

ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ
ΣΧΟΛΗ ΕΦΑΡΜΟΣΜΕΝΩΝ ΜΑΘΗΜΑΤΙΚΩΝ ΚΑΙ ΦΥΣΙΚΩΝ
ΕΠΙΣΤΗΜΩΝ

Εξαμηνιαία Εργασία στο Μάθημα των Βάσεων Δεδομένων

Εαρινό Εξάμηνο 2022-2023



ΟΜΑΔΑ 22

ΦΟΙΤΗΤΕΣ:

ΟΝΟΜΑΤΕΠΩΝΥΜΟ: ΠΕΡΒΟΛΑΡΑΚΗΣ ΔΗΜΤΡΙΟΣ

ΑΡΙΘΜΟΣ ΜΗΤΡΩΟΥ: ge19067

ΟΝΟΜΑΤΕΠΩΝΥΜΟ: ΠΑΤΟΥΧΕΑΣ ΕΛΕΥΘΕΡΙΟΣ

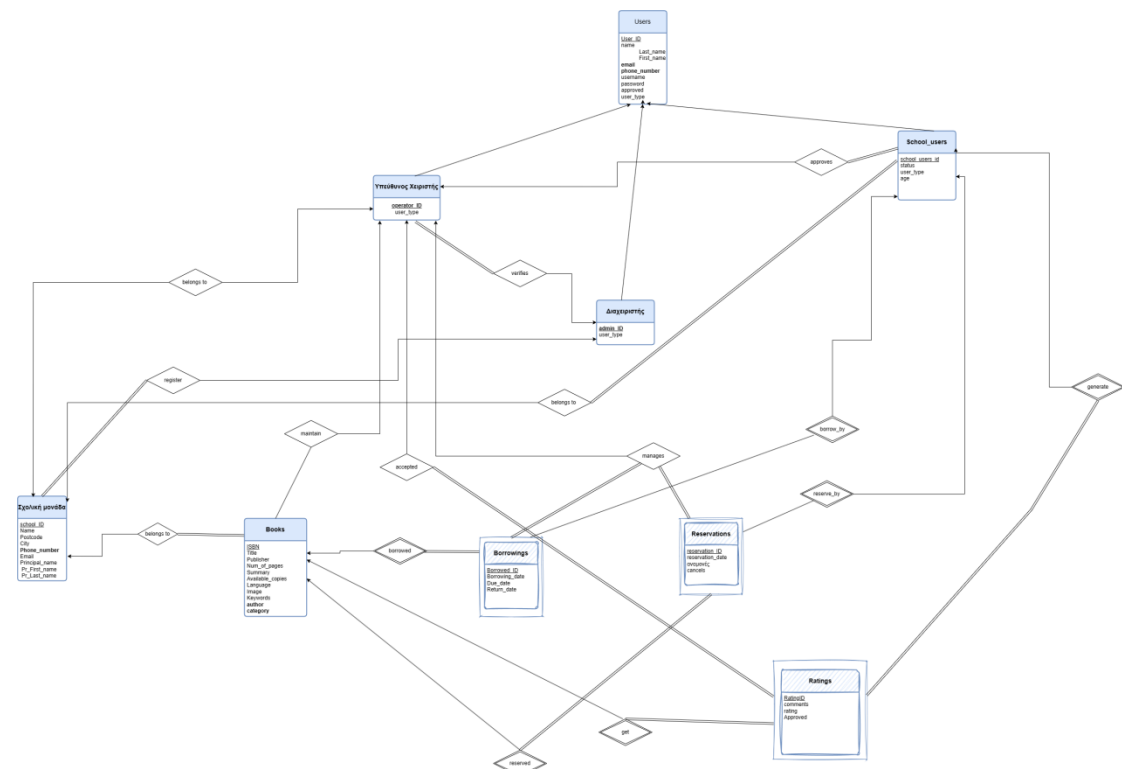
ΑΡΙΘΜΟΣ ΜΗΤΡΩΟΥ: ge19030

ΟΝΟΜΑΤΕΠΩΝΥΜΟ: ΣΠΕΝΤΖΟΥΡΗΣ ΙΩΑΝΝΗΣ

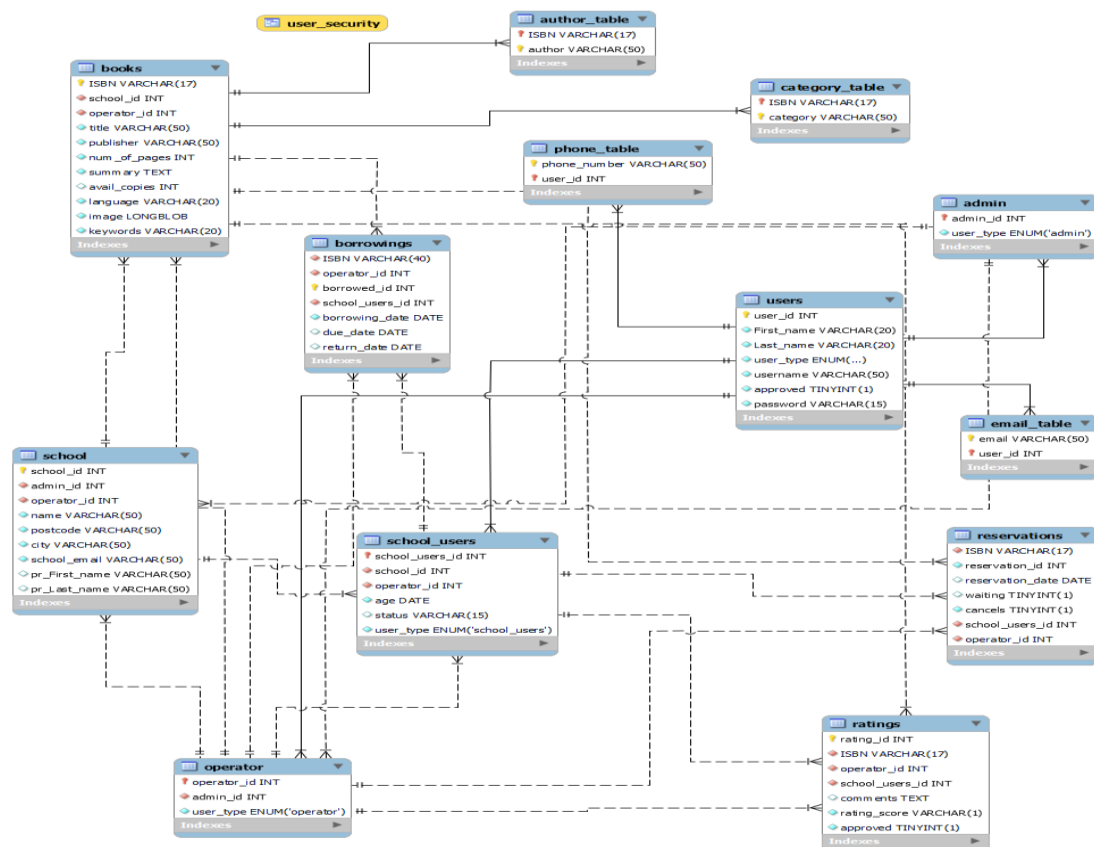
ΑΡΙΘΜΟΣ ΜΗΤΡΩΟΥ: ge19038

ΑΘΗΝΑ, ΙΟΥΝΙΟΣ 2023

ΕΡ ΔΙΑΓΡΑΜΜΑ



ΣΧΕΣΙΑΚΟ ΔΙΑΓΡΑΜΜΑ



Σχολιασμός ER διαγράμματος:

Τα υπογραμμισμένα attributes είναι τα primary keys της κάθε οντότητας, ενώ τα μαυρισμένα attributes είναι αυτά για τα οποία δημιουργήθηκε ένα ξεχωριστό table στο σχεσιακό διάγραμμα.

Σχολιασμός σχεσιακού διαγράμματος:

Η βάση μας αποτελείται από μια κύρια οντότητα, τους χρήστες της ιστοσελίδας (users). Η κατηγοριοποίηση των χρηστών γίνεται μέσω της εντολής “ENUM” της MySQL και έτσι μπορούμε να την χωρίσουμε σε τρεις μικρότερες οντότητες, τον διαχειριστή της ιστοσελίδας (admin), τους χειριστές από κάθε σχολείο (operators) και τους χρήστες από τα σχολεία (school_users), οι οποίοι μπορεί να είναι είτε μαθητές, είτε καθηγητές.

Κάθε χρήστης ανήκει σε ένα σχολείο και ένα σχολείο έχει έναν μοναδικό χειριστή. Γι’ αυτό υπάρχει και η οντότητα school που περιέχει όλα τα σχολεία. Κάθε σχολείο περιέχει συγκεκριμένα βιβλία, για τα οποία έχουμε δημιουργήσει μια οντότητα books, η οποία συνδέεται με το *author_table*, το οποίο περιέχει τους συγγραφείς των βιβλίων και το *category_table*, το οποίο έχει τις κατηγορίες του κάθε βιβλίου. Επίσης, υπάρχουν και τα *email_table*, *phone_table*, τα οποία είναι συνδεδεμένα με το users και εκφράζουν τα email και αριθμούς τηλεφώνων των χρηστών αντίστοιχα. Τα tables τα οποία τα έχουμε γράψει πλαγίως, ήταν attributes στο ER διάγραμμα, αλλά τα μετατρέψαμε σε οντότητες στο σχεσιακό, επειδή παραπάνω από 1 από αυτά αντιστοιχεί σε ένα στοιχείο των κύριων οντοτήτων.

Τέλος, έχουμε τις οντότητες borrowings, reservations και ratings, που αφορούν τη λειτουργία της ιστοσελίδας στην οποία ο χρήστης μπορεί να κάνει κράτηση σε ένα βιβλίο και μετά ο αντίστοιχος χειριστής του σχολείου να μετατρέψει (αν υπάρχει διαθεσιμότητα) την κράτηση σε δανεισμό βιβλίου. Τέλος, ο χρήστης για κάθε βιβλίο το οποίο έχει δανειστεί, μπορεί να το βαθμολογήσει και να γράψει ένα σχολιασμό.

Constraints

- Μέσω της εντολής “check”, ορίζουμε το attribute “status”, να μπορεί να πάρει μόνο τις τιμές “student”, “teacher”. Επίσης, ορίζουμε το attribute “rating_score” να πάρει τιμές ακεραίων από το 1 ως το 5.
- Οι υπόλοιποι περιορισμοί έγιναν μέσω triggers.

Φτιάχνουμε ένα event το οποίο να διαγράφει τις κρατήσεις μια βδομάδα μετά τη δημιουργία τους:

```

CREATE EVENT IF NOT EXISTS `delete_reservations_event`
ON SCHEDULE
    EVERY 1 DAY HOUR
ON COMPLETION PRESERVE
COMMENT 'Clean up reservations.'
DO
    DELETE FROM reservations
    WHERE reservation_date < DATE_SUB(NOW(), INTERVAL 7
DAY)

```

Trigger για περιορισμούς στις κρατήσεις:

- Ένας μαθητής μπορεί να κάνει το μέγιστο 2 κρατήσεις τη βδομάδα.
- Ένας καθηγητής μπορεί να κάνει το μέγιστο μια κράτηση τη βδομάδα.
- Ο χρήστης δεν μπορεί να κάνει κράτηση αν το βιβλίο δεν έχει επιστραφεί στην ώρα του.
- Ο χρήστης δεν μπορεί να κάνει κράτηση για ένα τίτλο, εάν τον έχει ήδη στην κατοχή του.

```

DELIMITER $
CREATE TRIGGER chk_num_of_reservations BEFORE INSERT ON reservations
FOR EACH ROW
BEGIN
    IF (new.school_users_id = (SELECT r.school_users_id from reservations r INNER
JOIN school_users s ON s.school_users_id = r.school_users_id INNER JOIN books b ON
b.ISBN = r.ISBN
    WHERE s.status = "student" AND r.ISBN = new.ISBN AND
DATEDIFF(CURDATE(),r.reservation_date) <7 GROUP BY r.school_users_id HAVING COUNT(*)=
2) ) THEN
        SIGNAL SQLSTATE '45000'
        SET MESSAGE_TEXT = 'check constraint on reservations failed - A student can
only make 2 reservations a week.';
    END IF;
    IF (new.school_users_id = (SELECT r.school_users_id from reservations r INNER JOIN
school_users s ON s.school_users_id = r.school_users_id INNER JOIN books b ON b.ISBN =
r.ISBN
    WHERE s.status = "teacher" AND r.ISBN = new.ISBN AND
DATEDIFF(CURDATE(),r.reservation_date) <7 GROUP BY r.school_users_id HAVING COUNT(*)=
1) ) THEN
        SIGNAL SQLSTATE '45000'
        SET MESSAGE_TEXT = 'check constraint on reservations failed - A teacher can
only make 1 reservations a week.';
    END IF;
    IF(new.school_users_id = (select school_users_id from borrowings WHERE return_date IS
NULL AND datediff(CURDATE(),due_date)>=1 ) ) THEN
        SIGNAL SQLSTATE '45000'
        SET MESSAGE_TEXT = 'check constraint on reservations failed - A user cannot
make reservation if a book has not been returned on time.';
    END IF;
    IF (new.ISBN = (SELECT ISBN FROM borrowings WHERE school_users_id =
new.school_users_id AND return_date is null AND ISBN=new.ISBN) ) THEN
        SIGNAL SQLSTATE '45000'
        SET MESSAGE_TEXT = 'check constraint on reservations failed - A user cannot
make reservation if the same user has already borrowed the title.';
    END IF;
END $
DELIMITER ;

```

Trigger για περιορισμούς στους δανεισμούς:

- Ένας μαθητής μπορεί να κάνει το μέγιστο 2 δανεισμούς τη βδομάδα.
- Ένας καθηγητής μπορεί να κάνει το μέγιστο έναν δανεισμό τη βδομάδα.
- Ο χρήστης δεν μπορεί να κάνει δανεισμό, αν έχει αργήσει να επιστρέψει ένα τίτλο ή δεν τον έχει επιστρέψει καθόλου (έχοντας περάσει το όριο δανεισμού)
- Ο χρήστης δεν μπορεί να κάνει δανεισμό για ένα τίτλο, εάν δεν υπάρχει διαθέσιμο αντίτυπο.

```
DELIMITER $
CREATE TRIGGER chk_borrowings BEFORE INSERT ON borrowings
FOR EACH ROW
BEGIN
  IF (new.school_users_id = (SELECT bor.school_users_id from borrowings bor INNER JOIN
school_users s ON s.school_users_id = bor.school_users_id INNER JOIN books b ON
b.ISBN = bor.ISBN
WHERE s.status = "student" and s.school_users_id=new.school_users_id AND
DATEDIFF(CURDATE(),bor.borrowing_date) <7 GROUP BY bor.school_users_id HAVING
COUNT(*)= 2) ) THEN
    SIGNAL SQLSTATE '45000'
    SET MESSAGE_TEXT = 'check constraint on borrowings failed - A student
can only borrow 2 books a week.';
  END IF;
  IF (new.school_users_id = (SELECT bor.school_users_id from borrowings bor INNER JOIN
school_users s ON s.school_users_id = bor.school_users_id INNER JOIN books b ON
b.ISBN = bor.ISBN
WHERE s.status = "teacher" and s.school_users_id=new.school_users_id AND
DATEDIFF(CURDATE(),bor.borrowing_date) <7 GROUP BY bor.school_users_id HAVING
COUNT(*)= 1 ) ) THEN
    SIGNAL SQLSTATE '45000'
    SET MESSAGE_TEXT = 'check constraint on borrowings failed - A teacher
can only borrow 1 book a week.';
  END IF;
  IF (new.school_users_id = (SELECT school_users_id from borrowings WHERE
new.school_users_id =school_users_id and (return_date is null OR
DATEDIFF(CURDATE(),due_date) > 7)) ) THEN
    SIGNAL SQLSTATE '45000'
    SET MESSAGE_TEXT = 'check constraint on borrowings failed - A user
cannot borrow a book if returns are delayed or still open.';
  END IF;
  IF (new.ISBN = (SELECT ISBN FROM books WHERE avail_copies = 0 AND ISBN = new.ISBN))
  THEN
    SIGNAL SQLSTATE '45000'
    SET MESSAGE_TEXT = 'check constraint on borrowings failed - A user
cannot borrow a book if there is no available copie of book.';
  END IF;
END $
DELIMITER ;
```

Trigger, η due_date, έχοντας null τιμή αρχικά, να της δίνεται τιμή μια βδομάδα μετά την ημερομηνία του δανεισμού.

```
DELIMITER $
CREATE TRIGGER chk_due_date BEFORE INSERT ON borrowings
FOR EACH ROW
BEGIN
    IF (new.return_date IS NULL) THEN
        SET NEW.due_date = date_add(new.borrowing_date, INTERVAL 7 DAY);
    END IF;
END $
DELIMITER ;
```

DDL και DML scripts :

Τα δύο είδη scripts βρίσκονται στο github repository στον φάκελο «DDL_DML_scripts».

DDL Scripts:

Στο (git repository) έχουμε το εξής DDL script σε μορφή sql αρχείου:

DDL_script.sql

Δημιουργεί στη βάση όλους τους πίνακες που έχουμε ορίσει. Στο DDL script δημιουργούνται όλες οι εξαρτήσεις primary key, foreign keys, τα check constraints και τα triggers της βάσης.

Ως παράδειγμα εμφανίζουμε τη δημιουργία του πίνακα για την οντότητα school_users:

```
CREATE TABLE IF NOT EXISTS school_users (
    school_users_id INT UNSIGNED NOT NULL,
    school_id INT UNSIGNED NOT NULL,
    operator_id INT UNSIGNED NOT NULL,
    age DATE NOT NULL,
    status VARCHAR(15),
    PRIMARY KEY (school_users_id),
    user_type ENUM('school_users') NOT NULL REFERENCES users(user_type),
    CHECK(status IN ('student', 'teacher')),
    CONSTRAINT fk_users_school_users
    FOREIGN KEY (school_users_id) REFERENCES users (user_id)
        ON DELETE RESTRICT
    ON UPDATE CASCADE,
    CONSTRAINT fk_school_school_users
    FOREIGN KEY (school_id) REFERENCES school (school_id)
        ON DELETE RESTRICT
    ON UPDATE CASCADE,
    CONSTRAINT fk_operator_school_users
    FOREIGN KEY (operator_id) REFERENCES operator (operator_id)
```

```
        ON DELETE RESTRICT
        ON UPDATE CASCADE
    )ENGINE = InnoDB;
```

DML Scripts:

Στα DML scripts ανήκουν οι εντολές insert που εισάγουν τα δεδομένα μας στην βάση, όπως και τα queries τα οποία χρησιμοποιήθηκαν για την ορθή λειτουργία του CRUD στο UI της βάσης.

Το αρχείο με τα insert data βρίσκεται στο github repository, με την ονομασία:

DML_script.sql

Επίσης, η εισαγωγή (γεννήτρια) των δεδομένων έγινε μέσω της βιβλιοθήκης faker της python. Ο κώδικας που χρησιμοποιήθηκε για την υλοποίηση του DML script για τα insert data βρίσκεται και αυτός στο git repo με ονομασία fake_data.py.

Queries

3.1. (Διαχειριστής)

3.1.1. Παρουσίαση λίστας με συνολικό αριθμό δανεισμών ανά σχολείο(Κριτήρια αναζήτησης: έτος, ημερολογιακός μήνας). (Οι μεταβλητές {year},{month} παίρνουν την τιμή τους κατ' επιλογήν του χρήστη μέσω της Python. Βρίσκονται σε αυτή τη μορφή, ώστε να πληρούνται τα κριτήρια αναζήτησης).

```
select sch.name as school, count(borrowed_id) as count from
borrowings bor
INNER JOIN operator op ON bor.operator_id = op.operator_id
INNER JOIN school sch ON sch.operator_id = op.operator_id
WHERE MONTH(borrowing_date) = '{month}' AND
YEAR(borrowing_date) = '{year}'
GROUP BY sch.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;

Για τους συγγραφείς:

```
SELECT ct.category, author FROM books b
      INNER JOIN category_table ct ON b.ISBN = ct.ISBN
      LEFT JOIN author_table aut ON ct.ISBN = aut.ISBN
      WHERE category='{category}';
```

Για τους εκπαιδευτικούς:

```
SELECT category, us.First_name, us.Last_name FROM books b
      INNER JOIN category_table ct ON b.ISBN = ct.ISBN
      INNER JOIN borrowings bor ON bor.ISBN = ct.ISBN
      INNER JOIN school_users schu ON bor.school_users_id
= schu.school_users_id
      INNER JOIN users us ON us.user_id =
schu.school_users_id
      WHERE schu.status = "teacher" AND borrowing_date >
DATE_ADD(curdate(),interval -1 year)
      AND category='{category}';
```

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

```
SELECT subquery.First_name, subquery.Last_name, MAX(subquery.borrow_count)
AS max_borrow_count
      FROM (
      SELECT schu.school_users_id, us.First_name, us.Last_name,
COUNT(bor.borrowed_id) AS borrow_count
      FROM books b
      INNER JOIN borrowings bor ON bor.ISBN = b.ISBN
      INNER JOIN school_users schu ON bor.school_users_id =
schu.school_users_id
      INNER JOIN users us ON us.user_id = schu.school_users_id
      WHERE YEAR(CURDATE()) - DATE_FORMAT(schu.age, "%Y") < 40
      AND schu.status = "teacher"
      GROUP BY schu.school_users_id, us.First_name, us.Last_name
      ) AS subquery
      GROUP BY subquery.First_name, subquery.Last_name;
```

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

```
select distinct aut.author from author_table aut
      LEFT JOIN borrowings bor ON bor.ISBN = aut.ISBN
      WHERE not exists (select 1 from borrowings inner join
author_table aut1 on bor.ISBN = aut1.ISBN );
```


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

```
select bor.operator_id, us.First_name, us.Last_name, count(bor.borrowed_id)
as count from users us
        inner join operator op on op.operator_id = us.user_id
        inner join borrowings bor on bor.operator_id =
op.operator_id
        inner join operator op1 on op1.operator_id =
op.operator_id WHERE YEAR(bor.borrowing_date) IN (
        SELECT DISTINCT YEAR(borrowing_date)
        FROM borrowings
)
group by bor.operator_id, us.First_name, us.Last_name
having count(bor.borrowed_id) > 20 ;
```

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 ct1.category as cat1, ct2.category as cat2, count(bor.borrowed_id) as
count from borrowings bor
        inner join books b on b.ISBN = bor.ISBN
        inner join category_table ct1 on ct1.ISBN = b.ISBN
        cross join category_table ct2 on ct1.category <>
ct2.category AND ct1.category < ct2.category AND ct1.ISBN = ct2.ISBN
group by ct1.category, ct2.category
ORDER BY
COUNT(bor.borrowed_id) DESC
limit 3;
```

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

```
with aut_five_less_max (author, count_of_books_per_author) as
(select aut.author, count(b.ISBN) as
count_of_books_per_author from books b
        inner join author_table aut on aut.ISBN = b.ISBN
        group by aut.author),
most_books_author (max_books_author) as
(select max(count_of_books_per_author) as max_books_author
from aut_five_less_max)
select aflm.author, aflm.count_of_books_per_author
from aut_five_less_max aflm
join most_books_author mba
on mba.max_books_author - 5 > aflm.count_of_books_per_author;
```

3.2. (Χειριστής)

3.2.1. Παρουσίαση όλων των βιβλίων κατά Τίτλο, Συγγραφέα (Κριτήρια αναζήτησης: τίτλος κατηγορία/συγγραφέας/αντίτυπα).

```
SELECT b.isbn,b.title,b.publisher, b.num_of_pages, b.avail_copies,
b.language, a.author, c.category FROM books b inner join author_table a on
a.ISBN = b.ISBN inner join school sch on sch.school_id = b.school_id
inner join category_table c on c.ISBN= b.ISBN
WHERE title='{title}' AND category='{category}' AND
author='{author}' AND avail_copies = {avail_copies} AND b.operator_id
={session.get('user_id')}
```

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 Καθυστέρησης).

```
SELECT * FROM borrowings b INNER JOIN users u on u.user_id
=b.school_users_id where return_date is NULL AND
b.operator_id={operator_id}
AND u.First_name='{FirstName}' and u.Last_name='{LastName}'
AND DATEDIFF(curdate(),b.due_date) = {late_days}
```

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

```
select avg(rating_score) as score,r.school_users_id,category from ratings r
INNER JOIN category_table c on c.Isbn = r.ISBN inner join borrowings bor
on bor.school_users_id = r.school_users_id inner join books
b on b.ISBN=r.ISBN inner join operator op on op.operator_id =b.operator_id
WHERE
r.school_users_id = {sch_user_id} and c.category =
'{category}' and op.operator_id = {session.get('user_id')} group by
r.school_users_id , c.category;
```

3.3. (Απλός χρήστης)

3.3.1. Όλα τα βιβλία που έχουν καταχωριστεί (Κριτήρια αναζήτησης: τίτλος/κατηγορία/ συγγραφέας), δυνατότητα επιλογής βιβλίου και δημιουργία αιτήματος κράτησης.

```

SELECT b.isbn,b.title,b.publisher, b.num_of_pages, b.avail_copies,
b.language, a.author, c.category FROM books b inner join author_table a on
a.ISBN = b.ISBN inner join school sch on sch.school_id = b.school_id
inner join category_table c on c.ISBN= b.ISBN
WHERE title='{title}' AND category='{category}' AND
author='{author}'

```

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

```

SELECT * FROM borrowings WHERE
school_users_id={session.get('user_id')}

```

View

Στο DDL script, επίσης, προσθέσαμε και ένα view, το οποίο έχει ως στόχο να αποτρέπει τον διαχειριστή από το να βλέπει τα ευαίσθητα στοιχεία των χρηστών.

```

create view user_security as
select
u.user_id,
concat(substr(email,1,2), '*****', substr(email, -4)) email,
concat('*****') password,
concat('*****') username,
concat(substr(phone_number,1,2), '*****', substr(phone_number, -2))
phone_number
from users u inner join email_table e on e.user_id=u.user_id inner
join phone_table p on p.user_id = u.user_id

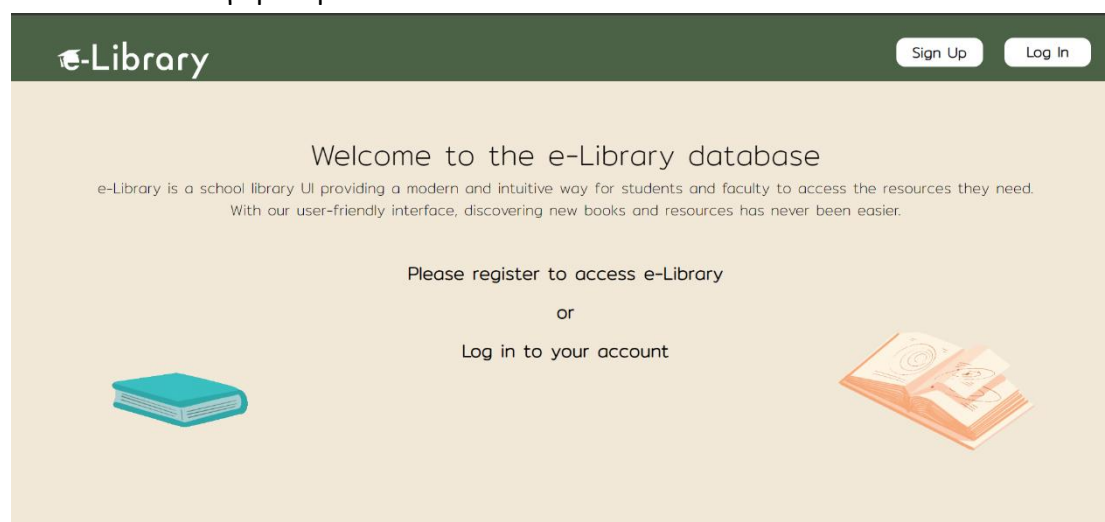
```

Indexing

Αρχικά, σημειώνουμε ότι στην MySQL, στις κολώνες primary keys δημιουργούνται αυτόματα ευρετήρια, το οποίο είναι πολύ σημαντικό, αφού τα primary keys αποτελούν συνήθη τρόπο εύρεσης queries, (όπως και συνδεσιμότητα tables, μέσω της JOIN), αλλά και σε triggers για να εισαχθούν περιορισμοί. Επίσης, ευρετήρια μπαίνουν αυτόματα και σε primary keys με παραπάνω από ένα attribute. Ακόμα, δημιουργήθηκαν ευρετήρια σε attributes, τα οποία χρησιμοποιήθηκαν πολύ για τα queries που ζητήθηκαν στην εργασία. Τέτοια είναι τα First_name, Last_name της οντότητας users και το category, της οντότητας category_table (όπως και το email ή το approved, αλλά αυτά χρησιμοποιήθηκαν για queries για την λειτουργία της ιστοσελίδας ανεξάρτητα των ερωτήσεων της εργασίας). Παρόλο που τα category και email, είναι attributes που ανήκουν σε composite primary keys, τους βάλαμε ευρετήρια, επειδή στα queries χρησιμοποιούνται μόνο τους. Ευρετήρια δημιουργήθηκαν και για τα dates στις οντότητες borrowings και reservations, επειδή χρησιμοποιήθηκαν πολύ για τους περιορισμούς στα triggers. Τέλος, τα indexes υπάρχουν στο file DDL_script.sql στο github repository.

User Manual

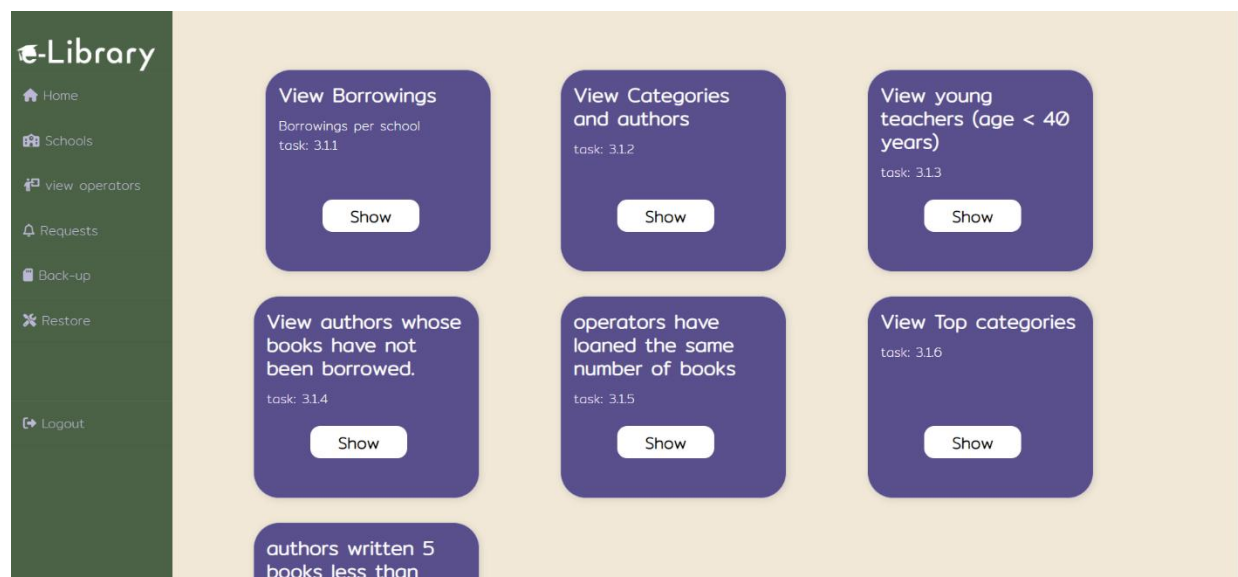
Κάθε επισκέπτης στην ιστοσελίδα έχει την επιλογή να κάνει εγγραφή σε αυτήν ή να συνδεθεί στο λογαριασμό του :



Στην sign up επιλέγει σαν τι χρήστης θα χρησιμοποιεί την εφαρμογή δηλαδή σαν μέλος της σχολικής μονάδας ή σαν χειριστής σε επίπεδο σχολικής μονάδας και υπάρχει κατάλληλη φόρμα στην οποία θα εισάγει τα απαραίτητα στοιχεία. Όσον αφορά τον χειριστή θα πρέπει να εισάγει και όλα τα στοιχεία του σχολείου του.

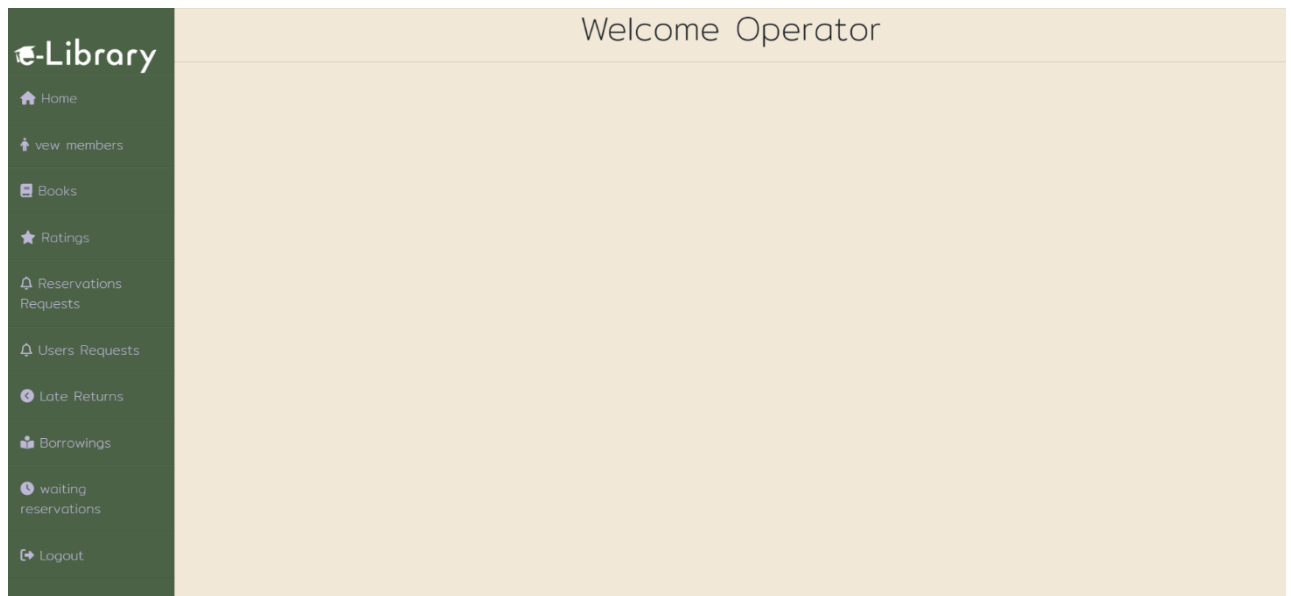
Κεντρικός διαχειριστής σε επίπεδο Δικτύου Σχολικών Βιβλιοθηκών

Ο κεντρικός διαχειριστής κάνει εισαγωγή στην βάση με τα στοιχεία που του έχουν δοθεί και βλέπει την συγκεκριμένη ιστοσελίδα:



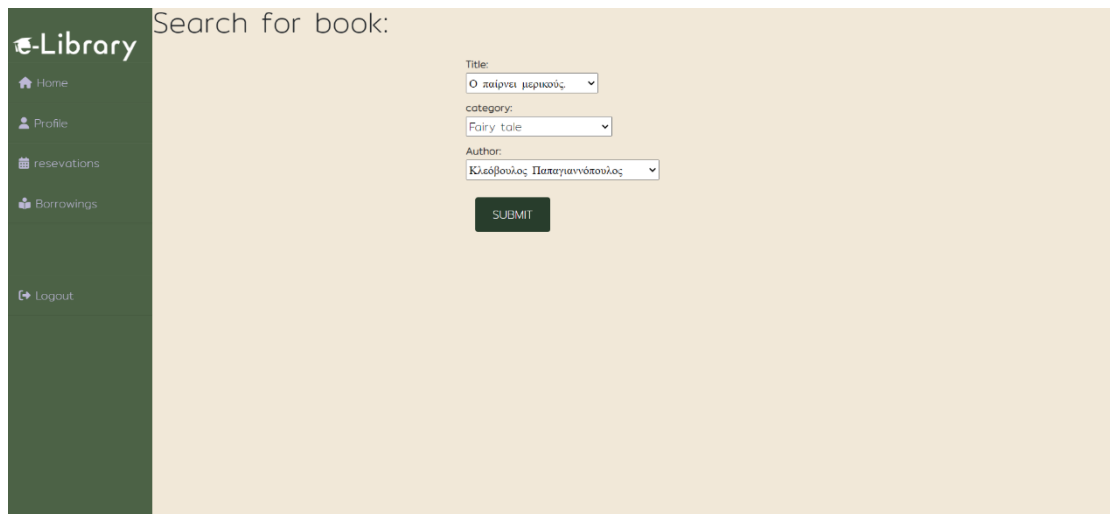
Στο Home θα μπορεί να δει και να εκτελέσει τα ερωτήματα της εργασίας, στο Schools θα μπορεί να τροποποιήσει και να διαγράψει τα σχολεία που είναι εγγεγραμμένα στην ιστοσελίδα μας, των οποίων οι χειριστές έχουν εγκριθεί από τον διαχειριστή αλλά και αυτά που δεν έχουν εγκριθεί ακόμα οι χειριστές τους. Για να δει τα αιτήματα των χειριστών που θέλουν να εισέλθουν στην βάση θα μεταβαίνουν στα Requests και θα εγκρίνουν ή θα απορρίπτουν τους χειριστές. Σε περίπτωση που δεν εγκριθούν τότε το σχολείο και το οποίο δήλωσαν θα διαγράφεται αυτόματα. Η παραδοχή που κάναμε με την καταχώριση των σχολείων από τον διαχειριστή είναι να τις εισάγει ο χειριστής του σχολείου και ο διαχειριστής μέσω της έγκρισης του χειριστή και της δυνατότητας τροποποίησης των στοιχείων που εισάγει ο εκάστοτε χειριστής για το σχολείο του. Τέλος από τα view operators μπορεί να δει τα στοιχεία τους και να τους διαγράψει από την βάση (βεβαιωθείτε ότι έχετε τρέξει το view user_security για να εμφανιστούν), ενώ με τα κουμπιά backup και restore δημιουργεί backup για την βάση και κάνει restore την βάση.

Χειριστής



Εδώ ο χειριστής στο view members θα βλέπει τα μέλη του σχολείου του που είναι εγγεγραμμένα στην ιστοσελίδα μας τα οποία θα μπορεί να διαγράφει μόνιμα από την βάση και να απενεργοποιεί τους λογαριασμούς τους. Στα books μπορεί να βλέπει να τροποποιεί, να εισάγει καινούρια βιβλία και να καταχωρεί δανεισμό βιβλίου εάν τηρούνται οι προϋποθέσεις. Στα ratings μπορεί να εγκρίνει σχόλια μαθητών και να βρίσκει μέσο όρο αξιολόγησης ανά δανειζόμενο. Στα reservation requests θα εγκρίνει, απορρίπτει ή θα βάζει σε αναμονή τις κρατήσεις, όπου σε περίπτωση αναμονής, η κράτηση θα εμφανίζεται στον τομέα waiting reservations και όποτε γίνει available το book θα εγκρίνει τον δανεισμό του και θα διαγράφεται η κράτηση αυτόματα από το waiting reservations. Στο Users requests θα εμφανίζονται οι αιτήσεις εγγραφής των χρηστών. Στο late returns θα μπορεί να αναζητήσει χρήστες που έχουν καθυστερήσει την επιστροφή βιβλίου και το έχουν στην κατοχή τους. Τέλος στα borrowings θα μπορεί να δει τους δανεισμούς που έχει καταχωρίσει και να δηλώσει την επιστροφή του βιβλίου.

Χρήστης



The screenshot displays the 'e-Library' web application interface. On the left is a dark green sidebar with navigation links: Home, Profile, reservations, Borrowings, and Logout. The main content area has a light beige background and is titled 'Search for book:'. It contains three dropdown menus for searching: 'Title' (set to 'Ο παίρνει μερικούς'), 'category' (set to 'Fairy tale'), and 'Author' (set to 'Κλεόβουλος Παπαγιαννόπουλος'). A dark green 'SUBMIT' button is positioned below these fields.

Στην Home κάθε χρήστης θα μπορεί να κάνει αναζήτηση βιβλίου να το δει και να το αξιολογήσει καθώς και να το κάνει κράτηση εάν τηρεί τις προϋποθέσεις. Στο Profile εάν είναι μαθητής μπορεί να δει τα προσωπικά του στοιχεία, ενώ εάν είναι καθηγητής μπορεί και να τα τροποποιήσει. Στα reservations θα βλέπει τις αιτήσεις για δανεισμούς που έχει κάνει μέσα σε μία εβδομάδα ενώ στα borrowings θα μπορεί να δει τους δανεισμούς που έχει πραγματοποιήσει.

Οδηγίες Εγκατάστασης της Εφαρμογής

Το repository της βάσεις δεδομένων

<https://github.com/pervolarakis2001/Library-database/tree/main>

Βήμα 1^ο – Κατέβασμα του repository Μέσω git

Για την εγκατάσταση μεταβείτε στον παραπάνω link. Πρέπει πρώτα να γίνει clone τοπικά της εφαρμογής. Αυτό μπορεί να γίνει μέσω του GitHub desktop, είτε μέσω terminal με την εντολή git clone <https://github.com/pervolarakis2001/Library-database/tree/main> στο τοπικό directory που επιθυμούμε να εγκαταστήσουμε την εφαρμογή.

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

Για την εγκατάσταση της βάσης στον υπολογιστή μας χρειαζόμαστε έναν SQL Server (συγκεκριμένα χρησιμοποιήσαμε mysql μέσω Mysql Workbench) και ένα DBMS (εδώ χρησιμοποιήσαμε Mysql Workbench). Για την εγκατάσταση της βάσης αρκεί να δημιουργήσουμε μία σύνδεση με mysql Server και να τρέξουμε τα scripts DDL_script.sql, DML_script.sql που βρίσκονται στον φάκελο DDL_DML_scripts, με αυτήν την σειρά.

Βήμα 3ο - Launch της εφαρμογής μέσω local host

Πρώτα εγκαταστήστε όλες τις απαραίτητες βιβλιοθήκες που αναφέρονται στο αρχείο requirements.txt μέσω της εντολής pip install -r requirements.txt . Μεταβείτε στην __init__.py του φακέλου Library αλλάξτε τον κωδικό και εισάγεται τον δικό σας και ότι άλλη πληροφορία διαφέρει από το δικό σας πρόγραμμα. Τέλος μεταβείτε στην main.py του ίδιου φακέλου και τρέξτε την. Στο τερματικό θα εμφανιστεί ένα link της μορφής <http://localhost:5000>. Κάντε το copy + paste σε έναν browser και η εφαρμογή θα είναι έτοιμη για χρήση.