PERSONAL FINANCE HEALTH ANALYZER AND OPTIMIZER

**Introduction:** This project is a MySQL system designed to help individuals analyze and optimize the personal finances. It will include tracking income expenses, savings, investments and debts it will provide actional able insights that will help improving financial wellbeing the system will use

* MySQL concepts to generate reports
* Identify Financial trends
* Suggests optimization strategies

**Pre- requisites** for this project: in order to there are some pre-requisites

1. MYSQL 8.0 command line client
2. MYSQL 8.0 Workbench CE
3. MYSQL statements
4. MYSQL operations
5. MYSQL clauses
6. MYSQL constraints
7. MYSQL sub-queries
8. MYSQL Joins
9. User permissions (GRANT & REVOKE)
10. Transactions

These are the concepts or technologies which the user needs to-be fundamentally strong

**Key Features of this project**

This project will contain several important features of real-world Finance Management such as

1. Income and expense tracking- it is used to track monthly income sources and categories expenses
2. Savings and investment analysis- Monitoring savings account and investment portfolio along with calculating returns
3. Debt management- it is used to track debts, (credit cards, loans etc.) and calculate interest payments along with suggesting debt re-payment strategies
4. Financial health scoring- It is used to generate a financial health score based on income expenses, investments, saving and debts it is used to assess financial stability
5. Budget Optimization- It is used to suggest budget allocation that are optimal based on previous data
6. User permissions- Used to provide role-based access. It is also used to restrict access on sensitive financial data.

**Schema:**

In order to perform this project, there are various parameters required:

* **Database**: create a new database for this project with the project name- Personal\_Finance\_Health
* **Tables required**:

1. Users - this table should contain user’s details (User\_id, User\_name, Role, password)
2. Income- Used to store income sources of user (Income\_id, User\_id, source, Amount, Date)
3. Expense- It is used to collect expense (Expense\_id, User\_id, Category, Amount, Date)
4. Savings- Used to store savings details (Saving\_id, User\_id, Account\_type, Amount, Date)
5. Investments details- (Investment\_id, User\_id, Type, Amount, Return\_rate, Date)
6. Debts- used to store debts details (Debt\_id, User\_id, Type, Amount, Intreset\_rate, Due\_date)
7. FinancialHealth – Used to store financial health scores (Health\_id, User\_id, Score, Date)

* With the use of constraint keys, the users table must be linked with all the other table with the use of primary key and foreign key, this is called One-to-Many relationship.
* **Users:**
  + Admin – with admin privileged
  + User – with select only data privileges
* **Implementation**: In order to design the structure for this project and generate analysis based on the data here are the steps of implementation for this project
  + - Creatin Database and Creating necessary tables.

**Insertion**

Inserting values more than 100 rows in all 7 tables to perform the analysis. Apart from the auto-increment columns all the columns have to be filled with data

1- Users: There should be 10 users among those 10- 1 should be named as Admin whereas the rest 9 will be users,

2- Income: For each user there are different source of income along with profit generated they have been assigned to a particular user id that matches the user ids from the table users

3- Expenses: This table covers the expenses of all 9 users and it will classify those expenses into various categories

4- Savings: This table provides the values for each and every saving medium for each user

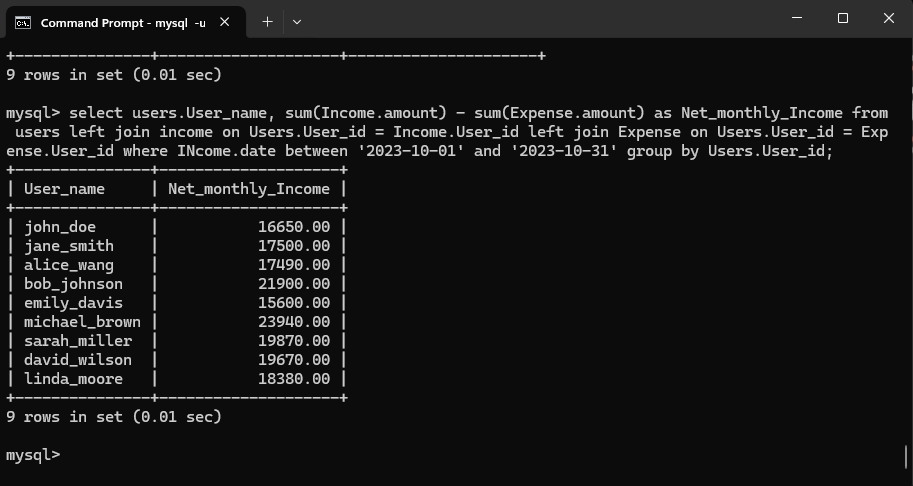
5- Investment: This tables used to provide value of each investment done by the user and the return they have received

6- Debts: this table is used to provide values that are loan information of each user and how much amount they have loaned and at what interest

**Requirements:**

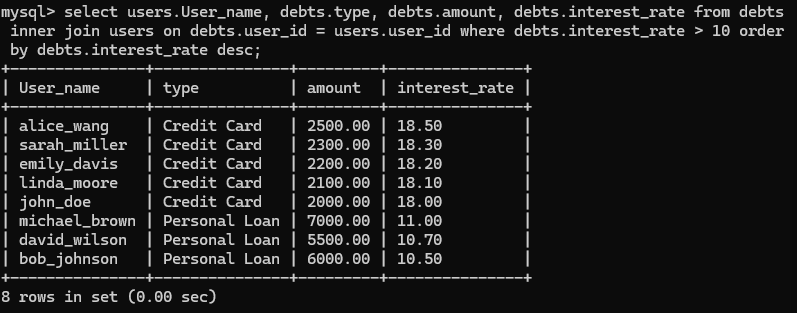
* Calculating monthly net income for each user.in order to fetch details for this requirement 3 tables have to be used – user, income and expenses - all the tables have to be joined

MySQL> select users.User\_name, sum(Income.amount) - sum(Expense.amount) as Net\_monthly\_Income from users left join income on Users.User\_id = Income.User\_id left join Expense on Users.User\_id = Expense.User\_id where Income.date between '2023-10-01' and '2023-10-31' group by Users.User\_id;



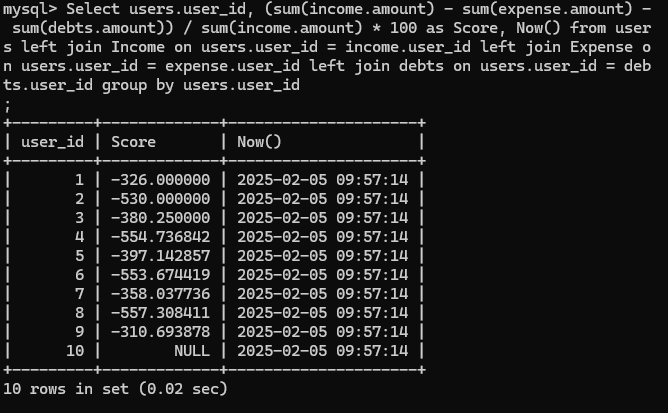
* Identifying high interest debts:

MySQL> select users.User\_name, debts.type, debts.amount, debts.interest\_rate from debts inner join users on debts.user\_id = users.user\_id where debts.interest\_rate > 10 order by debts.interest\_rate desc;



* Generate Financial health Score: Using the data in this requirement the user will calculate financial health score with the help of 4 table – user income, expense and debts

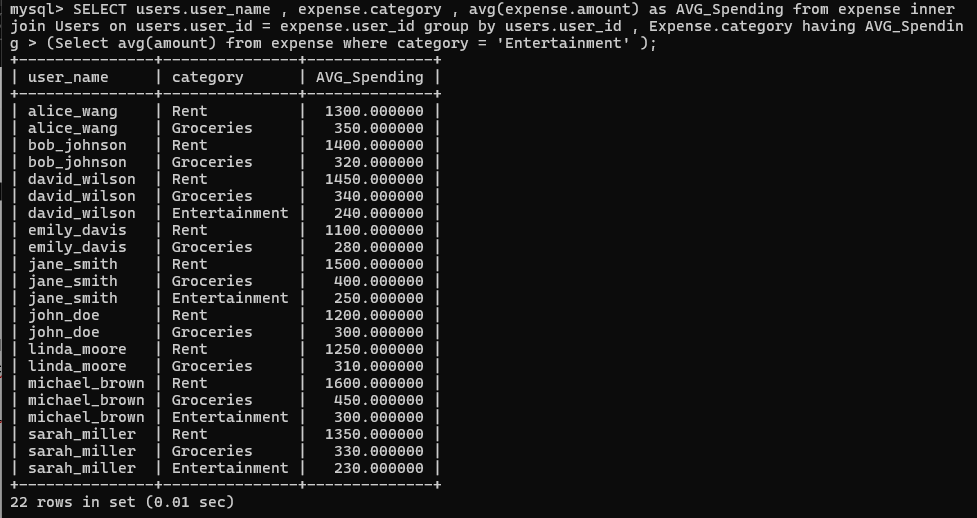
MYSQL> Select users.user\_id, (sum(income.amount) - sum(expense.amount) - sum(debts.amount)) / sum(income.amount) \* 100 as Score, Now() from users left join Income on users.user\_id = income.user\_id left join Expense on users.user\_id = expense.user\_id left join debts on users.user\_id = debts.user\_id group by users.user\_id



* Budget Optimization:

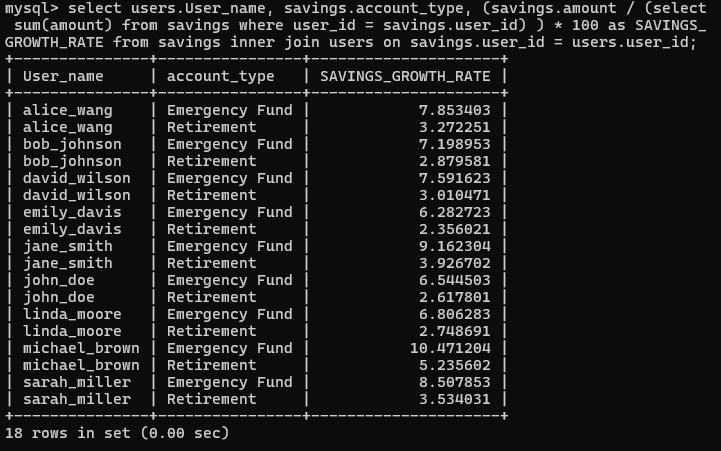
In this requirement the users will be provided the average spending in based on expenses

MYSQL> SELECT users.user\_name, expense.category, avg(expense.amount) as AVG\_Spending from expense inner join Users on users.user\_id = expense.user\_id group by users.user\_id , Expense.category having AVG\_Spending > (Select avg(amount) from expense where category = ‘Entertainment’ );



* Calculating savings growth rate:

MYSQL> select users.User\_name, savings.account\_type, (savings.amount ) (select sum(amount) from savings where user\_id = savings.user\_id) \* 100 as SAVINGS\_GROWTH\_RATE from savings inner join users on savings.user\_id = users.user\_id;



USER PERMISIONS:

* Create user ‘admin’@’localhost’ identified by ‘admin123’;
* Show grants for ‘admin’@’localhost’;
* Grant all privileges on personalfinancehealth .\* to ‘admin’@’localhost’ with grant option;
* Select users, host from mysql user;
* create user 'user'@'localhost' identified by'user123';
* grant select on personalfinancehealth.\* to 'user'@'localhost';

TRANSACTIONS

Over the course of the period in the data there are several updates when it comes to income, savings, investments, debts and expenses. In order to update the values, it is safer to update the values inside a transaction. There are 4 transactions to be performed on this project

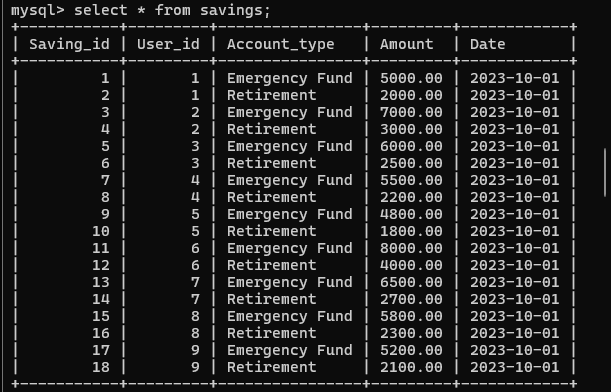
1. Record income and update savings.
2. Debt pay off and update savings.
3. Transport funds between savings account.
4. Record investments and update savings.

Demonstrations:

* RECORD INCOME AND UPDATE SAVINGS:

Mysql> start transaction;

Mysql> select \* from savings;



Mysql> select \* from income;

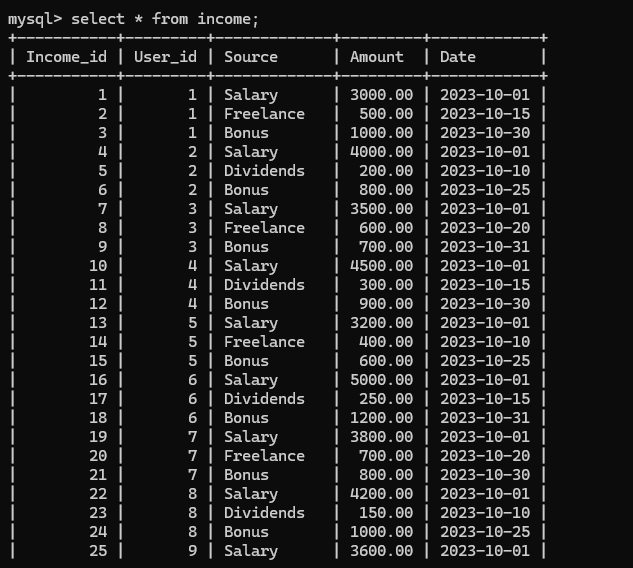
Mysql> SAVEPOINT s1;

Mysql> insert into income (User\_id, source, amount, date) values (1, ‘Side-Hustle’, 9000, ‘2023-10-09’):

Mysql> SAVEPOINT s2;

Mysql> update savings set amount=amount + 9000.00 where User\_id =1 and Account\_type = ‘Emergency Funds’;

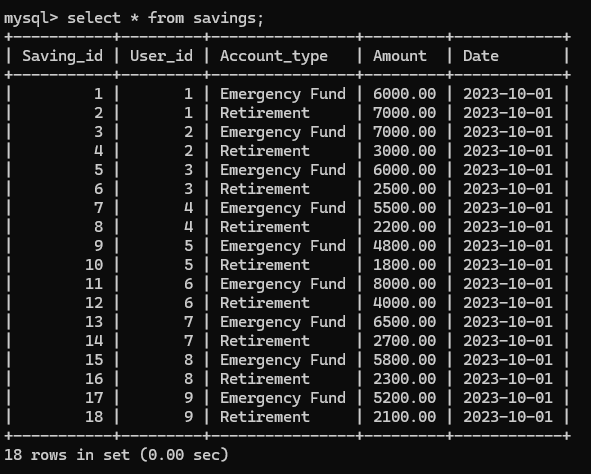
Mysql> COMMIT;



* DEBT PAY OFF AND UPDATE SAVINGS.

Mysql> start transaction;

Mysql> select \* from savings;



Mysql> select \* from debts;

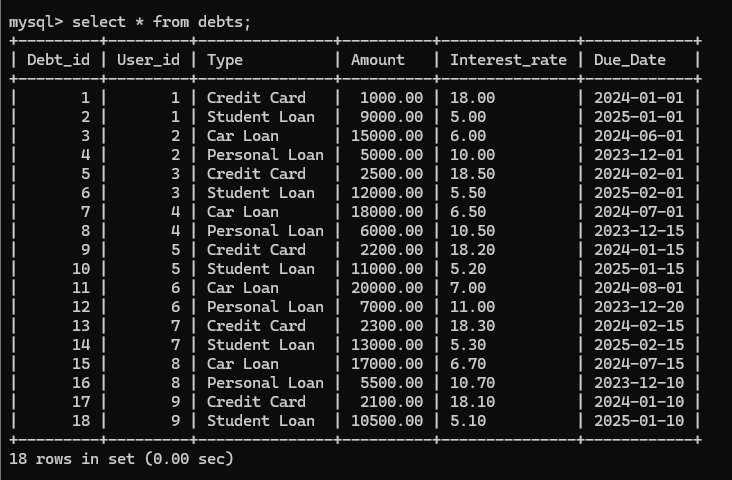
Mysql> SAVEPOINT s3;

Mysql> update savings set amount = amount - 1000.00 where user\_id = 1 and Account\_type = 'emergency fund';

Mysql> SAVEPOINT s4;

Mysql> update debts set amount = amount - 1000.00 where user\_id = 1;

Mysql> COMMIT;



* TRANSPORT FUNDS BETWEEN SAVINGS ACCOUNT

Mysql> start transaction;

Mysql> select \* from savings;

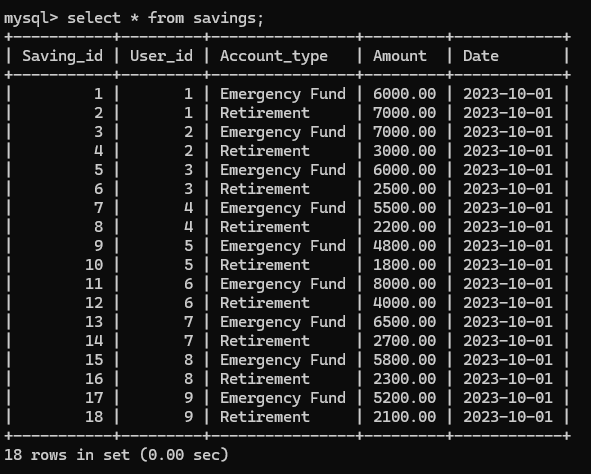
Mysql> SAVEPOINT s5;

Mysql> update savings set amount = amount - 5000.00 where user\_id = 1 and Account\_type = 'emergency fund';

Mysql> SAVEPOINT s6;

Mysql> update savings set amount = amount + 5000.00 where user\_id = 1 and Account\_type = 'Retirement';

Mysql> COMMIT;



* RECORD INVESTMENTS AND UPDATE SAVINGS.

Mysql> start transaction;

Mysql> select \* from savings;

Mysql> select \* from investment;

Mysql> SAVEPOINT s7;

Mysql> update savings set amount = amount - 2000.00 where user\_id = 1 and Account\_type = 'Emergency fund';

Mysql> SAVEPOINT s8;

Mysql> insert into investments (user\_id, type, amount, Return\_rate, date) values (1,'Mutual Funds',2000.00, 600, '2023-10-13');

Mysql> COMMIT;

