📘 SQL ASSIGNMENT SHEET – BANKING DOMAIN

Course: SQL Server Basics + Core Querying
Domain: Banking
Difficulty: Beginner → Intermediate → Advanced
Submission: Query File (.sql) or SSMS Screenshot Export

📘 PART A – BASIC SQL (40 Marks)

Write SQL queries for each question.

Q1. Display the list of all branches located in Mumbai.

Table: Branches

Q2. List all customers whose last name is 'Sharma'.

Table: Customers

Q3. Show only the first name and email of customers whose email is not NULL.

Q4. Display accounts with a balance greater than 10,000.

Table: Accounts

Q5. Show the top 3 accounts with the highest balance.

Q6. Retrieve the customers born after the year 1990.

Q7. List all unique cities where branches exist.

Q8. Display customers whose first name starts with the letter 'A'.

Use LIKE

Q9. Show all accounts that belong to customer ID = 1.

Q10. Order customers by last name in descending order.

🟩 **PART B – JOIN + FILTERING (40 Marks)**

Q11. List customers along with their account details

(Full Name, AccountID, Balance)

Q12. Show all customers who do NOT have any accounts.

Use LEFT JOIN

Q13. Display all accounts along with their branch name and branch city.

Q14. List all transactions with Customer Full Name and Branch Name.

Q15. Show customers whose total balance (sum of all accounts) is greater than 50,000.

Q16. Find all accounts where the balance is between 5,000 and 30,000.

Q17. Retrieve customers whose account type is Savings (S).

Q18. Display customers with more than one account.

Q19. Display the number of accounts opened in each branch.

Group By Branch

Q20. List all customers with the following balance category:

- Balance < 5000 → "Low"
- 5000–50000 → "Medium"
- 50000 → "High"
  Use CASE expression

🟧 PART C – SUBQUERIES, VIEWS, FUNCTIONS, STORED PROCEDURES (50 Marks)

Q21. Write a subquery to show customers whose balance is above the average balance of all accounts.

Q22. Show customers who have at least one transaction.

Use EXISTS

Q23. Create a VIEW named v_CustomerBalance that displays:

- CustomerID
- FullName
- TotalBalance

Q24. Create a Scalar Function:

dbo.ufn_GetAge(@CustomerID INT)
→ Return the age in years of that customer.

Q25. Create a Table-Valued Function:

dbo.ufn_GetAccountsByBranch(@BranchID INT)
→ Return all accounts of that branch.

Q26. Create a stored procedure usp_GetCustomerTransactions with input parameter @CustomerID

→ Show all transactions of that customer's accounts.

Q27. Create a stored procedure usp_DepositAmount that:

- Accepts AccountID and Amount
- Adds balance
- Inserts a transaction record
- Uses TRY…CATCH

Q28. Create a non-clustered index on Accounts(CustomerID) including Balance.

Q29. Write a CTE to list customers above 30 years of age.

Q30. Using a subquery, list customers whose account balance is greater than customer ID 1's total balance.

🟪 PART D – CASE STUDY (30 Marks)

Bank Name: Modern Bank Pvt Ltd

Business Requirement:

You are assigned to create analytical SQL queries for bank decision-making.

Using the given database tables (Customers, Accounts, Transactions, Branches), write SQL queries for the tasks.

Task 1: Find the top 5 customers with the highest total balance (sum of all accounts).

Task 2: List all customers who did not do any transaction in the last 90 days.

Task 3: Identify branches whose total deposits exceed 1,00,000.

Task 4: Identify customers who transferred money (TranType = 'T') between two accounts.

Task 5: Show each branch’s average account balance along with the number of customers in that branch.