CS374 – Intro to Database Management

Application Development Project

Rubric for Second Deliverable

Group Member #1: Connor Weldy

Group Member #2: Nozomu Ohno

Group Member #3: Michael Laramie

Name	Requirements	Points	Awarded
Description of Application	• An overview of your application • System requirements (e.g. hardware, DBMS, other software) • A detailed description of your application • Are there features that will not be implemented? What are they, and why won’t you fulfill them?	10	
Project Management - Schedule	• Detailed schedule of who will do what part of project, by when	5	
Logical Diagram	• Logical diagram in UML or E-R • Discussion of how your data model will satisfy the needs of your application • Discussion of alternative designs that you did not do (and why)	15	
Queries Required	• Required queries in English (not SQL) • What entities and/or relationships are required for each query? • How will each query satisfy the needs of your application	15	
Grammar, punctuation, syntax, and references	• Follow rules from the Penguin handbook on writing • References as appropriate (e.g. if you are modeling your application after an existing application, make note of that)	5	

Description of Application

Application overview:

Our application will be able to keep track of multiple users and their financial information. First, a user will either sign in or create a new user account. After the login process has been completed, the user can choose an account (checking, saving, business, etc.). From within their specific account, a user will be able to input a transaction that contains the information of the transaction itself. The application will then generate different statistics based on the transactions entered such as amount spent per category, amount spent per month, etc.

System Requirements:

- Implementation Language: Python
    - Interface used: PyQt5 and QtDesigner
- DBMS: SQL Server
- Runs on a Computer
- Whitworth Internet connection – connecting to CS1 server on SQL Server

Detailed Description:

The application takes user input to create the username/password that will make up the user or to sign into an account made previously. If the user makes a new account, the account will be inserted into the table for future sign-ins. The user can have multiple accounts, such as savings, checking, or some other type of account. Then, a user can enter a transaction for one of their particular accounts, these transactions will consist of transaction amount, the category of the transaction, and the date of the transaction, and the account ID. From there we will use queries to get the information we need to calculate statistics based off those transactions like total spent during the month of June, or total money spent on Groceries. With these statistics the application can also display the transaction history of the past week/month as well. The application will have a user interface that will display an overview of a user's accounts. A user can then specify to look at a more detailed description of one of their accounts. If so, a user will be able to see more statistics about that specific account.

Project Management and Schedule

- Project Deliverable 2 - Due Nov. 18th
- Learn the basics of Python and abstract pseudocode for the application – By Nov. 30th
    - Python basics to learn:
        - strings
        - functions
        - for loops
        - tabular syntax
- Project Status Update – Due Nov. 30th (Tuesday after Thanksgiving break)
- After Nov. 30th: Start implementation of the pseudocode (Coding part)
- UPDATE: Added the tables needed in SQL server.

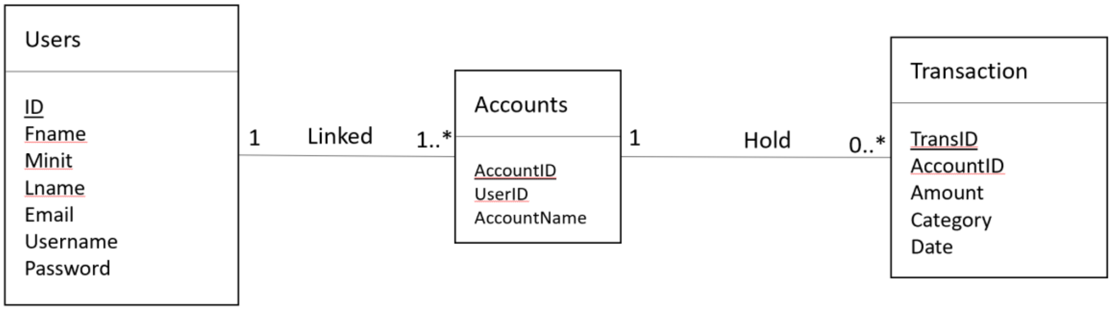
- Meet Thursday December 2nd after the test, come up with next meeting date
- Meet Sunday December 5th afternoon
- Finish the project itself – Dec. 13th (Monday on the finals week)
- Individual Write-up – Due Dec. 16th
- Final Presentation: Dec. 17th (Friday) Last day of the Finals

Roles:

- Michael & Nozomu: write the queries in SQL
- Connor: implement the SQL as an embedded SQL in Python (as functions)
- All: Implement user interface.

Logical Diagram

UML Diagram



Discussion

In our UML design above, we have three tables. The first table on the left named “Users” contains all user information. Each user has a unique ID, a first name (Fname), a last name (lname), an email (Email), a username (Username) and a password (Password). The second table named “Accounts” contains information about an account. Each account has a unique ID (AccountID), a user ID which is a foreign key to the Users table (UserID) and an account name (AccountName). The third and final table called “Transactions” holds all information about a transaction. It will contain a unique transaction ID (TransID), an account ID (AccountID) which is a foreign key to the Accounts table (AccountID), a transaction amount (Amount), a category (Category), and the date of the transaction (Date).

Each user in our database is linked to one or many accounts. Accounts have only one user. Accounts hold several transactions, but transactions only have one account.

We at first thought about having another table for the relationship between transaction and account; however, we defined a transaction to only have one account, so therefore it makes sense to not have this fourth table.

Queries Required

Queries in English:

1. Add a user with attributes name, email, username, and password.

2. Add an account with account name.
3. Add transactions with amount, category and date.
4. Find the current amount that the account has.
5. Find total amount deposited and withdrew for a user.
6. Find total transaction history for the past week
7. Find total transaction history for the past month
8. Determine categorical spending for a user
9. Determine average spending on a day/week/monthly basis

Entities/relationships required:

- Each user can have multiple accounts
- Accounts hold all transactions

Satisfying the needs of the application

- Query 1 satisfies the need to add new users to the database.
- Query 2 satisfies the need to add a new account for users to the database.
- Query 3 satisfies the need to add a new transaction for each account to the database.
- Query 4 – 9 satisfies the need for users to be able to interact with the transactions and their accounts.