Personal finance health analyzer

Introduction:

This project is MySQL base region design to help individuals and analyze and optimize the personal finances. It will include tracking income expenses savings investments and debts.it will provide actionable insights that will help improve financial wellbeing. The system will use MySQL concepts to generate reports, identify financial trends and suggest optimization strategies

Perquisites for this project:

In order to perform this project, there are some prerequisites that the user ha
1.Mysql 8.0 command line client
2.Mysql 8.0 workbench ce
3.My SQL statements
4.Mysql operations
5.Mysql clauses
6.Mysql constraints
7.Mysql subqueries
8.Mysql joins
9.User permissions (grant and revoke)
10.Transactions
These are the concepts are technologies which the user needs to be fundamentally strong

Key features of the project:

This project will contain several important features of real-world finance management such as:

1. Income and expanse tracking
   It is used to track monthly income sources and categorize expenses
2. Savings and investment analysis:
   Monitoring savings accounts and investing portfolio along with calculating returns

3. Debt management it is used track debts credit card loans etc. and calculate interest payments along with debt repayment strategies
4. Financial health scoring:

   It is used to generate a financial health score based on income expenses savings 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 roll base access.It is also used to restrict access on sensitive financial data

Schema:1
In order to perform this project, there are various parameters required

- **Database:**

Create a new database for this project with the project name personalFinanceProject

Schema 2: Tables required

Used to store investment details Table 1 users

Table1: should contain users details (user_id,username,role, password)

Table2: income Use to store income sources of the user(income_id,user_id,source,amount,date)

Table 3: expenses

It is used to store expanse details of the user (expense_id,user_id,category,amount,date)

Table 4: savings

Use to store savings details (saving_id,user_id,account_type,amount,date)

Table 5: investments

(investment_id,user_id,type,amount,return_rate,date)

Table 6 : Debts

Used to store debts details (debt_id,user_id,type,amount,intrest_rate,due_date)

Table 7 :finance health

Used to store financial health score (health_id,user_id,score,date)

Schema 3 : relationships

With the use of constraint keys, the users table must be linked with all the other tables with the use of primary key and foreign key. This is called a one-to-many relationship.

Schema 4: users

1.admin – with admin privileges(all)
2.user – with user privileges(select)

Implementation:

In order to design structure this project and generate analysis based on the data here are steps of implementation for this project

Step 1: Creating database and creating the necessary tables

1.creating database

```
Create database personalfinancehealth;
```

2.create table users

```
create table users (
user_id int auto_increment primary key,
username varchar (100) not null,
role Enum("admin","user") not null,
password varchar(50) not null,
unique(username)
);
```

3.create table income

```
income_id int auto_increment, user_id int,
source varchar(100),
amount decimal(10,2) not null,
date date,
primary key(income_id),
foreign key(user_id) references
users(user_id)
);
```

4.create table expenses

```
            create table expenses (
               expense_id int auto_increment,
               user_id int,
               category varchar(100),
               amount decimal(10,2) not null,
               date date,
               primary key(expense_id),
               foreign key(user_id) references
               users(user_id)
               );
```

5.create table savings

```
            create table savings (
                saving_id int auto_increment,
                user_id int,
                account_type varchar(100),
                amount decimal(10,2) not null,
                date date,
                primary key(saving_id),
                foreign key(user_id) references
                users(user_id)
            );
```

6.create table investment details

```
create table investment_details(
investment_id int auto_increment,
user_id int,
type varchar(100),
amount decimal(10,2) not null,
date date,
Return_ratae decimal(5,2),
primary key(investment_id),
foreign key(user_id) references
users(user_id));
```

7.create table storing debts

```
create table debt_details(
debt_id int auto_increment,
user_id int,
type varchar(100),
amount decimal(10,2) not null,
date date,
Interest_rate decimal(5,2),
Due_date date,
primary key(debt_id),
foreign key(user_id) references
users(user_id)
 );
```

8.create table storing details of financial health scores

```
create table financial_score(
health_id int auto_increment,
user_id int,
score decimal(5,2),
date date,
primary key(health_id),
foreign key(user_id) references
users(user_id)
);
```

<u>Step 2</u>

Inserting values more then hundred rows in all 7 tables to perform the analysis. Apart from the autoincrement columns all the columns have to be filled with data

<u>Table 1: users</u>

There should be 10 users among those 10 1 should be named as admin where as the rest 9 will be users

```
INSERT INTO Users (username, role, password) VALUES
('john_doe', 'user', 'hashed_password_123'),
('jane_smith', 'user', 'hashed_password_456'),
('alice_wang', 'user', 'hashed_password_789'),
('bob_johnson', 'user', 'hashed_password_101'),
('emily_davis', 'user', 'hashed_password_112'),
('michael_brown', 'user', 'hashed_password_131'),
('sarah_miller', 'user', 'hashed_password_415'),
('david_wilson', 'user', 'hashed_password_161'),
('linda_moore', 'user', 'hashed_password_718'),
('admin_user', 'admin', 'hashed_password_919');
```

<u>Table 2: income</u>

For each user there are different streams of income along with the profit or amount generated. They have been assigned to a particular user id that matches the user id’s from the table users

```
INSERT INTO Income (user_id, source, amount, date) VALUES
(1, 'Salary', 3000.00, '2023-10-01'),
(1, 'Freelance', 500.00, '2023-10-15'),
(1, 'Bonus', 1000.00, '2023-10-30'),
(2, 'Salary', 4000.00, '2023-10-01'),
(2, 'Dividends', 200.00, '2023-10-10'),
(2, 'Bonus', 800.00, '2023-10-25'),
(3, 'Salary', 3500.00, '2023-10-01'),
(3, 'Freelance', 600.00, '2023-10-20'),
(3, 'Bonus', 700.00, '2023-10-31'),
(4, 'Salary', 4500.00, '2023-10-01'),
(4, 'Dividends', 300.00, '2023-10-15'),
(4, 'Bonus', 900.00, '2023-10-30'),
(5, 'Salary', 3200.00, '2023-10-01'),
(5, 'Freelance', 400.00, '2023-10-10'),
(5, 'Bonus', 600.00, '2023-10-25'),
(6, 'Salary', 5000.00, '2023-10-01'),
(6, 'Dividends', 250.00, '2023-10-15'),
(6, 'Bonus', 1200.00, '2023-10-31'),
(7, 'Salary', 3800.00, '2023-10-01'),
(7, 'Freelance', 700.00, '2023-10-20'),
(7, 'Bonus', 800.00, '2023-10-30'),
(8, 'Salary', 4200.00, '2023-10-01'),
(8, 'Dividends', 150.00, '2023-10-10'),
(8, 'Bonus', 1000.00, '2023-10-25'),
(9, 'Salary', 3600.00, '2023-10-01'),
(9, 'Freelance', 550.00, '2023-10-15'),
(9, 'Bonus', 750.00, '2023-10-31');
```

Table3: expenses

```
INSERT INTO Expenses (user_id, category, amount, date) VALUES
-- John Doe
(1, 'Rent', 1200.00, '2023-10-01'),
(1, 'Groceries', 300.00, '2023-10-05'),
(1, 'Utilities', 150.00, '2023-10-07'),
(1, 'Entertainment', 200.00, '2023-10-10'),
(1, 'Transportation', 100.00, '2023-10-15'),

-- Jane Smith
(2, 'Rent', 1500.00, '2023-10-01'),
(2, 'Groceries', 400.00, '2023-10-05'),
(2, 'Utilities', 200.00, '2023-10-07'),
(2, 'Entertainment', 250.00, '2023-10-10'),
(2, 'Transportation', 150.00, '2023-10-15'),
-- Alice Wang
(3, 'Rent', 1300.00, '2023-10-01'),
(3, 'Groceries', 350.00, '2023-10-05'),
(3, 'Utilities', 180.00, '2023-10-07'),
(3, 'Entertainment', 220.00, '2023-10-10'),
(3, 'Transportation', 120.00, '2023-10-15'),

-- Bob Johnson
(4, 'Rent', 1400.00, '2023-10-01'),
(4, 'Groceries', 320.00, '2023-10-05'),
(4, 'Utilities', 160.00, '2023-10-07'),
(4, 'Entertainment', 210.00, '2023-10-10'),
(4, 'Transportation', 110.00, '2023-10-15'),

-- Emily Davis
(5, 'Rent', 1100.00, '2023-10-01'),
(5, 'Groceries', 280.00, '2023-10-05'),
(5, 'Utilities', 140.00, '2023-10-07'),
(5, 'Entertainment', 190.00, '2023-10-10'),
(5, 'Transportation', 90.00, '2023-10-15'),

-- Michael Brown
(6, 'Rent', 1600.00, '2023-10-01'),
(6, 'Groceries', 450.00, '2023-10-05'),
(6, 'Utilities', 220.00, '2023-10-07'),
(6, 'Entertainment', 300.00, '2023-10-10'),
(6, 'Transportation', 200.00, '2023-10-15'),

-- Sarah Miller
(7, 'Rent', 1350.00, '2023-10-01'),
(7, 'Groceries', 330.00, '2023-10-05'),
(7, 'Utilities', 170.00, '2023-10-07'),
(7, 'Entertainment', 230.00, '2023-10-10'),
```

(7, 'Transportation', 130.00, '2023-10-15'),

-- David Wilson

(8, 'Rent', 1450.00, '2023-10-01'),

(8, 'Groceries', 340.00, '2023-10-05'),

(8, 'Utilities', 190.00, '2023-10-07'),

(8, 'Entertainment', 240.00, '2023-10-10'),

(8, 'Transportation', 140.00, '2023-10-15'),

-- Linda Moore

(9, 'Rent', 1250.00, '2023-10-01'),

(9, 'Groceries', 310.00, '2023-10-05'),

(9, 'Utilities', 160.00, '2023-10-07'),

(9, 'Entertainment', 210.00, '2023-10-10'),

(9, 'Transportation', 110.00, '2023-10-15');

Table 4: savings

INSERT INTO Savings (user_id, account_type, amount, date) VALUES

-- John Doe

(1, 'Emergency Fund', 5000.00, '2023-10-01'),

(1, 'Retirement', 2000.00, '2023-10-01'),

-- Jane Smith

(2, 'Emergency Fund', 7000.00, '2023-10-01'),

(2, 'Retirement', 3000.00, '2023-10-01'),

-- Alice Wang

(3, 'Emergency Fund', 6000.00, '2023-10-01'),

(3, 'Retirement', 2500.00, '2023-10-01'),

-- Bob Johnson

(4, 'Emergency Fund', 5500.00, '2023-10-01'),

(4, 'Retirement', 2200.00, '2023-10-01'),

```
-- Emily Davis
(5, 'Emergency Fund', 4800.00, '2023-10-01'),
(5, 'Retirement', 1800.00, '2023-10-01'),

-- Michael Brown
(6, 'Emergency Fund', 8000.00, '2023-10-01'),
(6, 'Retirement', 4000.00, '2023-10-01'),

-- Sarah Miller
(7, 'Emergency Fund', 6500.00, '2023-10-01'),
(7, 'Retirement', 2700.00, '2023-10-01'),

-- David Wilson
(8, 'Emergency Fund', 5800.00, '2023-10-01'),
(8, 'Retirement', 2300.00, '2023-10-01'),

-- Linda Moore
(9, 'Emergency Fund', 5200.00, '2023-10-01'),
(9, 'Retirement', 2100.00, '2023-10-01');
```

Table5: investments

This table is used provide the value of each investment done by the user and the return they have received

```
INSERT INTO Investment_details (user_id, type, amount, return_rate, date)
VALUES
-- John Doe
(1, 'Stocks', 10000.00, 8.50, '2023-10-01'),
(1, 'Bonds', 5000.00, 3.00, '2023-10-01'),

-- Jane Smith
(2, 'Mutual Funds', 15000.00, 6.00, '2023-10-01'),
(2, 'Real Estate', 20000.00, 5.00, '2023-10-01'),
-- Alice Wang
(3, 'Stocks', 12000.00, 8.00, '2023-10-01'),
(3, 'Bonds', 6000.00, 3.50, '2023-10-01'),

-- Bob Johnson
(4, 'Mutual Funds', 18000.00, 6.50, '2023-10-01'),
(4, 'Real Estate', 25000.00, 5.50, '2023-10-01'),

-- Emily Davis
(5, 'Stocks', 11000.00, 8.20, '2023-10-01'),
(5, 'Bonds', 5500.00, 3.20, '2023-10-01'),

-- Michael Brown
(6, 'Mutual Funds', 20000.00, 7.00, '2023-10-01'),
(6, 'Real Estate', 30000.00, 6.00, '2023-10-01'),

-- Sarah Miller
(7, 'Stocks', 13000.00, 8.30, '2023-10-01'),
(7, 'Bonds', 7000.00, 3.30, '2023-10-01'),

-- David Wilson
(8, 'Mutual Funds', 17000.00, 6.70, '2023-10-01'),
(8, 'Real Estate', 22000.00, 5.70, '2023-10-01'),

-- Linda Moore
(9, 'Stocks', 10500.00, 8.10, '2023-10-01'),
(9, 'Bonds', 5200.00, 3.10, '2023-10-01');
```

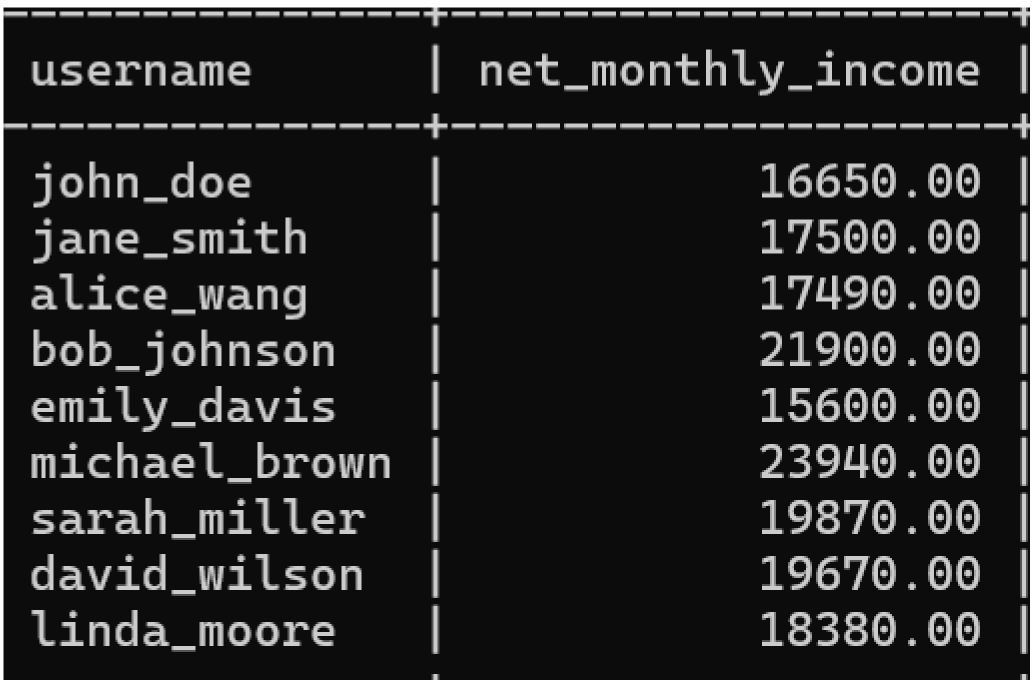
<u>Table 6:debt_details</u>

```
INSERT INTO Debt_details(user_id, type, amount, interest_ratedes, due_date)
VALUES
-- John Doe
(1, 'Credit Card', 2000.00, 18.00, '2024-01-01'),
(1, 'Student Loan', 10000.00, 5.00, '2025-01-01'),

-- Jane Smith
(2, 'Car Loan', 15000.00, 6.00, '2024-06-01'),
(2, 'Personal Loan', 5000.00, 10.00, '2023-12-01'),

-- Alice Wang
(3, 'Credit Card', 2500.00, 18.50, '2024-02-01'),
(3, 'Student Loan', 12000.00, 5.50, '2025-02-01'),

-- Bob Johnson
(4, 'Car Loan', 18000.00, 6.50, '2024-07-01'),
(4, 'Personal Loan', 6000.00, 10.50, '2023-12-15'),

-- Emily Davis
(5, 'Credit Card', 2200.00, 18.20, '2024-01-15'),
(5, 'Student Loan', 11000.00, 5.20, '2025-01-15'),

-- Michael Brown
(6, 'Car Loan', 20000.00, 7.00, '2024-08-01'),
(6, 'Personal Loan', 7000.00, 11.00, '2023-12-20'),

-- Sarah Miller
(7, 'Credit Card', 2300.00, 18.30, '2024-02-15'),
(7, 'Student Loan', 13000.00, 5.30, '2025-02-15'),

-- David Wilson
(8, 'Car Loan', 17000.00, 6.70, '2024-07-15'),
(8, 'Personal Loan', 5500.00, 10.70, '2023-12-10'),
-- Linda Moore
(9, 'Credit Card', 2100.00, 18.10, '2024-01-10'),
(9, 'Student Loan', 10500.00, 5.10, '2025-01-10');
```

Requirement 1:

Calculating monthly net income for each user in order to fetch the details for this requirement 3 tables have to be users income expanses all the tables have to be joined

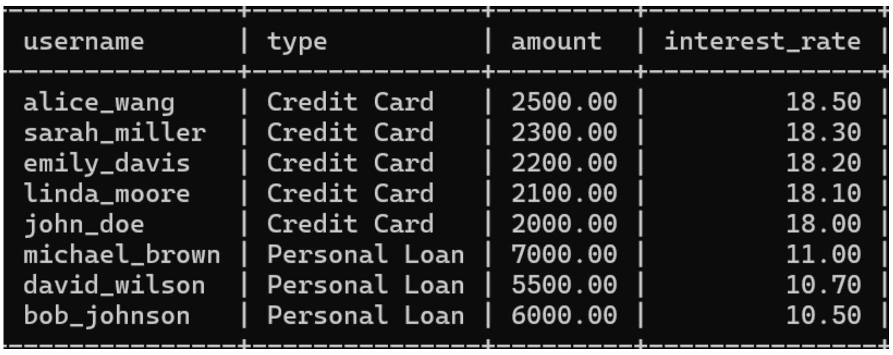
```
select users.username,sum(income.amount)-sum(expenses.amount)
    -> as net_monthly_income from users
    -> left join income on users.user_id=income.user_id
    -> left join expenses on users.user_id=expenses.user_id
    -> where income.date between "2023-10-01"and"2023-10-31"
    -> group by users.user_id;
```



```
+---------------+--------------------+
| username      | net_monthly_income |
+---------------+--------------------+
| john_doe      |           16650.00 |
| jane_smith    |           17500.00 |
| alice_wang    |           17490.00 |
| bob_johnson   |           21900.00 |
| emily_davis   |           15600.00 |
| michael_brown |           23940.00 |
| sarah_miller  |           19870.00 |
| david_wilson  |           19670.00 |
| linda_moore   |           18380.00 |
```

Requirement 2: identity high interest debts

Select users.username,debt_details.type,debt_details.amount,debt_details.interest_rate from debt_details
    -> inner join users on debt_details.user_id=users.user_id
    -> where debt_details.interest_rate > 10
    -> order by debt_details.interest_rate desc;

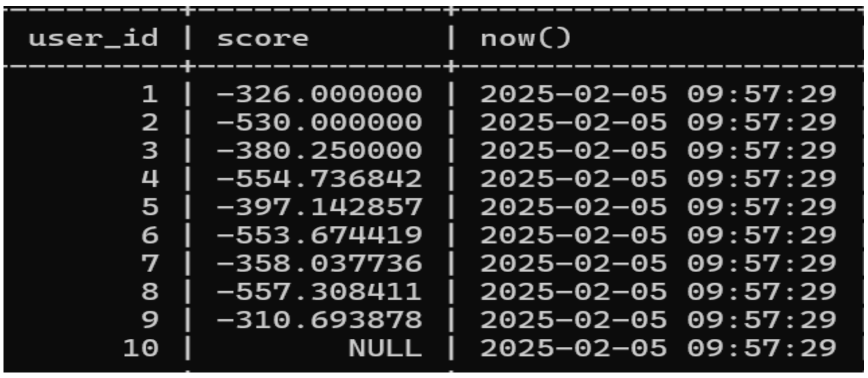


```
+--------------+---------------+---------+---------------+
| username     | type          | amount  | interest_rate |
+--------------+---------------+---------+---------------+
| alice_wang   | Credit Card   | 2500.00 |         18.50 |
| sarah_miller | Credit Card   | 2300.00 |         18.30 |
| emily_davis  | Credit Card   | 2200.00 |         18.20 |
| linda_moore  | Credit Card   | 2100.00 |         18.10 |
| john_doe     | Credit Card   | 2000.00 |         18.00 |
| michael_brown| Personal Loan | 7000.00 |         11.00 |
| david_wilson | Personal Loan | 5500.00 |         10.70 |
| bob_johnson  | Personal Loan | 6000.00 |         10.50 |
+--------------+---------------+---------+---------------+
```

Requirement:3 generating financial health score

In this requirement the user will generate the financial health score

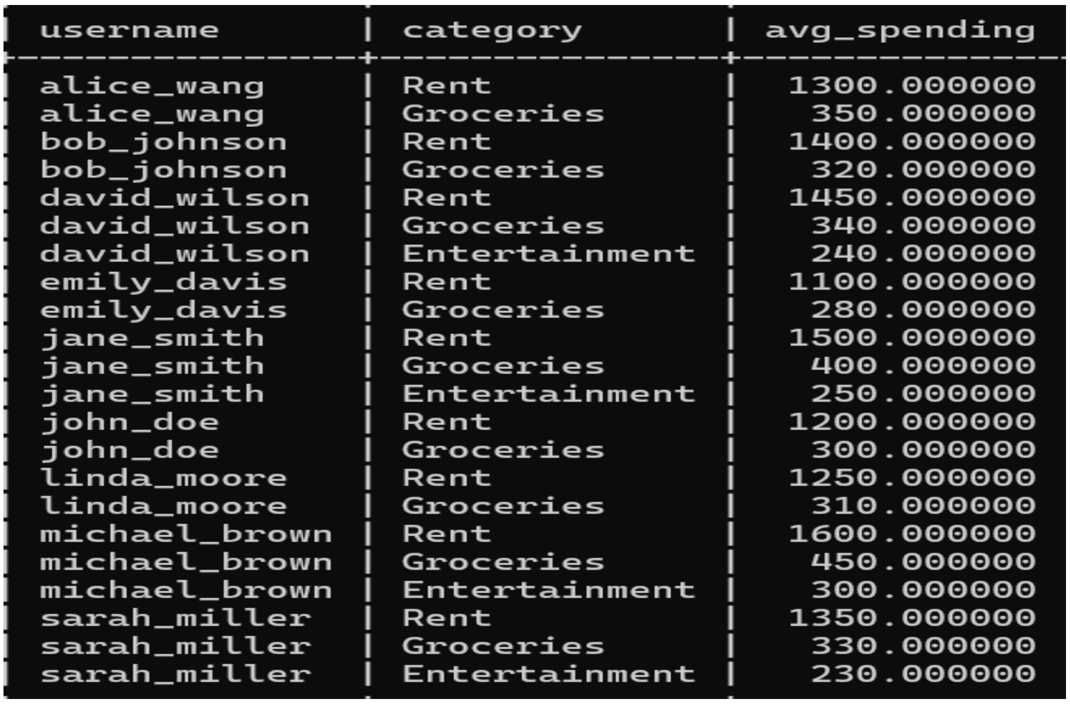
select users.user_id,(sum(income.amount)-sum(expenses.amount)-sum(debt_details.amount))/sum(income.amount)*100
    -> as score,now()
    -> from users
    -> left join income on users.user_id=income.user_id
    -> left join expenses on users.user_id=expenses.user_id
    -> left join debt_details on users.user_id=debt_details.user_id
    -> group by users.user_id;



```
+---------+-------------+---------------------+
| user_id | score       | now()               |
+---------+-------------+---------------------+
|       1 | -326.000000 | 2025-02-05 09:57:29 |
|       2 | -530.000000 | 2025-02-05 09:57:29 |
|       3 | -380.250000 | 2025-02-05 09:57:29 |
|       4 | -554.736842 | 2025-02-05 09:57:29 |
|       5 | -397.142857 | 2025-02-05 09:57:29 |
|       6 | -553.674419 | 2025-02-05 09:57:29 |
|       7 | -358.037736 | 2025-02-05 09:57:29 |
|       8 | -557.308411 | 2025-02-05 09:57:29 |
|       9 | -310.693878 | 2025-02-05 09:57:29 |
|      10 |        NULL | 2025-02-05 09:57:29 |
```

Requirement 4: Budget Expenses
In this requirement the users will be provided with their average spending each expense

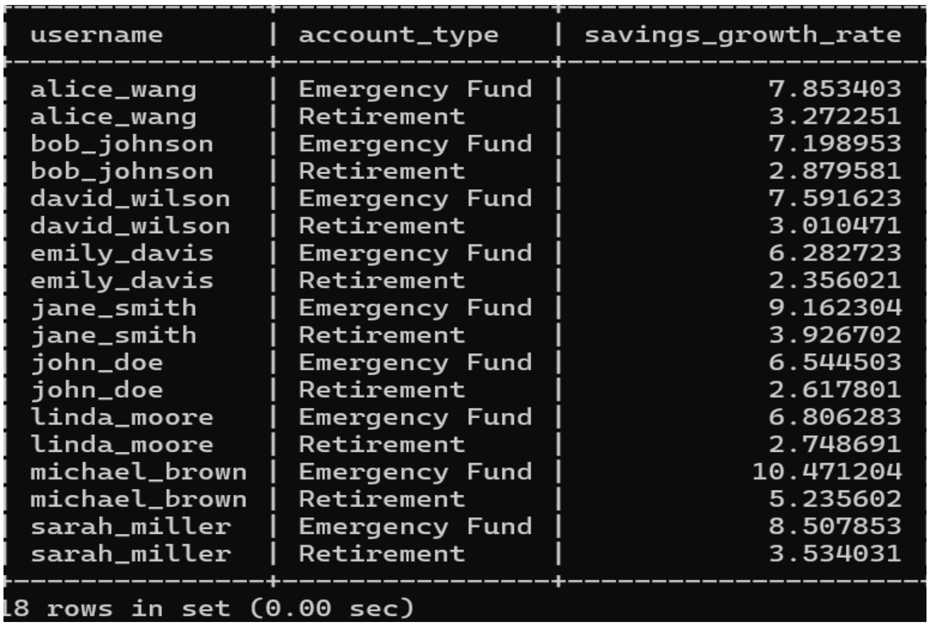
```
select users.username,expenses.category,avg(expenses.amount)
    -> as avg_spending
    -> from expenses
    -> inner join users on expenses.user_id=users.user_id
    -> group by users.user_id,expenses.category
    -> having avg_spending > (select avg(amount)from
    -> expenses where category ="entertainment");
```



```
| username       | category      | avg_spending
+----------------+---------------+-------------
| alice_wang     | Rent          |  1300.000000
| alice_wang     | Groceries     |   350.000000
| bob_johnson    | Rent          |  1400.000000
| bob_johnson    | Groceries     |   320.000000
| david_wilson   | Rent          |  1450.000000
| david_wilson   | Groceries     |   340.000000
| david_wilson   | Entertainment |   240.000000
| emily_davis    | Rent          |  1100.000000
| emily_davis    | Groceries     |   280.000000
| jane_smith     | Rent          |  1500.000000
| jane_smith     | Groceries     |   400.000000
| jane_smith     | Entertainment |   250.000000
| john_doe       | Rent          |  1200.000000
| john_doe       | Groceries     |   300.000000
| linda_moore    | Rent          |  1250.000000
| linda_moore    | Groceries     |   310.000000
| michael_brown  | Rent          |  1600.000000
| michael_brown  | Groceries     |   450.000000
| michael_brown  | Entertainment |   300.000000
| sarah_miller   | Rent          |  1350.000000
| sarah_miller   | Groceries     |   330.000000
| sarah_miller   | Entertainment |   230.000000
```

Requirement 5: calculating savings growth rate

```
Select  users.username,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;
```

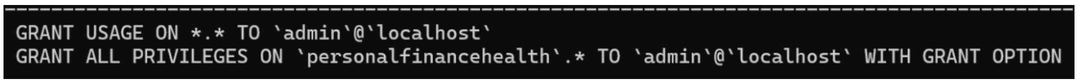


```
+----------------+----------------+---------------------+
| username       | account_type   | savings_growth_rate |
+----------------+----------------+---------------------+
| alice_wang     | Emergency Fund |            7.853403 |
| alice_wang     | Retirement     |            3.272251 |
| bob_johnson    | Emergency Fund |            7.198953 |
| bob_johnson    | Retirement     |            2.879581 |
| david_wilson   | Emergency Fund |            7.591623 |
| david_wilson   | Retirement     |            3.010471 |
| emily_davis    | Emergency Fund |            6.282723 |
| emily_davis    | Retirement     |            2.356021 |
| jane_smith     | Emergency Fund |            9.162304 |
| jane_smith     | Retirement     |            3.926702 |
| john_doe       | Emergency Fund |            6.544503 |
| john_doe       | Retirement     |            2.617801 |
| linda_moore    | Emergency Fund |            6.806283 |
| linda_moore    | Retirement     |            2.748691 |
| michael_brown  | Emergency Fund |           10.471204 |
| michael_brown  | Retirement     |            5.235602 |
| sarah_miller   | Emergency Fund |            8.507853 |
| sarah_miller   | Retirement     |            3.534031 |
+----------------+----------------+---------------------+
18 rows in set (0.00 sec)
```

Step 4: user permissions:

create user "admin"@"localhost"identified by "admin123";

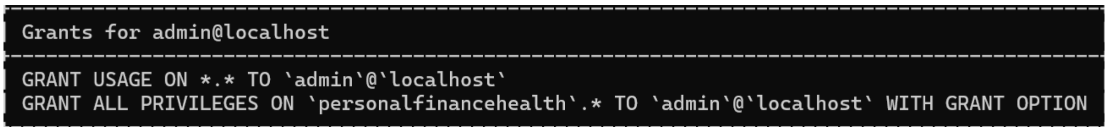
show grants for "admin"@"localhost";



```
GRANT USAGE ON *.* TO `admin`@`localhost`
GRANT ALL PRIVILEGES ON `personalfinancehealth`.* TO `admin`@`localhost` WITH GRANT OPTION
```

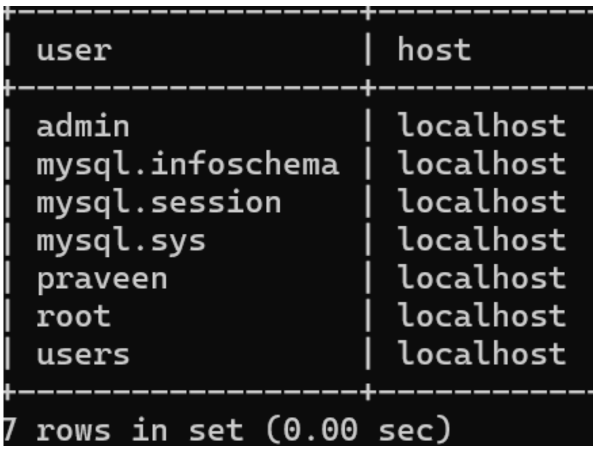
grant all privileges on personalfinancehealth.* to "admin"@"localhost" with grant option;

show grants for "admin"@"localhost";



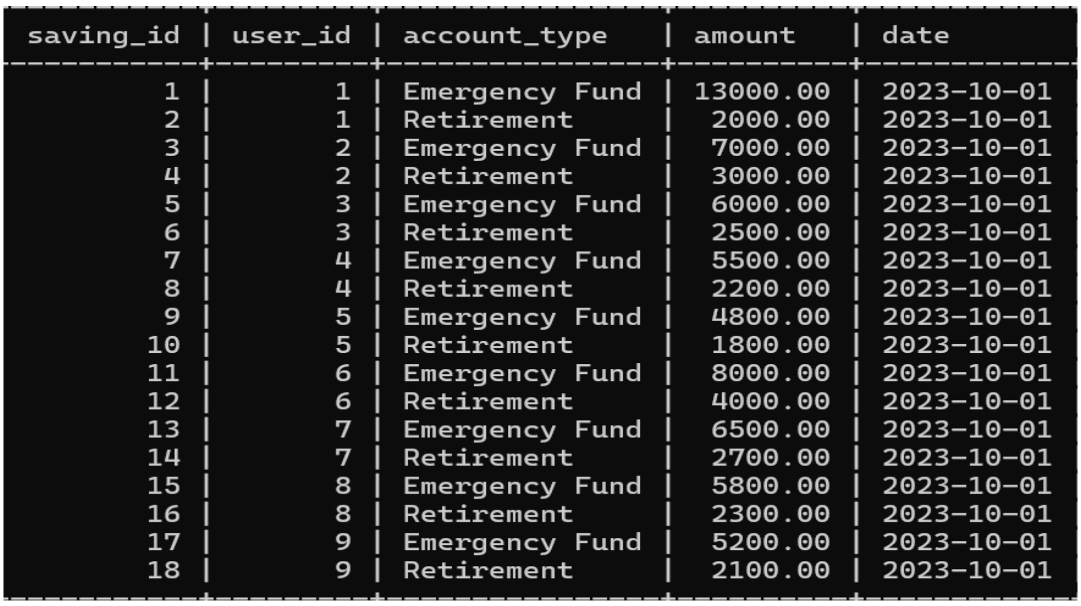
```
Grants for admin@localhost
----------------------------------------------------------------------------------------------
GRANT USAGE ON *.* TO `admin`@`localhost`
GRANT ALL PRIVILEGES ON `personalfinancehealth`.* TO `admin`@`localhost` WITH GRANT OPTION
```

select user,host from mysql.user;

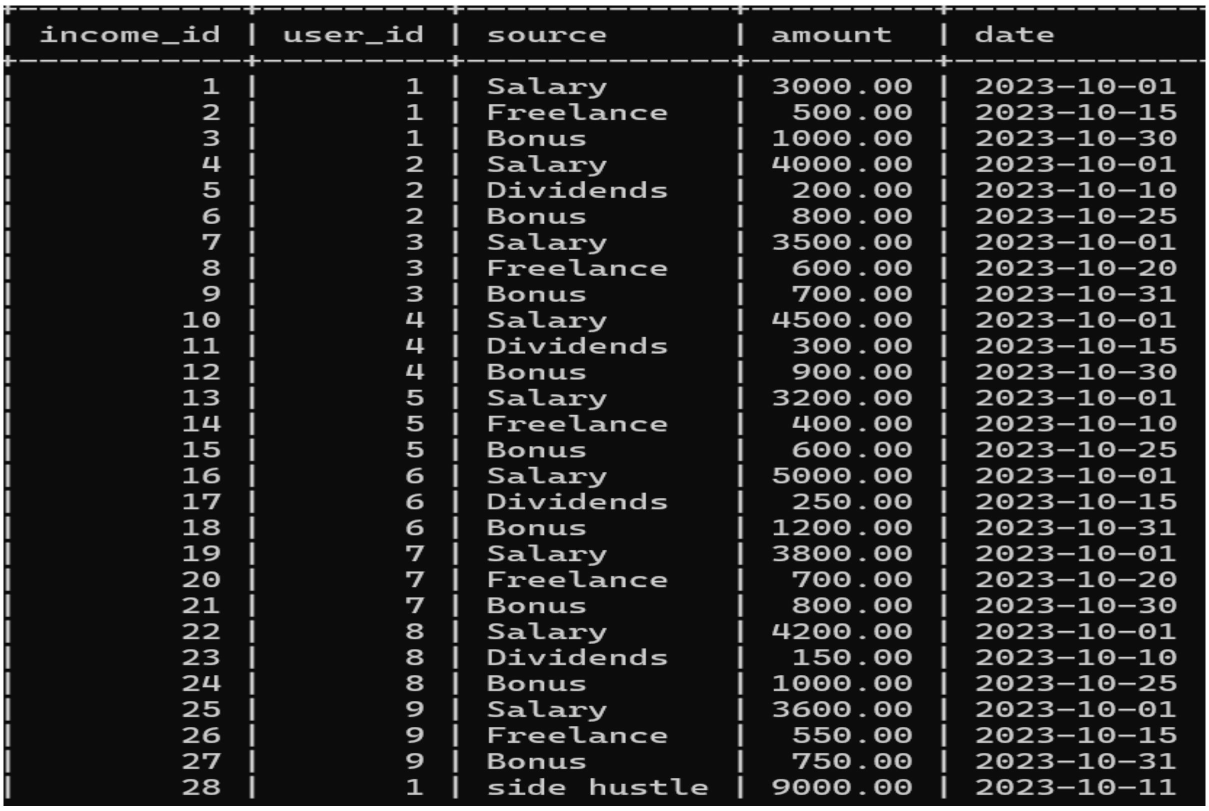


```
+------------------+-----------+
| user             | host      |
+------------------+-----------+
| admin            | localhost |
| mysql.infoschema | localhost |
| mysql.session    | localhost |
| mysql.sys        | localhost |
| praveen          | localhost |
| root             | localhost |
| users            | localhost |
+------------------+-----------+
7 rows in set (0.00 sec)
```

Transaction1: record income and update savings

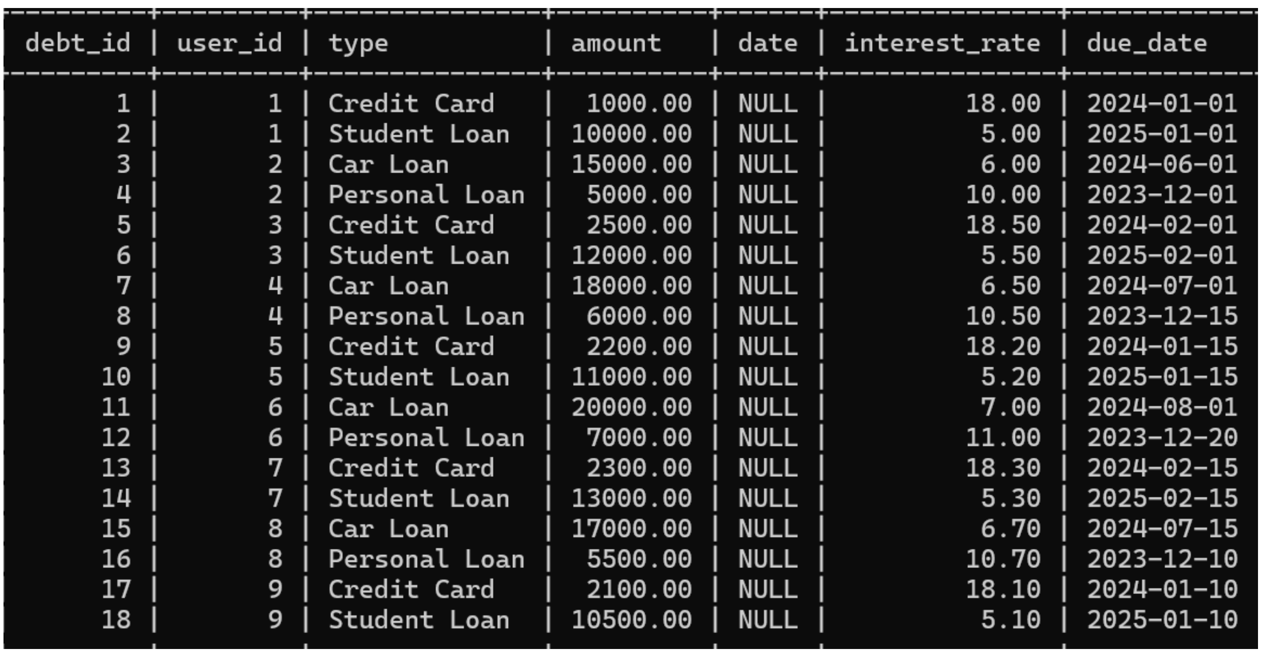


```
-----------+---------+----------------+----------+------------
 saving_id | user_id | account_type   | amount   | date
-----------+---------+----------------+----------+------------
         1 |       1 | Emergency Fund | 13000.00 | 2023-10-01
         2 |       1 | Retirement     |  2000.00 | 2023-10-01
         3 |       2 | Emergency Fund |  7000.00 | 2023-10-01
         4 |       2 | Retirement     |  3000.00 | 2023-10-01
         5 |       3 | Emergency Fund |  6000.00 | 2023-10-01
         6 |       3 | Retirement     |  2500.00 | 2023-10-01
         7 |       4 | Emergency Fund |  5500.00 | 2023-10-01
         8 |       4 | Retirement     |  2200.00 | 2023-10-01
         9 |       5 | Emergency Fund |  4800.00 | 2023-10-01
        10 |       5 | Retirement     |  1800.00 | 2023-10-01
        11 |       6 | Emergency Fund |  8000.00 | 2023-10-01
        12 |       6 | Retirement     |  4000.00 | 2023-10-01
        13 |       7 | Emergency Fund |  6500.00 | 2023-10-01
        14 |       7 | Retirement     |  2700.00 | 2023-10-01
        15 |       8 | Emergency Fund |  5800.00 | 2023-10-01
        16 |       8 | Retirement     |  2300.00 | 2023-10-01
        17 |       9 | Emergency Fund |  5200.00 | 2023-10-01
        18 |       9 | Retirement     |  2100.00 | 2023-10-01
-----------+---------+----------------+----------+------------
```



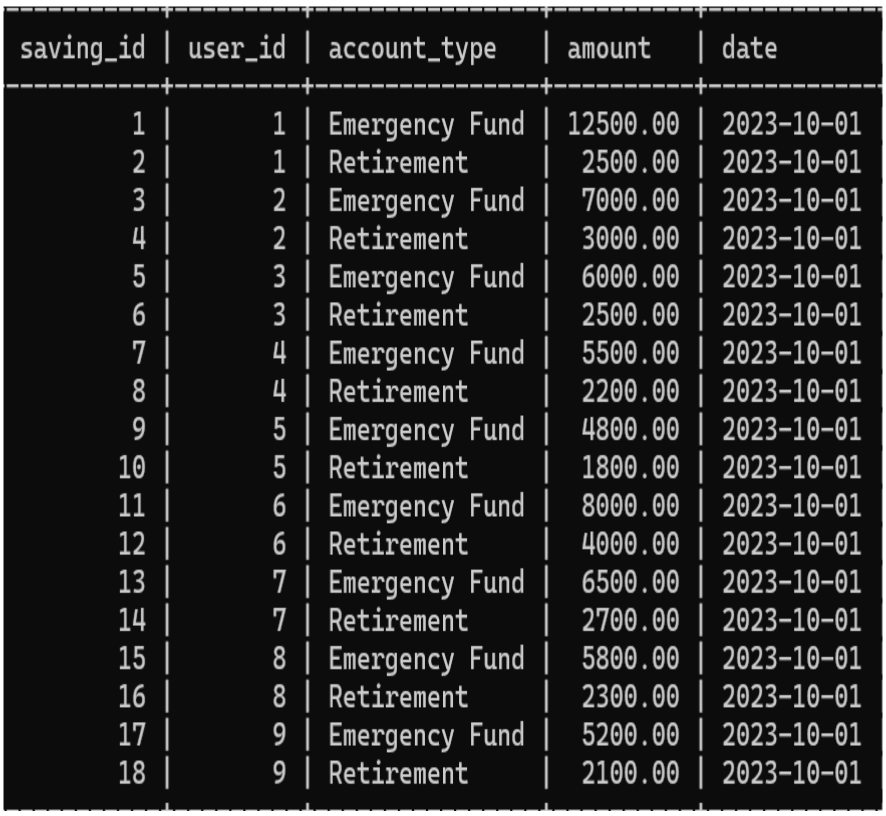
```
+-----------+---------+-------------+---------+------------
| income_id | user_id | source      | amount  | date
+-----------+---------+-------------+---------+------------
|         1 |       1 | Salary      | 3000.00 | 2023-10-01
|         2 |       1 | Freelance   |  500.00 | 2023-10-15
|         3 |       1 | Bonus       | 1000.00 | 2023-10-30
|         4 |       2 | Salary      | 4000.00 | 2023-10-01
|         5 |       2 | Dividends   |  200.00 | 2023-10-10
|         6 |       2 | Bonus       |  800.00 | 2023-10-25
|         7 |       3 | Salary      | 3500.00 | 2023-10-01
|         8 |       3 | Freelance   |  600.00 | 2023-10-20
|         9 |       3 | Bonus       |  700.00 | 2023-10-31
|        10 |       4 | Salary      | 4500.00 | 2023-10-01
|        11 |       4 | Dividends   |  300.00 | 2023-10-15
|        12 |       4 | Bonus       |  900.00 | 2023-10-30
|        13 |       5 | Salary      | 3200.00 | 2023-10-01
|        14 |       5 | Freelance   |  400.00 | 2023-10-10
|        15 |       5 | Bonus       |  600.00 | 2023-10-25
|        16 |       6 | Salary      | 5000.00 | 2023-10-01
|        17 |       6 | Dividends   |  250.00 | 2023-10-15
|        18 |       6 | Bonus       | 1200.00 | 2023-10-31
|        19 |       7 | Salary      | 3800.00 | 2023-10-01
|        20 |       7 | Freelance   |  700.00 | 2023-10-20
|        21 |       7 | Bonus       |  800.00 | 2023-10-30
|        22 |       8 | Salary      | 4200.00 | 2023-10-01
|        23 |       8 | Dividends   |  150.00 | 2023-10-10
|        24 |       8 | Bonus       | 1000.00 | 2023-10-25
|        25 |       9 | Salary      | 3600.00 | 2023-10-01
|        26 |       9 | Freelance   |  550.00 | 2023-10-15
|        27 |       9 | Bonus       |  750.00 | 2023-10-31
|        28 |       1 | side hustle | 9000.00 | 2023-10-11
```

Transaction 2: debt payoff and update savings



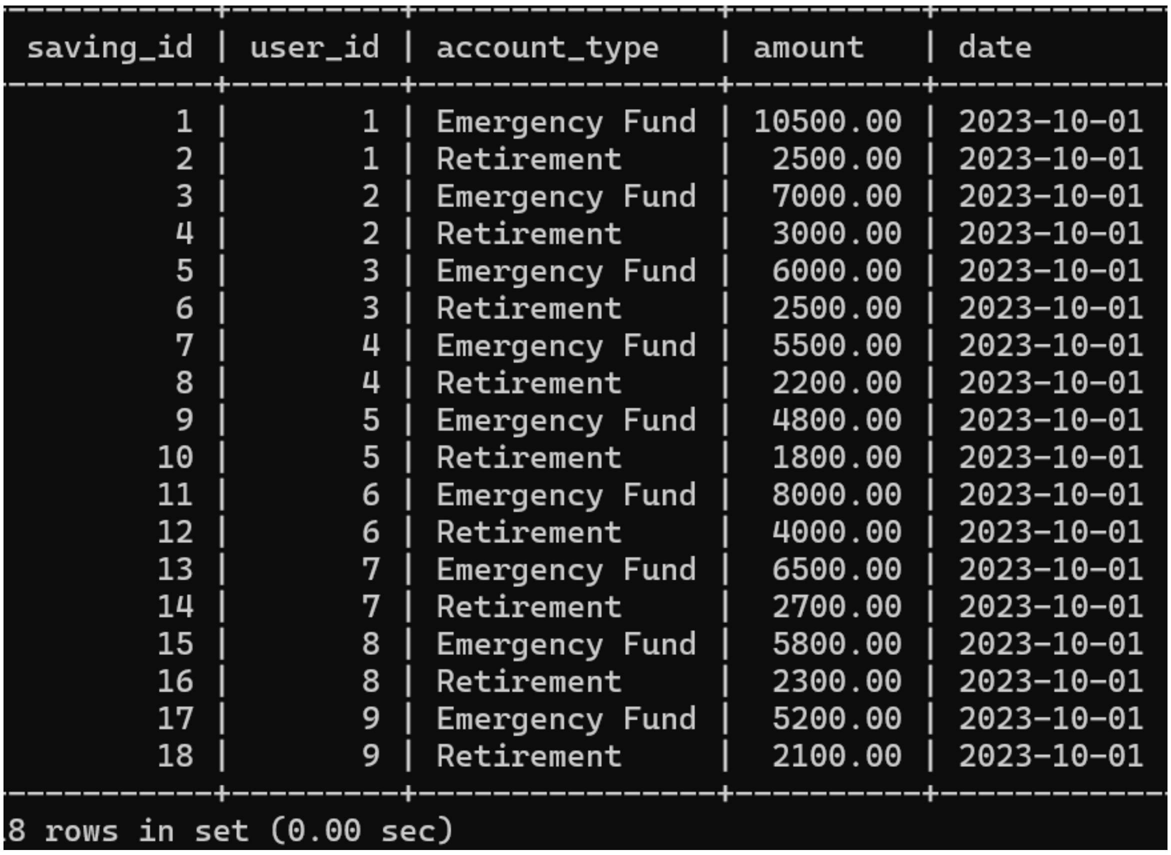
```
---------+---------+---------------+----------+------+---------------+------------
 debt_id | user_id | type          | amount   | date | interest_rate | due_date
---------+---------+---------------+----------+------+---------------+------------
       1 |       1 | Credit Card   |  1000.00 | NULL |         18.00 | 2024-01-01
       2 |       1 | Student Loan  | 10000.00 | NULL |          5.00 | 2025-01-01
       3 |       2 | Car Loan      | 15000.00 | NULL |          6.00 | 2024-06-01
       4 |       2 | Personal Loan |  5000.00 | NULL |         10.00 | 2023-12-01
       5 |       3 | Credit Card   |  2500.00 | NULL |         18.50 | 2024-02-01
       6 |       3 | Student Loan  | 12000.00 | NULL |          5.50 | 2025-02-01
       7 |       4 | Car Loan      | 18000.00 | NULL |          6.50 | 2024-07-01
       8 |       4 | Personal Loan |  6000.00 | NULL |         10.50 | 2023-12-15
       9 |       5 | Credit Card   |  2200.00 | NULL |         18.20 | 2024-01-15
      10 |       5 | Student Loan  | 11000.00 | NULL |          5.20 | 2025-01-15
      11 |       6 | Car Loan      | 20000.00 | NULL |          7.00 | 2024-08-01
      12 |       6 | Personal Loan |  7000.00 | NULL |         11.00 | 2023-12-20
      13 |       7 | Credit Card   |  2300.00 | NULL |         18.30 | 2024-02-15
      14 |       7 | Student Loan  | 13000.00 | NULL |          5.30 | 2025-02-15
      15 |       8 | Car Loan      | 17000.00 | NULL |          6.70 | 2024-07-15
      16 |       8 | Personal Loan |  5500.00 | NULL |         10.70 | 2023-12-10
      17 |       9 | Credit Card   |  2100.00 | NULL |         18.10 | 2024-01-10
      18 |       9 | Student Loan  | 10500.00 | NULL |          5.10 | 2025-01-10
```

Transaction 3:transfer funds between savings account

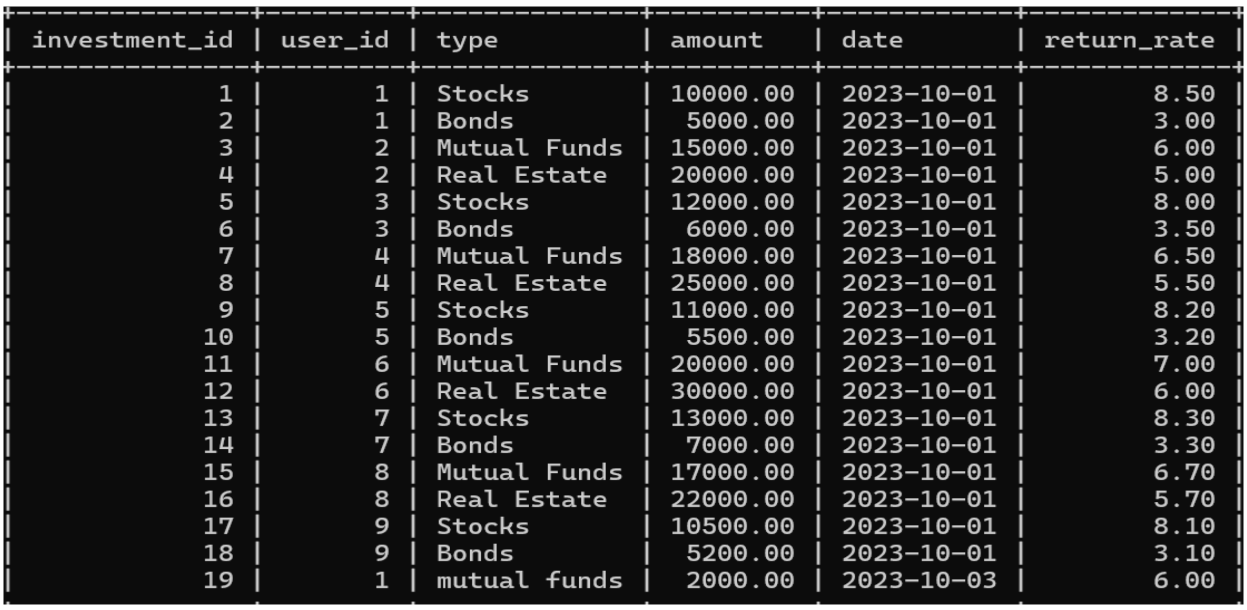


```
-----------+---------+----------------+----------+------------
 saving_id | user_id | account_type   | amount   | date
-----------+---------+----------------+----------+------------
         1 |       1 | Emergency Fund | 12500.00 | 2023-10-01
         2 |       1 | Retirement     |  2500.00 | 2023-10-01
         3 |       2 | Emergency Fund |  7000.00 | 2023-10-01
         4 |       2 | Retirement     |  3000.00 | 2023-10-01
         5 |       3 | Emergency Fund |  6000.00 | 2023-10-01
         6 |       3 | Retirement     |  2500.00 | 2023-10-01
         7 |       4 | Emergency Fund |  5500.00 | 2023-10-01
         8 |       4 | Retirement     |  2200.00 | 2023-10-01
         9 |       5 | Emergency Fund |  4800.00 | 2023-10-01
        10 |       5 | Retirement     |  1800.00 | 2023-10-01
        11 |       6 | Emergency Fund |  8000.00 | 2023-10-01
        12 |       6 | Retirement     |  4000.00 | 2023-10-01
        13 |       7 | Emergency Fund |  6500.00 | 2023-10-01
        14 |       7 | Retirement     |  2700.00 | 2023-10-01
        15 |       8 | Emergency Fund |  5800.00 | 2023-10-01
        16 |       8 | Retirement     |  2300.00 | 2023-10-01
        17 |       9 | Emergency Fund |  5200.00 | 2023-10-01
        18 |       9 | Retirement     |  2100.00 | 2023-10-01
```

Transaction 4: record investments and update savings



```
-----------+---------+----------------+----------+------------
 saving_id | user_id | account_type   | amount   | date
-----------+---------+----------------+----------+------------
         1 |       1 | Emergency Fund | 10500.00 | 2023-10-01
         2 |       1 | Retirement     |  2500.00 | 2023-10-01
         3 |       2 | Emergency Fund |  7000.00 | 2023-10-01
         4 |       2 | Retirement     |  3000.00 | 2023-10-01
         5 |       3 | Emergency Fund |  6000.00 | 2023-10-01
         6 |       3 | Retirement     |  2500.00 | 2023-10-01
         7 |       4 | Emergency Fund |  5500.00 | 2023-10-01
         8 |       4 | Retirement     |  2200.00 | 2023-10-01
         9 |       5 | Emergency Fund |  4800.00 | 2023-10-01
        10 |       5 | Retirement     |  1800.00 | 2023-10-01
        11 |       6 | Emergency Fund |  8000.00 | 2023-10-01
        12 |       6 | Retirement     |  4000.00 | 2023-10-01
        13 |       7 | Emergency Fund |  6500.00 | 2023-10-01
        14 |       7 | Retirement     |  2700.00 | 2023-10-01
        15 |       8 | Emergency Fund |  5800.00 | 2023-10-01
        16 |       8 | Retirement     |  2300.00 | 2023-10-01
        17 |       9 | Emergency Fund |  5200.00 | 2023-10-01
        18 |       9 | Retirement     |  2100.00 | 2023-10-01
-----------+---------+----------------+----------+------------
8 rows in set (0.00 sec)
```



```
+---------------+---------+--------------+----------+------------+-------------+
| investment_id | user_id | type         | amount   | date       | return_rate |
+---------------+---------+--------------+----------+------------+-------------+
|             1 |       1 | Stocks       | 10000.00 | 2023-10-01 |        8.50 |
|             2 |       1 | Bonds        |  5000.00 | 2023-10-01 |        3.00 |
|             3 |       2 | Mutual Funds | 15000.00 | 2023-10-01 |        6.00 |
|             4 |       2 | Real Estate  | 20000.00 | 2023-10-01 |        5.00 |
|             5 |       3 | Stocks       | 12000.00 | 2023-10-01 |        8.00 |
|             6 |       3 | Bonds        |  6000.00 | 2023-10-01 |        3.50 |
|             7 |       4 | Mutual Funds | 18000.00 | 2023-10-01 |        6.50 |
|             8 |       4 | Real Estate  | 25000.00 | 2023-10-01 |        5.50 |
|             9 |       5 | Stocks       | 11000.00 | 2023-10-01 |        8.20 |
|            10 |       5 | Bonds        |  5500.00 | 2023-10-01 |        3.20 |
|            11 |       6 | Mutual Funds | 20000.00 | 2023-10-01 |        7.00 |
|            12 |       6 | Real Estate  | 30000.00 | 2023-10-01 |        6.00 |
|            13 |       7 | Stocks       | 13000.00 | 2023-10-01 |        8.30 |
|            14 |       7 | Bonds        |  7000.00 | 2023-10-01 |        3.30 |
|            15 |       8 | Mutual Funds | 17000.00 | 2023-10-01 |        6.70 |
|            16 |       8 | Real Estate  | 22000.00 | 2023-10-01 |        5.70 |
|            17 |       9 | Stocks       | 10500.00 | 2023-10-01 |        8.10 |
|            18 |       9 | Bonds        |  5200.00 | 2023-10-01 |        3.10 |
|            19 |       1 | mutual funds |  2000.00 | 2023-10-03 |        6.00 |
```

Conclusion

using this project the users can maintain a good personal finance score here by helping them while making feature financial decisions and maintain a good track of all the feature assets liabilities.
with the table such as income expenses debts savings investments the user generated the financial health table which provided them the scores of their personal finance.
Many analytics were produced apart from that many other analytics can also be perform based on the data such as finding the debt to ratio (debt burden analysis, investment performance analysis, monthly expense trends and so forth

The code has to be stored in .SQL file prepare a report of the findings along with SQL file and it will be hosted in GitHub

Feature enhancement

The project can be updated with some more features which are external in nature compared to SQL such as
1.integrating with Api to fetch real-life data
2.data visualization to create interactive dashboard reports
3.machine learning implementing machine learning models to predict financial health using certain algorithms and provides recommendations

Impact of the project

this project demonstrates proficiency in MySQL and also the ability to solve real world problems using database systems.
It is a unique and practical addition for a portfolio and will help the user advance in the field of database systems