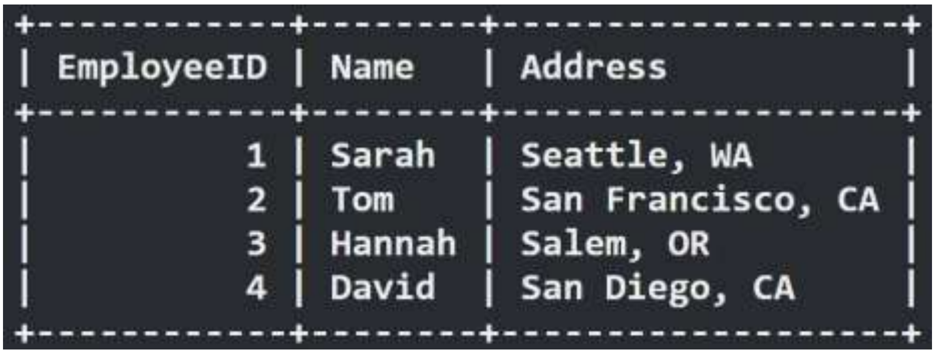


```
+------------+--------+-------------------+
| EmployeeID | Name   | Address           |
+------------+--------+-------------------+
|          1 | Sarah  | Seattle, WA       |
|          2 | Tom    | San Francisco, CA |
|          3 | Hannah | Salem, OR         |
|          4 | David  | San Diego, CA     |
|          5 | Sophie | Los Angeles, CA   |
+------------+--------+-------------------+
```

5. Display the details of Employees whose address starts with 'S' Query:-

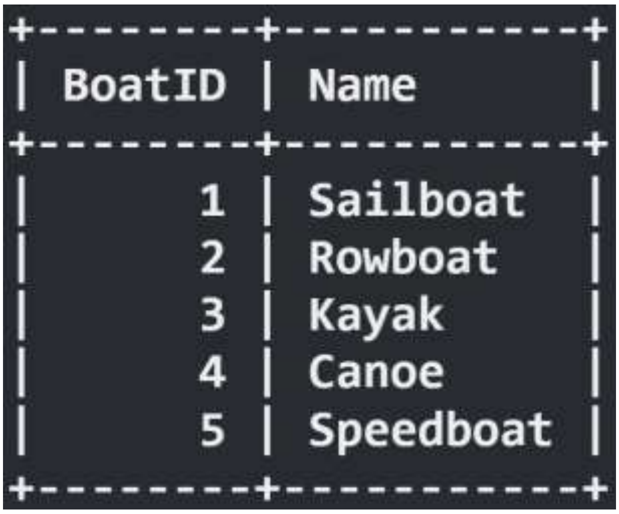
SELECT * FROM Employees WHERE Address LIKE 'S%';

Output:-



```
+------------+--------+--------------------+
| EmployeeID | Name   | Address            |
+------------+--------+--------------------+
|          1 | Sarah  | Seattle, WA        |
|          2 | Tom    | San Francisco, CA  |
|          3 | Hannah | Salem, OR          |
|          4 | David  | San Diego, CA      |
+------------+--------+--------------------+
```

- **Given Table Boats:-**



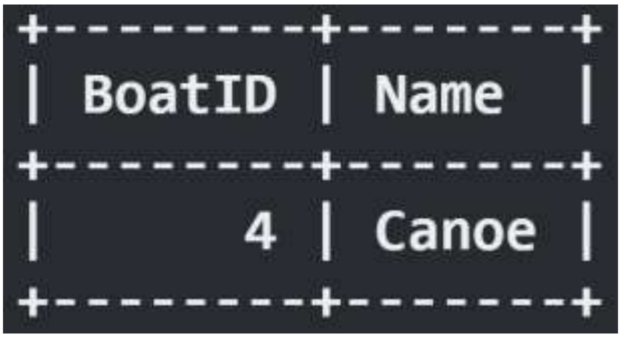
```
+--------+-----------+
| BoatID | Name      |
+--------+-----------+
|      1 | Sailboat  |
|      2 | Rowboat   |
|      3 | Kayak     |
|      4 | Canoe     |
|      5 | Speedboat |
+--------+-----------+
```

6. List the details of the boat ending with 'e'

Query:-

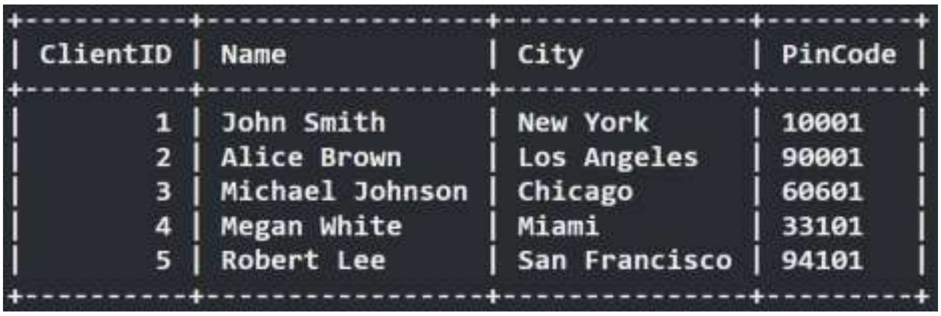
SELECT * FROM Boats WHERE Name LIKE '%e';

Output:-



```
+--------+-------+
| BoatID | Name  |
+--------+-------+
|      4 | Canoe |
+--------+-------+
```

- **Given Table Client:-**

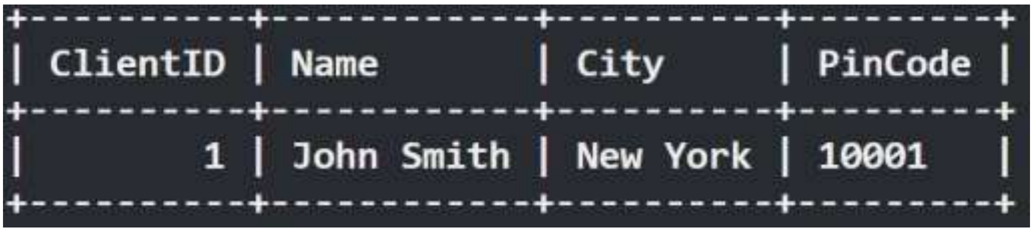


```
+----------+-----------------+---------------+---------+
| ClientID | Name            | City          | PinCode |
+----------+-----------------+---------------+---------+
|        1 | John Smith      | New York      | 10001   |
|        2 | Alice Brown     | Los Angeles   | 90001   |
|        3 | Michael Johnson | Chicago       | 60601   |
|        4 | Megan White     | Miami         | 33101   |
|        5 | Robert Lee      | San Francisco | 94101   |
+----------+-----------------+---------------+---------+
```

7. List the details of clients having ‘h’ as a 3rd character in his/her name.

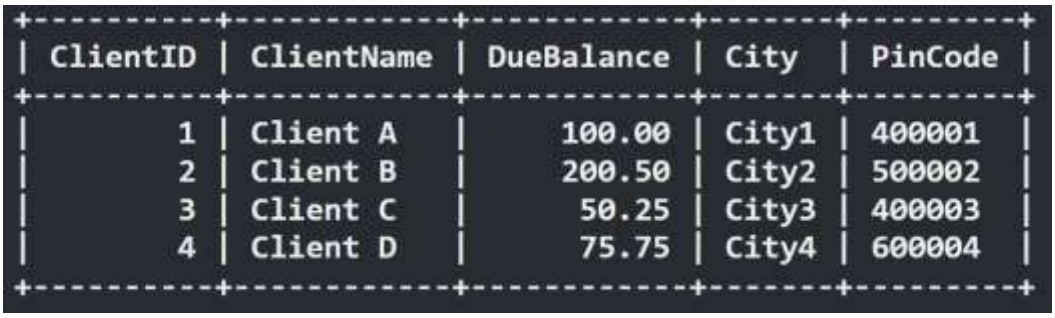
Query:-

SELECT * FROM Clients WHERE SUBSTRING(Name, 3, 1) = 'h'; **Output:-**



```
+----------+------------+----------+---------+
| ClientID | Name       | City     | PinCode |
+----------+------------+----------+---------+
|        1 | John Smith | New York | 10001   |
+----------+------------+----------+---------+
```

- **Given Table Costumer**

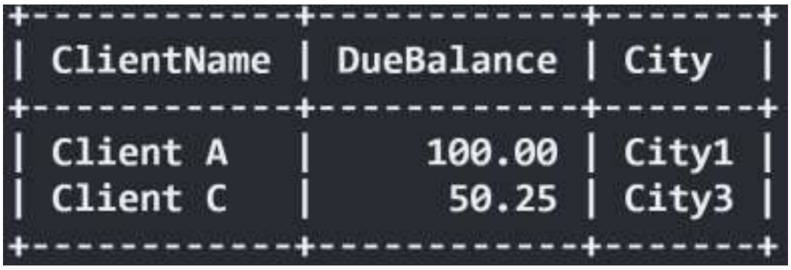


```
+----------+------------+------------+-------+---------+
| ClientID | ClientName | DueBalance | City  | PinCode |
+----------+------------+------------+-------+---------+
|        1 | Client A   |     100.00 | City1 | 400001  |
|        2 | Client B   |     200.50 | City2 | 500002  |
|        3 | Client C   |      50.25 | City3 | 400003  |
|        4 | Client D   |      75.75 | City4 | 600004  |
+----------+------------+------------+-------+---------+
```

8. List Client Name, due balance and city whose pin code starts with 4.

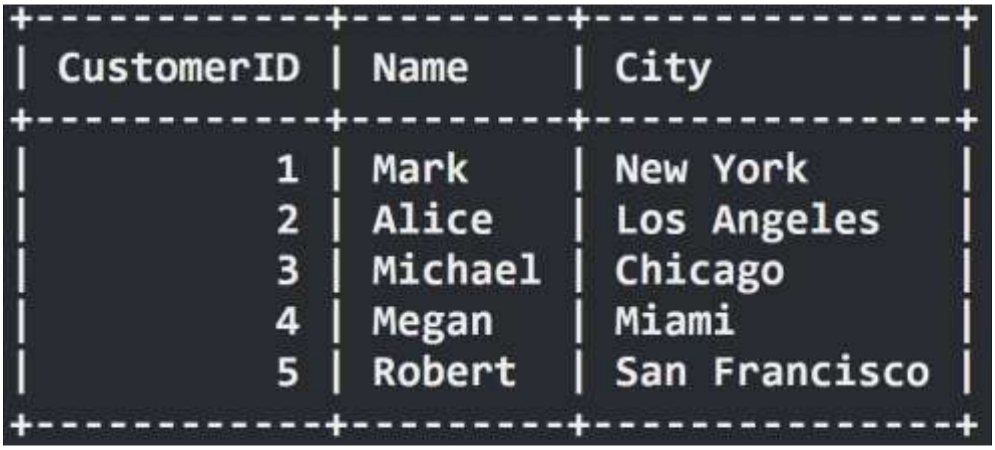
Query:-

SELECT ClientName, DueBalance, City
FROM Clients
WHERE LEFT(PinCode, 1) = '4'; **Output:-**



```
+------------+------------+-------+
| ClientName | DueBalance | City  |
+------------+------------+-------+
| Client A   |     100.00 | City1 |
| Client C   |      50.25 | City3 |
+------------+------------+-------+
```

- **Given Table Customer:-**



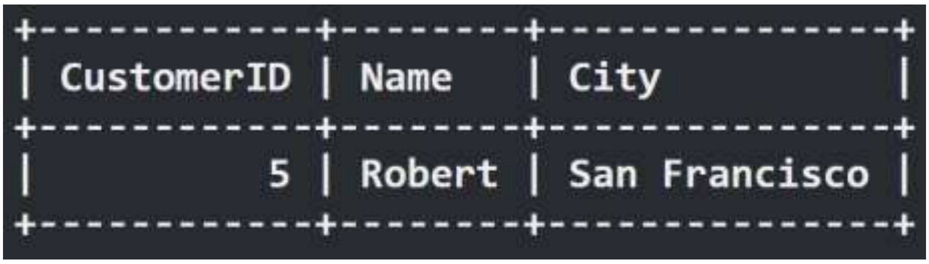
```
+------------+---------+---------------+
| CustomerID | Name    | City          |
+------------+---------+---------------+
|          1 | Mark    | New York      |
|          2 | Alice   | Los Angeles   |
|          3 | Michael | Chicago       |
|          4 | Megan   | Miami         |
|          5 | Robert  | San Francisco |
+------------+---------+---------------+
```

9. List all customers whose city contains ‘a’ as a second character.

Query:-

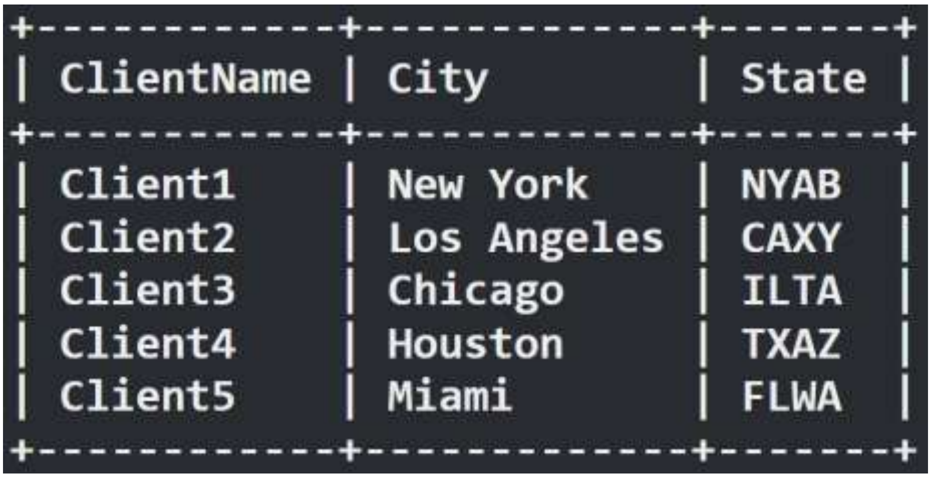
SELECT * FROM Customers WHERE City LIKE '_a%';

Output:-



```
+------------+--------+---------------+
| CustomerID | Name   | City          |
+------------+--------+---------------+
|          5 | Robert | San Francisco |
+------------+--------+---------------+
```

- **Given Table Client:-**



```
+------------+-------------+-------+
| ClientName | City        | State |
+------------+-------------+-------+
| Client1    | New York    | NYAB  |
| Client2    | Los Angeles | CAXY  |
| Client3    | Chicago     | ILTA  |
| Client4    | Houston     | TXAZ  |
| Client5    | Miami       | FLWA  |
+------------+-------------+-------+
```

10. List client names and city whose state has ‘a’ as fourth or fifth character.

Query:-

SELECT ClientName, City FROM Clients WHERE SUBSTRING(State, 4, 1)= 'A'

EXPERIMENT NO: 6

AIM:-

Aggregate Functions & DML Queries:

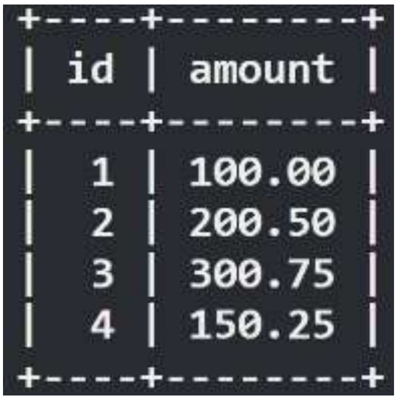
1. **List total deposit from deposit.**
2. **Give Maximum loan given to a customer.**
3. **Describe the average age of all the sailors.**
4. **Count total number of customers**
5. **Count the total number of customer 's' cities.**
6. **Display total target for the salesman.**
7. **Update the salary of the employee having 10000 to 11500**
8. **Update the city of clients from Bangalore to Bengaluru.**
9. **Give the 15% hike in the salary of all the Employees. Rename that column to 'New Salary'**
10. **Increase the sell price of all products by 20% and label new column as 'New Sell Price' (Do not update the table)**
11. **Provide the count of customers staying in 'Bombay'**

Introduction/Purpose:-

1. **List total deposit from deposit.**

To list the total deposit amount from a table named "deposit" using a Data Manipulation Language (DML) query, you can use an aggregate function like SUM(). Assuming you have a table named "deposit" with a column named "amount" that represents the deposit amounts.

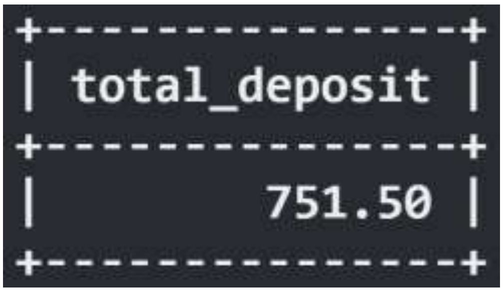
Example:-



```
+----+--------+
| id | amount |
+----+--------+
|  1 | 100.00 |
|  2 | 200.50 |
|  3 | 300.75 |
|  4 | 150.25 |
+----+--------+
```

Query:-

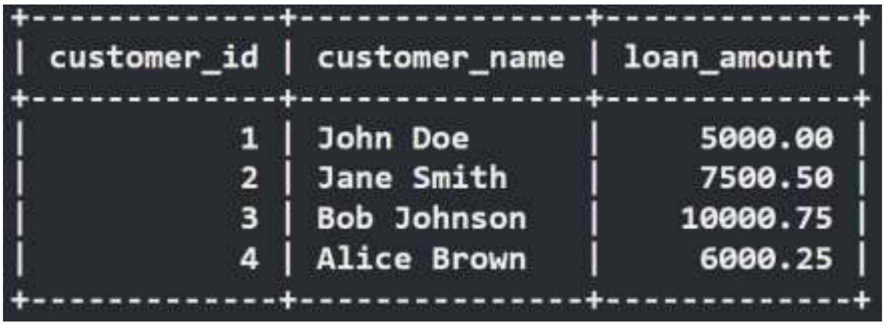
SELECT SUM(amount) AS total_deposit
FROM deposit; **Output:-**



```
+---------------+
| total_deposit |
+---------------+
|        751.50 |
+---------------+
```

2. Give Maximum loan given to a customer.

To find the maximum loan given to a customer, you can use the MAX() aggregate function in SQL. Assuming you have a table named "loans" with a column named "loan_amount" that represents the loan amounts given to customers. **Example:**

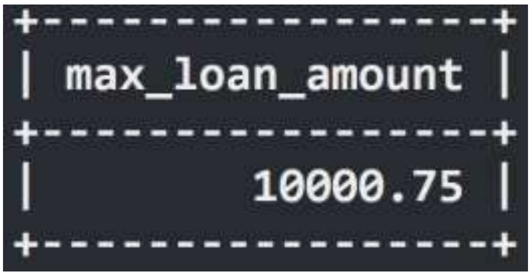


```
+-------------+---------------+-------------+
| customer_id | customer_name | loan_amount |
+-------------+---------------+-------------+
|           1 | John Doe      |     5000.00 |
|           2 | Jane Smith    |     7500.50 |
|           3 | Bob Johnson   |    10000.75 |
|           4 | Alice Brown   |     6000.25 |
+-------------+---------------+-------------+
```

Query:-

SELECT MAX(loan_amount) AS max_loan_amount FROM loans;

Output:-

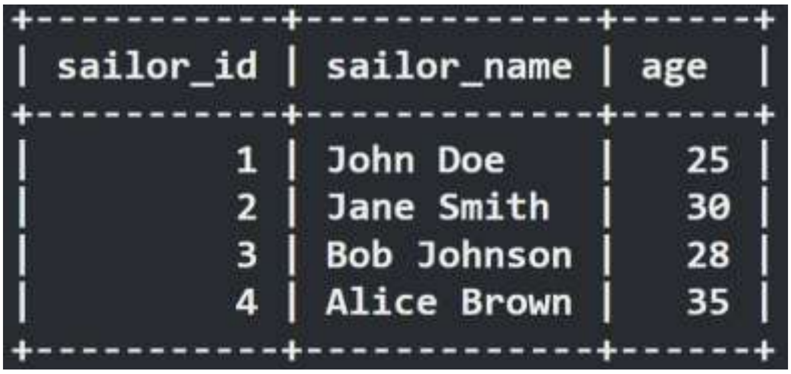


```
+-----------------+
| max_loan_amount |
+-----------------+
|        10000.75 |
+-----------------+
```

3. Describe the average age of all the sailors.

To describe the average age of all the sailors in a database table, you can use SQL and an appropriate query. Assuming you have a table named "sailors" with a column named "age" that contains the age of each sailor, you can calculate the average age using the AVG() aggregate function.

Example:-

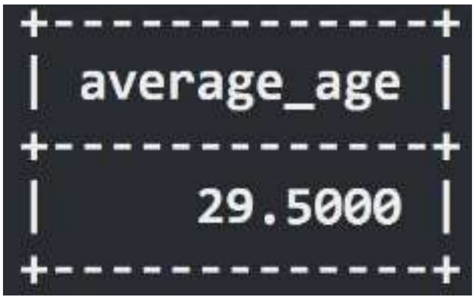


```
+-----------+-------------+------+
| sailor_id | sailor_name | age  |
+-----------+-------------+------+
|         1 | John Doe    |   25 |
|         2 | Jane Smith  |   30 |
|         3 | Bob Johnson |   28 |
|         4 | Alice Brown |   35 |
+-----------+-------------+------+
```

Query:-

SELECT AVG(age) AS average_age
FROM sailors;

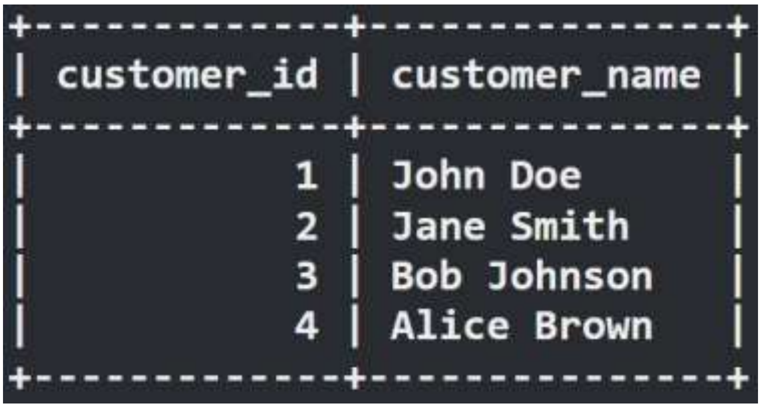
Output:-



```
+-------------+
| average_age |
+-------------+
|     29.5000 |
+-------------+
```

4. Count total number of customers

To count the total number of customers in a database table, you can use the SQL COUNT() function. Assuming you have a table named "customers" with a column named "customer_id" that represents unique customer IDs **Example:**

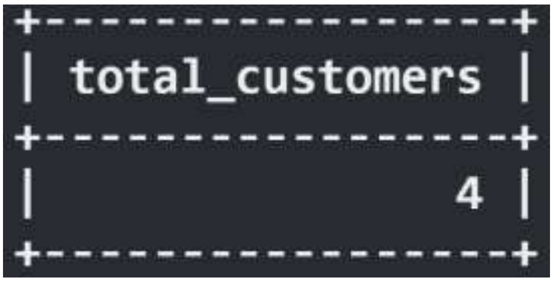


```
+-------------+---------------+
| customer_id | customer_name |
+-------------+---------------+
|           1 | John Doe      |
|           2 | Jane Smith    |
|           3 | Bob Johnson   |
|           4 | Alice Brown   |
+-------------+---------------+
```

Query:-

SELECT COUNT(customer_id) AS total_customers FROM customers;

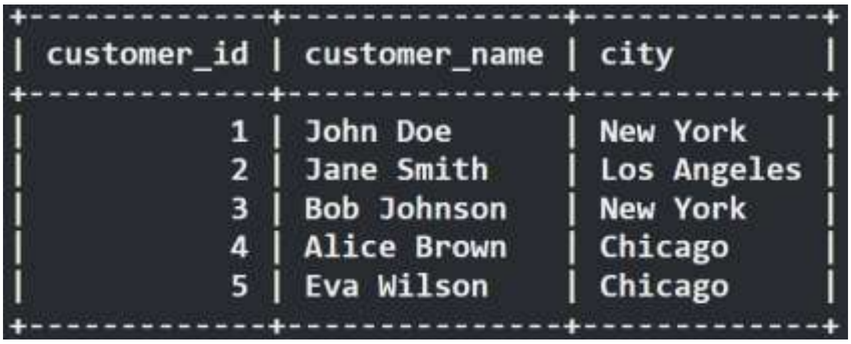
Output:-



```
+-----------------+
| total_customers |
+-----------------+
|               4 |
+-----------------+
```

5. Count the total number of customer 's' cities.

To count the total number of unique cities that customers are from in a database table, you can use the SQL COUNT(DISTINCT column_name) function along with a GROUP BY clause. Assuming you have a table named "customers" with a column named "city" that represents the city where each customer lives. **Example:**

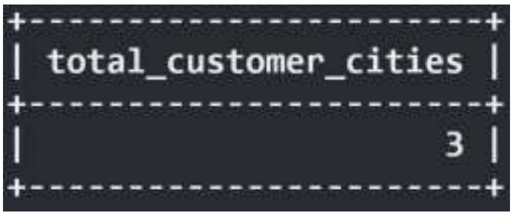


```
+-------------+---------------+-------------+
| customer_id | customer_name | city        |
+-------------+---------------+-------------+
|           1 | John Doe      | New York    |
|           2 | Jane Smith    | Los Angeles |
|           3 | Bob Johnson   | New York    |
|           4 | Alice Brown   | Chicago     |
|           5 | Eva Wilson    | Chicago     |
+-------------+---------------+-------------+
```

Query:-

SELECT COUNT(DISTINCT city) AS total_customer_cities FROM customers;

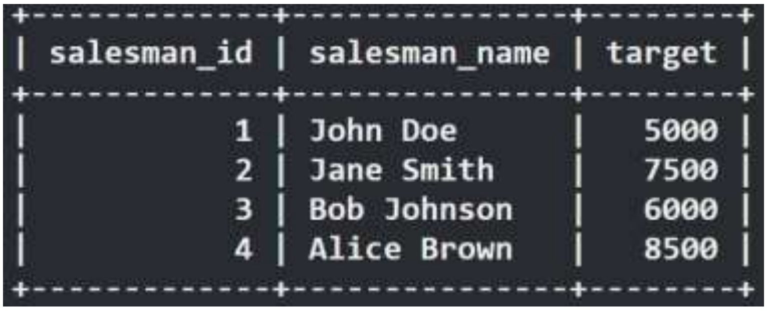
Output:-



```
+-----------------------+
| total_customer_cities |
+-----------------------+
|                     3 |
+-----------------------+
```

6. Display total target for the salesman.

To display the total target for the salesman, you need to have a table that contains information about salesmen and their respective targets. Assuming you have a table named "salesmen" with columns "salesman_id" and "target," where "salesman_id" is a unique identifier for each salesman, and "target" is the sales target for each salesman. **Example:**

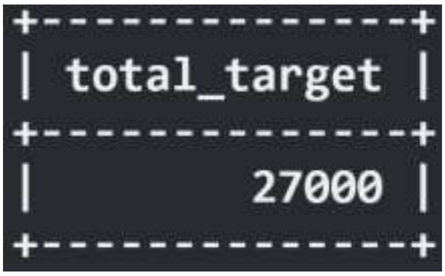


```
+-------------+---------------+--------+
| salesman_id | salesman_name | target |
+-------------+---------------+--------+
|           1 | John Doe      |   5000 |
|           2 | Jane Smith    |   7500 |
|           3 | Bob Johnson   |   6000 |
|           4 | Alice Brown   |   8500 |
+-------------+---------------+--------+
```

Query:-

SELECT SUM(target) AS total_target
FROM salesmen;

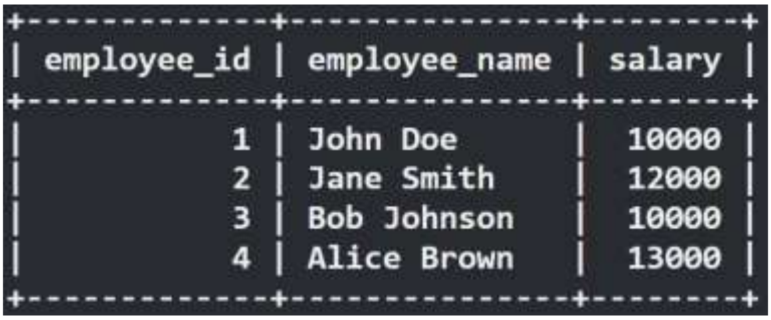
Output:-



```
+--------------+
| total_target |
+--------------+
|        27000 |
+--------------+
```

7. Update the salary of the employee having 10000 to 11500

To update the salary of an employee whose current salary is 10000 to a new salary of 11500, you can use an SQL UPDATE statement with a WHERE clause to specify the condition for the update. Assuming you have a table named "employees" with columns "employee_id" and "salary," and you want to update the salary of the employee with a salary of 10000. **Example:**

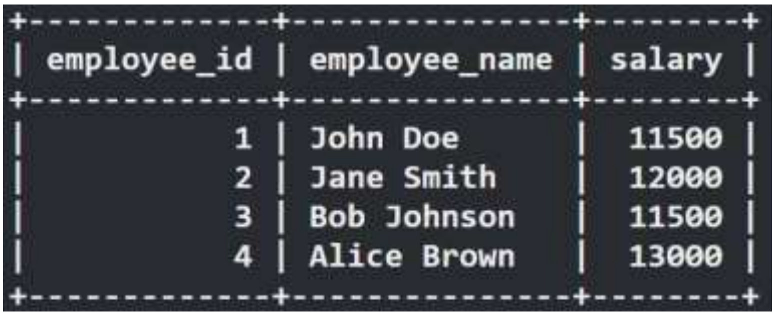


```
+-------------+---------------+--------+
| employee_id | employee_name | salary |
+-------------+---------------+--------+
|           1 | John Doe      |  10000 |
|           2 | Jane Smith    |  12000 |
|           3 | Bob Johnson   |  10000 |
|           4 | Alice Brown   |  13000 |
+-------------+---------------+--------+
```

Query:-

UPDATE employees
SET salary = 11500
WHERE salary = 10000;
SELECT * FROM employees;

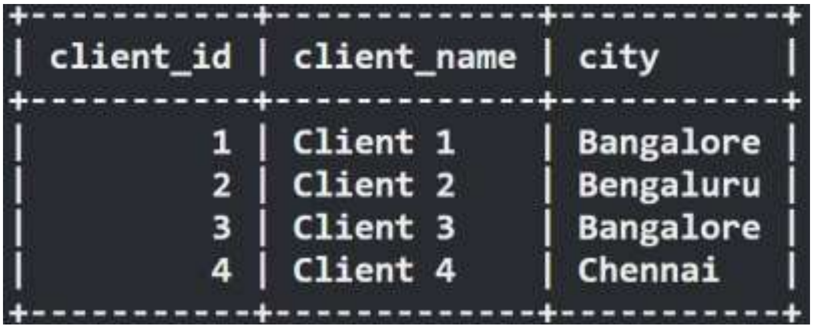
Output:-



```
+-------------+---------------+--------+
| employee_id | employee_name | salary |
+-------------+---------------+--------+
|           1 | John Doe      |  11500 |
|           2 | Jane Smith    |  12000 |
|           3 | Bob Johnson   |  11500 |
|           4 | Alice Brown   |  13000 |
+-------------+---------------+--------+
```

8. Update the city of clients from Bangalore to Bengaluru.

To update the city of clients from "Bangalore" to "Bengaluru" in a database table, you can use an SQL UPDATE statement with a WHERE clause to specify the condition for the update. Assuming you have a table named "clients" with columns "client_id" and "city," and you want to update the city for clients who are currently in "Bangalore”. **Example:**

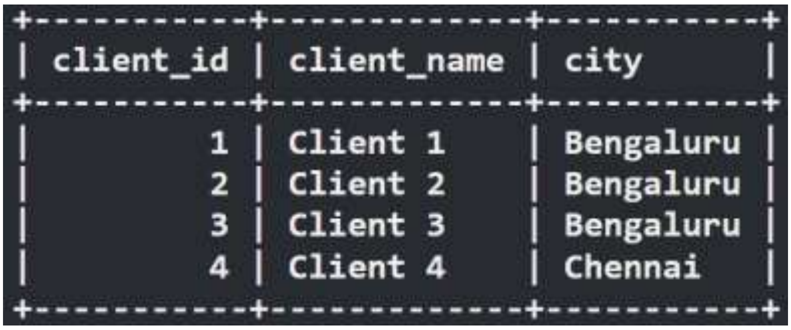


```
+-----------+-------------+-----------+
| client_id | client_name | city      |
+-----------+-------------+-----------+
|         1 | Client 1    | Bangalore |
|         2 | Client 2    | Bengaluru |
|         3 | Client 3    | Bangalore |
|         4 | Client 4    | Chennai   |
+-----------+-------------+-----------+
```

Query:-

UPDATE clients
SET city = 'Bengaluru'
WHERE city = 'Bangalore';
SELECT* FROM clients;

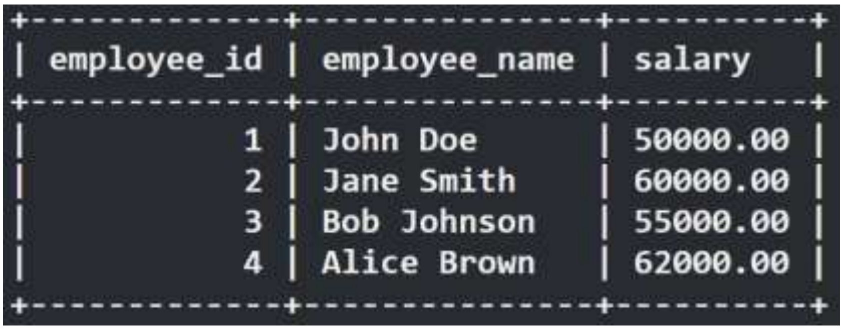
Output:-



```
+-----------+-------------+-----------+
| client_id | client_name | city      |
+-----------+-------------+-----------+
|         1 | Client 1    | Bengaluru |
|         2 | Client 2    | Bengaluru |
|         3 | Client 3    | Bengaluru |
|         4 | Client 4    | Chennai   |
+-----------+-------------+-----------+
```

9. Give the 15% hike in the salary of all the Employees. Rename that column to ‘New Salary’.

To give a 15% hike in the salary of all employees and rename that column to "New Salary," you can use an SQL UPDATE statement to perform the salary increase and an SQL ALTER TABLE statement to rename the column. **Example:**



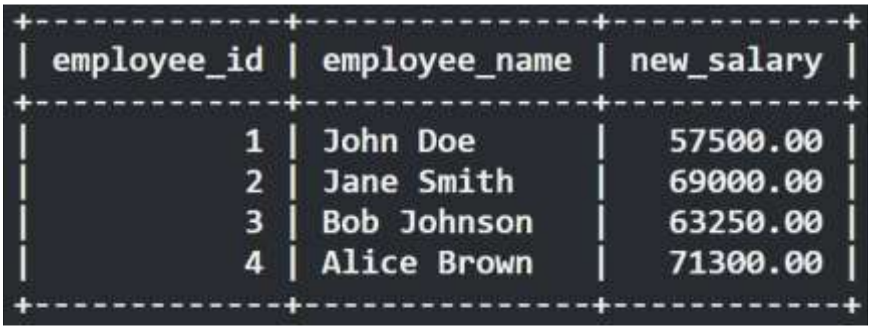
```
+-------------+---------------+----------+
| employee_id | employee_name | salary   |
+-------------+---------------+----------+
|           1 | John Doe      | 50000.00 |
|           2 | Jane Smith    | 60000.00 |
|           3 | Bob Johnson   | 55000.00 |
|           4 | Alice Brown   | 62000.00 |
+-------------+---------------+----------+
```

Query:-

UPDATE employees
SET salary = salary * 1.15; -- Apply a 15% salary increase
ALTER TABLE employees
CHANGE COLUMN salary new_salary DECIMAL(10, 2);

SELECT * FROM employees;

Output:-



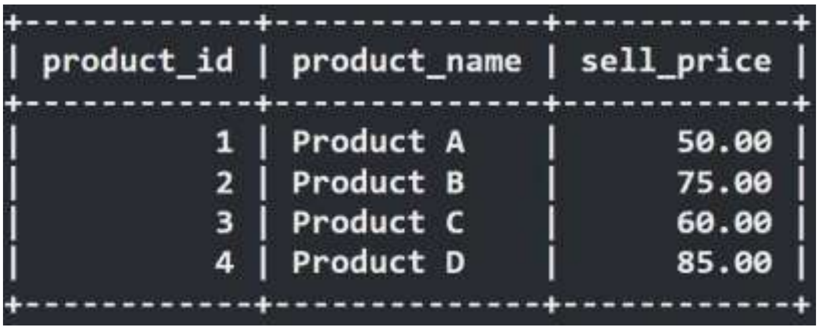
```
+-------------+---------------+------------+
| employee_id | employee_name | new_salary |
+-------------+---------------+------------+
|           1 | John Doe      |   57500.00 |
|           2 | Jane Smith    |   69000.00 |
|           3 | Bob Johnson   |   63250.00 |
|           4 | Alice Brown   |   71300.00 |
+-------------+---------------+------------+
```

10. Increase the sell price of all products by 20% and label new column as 'New Sell Price' (Do not update the table)

If you want to calculate and display a 20% increase in the selling price of all products without updating the table itself, you can achieve this by using a SELECT statement to calculate the new sell price and label it as "New Sell

Price."

Example:

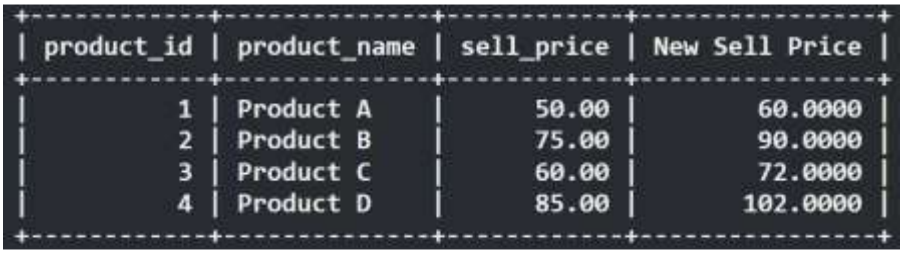


```
+------------+--------------+------------+
| product_id | product_name | sell_price |
+------------+--------------+------------+
|          1 | Product A    |      50.00 |
|          2 | Product B    |      75.00 |
|          3 | Product C    |      60.00 |
|          4 | Product D    |      85.00 |
+------------+--------------+------------+
```

Query:-

SELECT product_id, product_name, sell_price, (sell_price * 1.20) AS "New
Sell Price"
FROM products;
SELECT *FROM products;

Output:-

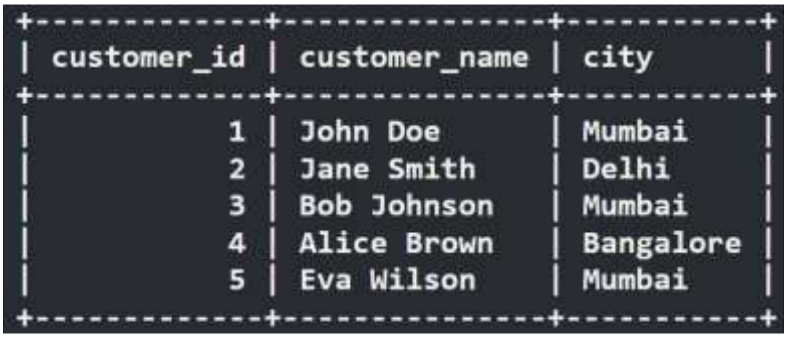


```
+------------+--------------+------------+----------------+
| product_id | product_name | sell_price | New Sell Price |
+------------+--------------+------------+----------------+
|          1 | Product A    |      50.00 |        60.0000 |
|          2 | Product B    |      75.00 |        90.0000 |
|          3 | Product C    |      60.00 |        72.0000 |
|          4 | Product D    |      85.00 |       102.0000 |
+------------+--------------+------------+----------------+
```

11. Provide the count of customers staying in 'Bombay'

To count the number of customers staying in 'Bombay' (now known as Mumbai), you can use the SQL COUNT() function along with a WHERE clause to specify the condition. Assuming you have a table named "customers" with a column named "city" that represents the city where each customer lives.

Example:

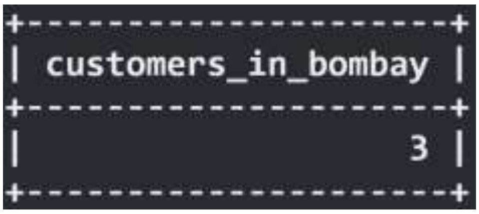


```
+-------------+---------------+-----------+
| customer_id | customer_name | city      |
+-------------+---------------+-----------+
|           1 | John Doe      | Mumbai    |
|           2 | Jane Smith    | Delhi     |
|           3 | Bob Johnson   | Mumbai    |
|           4 | Alice Brown   | Bangalore |
|           5 | Eva Wilson    | Mumbai    |
+-------------+---------------+-----------+
```

Query:-

```
SELECT COUNT(*) AS customers_in_bombay
FROM customers
WHERE city = 'Mumbai';
```

Output:-



```
+---------------------+
| customers_in_bombay |
+---------------------+
|                   3 |
+---------------------+
```

EXPERIMENT NO: 7

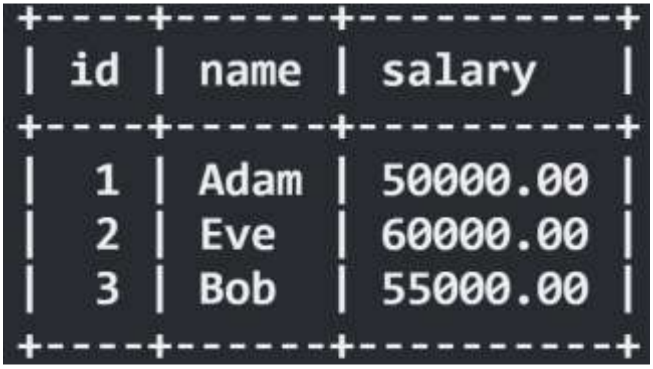
AIM:-

‘Join’ Queries:

1. Find the salary of Adam.
2. Find the city where Brooks works.
3. Display the sailor’s details whose boat is booked for 9th May, 98.
4. Display the day of ride and sailor name for boat 103.
5. Display the sailor name and its age for Red colored and 101 boats.
6. Display the sailor details whose boat is never booked.
7. Display the sailor name that has Red or Green Boat.
8. Display all sailor details and boat details and who has an Interlake boat.
9. Display sailors rating with boat details or the trip on 10th October, 98.
10. Display the sailor id and name whose age is more than 42 or who has a Blue colored boat.
11. Display name and rating of sailor whose boat name is Clipper.
12. List products whose selling price is more than 500 and less than equal to 750.
13. Describe the second highest salary of an employee.
14. Display the date of travel and sailor’s name whose age is between 35 and 65.
15. List all the employees working for ‘FBC’

1. Find the salary of Adam.

To find the salary of a person named "Adam" in a MySQL database, you would need to have a table that contains salary information along with the corresponding names. Assuming you have a table named "employees" with columns "name" and "salary." **Example:**

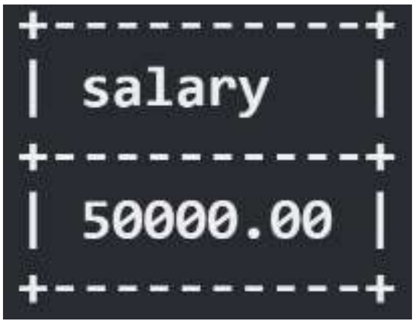


```
+----+------+----------+
| id | name | salary   |
+----+------+----------+
|  1 | Adam | 50000.00 |
|  2 | Eve  | 60000.00 |
|  3 | Bob  | 55000.00 |
+----+------+----------+
```

Query:-

```
SELECT salary
FROM employees
WHERE name = 'Adam';
```

Output:-

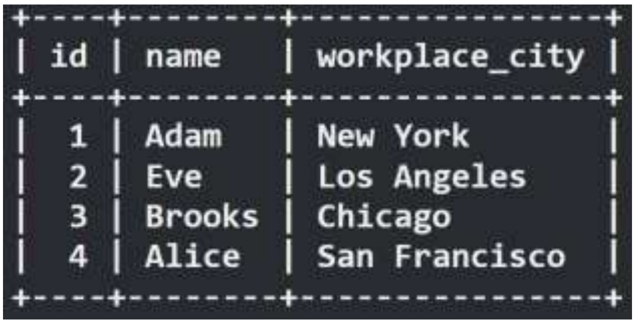


```
+----------+
| salary   |
+----------+
| 50000.00 |
+----------+
```

2. Find the city where Brooks works.

To find the city where someone named "Brooks" works, you would need a table that contains information about employees and their workplaces, including the city. Assuming you have a table named "employees" with

columns "name" and "workplace_city." **Example:**

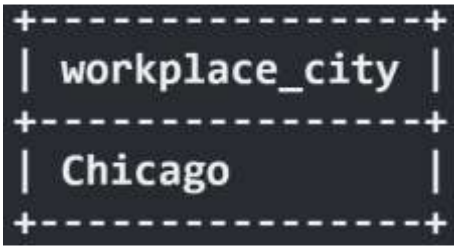


```
+----+--------+---------------+
| id | name   | workplace_city |
+----+--------+---------------+
|  1 | Adam   | New York      |
|  2 | Eve    | Los Angeles   |
|  3 | Brooks | Chicago       |
|  4 | Alice  | San Francisco |
+----+--------+---------------+
```

Query:-

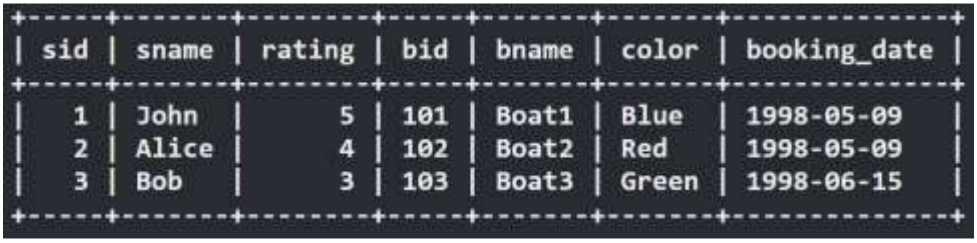
SELECT workplace_city
FROM employees
WHERE name = 'Brooks';

Output:-



```
+----------------+
| workplace_city |
+----------------+
| Chicago        |
+----------------+
```

3. Display the sailor's details whose boat is booked for 9th May, 98. To display the details of sailors whose boat is booked for May 9, 1998, you need to have a database schema that includes tables for sailors, boats, and bookings, and these tables should be appropriately linked with foreign keys. **Example:**

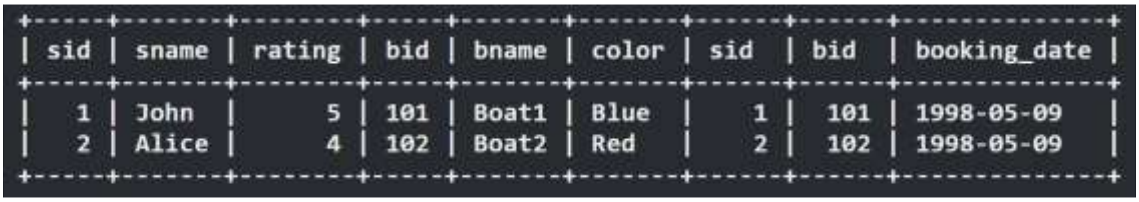


```
+-----+-------+--------+-----+-------+-------+--------------+
| sid | sname | rating | bid | bname | color | booking_date |
+-----+-------+--------+-----+-------+-------+--------------+
|   1 | John  |      5 | 101 | Boat1 | Blue  | 1998-05-09   |
|   2 | Alice |      4 | 102 | Boat2 | Red   | 1998-05-09   |
|   3 | Bob   |      3 | 103 | Boat3 | Green | 1998-06-15   |
+-----+-------+--------+-----+-------+-------+--------------+
```

Query:-

SELECT sailors.*, boats.*, bookings.*
FROM sailors
JOIN bookings ON sailors.sid = bookings.sid
JOIN boats ON bookings.bid = boats.bid
WHERE bookings.booking_date = '1998-05-09';

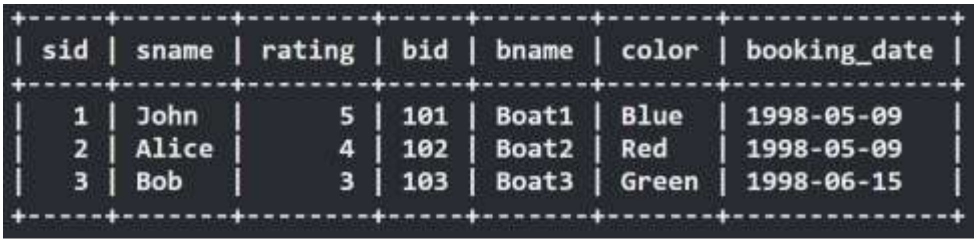
Output:-



```
+-----+-------+--------+-----+-------+-------+------+------+--------------+
| sid | sname | rating | bid | bname | color | sid  | bid  | booking_date |
+-----+-------+--------+-----+-------+-------+------+------+--------------+
|   1 | John  |      5 | 101 | Boat1 | Blue  |    1 |  101 | 1998-05-09   |
|   2 | Alice |      4 | 102 | Boat2 | Red   |    2 |  102 | 1998-05-09   |
+-----+-------+--------+-----+-------+-------+------+------+--------------+
```

4. Display the day of ride and sailor name for boat 103.

To display the day of the ride and the sailor's name for boat 103. **Example:**

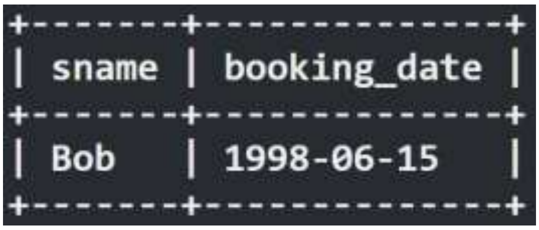


```
+-----+-------+--------+-----+-------+-------+--------------+
| sid | sname | rating | bid | bname | color | booking_date |
+-----+-------+--------+-----+-------+-------+--------------+
|   1 | John  |      5 | 101 | Boat1 | Blue  | 1998-05-09   |
|   2 | Alice |      4 | 102 | Boat2 | Red   | 1998-05-09   |
|   3 | Bob   |      3 | 103 | Boat3 | Green | 1998-06-15   |
+-----+-------+--------+-----+-------+-------+--------------+
```

Query:-

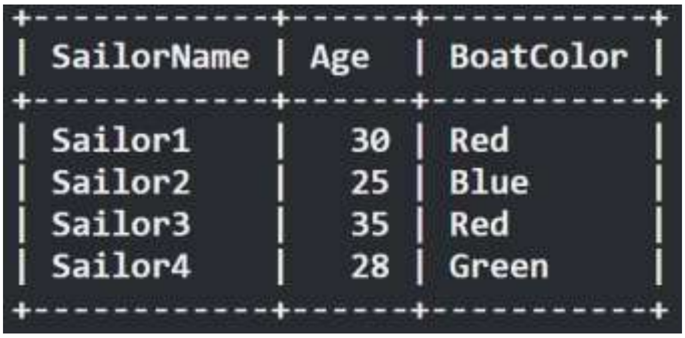
SELECT sailors.sname, bookings.booking_date
FROM sailors
JOIN bookings ON sailors.sid = bookings.sid
WHERE bookings.bid = 103;

Output:-



```
+-------+--------------+
| sname | booking_date |
+-------+--------------+
| Bob   | 1998-06-15   |
+-------+--------------+
```

5. Display the sailor name and its age for Red colored and 101 boats. To display the sailor name and age for sailors who have booked Red colored boats (color='Red') with boat ID 101 (bid=101), you'll need to calculate the age of the sailors based on their birthdate and then join the "sailors" and "boats" tables with the "bookings" table to filter the desired information. **Example:**



```
+------------+------+-----------+
| SailorName | Age  | BoatColor |
+------------+------+-----------+
| Sailor1    |   30 | Red       |
| Sailor2    |   25 | Blue      |
| Sailor3    |   35 | Red       |
| Sailor4    |   28 | Green     |
+------------+------+-----------+
```

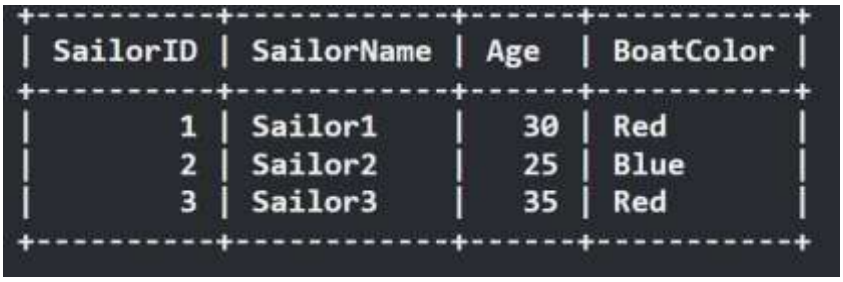
Query:-

SELECT sailors.sname, TIMESTAMPDIFF(YEAR, sailors.birthdate,
CURDATE()) AS age
FROM sailors
JOIN bookings ON sailors.sid = bookings.sid
JOIN boats ON bookings.bid = boats.bid
WHERE boats.color = 'Red' AND boats.bid = 101;

Output:-

6. Display the sailor details whose boat is never booked.

To display the sailor details whose boat is never booked, you can use a SQL query with a LEFT JOIN and a condition that checks for null values in the BoatID column of the Sailors table. **Example:**

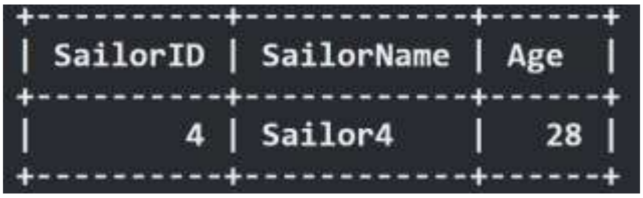


```
+----------+------------+------+-----------+
| SailorID | SailorName | Age  | BoatColor |
+----------+------------+------+-----------+
|        1 | Sailor1    |   30 | Red       |
|        2 | Sailor2    |   25 | Blue      |
|        3 | Sailor3    |   35 | Red       |
+----------+------------+------+-----------+
```

Query:-

SELECT S.SailorID, S.SailorName, S.Age
FROM Sailors S
LEFT JOIN Boats B ON S.BoatID = B.BoatID
WHERE B.BoatID IS NULL OR S.BoatID IS NULL;

Output:-

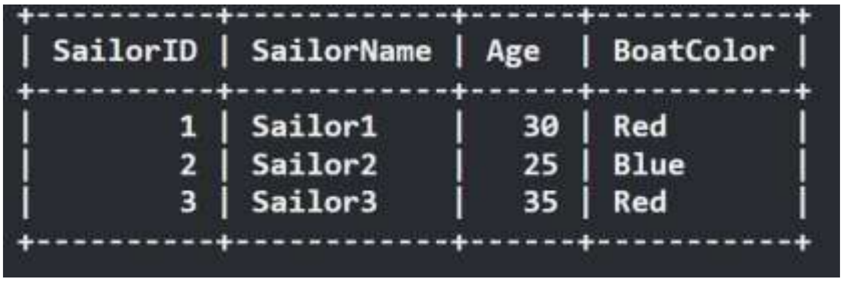


```
+----------+------------+------+
| SailorID | SailorName | Age  |
+----------+------------+------+
|        4 | Sailor4    |   28 |
+----------+------------+------+
```

Display

7. the sailor name that has Red or Green Boat.

To display the sailor names that have Red or Green boats, you can use a SQL query with a JOIN operation to combine the Sailors and Boats tables and then filter for Red or Green boats. **Example:**

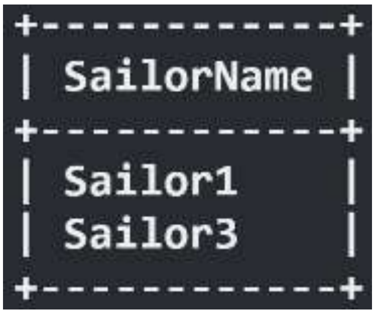


```
+----------+------------+------+-----------+
| SailorID | SailorName | Age  | BoatColor |
+----------+------------+------+-----------+
|        1 | Sailor1    |   30 | Red       |
|        2 | Sailor2    |   25 | Blue      |
|        3 | Sailor3    |   35 | Red       |
+----------+------------+------+-----------+
```

Query:-

SELECT S.SailorName
FROM Sailors S
JOIN Boats B ON S.BoatID = B.BoatID
WHERE B.BoatColor = 'Red' OR B.BoatColor = 'Green';

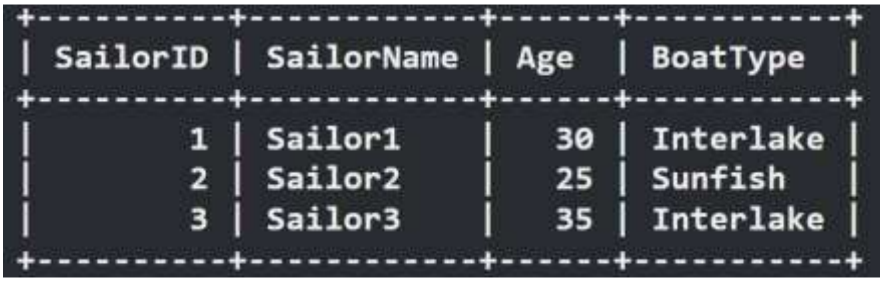
Output:-



```
+------------+
| SailorName |
+------------+
| Sailor1    |
| Sailor3    |
+------------+
```

Display

8. **all sailor details and boat details and who has an Interlake boat.** To display all sailor details and boat details for sailors who have an "Interlake" boat, you can use a SQL query that performs an inner join between the Sailors and Boats tables and filters for the boat type "Interlake." **Example:**

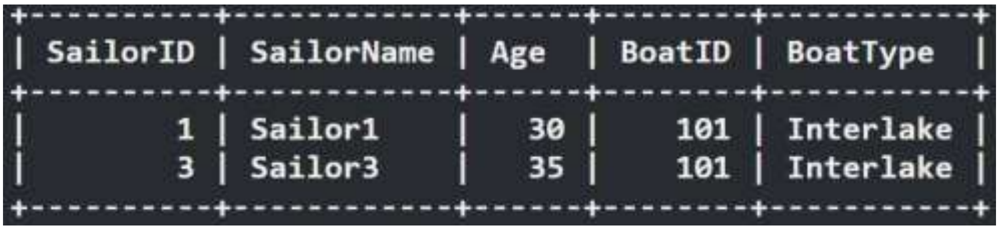


```
+----------+-----------+-----+-----------+
| SailorID | SailorName | Age | BoatType  |
+----------+-----------+-----+-----------+
|        1 | Sailor1    |  30 | Interlake |
|        2 | Sailor2    |  25 | Sunfish   |
|        3 | Sailor3    |  35 | Interlake |
+----------+-----------+-----+-----------+
```

Query:-

SELECT S.SailorID, S.SailorName, S.Age, B.BoatID, B.BoatType
FROM Sailors S
JOIN Boats B ON S.BoatID = B.BoatID
WHERE B.BoatType = 'Interlake';

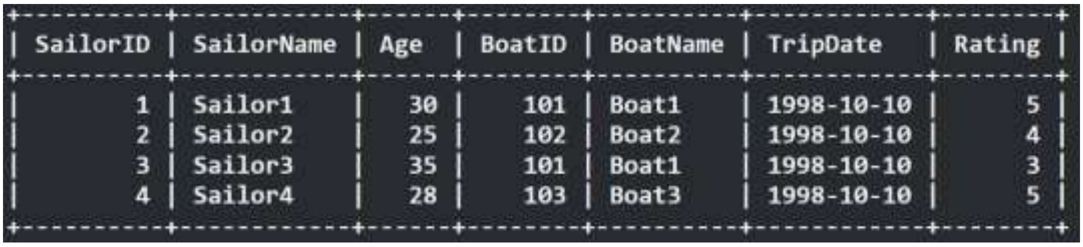
Output:-



```
+----------+------------+-----+--------+-----------+
| SailorID | SailorName | Age | BoatID | BoatType  |
+----------+------------+-----+--------+-----------+
|        1 | Sailor1    |  30 |    101 | Interlake |
|        3 | Sailor3    |  35 |    101 | Interlake |
+----------+------------+-----+--------+-----------+
```

Display

9. sailors rating with boat details or the trip on 10th October, 98. To display the sailors' ratings along with boat details for the trip on 10th October 1998, you would typically need a table that stores information about trips, including the date, boat, and sailor ratings. However, as your provided tables do not include trip information. **Example:**



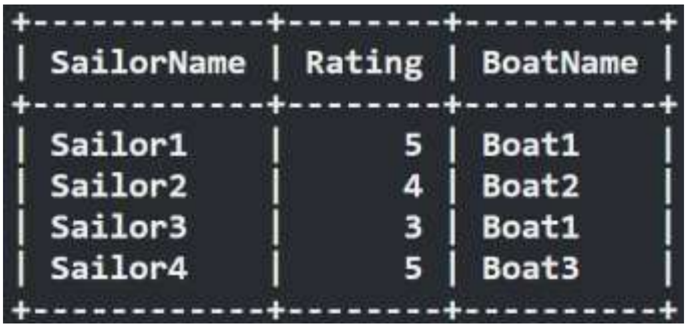
```
+----------+------------+-----+--------+----------+------------+--------+
| SailorID | SailorName | Age | BoatID | BoatName | TripDate   | Rating |
+----------+------------+-----+--------+----------+------------+--------+
|        1 | Sailor1    |  30 |    101 | Boat1    | 1998-10-10 |      5 |
|        2 | Sailor2    |  25 |    102 | Boat2    | 1998-10-10 |      4 |
|        3 | Sailor3    |  35 |    101 | Boat1    | 1998-10-10 |      3 |
|        4 | Sailor4    |  28 |    103 | Boat3    | 1998-10-10 |      5 |
+----------+------------+-----+--------+----------+------------+--------+
```

Query:-

```
SELECT S.SailorName, T.Rating, B.BoatName
FROM Sailors S
JOIN Trips T ON S.SailorID = T.SailorID
JOIN Boats B ON T.BoatID = B.BoatID
WHERE T.TripDate = '1998-10-10';
```

Output:-

10. the sailor id and name whose age is more than 42 or who has a

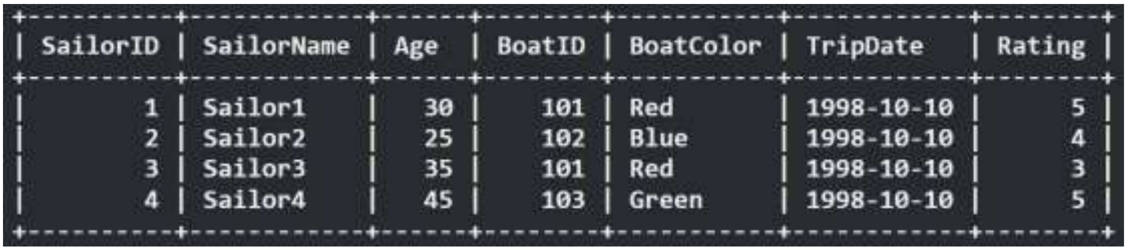


```
+------------+--------+----------+
| SailorName | Rating | BoatName |
+------------+--------+----------+
| Sailor1    |      5 | Boat1    |
| Sailor2    |      4 | Boat2    |
| Sailor3    |      3 | Boat1    |
| Sailor4    |      5 | Boat3    |
+------------+--------+----------+
```

Display

Blue colored boat.

To display the sailor ID and name for sailors whose age is more than 42 or who have a Blue colored boat, you can use a SQL query with the OR operator to combine the two conditions. **Example:**

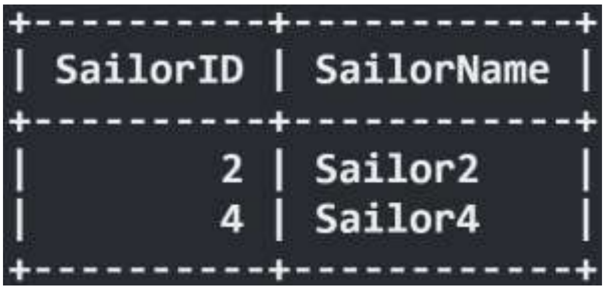


```
+----------+------------+------+--------+-----------+------------+--------+
| SailorID | SailorName | Age  | BoatID | BoatColor | TripDate   | Rating |
+----------+------------+------+--------+-----------+------------+--------+
|        1 | Sailor1    |   30 |    101 | Red       | 1998-10-10 |      5 |
|        2 | Sailor2    |   25 |    102 | Blue      | 1998-10-10 |      4 |
|        3 | Sailor3    |   35 |    101 | Red       | 1998-10-10 |      3 |
|        4 | Sailor4    |   45 |    103 | Green     | 1998-10-10 |      5 |
+----------+------------+------+--------+-----------+------------+--------+
```

Query:-

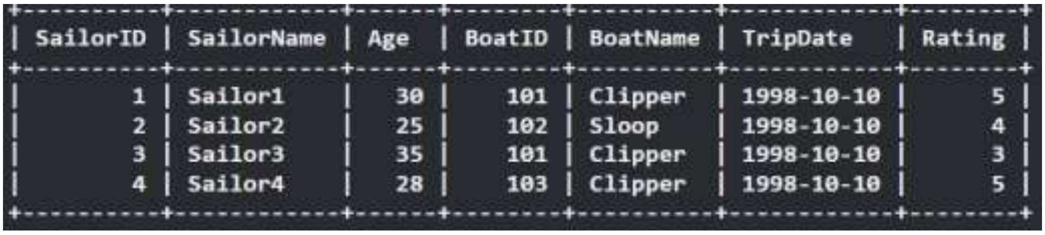
SELECT SailorID, SailorName
FROM Sailors
WHERE Age > 42 OR BoatID IN (SELECT BoatID FROM Boats WHERE BoatColor = 'Blue');

Output:-



```
+----------+------------+
| SailorID | SailorName |
+----------+------------+
|        2 | Sailor2    |
|        4 | Sailor4    |
+----------+------------+
```

11. Display name and rating of sailor whose boat name is Clipper. To display the name and rating of sailors whose boat's name is "Clipper," you can use a SQL query with a JOIN operation to combine the Sailors and Boats tables based on the boat name. **Example:**

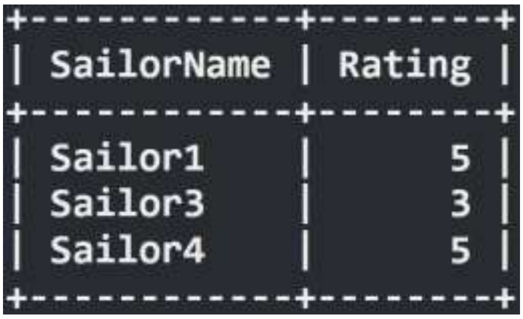


```
+----------+------------+-----+--------+----------+------------+--------+
| SailorID | SailorName | Age | BoatID | BoatName | TripDate   | Rating |
+----------+------------+-----+--------+----------+------------+--------+
|        1 | Sailor1    |  30 |    101 | Clipper  | 1998-10-10 |      5 |
|        2 | Sailor2    |  25 |    102 | Sloop    | 1998-10-10 |      4 |
|        3 | Sailor3    |  35 |    101 | Clipper  | 1998-10-10 |      3 |
|        4 | Sailor4    |  28 |    103 | Clipper  | 1998-10-10 |      5 |
+----------+------------+-----+--------+----------+------------+--------+
```

Query:-

SELECT S.SailorID, S.SailorName, S.Age, B.BoatID, B.BoatName, T.TripDate, T.Rating
FROM Sailors S
JOIN Trips T ON S.SailorID = T.SailorID
JOIN Boats B ON T.BoatID = B.BoatID;

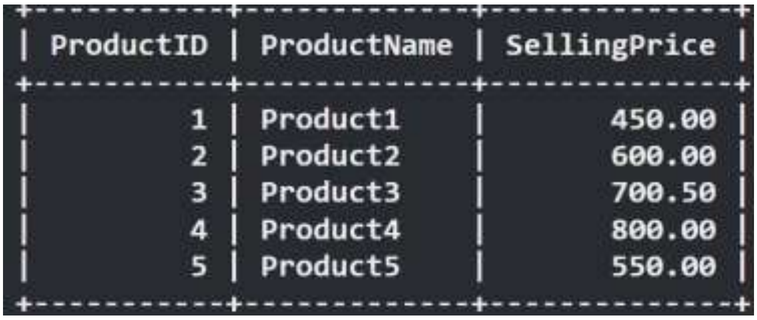
Output:-



```
+------------+--------+
| SailorName | Rating |
+------------+--------+
| Sailor1    |      5 |
| Sailor3    |      3 |
| Sailor4    |      5 |
+------------+--------+
```

12. List products whose selling price is more than 500 and less than equal to 750.

To list products whose selling price is more than 500 and less than or equal to 750, you can use a SQL query with a WHERE clause to filter the products based on their selling prices. Assuming you have a table named Products with a column named SellingPrice. **Example:**

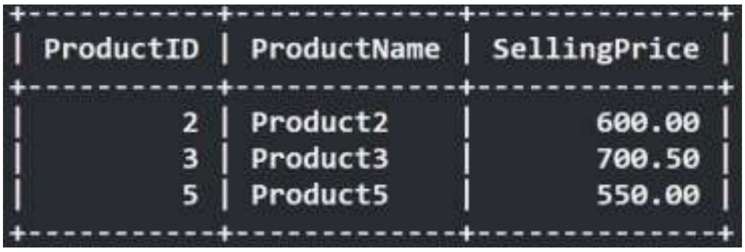


```
+-----------+-------------+--------------+
| ProductID | ProductName | SellingPrice |
+-----------+-------------+--------------+
|         1 | Product1    |       450.00 |
|         2 | Product2    |       600.00 |
|         3 | Product3    |       700.50 |
|         4 | Product4    |       800.00 |
|         5 | Product5    |       550.00 |
+-----------+-------------+--------------+
```

Query:-

SELECT *
FROM Products
WHERE SellingPrice > 500 AND SellingPrice <= 750;

Output:-

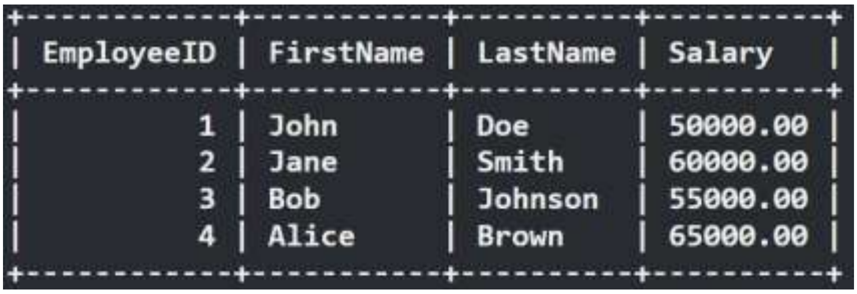


```
+-----------+-------------+--------------+
| ProductID | ProductName | SellingPrice |
+-----------+-------------+--------------+
|         2 | Product2    |       600.00 |
|         3 | Product3    |       700.50 |
|         5 | Product5    |       550.00 |
+-----------+-------------+--------------+
```

13. Describe the second highest salary of an employee.

To find the second highest salary of an employee in a database, you can use a SQL query that involves ordering the salaries in descending order and then selecting the salary value that comes second in the ordered list. The exact query may vary depending on the database system you are using.

Example:

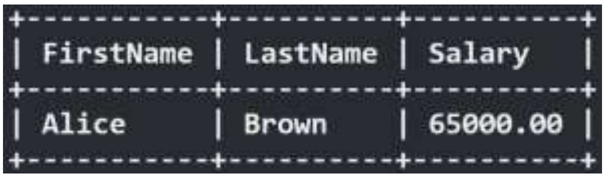


```
+------------+-----------+----------+----------+
| EmployeeID | FirstName | LastName | Salary   |
+------------+-----------+----------+----------+
|          1 | John      | Doe      | 50000.00 |
|          2 | Jane      | Smith    | 60000.00 |
|          3 | Bob       | Johnson  | 55000.00 |
|          4 | Alice     | Brown    | 65000.00 |
+------------+-----------+----------+----------+
```

Query:-

SELECT FirstName, LastName, Salary
FROM Employees
WHERE Salary = (SELECT MAX(Salary) FROM Employees);

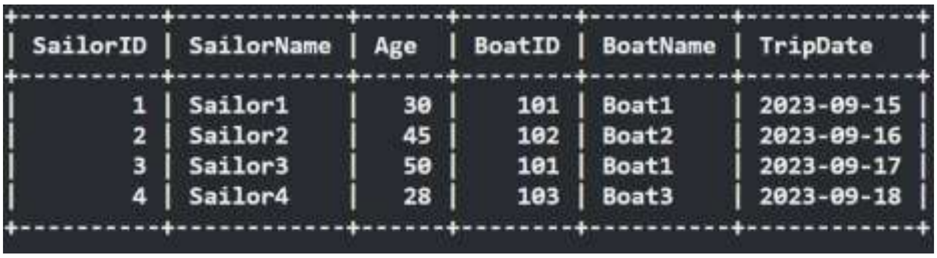
Output:-



```
+-----------+----------+----------+
| FirstName | LastName | Salary   |
+-----------+----------+----------+
| Alice     | Brown    | 65000.00 |
+-----------+----------+----------+
```

14. Display the date of travel and sailor’s name whose age is between 35 and 65.

To display the date of travel and sailor's name whose age is between 35 and 65, you can use a SQL query that joins the Sailors and Trips tables based on the SailorID and then applies a filter on the sailor's age. **Example:**

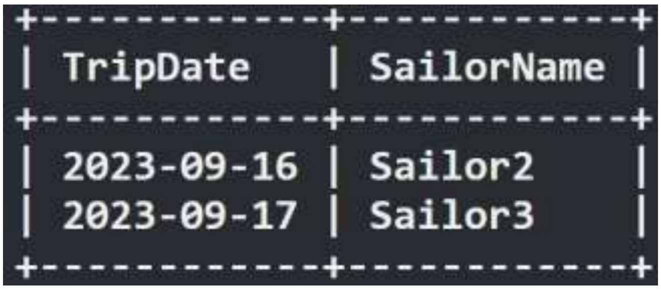


```
+----------+------------+------+--------+----------+------------+
| SailorID | SailorName | Age  | BoatID | BoatName | TripDate   |
+----------+------------+------+--------+----------+------------+
|        1 | Sailor1    |   30 |    101 | Boat1    | 2023-09-15 |
|        2 | Sailor2    |   45 |    102 | Boat2    | 2023-09-16 |
|        3 | Sailor3    |   50 |    101 | Boat1    | 2023-09-17 |
|        4 | Sailor4    |   28 |    103 | Boat3    | 2023-09-18 |
+----------+------------+------+--------+----------+------------+
```

Query:-

SELECT T.TripDate, S.SailorName
FROM Trips T
JOIN Sailors S ON T.SailorID = S.SailorID
WHERE S.Age BETWEEN 35 AND 65;

Output:-



```
+------------+------------+
| TripDate   | SailorName |
+------------+------------+
| 2023-09-16 | Sailor2    |
| 2023-09-17 | Sailor3    |
+------------+------------+
```

15. List all the employees working for ‘FBC’

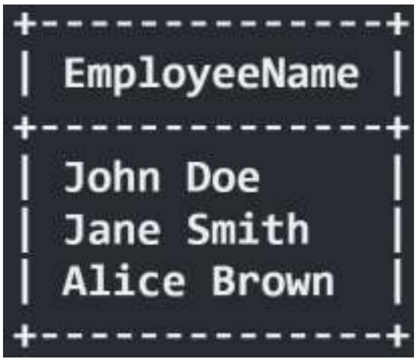
To list all the employees working for the company 'FBC', you would typically need a table that associates employees with their respective companies. Assuming you have a table named Employees that contains information about employees and the company they work for. **Example:**

Query:-

SELECT EmployeeName
FROM Employees

WHERE Company = 'FBC';

Output:-



```
+--------------+
| EmployeeName |
+--------------+
| John Doe     |
| Jane Smith   |
| Alice Brown  |
+--------------+
```

EXPERIMENT NO: 8

AIM:-

Join Queries:

1. Display all the employee names and the city where they work.

2. Display the employee name and company 's' name having a salary more than 15000.

3. Find the average rating and age of all sailors.

4. List various products available.

5. Display the names of salesmen who have salaries more than 2850.

6. Change the cost price of Trousers to 950

7. List all the clients having 'a'as a second character in their names.

8. List all the products whose QtyonHand is less than Reorder Level.

9. Print the description and total qty sold for each product.

10. Find out all the products which have been sold to 'Ivan Bayross' 11. Find the names of all clients who have purchased Trousers.

12. Find the products and their quantities for the orders placed by client C00001 and C00002.

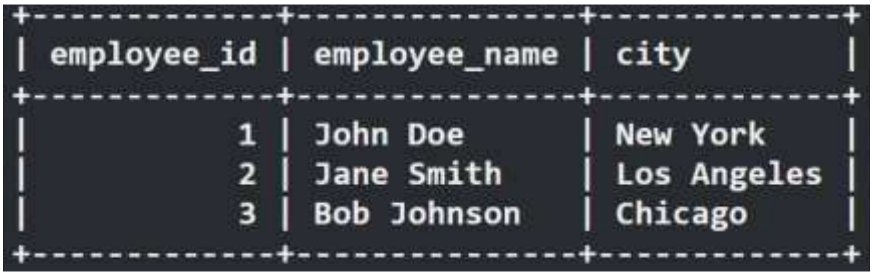
13. List the client details who place order no. O19001.

14. List the name of clients who have placed orders worth Rs. 10000 or more.

15. Find the total of Qty ordered for each Order.

1. Display all the employee names and the city where they work.

To display all the employee names and the city where they work, you would need to run a SQL query. Assuming you already have a table named "employees" with columns for employee names and cities (as mentioned in the previous response). **Example:**

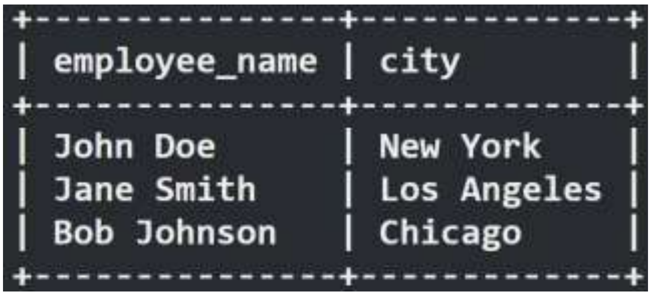


```
+-------------+---------------+-------------+
| employee_id | employee_name | city        |
+-------------+---------------+-------------+
|           1 | John Doe      | New York    |
|           2 | Jane Smith    | Los Angeles |
|           3 | Bob Johnson   | Chicago     |
+-------------+---------------+-------------+
```

Query:-

SELECT employee_name, city
FROM employees;

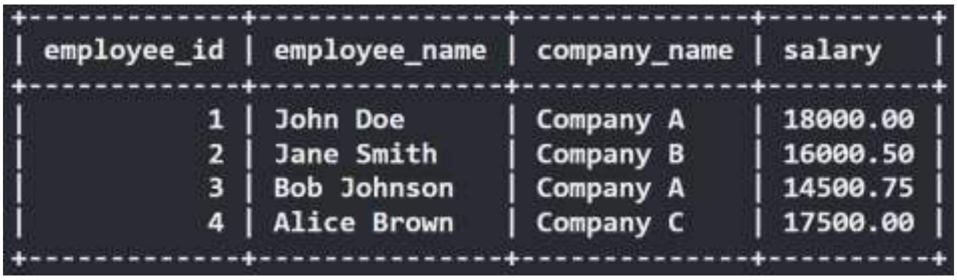
Output:-



```
+---------------+-------------+
| employee_name | city        |
+---------------+-------------+
| John Doe      | New York    |
| Jane Smith    | Los Angeles |
| Bob Johnson   | Chicago     |
+---------------+-------------+
```

2. Display the employee name and company ‘s’ name having a salary more than 15000.

To display the employee name and company name for employees with a salary greater than 15000 and whose company name contains the letter 's', **Example:**

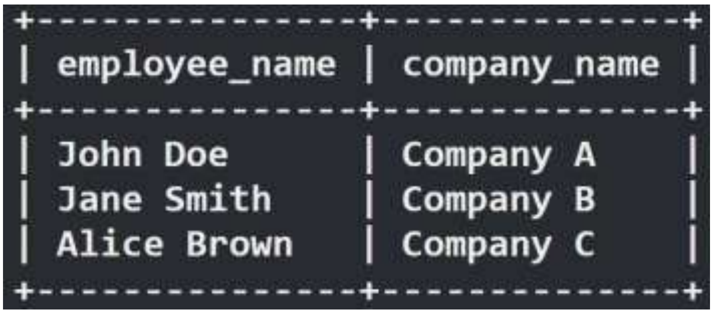


```
+-------------+---------------+--------------+----------+
| employee_id | employee_name | company_name | salary   |
+-------------+---------------+--------------+----------+
|           1 | John Doe      | Company A    | 18000.00 |
|           2 | Jane Smith    | Company B    | 16000.50 |
|           3 | Bob Johnson   | Company A    | 14500.75 |
|           4 | Alice Brown   | Company C    | 17500.00 |
+-------------+---------------+--------------+----------+
```

Query:-

SELECT employee_name, company_name
FROM employees
WHERE salary > 15000;

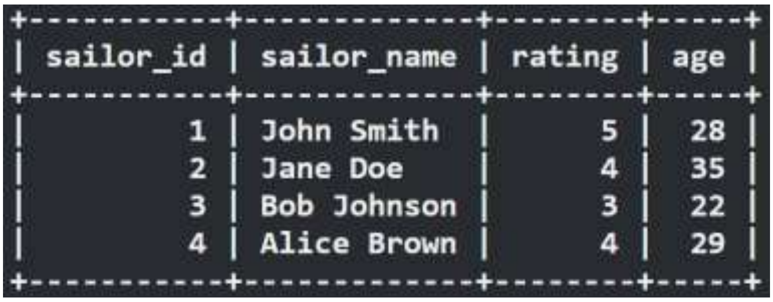
Output:-



```
+---------------+--------------+
| employee_name | company_name |
+---------------+--------------+
| John Doe      | Company A    |
| Jane Smith    | Company B    |
| Alice Brown   | Company C    |
+---------------+--------------+
```

3. Find the average rating and age of all sailors.

To find the average rating and age of all sailors. **Example:**

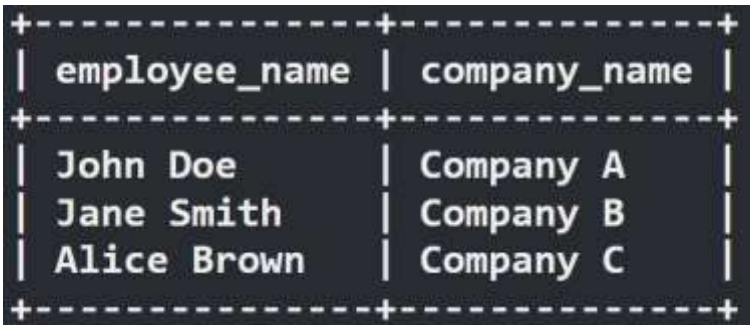


```
+-----------+-------------+--------+-----+
| sailor_id | sailor_name | rating | age |
+-----------+-------------+--------+-----+
|         1 | John Smith  |      5 |  28 |
|         2 | Jane Doe    |      4 |  35 |
|         3 | Bob Johnson |      3 |  22 |
|         4 | Alice Brown |      4 |  29 |
+-----------+-------------+--------+-----+
```

Query:-

SELECT AVG(rating) AS average_rating, AVG(age) AS average_age FROM sailors;

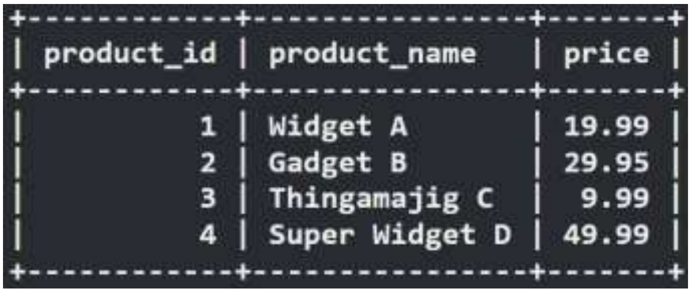
Output:-



```
+---------------+--------------+
| employee_name | company_name |
+---------------+--------------+
| John Doe      | Company A    |
| Jane Smith    | Company B    |
| Alice Brown   | Company C    |
+---------------+--------------+
```

4. List various products available.

To list various products available, you would typically need access to a database or a dataset that contains product information. **Example:**

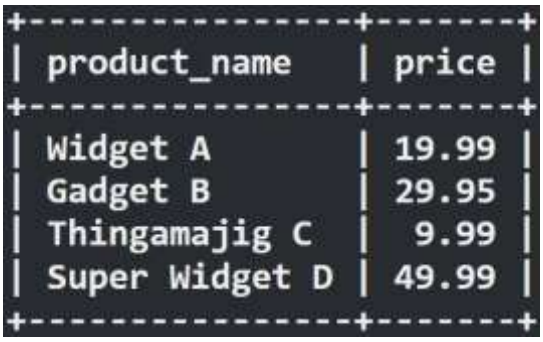


```
+------------+-----------------+-------+
| product_id | product_name    | price |
+------------+-----------------+-------+
|          1 | Widget A        | 19.99 |
|          2 | Gadget B        | 29.95 |
|          3 | Thingamajig C   |  9.99 |
|          4 | Super Widget D  | 49.99 |
+------------+-----------------+-------+
```

Query:-

SELECT product_name, description, price
FROM products;

Output:-

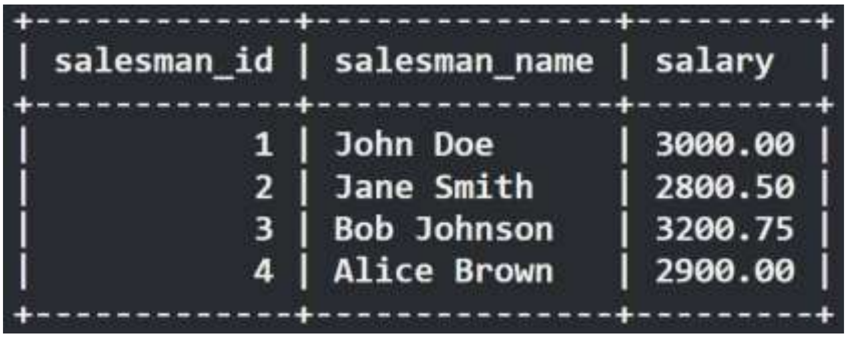


```
+-----------------+-------+
| product_name    | price |
+-----------------+-------+
| Widget A        | 19.99 |
| Gadget B        | 29.95 |
| Thingamajig C   |  9.99 |
| Super Widget D  | 49.99 |
+-----------------+-------+
```

5. Display the names of salesmen who have salaries more than 2850.

To display the names of salesmen who have salaries greater than 2850.

Example:-

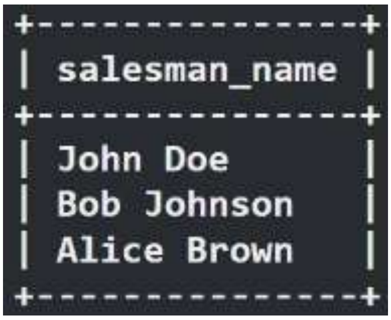


```
+-------------+---------------+---------+
| salesman_id | salesman_name | salary  |
+-------------+---------------+---------+
|           1 | John Doe      | 3000.00 |
|           2 | Jane Smith    | 2800.50 |
|           3 | Bob Johnson   | 3200.75 |
|           4 | Alice Brown   | 2900.00 |
+-------------+---------------+---------+
```

Query:-

```
SELECT salesman_name
FROM salesmen
WHERE salary > 2850;
```

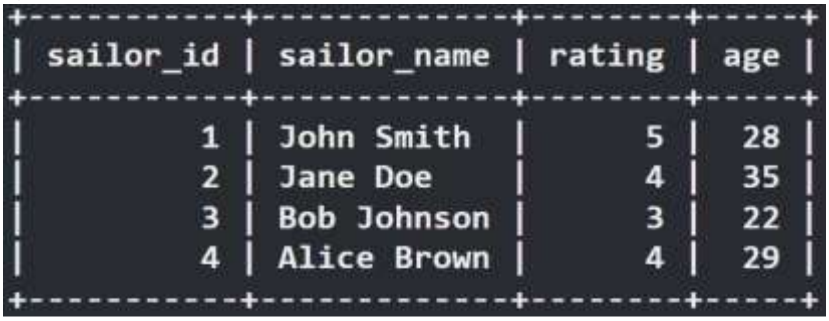
Output:-



```
+---------------+
| salesman_name |
+---------------+
| John Doe      |
| Bob Johnson   |
| Alice Brown   |
+---------------+
```

6. Change the cost price of Trousers to 950

To calculate the average rating and age of all sailors,This query calculates the average rating and average age of all sailors in the "sailors" table and aliases the result columns as "average_rating" and "average_age" for clarity. **Example:**

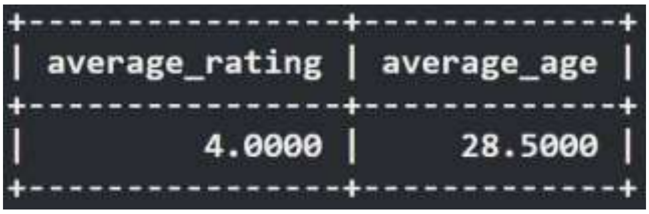


```
+-----------+-------------+--------+-----+
| sailor_id | sailor_name | rating | age |
+-----------+-------------+--------+-----+
|         1 | John Smith  |      5 |  28 |
|         2 | Jane Doe    |      4 |  35 |
|         3 | Bob Johnson |      3 |  22 |
|         4 | Alice Brown |      4 |  29 |
+-----------+-------------+--------+-----+
```

Query:-

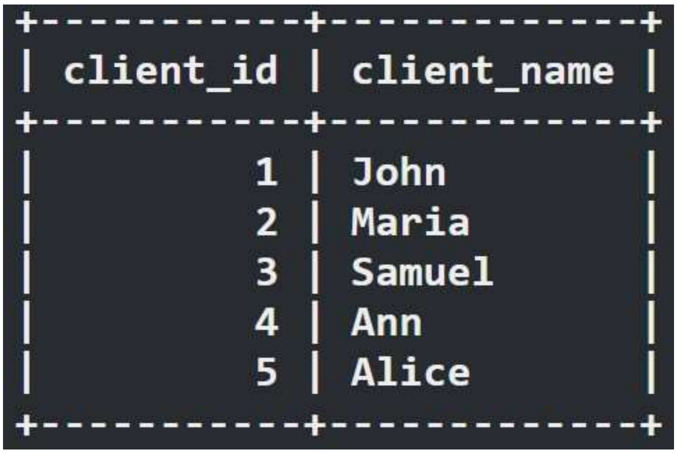
SELECT AVG(rating) AS average_rating, AVG(age) AS average_age FROM sailors;

Output:-



```
+----------------+-------------+
| average_rating | average_age |
+----------------+-------------+
|         4.0000 |     28.5000 |
+----------------+-------------+
```

7. List all the clients having ‘a’as a second character in their names. To list all the clients whose names have 'a' as the second character, this query retrieves the client names from the "clients" table where the second character in the name is 'a'. **Example:**



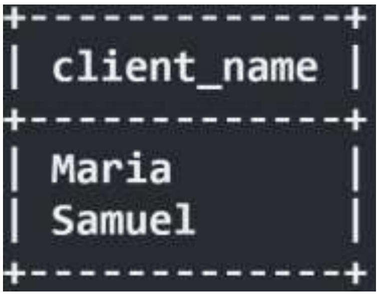
```
+-----------+-------------+
| client_id | client_name |
+-----------+-------------+
|         1 | John        |
|         2 | Maria       |
|         3 | Samuel      |
|         4 | Ann         |
|         5 | Alice       |
+-----------+-------------+
```

Query:-

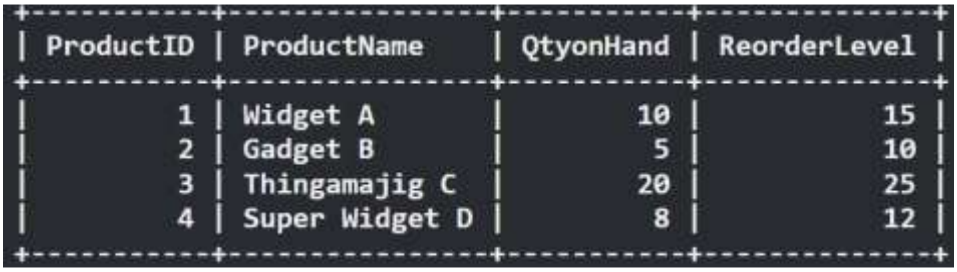
SELECT client_name
FROM clients
WHERE SUBSTRING(client_name, 2, 1) = 'a';

Output:-

8. List all the products whose QtyonHand is less than Reorder Level. To list all the products whose "QtyonHand" is less than the "Reorder Level",This query retrieves the product names from the "products" table where the quantity on hand is less than the reorder level. **Example:**



```
+-------------+
| client_name |
+-------------+
| Maria       |
| Samuel      |
+-------------+
```

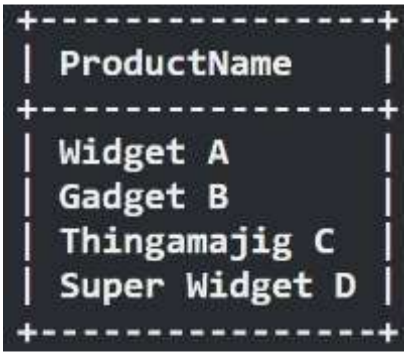


```
+-----------+----------------+-----------+--------------+
| ProductID | ProductName    | QtyonHand | ReorderLevel |
+-----------+----------------+-----------+--------------+
|         1 | Widget A       |        10 |           15 |
|         2 | Gadget B       |         5 |           10 |
|         3 | Thingamajig C  |        20 |           25 |
|         4 | Super Widget D |         8 |           12 |
+-----------+----------------+-----------+--------------+
```

Query:-

SELECT ProductName
FROM products
WHERE QtyonHand < ReorderLevel;

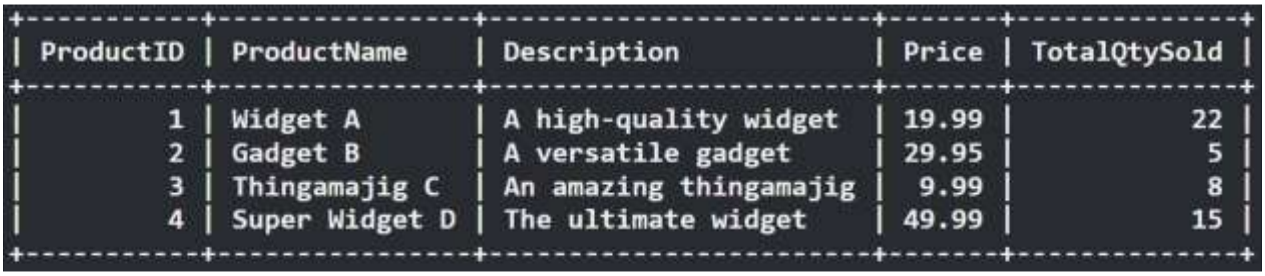
Output:-



```
+----------------+
| ProductName    |
+----------------+
| Widget A       |
| Gadget B       |
| Thingamajig C  |
| Super Widget D |
+----------------+
```

9. Print the description and total qty sold for each product.

To print the description and total quantity sold for each product,This query joins the "products" and "sales" tables based on the product ID, calculates the total quantity sold for each product, and groups the results by product name and description. **Example:**

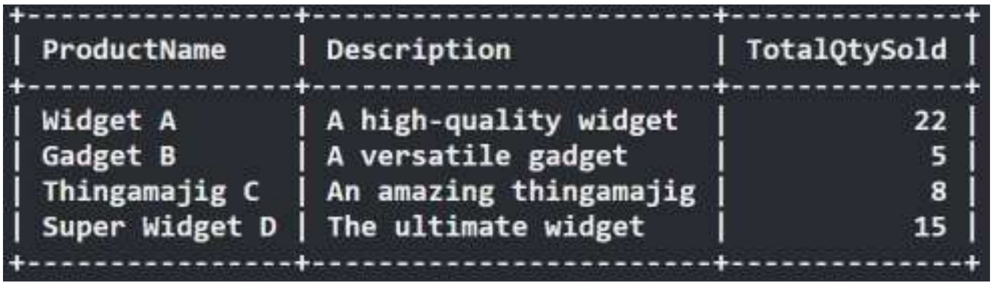


```
+-----------+----------------+-----------------------+-------+--------------+
| ProductID | ProductName    | Description           | Price | TotalQtySold |
+-----------+----------------+-----------------------+-------+--------------+
|         1 | Widget A       | A high-quality widget | 19.99 |           22 |
|         2 | Gadget B       | A versatile gadget    | 29.95 |            5 |
|         3 | Thingamajig C  | An amazing thingamajig|  9.99 |            8 |
|         4 | Super Widget D | The ultimate widget   | 49.99 |           15 |
+-----------+----------------+-----------------------+-------+--------------+
```

Query:-

SELECT p.ProductName, p.Description, SUM(s.QuantitySold) AS TotalQtySold
FROM products p
LEFT JOIN sales s ON p.ProductID = s.ProductID
GROUP BY p.ProductName, p.Description;

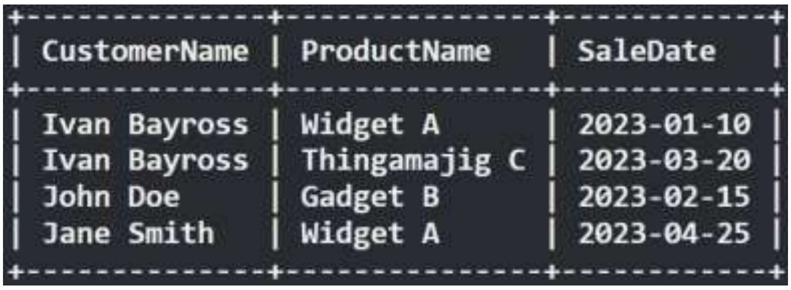
Output:-



```
+----------------+------------------------+--------------+
| ProductName    | Description            | TotalQtySold |
+----------------+------------------------+--------------+
| Widget A       | A high-quality widget  |           22 |
| Gadget B       | A versatile gadget     |            5 |
| Thingamajig C  | An amazing thingamajig |            8 |
| Super Widget D | The ultimate widget    |           15 |
+----------------+------------------------+--------------+
```

10. Find out all the products which have been sold to ‘Ivan Bayross’

To find out all the products sold to 'Ivan Bayross, this query selects the product names from the "products" table by joining the "sales" and "customers" tables based on customer and product IDs and filtering for the customer name 'Ivan Bayross.' **Example:**

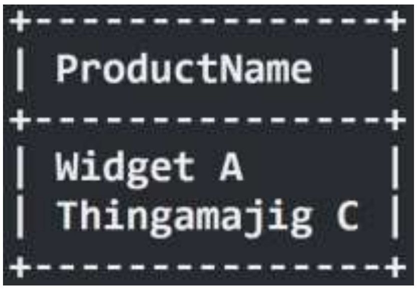


```
+--------------+----------------+------------+
| CustomerName | ProductName    | SaleDate   |
+--------------+----------------+------------+
| Ivan Bayross | Widget A       | 2023-01-10 |
| Ivan Bayross | Thingamajig C  | 2023-03-20 |
| John Doe     | Gadget B       | 2023-02-15 |
| Jane Smith   | Widget A       | 2023-04-25 |
+--------------+----------------+------------+
```

Query:-

SELECT p.ProductName
FROM products p
INNER JOIN sales s ON p.ProductID = s.ProductID
INNER JOIN customers c ON s.CustomerID = c.CustomerID
WHERE c.CustomerName = 'Ivan Bayross';

Output:-

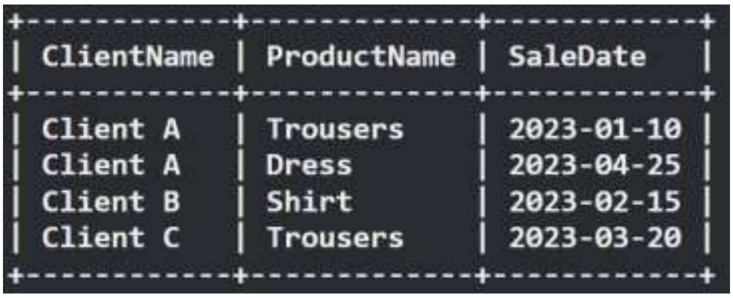


```
+---------------+
| ProductName   |
+---------------+
| Widget A      |
| Thingamajig C |
+---------------+
```

11. Find the names of all clients who have purchased Trousers.

To find the names of all clients who have purchased "Trousers," this query selects the client names (ClientName) from the "clients" table by joining the "sales" and "products" tables based on client and product IDs and filtering for the product name 'Trousers.'

Example:

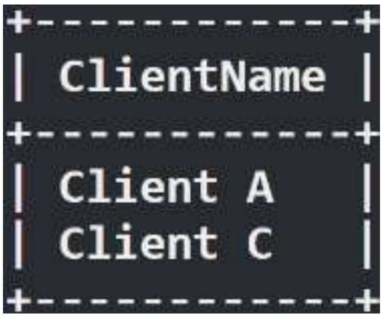


```
+------------+-------------+------------+
| ClientName | ProductName | SaleDate   |
+------------+-------------+------------+
| Client A   | Trousers    | 2023-01-10 |
| Client A   | Dress       | 2023-04-25 |
| Client B   | Shirt       | 2023-02-15 |
| Client C   | Trousers    | 2023-03-20 |
+------------+-------------+------------+
```

Query:-

SELECT c.ClientName
FROM clients c
INNER JOIN sales s ON c.ClientID = s.ClientID
INNER JOIN products p ON s.ProductID = p.ProductID
WHERE p.ProductName = 'Trousers';

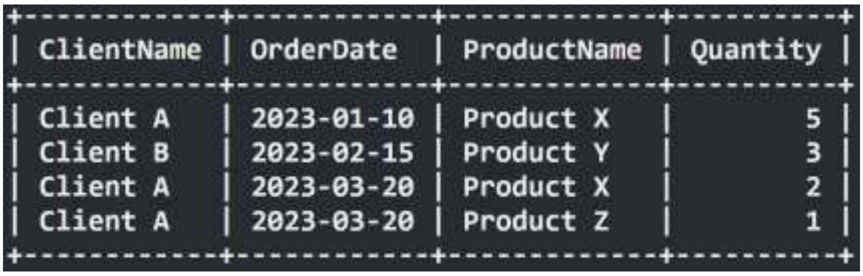
Output:-



```
+------------+
| ClientName |
+------------+
| Client A   |
| Client C   |
+------------+
```

12. Find the products and their quantities for the orders placed by client C00001 and C00002.

To find the products and their quantities for the orders placed by clients C00001 and C00002,his query selects the client name (ClientName), product name (ProductName), and quantity (Quantity) from the respective tables. It combines the tables through the relationships and filters the results to include only clients C00001 and C00002. **Example:**

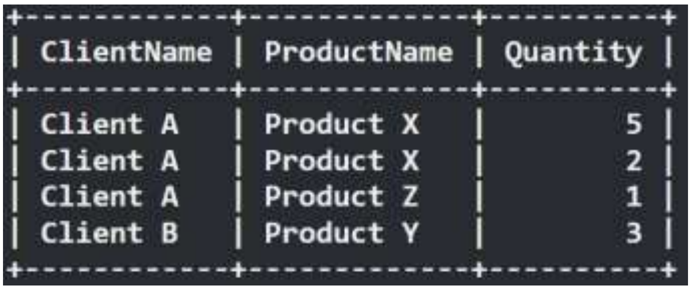


```
+------------+------------+-------------+----------+
| ClientName | OrderDate  | ProductName | Quantity |
+------------+------------+-------------+----------+
| Client A   | 2023-01-10 | Product X   |        5 |
| Client B   | 2023-02-15 | Product Y   |        3 |
| Client A   | 2023-03-20 | Product X   |        2 |
| Client A   | 2023-03-20 | Product Z   |        1 |
+------------+------------+-------------+----------+
```

Query:-

```
SELECT c.ClientName, p.ProductName, od.Quantity
FROM clients c
INNER JOIN orders o ON c.ClientID = o.ClientID
INNER JOIN order_details od ON o.OrderID = od.OrderID
INNER JOIN products p ON od.ProductID = p.ProductID
WHERE c.ClientID IN ('C00001', 'C00002');
```

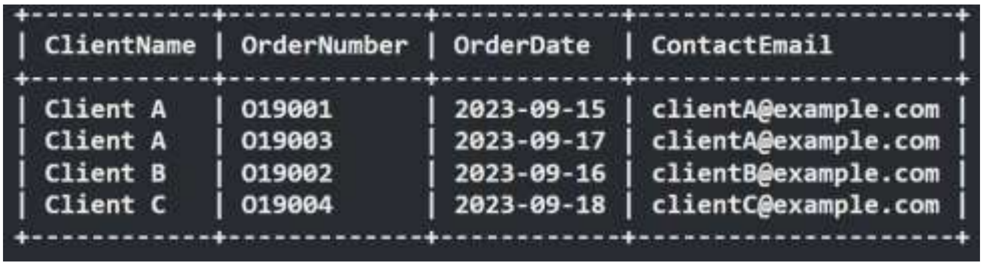
Output:-



```
+------------+-------------+----------+
| ClientName | ProductName | Quantity |
+------------+-------------+----------+
| Client A   | Product X   |        5 |
| Client A   | Product X   |        2 |
| Client A   | Product Z   |        1 |
| Client B   | Product Y   |        3 |
+------------+-------------+----------+
```

13. List the client details who place order no. O19001.

To list the client details who placed order no. O19001, this query selects the client ID (ClientID), client name (ClientName), order ID (OrderID), and order date (OrderDate) from the respective tables. It joins the "clients" and "orders" tables based on the client ID and filters the results to include only order no. O19001. **Example:**

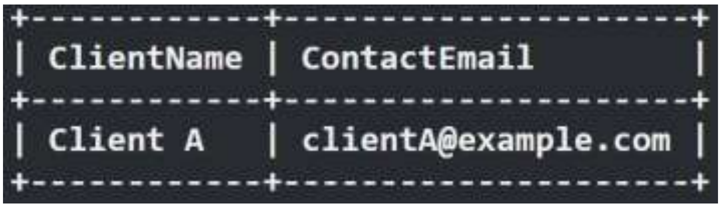


```
+------------+------------+------------+--------------------+
| ClientName | OrderNumber | OrderDate  | ContactEmail       |
+------------+------------+------------+--------------------+
| Client A   | O19001     | 2023-09-15 | clientA@example.com |
| Client A   | O19003     | 2023-09-17 | clientA@example.com |
| Client B   | O19002     | 2023-09-16 | clientB@example.com |
| Client C   | O19004     | 2023-09-18 | clientC@example.com |
+------------+------------+------------+--------------------+
```

Query:-

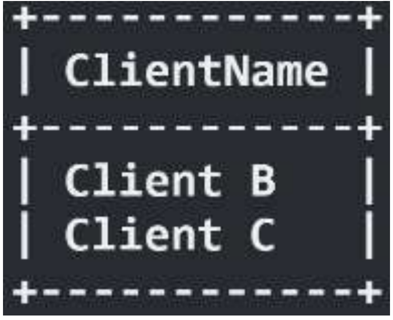
SELECT Clients.ClientName, Clients.ContactEmail
FROM Clients
JOIN Orders ON Clients.ClientID = Orders.ClientID
WHERE Orders.OrderNumber = 'O19001';

Output:-

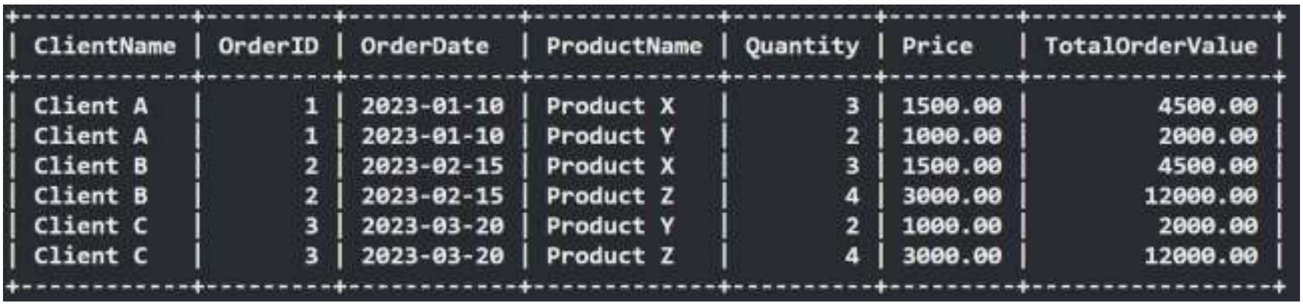


```
+------------+--------------------+
| ClientName | ContactEmail       |
+------------+--------------------+
| Client A   | clientA@example.com |
+------------+--------------------+
```

14. List the name of clients who have placed orders worth Rs. 10000 or more. To list the names of clients who have placed orders worth Rs. 10,000 or more,This query selects the client names (ClientName) from the "clients" table by joining the "orders" and "order_details" tables based on the client and order IDs. It then calculates the total order value for each client and filters the results to include only clients with orders worth Rs. 10,000 or more. **Example:**



```
+------------+
| ClientName |
+------------+
| Client B   |
| Client C   |
+------------+
```



```
+------------+---------+------------+-------------+----------+---------+-----------------+
| ClientName | OrderID | OrderDate  | ProductName | Quantity | Price   | TotalOrderValue |
+------------+---------+------------+-------------+----------+---------+-----------------+
| Client A   |       1 | 2023-01-10 | Product X   |        3 | 1500.00 |         4500.00 |
| Client A   |       1 | 2023-01-10 | Product Y   |        2 | 1000.00 |         2000.00 |
| Client B   |       2 | 2023-02-15 | Product X   |        3 | 1500.00 |         4500.00 |
| Client B   |       2 | 2023-02-15 | Product Z   |        4 | 3000.00 |        12000.00 |
| Client C   |       3 | 2023-03-20 | Product Y   |        2 | 1000.00 |         2000.00 |
| Client C   |       3 | 2023-03-20 | Product Z   |        4 | 3000.00 |        12000.00 |
+------------+---------+------------+-------------+----------+---------+-----------------+
```

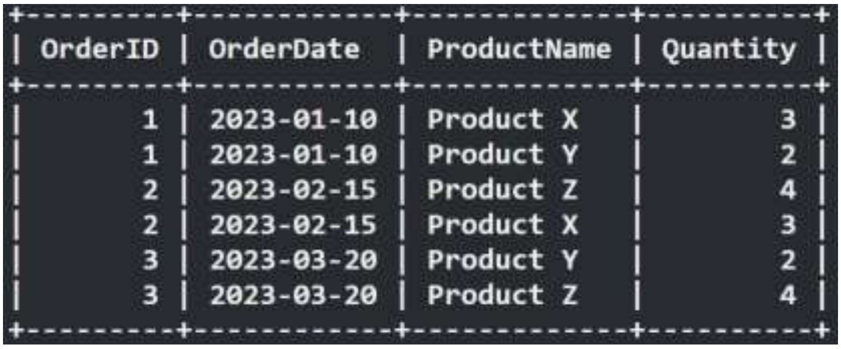
Query:-

SELECT c.ClientName
FROM clients c
INNER JOIN orders o ON c.ClientID = o.ClientID
INNER JOIN order_details od ON o.OrderID = od.OrderID
GROUP BY c.ClientName
HAVING SUM(od.Price * od.Quantity) >= 10000.00;

Output:-

15. Find the total of Qty ordered for each Order.

To find the total quantity ordered for each order, this query selects the order ID (OrderID) and calculates the total quantity ordered (TotalQuantityOrdered) by summing the quantity from the "order_details" table. It groups the results by order ID. **Example:**

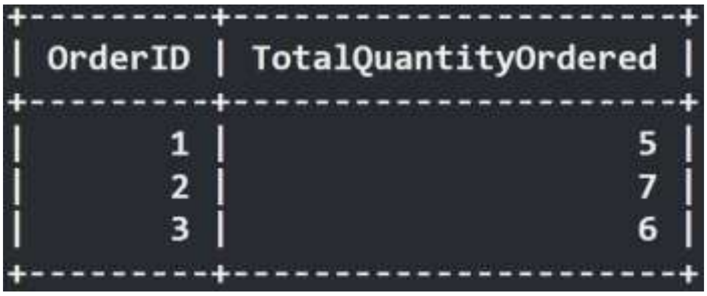


```
+---------+------------+-------------+----------+
| OrderID | OrderDate  | ProductName | Quantity |
+---------+------------+-------------+----------+
|       1 | 2023-01-10 | Product X   |        3 |
|       1 | 2023-01-10 | Product Y   |        2 |
|       2 | 2023-02-15 | Product Z   |        4 |
|       2 | 2023-02-15 | Product X   |        3 |
|       3 | 2023-03-20 | Product Y   |        2 |
|       3 | 2023-03-20 | Product Z   |        4 |
+---------+------------+-------------+----------+
```

Query:-

```
SELECT o.OrderID, SUM(od.Quantity) AS TotalQuantityOrdered
FROM orders o
INNER JOIN order_details od ON o.OrderID = od.OrderID GROUP BY
o.OrderID;
```

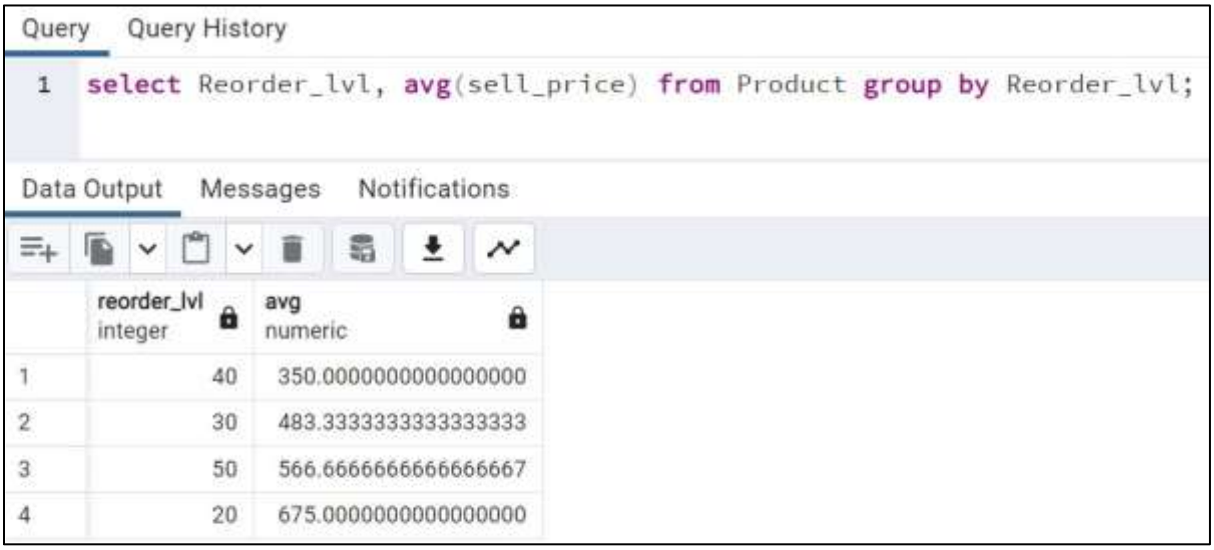
Output:-



```
+---------+----------------------+
| OrderID | TotalQuantityOrdered |
+---------+----------------------+
|       1 |                    5 |
|       2 |                    7 |
|       3 |                    6 |
+---------+----------------------+
```

PRACTICAL 9 – Miscellaneous Queries

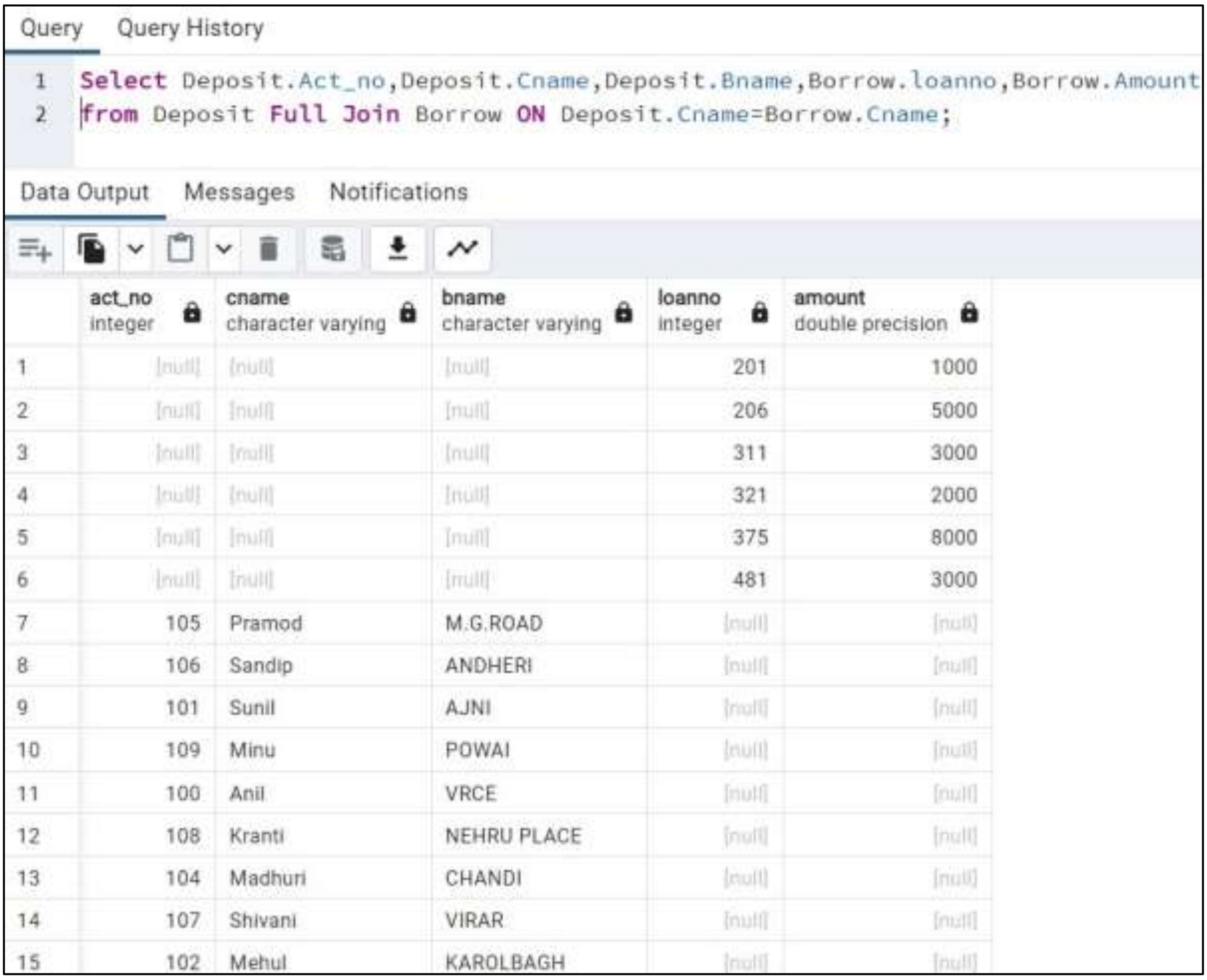
1. Find the average rate for each Order.



```
select Reorder_lvl, avg(sell_price) from Product group by Reorder_lvl;
```

	reorder_lvl integer	avg numeric
1	40	350.0000000000000000
2	30	483.3333333333333333
3	50	566.6666666666666667
4	20	675.0000000000000000

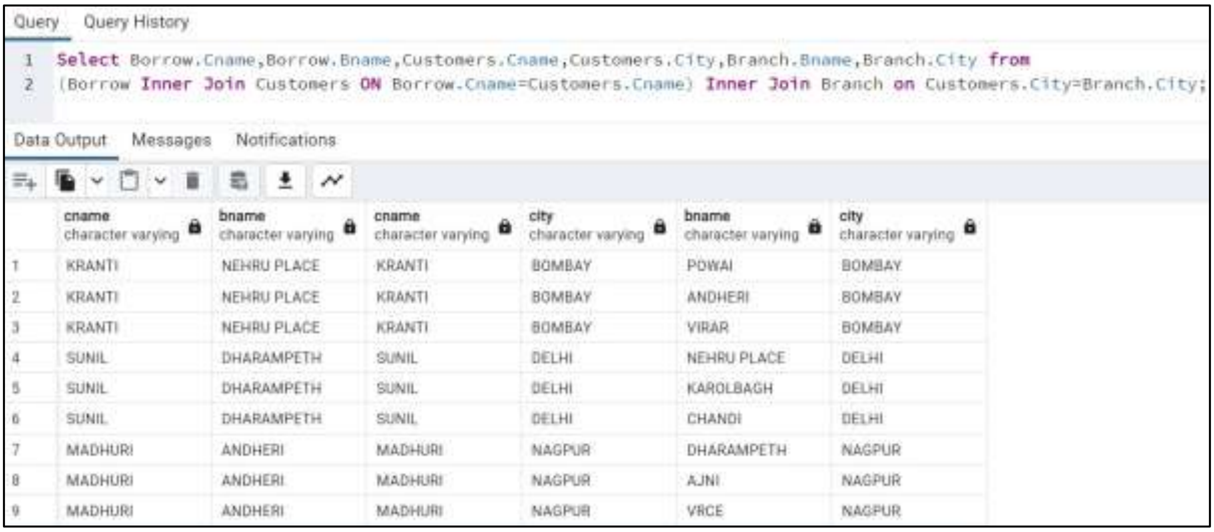
2. Give the loan details of all the customers.



```sql
Select Deposit.Act_no,Deposit.Cname,Deposit.Bname,Borrow.loanno,Borrow.Amount
from Deposit Full Join Borrow ON Deposit.Cname=Borrow.Cname;
```

	act_no integer	cname character varying	bname character varying	loanno integer	amount double precision
1	[null]	[null]	[null]	201	1000
2	[null]	[null]	[null]	206	5000
3	[null]	[null]	[null]	311	3000
4	[null]	[null]	[null]	321	2000
5	[null]	[null]	[null]	375	8000
6	[null]	[null]	[null]	481	3000
7	105	Pramod	M.G.ROAD	[null]	[null]
8	106	Sandip	ANDHERI	[null]	[null]
9	101	Sunil	AJNI	[null]	[null]
10	109	Minu	POWAI	[null]	[null]
11	100	Anil	VRCE	[null]	[null]
12	108	Kranti	NEHRU PLACE	[null]	[null]
13	104	Madhuri	CHANDI	[null]	[null]
14	107	Shivani	VIRAR	[null]	[null]
15	102	Mehul	KAROLBAGH	[null]	[null]

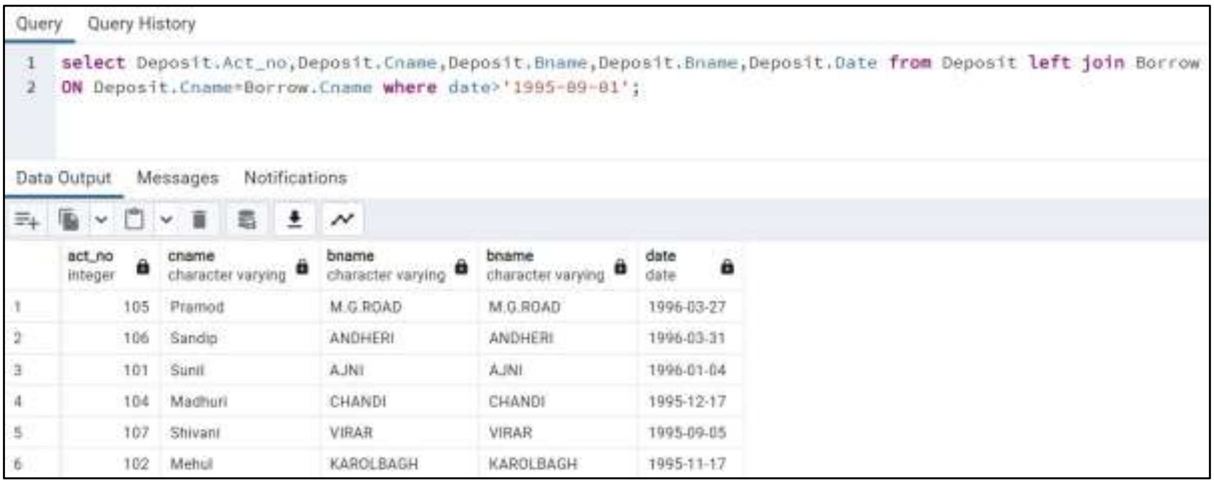
3. List the customer’s name having loan account in the same branch city they live in.



```
Select Borrow.Cname,Borrow.Bname,Customers.Cname,Customers.City,Branch.Bname,Branch.City from
(Borrow Inner Join Customers ON Borrow.Cname=Customers.Cname) Inner Join Branch on Customers.City=Branch.City;
```

	cname character varying	bname character varying	cname character varying	city character varying	bname character varying	city character varying
1	KRANTI	NEHRU PLACE	KRANTI	BOMBAY	POWAI	BOMBAY
2	KRANTI	NEHRU PLACE	KRANTI	BOMBAY	ANDHERI	BOMBAY
3	KRANTI	NEHRU PLACE	KRANTI	BOMBAY	VIRAR	BOMBAY
4	SUNIL	DHARAMPETH	SUNIL	DELHI	NEHRU PLACE	DELHI
5	SUNIL	DHARAMPETH	SUNIL	DELHI	KAROLBAGH	DELHI
6	SUNIL	DHARAMPETH	SUNIL	DELHI	CHANDI	DELHI
7	MADHURI	ANDHERI	MADHURI	NAGPUR	DHARAMPETH	NAGPUR
8	MADHURI	ANDHERI	MADHURI	NAGPUR	AJNI	NAGPUR
9	MADHURI	ANDHERI	MADHURI	NAGPUR	VRCE	NAGPUR

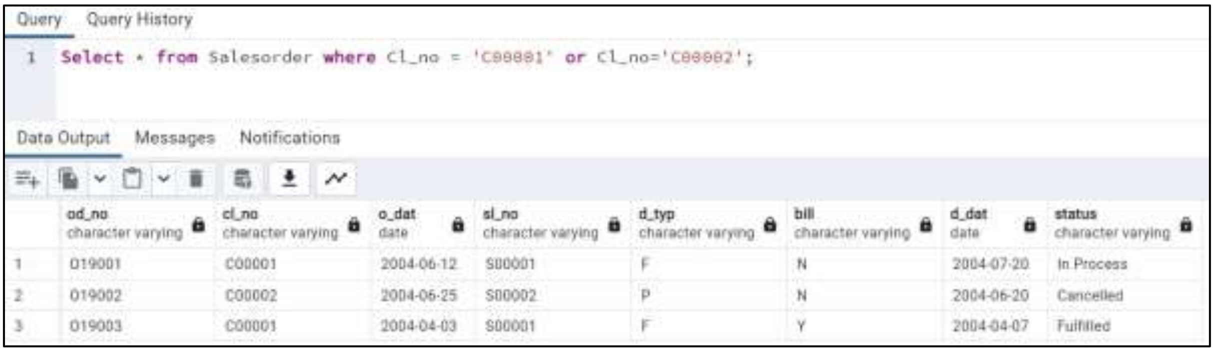
4. **Provide the loan details of all the customers who have opened their accounts after August 1995.**



```
select Deposit.Act_no,Deposit.Cname,Deposit.Bname,Deposit.Bname,Deposit.Date from Deposit left join Borrow
ON Deposit.Cname=Borrow.Cname where date>'1995-09-01';
```

	act_no integer	cname character varying	bname character varying	bname character varying	date date
1	105	Pramod	M.G.ROAD	M.G.ROAD	1996-03-27
2	106	Sandip	ANDHERI	ANDHERI	1996-03-31
3	101	Sunil	AJNI	AJNI	1996-01-04
4	104	Madhuri	CHANDI	CHANDI	1995-12-17
5	107	Shivani	VIRAR	VIRAR	1995-09-05
6	102	Mehul	KAROLBAGH	KAROLBAGH	1995-11-17

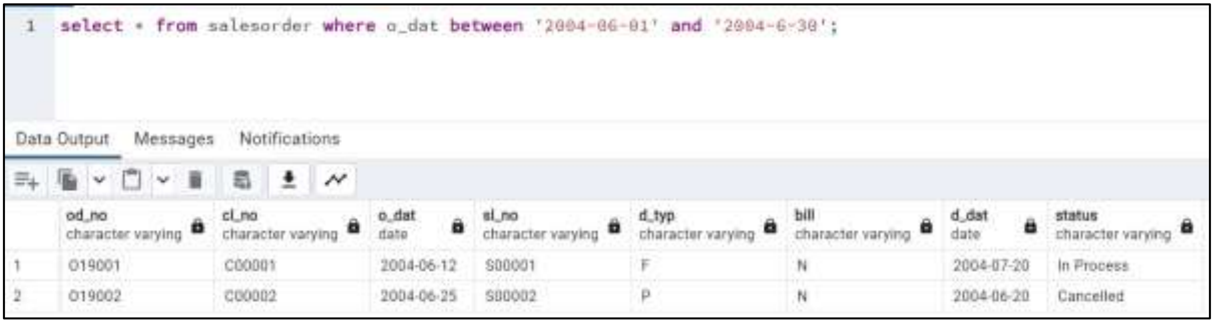
5. **List the order information for client C00001 and C00002.**



```
Select * from Salesorder where Cl_no = 'C00001' or Cl_no='C00002';
```

	od_no character varying	cl_no character varying	o_dat date	sl_no character varying	d_typ character varying	bill character varying	d_dat date	status character varying
1	019001	C00001	2004-06-12	S00001	F	N	2004-07-20	In Process
2	019002	C00002	2004-06-25	S00002	P	N	2004-06-20	Cancelled
3	019003	C00001	2004-04-03	S00001	F	Y	2004-04-07	Fulfilled

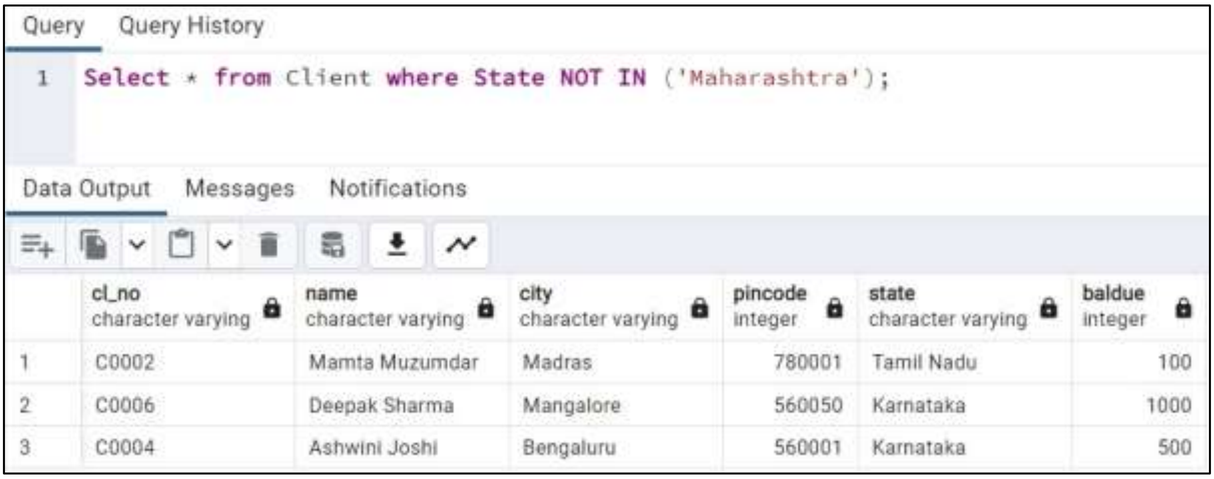
6. List all the information for the order placed in the month of June.



```
select * from salesorder where o_dat between '2004-06-01' and '2004-6-30';
```

	od_no character varying	cl_no character varying	o_dat date	sl_no character varying	d_typ character varying	bill character varying	d_dat date	status character varying
1	019001	C00001	2004-06-12	S00001	F	N	2004-07-20	In Process
2	019002	C00002	2004-06-25	S00002	P	N	2004-06-20	Cancelled

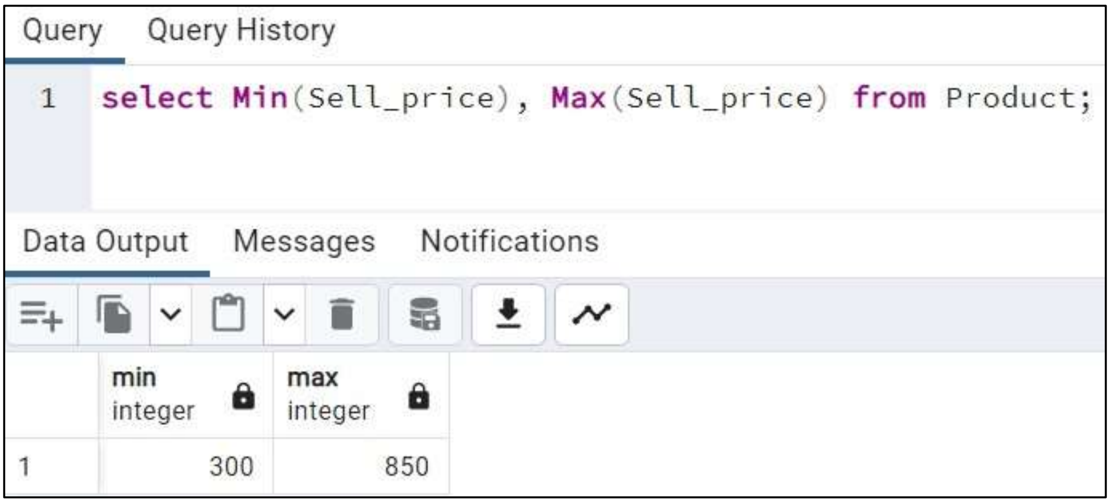
7. List the details of clients who do not stay in Maharashtra.



```
Select * from Client where State NOT IN ('Maharashtra');
```

	cl_no character varying	name character varying	city character varying	pincode integer	state character varying	baldue integer
1	C0002	Mamta Muzumdar	Madras	780001	Tamil Nadu	100
2	C0006	Deepak Sharma	Mangalore	560050	Karnataka	1000
3	C0004	Ashwini Joshi	Bengaluru	560001	Karnataka	500

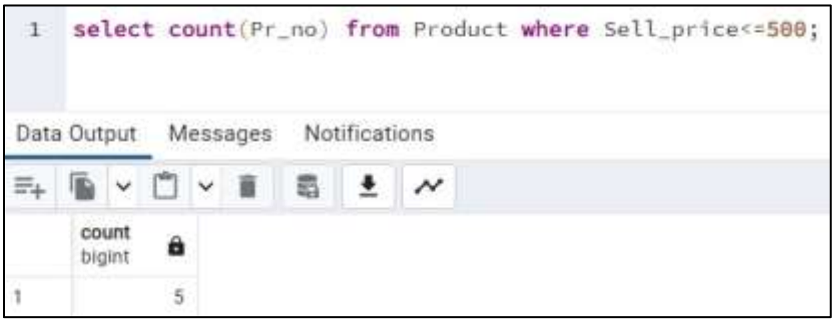
8. **Determine the maximum and minimum product price. Rename the output as Max_Price and Min_Price.**



```
select Min(Sell_price), Max(Sell_price) from Product;
```

	min integer	max integer
1	300	850

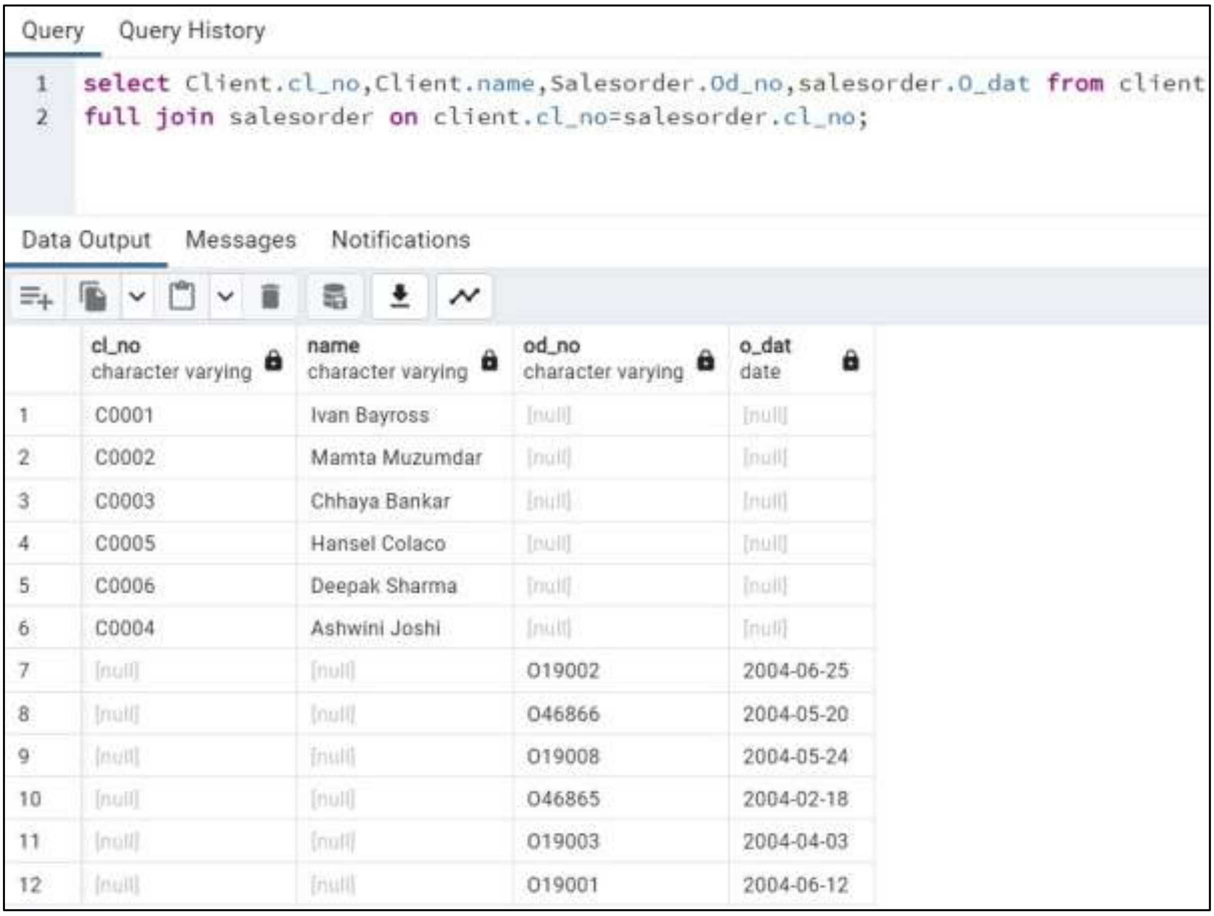
9. **Count the number of products having price less than or equal to 500.**



```
select count(Pr_no) from Product where Sell_price<=500;
```

```
   count
   bigint
1      5
```

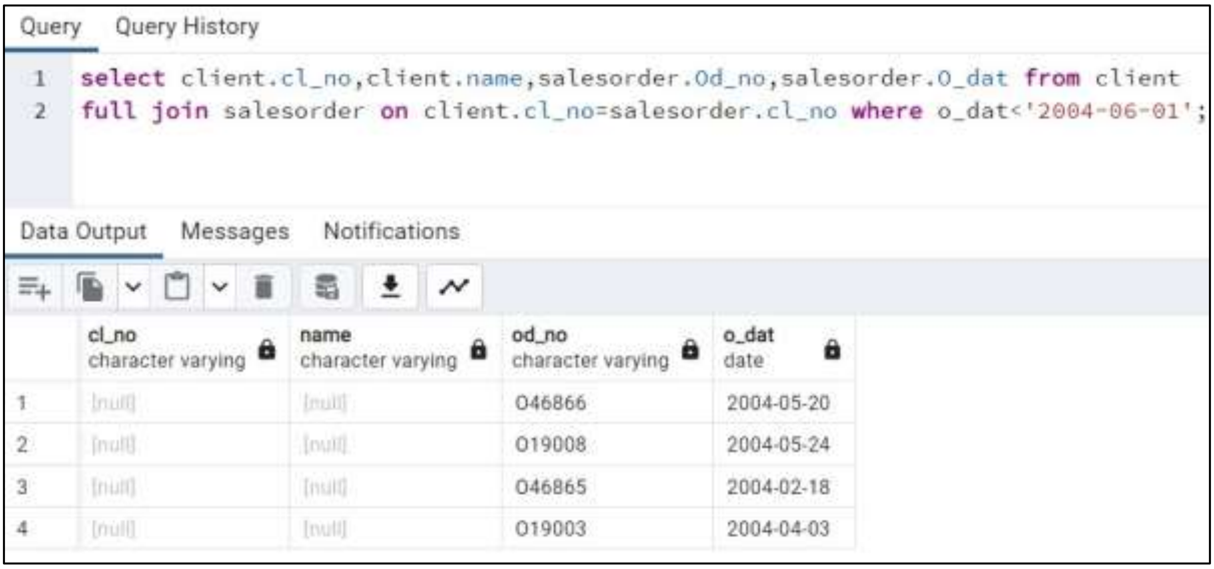
10. List the order number and the day on which client placed an order.



```
select Client.cl_no,Client.name,Salesorder.Od_no,salesorder.O_dat from client
full join salesorder on client.cl_no=salesorder.cl_no;
```

	cl_no character varying	name character varying	od_no character varying	o_dat date
1	C0001	Ivan Bayross	[null]	[null]
2	C0002	Mamta Muzumdar	[null]	[null]
3	C0003	Chhaya Bankar	[null]	[null]
4	C0005	Hansel Colaco	[null]	[null]
5	C0006	Deepak Sharma	[null]	[null]
6	C0004	Ashwini Joshi	[null]	[null]
7	[null]	[null]	O19002	2004-06-25
8	[null]	[null]	O46866	2004-05-20
9	[null]	[null]	O19008	2004-05-24
10	[null]	[null]	O46865	2004-02-18
11	[null]	[null]	O19003	2004-04-03
12	[null]	[null]	O19001	2004-06-12

11. List the clients who placed order before June.



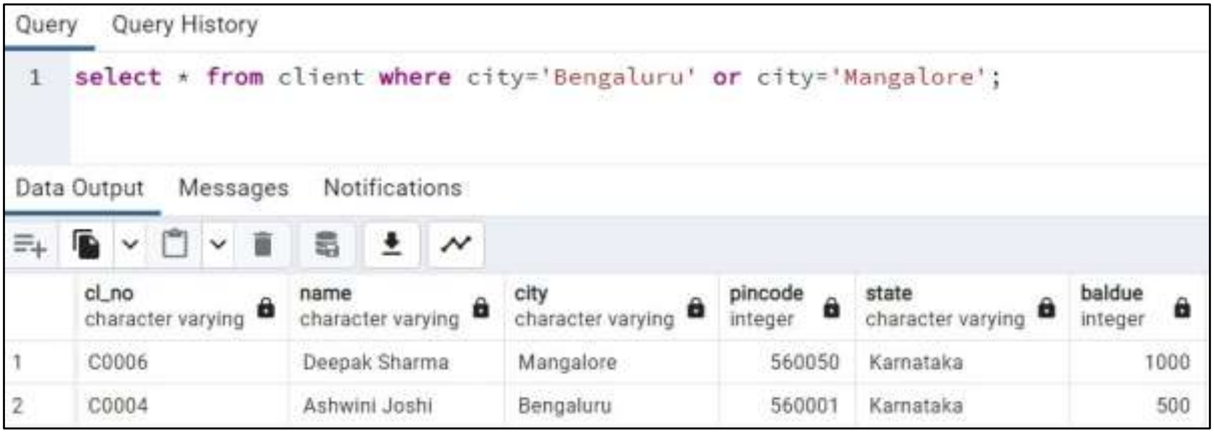
Query　Query History

```sql
select client.cl_no,client.name,salesorder.Od_no,salesorder.O_dat from client
full join salesorder on client.cl_no=salesorder.cl_no where o_dat<'2004-06-01';
```

Data Output　Messages　Notifications

	cl_no character varying	name character varying	od_no character varying	o_dat date
1	[null]	[null]	046866	2004-05-20
2	[null]	[null]	019008	2004-05-24
3	[null]	[null]	046865	2004-02-18
4	[null]	[null]	019003	2004-04-03

12. List all the clients who stays in Bengaluru or Mangalore



Query　Query History

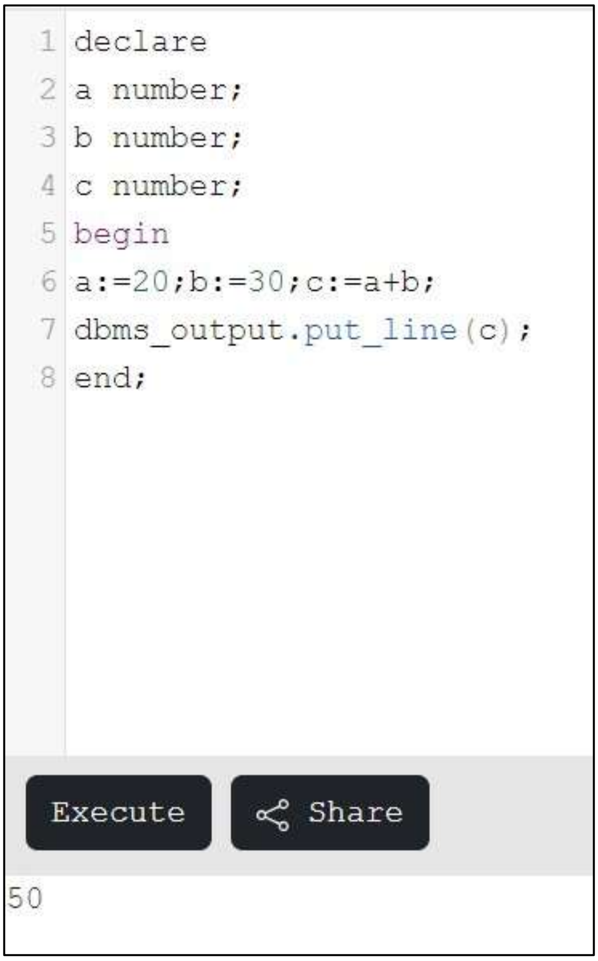
```sql
select * from client where city='Bengaluru' or city='Mangalore';
```

Data Output　Messages　Notifications

	cl_no character varying	name character varying	city character varying	pincode integer	state character varying	baldue integer
1	C0006	Deepak Sharma	Mangalore	560050	Karnataka	1000
2	C0004	Ashwini Joshi	Bengaluru	560001	Karnataka	500

PRACTICAL 10 – PL/SQL Queries

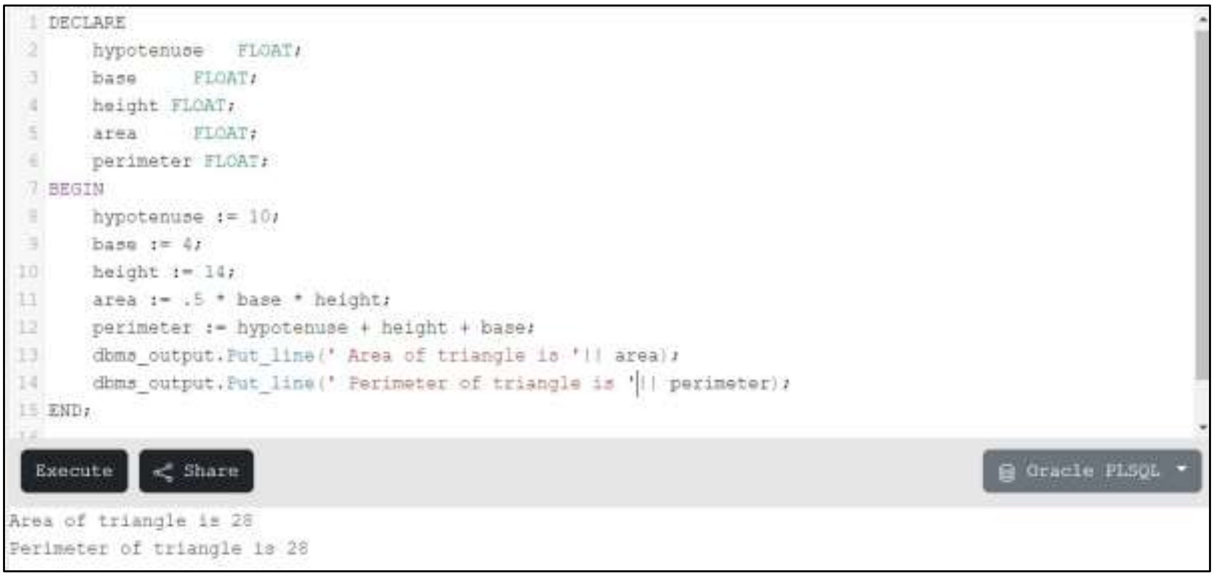
1. **Write a PL/SQL Block to Add 2 Numbers.**



```
declare
a number;
b number;
c number;
begin
a:=20;b:=30;c:=a+b;
dbms_output.put_line(c);
end;
```

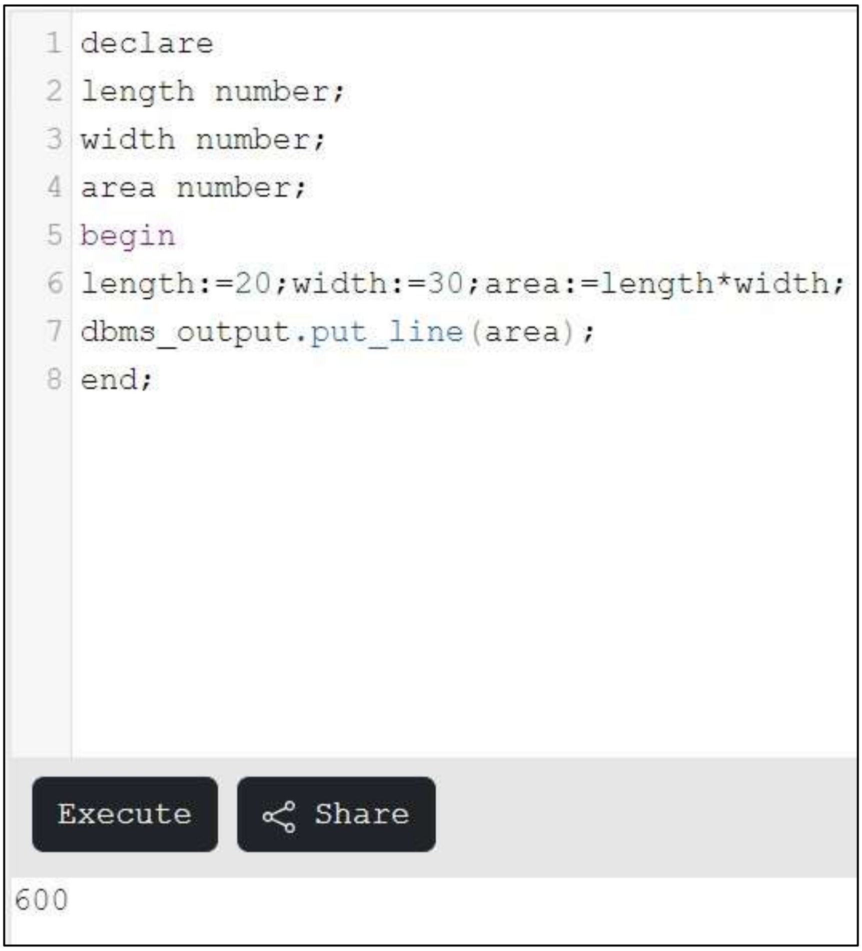
```
50
```

2. **Write a PL/SQL Block to find Area of Rectangle, Triangle and Square.**



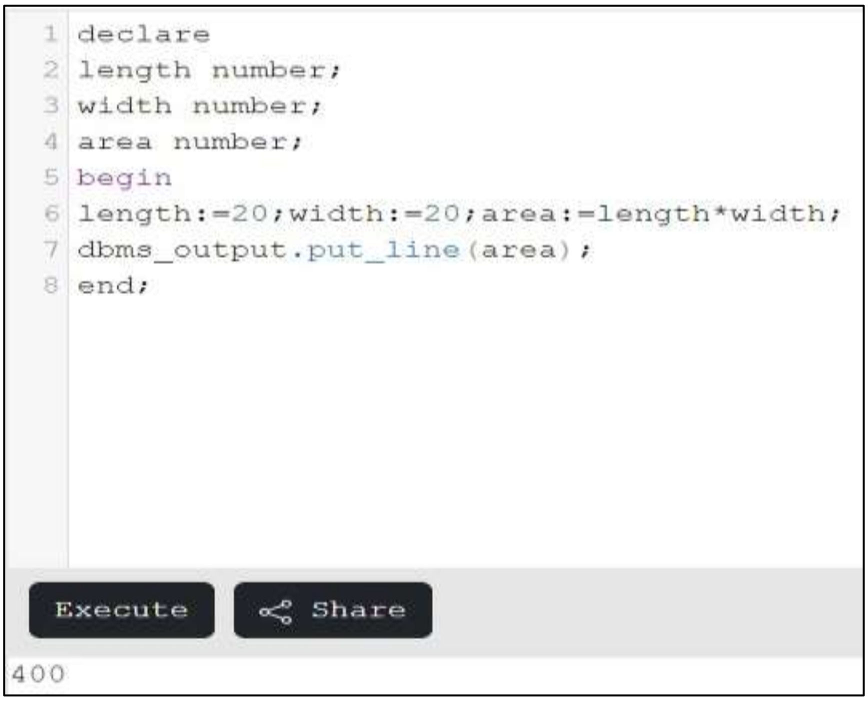
```
DECLARE
    hypotenuse   FLOAT;
    base      FLOAT;
    height FLOAT;
    area      FLOAT;
    perimeter FLOAT;
BEGIN
    hypotenuse := 10;
    base := 4;
    height := 14;
    area := .5 * base * height;
    perimeter := hypotenuse + height + base;
    dbms_output.Put_line(' Area of triangle is '|| area);
    dbms_output.Put_line(' Perimeter of triangle is '|| perimeter);
END;
```

```
Area of triangle is 28
Perimeter of triangle is 28
```



```
declare
length number;
width number;
area number;
begin
length:=20;width:=30;area:=length*width;
dbms_output.put_line(area);
end;
```

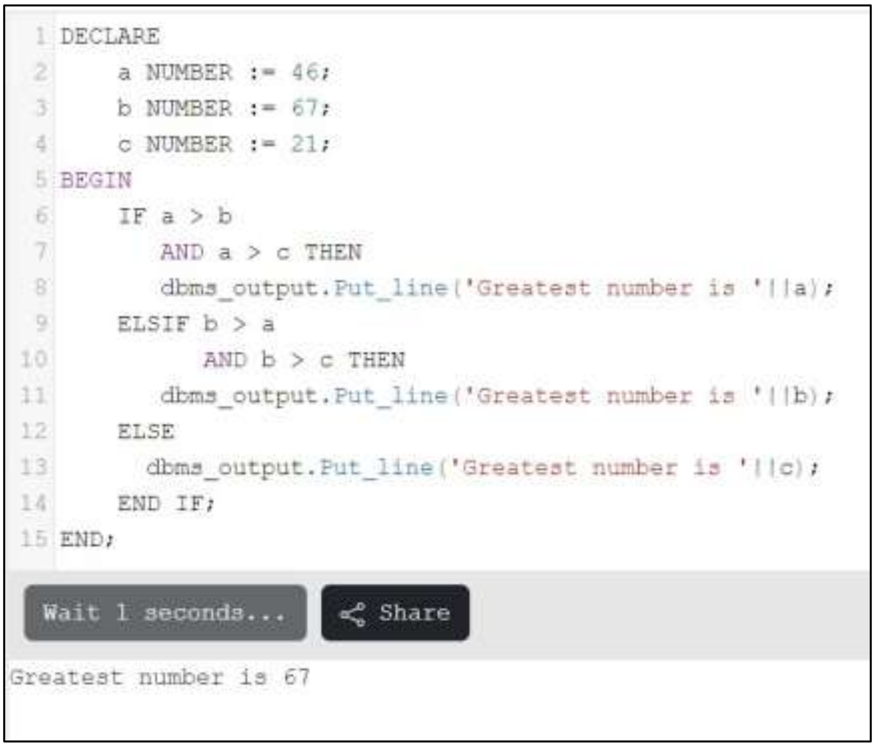
```
600
```



```
declare
length number;
width number;
area number;
begin
length:=20;width:=20;area:=length*width;
dbms_output.put_line(area);
end;
```

```
400
```

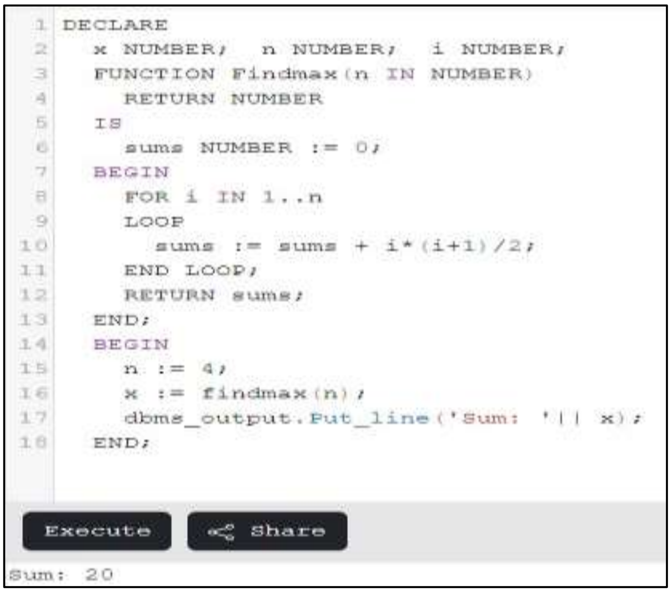
3. Write a PL/SQL Block to find Maximum of 3 numbers.



```
DECLARE
    a NUMBER := 46;
    b NUMBER := 67;
    c NUMBER := 21;
BEGIN
    IF a > b
       AND a > c THEN
       dbms_output.Put_line('Greatest number is '||a);
    ELSIF b > a
          AND b > c THEN
       dbms_output.Put_line('Greatest number is '||b);
    ELSE
      dbms_output.Put_line('Greatest number is '||c);
    END IF;
END;
```

```
Greatest number is 67
```

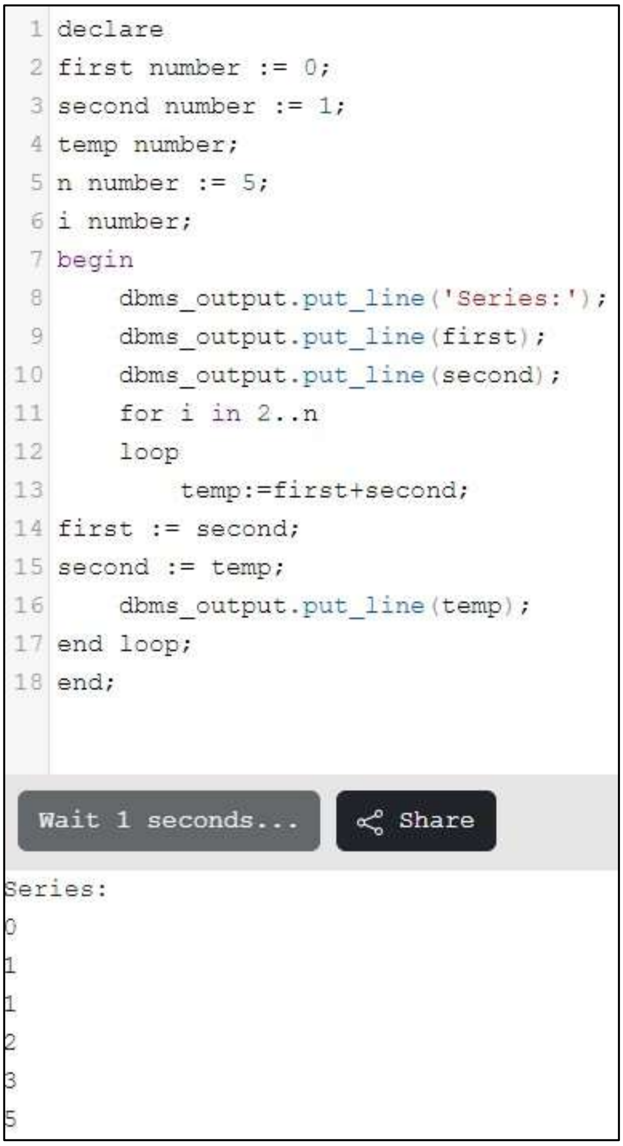
4. Write a PL/SQL Block to print sum of N Numbers using For Loop.



```
DECLARE
  x NUMBER;   n NUMBER;   i NUMBER;
  FUNCTION Findmax(n IN NUMBER)
    RETURN NUMBER
  IS
    sums NUMBER := 0;
  BEGIN
    FOR i IN 1..n
    LOOP
      sums := sums + i*(i+1)/2;
    END LOOP;
    RETURN sums;
  END;
  BEGIN
    n := 4;
    x := findmax(n);
    dbms_output.Put_line('Sum: '|| x);
  END;
```

```
Sum: 20
```

5. Write a PL/SQL Block to generate Fibonacci series of N numbers.



```
declare
first number := 0;
second number := 1;
temp number;
n number := 5;
i number;
begin
    dbms_output.put_line('Series:');
    dbms_output.put_line(first);
    dbms_output.put_line(second);
    for i in 2..n
    loop
        temp:=first+second;
first := second;
second := temp;
    dbms_output.put_line(temp);
end loop;
end;
```

```
Series:
0
1
1
2
3
5
```

++