CMPS 350 Web Development Fundamentals - Spring 2023
Lab 12 – Data Management using Prisma and SQLite Database
Part 2

Objective

This Lab is a continuation of Lab 11, and you will practice more advance concepts about Prisma.

- Data Modelling using Prisma Schema Language
- Cascade Delete and Update
- Relation Queries [Include, Select]
- Full Text Search [Search]
- Filtering and Sorting
- Aggregation, Grouping and Summarization

Project Setup

Download **Lab12-Data-Management** from the GitHub Repo and copy it to your repository.
Open the **BankingApp** in VS Code and complete the tasks below.

1. Run npm i to install the dependencies
2. add the **.env** file and add the following line [*DATABASE_URL="file:./dev.db"*]
3. **npx prisma db seed**
4. **npm run dev**

Update the database Shema

1. Modify the previous weeks **schema**.**prisma** file and a new model named "**owner**". See the entity relationship diagram below.
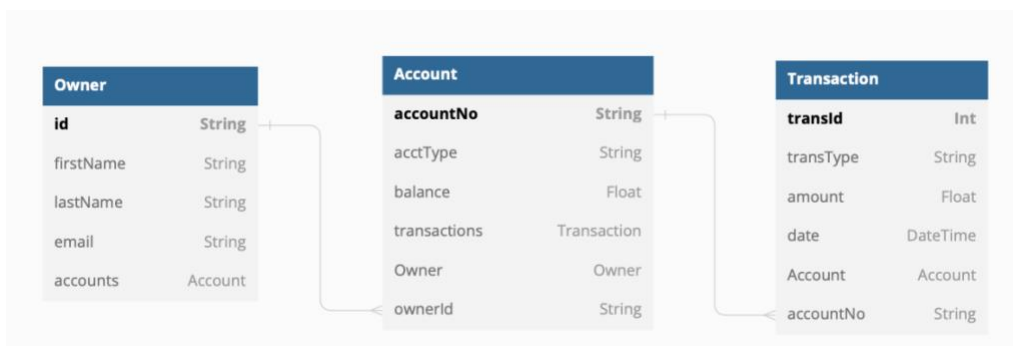

Figure 1 Banking Entities Diagram

- **id** is the primary key for owner and should be auto-assigned using cuid() as you did for the account model.

2. Export the models to your database by using the following Prisma command

   npx prisma migrate dev –name init

   Anytime you make changes to the models, you need to **npx prisma migrate dev**

3. To view your database content, run the following command `npx prisma studio`

Querying the database with Prisma client

1. Add to the **accounts-repo.js** repository the following functions that use **Prisma client**. <mark>As you make progress</mark> test each function using a console app, Postman or Mocha:
   a. **getOwners()** : returns all the owners.
   b. **deleteOwner**(**ownerId**) : deletes the particular owner and all its accounts and transactions.
   c. **searchOwner(name) :** return the owner that contains the given name.
   d. **getTrans(accountNo, fromDate, toDate):** get transactions for a particular account for a date range.
   e. **getAvgBalance():** returns average account balance by account type.
   f. **getMinAndMaxBalance() :** returns the minimum and maximum balance in the account table.
   g. **getTop3Accounts():** returns the top three accounts with the highest balance sorted in descending order.
   h. **getTransSum(accountNo , fromDate, toDate):** returns the total amount of withdrawals and deposits made within a specified date range for a given account sorted by trans date.
   i. **getOwnerReport(ownerId):** returns all the information about the owner, including their accounts, and for each account, it should also include all its transactions.

2. Test the app using the user interface by running the app **npm run dev**.