Indian Institute of Technology,
Gandhinagar

SQL Based Multi-Bank Performance Analytics Tool

Chandra Shekhar

B.Tech Mechanical Engineering

Contents

1 Introduction

The **Multi-Bank Performance Analytics Tool** is a comprehensive software solution implemented via a relational database in MySQL to analyze and compare key performance metrics across multiple banks. The system consolidates data related to banks, branches, customers, products, transactions, fees, and holidays to generate insightful reports supporting data-driven banking management.

This tool automates complex SQL queries for business intelligence purposes, including revenue analysis, customer retention, transaction trends, payment behavior, and holiday impact on financial activities. It enables banks to improve customer targeting, optimize product offerings, and enhance operational efficiency.

The database schema is designed for high data integrity and flexibility, supporting varied banking entities and transactional details. The project emphasizes normalized data organization and relationship enforcement to avoid redundancy and maintain consistency.

2 Objectives

The primary objectives of this project are:

- **Enable cross-bank analytics:** Aggregate and analyze transactional and customer data across multiple banks and their branches, allowing bank-level and branch-level performance comparisons.
- **Provide comprehensive financial insights:** Generate detailed revenue and fees reports by product, evaluate customer spending behavior using RFM analysis, and track payment methods trends.
- **Support decision-making with key metrics:** Measure customer retention rates, identify high-value customers contributing most revenue, and detect revenue patterns around official holidays.
- **Ensure data integrity and consistency:** Employ rigorous foreign key constraints and normalized design to maintain accurate and consistent multi-entity relationships.
- **Demonstrate complex query capabilities:** Implement advanced SQL queries with aggregations, window functions, and common table expressions to deliver nuanced analytics outputs.

3 System Description

The system is architected as a MySQL relational database comprising interlinked tables representing banks, branches, customers, financial products, transactions, transaction fees, and official holidays.

- **Entity Tables:**
  - `Banks`: Stores bank identifiers, names, and headquarters.
  - `Branches`: Captures branch details linked to banks including city and state.

  - `Customers`: Maintains customer information associated with banks.
  - `Products`: Defines banking products offered, categorized into deposits, loans, cards, etc.

- **Transactional Tables:**
  - `Transactions`: Records each financial transaction with references to customers, branches, and products.
  - `Fees`: Holds fee data linked to transactions for processing, service, or annual charges.

- **Auxiliary Tables:**
  - `Holidays`: Lists official holidays to analyze transaction activity on those dates.

Relationships are enforced via foreign keys ensuring that data between banks, customers, branches, and transactions is consistent. This supports complex joins and aggregations for performance analytics.

4 Inputs and Outputs

4.1 Inputs

Data inputs for this tool come primarily from bank operational systems and include:

- Bank details: Names, locations, and identifiers.
- Branch and customer data: Branch locations and customer demographics.
- Product catalogs: Information on deposit accounts, loan types, credit cards, and fixed deposits.
- Transaction logs: Entries with dates, amounts, payment methods, and associated customers and branches.
- Fees: Charges applied to individual transactions.
- Holidays: Official non-working days impacting transactions.

4.2 Outputs

Analytic queries produce valuable outputs including:

- **Total revenue per bank:** Aggregated transaction values indicating bank performance.
- **Top customers by revenue:** Ranked customers per bank by transaction volume.
- **Branch revenue trends:** Monthly summaries highlighting branch financial health over time.

- **Customer retention rates:** Percentage of customers with multiple transactions, reflecting loyalty.
- **RFM analysis results:** Recency, frequency, and monetary values per customer for segmentation.
- **Payment method trends:** Frequency and value of transactions by payment type.
- **Product revenue and fee summaries:** Combined income and fees generated for various bank products.
- **High-value customer identification:** Customers accounting for 80% of bank revenues.
- **Holiday transaction impact:** Bank revenues generated on national holidays.

These outputs are suitable for direct consumption by bank analysts or feeding into dashboard tools.

5 Functional Modules

The system's functionality is implemented through distinct interrelated modules:

- **Data Storage Module:** Normalized schema organizing banks, branches, customers, products, transactions, fees, and holidays.
- **Data Ingestion Module:** SQL statements to insert sample and real operational data into the database.
- **Analytics Module:** Complex SQL queries and common table expressions (CTEs) for aggregations, rankings, and segmentation.
- **Reporting Module:** Queries structured to return actionable metrics and rankings for business intelligence.

6 Key SQL Query Examples

A few key SQL analytics queries implemented are:

- **Total Revenue Per Bank:** This query calculates the aggregate transaction amounts grouped by each bank, providing a straightforward measure of the total income generated by every bank in the system. It joins the transactions with their corresponding customers and banks to ensure accurate association.
- **Top Customers By Revenue Per Bank:** This query identifies the leading customers for each bank based on the total sum of their transaction amounts. It ranks customers in descending order of spending to help the bank recognize high-value clients and tailor services accordingly.

- **Customer Retention Rate:** This analysis calculates the percentage of customers who have made multiple transactions, indicating the level of repeat engagement and loyalty. It leverages a common table expression to filter customers with more than one transaction and then computes the retention rate as a ratio of repeat customers to total customers per bank.

These queries illustrate the power of relational SQL to provide detailed insights into banking operations.

7 Error Handling and Data Integrity

Although primarily an analytics tool, the system enforces data integrity via:

- **Foreign Key Constraints:** Ensuring references between customers, banks, branches, and transactions remain valid.
- **Data Type Enforcement:** Specified data types uphold consistency (e.g., decimal types for money, dates for transaction dates).
- **Nullability Rules:** Mandatory fields prevent incomplete data entries.

These measures guarantee that the data analyzed is accurate and reliable for decision-making.

8 Future Enhancements

Potential improvements include:

- Adding temporal analytics for trend forecasting using machine learning.
- Extending schema to include more detailed customer demographics and transaction metadata.

9 Conclusion

The SQL Based Multi-Bank Performance Analytics Tool offers a robust database foundation for analyzing transactional and customer data across multiple banks. Its normalized schema and advanced SQL querying support a wide range of critical banking business intelligence, from revenue tracking to customer retention and payment method trends. Such insights empower management to optimize products, improve customer relationships, and increase profitability. This project exemplifies how well-designed database architecture and SQL analytics can drive strategic decisions in the banking sector.