Database Systems Project

Library Management System

-Arvind Ram (2020A7PS1210P)

-Rishabh Kumar (2020A7PS1211P)

System Requirements

1. Introduction

a. Purpose

The purpose of this document is to build a software to help libraries keep track of the members of the library along with keeping a record of the books issued and present in the library.

b. Scope

The library management system is based on a relational database model which would make it convenient for libraries all around the world to keep a record of their activities ensuring their smooth functioning.

c. References

i. https://www.geeksforgeeks.org/
ii. https://stackoverflow.com/
iii. https://www.youtube.com/c/pedrotechnologies

d. Overview

The project makes use of React.js for frontend, Node.js for backend and MySQL for the relational database.

2. Functional Requirements

The functional requirements of the project includes:

a. Create a new account with the library
b. Delete their account
c. Update account information
d. Add new Books
e. Search for availability of books and find its location
f. Delete books
g. Issue books to members
h. Calculate Dues
i. Book rooms in the library

3. Data Requirements

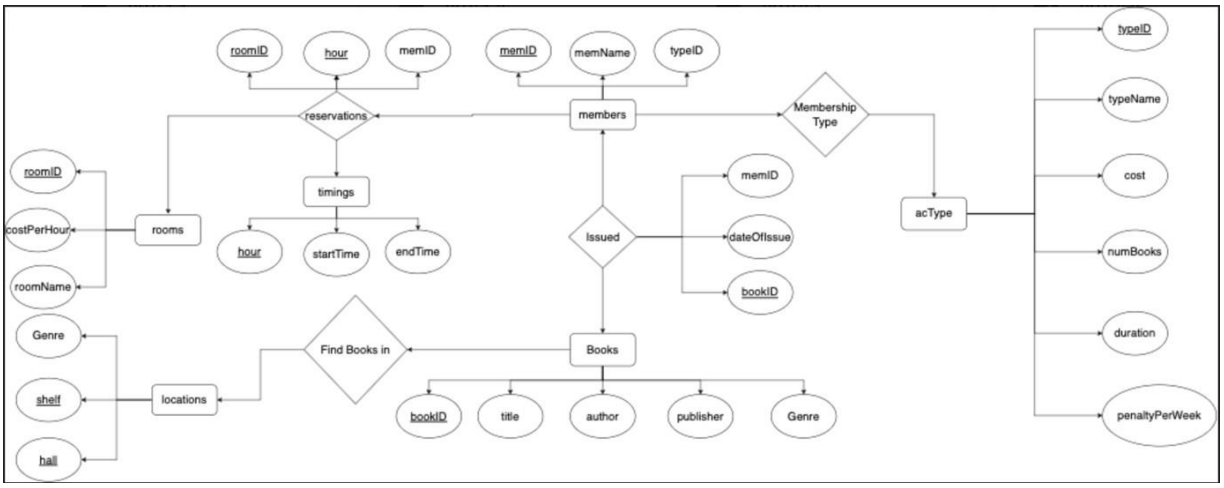
The data requirements of the project includes:

a. The list of Account types and its specifications would be given
b. The timings for each slot would be given for room bookings

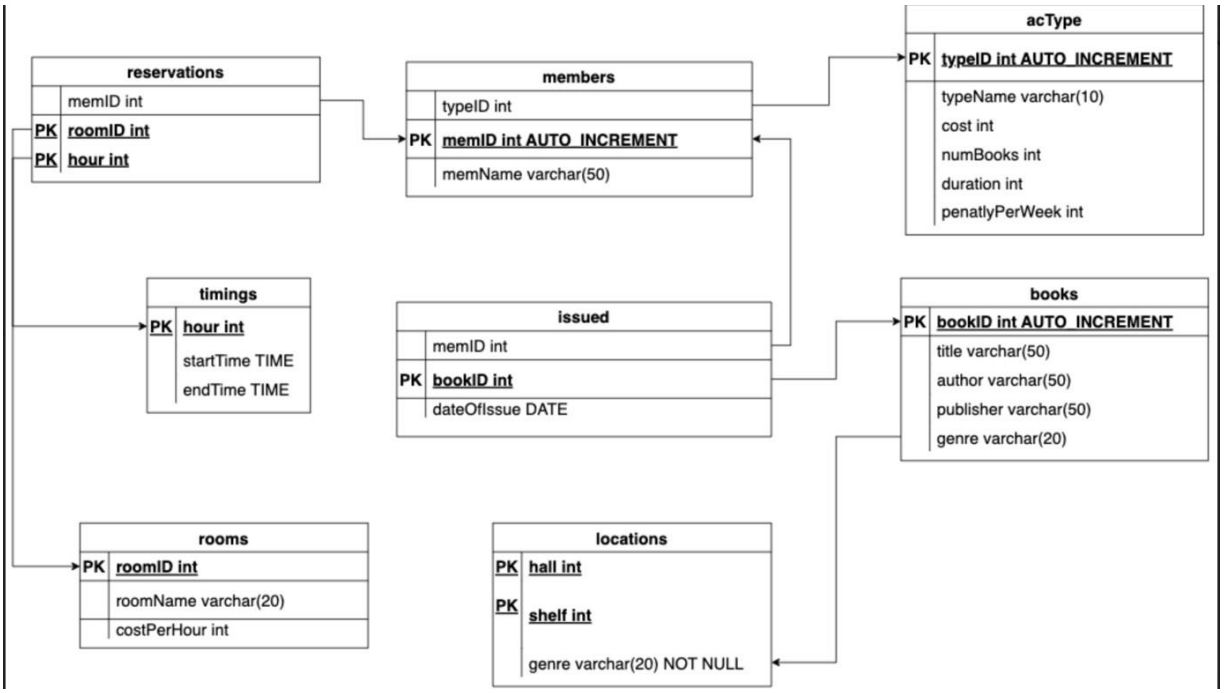
c. The list of rooms available for booking
d. The location of every genre of books

System Modelling

1. ER Diagram



2. Schema Design



3. Data Normalization

a. Members

R(memID, memName, typeID)

FD's - memID -> memName

memID -> typeID

1NF – No multivalued Attributes. Hence, in 1 NF.

2NF – Absence of partial dependency. Hence in 2NF.

3NF – No transitive dependency. Hence, in 3NF.

BCNF – LHS is always a candidate key. Hence, in BCNF.

b. acType

R(typeID, typeName, cost, numBooks, duration, penaltyPerWeek)

FD's - typeID -> typeName

typeID -> cost

typeID -> numBooks

typeID -> duration

typeID -> penaltyPerWeek

1NF – No multivalued Attributes. Hence, in 1 NF.

2NF – Absence of partial dependency. Hence in 2NF.

3NF – No transitive dependency. Hence, in 3NF.

BCNF – LHS is always a candidate key. Hence, in BCNF.

c. Books

R(bookID, title, author, publisher, genre)

FD's - bookID -> title

bookID -> author

bookID -> publisher

bookID -> genre

1NF – No multivalued Attributes. Hence, in 1 NF.

2NF – Absence of partial dependency. Hence in 2NF.

3NF – No transitive dependency. Hence, in 3NF.

BCNF – LHS is always a candidate key. Hence, in BCNF.

d. Issued

R(bookID, memID, dateOfIssue)

FD's – bookID -> memID

bookID -> dateOfIssue

1NF – No multivalued Attributes. Hence, in 1 NF.

2NF – Absence of partial dependency. Hence in 2NF.

3NF – No transitive dependency. Hence, in 3NF.

BCNF – LHS is always a candidate key. Hence, in BCNF.

e. Rooms

R(<u>roomID</u>, roomName, costPerHour)

FD's – roomID -> romName

roomID -> costPerHour

1NF – No multivalued Attributes. Hence, in 1 NF.

2NF – Absence of partial dependency. Hence in 2NF.

3NF – No transitive dependency. Hence, in 3NF.

BCNF – LHS is always a candidate key. Hence, in BCNF.

f. Timings

R(<u>hour</u>, startTime, endTime)

FD's – hour -> startTime

hour -> endTime

1NF – No multivalued Attributes. Hence, in 1 NF.

2NF – Absence of partial dependency. Hence in 2NF.

3NF – No transitive dependency. Hence, in 3NF.

BCNF – LHS is always a candidate key. Hence, in BCNF.

g. Reservations

R(<u>roomID</u>, <u>hour</u>, memID)

FD's – roomID, hour -> memID

1NF – No multivalued Attributes. Hence, in 1 NF.

2NF – Absence of partial dependency. Hence in 2NF.

3NF – No transitive dependency. Hence, in 3NF.

BCNF – LHS is always a candidate key. Hence, in BCNF.

h. Locations

R(hall, shelf, genre)

FD's – hall, shelf -> genre

1NF – No multivalued Attributes. Hence, in 1 NF.

2NF – Absence of partial dependency. Hence in 2NF.

3NF – No transitive dependency. Hence, in 3NF.

BCNF – LHS is always a candidate key. Hence, in BCNF.

4. List of Tables

a. Members
b. acType
c. Books
d. Rooms
e. Timings
f. Reservations
g. Locations
h. Rooms

5. Additional Components

a. Transactions have been implemented to ensure concurrency and consistency of thedatabase.
b. Return date is dependent on the type of account – better the account, the longer the book can be kept. Exact durations can be found in the SQL file.
c. Amount overdue is calculated on a penalty per week basis. Each week that a book is overdue, a certain amount is added to the amount overdue.