SUBMITTED BY: SHUBHDEEP

PROJECT REPORT

PERSONAL FINANCE TRACKER

(SQL-Based Mini Project)

Introduction

A SQL-based database project called the Personal Finance Tracker was developed to assist people in effectively managing their financial lives. It enables users to keep track of and document their earnings, outlays, budgets, savings objectives, and ongoing bills. The system is made to mimic a real-world personal finance setting, allowing users to track their monthly spending patterns and use SQL queries to obtain insightful data.

Abstract

The goal of this project is to design and implement a query-optimized, normalized SQL database offering a comprehensive system for managing personal finances. Real-world use cases are supported, from tracking monthly expenses and bills to warning users about risks to savings goals and budget overspending. Using window functions like LAG(), sophisticated queries were constructed for intelligent insights like monthly trend analysis, recurring due alerts, and low balance detection.

Tools Used

- **Database Platform**: MySQL Workbench
- **Languages Used**: SQL (DDL, DML, Joins, Aggregations, Window Functions)
- **Concepts Applied**: Normalization, Foreign Keys, Data Integrity, Analytical Queries
- **Entity Relationship Diagram Tool:** dbdiagram.io

Steps Involved in Building the Project

1. **Database Design**
   Created a normalized relational schema with 9 tables including Users, Accounts, Transactions, Budgets, Tags, Categories, RecurringTransactions, SavingsGoals, and TransactionTags.
2. **Table Creation**
   Wrote CREATE TABLE statements for all tables with proper data types, primary keys, and foreign key constraints to maintain data integrity.
3. **Data Insertion**
   Inserted realistic records for 5 users, covering different months of income, rent, groceries, electricity bills, and savings goals.
4. **Advanced Query Writing**
   Developed 9+ SQL queries to analyze user spending patterns, monthly summaries, net balances, budget comparisons, recurring alerts, and trend analysis using LAG().
5. **Validation and Output Testing**
   Executed SELECT queries to test data accuracy and ensured logical outputs from all queries by running them in MySQL Workbench.

Key SQL Queries and Their Real-World Applications

1. Monthly Expense Summary (Query 1)
   This query tracks each user's monthly spending by category. It helps admin or users analyze where the most money is being spent — useful for financial planning, dashboards, and behaviour tracking.

2. User-wise Income & Expense Overview (Query 2)
   Generates a personalized monthly financial report for any user. Useful in apps where logged-in users want to review their income vs. expense breakdown for each month.

3. Budget vs. Actual Expense Comparison (Query 3)
   Compares a user's set budget against their actual spending category-wise and highlights overspending. This logic is used in every finance app to alert users when they exceed limits.

4. Net Monthly Balance Tracker (Query 4)
   Calculates net balance (Income - Expense) per user per month and shows if it's POSITIVE or NEGATIVE. Very useful for financial health tracking and alerts.

5. Spending by Tags (Query 5)
   Shows how much a user spends on tagged items like Groceries, Rent, Medical, etc. Useful for categorizing expenses and visualizing spending patterns in pie charts or graphs.

6. Low Balance Alert (Query 6)
   Identifies accounts where balance falls below ₹1000 and flags them. This mimics bank alerts or notifications for topping up wallets/accounts.

7. Savings Goal Risk Detection (Query 7)
   Flags savings goals that are at risk of not being met due to low progress and upcoming deadline. Great for setting reminders or encouraging users to save more.

8. Upcoming Recurring Transactions (Query 8)
   Notifies users about bills or payments that are due in the next 10 days. This is a common feature in modern apps like Cred or Paytm.

9. Month-over-Month Expense Trends (Query 9)
   Uses LAG() window function to compare current month's expense to previous month's — shows Increase/Decrease/No Change. This is powerful for trend visualizations and financial coaching apps.

CONCLUSION

A full, practical implementation of a SQL-based Personal Finance Tracker system is demonstrated in this project. It displays the breadth of knowledge in data handling and analytics, from creating normalized relational tables to adding realistic user data and crafting expert-level queries. The system is useful for any personal finance application because the queries created offer actionable insights like goal tracking, low balance alerts, and spending trends. This practical project improves SQL skills and provides a strong basis for careers in analytics, dashboard-based finance, or backend development.