CS 3200 Final Project

Ashvika Boopathy, Sribindu Sreepada

Introduction/Overview:

For our project, we created a library database that can be used by librarians in a library network to manage patron, account, librarian, and book data. We obtained a Goodreads CSV file containing information about a book's id, name, authors, isbn, published date, publisher, language, and page count. The dataset was obtained from the following link on kaggle: https://www.kaggle.com/datasets/bahramjannesarr/goodreads-book-datasets-10m?resource=download&select=book1000k-1100k.csv

README:

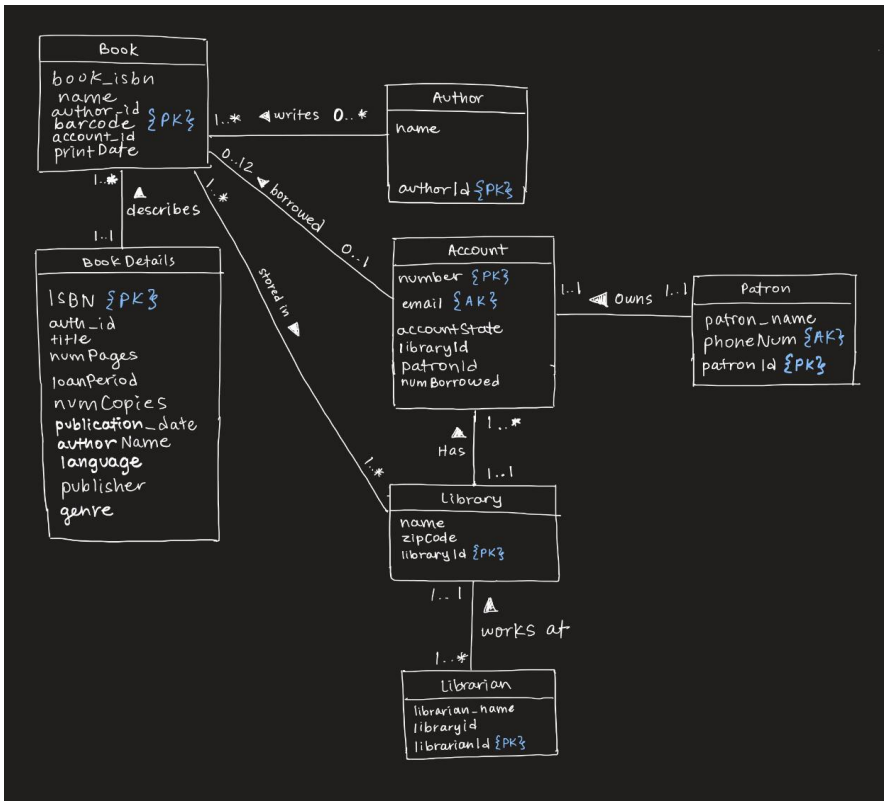
1. Download the minilib_db dump file and minilib.py file
2. Open the sql scripts on MySQLWorkbench
3. Run the scripts using the lightning bolt icon. The database and all the tables should be seen on the left under SCHEMAS. All procedures should also be present under the procedures section
4. Go to terminal and install PyMySQL using the following command if not already installed: pip install pymysql
5. Open Anaconda, PyCharm, or another text editor that supports Python to run the minilib_app.py file
6. Pandas and matplotlib should be installed in your python application. Make sure that the pandas and matplotlib libraries are installed. The commands are: pip install pandas and pip install matplotlib
7. Run the file
8. Enter inputs for the prompts generated from the python file. Refer to the user flow for specific details

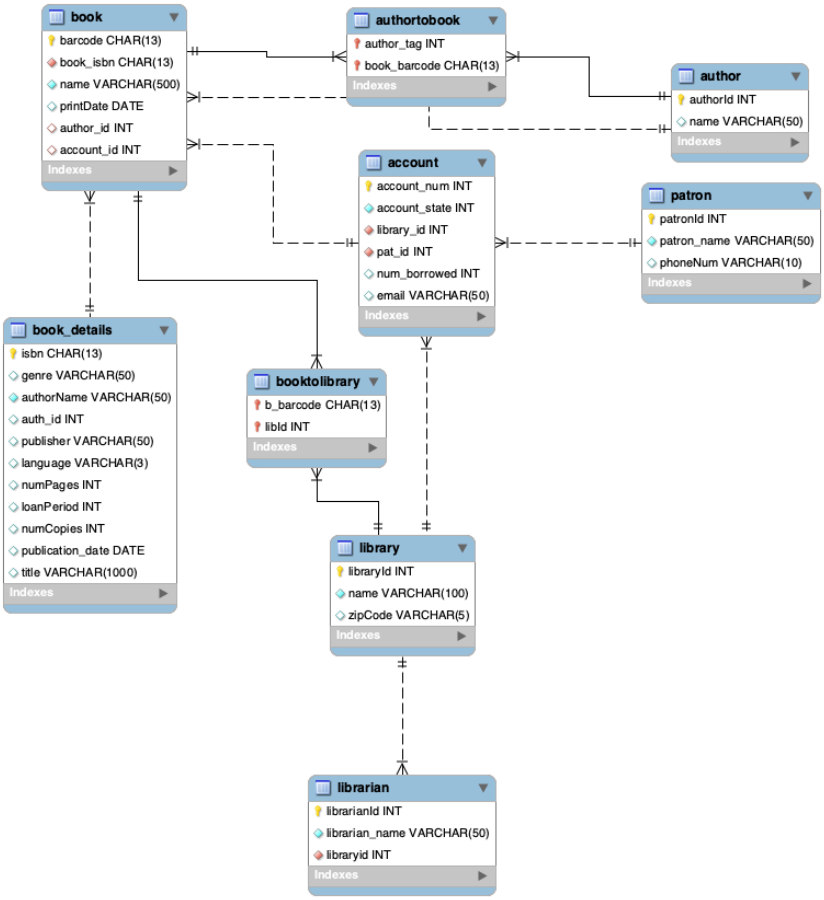
Technical Specifications:

The host language we are using for this project is Python. The type of database we are using is a MySQL database. There are no machine restrictions for the project, and both Sri and Ashvika are Mac users. For analyzing our data we used pandas to create a simple data frame from the data in the MySQL tables, and matplotlib to analyze the data in the data frame. We also used the PyMySql library to connect python to SQL .

UML and EER:

Diagrams are below and also included in a separate pdf file (library_UML_EER.pdf)


Book
book_isbn
name
author_id {PK}
barcode
account_id
printDate
Author
name
authorId {PK}
writes
describes
borrowed
stored in
Book Details
ISBN {PK}
auth_id
title
numPages
loanPeriod
numCopies
publication_date
author Name
language
publisher
genre
Account
number {PK}
email {AK}
accountState
libraryId
patronId
numBorrowed
Patron
patron_name
phoneNum {AK}
patron Id {PK}
owns
Has
Library
name
zipCode
libraryId {PK}
works at
Librarian
librarian_name
libraryId
librarianId {PK}


book
barcode CHAR(13)
book_isbn CHAR(13)
name VARCHAR(500)
printDate DATE
author_id INT
account_id INT
Indexes
authortobook
author_tag INT
book_barcode CHAR(13)
Indexes
author
authorId INT
name VARCHAR(50)
Indexes
account
account_num INT
account_state INT
library_id INT
pat_id INT
num_borrowed INT
email VARCHAR(50)
Indexes
patron
patronId INT
patron_name VARCHAR(50)
phoneNum VARCHAR(10)
Indexes
book_details
isbn CHAR(13)
genre VARCHAR(50)
authorName VARCHAR(50)
auth_id INT
publisher VARCHAR(50)
language VARCHAR(3)
numPages INT
loanPeriod INT
numCopies INT
publication_date DATE
title VARCHAR(1000)
Indexes
booktolibrary
b_barcode CHAR(13)
libId INT
Indexes
library
libraryId INT
name VARCHAR(100)
zipCode VARCHAR(5)
Indexes
librarian
librarianId INT
librarian_name VARCHAR(50)
libraryid INT
Indexes

User Flow:

1. Run the python file
2. Enter your MySQL username and password to connect to the server
3. The first operation will be a read on the book_details table. Answer the prompt, giving a genre from the printed list. Repeat if the input is invalid.
4. The next operation will be updating a patron's phone number. Answer the prompt, giving a patron name from the printed list. Then enter a 10 digit phone number. Repeat if the inputs are invalid.
5. The third operation will be deleting a patron from the patron table. Enter the patron's name, id, and account number that you would like to delete as it is in the printed list. Repeat if the inputs are invalid.
6. The fourth operation will be a second delete operation. This function will delete a user-specified librarian. Enter the librarian's name as it is in the printed list. Repeat if the inputs are valid.
7. The fifth operation will be creating a tuple to add to the book table. Input the values as indicated by the prompt. Repeat if the inputs are invalid.
8. One of the bonus operations is to print a bar chart displaying the Number of Books per Genre. Another bonus operation prints the number of books published by each publisher in a pie chart. The last bonus operation shows a scatterplot plotting the number of copies a book has correlated with the loan period. It can be seen that there is a positive correlation. No inputs needed to run these.

Lessons Learned:

1. Technical expertise gained

We were much more comfortable with creating database tables in SQL and using the CRUD commands by the end of the project. Creating and organizing the tables according to foreign key references was something that was new and challenging for us. We also did not have much experience with inputting data before. We learnt how to manually input data in SQL and also learnt how to use insert statements to input data.

2. Insights, time management insights, data domain insights etc.

We often encountered an update error on the child class such as "Error Code: 1452. Cannot add or update a child row: a foreign key constraint fails". Both of us did not have experience with dealing with this error, but after discussion we realized that we had to change the NOT NULL constraints in the parent tables before we could make changes to the child table. We also realized that our tables were very much dependent on other tables and making an edit in one table meant we had to change other tables

as well. We both handled time management well. We were able to find times where we could work on the project and did not procrastinate any work.

3. Realized or contemplated alternative design / approaches to the project

We initially intended to have certain aspects of the database accessible to both the librarian and patron. For example, the patron would have user access to alter their contact information and also be able to view information about books in the library. After contemplating this idea and the project requirements, we thought that the CRUD operations would be better suited for the librarian to access rather than the patron. It does not make sense for the patron to be able to update information regarding books and other patrons.

Future Work:

1. Planned uses of the database

The database can be used by librarians in Boston to update information on patrons, their accounts, books, specific details for the books, library information, and librarian information. The purpose of this database is for librarians to be able to check if a book is available based on a patron's request, and for a librarian to edit patron, library, and librarian information.

2. Potential areas for added functionality

One potential area for added functionality includes recognizing if the user entered a book title or patron name without upper case letters. Right now, if the user enters a name in all lower case letters python returns an error message because it does not recognize the entered name. In the future we would like to include all valid inputs such as entries with all uppercase letters, all lowercase letters, and uppercase letters in the middle of an entry.

Another area of added functionality is allowing the patron to edit their contact details, and also allowing the patron to request books that they would like the library to have. Currently this application is tailored towards librarians, and changing it to allow certain aspects accessible to the patron would be beneficial. We could have a table called patron_recommendations and once a book that a patron requests is in the library it could be added to the book and book_details tables.