Fall 2024

Opulence Net Worth Tracker

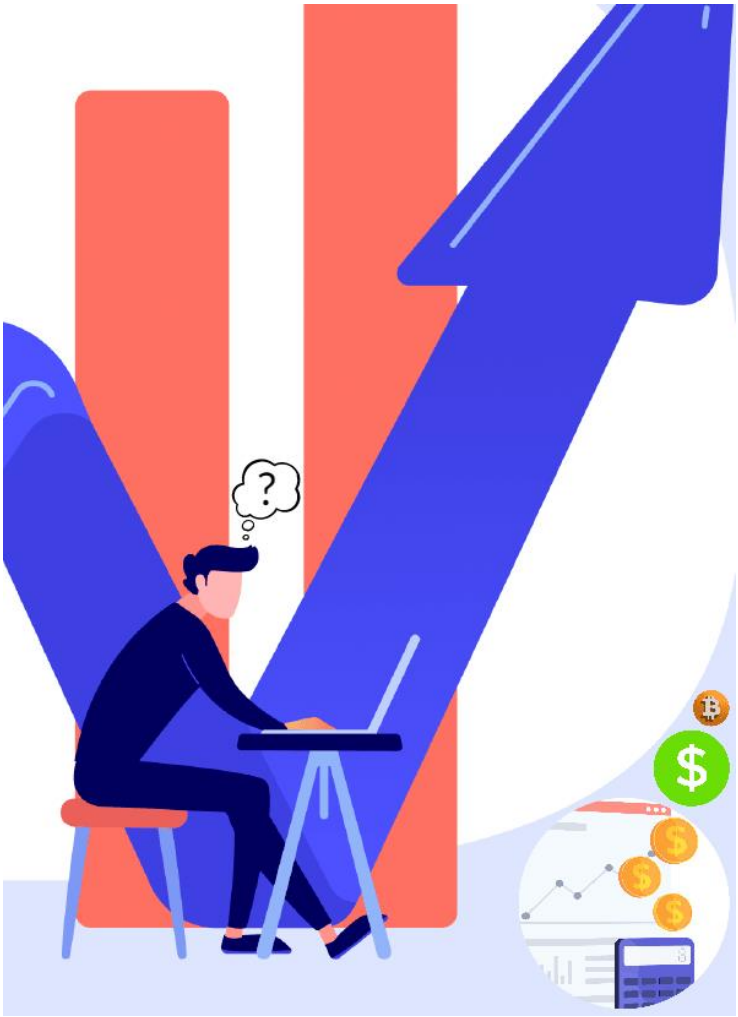
PREMIER OPULENCE WEALTH LLC RELATIONAL DATABASE DESIGN

CREATED BY GEORGE KHRAMTCHENKO
FOUNDER & LEAD SOFTWARE DEVELOPER

Contents

Description

The **Opulence Net Worth Tracker** (ONWT) app is a personal net worth tracker app created by Premier Opulence Wealth LLC— *spearheaded by the technical and administrative developers' efforts of George Khramtchenko*— designed for the Android operating system to easily calculate and interpret user net worth. OWNT allows users to enjoy a seamless app experience with easy-to-use in-app navigation, informative questions and explanations, and provides a personalized net worth analysis in comparing the user's calculated net worth to the median net worth of others in the same age group as the user. Upon answering all the questions relating to user assets and liabilities— which are the two current criteria used for calculating net worth— the user's net worth is displayed with a red-to-green progress bar visually demonstrating the user's financial standing of their net worth in relation to the net worth of other users in the same age category (*Figure 2*). Once satisfied with the services of the app, the user may choose to share their net worth with friends and family or return to the main menu to start over.



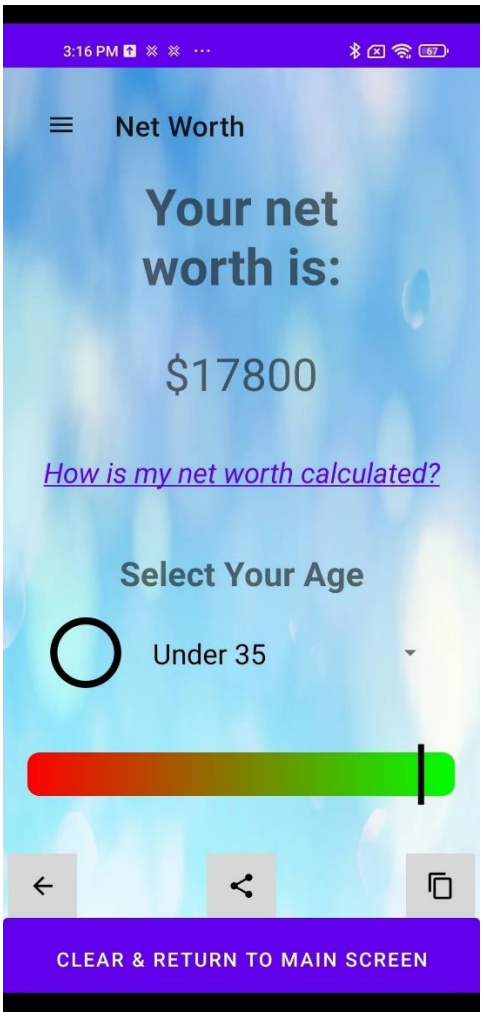


Figure 1 *(left): launch screen image;* ***Figure 2*** *(right): Calculated net worth screen image*

Objective

The Opulence Net Worth Tracker app has an accompanying website where users can learn more about the app, contact support, and visit Google Play to download the app. While there exist no such services for the user to login to a personal user account and keep track of their financial data, for the purposes of this effort it will be assumed that, theoretically, customers can create and login to their customer accounts and keep detailed track of their net worth online in lieu of the ONWT app directly. The hypothetical actions users can execute and the interactions they can experience are the primary focus of this SQL implementation.

The objective of this effort and documentation is to provide a detailed relational database design model for the Opulence Net Worth Tracker software geared towards its theoretical online web application. This relational database model includes the business rules, use cases, a conceptual ERD diagram, a logical ERD diagram, as well as the SQL code necessary to create and implement these diagrams. The result consists of a fully functional relational database with data and appropriate data relationships already established.

Premier Opulence Wealth Business Rules

Business rules describe the operations, definitions, and constraints that apply to an organization. In the context of database systems, business rules dictate how data can be created or retrieved and deleted, ensuring data integrity consistency and adherence to business policy requirements. In the development of a database nouns become entities, and verbs are used to ensure data consistency and referential integrity.

1. A **user** must **create** an **Opulence user account** for use of the software.
2. A **customer** can **create one** valid Opulence account associated only with their identity.
3. A **customer** can **submit** any number of **unique support tickets**.
4. A **submitted user support ticket** must be tied in with the user's **Customer ID**.
5. Each **user account** must have a **unique Account ID**.
6. Each **user account** can have only **one membership/subscription** active at any time.
7. Each **user account** can have **multiple payment profiles** or no payment profiles at all.
8. Each **user account** can have **unique finance entries** entered, tied to a user Account ID.
9. Each **subscription** can either be monthly (with or without auto-renewal) or lifetime.
10. A **user subscription** is not required for the operation of the finance software.
11. Each **payment profile** must have a **unique Profile ID** and contain at least one card.
12. Each **payment profile** ID must be tied to a **unique user Account ID**.
13. Each **payment profile** can contain **multiple cards** if the one card minimum is met.
14. Each **card** entry must be unique and be tied to a **Profile ID**.
15. Each **card** entry can contain only **one card**.
16. There is no limit to how many **card entries** may belong to a **payment profile**.
17. Each **finance entry** can have unique expenses and earnings data.

18. Each **liabilities entry** must have a **unique entry ID** that ties it to a finance entry.
19. Each **expenses entry** must have a **unique entry ID** that ties it to a finance entry.

Premier Opulence Wealth Use Cases

Use cases include the functionality that the system supports, focusing on interaction between the user and the system. A use case is a detailed description of how users interact with a system to achieve a specific goal (such as account or data management, etc.).

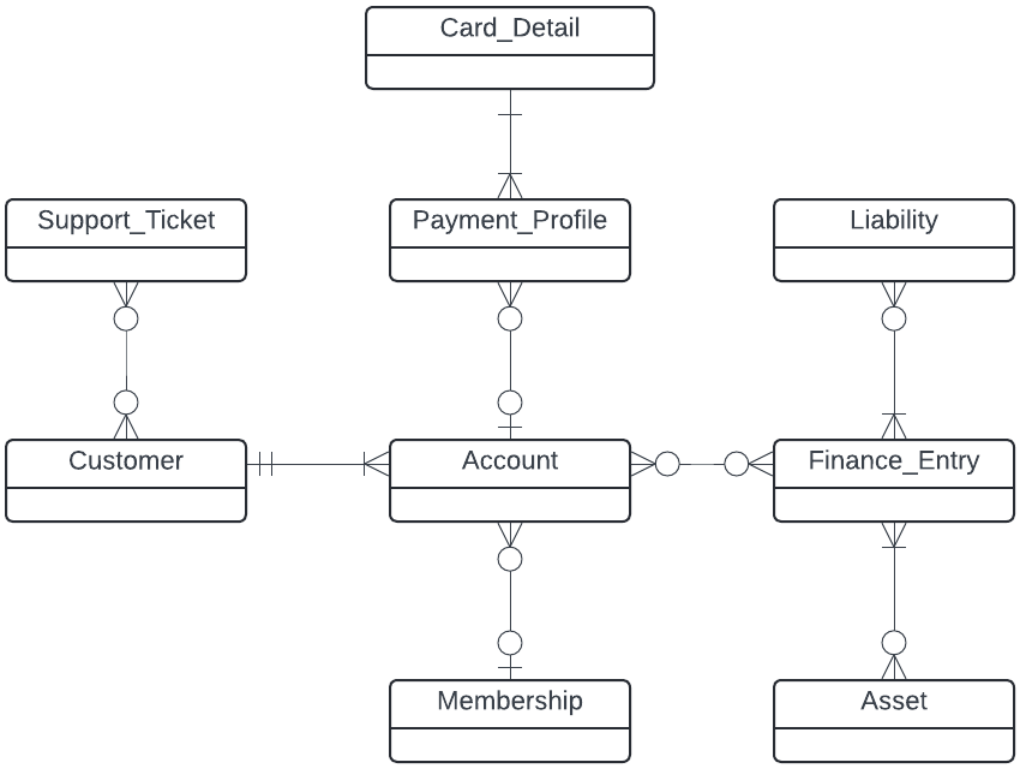
Once any account details or financial entries are created for the first time or are updated via the theoretical web portal, a new SQL query would run automatically to store this new or updated data in the database to affect the use cases defined below.

Note: each table is pre-populated with 10 previously registered customers.

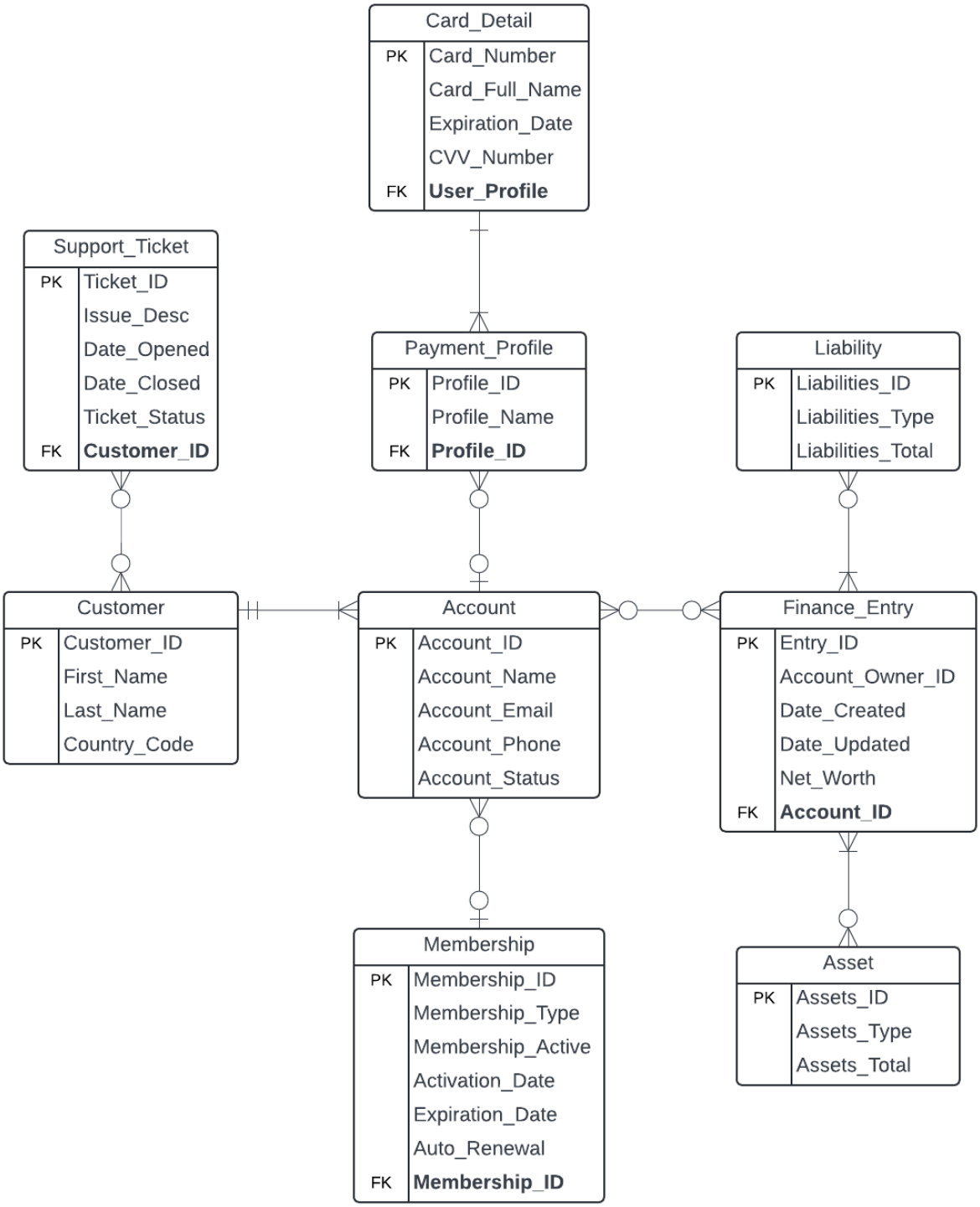
1. Customer George Khramtchenko creates an account (customer table is auto populated).
2. Customer George Khramtchenko contacts support (a support ticket is opened).
3. Customer Matt Klein closes his account after resolving a ticket (account status inactive)
4. Customer Gregoriy Murashkin updates his credit card details by supplying a new card, updating his card name, number, expiration date and CCV number.
5. Customer Joseph Abramowicz changes his membership subscription from monthly to lifetime.
6. Customer Olavi Heikkinen changes his account email address and phone number. The following day, Olavi temporarily disables his account for security purposes.
7. Customer Lina Maes updates her net worth in the finance entry without providing assets or liabilities data and resubscribes to the subscription after newfound satisfaction with the product's software.
8. Customer Antoine Auclair updates his Country code to the United States after moving there but opens a customer support ticket for unexpected changes to the features he has available to his account in the new location.
9. Customer Lina Maes inputs her assets and liabilities data for her previously created finance entry.
10. Customer Antoine Auclair receives a satisfactory reply from customer support which resolves his functionality issue and updates his net worth, assets, and liabilities entries (only one asset and liability entry).

Premier Opulence Wealth Database Conceptual ERD

A conceptual entity-relationship diagram (ERD) provides a high-level overview of the system without going into detail about entity attributes, instead focusing on the key entities and the relationships between them. For the purposes of this project, this diagram is simply an example of such an ERD and has not been necessarily completed to accurately reflect Premier Opulence Wealth's business model.



Premier Opulence Wealth Database Logical ERD



Premier Opulence Wealth Database Implementation

ERD SQL Implementation

Note: please see the complimentary SQL files attached outside of this report for the full executable code.

Finance entry type: 1-100 for assets, 101-200 select all liabilities

DDL: Data Definition Language

DDL details the database schemas and descriptions of how the data exists in the database.

DDL to drop tables

```
DROP TABLE IF EXISTS Assets;
DROP TABLE IF EXISTS Liabilities;
DROP TABLE IF EXISTS FinanceEntries;
DROP TABLE IF EXISTS Membership;
DROP TABLE IF EXISTS CardDetails;
DROP TABLE IF EXISTS
PaymentProfile;
DROP TABLE IF EXISTS Account;
DROP TABLE IF EXISTS SupportTicket;
DROP TABLE IF EXISTS Customer;
```

DDL to create tables

```
CREATE TABLE Customer(
  Customer_ID INT NOT NULL PRIMARY KEY,
  First_Name VARCHAR(255),
  Last_Name VARCHAR(255),
  Country_Code CHAR(3)
);
```

```
CREATE TABLE SupportTicket(
Ticket_ID INT NOT NULL PRIMARY KEY,
Customer_ID INT,
Issue_Type VARCHAR(50) NOT NULL,
Issue_Desc VARCHAR(255),
Date_Opened DATE NOT NULL,
Date_Closed DATE NOT NULL,
Ticket_Status VARCHAR(25),
FOREIGN KEY (Customer_ID) REFERNECES
Customer(Customer_ID)
);
```

```
CREATE TABLE Account(
Account_ID INT NOT NULL PRIMARY KEY,
Account_Name VARCHAR(255) NOT NULL,
Account_Email VARCHAR(255) NOT NULL,
Account_Phone VARCHAR(15) NOT NULL,
Account_Status VARCHAR(50) NOT NULL,
);
```

```
CREATE TABLE PaymentProfile(
Profile_ID INT NOT NULL PRIMARY KEY,
Profile_Name VARCHAR(255) NOT NULL,
FOREIGN KEY (Profile_ID) REFERENCES
Account(Account_ID)
);
```

```
CREATE TABLE CardDetails(
Card_Number INT NOT NULL PRIMARY KEY,
User_Profile INT,
Full_Name VARCHAR(255) NOT NULL,
Expiration_Date DATE NOT NULL,
CVV_Number INT NOT NULL,
FOREIGN KEY (User_Profile) REFERENCES
PaymentProfile(Profile_ID)
);
```

```
CREATE TABLE Membership(
  Membership_ID INT NOT NULL PRIMARY KEY,
  Membership_Type VARCHAR(50) NOT NULL,
  Membership_Active VARCHAR(1) NOT NULL,
  Activation_Date DATE,
  Expiration_Date DATE,
  Auto_Renewal VARCHAR(3),
  FOREIGN KEY (Membership_ID)
  REFERENCES Account(Account_ID)
);
```

```
CREATE TABLE FinanceEntries(
  Entry_ID INT NOT NULL PRIMARY KEY,
  Account_Owner_ID INT,
  Date_Created DATE,
  Date_Updated DATE,
  Net_Worth INT,
  FOREIGN KEY (Account_Owner_ID)
  REFERENCES Account(Account_ID)
);
```

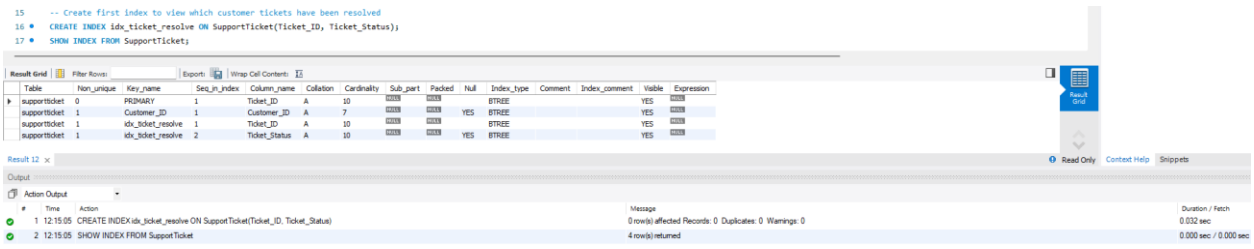
```
CREATE TABLE Liabilities (
  Liabilities_ID INT NOT NULL PRIMARY KEY,
  Liabilities_Type VARCHAR(255),
  Liabilities_Total VARCHAR(255)
);
```

```
CREATE TABLE Assets(
  Assets_ID INT NOT NULL PRIMARY KEY,
  Assets_Type VARCHAR(255),
  Assets_Total VARCHAR(255)
);
```

Index Creation

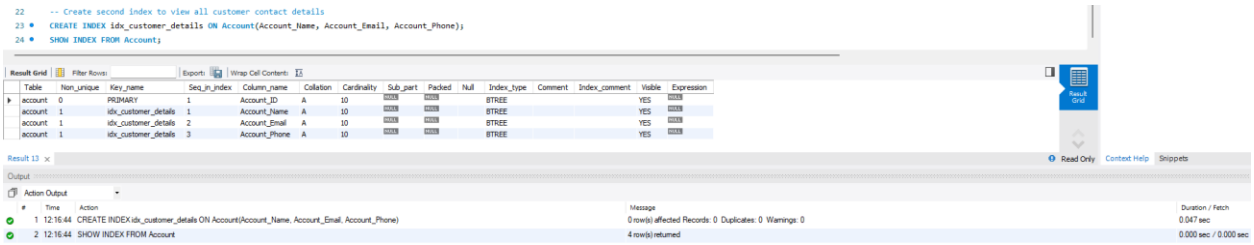
CREATE INDEX idx_ticket_resolve ON SupportTicket(Ticket_ID, Ticket_Status);

SHOW INDEX FROM SupportTicket;



CREATE INDEX idx_customer_details ON Account(Account_Name, Account_Email, Account_Phone);

SHOW INDEX FROM Account;



The following two indexes were created to speed up query execution, to effectively pull the requested data from the tables to display all customer support tickets, as well as customer details such as the account name, email, and phone number.

ALTER TABLE SupportTicket

DROP INDEX idx_ticket_resolve;

ALTER TABLE Account

DROP INDEX idx_customer_details;

DML: Data Manipulation Language

DML details how data manipulation occurs and includes SQL statements such as SELECT, INSERT, UPDATE, DELETE, and others and is used to store, modify, retrieve, delete, and update data in a database.

Pre-Populate Tables with Initial Data (DML)

```
-- Populate the Customer table with data
INSERT INTO Customer(Customer_ID, First_Name, Last_Name, Country_Code) VALUES
(1, 'John', 'Doe', 'USA'),
(2, 'Joseph', 'Abramowicz', 'POL'),
(3, 'Matteo', 'Klein', 'DEU'),
(4, 'Lina', 'Maes', 'BEL'),
(5, 'Xi', 'Zhao', 'CH'),
(6, 'Pedro', 'Diaz', 'DOM'),
(7, 'Gregoriy', 'Murashkin', 'RUS'),
(8, 'Antoine', 'Auclair', 'FRA'),
(9, 'Artem', 'Kovalenko', 'UKR'),
(10, 'Olavi', 'Heikkinen', 'FIN');
```

```sql
-- Populate the SupportTicket table with data
INSERT INTO SupportTicket(Ticket_ID, Customer_ID, Issue_Type, Issue_Desc, Date_Opened, Date_Closed, Ticket_Status) VALUES
(1, 1, 'Subscription', 'Software crash when activating subscription', '2023-06-05', '2023-06-14', 'Resolved'),
(2, 1, 'Subscription', 'Software crash when activating subscription', '2023-06-15', '2023-07-15', 'Resolved'),
(3, 5, 'Software', 'Software glitch in displaying personal net worth', '2023-07-24', '2023-07-29', 'Resolved'),
(4, 3, 'Account', 'Question regarding account ownership transfer', '2023-07-27', NULL, 'Unresolved'),
(5, 8, 'Subscription', 'Incorrect pricing displayed for account status', '2023-08-06', '2023-08-14', 'Resolved'),
(6, 9, 'Miscellaneous', 'General questions regarding product use', '2023-08-27', '2023-09-09', 'Resolved'),
(7, 8, 'Software', 'Software glitch when logging out of account', '2023-09-17', '2023-09-19', 'Resolved'),
(8, 7, 'Subscription', 'Cancel subscription and recieve refund', '2023-09-23', NULL, 'Unresolved'),
(9, 4, 'Account', 'Identity confirmation questions', '2023-09-26', '2023-10-02', 'Resolved'),
(10, 1, 'Account', 'Change email to update contact details', '2023-11-05', '2023-11-13', 'Resolved');
```

```sql
-- Populate the Account table with data
INSERT INTO Account(Account_ID, Account_Name, Account_Email, Account_Phone, Account_Status) VALUES
(1, 'Johnathan', 'jdoe@gmail.com', '7120877401', 'Active'),
(2, 'Joseph', 'jambram@gmail.com', '3552419545', 'Active'),
(3, 'Matteo', 'mklein024@gmail.com', '6536708514', 'Active'),
(4, 'Lina', 'linaloves@yahoo.com', '3589389293', 'Active'),
(5, 'Xi', 'xifest@gmail.com', '8591400362', 'Inactive'),
(6, 'Pedro', 'pedrosworld@gmail.com', '4309977943', 'Active'),
(7, 'Gregoriy', 'rusgreg11@gmail.com', '1754935526', 'Active'),
(8, 'Antoine', 'atoantony@gmail.com', '3343324385', 'Inactive'),
(9, 'Artem', 'artema@gmail.com', '1010708905', 'Active'),
(10, 'Olavi', 'olaviheik@gmail.com', '9541868424', 'Active');
```

```sql
-- Populate the PaymentProfile table with data
INSERT INTO PaymentProfile(Profile_Name, Profile_ID) VALUES
('JJ Doe', 1),
('Joss A', 2),
('Matt Klein', 3),
('Lina M', 4),
('XiZ', 5),
('Pedrodiaz', 6),
('Gregoriy', 7),
('Antoine Auc', 8),
('Artem K', 9),
('O Heikkinen', 10);
```

```sql
-- Populate the CardDetails table with data
INSERT INTO CardDetails(User_Profile, Card_Number, Full_Name, Expiration_Date, CVV_Number) VALUES
(1, 98878515, 'Johnathan Doe', '2025-08-01', '264'),
(2, 64393131, 'Joseph Abramowicz', '2026-04-01', '364'),
(3, 93049281, 'Matteo Klein', '2027-10-01', '496'),
(4, 18366677, 'Lina Maes', '2024-04-01', '254'),
(5, 40180280, 'Xi Zhao', '2028-11-01', '275'),
(6, 38809172, 'Pedro Diaz', '2025-12-01', '864'),
(7, 40355744, 'Gregoriy Murashkin', '2024-02-01', '907'),
(8, 26561939, 'Antoine Auclair', '2027-03-01', '375'),
(9, 44888755, 'Artem Kovalenko', '2024-07-01', '889'),
(10, 90411740, 'Olavi Heikkinen', '2026-05-01', '142');
```

```sql
-- Populate the Membership table with data
INSERT INTO Membership(Membership_ID, Membership_Type, Membership_Active, Activation_Date, Expiration_Date, Auto_Renewal) VALUES
(1, 'Lifetime', 'Y', '2023-06-18', Null, Null),
(2, 'Monthly', 'N', '2023-08-03', '2023-09-03', 'No'),
(3, 'Monthly', 'Y', '2023-09-23', '2023-10-23', 'Yes'),
(4, 'Cancelled', 'N', '2023-10-08', '2023-11-08', 'No'),
(5, 'Lifetime', 'Y', '2024-01-04', Null, Null),
(6, 'Monthly', 'Y', '2024-01-06', '2024-02-06', 'No'),
(7, 'Monthly', 'Y', '2024-02-08', '2024-03-08', 'Yes'),
(8, 'Lifetime', 'Y', '2024-03-24', Null, Null),
(9, 'Cancelled', 'Y', '2024-05-04', '2024-06-04', 'No'),
(10, 'Lifetime', 'Y', '2024-07-02', Null, Null);
```

```sql
-- Populate the FinanceEntries table with data
INSERT INTO FinanceEntries(Entry_ID, Account_Owner_ID, Date_Created, Net_Worth) VALUES
(1, 1, '2023-08-05', 153335),
(2, 2, '2023-08-13', 1429455),
(3, 3, '2023-10-15', 1142524),
(4, 4, NULL, NULL),
(5, 5, '2024-01-10', 895543),
(6, 6, '2024-01-13', 100012),
(7, 7, '2024-03-06', 452965),
(8, 8, '2024-03-28', 775085),
(9, 9, NULL, NULL),
(10, 10, '2024-07-27', 3947506);
```

```sql
-- Populate the Liabilities table with data
INSERT INTO Liabilities(Liabilities_ID, Liabilities_Type, Liabilities_Total) VALUES
(1, 'Undetermined', 25000),
(2, 'Personal', 12478),
(3, 'Loans', 249456),
(4, 'Business', 55343),
(5, 'Shopping', 999),
(6, 'Undetermined', 13454),
(7, 'Online Shopping', 7000),
(8, 'Restaurants', 4000),
(9, 'Taxes', 27500),
(10, 'Undetermined', 97847);

-- Populate the Assets table with data
INSERT INTO Assets(Assets_ID, Assets_Type, Assets_Total) VALUES
(1, 'Business', 178335),
(2, 'Investment', 249456),
(3, 'Tax Return', 3212364),
(4, 'Business', 6436466),
(5, 'Salary', 7454),
(6, 'Gift', 45856),
(7, 'Online Business', 84656),
(8, 'Investment', 9567),
(9, 'Lottery', 864453),
(10, 'Undetermined', 57437);
```

```sql
SELECT * FROM Customer;
SELECT * FROM SupportTicket;
SELECT * FROM Account;
SELECT * FROM PaymentProfile;
SELECT * FROM CardDetails;
SELECT * FROM Membership;
SELECT * FROM FinanceEntries;
SELECT * FROM Liabilities;
SELECT * FROM Assets;

SELECT Ticket_ID, Ticket_Status
FROM SupportTicket;

SELECT Account_Name,
Account_Email, Account_Phone
FROM Account;
```

Executing 10 Use Cases for Premier Opulence Wealth Database (DML)

Use Case 1

Customer George Khramtchenko creates an account (customer table is auto-populated).

```
INSERT INTO Customer(Customer_ID, First_Name, Last_Name, Country_Code)

VALUES (11, 'George', 'Khramtchenko', 'USA');

INSERT INTO Account(Account_ID, Account_Name, Account_Email, Account_Phone, Account_Status)
VALUES (11, 'George', 'gkhramtchenko738@g.rwu.edu', '9784753347', 'Active');
```

Use Case 2

Customer George Khramtchenko contacts support (a support ticket is opened).

```
INSERT INTO SupportTicket(Ticket_ID, Customer_ID, Issue_Type, Issue_Desc, Date_Opened,
Date_Closed, Ticket_Status)

VALUES (11, 11, 'Subscription', 'Subscription terms are unclear!', '2023-03-23', '2023-03-27', 'Resolved');
```

Use Case 3

Customer Matteo Klein closes his account after ticket is resolved, account status is set to inactive.

```
UPDATE SupportTicket

SET Date_Closed = '2023-07-27'

WHERE Customer_ID = 3;


UPDATE Account

SET Account_Status = 'Inactive'

WHERE Account_ID = 3;
```

Use Case 4

Customer Gregoriy Murashkin updates his credit card details by supplying a new card, updating his card name, number, expiration date and CCV number.

```
UPDATE CardDetails

SET Full_Name = 'Gregoriy A. Murashkin', Card_Number = 49574857, Expiration_Date = '2029-04-01',
CVV_Number = '264'

WHERE User_Profile = 7;
```

Use Case 5

Customer Joseph Abramowicz changes his membership subscription from monthly to lifetime.

```
UPDATE Membership

SET Membership_Type = 'Lifetime', Membership_Active = 'Y', Activation_Date ='2023-09-15',
Expiration_Date = NULL, Auto_Renewal = NULL

WHERE Membership_ID = 2;
```

Use Case 6

Customer Olavi Heikkinen changes his account email address and phone number. The following day, Olavi temporarily disables his account for security purposes.

```
UPDATE Account

SET Account_Email = 'olavi837@gmail.com', Account_Phone = '95418684358'

WHERE Account_ID = 10;


UPDATE Account

SET Account_Status = 'Temporarily Disabled'

WHERE Account_ID = 10;
```

Use Case 7

Customer Lina Maes updates her net worth in the finance entry without providing assets or liabilities data and resubscribes to the subscription after newfound satisfaction with the product's software.

```
UPDATE FinanceEntries

SET Date_Updated = '2024-02-26', Net_Worth = '374958'

WHERE Entry_ID = 4;


UPDATE Membership

SET Membership_Type = 'Monthly', Membership_Active = 'Y', Activation_Date = '2024-02-26',
Expiration_Date = '2024-03-26', Auto_Renewal = 'Yes'

WHERE Membership_ID = 4;
```

Use Case 8

Customer Antoine Auclair updates his Country code to the United States after moving there but opens a customer support ticket for unexpected changes to the features he has available to his account.

```
UPDATE Customer

SET Country_Code = 'USA'

WHERE Customer_ID = 8;


INSERT INTO SupportTicket(Ticket_ID, Customer_ID, Issue_Type, Issue_Desc, Date_Opened,
Date_Closed, Ticket_Status) VALUES

(12, 8, 'Account', 'Limited feature functionality after changing country settings to the United States',
'2024-04-25', NULL, 'Unresolved');
```

Use Case 9

Customer Lina Maes inputs her assets and liabilities data for her previously created finance entry.

UPDATE Liabilities

SET Liabilities_Type = 'Restaurant Losses', Liabilities_Total = '40000'

WHERE Liabilities_ID = 4;

UPDATE Assets

SET Assets_Type = 'Tax Returns', Assets_Total = '32750'

WHERE Assets_ID = 4;

Use Case 10

Customer Antoine Auclair receives a satisfactory reply from customer support which resolves his functionality issue and updates his net worth, assets, and liabilities entries (only one asset and liability entry).

UPDATE SupportTicket

SET Date_Closed = '2024-04-29'

WHERE Customer_ID = 8;

UPDATE FinanceEntries

SET Date_Updated = '2024-04-29', Net_Worth = '785913'

WHERE Entry_ID = 8;

UPDATE Assets

SET Assets_Type = 'Investment', Assets_Total = '12500'

WHERE Assets_ID = 8;

UPDATE Liabilities

SET Liabilities_Type = 'Restaurant Losses', Liabilities_Total = '2000'

WHERE Liabilities_ID = 8;

DCL: Data Control Language

DCL controls access to data within a database, as it defines the permissions that users can perform on the data. No such DCL queries currently exist in the Premier Opulence Wealth database.

- **GRANT**
- **REVOKE**

TCL: Transaction Control Language

TCL manages database operations that are treated as a single unit of work, meaning using commands such as those listed below. No such TCL queries currently exist in the Premier Opulence Wealth database.

- **COMMIT**
- **ROLLBACK**
- **SAVPOINT**
- **SET TRANSACTION**