**💰 Day 67 – SQL Challenge: Financial Transactions & Spending Insights**

We’ll work with four tables commonly found in fintech or expense-tracking systems.

1️⃣ **Customers**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| CustomerID | Name | Country | JoinDate | Age |
| 1 | Priya Nair | India | 2020-03-05 | 28 |
| 2 | David Lee | USA | 2021-06-15 | 35 |
| 3 | Fatima Noor | UAE | 2019-11-22 | 30 |
| 4 | Rohan Mehta | India | 2022-01-10 | 26 |
| 5 | Maria Garcia | Spain | 2021-02-20 | 33 |

2️⃣ **Accounts**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| AccountID | CustomerID | AccountType | Balance | OpenDate |
| 101 | 1 | Savings | 55000 | 2020-03-05 |
| 102 | 2 | Current | 88000 | 2021-06-15 |
| 103 | 3 | Savings | 67000 | 2019-11-22 |
| 104 | 4 | Current | 30000 | 2022-01-10 |
| 105 | 5 | Savings | 75000 | 2021-02-20 |

3️⃣ **Transactions**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| TransactionID | AccountID | Amount | TransactionType | TransactionDate | Channel |
| 1001 | 101 | 2000 | Credit | 2022-06-01 | UPI |
| 1002 | 101 | 1500 | Debit | 2022-06-05 | Card |
| 1003 | 102 | 3000 | Credit | 2022-06-07 | Online |
| 1004 | 102 | 2500 | Debit | 2022-06-08 | Card |
| 1005 | 103 | 4000 | Credit | 2022-06-10 | Branch |
| 1006 | 103 | 1000 | Debit | 2022-06-12 | UPI |
| 1007 | 104 | 5000 | Credit | 2022-06-15 | UPI |
| 1008 | 104 | 2000 | Debit | 2022-06-18 | Online |
| 1009 | 105 | 6000 | Credit | 2022-06-20 | Branch |
| 1010 | 105 | 1000 | Debit | 2022-06-22 | Card |

**4️⃣ Loans**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| LoanID | CustomerID | LoanType | LoanAmount | StartDate | InterestRate |
| 201 | 1 | Personal | 200000 | 2021-03-01 | 9.5 |
| 202 | 3 | Home | 500000 | 2020-05-10 | 8.0 |
| 203 | 5 | Car | 300000 | 2021-07-01 | 10.2 |

**💡 Day 67 – Advanced SQL Questions (Real-World + Interview-Level)**

1. Join Practice  
   Show customer name, account type, and their latest transaction date.
2. Aggregation + Conditional Logic  
   Calculate total credited and debited amount for each customer using CASE WHEN.
3. CTE + Aggregation  
   Using a CTE, calculate total transaction amount per customer, and list only those who transacted more than ₹5,000 in total.
4. Subquery Filtering  
   Find customers who made *at least one transaction* greater than ₹4,000.
5. Window Function (LAG)  
   For each account, calculate the number of days between consecutive transactions.
6. Ranking  
   Rank customers based on total credited amount (highest first).
7. Nested CTE + Analytics  
   Using nested CTEs, calculate each country’s total transaction volume and find which customer contributed the most in that country.
8. Correlated Subquery  
   Find customers whose balance is above their country’s average balance.
9. JOIN + Date Logic  
   List all loan holders who had an active loan before their first recorded transaction.
10. Real-World Finance Query (Advanced)  
    Identify customers whose total debit percentage (debit / total transactions \* 100) exceeds 40%.
11. 🚀 Bonus Challenge (Complex Analytical Logic)  
    Find the top loyal customer — defined as the one who has made the most consistent monthly transactions (one or more per month for maximum consecutive months).