SQL Practise Questions

1.Write a SQL query to create a table named Orders with the following columns:

- OrderID (INT, Primary Key, Auto-increment)
- CustomerID (INT)
- OrderDate (DATE)
- TotalAmount (DECIMAL(10, 2))

2.Write a SQL query to create two tables: Customers and Orders. The Customers table should have the following columns:

- CustomerID (INT, Primary Key, Auto-increment)
- CustomerName (NVARCHAR(100))
- Email (NVARCHAR(100), Unique)

3.Write a SQL query to add a column named Status of type VARCHAR(50) to the Orders table.

4.Write a SQL query to drop the Orders table.

5.Write a SQL query to insert a new record into the Orders table with the following values: CustomerID = 101, OrderDate = '2024-08-01', TotalAmount = 250.75, Status = 'Shipped'.

6.Write a SQL query to update the Status of the order with OrderID = 5 to 'Delivered'.

7.Write a SQL query to delete all records from the Orders table where TotalAmount is less than 100.

8.Write a SQL query to retrieve all columns from the Orders table where OrderDate is after January 1, 2024.

9.Write a SQL query to find the average TotalAmount of all orders.

10.Write a SQL query to find the total TotalAmount of orders grouped by Status, but only include groups where the total amount is greater than 500.

11.Write a SQL query to join Orders with a table named Customers on CustomerID and select the OrderID, CustomerName, and TotalAmount.

12.Write a SQL query to list all orders and include customer information if available. If there is no customer information, still include the order details.

13.Write a SQL query to find all orders with a TotalAmount greater than the average TotalAmount of all orders.

14.Write a SQL query to list each CustomerID and the total amount spent by that customer in a subquery.

15.Write a SQL query to create a stored procedure named GetOrdersByCustomer that takes CustomerID as a parameter and returns all orders for that customer.Execute the GetOrdersByCustomer stored procedure for CustomerID 101.

16.Write a SQL query to create a table-valued function named GetOrdersByDateRange that takes two parameters StartDate and EndDate, and returns all orders within that date range.Use the GetOrdersByDateRange function to find all orders between '2024-01-01' and '2024-06-30'.

17.Write a SQL query to create a view named OrderSummary that shows the OrderID, CustomerID, OrderDate, and TotalAmount of orders.Retrieve all records from the OrderSummary view where TotalAmount is greater than 200.