# 📄 Personal Finance Tracker

## 🔸 Introduction

Managing personal finances is a crucial aspect of everyday life. This project focuses on building a Personal Finance Tracker using SQL and PostgreSQL to help users monitor their income, expenses, and savings efficiently. The system provides insightful reports that help users analyze their spending habits and manage their budgets wisely.

## 🔸 Abstract

The objective of this project is to develop a budget tracking system using PostgreSQL. The system stores user information, income records, and categorized expense data. Various SQL queries and views are used to summarize and analyze user financial data, such as monthly expenses, category-wise breakdowns, and net savings. This solution helps in visualizing financial health through data-driven insights.

## 🔸 *Tools Used*

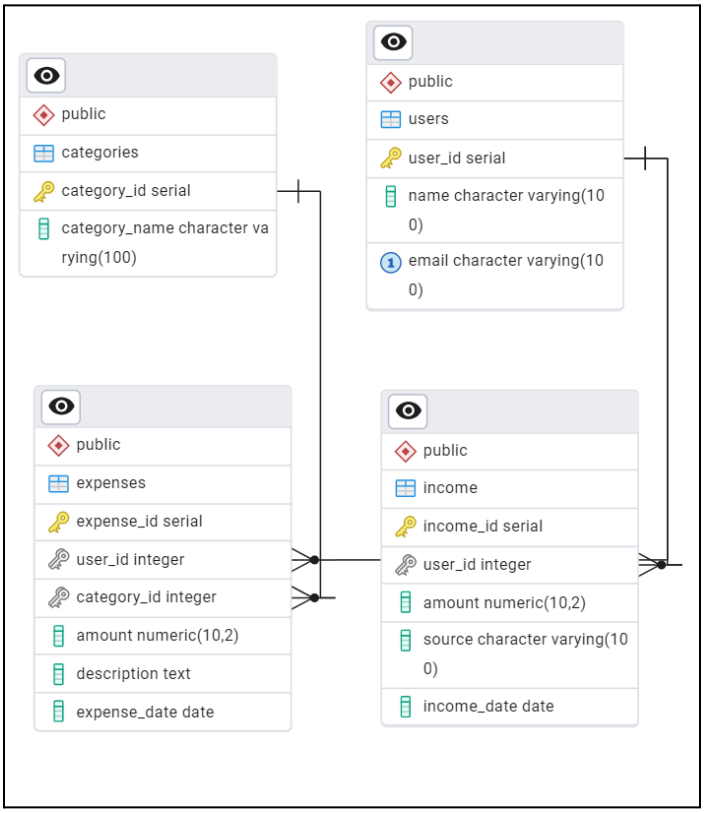
- Database: PostgreSQL
- IDE: pgAdmin / psql CLI
- Language: SQL
- Export: CSV files for reports
- Documentation: MS Word / Google Docs for report

## 🔸 Steps Involved

1. Database Design:  Designed normalized schema with 4 main tables: Users, Categories, Income, Expenses.
2. Schema Implementation: Created tables with appropriate data types, foreign key constraints, and relationships.
3. Data Insertion: Populated database with 10 users, 8 categories, 30+ income records, and 40+ expense records.
4. Query Writing
   Wrote multiple SQL queries:
   - Monthly expense summary
   - Category-wise spending
   - Balance tracking per user
   - Top 3 spending categories
   - Users spending more than their income
5. View Creation: Created a view user_balance to track total income, expenses, and remaining balance for each user.
6. Reporting: Exported views and query results to CSV files using \copy in psql and the export feature in pgAdmin.

## 🔸 Conclusion

The Personal Finance Tracker provides a structured and insightful way to manage finances using SQL. Through this project, I learned practical database design, query optimization, and how to turn raw financial data into useful information. These skills are highly transferable to real-world data analytics and backend development roles.




Query | Query History

```
SELECT e.user_id, u.name,
DATE_TRUNC('month', e.expense_date)::DATE AS month,
SUM(e.amount) AS total_expenses FROM expenses e
JOIN users u ON e.user_id = u.user_id
GROUP BY e.user_id, u.name, month ORDER BY e.user_id, month;
```

Data Output | Messages | Notifications

| | user_id integer | name character varying (100) | month date | total_expenses numeric |
|---|---|---|---|---|
| 1 | 1 | Alice | 2025-07-01 | 22200.00 |
| 2 | 2 | Bob | 2025-07-01 | 21800.00 |
| 3 | 3 | Charlie | 2025-07-01 | 21000.00 |
| 4 | 4 | David | 2025-07-01 | 9500.00 |
| 5 | 5 | Emma | 2025-07-01 | 22800.00 |
| 6 | 6 | Farah | 2025-07-01 | 3700.00 |
| 7 | 7 | George | 2025-07-01 | 18000.00 |
| 8 | 8 | Hina | 2025-07-01 | 17100.00 |
| 9 | 9 | Isha | 2025-07-01 | 8900.00 |
| 10 | 10 | John | 2025-07-01 | 20500.00 |

Monthly Expense Summary

Query | Query History

```
SELECT u.name, c.category_name, SUM(e.amount) AS total_spent
FROM expenses e JOIN users u ON e.user_id = u.user_id
JOIN categories c ON e.category_id = c.category_id
GROUP BY u.name, c.category_name ORDER BY u.name, total_spent DESC;
```

Data Output | Messages | Notifications

| | name character varying (100) | category_name character varying (100) | total_spent numeric |
|---|---|---|---|
| 1 | Alice | Rent | 15000.00 |
| 2 | Alice | Groceries | 3200.00 |
| 3 | Alice | Utilities | 2300.00 |
| 4 | Alice | Entertainment | 1200.00 |
| 5 | Alice | Medical | 500.00 |
| 6 | Bob | Rent | 14500.00 |
| 7 | Bob | Groceries | 3100.00 |
| 8 | Bob | Medical | 2200.00 |
| 9 | Bob | Dining Out | 1300.00 |
| 10 | Bob | Transport | 700.00 |
| 11 | Charlie | Rent | 15500.00 |

Category-Wise Spending Summary

Query | Query History

```sql
CREATE VIEW user_balance AS
SELECT u.name,COALESCE(SUM(i.amount), 0) AS total_income,
COALESCE(SUM(e.amount), 0) AS total_expense,
COALESCE(SUM(i.amount), 0) - COALESCE(SUM(e.amount), 0) AS balance
FROM users u LEFT JOIN income i ON u.user_id = i.user_id
LEFT JOIN expenses e ON u.user_id = e.user_id GROUP BY u.name;
```

Data Output | Messages | Notifications

```
CREATE VIEW

Query returned successfully in 153 msec.
```

Query | Query History

```sql
SELECT * FROM user_balance;
```

Data Output | Messages | Notifications

| | name character varying (100) | total_income numeric | total_expense numeric | balance numeric |
|---|---|---|---|---|
| 1 | David | 265500.00 | 28500.00 | 237000.00 |
| 2 | Alice | 302500.00 | 88800.00 | 213700.00 |
| 3 | Bob | 297500.00 | 87200.00 | 210300.00 |
| 4 | Hina | 138600.00 | 51300.00 | 87300.00 |
| 5 | Charlie | 230000.00 | 63000.00 | 167000.00 |
| 6 | John | 202000.00 | 41000.00 | 161000.00 |
| 7 | Isha | 235200.00 | 17800.00 | 217400.00 |
| 8 | Farah | 195600.00 | 11100.00 | 184500.00 |
| 9 | George | 126600.00 | 54000.00 | 72600.00 |
| 10 | Emma | 212000.00 | 68400.00 | 143600.00 |

Query | Query History

```sql
SELECT name, category_name, total_spent
FROM (SELECT u.name, c.category_name,SUM(e.amount) AS total_spent,
RANK() OVER (PARTITION BY u.name ORDER BY SUM(e.amount) DESC) AS rk
FROM expenses e JOIN users u ON u.user_id = e.user_id
JOIN categories c ON c.category_id = e.category_id
GROUP BY u.name, c.category_name) ranked WHERE rk <= 3;
```

Data Output | Messages | Notifications

| | name character varying (100) | category_name character varying (100) | total_spent numeric |
|---|---|---|---|
| 15 | Emma | Groceries | 2700.00 |
| 16 | Farah | Utilities | 2100.00 |
| 17 | Farah | Medical | 1200.00 |
| 18 | Farah | Transport | 400.00 |
| 19 | George | Rent | 13500.00 |
| 20 | George | Groceries | 3000.00 |
| 21 | George | Utilities | 1500.00 |
| 22 | Hina | Rent | 14500.00 |
| 23 | Hina | Dining Out | 1700.00 |

Query | Query History

```sql
SELECT u.name,ROUND(AVG(monthly_total), 2) AS avg_monthly_spending
FROM (SELECT user_id, DATE_TRUNC('month', expense_date) AS month,
SUM(amount) AS monthly_total FROM expenses GROUP BY user_id, month) monthly_expense
JOIN users u ON monthly_expense.user_id = u.user_id GROUP BY u.name;
```

Data Output | Messages | Notifications

| | name character varying (100) | avg_monthly_spending numeric |
|---|---|---|
| 1 | David | 9500.00 |
| 2 | Alice | 22200.00 |
| 3 | Bob | 21800.00 |
| 4 | Hina | 17100.00 |
| 5 | Charlie | 21000.00 |
| 6 | John | 20500.00 |
| 7 | Isha | 8900.00 |
| 8 | Farah | 3700.00 |
| 9 | George | 18000.00 |
| 10 | Emma | 22800.00 |

Query | Query History

```
SELECT u.name, DATE_TRUNC('month', COALESCE(i.income_date, e.expense_date))::DATE AS month,
COALESCE(SUM(i.amount), 0) AS total_income, COALESCE(SUM(e.amount), 0) AS total_expenses,
COALESCE(SUM(i.amount), 0) - COALESCE(SUM(e.amount), 0) AS net_savings FROM users u
LEFT JOIN income i ON u.user_id = i.user_id LEFT JOIN expenses e ON u.user_id = e.user_id
GROUP BY u.name, month ORDER BY u.name, month;
```

Data Output | Messages | Notifications

Showing rows

| | name character varying (100) | month date | total_income numeric | total_expenses numeric | net_savings numeric |
|---|---|---|---|---|---|
| 1 | Alice | 2025-07-01 | 302500.00 | 88800.00 | 213700.00 |
| 2 | Bob | 2025-07-01 | 297500.00 | 87200.00 | 210300.00 |
| 3 | Charlie | 2025-07-01 | 230000.00 | 63000.00 | 167000.00 |
| 4 | David | 2025-07-01 | 265500.00 | 28500.00 | 237000.00 |
| 5 | Emma | 2025-07-01 | 212000.00 | 68400.00 | 143600.00 |
| 6 | Farah | 2025-07-01 | 195600.00 | 11100.00 | 184500.00 |
| 7 | George | 2025-07-01 | 126600.00 | 54000.00 | 72600.00 |
| 8 | Hina | 2025-07-01 | 138600.00 | 51300.00 | 87300.00 |
| 9 | Isha | 2025-07-01 | 235200.00 | 17800.00 | 217400.00 |
| 10 | John | 2025-07-01 | 202000.00 | 41000.00 | 161000.00 |

Query | Query History

```
SELECT u.name,e.expense_date, SUM(e.amount) AS total_spent
FROM expenses e JOIN users u ON e.user_id = u.user_id
GROUP BY u.name, e.expense_date ORDER BY u.name, e.expense_date;
```

Data Output | Messages | Notifications

| | name character varying (100) | expense_date date | total_spent numeric |
|---|---|---|---|
| 1 | Alice | 2025-07-02 | 3200.00 |
| 2 | Alice | 2025-07-03 | 15000.00 |
| 3 | Alice | 2025-07-05 | 2300.00 |
| 4 | Alice | 2025-07-08 | 1200.00 |
| 5 | Alice | 2025-07-10 | 500.00 |
| 6 | Bob | 2025-07-03 | 3100.00 |
| 7 | Bob | 2025-07-04 | 14500.00 |
| 8 | Bob | 2025-07-06 | 700.00 |
| 9 | Bob | 2025-07-08 | 2200.00 |
| 10 | Bob | 2025-07-11 | 1300.00 |
| 11 | Charlie | 2025-07-02 | 3600.00 |

Query | Query History

```
SELECT u.name, c.category_name, e.amount, e.expense_date
FROM expenses e JOIN users u ON u.user_id = e.user_id
JOIN categories c ON c.category_id = e.category_id
WHERE e.amount > 5000 ORDER BY e.amount DESC;
```

Data Output | Messages | Notifications

| | name character varying (100) | category_name character varying (100) | amount numeric (10,2) | expense_date date |
|---|---|---|---|---|
| 1 | Charlie | Rent | 15500.00 | 2025-07-03 |
| 2 | John | Rent | 15500.00 | 2025-07-03 |
| 3 | Alice | Rent | 15000.00 | 2025-07-03 |
| 4 | Hina | Rent | 14500.00 | 2025-07-05 |
| 5 | Bob | Rent | 14500.00 | 2025-07-04 |
| 6 | Emma | Rent | 14000.00 | 2025-07-05 |
| 7 | George | Rent | 13500.00 | 2025-07-03 |

Query | Query History

```
SELECT u.name,c.category_name,
ROUND(SUM(e.amount) * 100.0 / SUM(SUM(e.amount))
OVER (PARTITION BY u.name), 2) AS percentage
FROM expenses e JOIN users u ON e.user_id = u.user_id
JOIN categories c ON e.category_id = c.category_id
GROUP BY u.name, c.category_name;
```

Data Output | Messages | Notifications

| | name character varying (100) | category_name character varying (100) | percentage numeric |
|---|---|---|---|
| 32 | Hina | Rent | 84.80 |
| 33 | Isha | Education | 33.71 |
| 34 | Isha | Groceries | 39.33 |
| 35 | Isha | Transport | 6.74 |
| 36 | Isha | Utilities | 20.22 |
| 37 | John | Entertainment | 5.85 |
| 38 | John | Groceries | 14.15 |
| 39 | John | Medical | 4.39 |
| 40 | John | Rent | 75.61 |

Query | Query History

```
SELECT u.name,COUNT(DISTINCT e.expense_id) AS total_expenses,
COUNT(DISTINCT i.income_id) AS total_incomes FROM users u
LEFT JOIN expenses e ON u.user_id = e.user_id
LEFT JOIN income i ON u.user_id = i.user_id
GROUP BY u.name ORDER BY total_expenses DESC;
```

Data Output | Messages | Notifications

| | name character varying (100) | total_expenses bigint | total_incomes bigint |
|---|---|---|---|
| 1 | David | 5 | 3 |
| 2 | Bob | 5 | 4 |
| 3 | Alice | 5 | 4 |
| 4 | John | 4 | 2 |
| 5 | Charlie | 4 | 3 |
| 6 | Emma | 4 | 3 |
| 7 | Isha | 4 | 2 |
| 8 | Hina | 3 | 3 |
| 9 | Farah | 3 | 3 |
| 10 | George | 3 | 3 |

Query | Query History

```
SELECT c.category_name, SUM(e.amount) AS total_spent
FROM expenses e JOIN categories c
ON c.category_id = e.category_id
GROUP BY c.category_name ORDER BY total_spent DESC;
```

Data Output | Messages | Notifications

| | category_name character varying (100) | total_spent numeric |
|---|---|---|
| 1 | Rent | 102500.00 |
| 2 | Groceries | 24900.00 |
| 3 | Utilities | 11800.00 |
| 4 | Education | 9500.00 |
| 5 | Medical | 4800.00 |
| 6 | Dining Out | 4600.00 |
| 7 | Entertainment | 4400.00 |
| 8 | Transport | 3000.00 |

Query | Query History

```
SELECT u.name, DATE_TRUNC('week', e.expense_date)::DATE AS week_start,
SUM(e.amount) AS weekly_spent FROM expenses e
JOIN users u ON e.user_id = u.user_id
GROUP BY u.name, week_start ORDER BY u.name, week_start;
```

Data Output | Messages | Notifications

| | name character varying (100) | week_start date | weekly_spent numeric |
|---|---|---|---|
| 9 | Emma | 2025-06-30 | 21200.00 |
| 10 | Emma | 2025-07-14 | 1600.00 |
| 11 | Farah | 2025-06-30 | 2100.00 |
| 12 | Farah | 2025-07-07 | 1600.00 |
| 13 | George | 2025-06-30 | 18000.00 |
| 14 | Hina | 2025-06-30 | 14500.00 |
| 15 | Hina | 2025-07-07 | 2600.00 |
| 16 | Isha | 2025-06-30 | 5900.00 |
| 17 | Isha | 2025-07-07 | 3000.00 |
| 18 | John | 2025-06-30 | 18400.00 |
| 19 | John | 2025-07-07 | 2100.00 |

Query | Query History

```
1  SELECT source, SUM(amount) AS total_income
2  FROM income GROUP BY source
3  ORDER BY total_income DESC;
```

Data Output | Messages | Notifications

| | source character varying (100) | total_income numeric |
|---|---|---|
| 1 | Salary | 497000.00 |
| 2 | Rent Income | 17000.00 |
| 3 | Bonus | 11000.00 |
| 4 | Freelance | 8200.00 |
| 5 | Other | 5000.00 |
| 6 | Dividends | 3000.00 |
| 7 | Part-time | 1700.00 |
| 8 | Reimburse | 1500.00 |
| 9 | Cashback | 1500.00 |
| 10 | Side Gig | 600.00 |

Query | Query History

```
1  SELECT u.name, MAX(e.amount) AS highest_expense
2  FROM expenses e
3  JOIN users u ON u.user_id = e.user_id
4  GROUP BY u.name ORDER BY highest_expense DESC;
```

Data Output | Messages | Notifications

| | name character varying (100) | highest_expense numeric |
|---|---|---|
| 1 | John | 15500.00 |
| 2 | Charlie | 15500.00 |
| 3 | Alice | 15000.00 |
| 4 | Bob | 14500.00 |
| 5 | Hina | 14500.00 |
| 6 | Emma | 14000.00 |
| 7 | George | 13500.00 |
| 8 | Isha | 3500.00 |
| 9 | David | 2900.00 |
| 10 | Farah | 2100.00 |

Query | Query History

```
1  SELECT * FROM user_balance
2  WHERE balance < 500000
3  ORDER BY balance;
```

Data Output | Messages | Notifications

| | name character varying (100) | total_income numeric | total_expense numeric | balance numeric |
|---|---|---|---|---|
| 1 | George | 126600.00 | 54000.00 | 72600.00 |
| 2 | Hina | 138600.00 | 51300.00 | 87300.00 |
| 3 | Emma | 212000.00 | 68400.00 | 143600.00 |
| 4 | John | 202000.00 | 41000.00 | 161000.00 |
| 5 | Charlie | 230000.00 | 63000.00 | 167000.00 |
| 6 | Farah | 195600.00 | 11100.00 | 184500.00 |
| 7 | Bob | 297500.00 | 87200.00 | 210300.00 |
| 8 | Alice | 302500.00 | 88800.00 | 213700.00 |
| 9 | Isha | 235200.00 | 17800.00 | 217400.00 |
| 10 | David | 265500.00 | 28500.00 | 237000.00 |