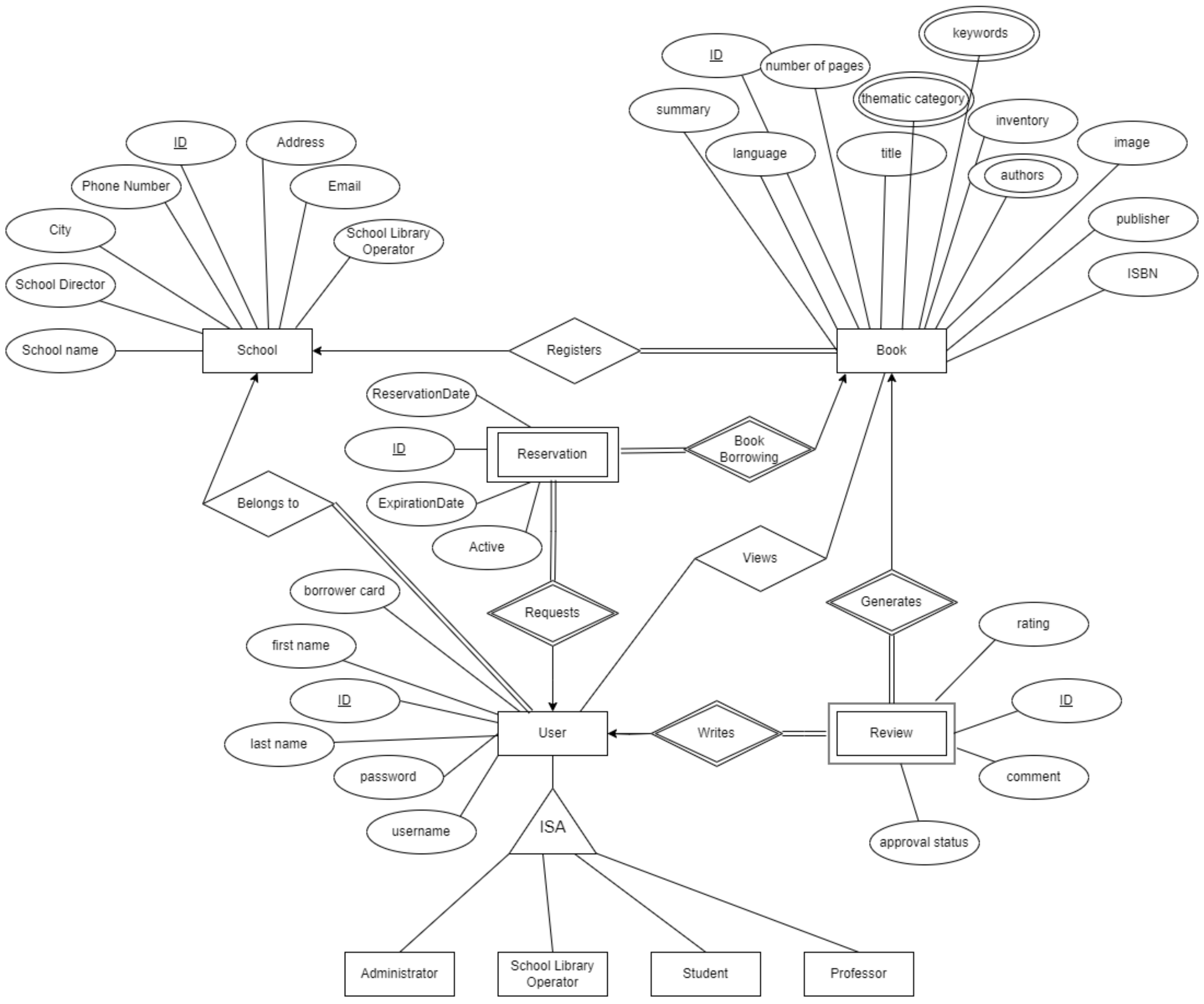
ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ

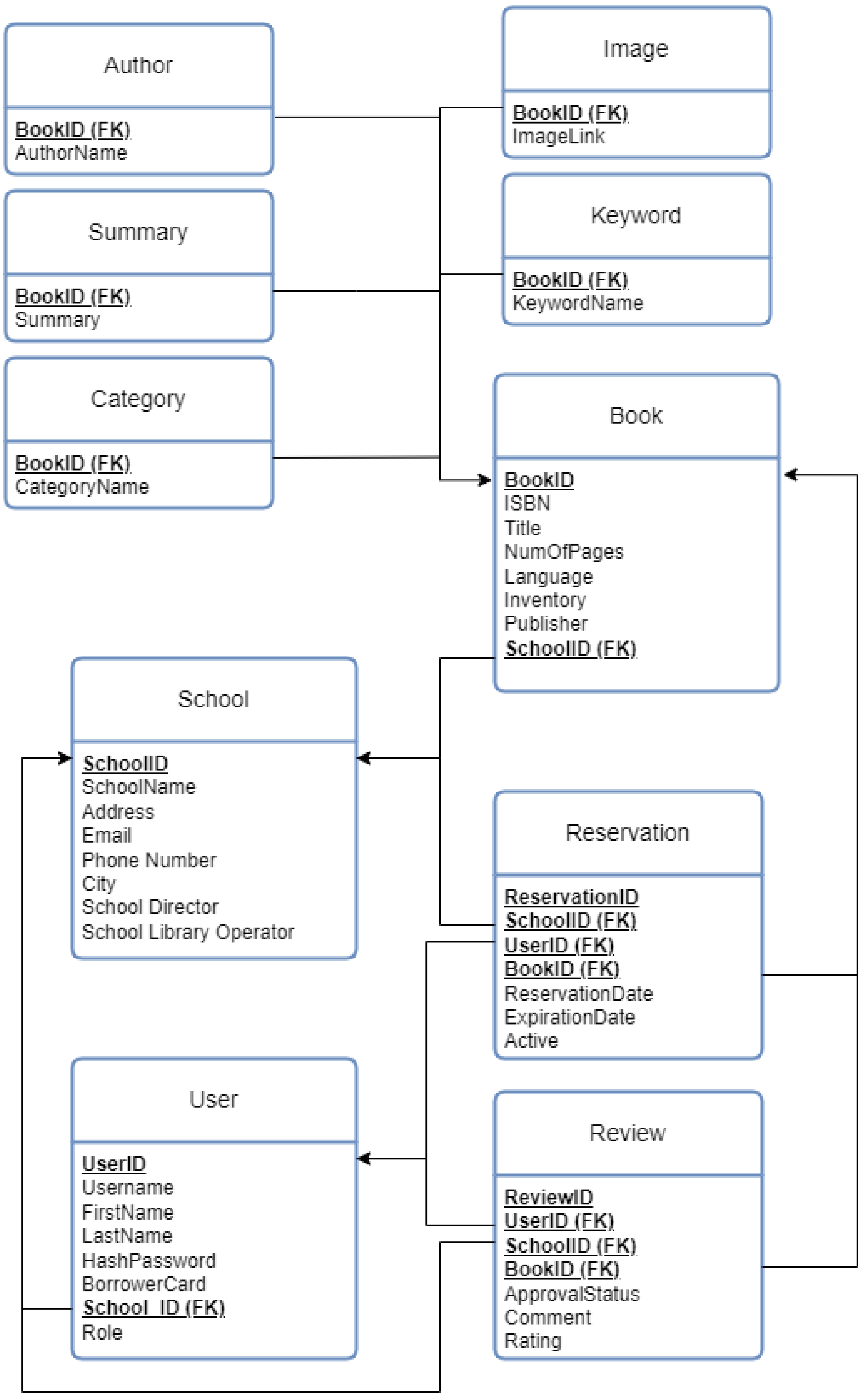
ΣΧΟΛΗ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ
ΕΡΓΑΣΤΗΡΙΟ ΣΥΣΤΗΜΑΤΩΝ ΒΑΣΕΩΝ ΓΝΩΣΕΩΝ ΚΑΙ ΔΕΔΟΜΕΝΩΝ
Ακ. έτος 2022-2023, 6ο εξάμηνο, ΣΗΜΜΥ

Βάσεις Δεδομένων

Αναφορά Εξαμηνιαίας Εργασίας

Ομάδα Project 50
Άγγελος Λουκάς - Α.Μ.: 03119877
Ελευθερία Καλογιάννη - Α.Μ.: 03119829




Author
BookID (FK)
AuthorName
Summary
BookID (FK)
Summary
Category
BookID (FK)
CategoryName
Image
BookID (FK)
ImageLink
Keyword
BookID (FK)
KeywordName
Book
BookID
ISBN
Title
NumOfPages
Language
Inventory
Publisher
SchoolID (FK)
School
SchoolID
SchoolName
Address
Email
Phone Number
City
School Director
School Library Operator
Reservation
ReservationID
SchoolID (FK)
UserID (FK)
BookID (FK)
ReservationDate
ExpirationDate
Active
User
UserID
Username
FirstName
LastName
HashPassword
BorrowerCard
School_ID (FK)
Role
Review
ReviewID
UserID (FK)
SchoolID (FK)
BookID (FK)
ApprovalStatus
Comment
Rating

DDL

```
CREATE TABLE School (
   SchoolID INT UNSIGNED PRIMARY KEY AUTO_INCREMENT ,
   LastUpdate TIMESTAMP NOT NULL DEFAULT CURRENT_TIMESTAMP ON UPDATE
CURRENT_TIMESTAMP,
   SchoolName VARCHAR(50),
   `Address` VARCHAR(50),
   `City` VARCHAR(50),
   PhoneNumber VARCHAR(20),
   Email VARCHAR(50),
   SchoolLibraryOperatorFullName VARCHAR(50),
   SchoolDirectorFullName VARCHAR(50)
)ENGINE=InnoDB DEFAULT CHARSET=utf8;

CREATE TABLE Book (
   BookID INT UNSIGNED PRIMARY KEY AUTO_INCREMENT,
   LastUpdate TIMESTAMP NOT NULL DEFAULT CURRENT_TIMESTAMP ON UPDATE
CURRENT_TIMESTAMP,
   SchoolID INT UNSIGNED NOT NULL,
   Title VARCHAR(255),
   Publisher VARCHAR(255),
   ISBN VARCHAR(13) NOT NULL,
   NumOfPages INT,
   Inventory BOOLEAN,
   Language VARCHAR(50),
   CONSTRAINT `fk_book_school` FOREIGN KEY (SchoolID) REFERENCES School (SchoolID)
)ENGINE=InnoDB DEFAULT CHARSET=utf8;

CREATE TABLE Author (
      BookID INT UNSIGNED,
   last_update TIMESTAMP NOT NULL DEFAULT CURRENT_TIMESTAMP ON UPDATE
CURRENT_TIMESTAMP,
   AuthorName VARCHAR(120),
   CONSTRAINT `fk_author_book` FOREIGN KEY (BookID) REFERENCES Book (BookID)
)ENGINE=InnoDB DEFAULT CHARSET=utf8;

CREATE TABLE Category (
   BookID INT UNSIGNED,
   LastUpdate TIMESTAMP NOT NULL DEFAULT CURRENT_TIMESTAMP ON UPDATE
CURRENT_TIMESTAMP,
   CategoryName VARCHAR(255),
   CONSTRAINT `fk_category_book` FOREIGN KEY (BookID) REFERENCES Book (BookID)
)ENGINE=InnoDB DEFAULT CHARSET=utf8;
```

```
CREATE TABLE Image (
   BookID INT UNSIGNED PRIMARY KEY AUTO_INCREMENT,
   LastUpdate TIMESTAMP NOT NULL DEFAULT CURRENT_TIMESTAMP ON UPDATE
CURRENT_TIMESTAMP,
   ImageLink VARCHAR(255),
   CONSTRAINT `fk_image_book` FOREIGN KEY (BookID) REFERENCES Book (BookID)
)ENGINE=InnoDB DEFAULT CHARSET=utf8;

CREATE TABLE Keyword (
   BookID INT UNSIGNED,
   LastUpdate TIMESTAMP NOT NULL DEFAULT CURRENT_TIMESTAMP ON UPDATE
CURRENT_TIMESTAMP,
   KeywordName VARCHAR(255),
   CONSTRAINT `fk_keyword_book` FOREIGN KEY (BookID) REFERENCES Book (BookID)
)ENGINE=InnoDB DEFAULT CHARSET=utf8;

CREATE TABLE USER(
   UserID INT UNSIGNED PRIMARY KEY AUTO_INCREMENT ,
   SchoolID INT UNSIGNED,
   Username VARCHAR(50),
   `Role` VARCHAR(20),
   FirstName VARCHAR(30),
   LastName VARCHAR(30),
   BorrowerCard VARCHAR(13),
   HashedPassword VARCHAR(100),
   CONSTRAINT `fk_user_school` FOREIGN KEY (SchoolID) REFERENCES School (SchoolID)
)ENGINE=InnoDB DEFAULT CHARSET=utf8;

CREATE TABLE Reservation (
   ReservationID INT UNSIGNED PRIMARY KEY AUTO_INCREMENT ,
   LastUpdate TIMESTAMP NOT NULL DEFAULT CURRENT_TIMESTAMP ON UPDATE
CURRENT_TIMESTAMP,
   SchoolID INT UNSIGNED NOT NULL,
   UserID INT UNSIGNED NOT NULL,
   BookID INT UNSIGNED NOT NULL,
   ReservationDate Date,
   ExpirationDate Date,
   Active VARCHAR(13),
   CONSTRAINT `fk_reservation_school` FOREIGN KEY (SchoolID) REFERENCES School (SchoolID),
   CONSTRAINT `fk_reservation_user` FOREIGN KEY (UserID) REFERENCES User (UserID),
   CONSTRAINT `fk_reservation_book` FOREIGN KEY (BookID) REFERENCES Book (BookID)
)ENGINE=InnoDB DEFAULT CHARSET=utf8;
```

```
CREATE TABLE Review (
   ReviewID INT UNSIGNED PRIMARY KEY AUTO_INCREMENT ,
   LastUpdate TIMESTAMP NOT NULL DEFAULT CURRENT_TIMESTAMP ON UPDATE
CURRENT_TIMESTAMP,
   SchoolID INT UNSIGNED NOT NULL,
   UserID INT UNSIGNED NOT NULL,
   BookID INT UNSIGNED NOT NULL,
   Rating INT UNSIGNED,
   Comment VARCHAR(255),
   ApprovalStatus VARCHAR(20),
   CONSTRAINT `fk_review_school` FOREIGN KEY (SchoolID) REFERENCES School (SchoolID),
   CONSTRAINT `fk_review_user` FOREIGN KEY (UserID) REFERENCES User (UserID),
   CONSTRAINT `fk_review_book` FOREIGN KEY (BookID) REFERENCES Book (BookID)
)ENGINE=InnoDB DEFAULT CHARSET=utf8;

CREATE TABLE Summary (
   BookID INT UNSIGNED PRIMARY KEY AUTO_INCREMENT,
   Summary VARCHAR(5000),
   CONSTRAINT `fk_summary_book` FOREIGN KEY (BookID) REFERENCES Book (BookID)
)ENGINE=InnoDB DEFAULT CHARSET=utf8;

create index index_author_bookid on Author(BookID);
create index index_category_bookid on Category(BookID);
create index index_keyword_bookid on Keyword(BookID);
create index index_user_schoolid on User(SchoolID);
```

4.1.1.List with the total number of loans per school (Search criteria: year, calendar month, e.g. January).

```
SELECT SchoolID,COUNT(*) AS NumberOfLoans
FROM Reservation
WHERE ReservationDate
      BETWEEN ' __STARTDATE__ ' AND ' __ENDDATE__ '
GROUP BY SchoolID
ORDER BY SchoolID;
```

Dummy Data
2023-05-30
2023-06-04

4.1.2.For a given book category (user-selected), which authors belong to it and which teachers have borrowed books from that category in the last year?

```
--Which authors belong to it

SELECT MIN(a.AuthorName)
FROM Book b
JOIN Category c ON b.BookID = c.BookID
JOIN Author a ON b.BookID = a.BookID
WHERE c.CategoryName=' __CATEGORY__ '
GROUP BY a.AuthorName
ORDER BY a.AuthorName;
```

Dummy Data
History
Religion

```
--Which teachers have borrowed

SELECT u.UserID,u.FirstName,u.LastName
FROM User u
JOIN Reservation r ON u.UserID=r.UserID
JOIN Category c ON r.BookID=c.BookID
WHERE r.ReservationDate
      BETWEEN 'LASTYEARDATE'AND 'NOWDATE'
      AND u.Role='Professor'
      AND c.CategoryName='__CATEGORY__'
      AND r.Active != 'Declined'
      AND r.Active != 'Pending'
GROUP BY r.ReservationID;
```

4.1.3.Find young teachers (age < 40 years) who have borrowed the most books and the number of books.

```
SELECT u.UserID, u.FirstName, u.LastName, COUNT(u.UserID) AS Borrowed_books
FROM User u
JOIN Reservation r ON u.UserID = r.UserID
WHERE u.Role = 'Professor'
    AND r.Active != 'Declined'
    AND r.Active != 'Pending'
GROUP BY u.UserID, u.FirstName, u.LastName
ORDER BY Borrowed_books DESC;
```

4.1.4.Find authors whose books have not been borrowed.

```
SELECT a1.AuthorName
FROM (SELECT a1.AuthorName
    FROM Author a1
    GROUP BY a1.AuthorName) a1
WHERE a1.AuthorName NOT IN (
    SELECT a.AuthorName
    FROM Author a
    JOIN Book b ON a.BookID = b.BookID
    JOIN Reservation r ON b.BookID = r.BookID
    GROUP BY a.AuthorName
);
```

4.1.5.Which operators have loaned the same number of books in a year with more than 20 loans?

```
SELECT t.ReservationPerSchoolCount, GROUP_CONCAT(t.SchoolLibraryOperatorFullName) AS
SchoolsWithSameCount
FROM (
  SELECT s.SchoolLibraryOperatorFullName, s.SchoolID, COUNT(*) AS ReservationPerSchoolCount
  FROM Reservation r
  JOIN School s ON s.SchoolID = r.SchoolID
  WHERE r.Active != 'Declined'
    AND r.Active != 'Pending'
  GROUP BY r.SchoolID
  HAVING ReservationPerSchoolCount > 20
) t
GROUP BY t.ReservationPerSchoolCount
HAVING COUNT(*) > 1;
```

4.1.6.Many books cover more than one category. Among field pairs (e.g., history and poetry) that are common in books, find the top-3 pairs that appeared in borrowings.

```
SELECT c1.CategoryName, c2.CategoryName, COUNT(*) AS BorrowingCount
FROM Book b
JOIN Category c1 ON b.BookID = c1.BookID
JOIN Category c2 ON b.BookID = c2.BookID AND c1.CategoryName < c2.CategoryName
JOIN Reservation r ON b.BookID = r.BookID
WHERE r.Active != 'Declined'
   AND r.Active != 'Pending'
GROUP BY c1.CategoryName, c2.CategoryName
HAVING c1.CategoryName != c2.CategoryName
ORDER BY BorrowingCount DESC
LIMIT 3;
```

4.1.7.Find all authors who have written at least 5 books less than the author with the most books

```
SELECT a.AuthorName, COUNT(*) AS BookCount
FROM Author a
JOIN Book b ON a.BookID = b.BookID
GROUP BY a.AuthorName
HAVING BookCount <= (SELECT COUNT(*) AS BookCount2
      FROM Author
      JOIN Book ON Author.BookID = Book.BookID
      GROUP BY AuthorName
      ORDER BY BookCount2 DESC
      LIMIT1 )-5
ORDER BY BookCount DESC;
```

4.2.1. All books by Title, Author (Search criteria: title/ category/ author/ copies).

```
SELECT Book.Title,Count(*) AS BookCount
FROM Book
JOIN Author ON Book.BookID = Author.BookID
JOIN Category ON Book.BookID = Category.BookID
WHERE Book.SchoolID=' __OPERATORSCHOOLID__ '
   AND Book.Title = ' __TITLE__ '
   AND Author.AuthorName = ' __AUTHORNAME__ '
   AND Category.CategoryName = ' __CATEGORY__ '
GROUP BY Book.ISBN
ORDER BY Book.Title
```

Dummy Data
Julius Loukas

4.2.2.Find all borrowers who own at least one book and have delayed its return. (Search criteria: First Name, Last Name, Delay Days).

```
SELECT DISTINCT User.FirstName, User.LastName,GROUP_CONCAT( Reservation.ExpirationDate)
FROM User
JOIN Reservation ON User.UserID = Reservation.UserID
JOIN Book ON Reservation.BookID = Book.BookID
WHERE Reservation.ExpirationDate < '2023-06-02'
   AND Reservation.Active != 'Declined'
   AND Reservation.Active != 'Pending'
   AND User.SchoolID='__OPERATORSCHOOLID__'
   AND User.FirstName=' __FIRSTNAME__ '
   AND User.LastName=' __LASTNAME__ '
   AND Reservation.ExpirationDate=' __EXPIRATIONDATE__ '
GROUP BY User.FirstName, User.LastName
ORDER BY User.FirstName, User.LastName;
```

4.2.3.Average Ratings per borrower and category (Search criteria: user/category)

```
--By borrower

SELECT User.UserID,User.FirstName, User.LastName, AVG(Review.Rating) AS AverageRating
FROM User
JOIN Review ON User.UserID = Review.UserID
WHERE User.SchoolID='__OPERATORSCHOOLID__'
   AND Review.ApprovalStatus='Approved'
   AND User.UserID=' __USERID__ '
GROUP BY User.UserID
ORDER BY AverageRating DESC,User.FirstName, User.LastName ;

--By category

SELECT Category.CategoryName, AVG(Review.Rating) AS AverageRating
FROM Review
JOIN Book ON Review.BookID = Book.BookID
JOIN Category ON Category.BookID = Book.BookID
WHERE Review.SchoolID='__OPERATORSCHOOLID__'
   AND Review.ApprovalStatus='Approved'
   AND Category.CategoryName=' __CATEGORY__ '
GROUP BY Category.CategoryName
ORDER BY AverageRating DESC,Category.CategoryName ;
```

4.3.1.List with all books (Search criteria: title/category/author), ability to select a book and create a reservation request.

```
SELECT Book.Title, Book.ISBN, COUNT(*) AS BookCount,
      GROUP_CONCAT(IF(Book.Inventory = True, Book.BookID, NULL)) AS BookIDs
FROM Book
JOIN Author ON Book.BookID = Author.BookID
JOIN Category ON Book.BookID = Category.BookID
WHERE Book.SchoolID = '__SCHOOLID__'
   AND Book.Title = '__TITLE__'
   AND Author.AuthorName = '__AUTHOR__'
   AND Category.CategoryName = '__CATEGORY__'
GROUP BY Book.ISBN
ORDER BY Book.Title;


Insert into Reservation(
  `SchoolID`, `UserID`, `BookID`, `ReservationDate`,`ExpirationDate`, `Active`)
Values
(
   '{schoolid}','{userid}','{bookid}', Null, Null, 'Pending'
 );
```

4.3.2.List of all books borrowed by this user.

```
SELECT User.UserID, Book.Title
FROM Book
JOIN Reservation ON Book.BookID = Reservation.BookID
JOIN User ON Reservation.UserID = User.UserID
WHERE User.UserID='__USERID__'
   AND Reservation.Active != 'Declined'
   AND Reservation.Active != 'Pending'
ORDER BY User.UserID;
```