Practical-9

Aim : Miscellaneous Queries:

1. Find the average rate for each Order.
2. Give the loan details of all the customers.
3. List the customer name having loan account in the same branch city they live in.
4. Provide the loan details of all the customers who have opened their accounts after August
5. List the order information for client C00001 and C00002.
6. List all the information for the order placed in the month of june.
7. List the details of clients who do not stay in Maharashtra.
8. Determine the maximum and minimum product price. Rename the output as “Max_Price” and “Min_Price”
9. Count the number of products having price less than or equal to 500.
10. List the order number and the day on which client placed an order.
11. List the month and the date on which an order is to be delivered.
12. List the date, 25 days after today’s date.
13. Find the total of all the billed orders in the month of June.
14. List the products and orders from customers who have ordered less than 5 units of “Pull Overs”
15. Find the list of products and orders placed by “Ivan Bayrosss” and “Mamta Muzumdar”
16. List the clients who placed order before June
17. List all the clients who stays in “Bengaluru” or “Mangalore”

Note : Make sure to adjust the table and column names based on your database schema.

```
1) SELECT OrderNumber, AVG(Rate) AS AverageRate
FROM Orders
GROUP BY OrderNumber;

2) SELECT *
FROM Loans;

3)
SELECT c.CustomerName
FROM Customers c
JOIN Accounts a ON c.CustomerID = a.CustomerID
JOIN Branches b ON a.BranchID = b.BranchID
WHERE c.City = b.City;
```

In the above query, assuming you have the following tables:

- "Customers" table with columns CustomerID and CustomerName, among others.

- "Accounts" table with columns CustomerID and BranchID, among others.
- "Branches" table with columns BranchID and City, among others.

The query performs an inner join between the "Customers" and "Accounts" tables using the CustomerID column. It then joins the "Accounts" and "Branches" tables using the BranchID column. Finally, it includes a WHERE clause to filter the results where the customer's city matches the branch city.

```
4) SELECT l.*
FROM Loans l
JOIN Accounts a ON l.AccountID = a.AccountID
WHERE a.OpeningDate > TO_DATE('2022-08-01', 'YYYY-MM-DD');
```

In the above query, assuming you have a table named "Loans" that contains the loan details and a table named "Accounts" that contains the account details, the query performs an inner join between the two tables using the AccountID column

```
5)
SELECT *
FROM Orders
WHERE ClientID IN ('C00001', 'C00002');
```

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

```
7) SELECT *
FROM Clients
WHERE State <> 'Maharashtra';
```

```
8)
SELECT MAX(Price) AS Max_Price, MIN(Price) AS Min_Price
FROM Products;
```

In the above query, assuming you have a table named "Products" that contains the product details including the price, the SELECT statement calculates the maximum price using the MAX() function and renames the output column as "Max_Price." Similarly, it calculates the minimum price using the MIN() function and renames the output column as "Min_Price."

Executing this query will provide you with the maximum and minimum product prices, with the column names "Max_Price" and "Min_Price" respectively, from the "Products" table.

9) SELECT COUNT(*) AS ProductCount
FROM Products
WHERE Price <= 500;

10) SELECT OrderNumber, TO_CHAR(OrderDate, 'Day') AS OrderDay
FROM Orders;

11)
SELECT EXTRACT(MONTH FROM DeliveryDate) AS DeliveryMonth, EXTRACT(DAY FROM DeliveryDate) AS DeliveryDay
FROM Orders;

12)
SELECT SYSDATE + 25 AS DateAfter25Days
FROM DUAL;

In the above query, the SELECT statement retrieves the current date using the SYSDATE function. The "+ 25" adds 25 days to the current date.

The FROM clause includes the DUAL table, which is a dummy table in Oracle used for selecting single row results.

Executing this query will provide you with the date that is 25 days after today's date. The resulting date will be displayed as "DateAfter25Days".

13)
SELECT SUM(TotalAmount) AS TotalBilledAmount
FROM Orders
WHERE EXTRACT(MONTH FROM OrderDate) = 6; 14)

14)
SELECT c.CustomerName, p.ProductName, o.OrderNumber, o.Quantity
FROM Customers c
JOIN Orders o ON c.CustomerID = o.CustomerID
JOIN Products p ON o.ProductID = p.ProductID
WHERE p.ProductName = 'Pull Overs' AND o.Quantity < 5;

```
15) SELECT c.CustomerName, p.ProductName, o.OrderNumber, o.Quantity
FROM Customers c
JOIN Orders o ON c.CustomerID = o.CustomerID
JOIN Products p ON o.ProductID = p.ProductID
WHERE c.CustomerName IN ('Ivan Bayross', 'Mamta Muzumdar');
```

In the above query, assuming you have the following tables:

- Customers: Contains customer details
- Orders: Contains order details
- Products: Contains product details

The SELECT statement retrieves the customer name (CustomerName), product name (ProductName), order number (OrderNumber), and quantity (Quantity) from the respective tables.

The JOIN clauses are used to join the Customers, Orders, and Products tables based on their related columns.

The WHERE clause includes a condition:

- c.CustomerName IN ('Ivan Bayross', 'Mamta Muzumdar'): Filters the customers to only include Ivan Bayross and Mamta Muzumdar.

Executing this query will provide you with the list of products and orders placed by Ivan Bayross and Mamta Muzumdar. The resulting rows will include the customer name, product name, order number, and quantity.

```
16) SELECT DISTINCT c.ClientName
FROM Clients c
JOIN Orders o ON c.ClientID = o.ClientID
WHERE o.OrderDate < TO_DATE('2022-06-01', 'YYYY-MM-DD');
```

In the above query, assuming you have the following tables:

- Clients: Contains client details
- Orders: Contains order details

The SELECT statement retrieves the distinct client names (ClientName) from the Clients table.

The JOIN clause is used to join the Clients and Orders tables based on their related columns.

The WHERE clause includes a condition:

- o.OrderDate < TO_DATE('2022-06-01', 'YYYY-MM-DD'): Filters the orders to only include orders placed before June 2022. You can modify the date as per your requirement.

Executing this query will provide you with the list of clients who placed orders before June. The resulting rows will include the client names.

17)

```
SELECT *
FROM Clients
WHERE City IN ('Bengaluru', 'Mangalore');
```