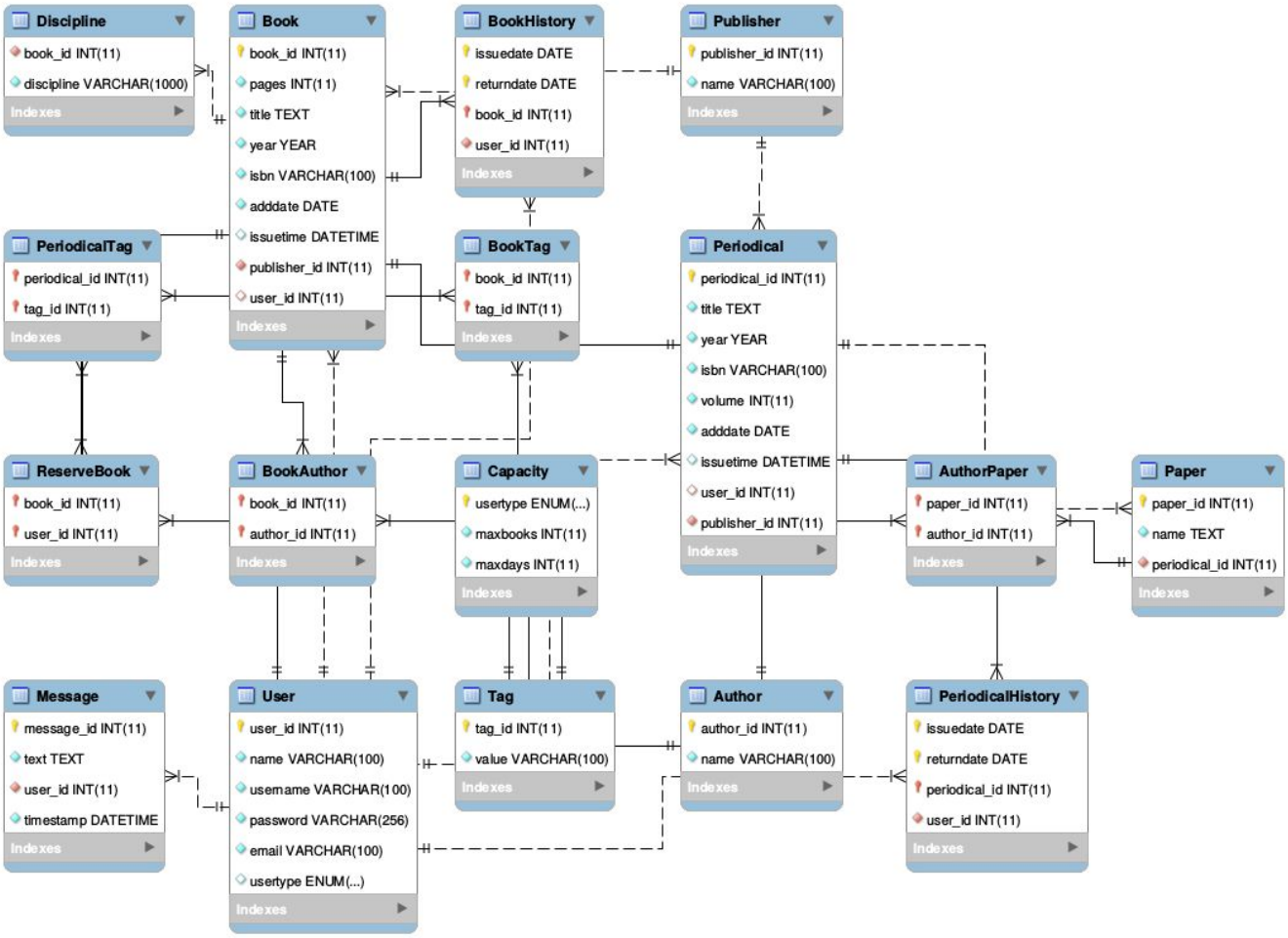
Library Management System

Vikas Gola | 2016UCS0023

Note: Checkout README.md to know 'How to Run' the implementations of questions.

Q1 and Q2 Normalised Database:

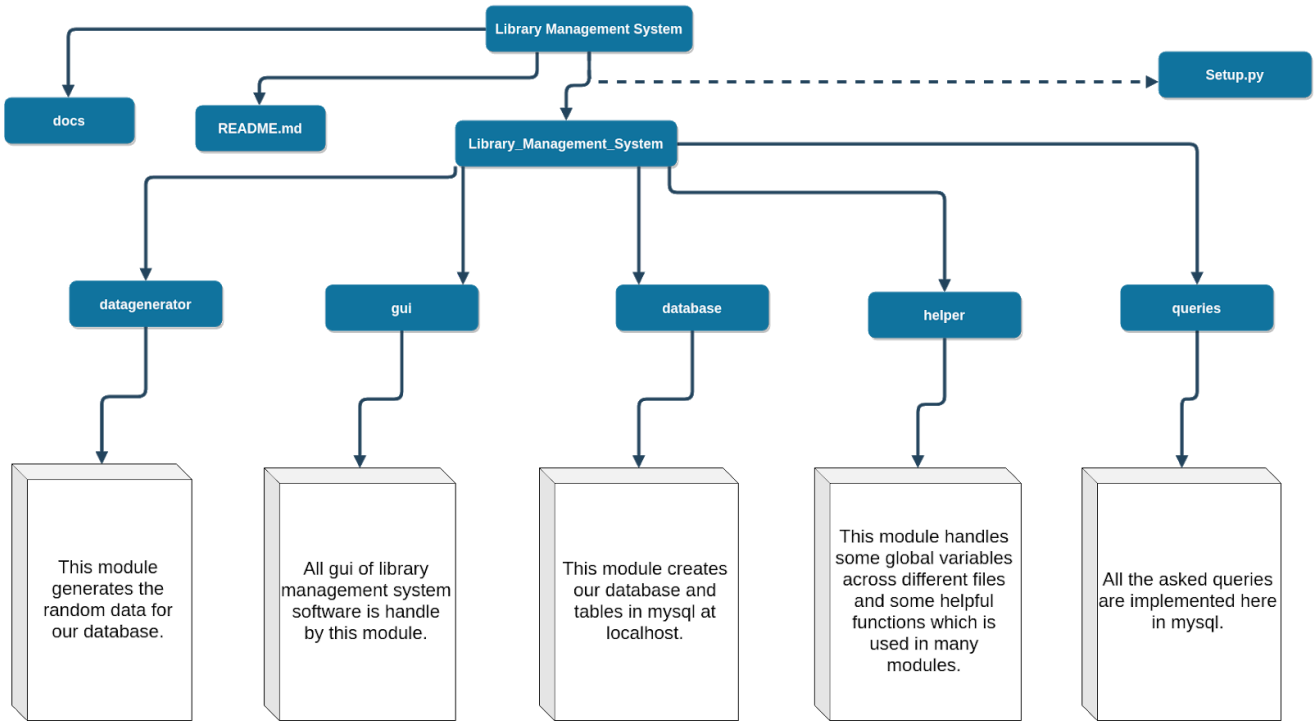


The Relational Model was made such that it is already Normalised. The model has been shown in the picture above where every block represent a table in database and each table contains some columns. Every table's name has been shown in block head. The relationship of different blocks or tables have been shown by the wires.

Q3 and Q4: Implementation and Queries

The implementation is in MYSQL using python with pymysql module. The structure of implementation of Library Management System is as follows:

File Structure of Library Management System Code



The ‘database’ module implements our complete Database with all tables in MySQL.
Data in every table have been added using ‘datagenerator’ module with generate the random data for our database.

Function ‘addUserType’ in module ‘database’ adds the ‘usertype’ column in the table ‘User’.

The ‘Capacity’ table contains all the details about how much user and for how much day user can issue Book or Periodical. This table is used whenever we have to calculate the fine of user and for finding that “if user can issue a book or not”.

All the asked queries have been implemented in ‘queries’ module in MySQL using pymysql. Queries have been implemented in functions with name respective to query number e.g. first query is implemented in function ‘query1’.

Q5: Relation Algebra

1. $\Pi\ (\sigma_{table\_schema = 'LibMS'}(\ information\_schema.columns\ ))$
2. $G_{Count(*)}(tablename)$
3. $G_{Count(*)}(\sigma_{Book.user\_id\ NOT\ NULL\ AND\ Book.book\_id = requested\_book}(Book))$
4. $\Pi_{Author.name}(\sigma_{cond}(Author \times Book \times BookAuthor))$
   Where con =
   $(Author.author\_id\ =\ BookAuhtor.author\_id\ And\ BookAuthor.book\_id\ =\ queried\_book\_id)$
5. $G_{Count(*)}\ (\sigma_{Book.user\_id = requested\_user}(Book))$
6. $\Pi_{Capacity.maxbooks}(\sigma_{cond}(Capacity\ \times\ User))$
   Where cond = (
   $Capacity.usertype\ =\ User.usertype\ AND\ User.user\_id\ =\ requested\ \_user\_id\ )$
7. $G_{Count(*)}(\sigma_{issuedate > date1\ AND\ issuedate < date2}(BookHistory))$
   $G_{Count(*)}(\sigma_{returndate > date1\ AND\ returndate < date2}(BookHistory))$
8. $\Pi_{Book.*,\ User.*}(\sigma_{cond}(Book\ \times\ User \times\ Capacity))$
   Where cond = Book.user_id = User.user_id AND (SELECT DATEDIFF(NOW(), Book.issuetime) AS days) > Capacity.maxdays AND Capacity.usertype = User.usertype
9. $\sigma_{Book.adddate\ BETWEEN\ 'date1'\ and\ 'date2'}(Book)$
10. $\Pi_{User.name,\ Book.title}(\sigma_{Book.userid = User.userid}(User \times Book))$
11. $\sigma_{Book.userid = queried\ \_user\ \_id}(Book)$

Q6: Fine Calculation

Query for calculating the fine for a particular user is implemented in ‘queries.py’ file in queries module. Also, Query for all the users which have dues on books or periodicals have implemented in same file with function name ‘question6’.

Q7: Discipline

Discipline was added to the database using function ‘addDiscipline’ which is in file ‘database.py’ contains in ‘database’ module.

Q8: GUI

Implementation of Gui of the Library Management System is present in the module ‘gui’. All the gui is build using the PyQT5 and pymysql libraries.

Q9: Reserve Issued Book

Reserve Issue Book is handle by table ‘ReserveBook’.