Εθνικό Μετσόβιο Πολυτεχνείο Σχολή Ηλεκτρολόγων Μηχανικών και Μηχανικών Υπολογιστών Βασεις Δεδομενων

ΕΞΑΜΗΝΙΑΙΑ ΕΡΓΑΣΙΑ 2022-23

ΟΜΑΔΑ 140 :

Δήμητρα Τσέγκου ΑΜ: el20633

Αποστολία Χρυσοβαλανοτού Σκέντζου ΑΜ:el20054

Ηλίας Νικιλετοπούλος ΑΜ:el16611

GITHUB: https://github.com/ilianor98/vaseis-project?fbclid=IwAR0tlfOTVDY-H8m4vauRRWAcFjtrhrBR85EYodmSqnSSkpzELePnTx4-3UA

ΠΕΡΙΕΧΟΜΕΝΑ:

1) ER ENTITY RELEATIONSHIP DIAGRAM 3

2) ΣΧΕΣΙΑΚΟ ΔΙΑΓΡΑΜΜΑ 4

3)Το σχεσιακό μοντέλο της ΒΔ 5

Α) Όλοι οι περιορισμούς που έχουν οριστεί, ανά κατηγορία, με σύντομη αιτιολόγηση για την επιλογή

Περιορισμοί ακεραιότητας 8

Περιορισμοί αναφορικής ακεραιότητας 9

Ακεραιότητα Πεδίου Τιμών 10

Περιορισμοί οριζόμενοι για τον χρήστη: 11

Β) Ευρετήρια που έχουν ορισθεί 11

Γ) Το σύστημα & οι γλώσσες προγραμματισμού που χρησιμοποιήθηκαν 11

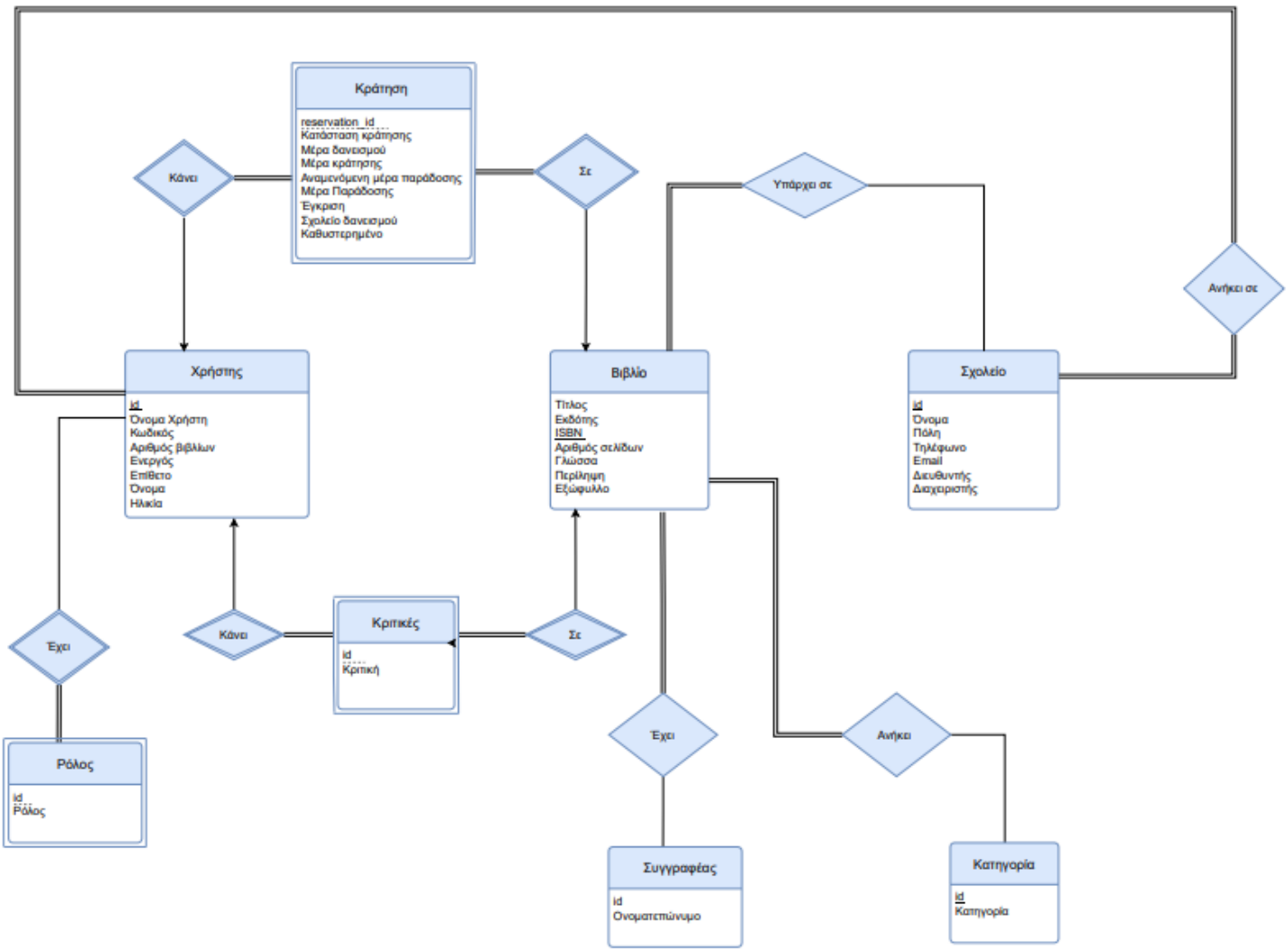
Δ) Βήματα για εγκατάσταση εφαρμοργης 11

4)SQL κώδικας

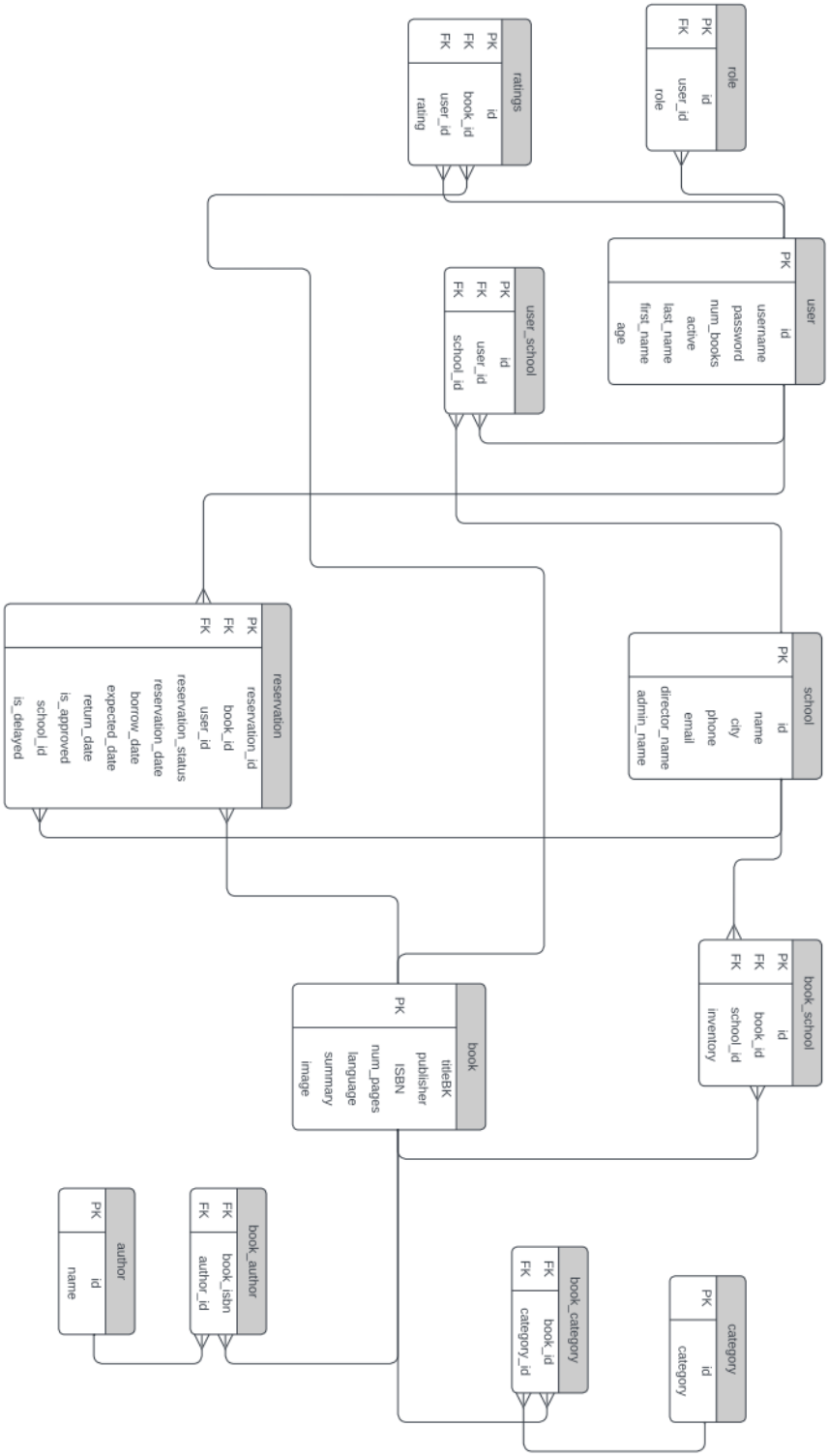
DDL & DML SCRIPT 12

QUERIES 16

ΔΙΑΓΡΑΜΜΑ ER (github arxeio er)



ΣΧΕΣΙΑΚΟ ΔΙΑΓΡΑΜΜΑ (github Database-Relational-Model)



Σχολιασμός Σχεσιακού Διαγράμματος

Στην βάση μας περιέχονται 3 βασικές οντότητες:

- → ο χρήστης (table =”user”),
- → το σχολείο (Table = “school”), το βιβλίο (Table = “book”) ,
- → ο συγγραφέας, (Table = “author”),
- → η κατηγορία(Table = “category”)

και άλλες οντότητες που σχετίζονται με αυτές,

- → ο ρόλος (Table = “role”),
- → οι κριτικές (Table = “ratings”) ,
- → οι κρατήσεις (Table = “reservations”) ,
- → ο συγγραφέας_βιβλίο (Table = “book_author”) ,
- → η κατηγορία_βιβλίο (Table = “book_category”)
- → ο χρήστης_σχολείο (Table = “user_school”).

Επιπλέον οι σχέσεις «κάνει» μεταξύ των οντοτήτων κράτηση και χρήστης, «σε» μεταξύ των οντοτήτων κράτηση και βιβλίο, «κάνει» μεταξύ των οντοτήτων χρήστης και κριτικές και σχέση «σε» μεταξύ οντοτήτων κριτικές και βιβλίο έχουν υλοποιηθεί με προσθήκη επιπρόσθετων attributes στους κατάλληλους πίνακες.

Το σχεσιακό μοντέλο της ΒΔ:

1. **user**
   - → id
   - → username
   - → password
   - → num_books
   - → active
   - → last_name
   - → first_name
   - → age
2. **school**
   - → id
   - → name
   - → city
   - → phone
   - → email
   - → director_name
   - → admin_name
3. **role**
   - → id
   - → user_id
   - → role
4. **reservation**
   - → reservation_id
   - → book_id
   - → user_id
   - → reservation_status
   - → reservation_date
   - → borrow_date
   - → expected_date
   - → return_date
   - → is_approved
   - → school_id
   - → is_delayed
5. **ratings**
   - → id
   - → book_id
   - → user_id
   - → rating

6. **category**
   - → id
   - → category
7. **user_school**
   - → id
   - → user_id
   - → school_id
8. **book_school**
   - → id
   - → book_id
   - → school_id
   - → inventory
9. **book_category**
   - → book_id
   - → category_id
10. **book_author**
   - → book_isbn
   - → author_id
11. **book**
   - → titleBK
   - → publisher
   - → ISBN
   - → num_pages
   - → summary
   - → image
   - → language
12. **author**
   - → id
   - → name

Α) Περιορισμοι που έχουν οριστεί, ανά κατηγορία, με σύντομη αιτιολόγηση για την επιλογή:

- **Περιορισμοί ακεραιότητας:**

  i. Όλα τα INTEGER πρωτεύοντα κλειδιά θέτονται αυτόματα με AUTO_INCREMENT οπότε δεν είναι NULL.

  ii. Επίσης δεν επιτρέπετε να είναι NULL τα εξής attributes:

  → user
  (username,password,num_books,active,last_name,first_name,age)

  → school
  (name,city,phone,email,director_name ,admin_name)

  → role
  (user_id,role)

  → reservation
  (book_id,user_id,reservation_status,school_id,)

  → ratings
  (book_id,user_id,rating)

  → category
  (category)

  → user_school
  (user_id,school_id)

  → book_school
  (book_id,school_id,inventory)

  → book_category
  (category_id,)

  → book_author
  (book_isbn)

  → book
  (titleBK,num_pages,language)

  → author
  (name)

❖ **Περιορισμοί αναφορικής ακεραιότητας:**

Εξωτερικά κλειδιά:

→ book_author
   i. FOREIGN KEY (`book_isbn`) REFERENCES `book` (`ISBN`)
   ii. FOREIGN KEY (`author_id`) REFERENCES `author` (`id`)

→ book_category
   i. FOREIGN KEY (`category_id`) REFERENCES `category` (`id`)
   ii. FOREIGN KEY (`book_id`) REFERENCES `book` (`ISBN`)

→ book_school
   i. FOREIGN KEY (`book_id`) REFERENCES `book` (`ISBN`)
   ii. FOREIGN KEY (`school_id`) REFERENCES `school` (`id`)

→ ratings
   i. FOREIGN KEY (`book_id`) REFERENCES `book` (`ISBN`)
   ii. FOREIGN KEY (`user_id`) REFERENCES `user` (`id`)

→ reservation
   i. FOREIGN KEY (`book_id`) REFERENCES `book` (`ISBN`)
   ii. FOREIGN KEY (`user_id`) REFERENCES `user` (`id`)

→ role
   i. FOREIGN KEY (`user_id`) REFERENCES `user` (`id`)

→ user_school
   i. FOREIGN KEY (`user_id`) REFERENCES `user` (`id`)
   ii. FOREIGN KEY (`school_id`) REFERENCES `school` (`id`)

- **Ακεραιότητα Πεδίου Τιμών:**
  i. INTEGER:
    - → id
    - → ISBN
    - → num_pages
    - → book_isbn
    - → author_id
    - → book_id
    - → category_id
    - → school_id
    - → inventory DEFAULT 0
    - → user_id
    - → rating
    - → reservation_id
    - → role DEFAULT 0
    - → phone
    - → num_books DEFAULT 0
    - → age

  ii. TINYINT:
    - → reservation_status DEFAULT 0
    - → is_approved DEFAULT 0
    - → is_delayed DEFAULT 0
    - → active DEFAULT 1

  iii. VARCHAR:
    - → name (50)
    - → titleBK (150)
    - → publicher (150)
    - → language (50)
    - → summary (255)
    - → image (255)
    - → category (50)
    - → username (20)
    - → password (10)
    - → last_name (50)
    - → first_name (50)

  iv. DATETIME:
    - → reservation_date
    - → borrow_date
    - → expected_date
    - → return_date

- **<u>Περιορισμοί οριζόμενοι για τον χρήστη:</u>**

  i. Δύο κρατήσεις ανά εβδομάδα για τους μαθητές. Δεν επιτρέπεται η κράτηση αν εκκρεμεί επιστροφή που καθυστέρησε ή κράτηση τίτλου που έχει εν ισχύ δανεισμό από τον ίδιο χρήστη. (`is_approved` DEFAULT 0, `is_delayed` tinyint(1) DEFAULT 0)
  ii. Για τις αξιολογήσεις βιβλίων ισχύει (`rating` >= 1 and `rating` <= 5)

2) Ευρετήρια που έχουν ορισθεί:

Δεν έχει γινει καποια χρηση ευρετηριων στη βαση μας.

Το σύστημα και οι γλώσσες προγραμματισμού που χρησιμοποιήθηκαν για την ανάπτυξη της εφαρμογής

Χρησημοποιήθηκαν η MySQL(MariaDB) ως DBMS ως το back-end του server.

Ακόμα, οι γλώσσες που χρησημοποιήθηκαν είναι οι εξής:

• Html: για το front-end σε συνδυασμό με CSS για styling

• Python: για την δημιουργία του back -end σε συνδυασμό με Flask και χρήση βιβλιοθήκης pymysql

Οδηγίες Εγκατάστασης

Το repository της βάσης δεδομένων στο github:

Οδηγίες Εγκατάστασης

Τα βήματα εγκατάστασης για την εφαρμογή μας είναι:

Για την εγκατάσταση της εφαρμογής πρέπει πρώτα να γίνει εγκατάσταση του xampp και σε αυτό ενεργοποίηση του apache και του mysql. Έπειτα, εγκατήσετε στη συσκευή σας το σχήμα της βάσης sql το οποίο θα βρείτε στο link του GitHub που παραδόθηκε παραπάνω “ilianor98/vaseis-project”. Σε αυτό το αρχείο θα κατεβάσετε τα παρακάτω αρχεία με την εξής σειρά:

- “create_schema.sql” : για την δημιουργία του σχήματος της βάσεως δεδομένων
- “procedures.sql “ : για την προθήκη των απαραίτητων procedures
- “insert_schema.sql” : για την εισαγωγή των mock data

Ύστερα, αφού κατεβάσετε τα αρχεία αυτά στην συσκευή σας (πχ σε έναν φάκελο temp), θα μπείτε στο σύστημα διαχείρισης του sql που έχετε και θα κάνετε source τα αρχεία με την ίδια σειρά.

Source C:/path/to (αρχεία)

Ανοίξτε το app.py:

```
config = {
    "user": "*****",
    "password": "****",
    "host": "localhost",
    "database": "vas",
    "charset": "utf8mb4",
    "cursorclass": pymysql.cursors.DictCursor,
}
```

Θα πρέπει να βάλετε στην περιοχή με τα αστεράκια user & password της SQL καλυτέρα χρήση του user root
Επιπρόσθετα θα πρέπει να γίνει αλλαγή στο directory στο τερμιναλ οπού κάνατε save το αρχείο του προτζεκτ.

Έπειτα στο τερμιναλ και γραψτε:

- pip install -r requirements.txt
  έπειτα στο τερμιναλ θα πρέπει να τρέξετε και το command:
- set FLASK_APP=app.py
- flask run

Ανοίξτε browser και πήγαινε στο localhost:5000/ για να δείτε την σελίδα μας.
Στην σελίδα μας μπορείτε να βάλετε για login:

- teststudent 1234512345 (student)
- agarnsworth1w 12345678 (admin)
- iliasnik password(master admin)

SQL κώδικας

DDL Script φακελος github(create_schema)

```sql
DROP DATABASE IF EXISTS vas1;
CREATE DATABASE vas1;
USE vas1;

CREATE TABLE `user` (
  `id` int(11) NOT NULL AUTO_INCREMENT,
  `username` varchar(20) NOT NULL,
  `password` varchar(10) NOT NULL,
  `num_books` int(11) NOT NULL DEFAULT 0,
  `active` tinyint(1) NOT NULL DEFAULT 1,
  `last_name` varchar(50) NOT NULL,
  `first_name` varchar(50) NOT NULL,
  `age` int(3) NOT NULL,
  PRIMARY KEY (`id`)
);

CREATE TABLE `role` (
  `id` int(11) NOT NULL AUTO_INCREMENT,
  `user_id` int(11) NOT NULL,
  `role` int(1) NOT NULL DEFAULT 0,
  PRIMARY KEY (`id`),
  UNIQUE KEY `user_id` (`user_id`),
  CONSTRAINT `role_ibfk_1` FOREIGN KEY (`user_id`) REFERENCES `user` (`id`)
);
```

```sql
CREATE TABLE `book` (
  `titleBK` varchar(150) NOT NULL,
  `publisher` varchar(150) DEFAULT NULL,
  `ISBN` int(11) NOT NULL,
  `num_pages` int(11) NOT NULL,
  `language` varchar(50) NOT NULL,
  `summary` varchar(255) DEFAULT NULL,
  `image` varchar(255) DEFAULT NULL,
  PRIMARY KEY (`ISBN`)
);

CREATE TABLE `school` (
  `id` int(11) NOT NULL AUTO_INCREMENT,
  `name` varchar(50) NOT NULL,
  `city` varchar(50) NOT NULL,
  `phone` int(11) NOT NULL,
  `email` varchar(50) NOT NULL,
  `director_name` varchar(50) NOT NULL,
  `admin_name` varchar(50) NOT NULL,
  PRIMARY KEY (`id`)
);

CREATE TABLE `user_school` (
  `id` int(11) NOT NULL AUTO_INCREMENT,
  `user_id` int(11) NOT NULL,
  `school_id` int(11) NOT NULL,
  PRIMARY KEY (`id`),
  KEY `user_id` (`user_id`),
  KEY `school_id` (`school_id`),
  CONSTRAINT `user_school_ibfk_1` FOREIGN KEY (`user_id`) REFERENCES `user` (`id`),
```

```
54      KEY `school_id` (`school_id`),
55      CONSTRAINT `user_school_ibfk_1` FOREIGN KEY (`user_id`) REFERENCES `user` (`id`),
56      CONSTRAINT `user_school_ibfk_2` FOREIGN KEY (`school_id`) REFERENCES `school` (`id`)
57    );
58
59    CREATE TABLE `book_school` (
60      `id` int(11) NOT NULL AUTO_INCREMENT,
61      `book_id` int(11) NOT NULL,
62      `school_id` int(11) NOT NULL,
63      `inventory` int(6) NOT NULL DEFAULT 0,
64      PRIMARY KEY (`id`),
65      KEY `book_id` (`book_id`),
66      KEY `school_id` (`school_id`),
67      CONSTRAINT `book_school_ibfk_1` FOREIGN KEY (`book_id`) REFERENCES `book` (`ISBN`),
68      CONSTRAINT `book_school_ibfk_2` FOREIGN KEY (`school_id`) REFERENCES `school` (`id`)
69    );
70
71    CREATE TABLE `author` (
72      `id` int(11) NOT NULL AUTO_INCREMENT,
73      `name` varchar(50) NOT NULL,
74      PRIMARY KEY (`id`)
75    );
76
77    CREATE TABLE `book_author` (
78      `book_isbn` int(11) NOT NULL,
79      `author_id` int(11) DEFAULT NULL,
80      KEY `book_isbn` (`book_isbn`),
81      KEY `author_id` (`author_id`),
82      CONSTRAINT `book_author_ibfk_1` FOREIGN KEY (`book_isbn`) REFERENCES `book` (`ISBN`),
83      CONSTRAINT `book_author_ibfk_2` FOREIGN KEY (`author_id`) REFERENCES `author` (`id`)
84    );
```

```
86    CREATE TABLE `category` (
87      `id` int(11) NOT NULL AUTO_INCREMENT,
88      `category` varchar(50) NOT NULL,
89      PRIMARY KEY (`id`)
90    );
91
92    CREATE TABLE `book_category` (
93      `book_id` int(11) NOT NULL,
94      `category_id` int(11) NOT NULL DEFAULT 0,
95      KEY `category_id` (`category_id`),
96      KEY `book_id` (`book_id`),
97      CONSTRAINT `book_category_ibfk_1` FOREIGN KEY (`category_id`) REFERENCES `category` (`id`),
98      CONSTRAINT `book_category_ibfk_2` FOREIGN KEY (`book_id`) REFERENCES `book` (`ISBN`)
99    );
100
101   CREATE TABLE `ratings` (
102     `id` int(11) NOT NULL AUTO_INCREMENT,
103     `book_id` int(11) NOT NULL,
104     `user_id` int(11) NOT NULL,
105     `rating` int(11) NOT NULL CHECK (`rating` >= 1 and `rating` <= 5),
106     PRIMARY KEY (`id`),
107     KEY `book_id` (`book_id`),
108     KEY `user_id` (`user_id`),
109     CONSTRAINT `ratings_ibfk_1` FOREIGN KEY (`book_id`) REFERENCES `book` (`ISBN`),
110     CONSTRAINT `ratings_ibfk_2` FOREIGN KEY (`user_id`) REFERENCES `user` (`id`)
111   );
112
113   CREATE TABLE `reservation` (
114     `reservation_id` int(11) NOT NULL AUTO_INCREMENT,
115     `book_id` int(11) NOT NULL,
116     `user_id` int(11) NOT NULL,
```

```
117     `reservation_status` tinyint(1) NOT NULL DEFAULT 0,
118     `reservation_date` datetime DEFAULT NULL,
119     `borrow_date` datetime DEFAULT NULL,
120     `expected_date` datetime DEFAULT NULL,
121     `return_date` datetime DEFAULT NULL,
122     `is_approved` tinyint(1) DEFAULT 0,
123     `school_id` int(11) NOT NULL,
124     `is_delayed` tinyint(1) DEFAULT 0,
125     PRIMARY KEY (`reservation_id`),
126     KEY `book_id` (`book_id`),
127     KEY `user_id` (`user_id`),
128     CONSTRAINT `reservation_ibfk_1` FOREIGN KEY (`book_id`) REFERENCES `book` (`ISBN`),
129     CONSTRAINT `reservation_ibfk_2` FOREIGN KEY (`user_id`) REFERENCES `user` (`id`)
130   );
131
```

Δημιουργεί στη Βάση μας όλους του πίνακες που έχουν ορισθεί καθώς δημιουργούνται και τα primary keys , τα check constrains , και τα triggers της βάσης.

DML SCRIPT

(ΦΑΚΕΛΟΣ GITHUB insert_schema.sql) & (ΦΑΚΕΛΟΣ queries.sql) & (ΦΑΚΕΛΟΣ procedures.sql)

Στην κατηγορία DML script περιέχει όλο το insert script δεδομένα που θα εισάγουμε στην βάση μας και στο φάκελο queries έχουμε τα υλοποίηση τα queries που ζητούνται στην εκφώνηση. Παράλληλα στον φάκελο procedures.sql έχουμε υλοποίηση διαδικασίες για την υλοποίηση των λειτουργιών που μας ζητούνται.

QUERIES:

(ΦΑΚΕΛΟΣ github queries.sql)

Υλοποίηση των queries της εκφώνησης:

Διαχειριστής

3.1.1.Παρουσίαση λίστας με συνολικό αριθμό δανεισμών ανά σχολείο (Κριτήρια αναζήτησης: έτος, ημερολογιακός μήνας πχ Ιανουάριος)

```
SELECT YEAR(reservation_date) AS year,
       MONTH(reservation_date) AS month,
       school_id,
       COUNT(*) AS count
FROM reservation
WHERE YEAR(reservation_date) = 2023 AND MONTH(reservation_date) = 6
GROUP BY YEAR(reservation_date), MONTH(reservation_date), school_id
ORDER BY YEAR(reservation_date), MONTH(reservation_date), school_id;
```

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;

```
select c.category , a.name, u.first_name, u.last_name
from user u
inner join role l on u.id = l.user_id
inner join reservation r on u.id = r.user_id
inner join book_author ba on r.book_id = ba.book_isbn
inner join author a on a.id = ba.author_id
inner join book_category bc on bc.book_id = r.book_id
inner join category c on bc.category_id = c.id
where l.role = 2 and c.id = 2;
```

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

```
SELECT u.first_name, u.last_name, r.user_id, COUNT(r.user_id) AS count
FROM user u
INNER JOIN reservation r ON u.id = r.user_id
INNER JOIN role l ON u.id = l.user_id
WHERE l.role = 2
GROUP BY r.user_id;
```

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

```
SELECT a.name
FROM author a
WHERE NOT EXISTS (
  SELECT 1
  FROM book_author ba
  INNER JOIN reservation r ON ba.book_isbn = r.book_id
  WHERE ba.author_id = a.id
);
```

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

```
SELECT b.titleBK, GROUP_CONCAT(DISTINCT a.name) AS author_names, GROUP_CONCAT(DISTINCT c.category) AS categories, b.ISBN
                FROM book b
                LEFT JOIN book_author ba ON b.ISBN = ba.book_isbn
                LEFT JOIN author a ON ba.author_id = a.id
                LEFT JOIN book_category bc ON b.ISBN = bc.book_id
                LEFT JOIN category c ON bc.category_id = c.id
                WHERE 1=1
```

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 category_1_id, c1.category AS category_1, bc2.category_id AS category_2_id, c2.category AS category_2, COUNT(r.book_id) AS reservation_count
    FROM book_category bc1
    INNER JOIN book_category bc2 ON bc1.book_id = bc2.book_id AND bc1.category_id < bc2.category_id
    INNER JOIN reservation r ON bc1.book_id = r.book_id
    INNER JOIN category c1 ON bc1.category_id = c1.id
    INNER JOIN category c2 ON bc2.category_id = c2.id
    GROUP BY bc1.category_id, c1.category, bc2.category_id, c2.category
    ORDER BY reservation_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 display_count
FROM book_author
GROUP BY author_id
HAVING COUNT(*) < (
    SELECT COUNT(*) - 4 AS top_count
    FROM book_author
    GROUP BY author_id
    ORDER BY top_count DESC
    LIMIT 1
)
ORDER BY display_count DESC;
```

Χειριστής

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

```
SELECT b.titleBK, GROUP_CONCAT(DISTINCT c.category) AS categories, GROUP_CONCAT(DISTINCT a.name) AS authors, SUM(bs.inventory) AS total_inventory
FROM book b
INNER JOIN book_category bc ON b.ISBN = bc.book_id
INNER JOIN category c ON bc.category_id = c.id
INNER JOIN book_author ba ON b.ISBN = ba.book_isbn
INNER JOIN author a ON ba.author_id = a.id
INNER JOIN book_school bs ON b.ISBN = bs.book_id
GROUP BY b.titleBK;
```

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 r.reservation_id, u.first_name, u.last_name, DATEDIFF(CURDATE(), r.expected_date) AS delayed_days
FROM reservation r
JOIN user u ON r.user_id = u.id
WHERE r.return_date IS NULL
  AND CURDATE() > r.expected_date
  order by delayed_days;

  SELECT r.reservation_id, u.first_name, u.last_name, DATEDIFF(CURDATE(), r.expected_date) AS delayed_days
FROM reservation r
JOIN user u ON r.user_id = u.id
WHERE r.return_date IS NULL
  AND CURDATE() > r.expected_date
  order by u.first_name ASC;

  SELECT r.reservation_id, u.first_name, u.last_name, DATEDIFF(CURDATE(), r.expected_date) AS delayed_days
FROM reservation r
JOIN user u ON r.user_id = u.id
WHERE r.return_date IS NULL
  AND CURDATE() > r.expected_date
  order by u.last_name ASC;
```

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

```
SELECT u.first_name, u.last_name, c.category, AVG(r.rating)
FROM user u
LEFT JOIN ratings r ON u.id = r.user_id
JOIN book_category bc ON r.book_id = bc.book_id
JOIN category c ON bc.category_id = c.id
WHERE u.id = 100
AND c.id = 1
ORDER BY c.category;
```

Χρήστης

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

```
titleBK category author
SELECT b.titleBK, GROUP_CONCAT(c.category) AS categories, GROUP_CONCAT(a.name) AS authors
FROM book b
INNER JOIN book_category bc ON b.ISBN = bc.book_id
INNER JOIN category c ON bc.category_id = c.id
INNER JOIN book_author ba ON b.ISBN = ba.book_isbn
INNER JOIN author a ON ba.author_id = a.id
GROUP BY b.titleBK;

SELECT a.name, GROUP_CONCAT(DISTINCT c.category) AS categories, GROUP_CONCAT(DISTINCT b.titleBK) AS books
FROM book b
INNER JOIN book_category bc ON b.ISBN = bc.book_id
INNER JOIN category c ON bc.category_id = c.id
INNER JOIN book_author ba ON b.ISBN = ba.book_isbn
INNER JOIN author a ON ba.author_id = a.id
GROUP BY a.name;

SELECT c.category, a.name, b.titleBK
FROM book b
INNER JOIN book_category bc ON b.ISBN = bc.book_id
INNER JOIN category c ON bc.category_id = c.id
INNER JOIN book_author ba ON b.ISBN = ba.book_isbn
INNER JOIN author a ON ba.author_id = a.id
GROUP BY c.category;
```

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

```
SELECT u.first_name, u.last_name, u.id, GROUP_CONCAT(b.titleBK) AS reserved_books
FROM user u
INNER JOIN reservation r ON u.id = r.user_id
INNER JOIN book b ON r.book_id = b.ISBN
where u.id = 392
GROUP BY u.id;
```