**Online Library Management System**

**Introduction**

The project Online Library Management system software for monitoring and keep all records of users, publisher, magazines and books in the library. This project was developed in java, which mainly focus on the basic functions in a library, like adding new users, librarian, adding new books, issued books, return books others.

1. **Gather and capture the necessary requirements (5 marks)**

**Necessary Requirements**

**Functions**

**View books-** This page of the function keeps all the function of the books.  
**Book\_count**- This page the record of serial number of the book and user ID

**Fullname-** This page contain the data for the user, like User ID,First name and Last name  
**Issuedbook-** This page contain the data for books that was taken by the user.  
**Issuedmagazine**  
librarian  
magazine  
publisher  
users

**Books** (bookID, bookName, book\_count, bookAuthor, bookEdition, shelfLocation)

**Member** (userID, userFirstName, userLastName, userEmail, userPhone) /

**Issue** (bookID, memberID, issueDate)

**Return**(bookID, memberID, returnDate)

**User Details**Name,ID,Email,Password,RegDate

**Librarian Details**  
Name, ID,Email

**Magazine Details**

MagID,VolNo,Mname,Magazine,MagazineShelf,Genre,Publisher

**Publisher Details**

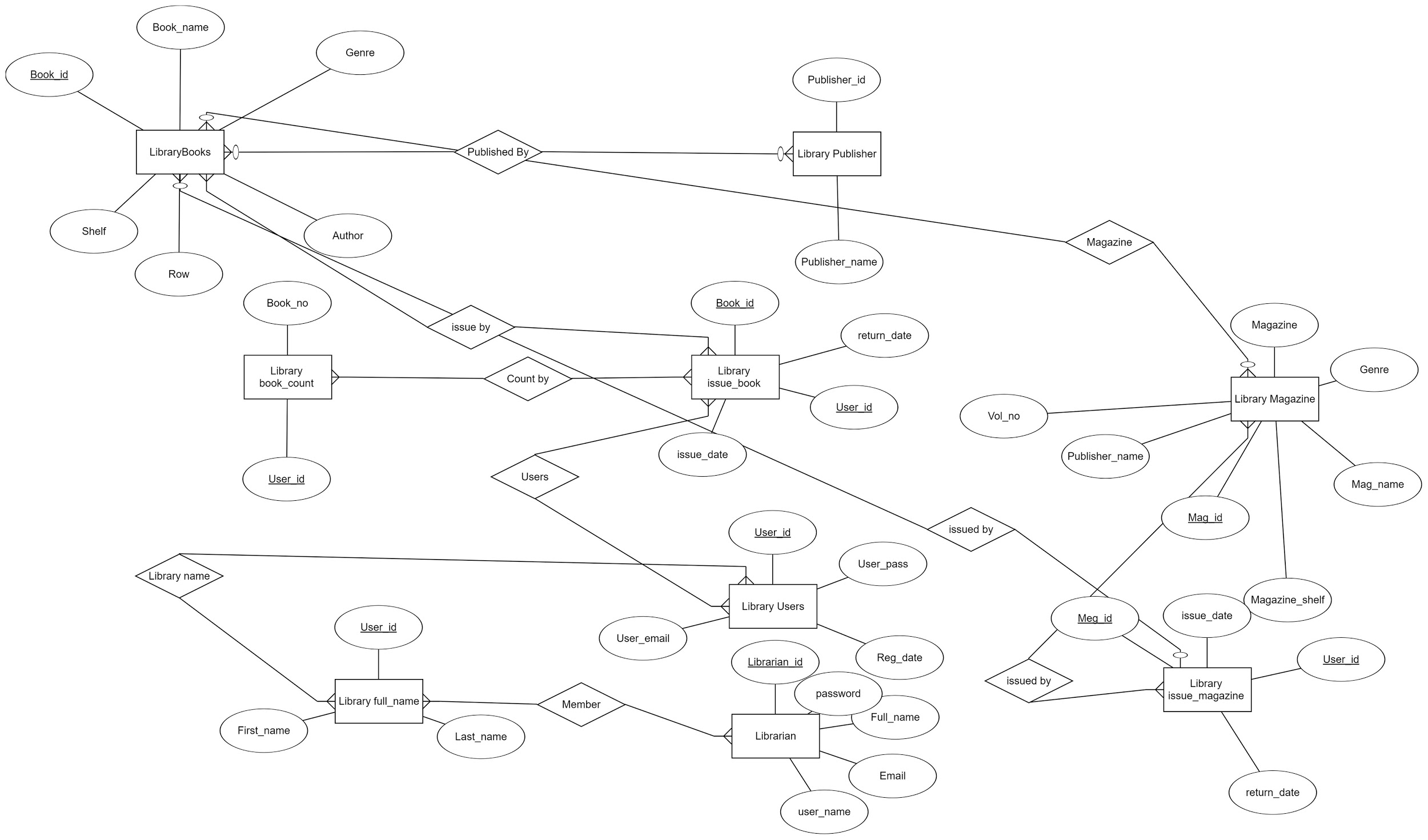
Name, ID  
**Books**

BookID, BookName, Genre, Author,Shelf,Row

**2. Model the requirement using the ER conceptual model (10 marks)**

**ER conceptual model**

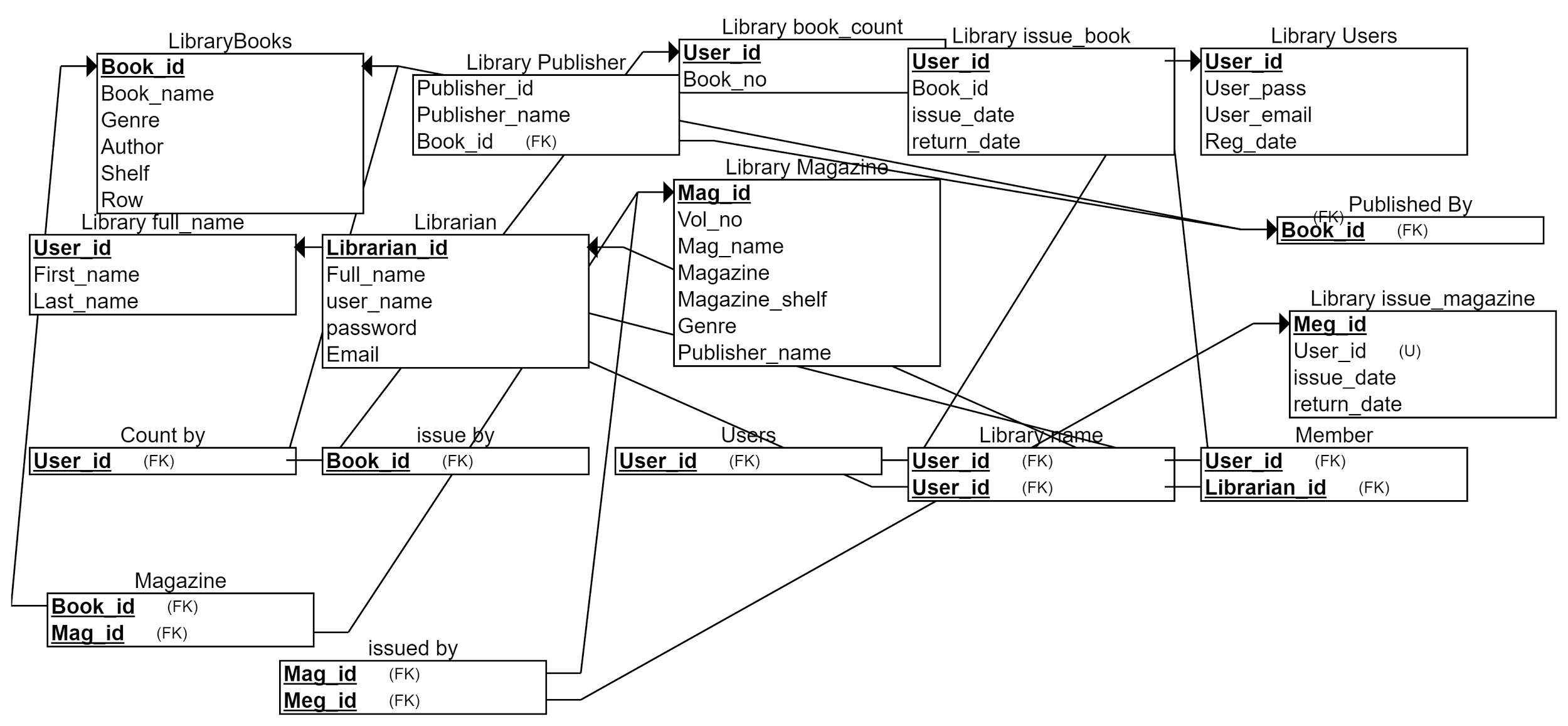
An entity relationship model (ER model) describes interrelated things of interest in a specific part of knowledge. A basic ER model is composed of entity types which is classified the things of specifies relationships that can exist between instances of entity types. Entities are characterised not only by relationships, but also by additional properties (attributes), which include identifiers called primary keys. Diagrams created to represent attributes as well as entities and relationships may be called entity attribute relationship diagrams, rather than entity relationship models.



3. **Produce a detailed database relation schema from the ER model (10 marks)**

**ER model**

An entity type within ER diagram is turned into a table for relational database schema. We need preferably keep the same name for the entity for schema or give it a sensible name but have to avoid DBMS reserved words as well as avoid the use of special characters.  
Each attribute turns into a column (attribute) in the table. The key attribute of the entity is the primary key of the table which is usually underlined. It can be composite if required but can never be null.



**4. Create the DDL to generate your schema (5 marks)**

**DDL**

Data Definition Language (DDL) is a standard for commands that define the different structures in a database. DDL statements create, modify, and remove database objects such as tables, indexes, and users. Common DDL statements are CREATE, ALTER, and DROP. We got this from auto DDL generated by **Phpmyadmin.**

-- phpMyAdmin SQL Dump  
-- version 4.7.9  
-- https://www.phpmyadmin.net/  
--  
-- Host: 127.0.0.1:3308  
-- Generation Time: May 03, 2018 at 07:39 AM  
-- Server version: 5.7.21  
-- PHP Version: 5.6.35  
  
SET SQL\_MODE = "NO\_AUTO\_VALUE\_ON\_ZERO";  
SET AUTOCOMMIT = 0;  
START TRANSACTION;  
SET time\_zone = "+00:00";  
  
--  
-- Database: `library`  
--  
CREATE DATABASE IF NOT EXISTS `library` DEFAULT CHARACTER SET utf8 COLLATE utf8\_general\_ci;  
USE `library`;  
  
-- --------------------------------------------------------

--  
-- Table structure for table `books`  
--  
DROP TABLE IF EXISTS `books`;  
CREATE TABLE IF NOT EXISTS `books` (  
 `BookID` int(11) NOT NULL AUTO\_INCREMENT,  
 `BookName` varchar(50) NOT NULL,  
 `Genre` varchar(10) NOT NULL,  
 `Author` varchar(30) DEFAULT NULL,  
 `Publisher` varchar(30) DEFAULT NULL,  
 `Shelf` varchar(5) NOT NULL,  
 `Row` varchar(5) NOT NULL,  
 PRIMARY KEY (`BookID`)  
) ENGINE=InnoDB AUTO\_INCREMENT=21 DEFAULT CHARSET=latin1;  
  
--  
-- Dumping data for table `books`  
--  
  
INSERT INTO `books` (`BookID`, `BookName`, `Genre`, `Author`, `Publisher`, `Shelf`, `Row`) VALUES  
(3, 'Harry Potter and Goblet of Fire', 'Fiction', 'J. K. Rowling', 'Pottermore', '1', 'B'),  
(4, 'Harry Potter and Deathly Hallow', 'Fiction', 'J. K. Rowling', 'Pottermore', '2', '23'),  
(5, 'Famous Five', 'sd', 'ds', 'dsd', '3', '3'),  
(6, 'akhjkd', 'hdfdj', 'jkshdkjh', 'hkjdfh', '4', 'd'),  
(13, 'The da Vinci Code', 'Thriller', 'Dan Brown', 'Doubleday', '5', '5'),  
(14, 'Pride and Prejudice', 'Romantic', 'Alexander Dumas', 'Pearson', '6', '9'),  
(15, 'To Kill A Mocking Bird', 'Emotional', 'Harper Lee', 'McGraw', '7', '8'),  
(16, 'The Perks Of being A Wallflower', 'Drama', 'Stephen Chbosky', 'Klett', '8', '1'),  
(17, 'The Hunger Games', 'Action', 'Suzanne Collins', 'Pearson', '9', '7'),  
(18, 'Madhushala', 'Life', 'H R Bacchan', 'Pushpalata', '10', '6'),  
(19, 'V for Vendetta ', 'Action', 'Alan Moore', 'Cambridge', '11', '9'),  
(20, 'The Immortalists', '', ' Chloe Benjamin', ' TIMES BESTSELLER', '12', '5');  
  
-- --------------------------------------------------------  
  
--  
-- Table structure for table `book\_count`  
--  
  
DROP TABLE IF EXISTS `book\_count`;  
CREATE TABLE IF NOT EXISTS `book\_count` (  
 `BookNo` int(11) DEFAULT '0',  
 `UserID` int(11) NOT NULL  
) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8\_bin;  
  
--  
-- Dumping data for table `book\_count`  
--  
INSERT INTO `book\_count` (`BookNo`, `UserID`) VALUES  
(1, 1),  
(2, 2),  
(0, 3),  
(0, 4),  
(0, 5);  
  
-- --------------------------------------------------------  
--  
-- Table structure for table `fullname`  
--  
DROP TABLE IF EXISTS `fullname`;  
CREATE TABLE IF NOT EXISTS `fullname` (  
 `UserID` int(11) DEFAULT NULL,  
 `FirstName` varchar(30) NOT NULL,  
 `LastName` varchar(30) DEFAULT NULL  
) ENGINE=InnoDB DEFAULT CHARSET=latin1;  
  
-- --------------------------------------------------------  
-- Table structure for table `issuedbook`  
--  
  
DROP TABLE IF EXISTS `issuedbook`;  
CREATE TABLE IF NOT EXISTS `issuedbook` (  
 `BookID` int(11) DEFAULT NULL,  
 `UserID` int(11) DEFAULT NULL,  
 `IssueDate` date NOT NULL,  
 `ReturnDate` date NOT NULL  
) ENGINE=InnoDB DEFAULT CHARSET=latin1;  
--  
-- Dumping data for table `issuedbook`  
--  
INSERT INTO `issuedbook` (`BookID`, `UserID`, `IssueDate`, `ReturnDate`) VALUES  
(5, 1, '2016-11-17', '2016-12-02'),  
(12, 2, '2016-11-17', '2016-12-02'),  
(6, 2, '2016-11-17', '2016-12-02');  
--  
-- Triggers `issuedbook`  
--  
DROP TRIGGER IF EXISTS `IssuedBook\_AFTER\_INSERT`;  
DELIMITER $$  
CREATE TRIGGER `IssuedBook\_AFTER\_INSERT` AFTER INSERT ON `issuedbook` FOR EACH ROW BEGIN  
 UPDATE Book\_Count   
 SET BookNo = BookNo+1  
 where UserID = NEW.UserID;  
END  
$$  
DELIMITER ;  
DROP TRIGGER IF EXISTS `count`;  
DELIMITER $$  
CREATE TRIGGER `count` AFTER DELETE ON `issuedbook` FOR EACH ROW BEGIN  
 UPDATE Book\_Count   
 SET BookNo = BookNo-1  
 where Book\_Count.UserID = OLD.UserID;  
END  
$$  
DELIMITER ;  
-- --------------------------------------------------------  
--  
-- Table structure for table `issuedmagazine`  
--  
DROP TABLE IF EXISTS `issuedmagazine`;  
CREATE TABLE IF NOT EXISTS `issuedmagazine` (  
 `MagID` int(11) DEFAULT NULL,  
 `UserID` int(11) DEFAULT NULL,  
 `IssueDate` date NOT NULL,  
 `ReturnDate` date NOT NULL  
) ENGINE=InnoDB DEFAULT CHARSET=latin1;  
--  
-- Triggers `issuedmagazine`  
--  
DROP TRIGGER IF EXISTS `IssuedMagazine\_AFTER\_DELETE`;  
DELIMITER $$  
CREATE TRIGGER `IssuedMagazine\_AFTER\_DELETE` AFTER DELETE ON `issuedmagazine` FOR EACH ROW BEGIN  
UPDATE Book\_Count  
SET BookNo = BookNo  
where Book\_Count.UserID = OLD.UserID;  
END  
$$  
DELIMITER ;  
DROP TRIGGER IF EXISTS `IssuedMagazine\_AFTER\_INSERT`;  
DELIMITER $$  
CREATE TRIGGER `IssuedMagazine\_AFTER\_INSERT` AFTER INSERT ON `issuedmagazine` FOR EACH ROW BEGIN  
UPDATE Book\_Count  
SET BookNo = BookNo +1   
where Book\_Count.UserID = NEW.UserID;  
END  
$$  
DELIMITER ;  
-- --------------------------------------------------------  
--  
-- Table structure for table `librarian`  
--  
DROP TABLE IF EXISTS `librarian`;  
CREATE TABLE IF NOT EXISTS `librarian` (  
 `LibrarianID` int(11) NOT NULL AUTO\_INCREMENT,  
 `FullName` varchar(30) NOT NULL,  
 `UserName` varchar(30) NOT NULL,  
 `Password` varchar(30) NOT NULL,  
 `Email` varchar(30) NOT NULL,  
 PRIMARY KEY (`LibrarianID`)  
) ENGINE=InnoDB AUTO\_INCREMENT=3 DEFAULT CHARSET=latin1;  
--  
-- Dumping data for table `librarian`  
--  
INSERT INTO `librarian` (`LibrarianID`, `FullName`, `UserName`, `Password`, `Email`) VALUES  
(1, 'test', 'test', '1234', 'test@gmail.com'),  
(2, 'test1', 'test1', '1234', 'test1@gmail.com');  
  
-- --------------------------------------------------------  
--  
-- Table structure for table `magazine`  
--  
  
DROP TABLE IF EXISTS `magazine`;  
CREATE TABLE IF NOT EXISTS `magazine` (  
 `MagID` int(11) NOT NULL AUTO\_INCREMENT,  
 `VolNo` int(11) NOT NULL,  
 `Mname` varchar(30) NOT NULL,  
 `Magazine` varchar(30) NOT NULL,  
 `MagazineShelf` varchar(10) NOT NULL,  
 `Genre` varchar(10) NOT NULL,  
 `Publisher` varchar(10) NOT NULL,  
 PRIMARY KEY (`MagID`)  
) ENGINE=InnoDB DEFAULT CHARSET=latin1;  
  
-- --------------------------------------------------------  
  
--  
-- Table structure for table `publisher`  
--  
  
DROP TABLE IF EXISTS `publisher`;  
CREATE TABLE IF NOT EXISTS `publisher` (  
 `PublisherID` int(11) NOT NULL AUTO\_INCREMENT,  
 `PublisherName` varchar(30) DEFAULT NULL,  
 PRIMARY KEY (`PublisherID`)  
) ENGINE=InnoDB AUTO\_INCREMENT=24 DEFAULT CHARSET=latin1;  
  
----  
  
INSERT INTO `publisher` (`PublisherID`, `PublisherName`) VALUES  
(1, 'hjkhdkj'),  
(2, 'Pottermore'),  
(3, 'dsd'),  
(4, 'Whittaker'),  
-- Dumping data for table `publisher`  
  
(5, 'ewhkje'),  
(6, 'dhskj'),  
(7, 'hkjdfh'),  
(8, 'gsjf'),  
(9, 'dshh'),  
(10, 'gdsh'),  
(11, 'Water'),  
(12, 'ew'),  
(13, 'dgjs'),  
(14, 'SGAJ'),  
(15, 'WHOKNOW'),  
(16, 'Doubleday'),  
(17, 'Pearson'),  
(18, 'McGraw'),  
(19, 'Klett'),  
(20, 'Pushpalata'),  
(21, 'Cambridge'),  
(22, 'TMH'),  
(23, ' TIMES BESTSELLER');  
  
-- --------------------------------------------------------  
  
--  
-- Table structure for table `users`  
--  
  
DROP TABLE IF EXISTS `users`;  
CREATE TABLE IF NOT EXISTS `users` (  
 `UserID` int(11) NOT NULL AUTO\_INCREMENT,  
 `UserPass` varchar(30) NOT NULL,  
 `RegDate` date NOT NULL,  
 `UserName` varchar(30) NOT NULL,  
 `Email` varchar(30) NOT NULL,  
 PRIMARY KEY (`UserID`)  
) ENGINE=InnoDB AUTO\_INCREMENT=6 DEFAULT CHARSET=latin1;  
--  
-- Dumping data for table `users`  
--  
  
INSERT INTO `users` (`UserID`, `UserPass`, `RegDate`, `UserName`, `Email`) VALUES  
(1, '1234', '2012-05-15', 'sree', 'sree@gmail.com'),  
(2, '1234', '2016-11-07', 'bee', 'bee@gmail.com'),  
(3, '1234', '2016-11-25', 'salam', 'salam@gmail.com'),  
(4, '1234', '2018-05-03', 'halim', 'halim@gmail.com'),  
(5, '1234', '2018-05-03', 'ibrahim', 'ibrahim@gmail.com');  
--  
-- Triggers `users`  
--  
DROP TRIGGER IF EXISTS `Users\_AFTER\_INSERT`;  
DELIMITER $$  
CREATE TRIGGER `Users\_AFTER\_INSERT` AFTER INSERT ON `users` FOR EACH ROW BEGIN  
INSERT INTO Book\_Count  
values(0,New.UserID);  
END  
$$  
DELIMITER ;

COMMIT;

**5. Data dictionary (5 marks)**

**Data dictionary**

A data dictionary is a collection of descriptions of the data objects or items in a data model for the benefit of programmers and others who need to refer to them. A first step in analyzing a system of objects with which users interact is to identify each object and its relationship to other objects.

## books

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Column** | **Type** | **Null** | **Default** | **Comments** |
| BookID *(Primary)* | int(11) | No |  |  |
| BookName | varchar(50) | No |  |  |
| Genre | varchar(10) | No |  |  |
| Author | varchar(30) | Yes | *NULL* |  |
| Publisher | varchar(30) | Yes | *NULL* |  |
| Shelf | varchar(5) | No |  |  |
| Row | varchar(5) | No |  |  |

### Indexes

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Keyname** | **Type** | **Unique** | **Packed** | **Column** | **Cardinality** | **Collation** | **Null** | **Comment** |
| PRIMARY | BTREE | Yes | No | BookID | 12 | A | No |  |

## book\_count

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Column** | **Type** | **Null** | **Default** | **Comments** |
| BookNo | int(11) | Yes | 0 |  |
| UserID | int(11) | No |  |  |

## fullname

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Column** | **Type** | **Null** | **Default** | **Comments** |
| UserID | int(11) | Yes | *NULL* |  |
| FirstName | varchar(30) | No |  |  |
| LastName | varchar(30) | Yes | *NULL* |  |

## issuedbook

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Column** | **Type** | **Null** | **Default** | **Comments** |
| BookID | int(11) | Yes | *NULL* |  |
| UserID | int(11) | Yes | *NULL* |  |
| IssueDate | date | No |  |  |
| ReturnDate | date | No |  |  |

## issuedmagazine

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Column** | **Type** | **Null** | **Default** | **Comments** |
| MagID | int(11) | Yes | *NULL* |  |
| UserID | int(11) | Yes | *NULL* |  |
| IssueDate | date | No |  |  |
| ReturnDate | date | No |  |  |

## 

## librarian

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Column** | **Type** | **Null** | **Default** | **Comments** |
| LibrarianID *(Primary)* | int(11) | No |  |  |
| FullName | varchar(30) | No |  |  |
| UserName | varchar(30) | No |  |  |
| Password | varchar(30) | No |  |  |
| Email | varchar(30) | No |  |  |

### Indexes

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Keyname** | **Type** | **Unique** | **Packed** | **Column** | **Cardinality** | **Collation** | **Null** | **Comment** |
| PRIMARY | BTREE | Yes | No | LibrarianID | 2 | A | No |  |

## magazine

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Column** | **Type** | **Null** | **Default** | **Comments** |
| MagID *(Primary)* | int(11) | No |  |  |
| VolNo | int(11) | No |  |  |
| Mname | varchar(30) | No |  |  |
| Magazine | varchar(30) | No |  |  |
| MagazineShelf | varchar(10) | No |  |  |
| Genre | varchar(10) | No |  |  |
| Publisher | varchar(10) | No |  |  |

### 

### Indexes

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Keyname** | **Type** | **Unique** | **Packed** | **Column** | **Cardinality** | **Collation** | **Null** | **Comment** |
| PRIMARY | BTREE | Yes | No | MagID | 0 | A | No |  |

## publisher

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Column** | **Type** | **Null** | **Default** | **Comments** |
| PublisherID *(Primary)* | int(11) | No |  |  |
| PublisherName | varchar(30) | Yes | *NULL* |  |

### Indexes

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Keyname** | **Type** | **Unique** | **Packed** | **Column** | **Cardinality** | **Collation** | **Null** | **Comment** |
| PRIMARY | BTREE | Yes | No | PublisherID | 23 | A | No |  |

## users

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Column** | **Type** | **Null** | **Default** | **Comments** |
| UserID *(Primary)* | int(11) | No |  |  |
| UserPass | varchar(30) | No |  |  |
| RegDate | date | No |  |  |
| UserName | varchar(30) | No |  |  |
| Email | varchar(30) | No |  |  |

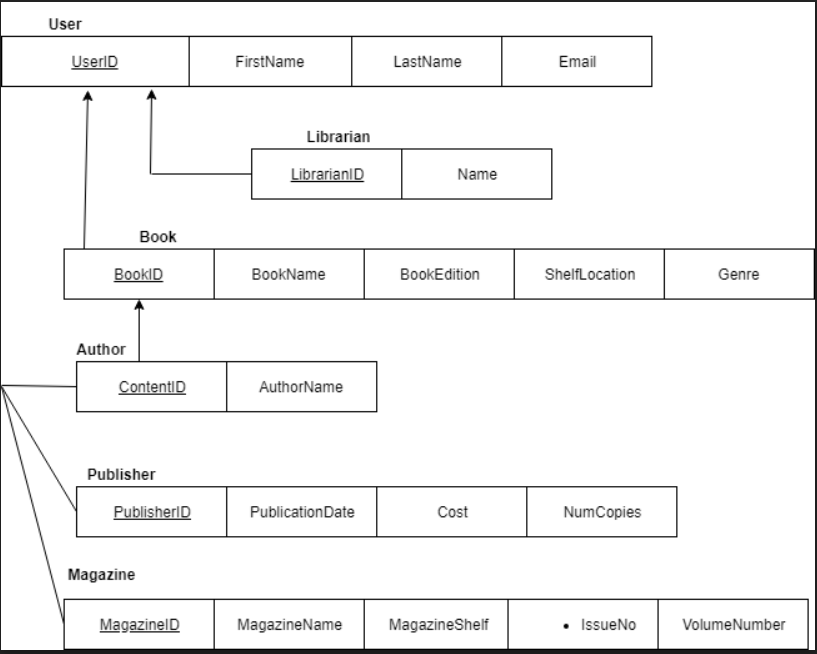
### Indexes

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Keyname** | **Type** | **Unique** | **Packed** | **Column** | **Cardinality** | **Collation** | **Null** | **Comment** |
| PRIMARY | BTREE | Yes | No | UserID | 5 | A | No |  |

6. Derive a set of functional dependencies (FD) and data constraints (e.g., the primary and foreign keys) (5 marks)

**Functional dependencies**

Functional dependency is a relationship that exists when one attribute uniquely determines another attribute. The library functional dependency shows high level functions like User, book, author, publisher and magazine. Well organized with data area to track better functionalities.

****

7.Use FDs to normalize your original database schema into 3NF. (5 marks)

**Normalization to 3NF**

Database Normalization is a technique of organizing the data in the database. Normalization is a systematic approach of decomposing tables to eliminate data redundancy (repetition) and undesirable characteristics like Insertion, Updation and Deletion Anomalies. It is a multi-step process that puts data into tabular form, removing duplicated data from the relation tables.

**UNF**

**User** (UserID,FirstName,LastName,Email,Phone,LibrarianID,LibrarianName,BookID, BookName, ISBN,BookEdition,ShelfLocation,Genre, ContentID,Author,AuthorName,PublisherID,PublicationDate,Cost, NumCopies,MagazineID,MagazineName, MagazineShelf,VolumeNo, IssueNo)

**1NF**

**User** ( UserID, FirstName, LastName, Email, Phone, LibrarianID, LibrarianName)

**Book**  (BookID, BookName, ISBN, BookEdition, NumCopies, VolumeNo, IssueNo, Author, AuthorName, ContentID, ShelfLocation, PublisherID, PublisherDate, Cost, NumCopies, MagazineID,MagazineShelf, VolumeNo, IssueNo)

**UserLine**  ( BookID, BookName, PublisherID(PK), BookEdition)

**2NF**

**User**  ( UserID, FirstName, LastName, Email, Phone, LibrarianID, LibrarianName)

**Book** (BookID, BookName, ISBN, BookEdition, NumCopies, VolumeNo, IssueNo, Author, AuthorName, ContentID, ShelfLocation, PublisherID, PublisherDate, Cost, NumCopies, MagazineID,MagazineShelf, VolumeNo, IssueNo)

**UserLine** ( BookID, BookName, PublisherID(PK), BookEdition)

**3NF**

**USER (**UserID, LibrarianID, BookIDpk)

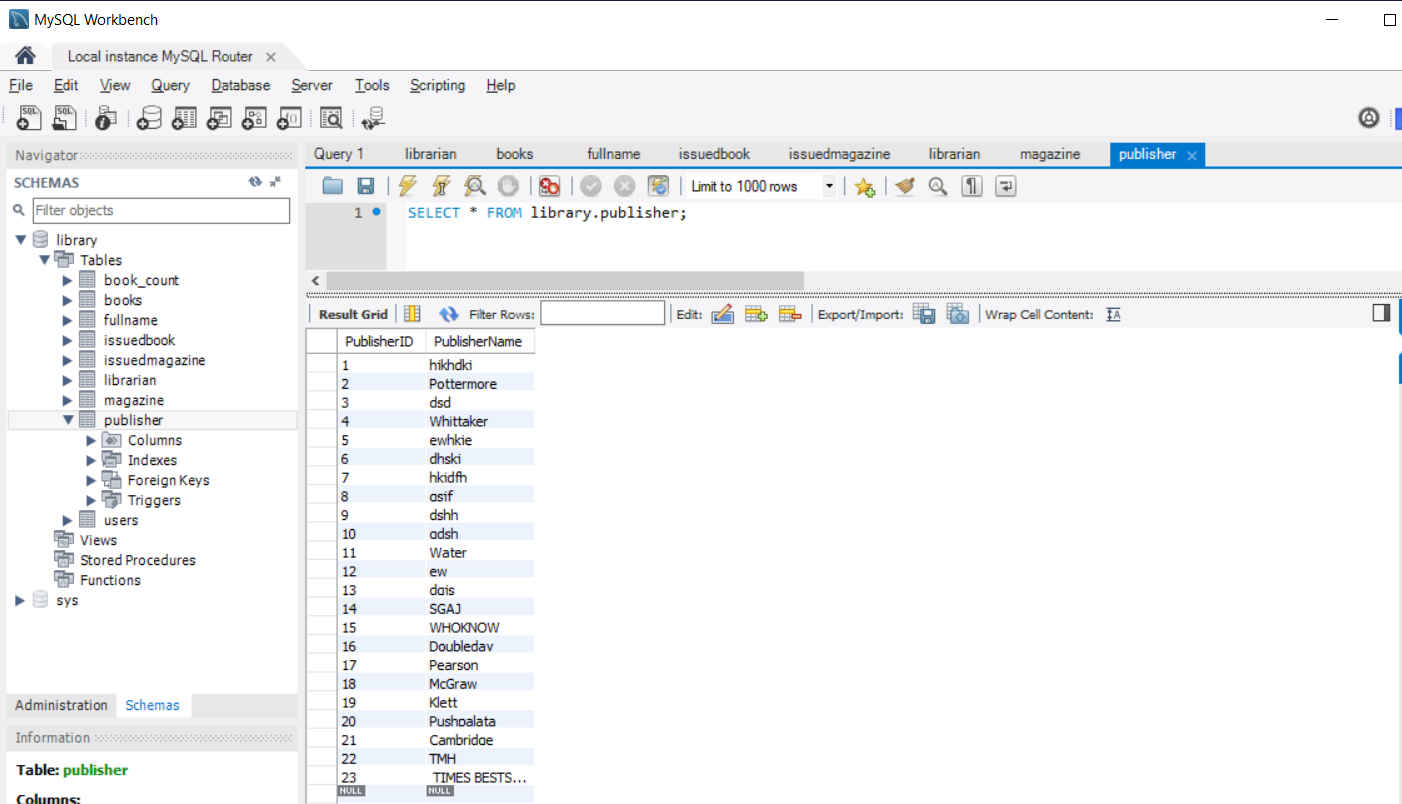
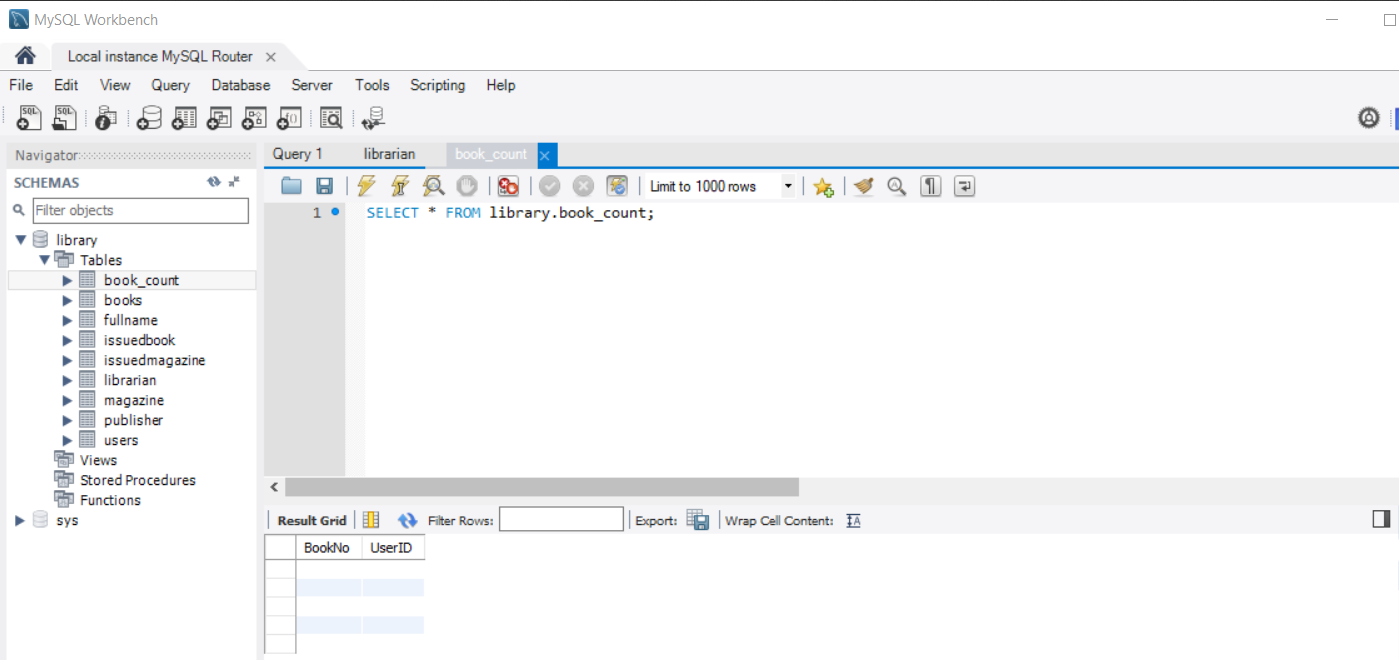
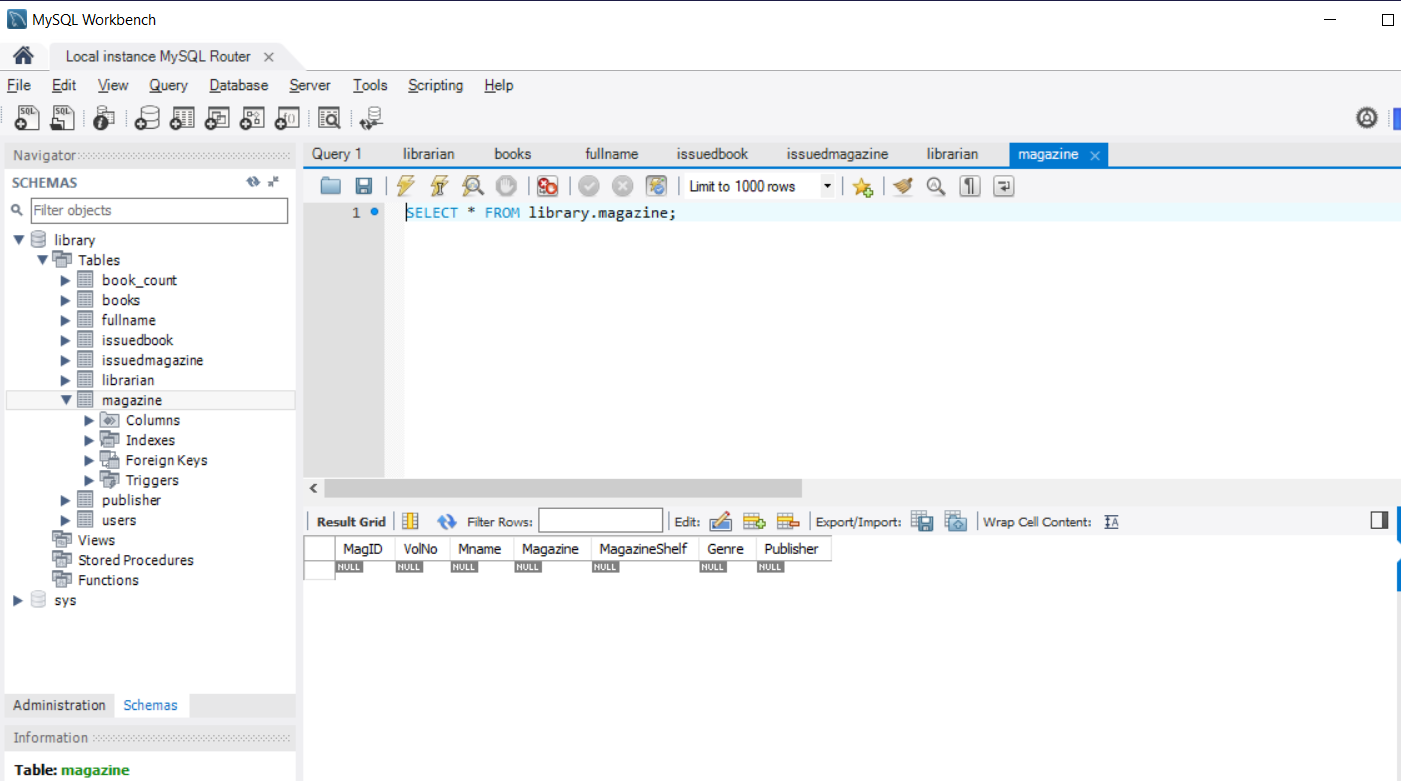
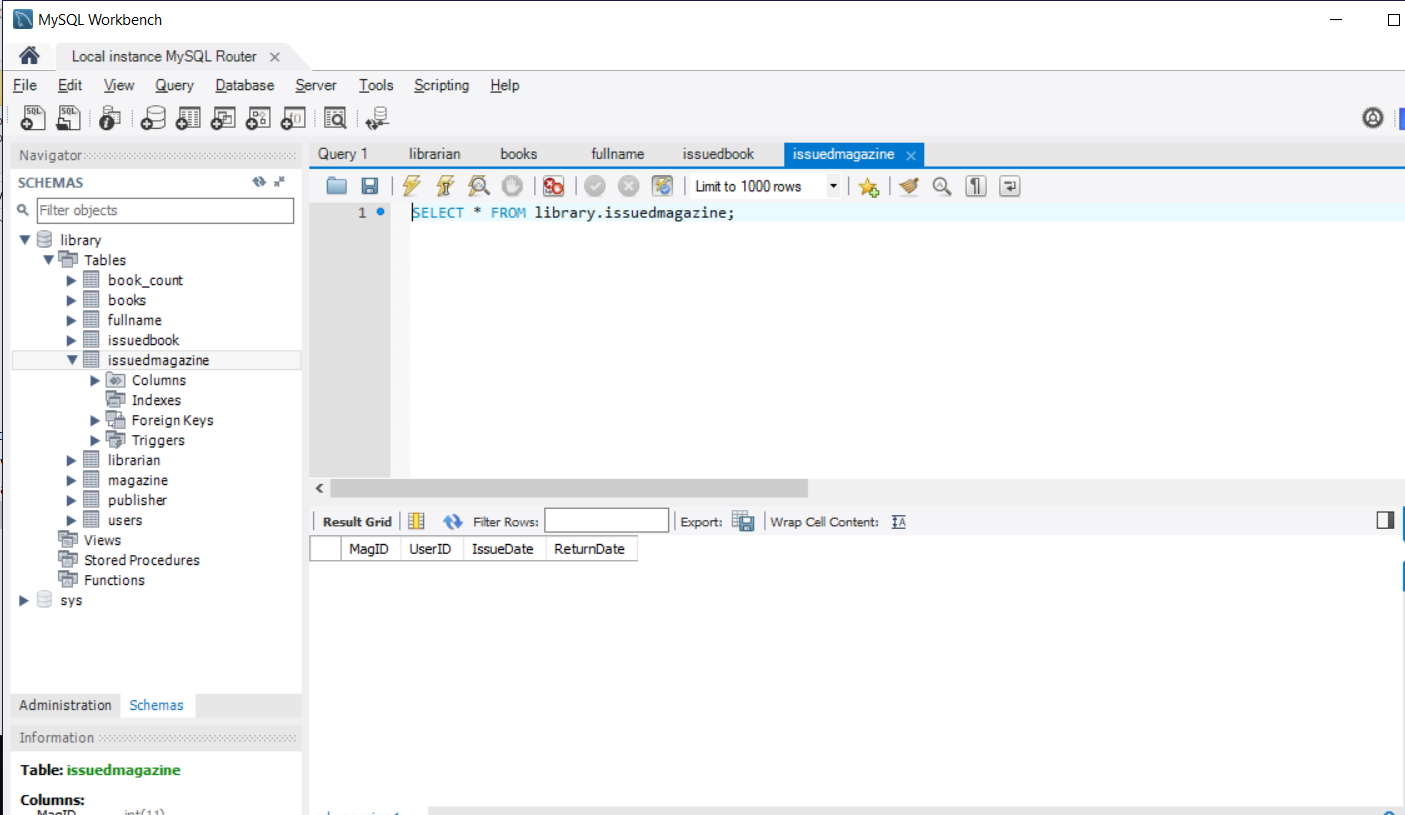
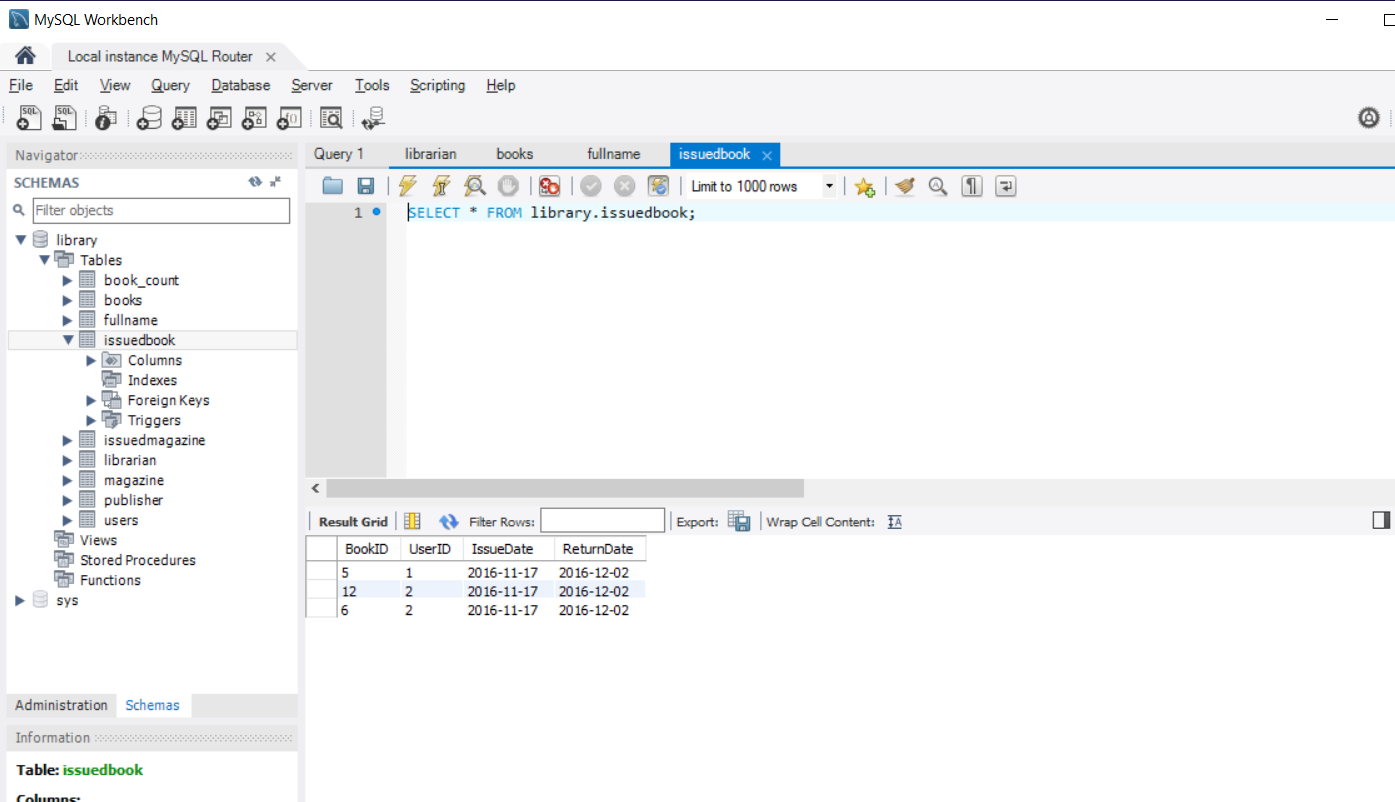
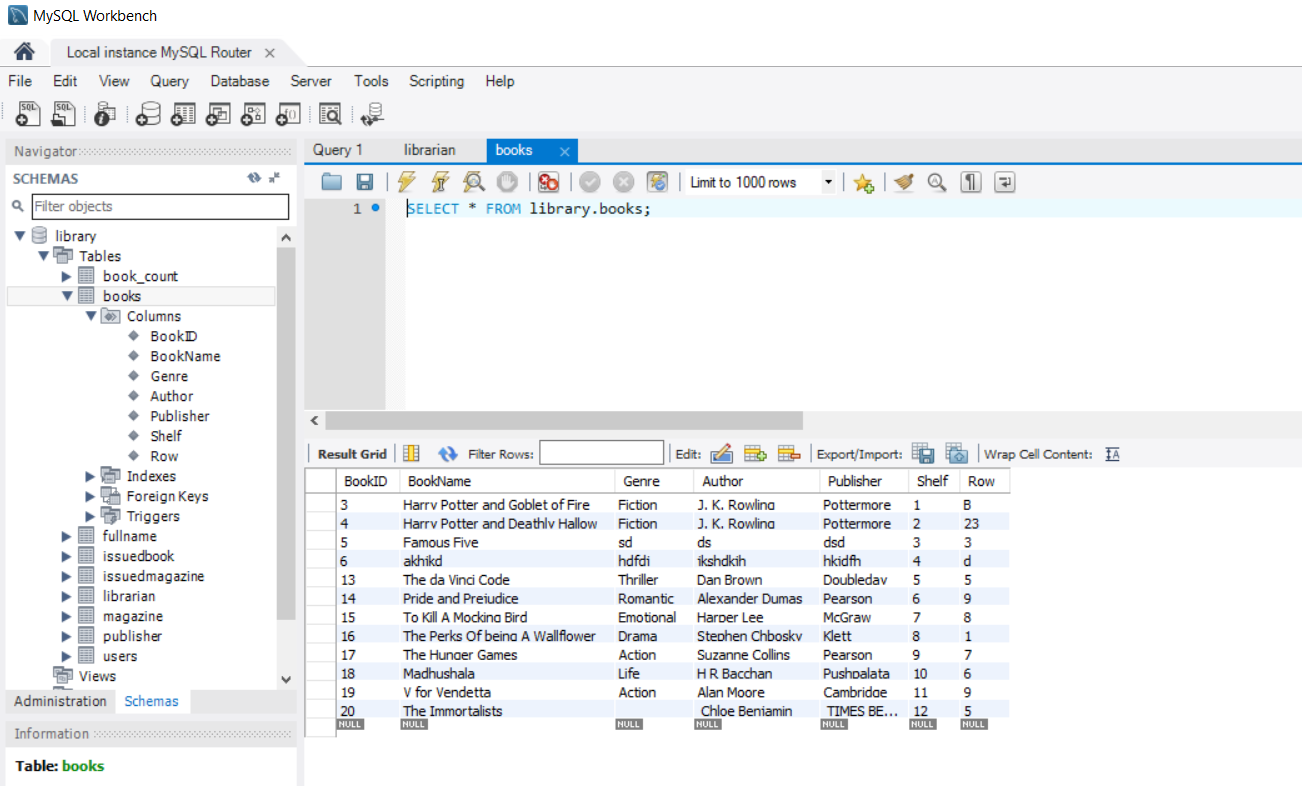
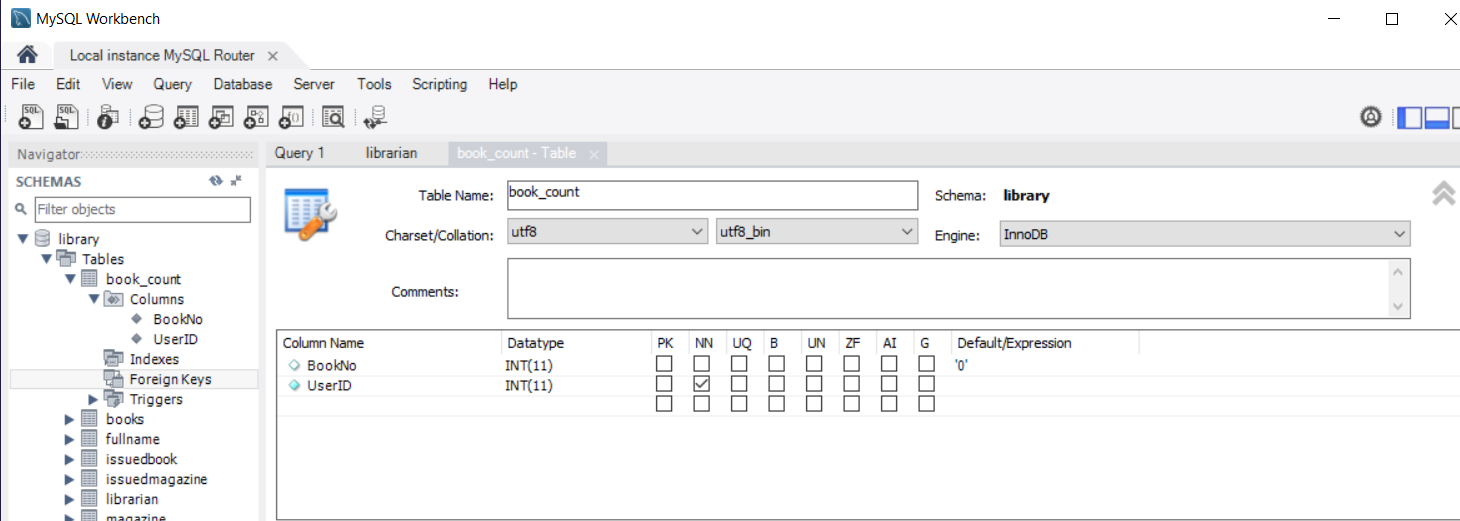
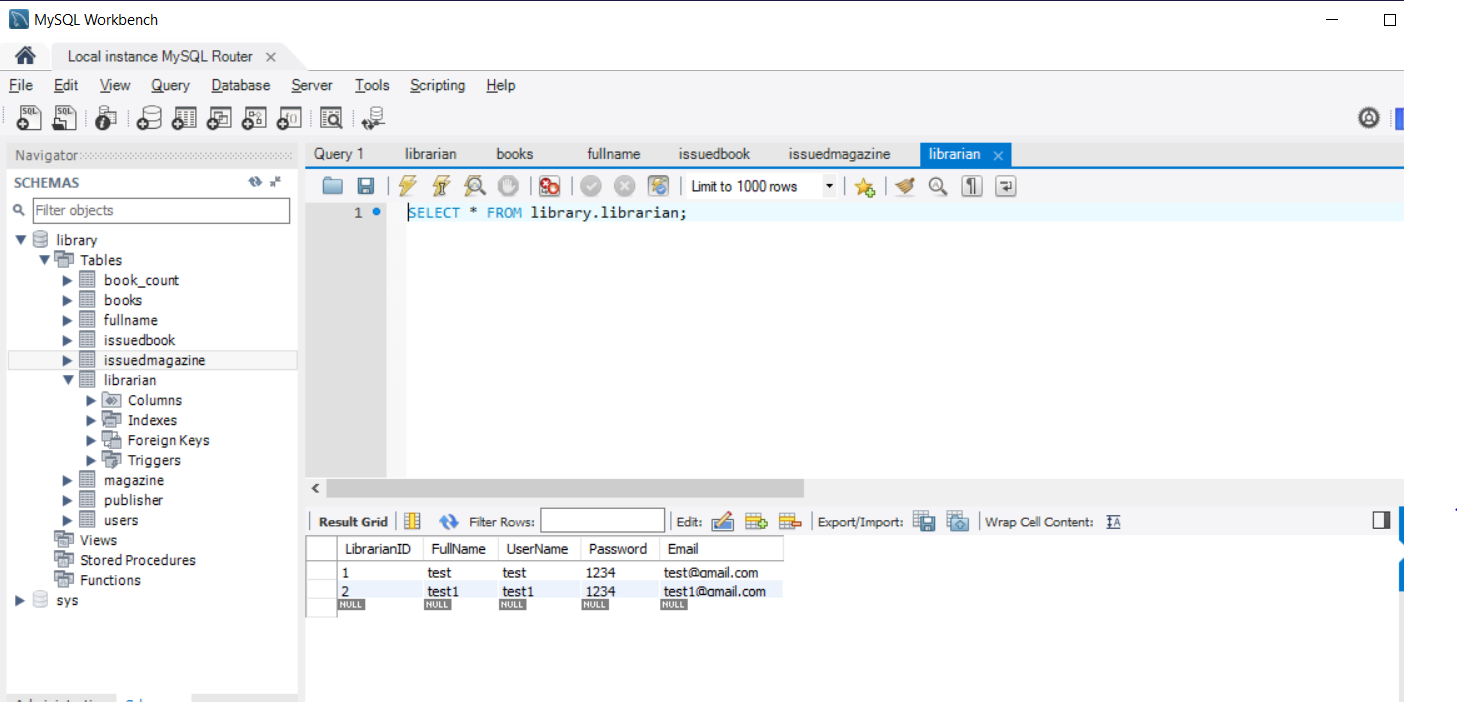
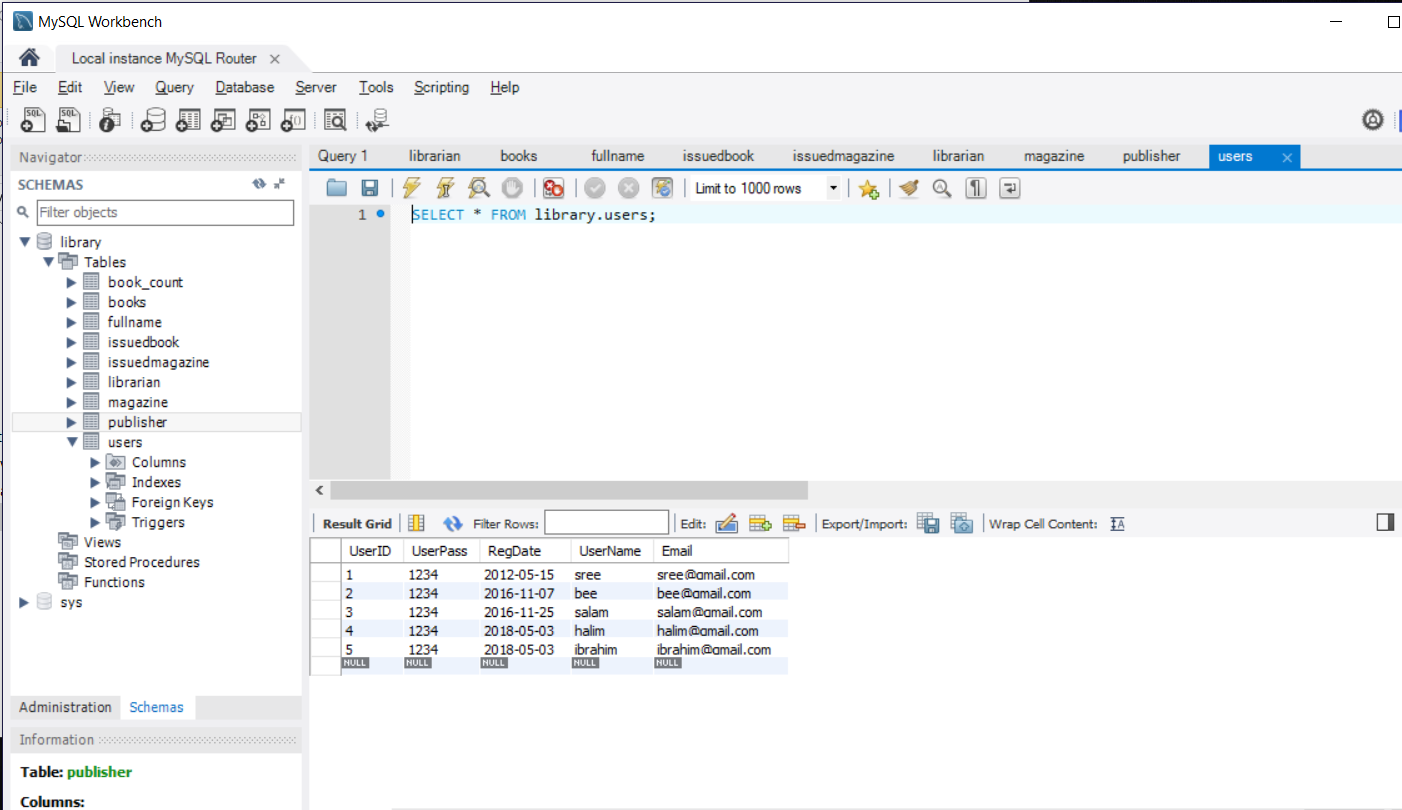
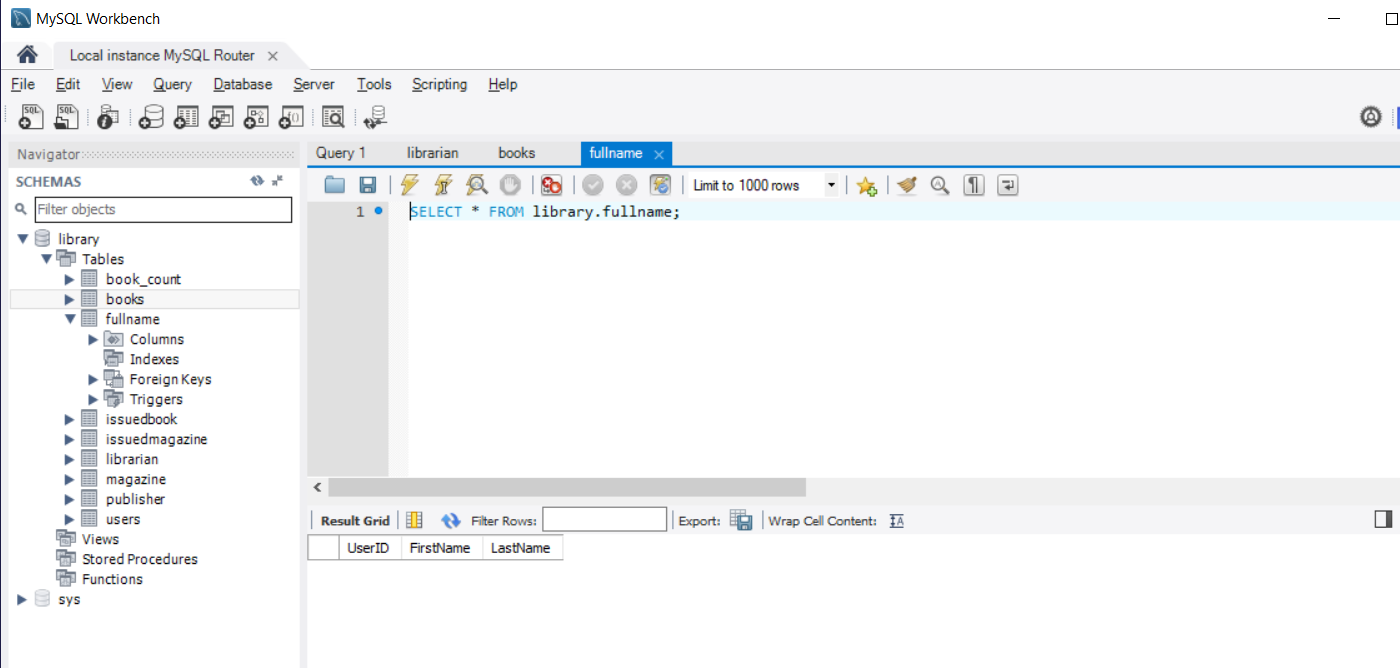
**Book**  (BookID, BookEdition, IssueNop)

**UserLine** (BookID, BookName, PublisherID)

**8. The online DBMS application based on the topic above (use MySQL). (30 marks)**

**Database**

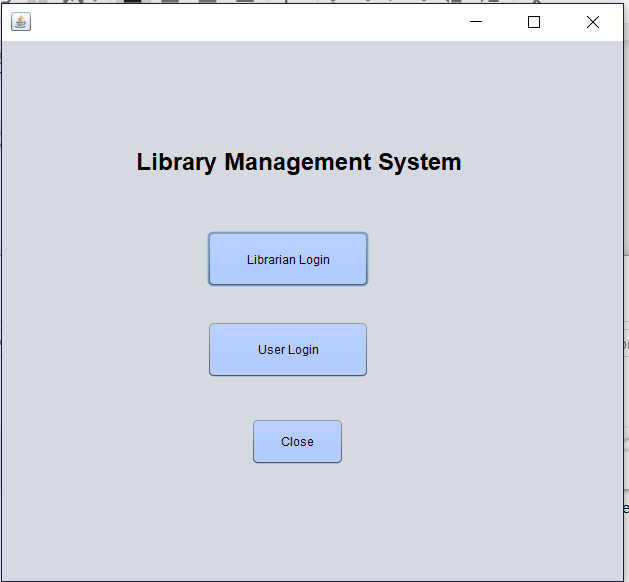
A database is a collection of information that is organized so that it can be easily accessed, managed and updated.

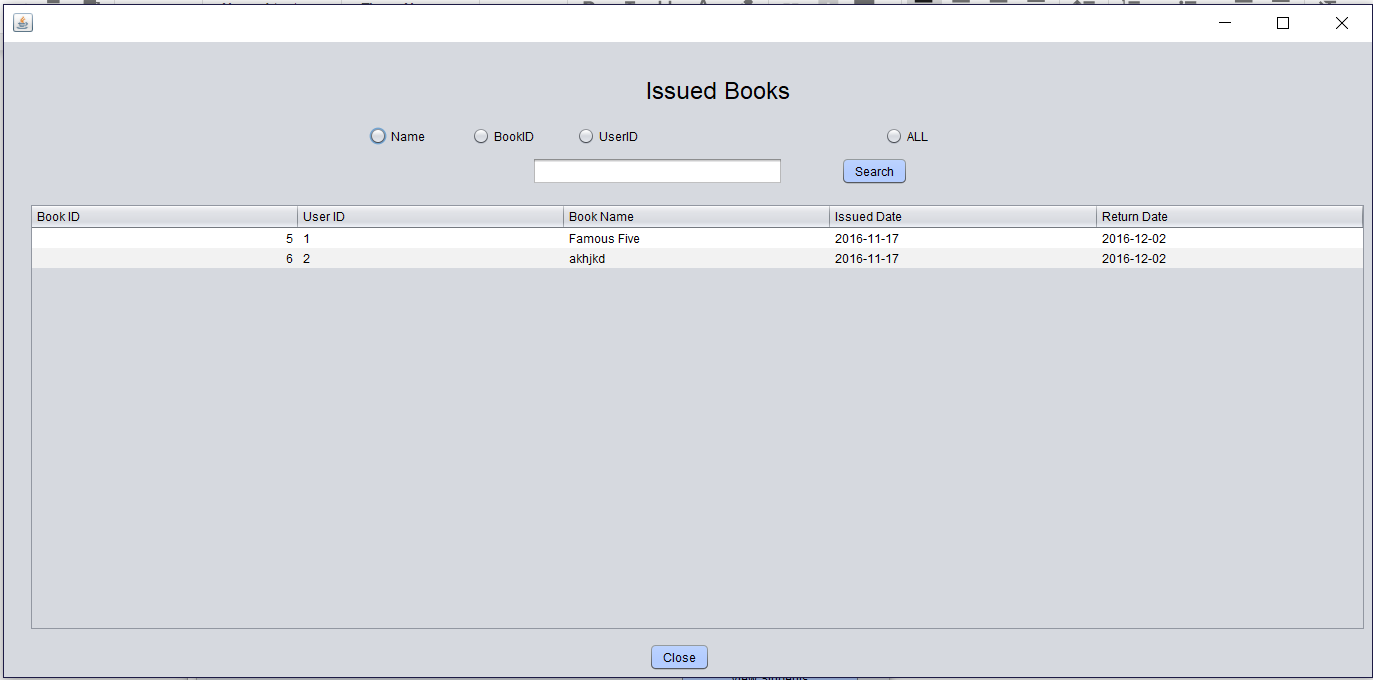
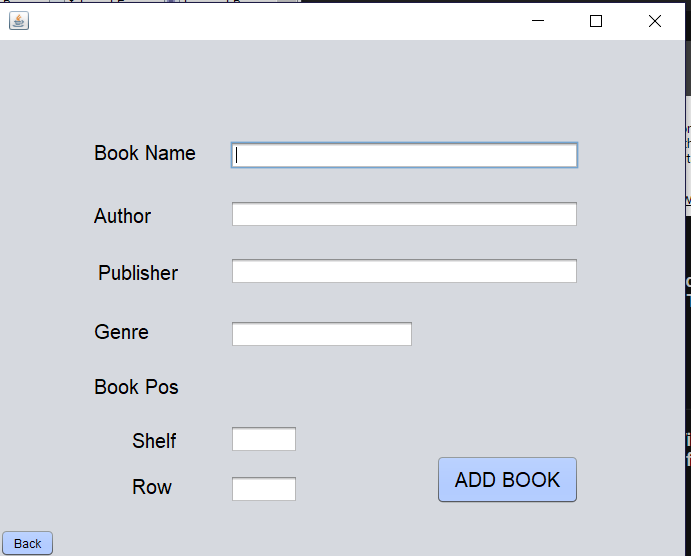
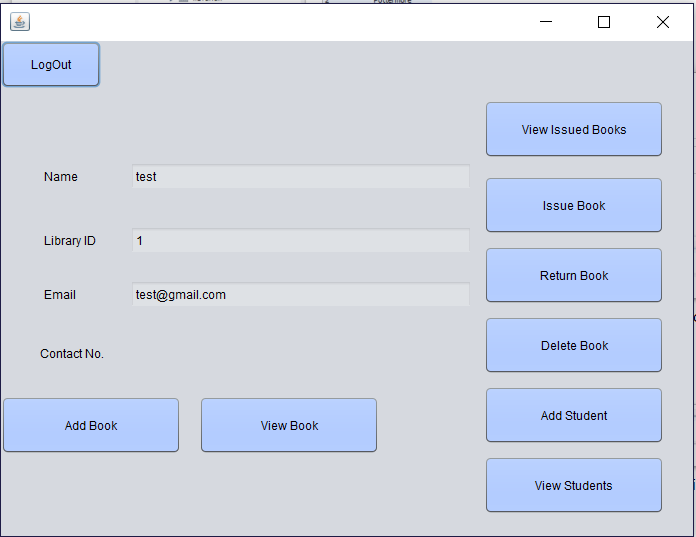
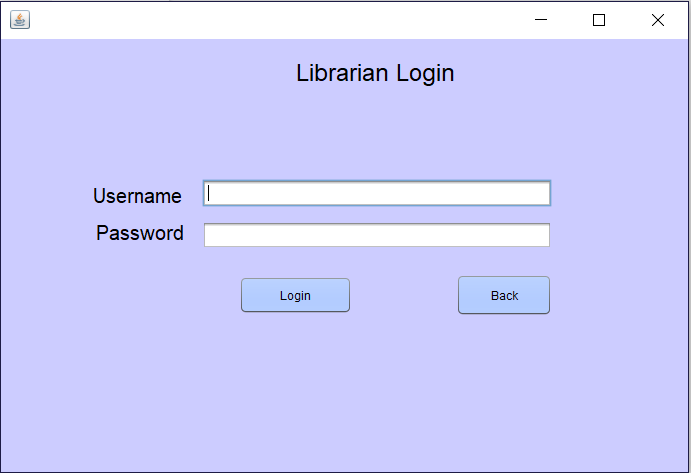
Data is organized into rows, columns and tables, and it is indexed to make it easier to find relevant information. Data gets updated, expanded and deleted as new information is added. Databases process workloads to create and update themselves, querying the data they contain and running applications against it.

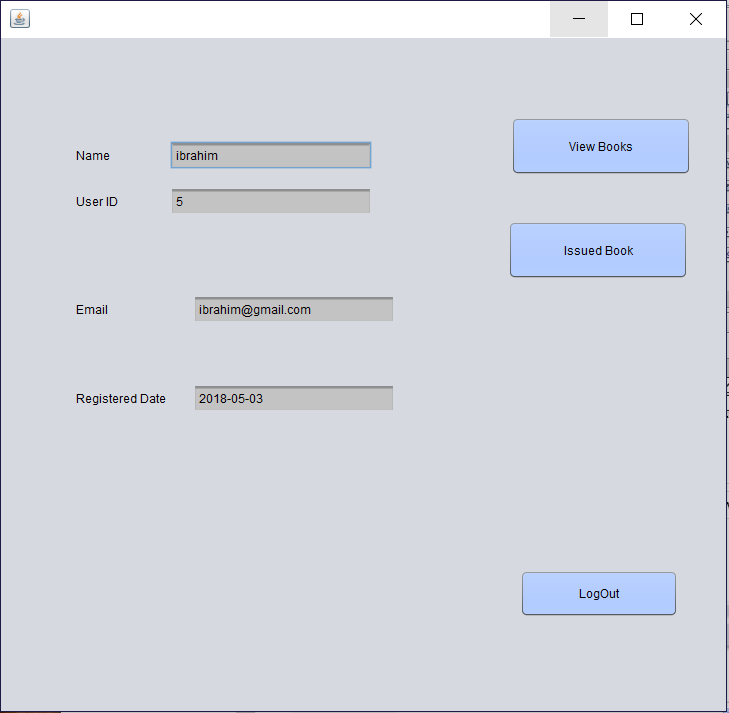
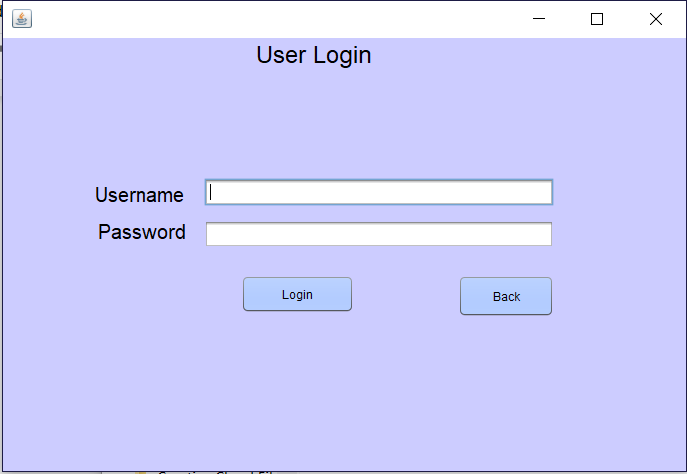
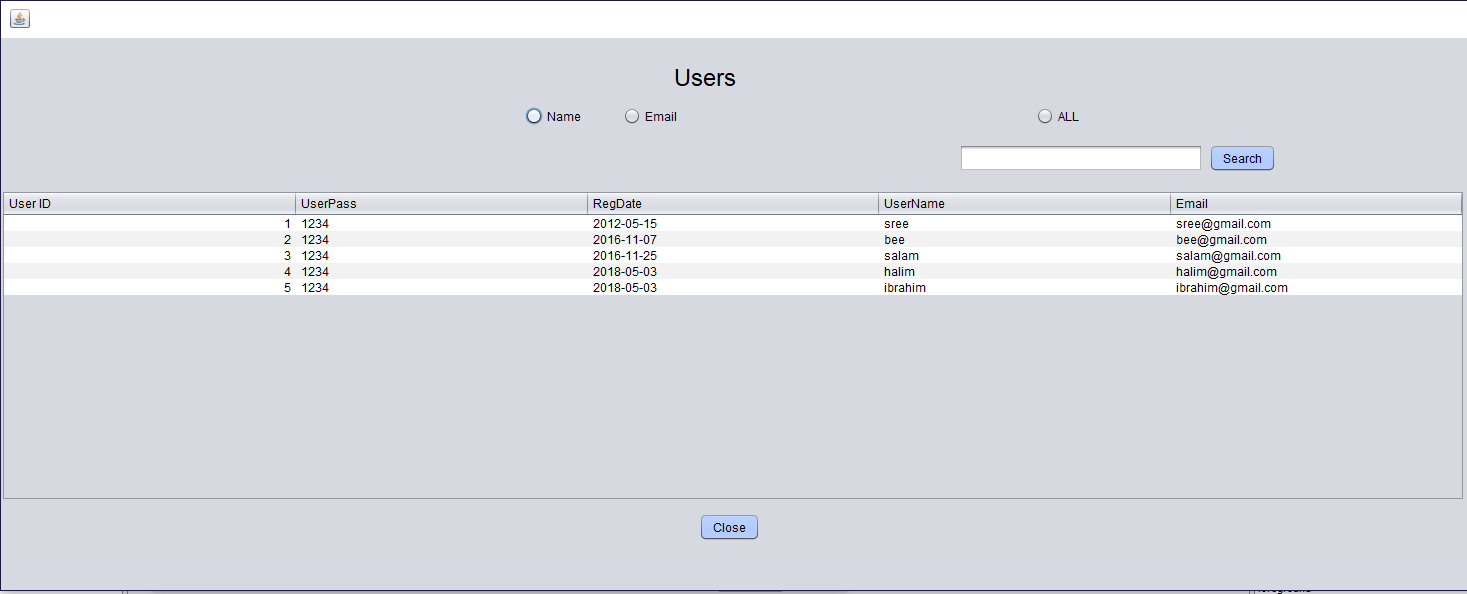
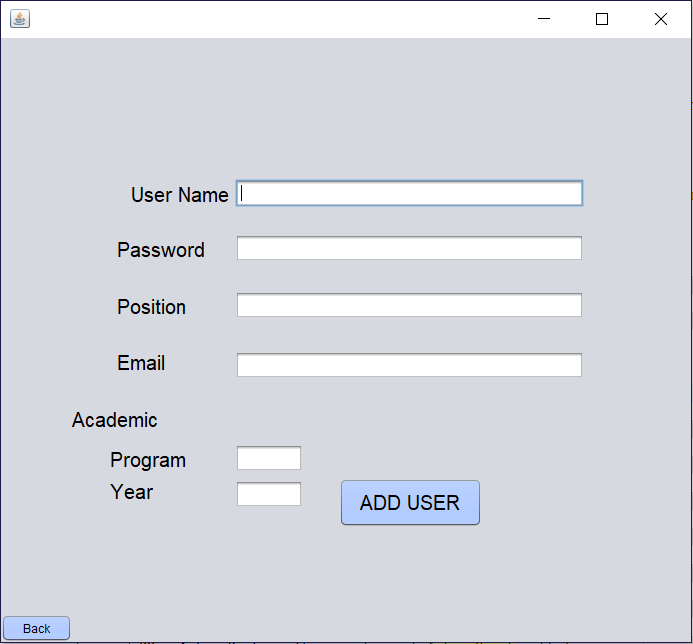
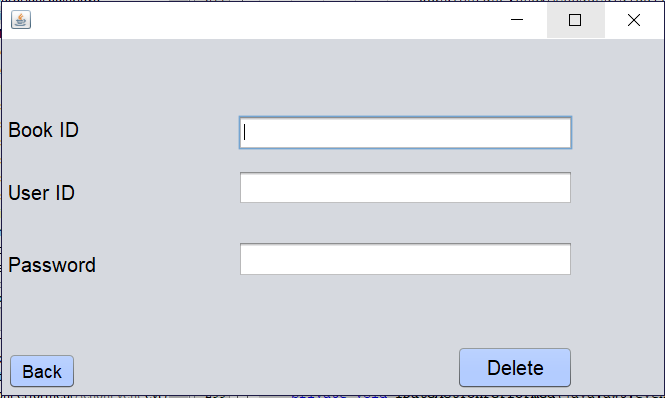
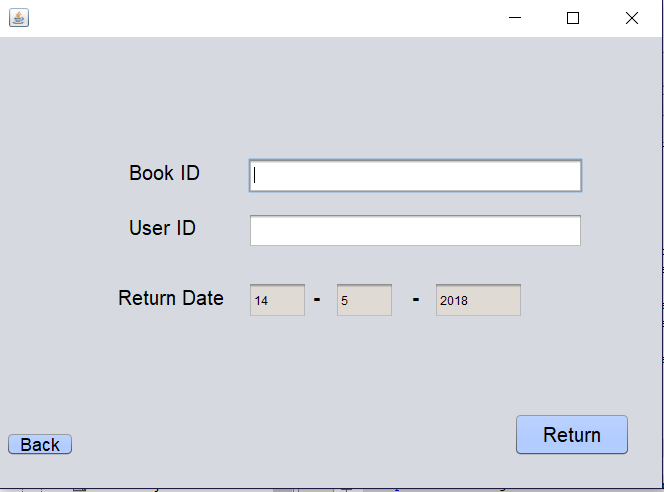
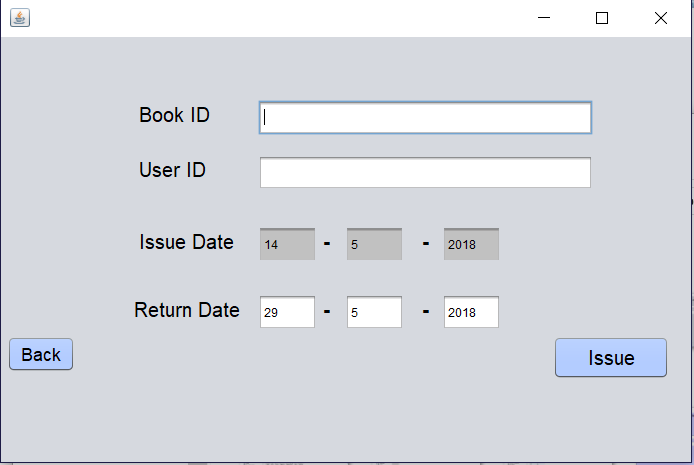
9. In your report appendix, you are required to submit the screenshot of the user  
interfaces. (5 marks)

**Appendix**

**Database library management system showing user login and UI.**







**Reference**

Kumar, A. (March). Online Library Management System. Retrieved from <https://phpgurukul.com/online-library-management-system>

J. (n.d.). Jatanpatel92/Online-Library-Administration-System. Retrieved from <https://github.com/jatanpatel92/Online-Library-Administration-System>