



ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ

ΣΧΟΛΗ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ ΚΑΙ
ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ

ΒΑΣΕΙΣ ΔΕΔΟΜΕΝΩΝ
ΑΝΑΦΟΡΑ ΕΞΑΜΗΝΙΑΙΑΣ ΕΡΓΑΣΙΑΣ

Εαρινό εξάμηνο 2022-2023

ΟΜΑΔΑ 118

Ονοματεπώνυμο: Δήμητρα Σεφεριάδη

Αριθμός Μητρώου: 03120131

Ονοματεπώνυμο: Ίρις Ελευθερία Παλιατσού

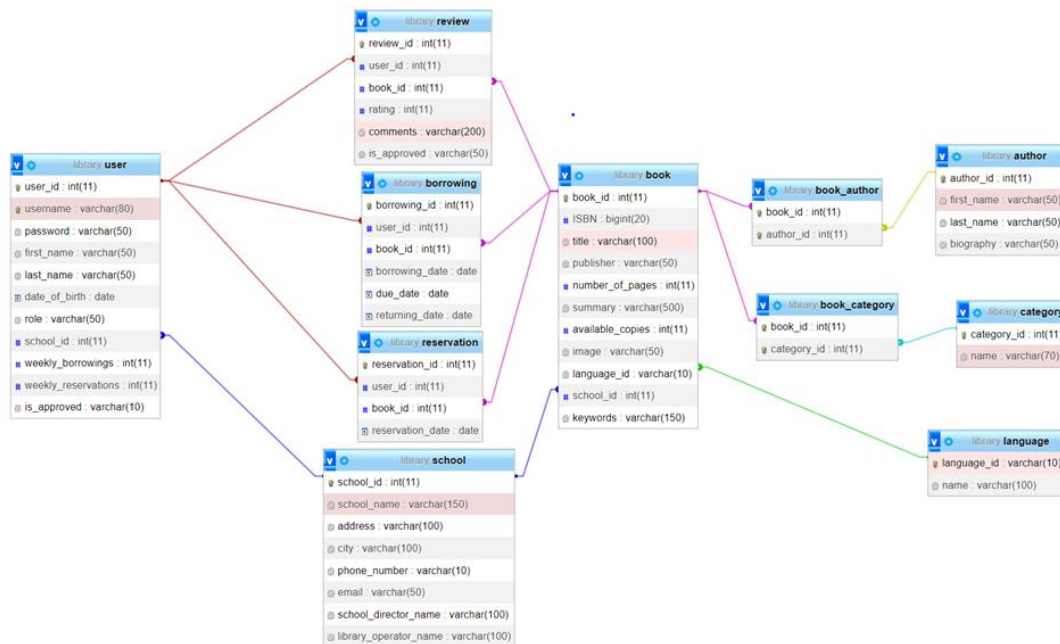
Αριθμός Μητρώου: 03120639

Ονοματεπώνυμο: Ελισάβετ Παπαδοπούλου

Αριθμός Μητρώου: 03120190

➤ ER και Σχεσιακό Διάγραμμα της Βάσης Δεδομένων

Παραθέτουμε το σχεσιακό διάγραμμα:



Και το διάγραμμα οντοτήτων:



Σχολιασμός Διαγράμματος Οντοτήτων

Στην βάση μας περιέχονται τρεις κύριες οντότητες: οι χρήστες (table “user”), τα σχολεία (table “school”) και τα βιβλία (table “book”). Επιπλέον οντότητες που σχετίζονται με τις τρεις προηγούμενες κύριες είναι οι συγγραφείς (table “author”), οι γλώσσες (table “language”) και οι κατηγορίες (table “category”). Οι δανεισμοί (table “borrowing”), οι

κρατήσεις (table “reservation”) και οι αξιολογήσεις (table “review”) των βιβλίων παρουσιάζονται ως σχέσεις μεταξύ των οντοτήτων User και Book.

Αναφορικά με τις σχέσεις:

Αρχικά, κάθε δανεισμός, κράτηση και αξιολόγηση πρέπει να γίνει από έναν ακριβώς χρήστη, ενώ ένας χρήστης μπορεί να πραγματοποιήσει από έναν/καμία έως πολλούς/πολλές αντίστοιχα.

Επίσης, κάθε χρήστης πρέπει να ανήκει σε ένα ακριβώς σχολείο, ενώ κάθε σχολείο μπορεί να έχει παραπάνω από έναν χρήστη.

Το βιβλίο αντίστοιχα μπορεί να έχει από κανέναν έως πολλούς δανεισμούς, κρατήσεις και αξιολογήσεις, ενώ κάθε ένα από αυτά πρέπει να αναφέρεται σε ένα ακριβώς βιβλίο.

Ακόμη, κάθε βιβλίο πρέπει να ανήκει σε ακριβώς ένα σχολείο, ενώ ένα σχολείο μπορεί να έχει παραπάνω από ένα βιβλία.

Τέλος, κάθε βιβλίο μπορεί να έχει από έναν και πάνω συγγραφείς, να ανήκει σε μία και πάνω κατηγορίες και να είναι γραμμένο σε ακριβώς μία γλώσσα, ενώ τόσο οι συγγραφείς αλλά και οι κατηγορίες δεν είναι απαραίτητο ότι έχουν κάποιο βιβλίο.

Σχολιασμός Σχεσιακού Διαγράμματος

Οι προαναφερθείσες κύριες οντότητες συμπληρώνονται στο σχεσιακό μας διάγραμμα με τους πίνακες (junction tables) book_author και book_category. Επιτρέπουν την ανάθεση περισσότερων του ενός συγγραφείς και κατηγορίες σε κάθε βιβλίο αν χρειαστεί.

Indexing Βάσης Δεδομένων

Για να ορίσουμε τα indices της βάσης, αρχικά γνωρίζουμε πως στην mysql, την οποία και χρησιμοποιήσαμε, τα Engine Indices δημιουργούνται αυτόματα για τα primary keys ενός πίνακα. Επομένως, όλα τα ids που χρησιμοποιούνται για πολλά queries, triggers, events είναι ήδη indexed, γεγονός πολύ σημαντικό για την γρήγορη αξιοποίηση τους.

Στα indexes μας συμπεριλάβαμε όλα τα foreign keys των πινάκων μας τα οποία δεν είναι primary keys.

Constraints Βάσης Δεδομένων

- Το ISBN των βιβλίων πρέπει να έχει ακριβώς 13 ψηφία.

- Το rating μιας αξιολόγησης (table “review”) πρέπει να είναι ένας αριθμός από το 1 ως το 5
- Το όρισμα is_approved του πίνακα της αξιολόγησης μπορεί να περιέχει μόνο τις λέξεις “yes”, “no” ή να είναι κενό
- Το τηλέφωνο “phone_number” ενός σχολείου πρέπει να έχει ακριβώς 10 ψηφία.
- Το όρισμα is_approved του πίνακα του χρήστη “user” μπορεί να περιέχει μόνο τις λέξεις “yes”, “no” ή να είναι κενό

➤ DDL και DML scripts

DDL scripts

Στην βάση δεδομένων μας () υπάρχουν τα αρχεία:

- delete_data_base.sql

Αυτό το αρχείο διαγράφει από την βάση όλους τους υπάρχοντες πίνακες:

- create_data_base.sql

Αυτό το αρχείο δημιουργεί την βάση, με όλους τους πίνακες, τις εξαρτήσεις primary/foreign keys, check constraints και triggers. Ενδεικτικά, η δημιουργία των δύο πρώτων πινάκων παρακάτω:

DML scripts

- Το final_dummy_data.sql αρχείο εισάγει όλα τα δεδομένα στην βάση.

Το αρχείο αυτό δημιουργήθηκε με χρήση της βιβλιοθήκης faker της Python. Ο κώδικας με τον οποίο δημιουργήθηκαν τα δεδομένα δίνεται στον φάκελο library με το όνομα: data_generator.py. Για να τρέξουμε τον κώδικα αυτόν, χρησιμοποιήθηκαν οι εξής εντολές, μέσω του terminal: **pip install faker** (για την χρήση της βιβλιοθήκης Faker) και **pip install mysql-connector-python** (για τη σύνδεση με τη βάση).

Από τον κώδικα αυτόν, παράγεται το αρχείο dummy_data.txt. Στη συνέχεια, μετονομάζουμε αυτό το αρχείο σε final_dummy_data.sql ώστε να το φορτώσουμε στη βάση.

➤ **Back_up files**

Προκειμένου να δημιουργήσουμε ένα **back-up** αρχείο για την βάση μας, ακολουθούμε τα παρακάτω βήματα:

1. Ανοίγουμε το terminal του υπολογιστή μας
2. Με cd μπαίνουμε στο σωστό directory (όπου είναι αποθηκευμένη η βάση μας και εκτελούμε τις παρακάτω εντολές:
 - a. `cd C:\xampp\mysql\bin`
 - b. `Mysqldump --databases library -u root -p > path\to\backup_file\backup_database_structure.sql`
3. Το path\to\backup_file πρόκειται για το path στο οποίο επιθυμούμε να δημιουργηθεί το αντίγραφο ασφαλείας

Για **restore** της βάσης δεδομένων μας, ακολουθούμε τα παρακάτω βήματα:

1. Ανοίγουμε το terminal του υπολογιστή μας
2. Με cd μπαίνουμε στο σωστό directory (όπου είναι αποθηκευμένη η βάση μας και εκτελούμε τις παρακάτω εντολές:
 - a. `mysql -u root -p`
 - b. `SOURCE path\to\backup\file`

➤ **Βήματα Εγκατάστασης**

Βήμα 1^ο - Κατέβασμα του repository μέσω git

Για την εγκατάσταση της εφαρμογής πρέπει να γίνει clone το git repo της εφαρμογής, είτε μέσω της εφαρμογής GitHub desktop, είτε μέσω terminal με την εντολή

git clone <https://github.com/elisavetpapadopoulou/Library-Data-Base>
στο directory που επιθυμούμε να εγκαταστήσουμε τη βάση.

Βήμα 2^ο – Εγκατάσταση της βάσης και εισαγωγή των dummy data

Για την εγκατάσταση της βάσης στον υπολογιστή μας χρειαζόμαστε έναν sql server, εμείς χρησιμοποιήσαμε mysql μέσω xampp. Αρχικά, κατασκευάσαμε ένα database, μέσω της εντολής
CREATE DATABASE library;

και τρέξαμε τα ακόλουθα αρχεία με τη δοθείσα σειρά και τις εντολές:
source delete_database.sql
source create_database.sql
source final_dummy_data.sql

Βήμα 3^ο – Launch της εφαρμογής μέσω Local host

Αρχικά, πρέπει να εγκατασταθούν όλες οι απαραίτητες βιβλιοθήκες που θα χρησιμοποιηθούν στην εφαρμογή μας. Βρίσκονται στο αρχείο requirments.txt το οποίο τρέχουμε στο terminal μέσω της εντολής

pip install -r requirments.txt

Καθώς η υλοποίηση της εφαρμογής γίνεται με pythοn πρέπει να είναι ήδη εγκαταστημένη στον υπολογιστή μας (σαν ομάδα χρησιμοποιήσαμε Visual Studio Code μέσω Anaconda).

Για να δούμε την εφαρμογή τρέχουμε το αρχείο **app.py**. Ανοίγουμε έναν browser στη διεύθυνση localhost:5000 και πρέπει να εμφανίζεται η ακόλουθη σελίδα:

Welcome to School Library Network!

[Login Page](#)

[Register Page](#)

➤ **User Manual**

Ο χρήστης έχει τη δυνατότητα να εγγραφεί στη βάση:


Register Page

Username:

Password:

First name:

Last name:

Date of birth: 

School:

Role

☐ student

☐ teacher

☐ operator

[Already have an account? Log In](#)

Ή αν έχει ήδη λογαριασμό να συνδεθεί κατευθείαν:

Login Page

[Don't have an account? Sign Up](#)

Ένας μαθητής ή καθηγητής πρέπει μετά την εγγραφή του πρέπει να γίνει αποδεκτός από τον χειριστή του σχολείου του και αντίστοιχα ένας χειριστής να γίνει αποδεκτός από τον κεντρικό διαχειριστή για να εισέλθει στη Σχολική Βιβλιοθήκη.

Ανάλογα τον ρόλο που έχει ο χρήστης μεταφέρεται στην ανάλογη σελίδα.

➤ **Student**

[Library Database](#) - [Home](#) [Personal Information](#) [Edit Password](#) [Logout](#)

User's Dashboard

[View books and make a reservation](#) [Write a review](#) [My reservations](#) [My reviews](#) [Borrowed books](#)

Αναφέρω για λόγους ευκολίας έναν χρήστη που ήδη υπάρχει στο dummy_data.txt

username: paulthornton

password: 8(ga3#hsOS

ή

username: jacquelinejenkins

password: UE3jkUig%A

➤ Teacher

[Library Database -](#) [Home](#) [Personal Information](#) [Edit Password](#) [Logout](#)

User's Dashboard

[Edit personal information](#) [View books and make a reservation](#) [Write a review](#) [My reservations](#) [My reviews](#) [Borrowed books](#)

Αναφέρω για λόγους ευκολίας έναν χρήστη που ήδη υπάρχει στο dummy_data.txt

username: james53

password: gwW!EGeL+6

➤ Operator

[Library Database -](#) [Home](#) [Personal Information](#) [Edit Password](#) [Logout](#)

Operator's Dashboard

[Record loans without reservation](#) [Record loans with reservation](#) [Record the return of a book](#) [View reservations](#) [Search in reservations for a specific user](#) [View loans](#) [Search in loans for a specific user](#) [View delayed returns](#) [Search in delayed loans for a specific user](#) [View all books](#) [Average ratings](#) [Add a new book](#) [Edit book](#) [Approve review](#) [Approve user](#) [Disable or delete user](#)

Αναφέρω για λόγους ευκολίας έναν χρήστη που ήδη υπάρχει στο dummy_data.txt

username: hopkinsjeffrey

password: JzY3SBxJV@

➤ Administrator

[Library Database -](#) [Home](#) [Personal Information](#) [Edit Password](#) [Logout](#)

Administrator's Dashboard

[Insert School](#) [Approve Operator](#) [Loan Statistics](#) [Category Statistics](#) [View young teachers who have borrowed the most books](#) [Unborrowed Authors](#) [Operators with Same Loans](#) [Common Category Pairs](#) [Authors with Fewer Books](#)

Αναφέρω για λόγους ευκολίας έναν χρήστη που ήδη υπάρχει στο dummy_data.txt

username: amber20

password:)GJgmymn!4

Υπάρχουν, φυσικά, όλες οι προαπαιτήσεις για τις λειτουργίες που μπορεί να εκτελέσει ο κάθε χρήστης όπως ακριβώς μας ζητούνται από την εκφώνηση. Αναλυτικότερα, παραθέτουμε τα queries που χρησιμοποιήθηκαν για το μέρος 3.

➤ Ανάπτυξη εφαρμογής και διεπαφής χρήστη

3.1 Administrator

3.1.1. Παρουσίαση λίστας με συνολικό αριθμό δανεισμών ανά σχολείο (κριτήρια αναζήτησης έτος, μήνας)

```
SELECT s.school_id, s.school_name, COUNT(*) AS total_loans
FROM borrowing b
INNER JOIN user u ON b.user_id = u.user_id
INNER JOIN school s ON u.school_id = s.school_id
WHERE YEAR(b.borrowing_date) = %s AND
MONTH(b.borrowing_date) = %s
GROUP BY s.school_id, s.school_name
```

3.1.2. Για δεδομένη κατηγορία βιβλίων (επιλέγει ο χρήστης), ποιοι συγγραφείς ανήκουν σε αυτήν και ποιοι εκπαιδευτικοί έχουν δανειστεί βιβλία αυτής της κατηγορίας το τελευταίο έτος;

Οι συγγραφείς που ανήκουν σε δεδομένη κατηγορία :

```
SELECT DISTINCT a.author_id, a.first_name, a.last_name
FROM author a
JOIN book_author ba ON a.author_id = ba.author_id
JOIN book_category bc ON ba.book_id = bc.book_id
JOIN category c ON bc.category_id = c.category_id
```

```
WHERE c.name = %s;
```

Οι εκπαιδευτικοί που έχουν δανειστεί βιβλία αυτής της κατηγορίας το τελευταίο έτος:

```
SELECT DISTINCT u.user_id, u.first_name, u.last_name
FROM user u
JOIN borrowing b ON u.user_id = b.user_id
JOIN book bo ON b.book_id = bo.book_id
JOIN book_category bc ON bo.book_id = bc.book_id
JOIN category c ON bc.category_id = c.category_id
WHERE c.name = %s
AND b.borrowing_date >= DATE_SUB(CURDATE(), INTERVAL 1
YEAR)
AND u.role = 'teacher';
```

3.1.3. Βρείτε τους νέους εκπαιδευτικούς (ηλικία < 40 ετών) που έχουν δανειστεί τα περισσότερα βιβλία και των αριθμό των βιβλίων.

```
SELECT u.user_id, u.first_name, u.last_name, COUNT(*) AS
num_borrowed_books
FROM user u
JOIN borrowing b ON u.user_id = b.user_id
JOIN book bo ON b.book_id = bo.book_id
WHERE u.role = 'Teacher'
AND TIMESTAMPDIFF(YEAR, u.date_of_birth, CURDATE()) < 40
GROUP BY u.user_id, u.first_name, u.last_name
ORDER BY num_borrowed_books DESC
LIMIT 10;
```

3.1.4. Βρείτε τους συγγραφείς των οποίων κανένα βιβλίο δεν έχει τύχει δανεισμού.

```
SELECT DISTINCT a.author_id, a.first_name, a.last_name
FROM author a
```

```
JOIN book_author ba ON a.author_id = ba.author_id
JOIN book b ON ba.book_id = b.book_id
LEFT JOIN borrowing bor ON b.book_id = bor.book_id
WHERE bor.borrowing_id IS NULL;
```

3.1.5. Ποιοι χειριστές έχουν δανείσει τον ίδιο αριθμό βιβλίων σε διάστημα ενός έτους με περισσότερους από 20 δανεισμούς;

```
SELECT COUNT(borrowing_id) FROM borrowing
JOIN user ON user.user_id=borrowing.user_id
WHERE user.school_id = %s AND YEAR(borrowing.borrowing_date) =
%s
```

(Επειδή το query έχει οριστεί μέσα σε ένα for loop για όλους τους operators βρίσκουμε τελικά το ζητούμενο μέσω ενός dictionary)

3.1.6. Πολλά βιβλία καλύπτουν περισσότερες από μια κατηγορίες. Ανάμεσα σε ζεύγη πεδίων (π.χ. ιστορία και ποίηση) που είναι κοινά στα βιβλία, βρείτε τα 3 κορυφαία (top-3) ζεύγη που εμφανίστηκαν σε δανεισμούς.

```
SELECT bc1.category_id AS category1_id, c1.name AS category1_name,
       bc2.category_id AS category2_id, c2.name AS category2_name,
       COUNT(*) AS borrowings_count
FROM borrowing b
JOIN book bo ON b.book_id = bo.book_id
JOIN book_category bc1 ON bo.book_id = bc1.book_id
JOIN book_category bc2 ON bo.book_id = bc2.book_id
JOIN category c1 ON bc1.category_id = c1.category_id
JOIN category c2 ON bc2.category_id = c2.category_id
WHERE bc1.category_id < bc2.category_id
GROUP BY bc1.category_id, c1.name, bc2.category_id, c2.name
ORDER BY borrowings_count DESC
LIMIT 3;
```

3.1.7. Βρείτε όλους τους συγγραφείς που έχουν γράψει τουλάχιστον 5 βιβλία λιγότερα από τον συγγραφέα με τα περισσότερα βιβλία.

```
SELECT author_id, COUNT(*) AS book_count
FROM book_author
GROUP BY author_id
ORDER BY book_count DESC
LIMIT 1
```

```
SELECT ba.author_id, COUNT(*) AS book_count, a.first_name,
a.last_name
FROM book_author ba
INNER JOIN author a ON ba.author_id = a.author_id
WHERE ba.author_id != %s
GROUP BY ba.author_id, a.first_name, a.last_name
HAVING COUNT(*) <= %s - 5
ORDER BY book_count DESC
```

3.2 Operator

3.2.2. Εύρεση όλων των δανειζόμενων που έχουν στην κατοχή τους τουλάχιστον ένα βιβλίο και έχουν καθυστερήσει την επιστροφή του. (Κριτήρια αναζήτησης: Όνομα, Επώνυμο, Ημέρες Καθυστερήσης).

```
SELECT u.first_name, u.last_name, DATEDIFF(CURDATE(),
b.due_date) AS delay_days
FROM borrowing b
INNER JOIN user u ON b.user_id = u.user_id
WHERE u.first_name = %s
AND u.last_name = %s
AND b.returning_date IS NULL
AND DATEDIFF(CURDATE(), b.due_date) = %s
```

3.2.3. Μέσος Όρος Αξιολογήσεων ανά δανειζόμενο και κατηγορία (Κριτήρια αναζήτησης: χρήστης/ κατηγορία)

```
SELECT u.username, c.name AS category, AVG(r.rating) AS  
average_rating  
FROM user u  
INNER JOIN review r ON u.user_id = r.user_id  
INNER JOIN book_category bc ON r.book_id = bc.book_id  
INNER JOIN category c ON bc.category_id = c.category_id  
WHERE u.username = %s  
AND c.name = %s  
GROUP BY u.username, c.name
```

3.3 User

3.3.2. Λίστα όλων των βιβλίων που έχει δανειστεί ο συγκεκριμένος χρήστης.

```
SELECT b.*  
FROM borrowing br  
INNER JOIN book b ON br.book_id = b.book_id  
WHERE br.user_id = %s
```

Στα ερωτήματα που δεν δίνουμε κάποιο query είναι επειδή στη python παίρνουμε πολλές υποπεριπτώσεις και τα queries που χρησιμοποιούμε είναι αρκετά αλλά απλά. Πιο αναλυτικά, όλα εμπεριέχονται στο αρχείο **app.py**.