Library Application Design Specification

Library Management Application has been developed for use by Library management for keeping track of books borrowed, borrowers and their fines (if any). This application records all the book loans and the borrowers using a MYSQL Database and applies fines if book returns are late.

The underlying data normalized and deployed onto a MYSQL database. It consists of 7 entities, each having a unique purpose in serving different functionalities of the user interface.

The table 'BOOK' has a list of all the books available in the library. The 'ISBN' attribute in this table has been referenced by the 'BOOK_AUTHOR', 'BOOK_AVAIL' and 'BOOK_LOANS'. The 'AUTHOR_ID' from 'AUTHORS' also is referenced by 'BOOK_AUTHOR'.

The availability of a book is stored on 'BOOK_AVAIL'. For the sake of easier testing, I have assumed the library to have only one copy of each book, but this can be changed by DB admin. The ISBN in this table is referenced from table 'BOOK'.

Creation of new borrower requires user to enter all the attributes of the 'BORROWER' table except for the 'CARD_ID'. Creation of 'CARD_ID' is automated by using the datetime stamp, as a string. For example, if a new borrower is created on 2018/11/29 23:15:30, the 'CARD_ID' would be 20181129231530.

'LOAN_ID' in 'BOOK_LOANS' is auto-incremented for every book loaned. This 'LOAN_ID' is referenced by the 'LOAN_ID' in 'FINES' table.

'FINES' table has 'FINE_AMT' and 'PAID', which specify the amount of fine the issued on a loan of a book and if that fine is paid in full or not. Fines for every loan is calculated when the book is returned, and a message is displayed saying "The book is being returned late by 2 days. Fine = $0.50".

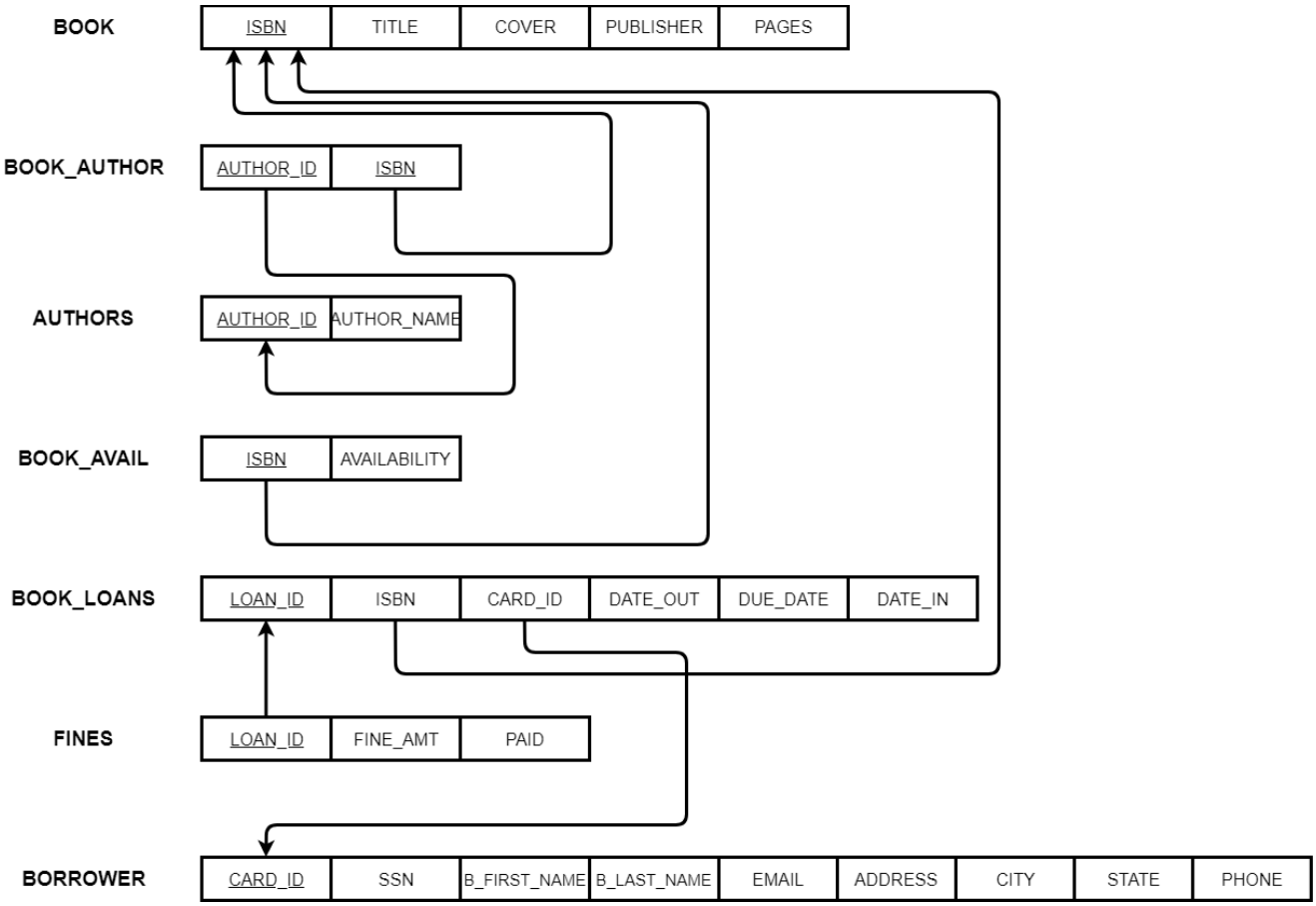
The reason for creating 'BOOK_AVAIL' table is that when user requests to loan a book, the application usually queries the database to see if the book has been already loaned. Instead, the 'BOOK_AVAIL' table will have the number of copies of the book available. This will also enable keeping track of the number of copies of the book the library has.

Message boxes are displayed to using the tkinter messagebox widget to notify the user if there is anything wrong. For example, when the user is trying to loan a book out of stock, it displays a message saying, "No copies of this book are available right now", or when the user tries to create a new borrower with already existing SSN, it displays a message saying "Borrower already exists with the same SSN.

Deletion of a borrower completely removes him from the database. There will be no record of any books loaned to him and any fines payed by him. This is assuming there is no need to store this information, storing which would only slow down querying the existing ones.

Books cannot be added or deleted from the database assuming there is not such requirement.

Below is the schema of the database with the assumptions above.



Limitations that have not been handled yet:

The UI is functional and intuitive but doesn't look good.

New books cannot be added into the database except by the admin.

A single borrower can take multiple copies of the same book at a time (if available).

Data validation while creating a new borrower is not done.

The application trusts the user to incur the fine amount from the borrower and doesn't verify if the amount is received.

Fines are not automatically accounted for every day but are calculated at the return of a loaned book.

You can search for borrower either with first name or the last name but not both because they are different attributes of the table ‘BORROWER’. (This can be handled but I was too late to notice it)

There is no concurrency control in place.