ΒΑΣΕΙΣ ΔΕΔΟΜΕΝΩΝ

Εξαμηνιαία Εργασία

Ακαδημαϊκό Έτος: 2022-2023

Ομάδα Project 114

Γεώργιος Κουρής ΑΜ: 03120116

Νικόλαος Λοβέρδος ΑΜ: 03120442

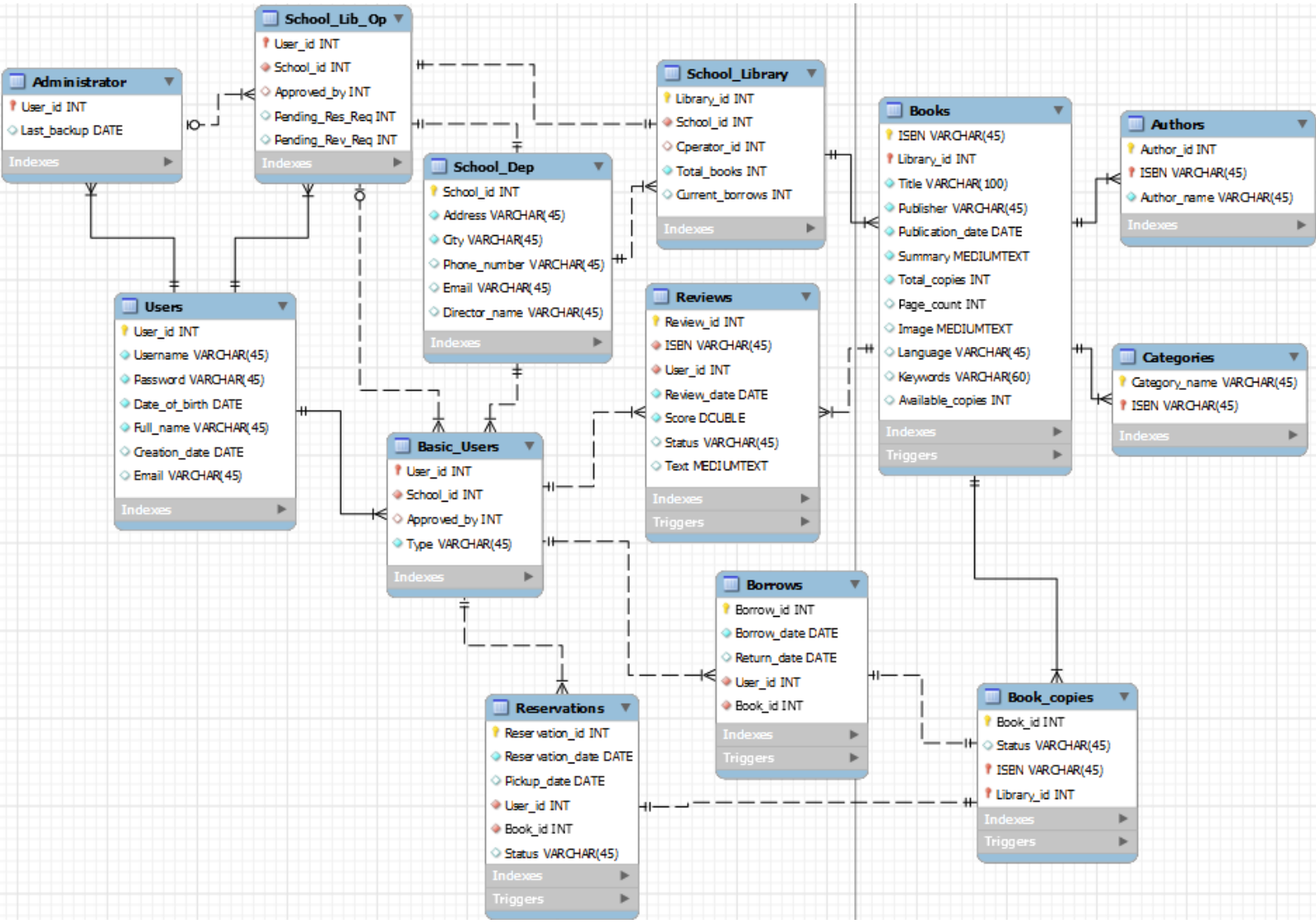
Αθανάσιος Αθανασόπουλος ΑΜ: 03120820

Σύστημα Διαχείρισης και Αποθήκευσης Πληροφοριών Σχολικής Βιβλιοθήκης

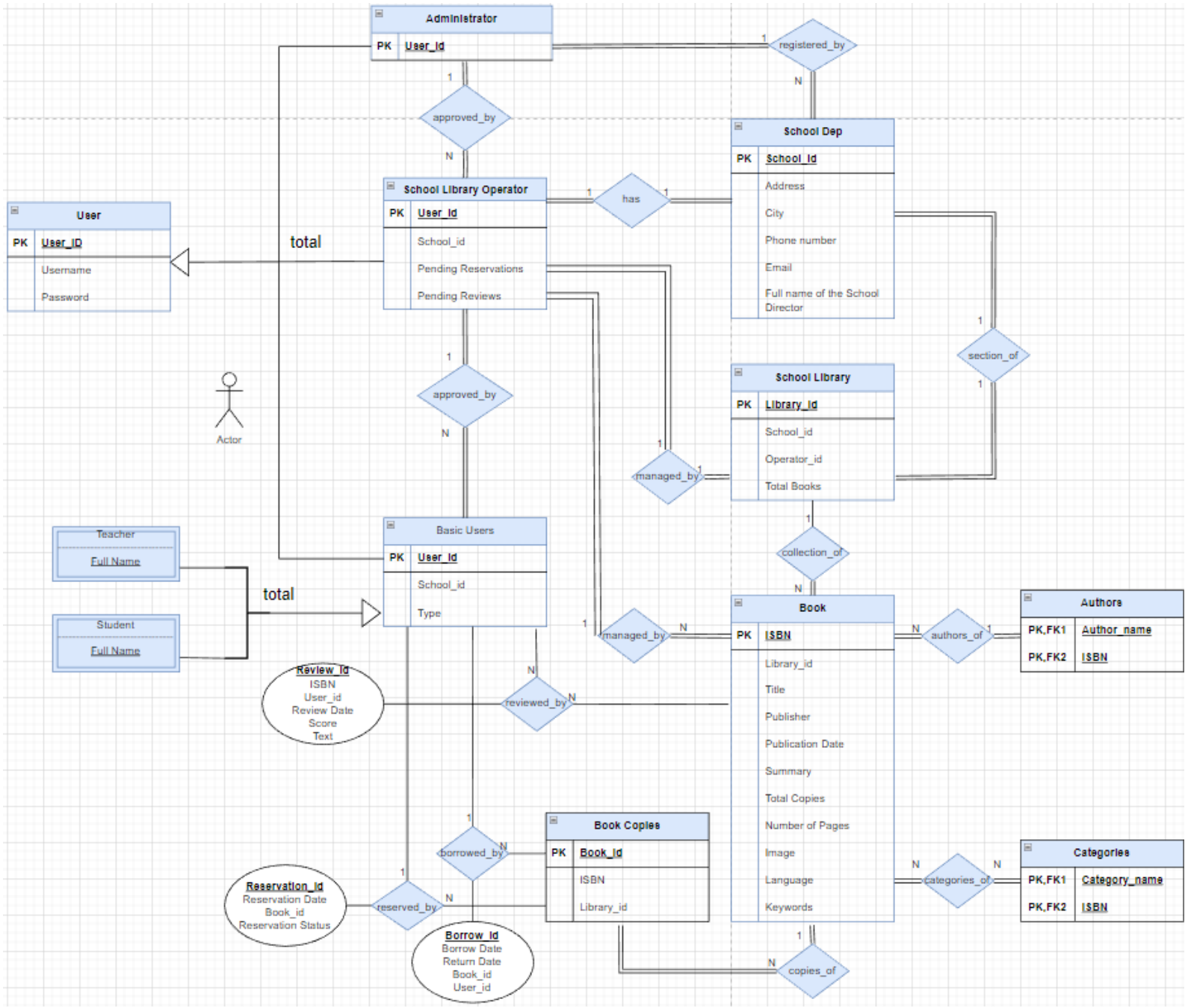
Σκοπός του συστήματος είναι η εύκολη διαχείριση των παρακάτω:

- Καταχώρηση στοιχείων κάθε σχολικής μονάδας (Ονομασία Σχολείου, Ταχυδρομική Δ/νση, Πόλη, Τηλέφωνο, email, Ονοματεπώνυμο Δ/ντη Σχολείου, Ονοματεπώνυμο Υπεύθυνου Χειριστή Σχολικής Βιβλιοθήκης).
- Καταχώρηση των βιβλίων που διαθέτει η κάθε σχολική μονάδα στη βιβλιοθήκη της και επεξεργασία των δεδομένων τους (τίτλος, εκδότης, ISBN, συγγραφείς, αριθμός σελίδων, περίληψη, διαθέσιμα αντίτυπα, εικόνα, θεματική κατηγορία, γλώσσα, λέξεις κλειδιά).
- Εξακρίβωση της ταυτότητας των χρηστών (μέσω username/password) και ανάθεση των ιδιοτήτων τους (με βάση τον ρόλο τους στο σύστημα).
- Δανεισμός βιβλίων.
- Κρατήσεις βιβλίων (πεπερασμένης διάρκειας) σε περίπτωση που κάποιο βιβλίο δεν είναι διαθέσιμο για δανεισμό..
- Αξιολογήσεις βιβλίων με βάση την κλίμακα Likert.

Σχεσιακό Διάγραμμα



ER Διάγραμμα



DDL και DML script

DDL script:

Στο DDL δημιουργούνται όλοι οι πίνακες που έχουμε ορίσει, οι primary key εξαρτήσεις, και τα triggers της βάσης.
Link στο github:

https://github.com/kourisgeorgios/librarydb/blob/main/Database-Library/schema.sql

DML script:

Στο DML πραγματοποιείται η εισαγωγή των αρχικών δεδομένων στη βάση μας.
Link στο github:

https://github.com/kourisgeorgios/librarydb/blob/main/Database-Library/insert_data.sql

Τα queries που ζητούνται έχουν υλοποιηθεί και βρίσκονται στο αρχείο routes.py

3.1.1 DEFAULT

```
SELECT S.School_id School, count(borrow_id) 'No_Borrows' FROM borrows B INNER JOIN Book_copies BC ON B.Book_id = BC.Book_id RIGHT JOIN School_Library S ON S.Library_id = BC.Library_id GROUP BY School_id;
```

3.1.1 YEAR

```
SELECT S.School_id School, count(borrow_id) 'No_Borrows' FROM borrows B INNER JOIN Book_copies BC ON B.Book_id = BC.Book_id RIGHT JOIN School_Library S ON S.Library_id = BC.Library_id WHERE year(borrow_date) = {year} GROUP BY School_id;
```

3.1.1 MONTH

```
SELECT S.School_id School, count(borrow_id) 'No_Borrows' FROM borrows B INNER JOIN Book_copies BC ON B.Book_id = BC.Book_id RIGHT JOIN School_Library S ON S.Library_id = BC.Library_id WHERE month(borrow_date) = {month} GROUP BY School_id ORDER BY School_id, year(borrow_date), month(borrow_date);
```

3.1.1 YEAR AND MONTH

```
SELECT S.School_id School, count(borrow_id) 'No_Borrows' FROM borrows B INNER JOIN Book_copies BC ON B.Book_id = BC.Book_id RIGHT JOIN School_Library S ON S.Library_id = BC.Library_id WHERE month(borrow_date) = {month} AND year(borrow_date) = {year} GROUP BY School_id ORDER BY School_id, year(borrow_date), month(borrow_date);
```

3.1.2a

```
SELECT A.Author_name
FROM Books B
INNER JOIN (SELECT ISBN,Category_name FROM Categories WHERE Category_name = 'Young Adult') as C ON B.isbn = C.isbn
INNER JOIN Authors A ON A.ISBN = B.ISBN
```

3.1.2b

```
SELECT U.User_id, U.Full_name
FROM Book_copies BC
INNER JOIN (SELECT ISBN,Category_name FROM Categories WHERE Category_name = 'Self-Help') as C ON BC.isbn = C.isbn
INNER JOIN Borrows B ON B.Book_id = BC.Book_id
INNER JOIN Users U ON U.User_id = B.User_id
```

3.1.3

```
SELECT U.User_id,U.Full_name, COUNT(borrow_id) No_Borrows FROM borrows B INNER JOIN Users U ON U.User_id = B.User_id INNER JOIN Basic_Users BU ON BU.User_id =
```

U.User_id WHERE (YEAR(CURRENT_DATE)-YEAR(U.Date_of_birth)<40) AND Type = 'Teacher' GROUP BY User_id ORDER BY No_Borrows DESC LIMIT 3;

3.1.4

```
SELECT A.Author_name
FROM Book_copies BC
INNER JOIN Authors A ON A.ISBN = BC.ISBN
LEFT JOIN Borrows Bo ON Bo.Book_id = BC.Book_id
GROUP BY Author_name
HAVING Count(borrow_id) = 0;
```

3.1.5

SELECT DISTINCT T.Operator_id Operator_1,T2.Operator_id Operator_2, T.C FROM (SELECT L.Operator_id, COUNT(Bo.borrow_id) AS C FROM Book_copies BC INNER JOIN (SELECT Borrow_id,Book_id FROM Borrows WHERE YEAR(CURRENT_DATE)-YEAR(Borrow_date)) Bo ON Bo.Book_id = BC.Book_id INNER JOIN School_Library L ON L.Library_id = BC.Library_id GROUP BY L.Operator_id HAVING C >= 20) AS T INNER JOIN (SELECT L.Operator_id,COUNT(Bo.borrow_id) AS C FROM Book_copies BC INNER JOIN (SELECT Borrow_id,Book_id FROM Borrows WHERE YEAR(CURRENT_DATE)-YEAR(Borrow_date)) Bo ON Bo.Book_id = BC.Book_id INNER JOIN School_Library L ON L.Library_id = BC.Library_id GROUP BY L.Operator_id HAVING C >= 20) AS T2 ON T.C = T2.C WHERE T.Operator_id != T2.Operator_id ORDER BY C;

3.1.6

SELECT Cat1, Cat2, COUNT(Borrow_id) No_borrows FROM (SELECT C1.Category_name Cat1, C2.Category_name Cat2, C1.ISBN FROM Categories C1 INNER JOIN Categories C2 ON C1.ISBN = C2.ISBN WHERE C1.Category_name != C2.Category_name) AS T INNER JOIN Book_copies BC ON BC.ISBN = T.ISBN INNER JOIN Borrows Bo ON Bo.Book_id = BC.Book_id GROUP BY Cat1, Cat2 ORDER BY No_borrows DESC LIMIT 3;

3.1.7a

SELECT Author_name, Count(ISBN) C FROM Authors GROUP BY Author_name HAVING C <= (SELECT Count(ISBN) C FROM Authors A GROUP BY Author_name ORDER BY C DESC LIMIT 1) - 5;

3.1.7b (συγγραφέας με τα περισσότερα βιβλία)

SELECT Author_name, Count(ISBN) C FROM Authors A GROUP BY Author_name ORDER BY C DESC LIMIT 1;

3.2.1 DEFAULT

SELECT BC.ISBN, B.Title, B.Publication_date, GROUP_CONCAT(DISTINCT Author_name) AS 'Authors', GROUP_CONCAT(DISTINCT Category_name) AS 'Categories', (SELECT CAST(AVG(Score) AS DECIMAL(3,2)) FROM Reviews WHERE ISBN = BC.ISBN) AS Score, B.Total_copies, GROUP_CONCAT(DISTINCT Book_id) AS 'IDs', BC.Library_id AS Lib FROM Book_copies BC INNER JOIN Books B ON B.ISBN = BC.ISBN INNER JOIN Categories C ON C.ISBN = BC.ISBN INNER JOIN Authors A ON A.ISBN = C.ISBN GROUP BY ISBN, Lib;

3.2.1 SEARCH BY TITLE

```
SELECT BC.ISBN, B.Title, B.Publication_date, GROUP_CONCAT(DISTINCT Author_name) AS 'Authors', GROUP_CONCAT(DISTINCT Category_name) AS 'Categories', (SELECT CAST(AVG(Score) AS DECIMAL(3,2)) FROM Reviews WHERE ISBN = BC.ISBN) AS Score, B.Total_copies, GROUP_CONCAT(DISTINCT Book_id) AS 'IDs', BC.Library_id AS Lib FROM Book_copies BC INNER JOIN Books B ON B.ISBN = BC.ISBN INNER JOIN Categories C ON C.ISBN = BC.ISBN INNER JOIN Authors A ON A.ISBN = C.ISBN WHERE B.Title = 'Animal, The' GROUP BY ISBN, Lib;
```

3.2.1 SEARCH BY CATEGORY

```
SELECT BC.ISBN, B.Title, B.Publication_date, GROUP_CONCAT(DISTINCT Author_name) AS 'Authors', CASE WHEN Cat2 IS NULL THEN Cat1 ELSE CONCAT(Cat1, ', ', Cat2) END AS 'Categories', (SELECT CAST(AVG(Score) AS DECIMAL(3,2)) FROM Reviews WHERE ISBN = BC.ISBN) AS Score, B.Total_copies, GROUP_CONCAT(DISTINCT Book_id) AS 'IDs', BC.Library_id AS 'Library' FROM Book_copies BC INNER JOIN Books B ON B.ISBN = BC.ISBN INNER JOIN (SELECT ISBN, Category_name AS Cat1 FROM Categories WHERE Category_name = 'Travel') C1 ON C1.ISBN = BC.ISBN LEFT JOIN (SELECT ISBN, Category_name AS Cat2 FROM Categories WHERE Category_name != 'Travel') C2 ON C2.ISBN = BC.ISBN INNER JOIN Authors A ON A.ISBN = BC.ISBN GROUP BY ISBN, Library;
```

3.2.1 SEARCH BY AUTHOR

```
SELECT BC.ISBN, B.Title, B.Publication_date, GROUP_CONCAT(DISTINCT Category_name) AS 'Categories', CASE WHEN Au2 IS NULL THEN Au1 ELSE CONCAT(Au1, ', ', Au2) END AS 'Authors', (SELECT CAST(AVG(Score) AS DECIMAL(3,2)) FROM Reviews WHERE ISBN = BC.ISBN) AS Score, B.Total_copies, GROUP_CONCAT(DISTINCT Book_id) AS 'IDs', BC.Library_id AS 'Library' FROM Book_copies BC INNER JOIN Books B ON B.ISBN = BC.ISBN INNER JOIN (SELECT ISBN, Author_name AS Au1 FROM Authors WHERE Author_name = 'Liam Ward') C1 ON C1.ISBN = BC.ISBN LEFT JOIN (SELECT ISBN, Author_name AS Au2 FROM Authors WHERE Author_name != 'Liam Ward') C2 ON C2.ISBN = BC.ISBN INNER JOIN Categories A ON A.ISBN = BC.ISBN GROUP BY ISBN, Library;
```

3.2.1 SEARCH BY TOTAL COPIES

```
SELECT BC.ISBN, B.Title, B.Publication_date, GROUP_CONCAT(DISTINCT Author_name) AS 'Authors', GROUP_CONCAT(DISTINCT Category_name) AS 'Categories', (SELECT CAST(AVG(Score) AS DECIMAL(3,2)) FROM Reviews WHERE ISBN = BC.ISBN) AS Score, B.Total_copies, GROUP_CONCAT(DISTINCT Book_id) AS 'IDs', BC.Library_id AS Lib FROM Book_copies BC INNER JOIN Books B ON B.ISBN = BC.ISBN INNER JOIN Categories C ON C.ISBN = BC.ISBN INNER JOIN Authors A ON A.ISBN = C.ISBN WHERE Total_copies = 5 GROUP BY ISBN, Lib;
```

3.2.2 DEFAULT

```
SELECT B.User_id, U.Full_name, B.Borrow_id, B.Book_id FROM Borrows B INNER JOIN Users U ON U.User_id = B.User_id WHERE Datediff(CURRENT_DATE,BORROW_DATE)>7 AND Return_date IS NULL ;
```

3.2.2 SEARCH BY FULL NAME

```
SELECT B.User_id, U.Full_name, B.Borrow_id, B.Book_id FROM Borrows B INNER JOIN Users U ON U.User_id = B.User_id WHERE Datediff(CURRENT_DATE,BORROW_DATE)>7 AND U.Full_name = 'Monica Stilliard'AND Return_date  IS  NULL;
```

3.2.2 SEARCH BY DAYS LATE

```
SELECT B.User_id, U.Full_name, B.Borrow_id, B.Book_id FROM Borrows B INNER JOIN Users U ON U.User_id = B.User_id WHERE Datediff(CURRENT_DATE,BORROW_DATE)>7 + CHANGE-->2 AND Return_date IS NULL;
```

3.2.3 DEFAULT USER/SCORE

```
SELECT U.Username,R.User_id,CAST(AVG(Score) AS DECIMAL(3,2)) AS Average_Score,Count(Score) No_reviews FROM Reviews R INNER JOIN Users U ON U.User_id = R.User_id GROUP BY User_id
```

3.2.3 DEFAULT CATEGORY/SCORE

```
SELECT Category_name ,CAST(AVG(R.Score) AS DECIMAL(3,2)) AS Average_Score, Count(Score) No_reviews FROM Reviews AS R INNER JOIN Categories C ON C.ISBN = R.ISBN Group BY Category_name
```

3.2.3 SCORE/CATEGORY

```
SELECT Category_name ,CAST(AVG(R.Score) AS DECIMAL(3,2)) AS Average_Score, Count(Score) No_reviews FROM Reviews AS R INNER JOIN Categories C ON C.ISBN = R.ISBN WHERE Category_name = 'Young Adult'
```

3.2.3 SCORE/USER_ID

```
SELECT U.Username,R.User_id,CAST(AVG(Score) AS DECIMAL(3,2)) AS Average_Score,Count(Score) No_reviews FROM Reviews R INNER JOIN Users U ON U.User_id = R.User_id WHERE R.User_id = 5
```

3.3.1 DEFAULT

```
SELECT BC.ISBN, B.Title, B.Publication_date, GROUP_CONCAT(DISTINCT Author_name) as 'Authors', GROUP_CONCAT(DISTINCT Category_name) as 'Categories', (SELECT CAST(AVG(Score)AS DECIMAL (3,2))FROM Reviews WHERE ISBN = BC.ISBN) AS Score, GROUP_CONCAT(DISTINCT Book_id) as 'IDs', BC.Library_id Lib FROM Book_copies BC INNER JOIN Books B ON B.ISBN = BC.ISBN INNER JOIN Categories C ON C.ISBN = BC.ISBN INNER JOIN Authors A ON A.ISBN = C.ISBN Group by ISBN,Lib;
```

3.3.1 SEARCH BY TITLE

SELECT BC.ISBN, B.Title, B.Publication_date, GROUP_CONCAT(DISTINCT Author_name) as 'Authors', GROUP_CONCAT(DISTINCT Category_name) as 'Categories', (SELECT CAST(AVG(Score)AS DECIMAL (3,2))FROM Reviews WHERE ISBN = BC.ISBN) AS Score, GROUP_CONCAT(DISTINCT Book_id) as 'IDs', BC.Library_id Lib FROM Book_copies BC INNER JOIN Books B ON B.ISBN = BC.ISBN INNER JOIN Categories C ON C.ISBN = BC.ISBN INNER JOIN Authors A ON A.ISBN = C.ISBN WHERE B.Title = '{data}' Group by ISBN,Lib;

3.3.1 SEARCH BY CATEGORY

SELECT BC.ISBN, B.Title, B.Publication_date, GROUP_CONCAT(DISTINCT Author_name) as 'Authors', CASE WHEN Cat2 IS NULL THEN Cat1 ELSE CONCAT(CAT1,', ',CAT2) END AS 'Categories', (SELECT CAST(AVG(Score)AS DECIMAL (3,2))FROM Reviews WHERE ISBN = BC.ISBN) AS Score, GROUP_CONCAT(DISTINCT Book_id) as 'IDs', BC.Library_id as 'Library' FROM Book_copies BC INNER JOIN Books B ON B.ISBN = BC.ISBN INNER JOIN (SELECT ISBN,Category_name Cat1 FROM Categories WHERE Category_name = '{data}') C1 ON C1.ISBN = BC.ISBN LEFT JOIN (SELECT ISBN,Category_name Cat2 FROM Categories WHERE Category_name != '{data}') C2 ON C2.ISBN = BC.ISBN INNER JOIN Authors A ON A.ISBN = BC.ISBN GROUP BY ISBN,Library;

3.3.1 SEARCH BY AUTHOR

SELECT BC.ISBN, B.Title, B.Publication_date, GROUP_CONCAT(DISTINCT Category_name) as 'Categories', CASE WHEN Au2 IS NULL THEN Au1 ELSE CONCAT(Au1,', ',Au2) END AS 'Authors', (SELECT CAST(AVG(Score)AS DECIMAL (3,2))FROM Reviews WHERE ISBN = BC.ISBN) AS Score, GROUP_CONCAT(DISTINCT Book_id) as 'IDs', BC.Library_id as 'Library' FROM Book_copies BC INNER JOIN Books B ON B.ISBN = BC.ISBN INNER JOIN (SELECT ISBN,Author_name Au1 FROM Authors WHERE Author_name = '{data}') C1 ON C1.ISBN = BC.ISBN LEFT JOIN (SELECT ISBN,Author_name Au2 FROM Authors WHERE Author_name != '{data}') C2 ON C2.ISBN = BC.ISBN INNER JOIN Categories A ON A.ISBN = BC.ISBN GROUP BY ISBN,Library;

3.3.2

SELECT B.* FROM BORROWS B INNER JOIN USERS U ON U.User_id = B.User_id WHERE U.Username = '{Username}';

Ανάπτυξη της Εφαρμογής

Για την υλοποίηση της ΒΔ χρησιμοποιήθηκαν η MySQL (MariaDB) και το MySQL Workbench. Για τον web server χρησιμοποιήθηκε Flask (Python) για το server side και HTML για το client side της εφαρμογής.

Εκδόσεις:

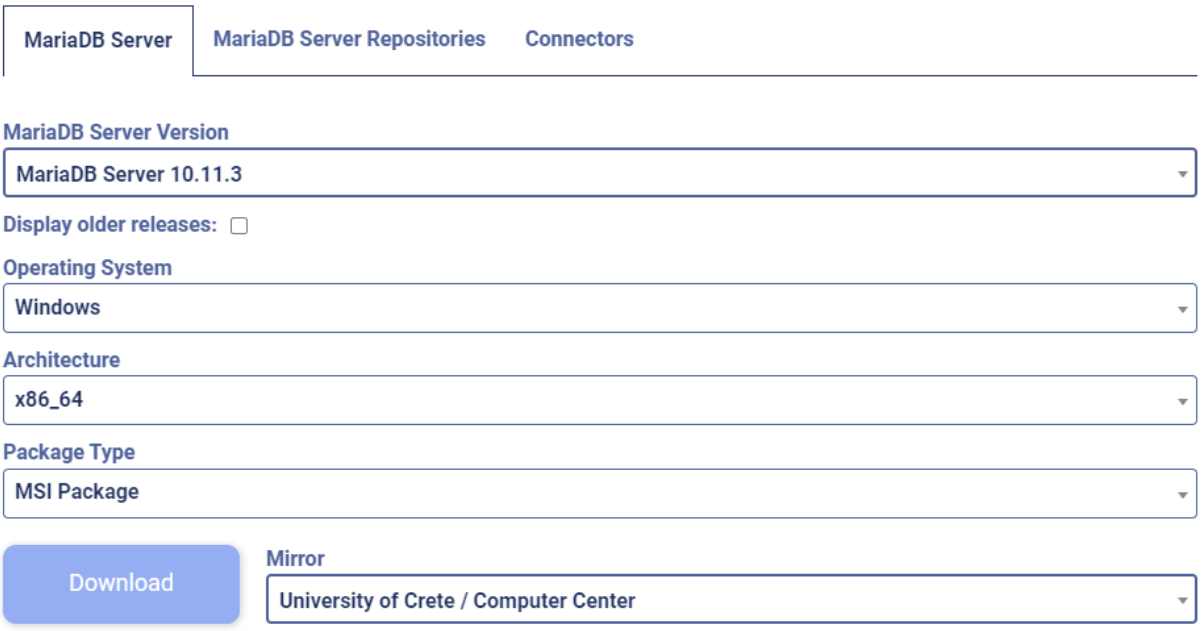
- MariaDB 10.11
- MySQL Workbench 8.0 CE
- Flask 2.1.1
- Python 3.11

Εγκατάσταση Απαραίτητου Λογισμικού

Windows 11

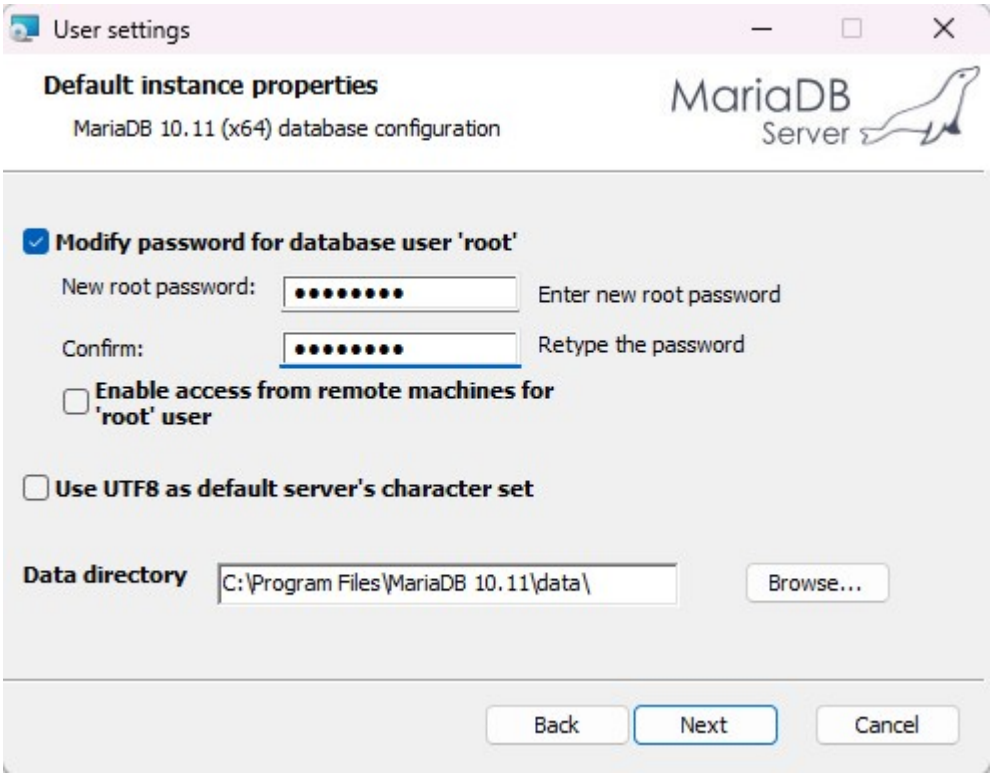
Για την εγκατάσταση της **MariaDB**:

1. Ακολουθούμε τον σύνδεσμο Download MariaDB Server - MariaDB.org και κατεβάζουμε την έκδοση του MariaDB που μας αφορά (10.11).



2. Τρέχουμε το αρχείο που κατεβάσαμε (mariadb-10.11.3-winx64.msi).
3. Διαβάζουμε προσεκτικά το License Agreement και αποδεχόμαστε τους όρους αν θέλουμε να συνεχίσουμε με την εγκατάσταση.
4. Επιλέγουμε την τοποθεσία εγκατάστασης του MariaDB (C:\Program Files\MariaDB 10.11\).

5. Θέτουμε password για τον χρήστη root και ορίζουμε το data directory (C:\Program Files\MariaDB 10.11\data\).



6. Πατάμε εγκατάσταση.
7. Ανοίγουμε το cmd prompt και πληκτρολογούμε:
   cd "C:\Program Files\MariaDB 10.11\bin"

```
Microsoft Windows [Version 10.0.22621.1702]
(c) Microsoft Corporation. All rights reserved.

C:\Windows\System32>cd "C:\Program Files\MariaDB 10.11\bin"
```

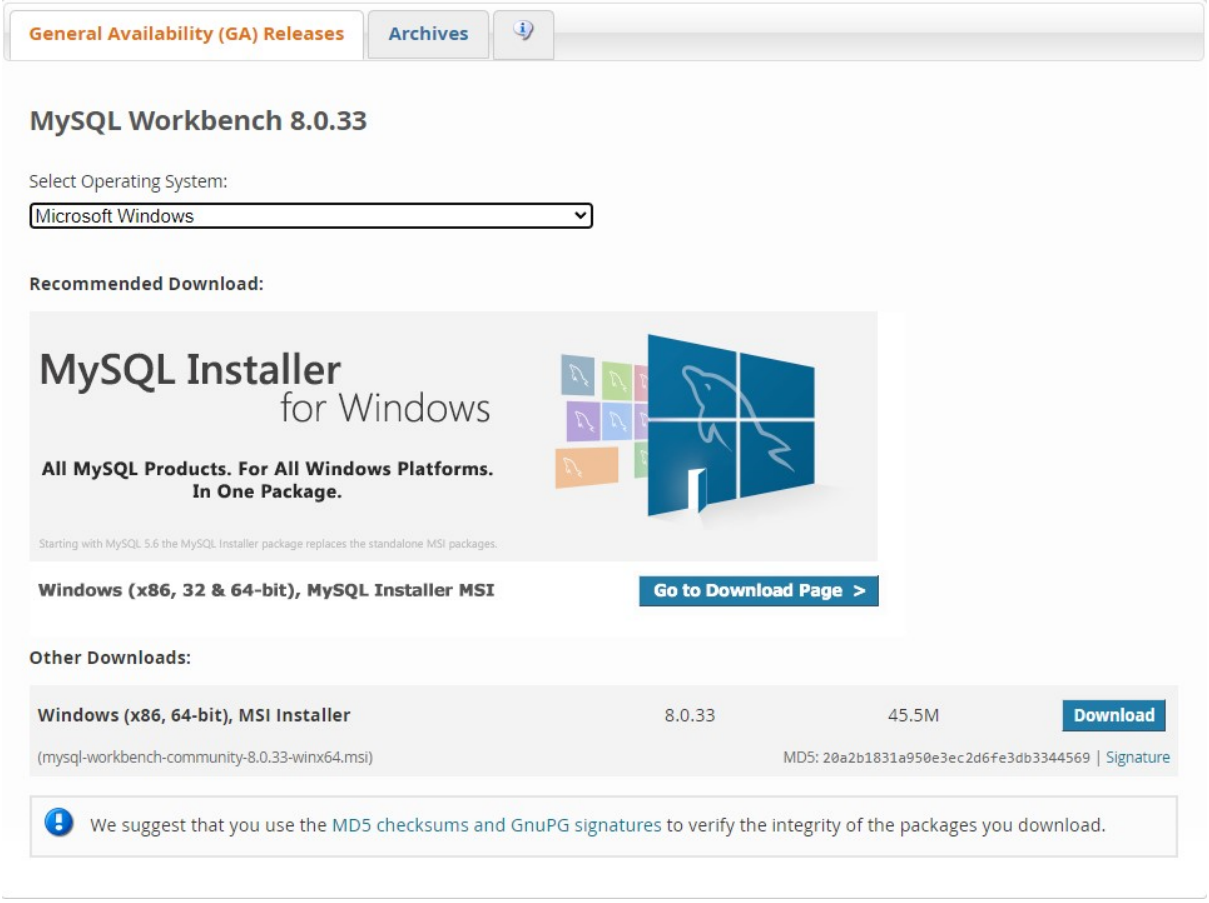
8. Πληκτρολογούμε:
   mysql -u root -p

```
C:\Program Files\MariaDB 10.11\bin>mysql -u root -p
Enter password:
```

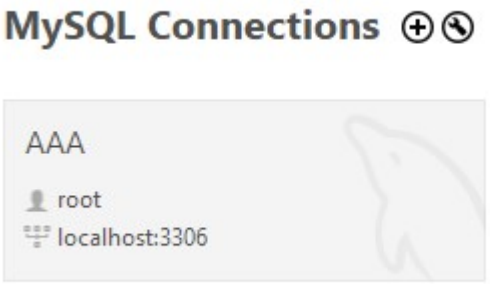
9. Χρησιμοποιούμε τον κωδικό που ορίσαμε στο βήμα 5.

Για το **MySQL Workbench**:

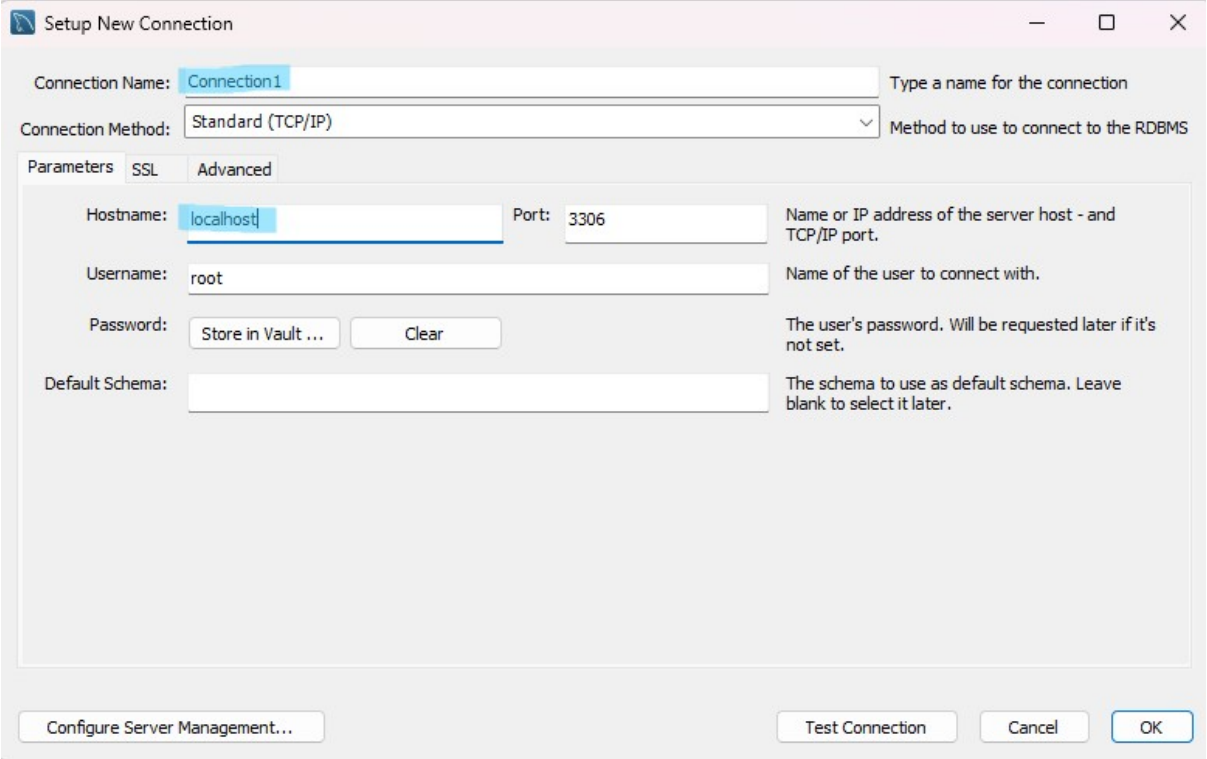
1. Κατεβάζουμε το MySQL Workbench μέσω του συνδέσμου (MySQL :: Download MySQL Workbench).



2. Τρέχουμε το αρχείο που κατεβάσαμε (mysql-workbench-community-8.0.33-winx64.msi) και εγκαθιστούμε το MySQL Workbench.
3. Ανοίγουμε το MySQL Workbench και δημιουργούμε μία σύνδεση με τον server (Πατάμε το “+”).



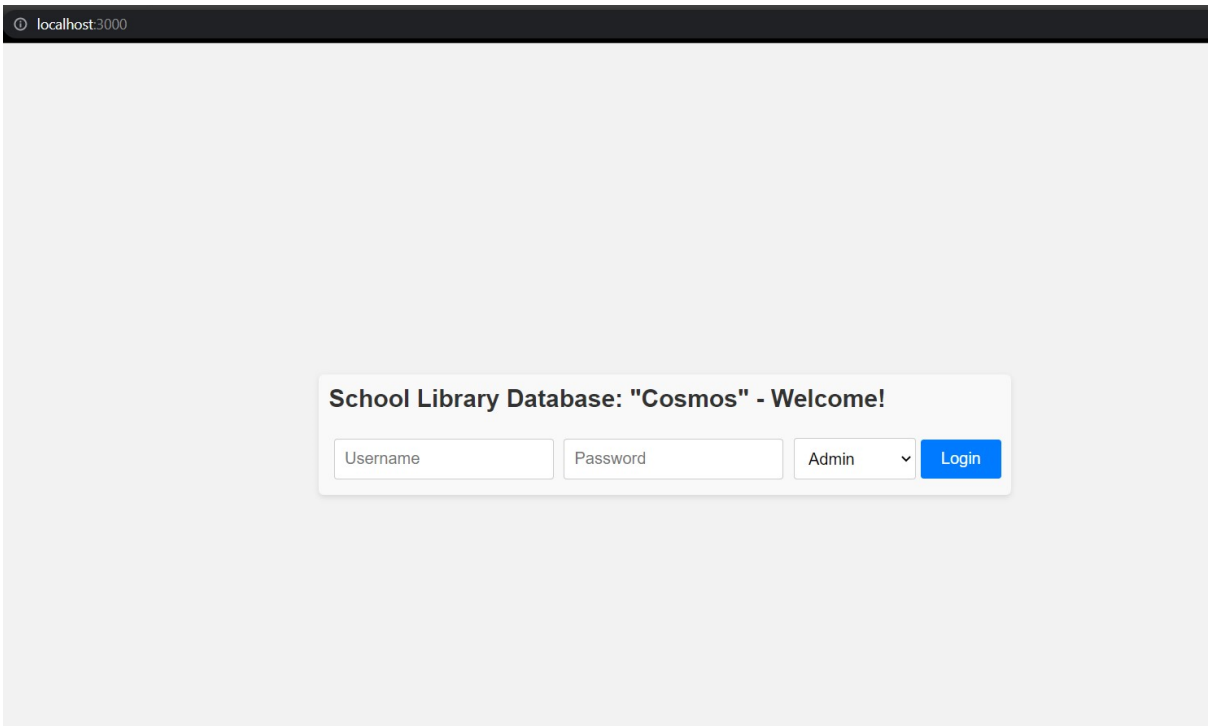
4. Ονομάζουμε την σύνδεσή μας και ορίζουμε ως Hostname το localhost και χρησιμοποιούμε το κατάλληλο username και password.



5. Ελέγχουμε την σύνδεσή μας και πατάμε OK.
6. Τρέχουμε τα αρχεία schema.sql και insert_data.sql με αυτή την σειρά.

Για την **Εφαρμογή:**

1. Εγκαθιστούμε την έκδοση της Python που μας ενδιαφέρει (Python 3.11) μέσω του συνδέσμου Download Python | Python.org.
2. Εγκαθιστούμε τις απαραίτητες βιβλιοθήκες για την σωστή λειτουργία της εφαρμογής μέσω της εντολής “pip install -r requirements.txt”.
3. Έτσι μπορούμε να τρέξουμε την εφαρμογή μέσω του run.py.
4. Ανοίγουμε έναν browser και πληκτρολογούμε στο search bar “localhost:3000”.



Σύνδεσμος για το git repo της εφαρμογής

https://github.com/kourisgeorgios/librarydb.git

Σημείωση:
Για τη σύνδεση στην ιστοσελίδα ζητώνται credentials. Έχουμε ορίσει και τους ακόλουθους χρήστες, για τη διευκόλυνση της εξέτασης:

Administrator με:
username: admin
password: admin
type: Administrator

School Library Operator με:
username: operator
password: operator
type: Operator

Teacher με:
username: teacher
password: teacher
type: User

Student με:
username: student
password: student
type: User