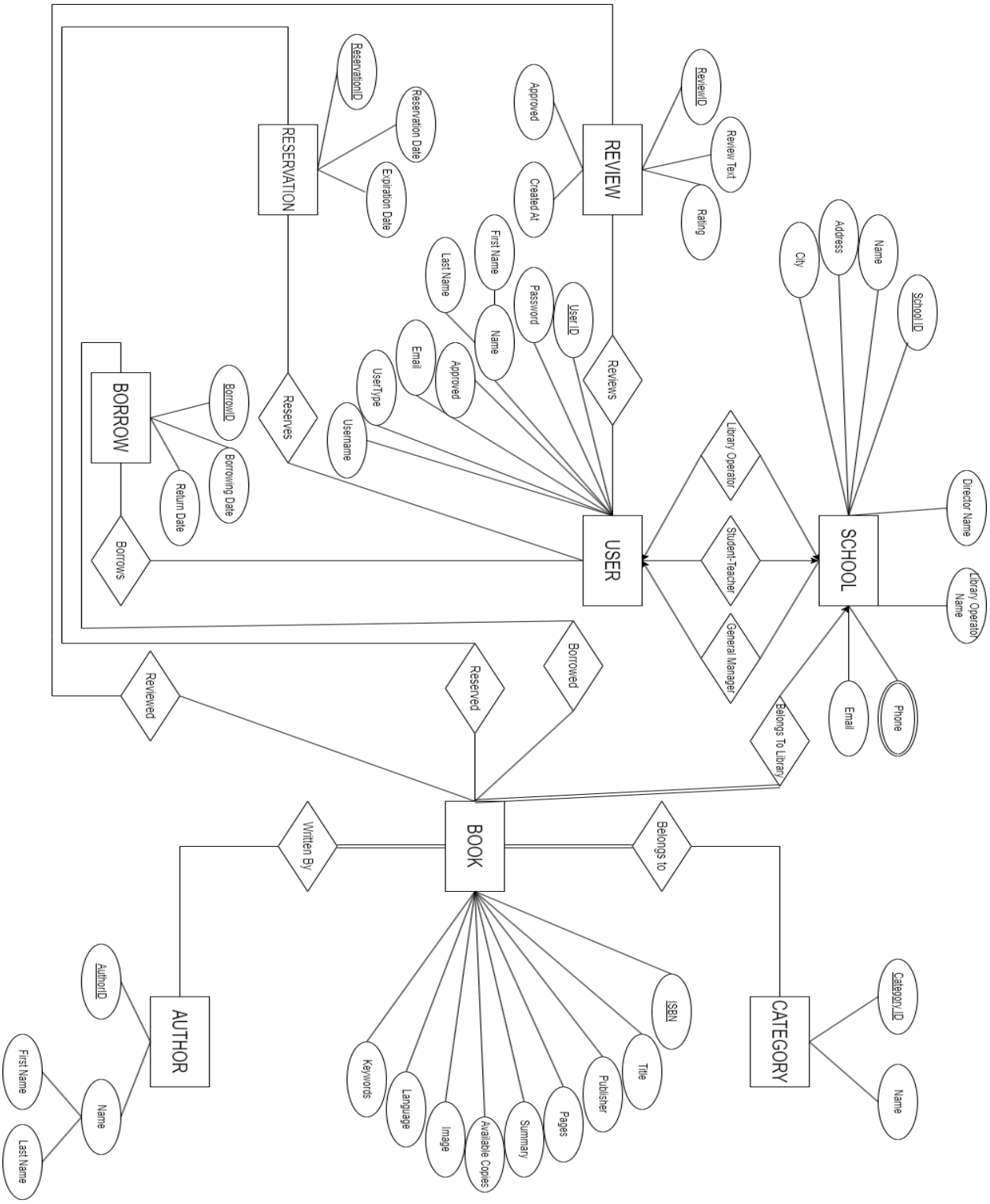
Εξαμηνιαία Εργασία στο Μάθημα των Βάσεων Δεδομένων

Εαρινό Εξάμηνο 2022-2023, Ομάδα 115

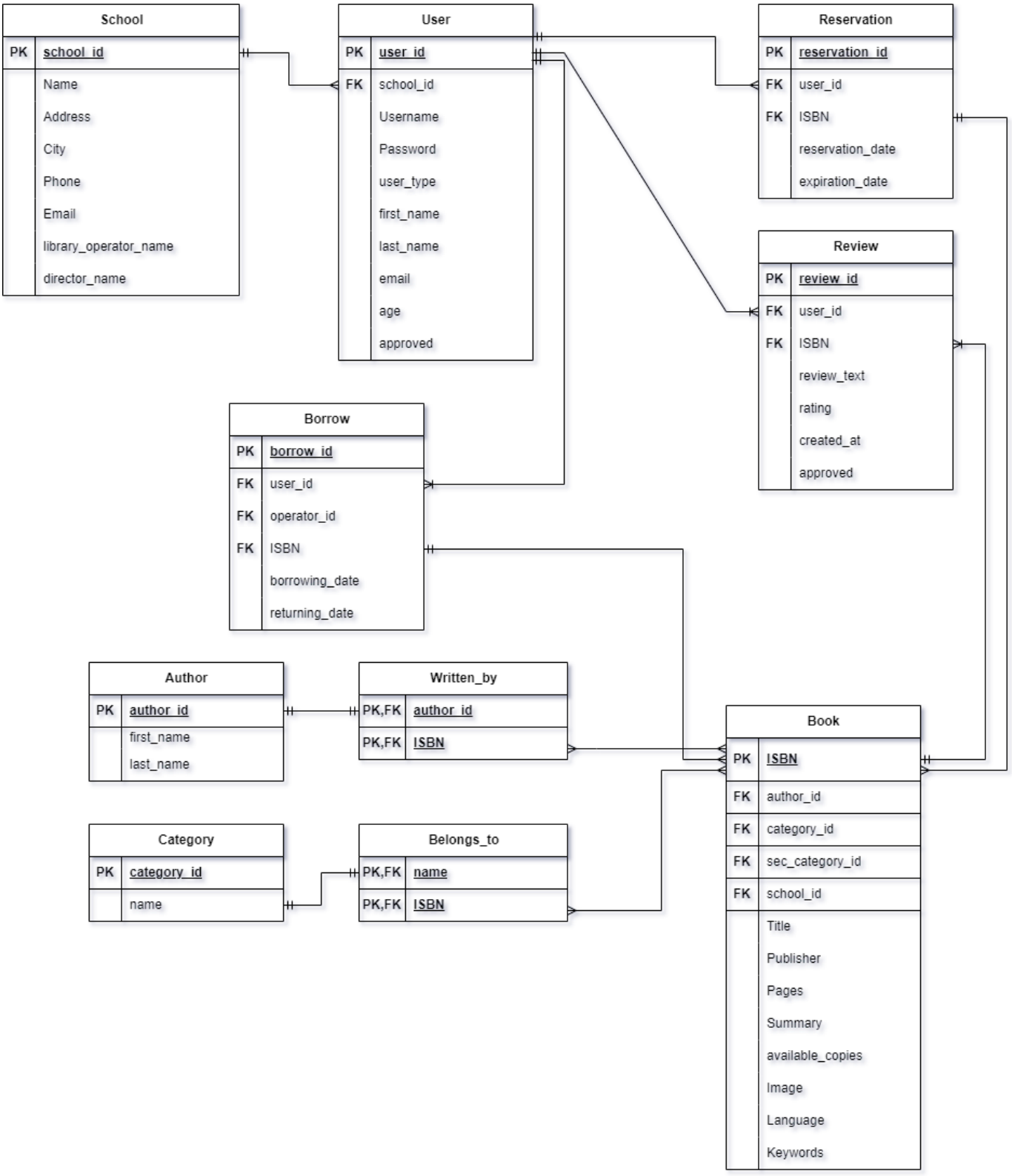
Φοιτητές:

Όνομα: Γεωργία Μανιφάβα, Ιωαννης Νεκταριος Ηλιοπουλος, Παππάς Κωνσταντίνος, **ΑΜ:** 03118001, 03119919, 03114905

ER διάγραμμα:



Σχεσιακό διάγραμμα:



Σχόλια: Το entity users κατηγοριοποιείται, μέσω της εντολής enum στο user_type attribute, σε administrator, library_operator, student, professor. Θεωρούμε ότι κάθε χρήστης έχει ένα μοναδικό school_id και κάθε σχολείο έχει έναν μοναδικό χειριστή.

Σημείωση: Όσον αφορά τα foreign keys έχουν οριστεί on delete και on update cascade, έτσι ώστε όταν γίνεται διαγραφή ή ανανέωση σε πίνακα γονέα, να προσαρμόζονται κατάλληλα και τα αντίστοιχα δεδομένα που σχετίζονται μέσω foreign keys με τα primary keys του πίνακα γονέα.

DDL και DML scripts:

DDL link: https://github.com/ntua-el14905/database/blob/main/sql_code/create_db.sql

DML link: https://github.com/ntua-el14905/database/blob/main/sql_code/dml.sql

Views:

Δεδομένου ότι τα views είναι queries που έχουν εκτελεστεί και λειτουργούν σαν ανανεώσιμα tables της βάσης και άρα καταλαμβάνουν επιπλέον αποθηκευτικό χώρο, αποφασίσαμε να επιλέξουμε να εντάξουμε σε αυτά δύο από τα queries του διαχειριστή: το 3.1.5 και το 3.1.7:

```
CREATE VIEW authors_comparison AS
select CONCAT(a.first_name, ' ', a.last_name) AS author_name
from book b
join author a on a.author_id = b.author_id
group by b.author_id
having count(isbn) <= (SELECT MAX(book_count) - 5
                       FROM (SELECT COUNT(isbn) AS book_count
                             FROM book
                             GROUP BY author_id) AS counts);

CREATE VIEW operators_comparison AS
SELECT CONCAT(U.first_name, ' ', U.last_name) AS user_name, COUNT(*) AS loan_count
FROM Borrow B
JOIN User U ON B.operator_id = U.user_id
WHERE B.borrowing_date >= DATE_SUB(CURDATE(), INTERVAL 12 MONTH)
GROUP BY B.operator_id
HAVING loan_count > 20
AND loan_count = (
   SELECT COUNT(*) AS same_loan_count
   FROM Borrow
   WHERE borrowing_date >= DATE_SUB(CURDATE(), INTERVAL 12 MONTH)
   GROUP BY operator_id
   HAVING operator_id = B.operator_id
);
```

Indexing:

CREATE INDEX idx_u_school_id ON user (school_id);

CREATE INDEX idx_b_school_id ON book (school_id);

CREATE INDEX idx_author_id ON book (author_id);

CREATE INDEX idx_category_id ON book (category_id);

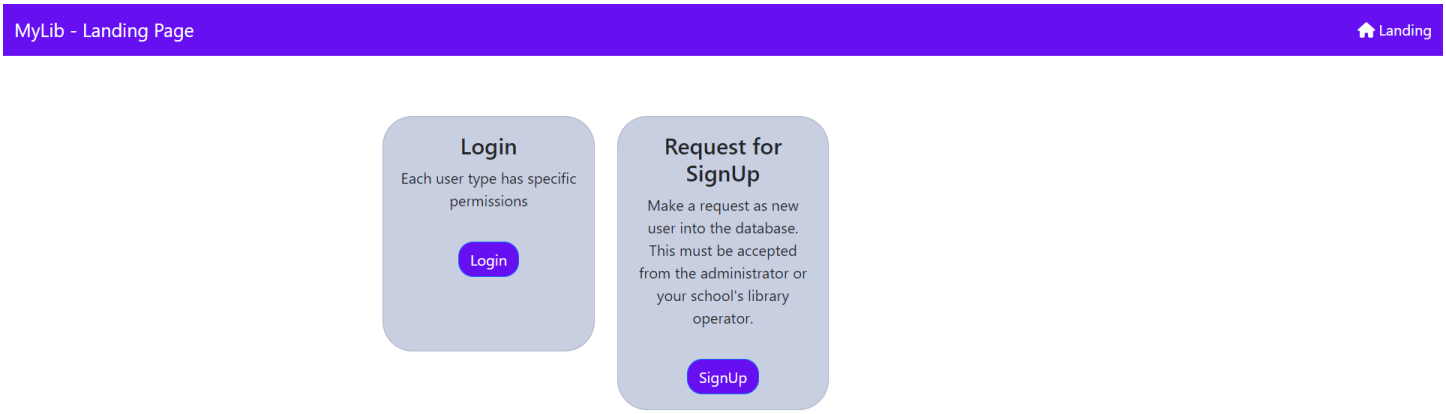
CREATE INDEX idx_user_id ON borrow (user_id);

CREATE INDEX idx_isbn ON borrow (isbn);

Τα ευρετήρια εξυπηρετούν στην ταχύτερη προσπέλαση των tables, χωρίς να χρειάζεται να διαβαστούν όλες οι γραμμές. Τα indexes όμως επιβαρύνουν τον αποθηκευτικό χώρο. Η επιλογή των παραπάνω indexes έγινε με κριτήριο την συχνή εμφάνιση των παραπάνω columns σε διάφορα joins που χρησιμοποιήθηκαν σε διάφορα queries, προκειμένου να επιταχυνθεί η εκτέλεσή τους. Επιπλέον, στη MySQL, ορίζονται αυτόματα ως ευρετήρια όλα τα primary keys.

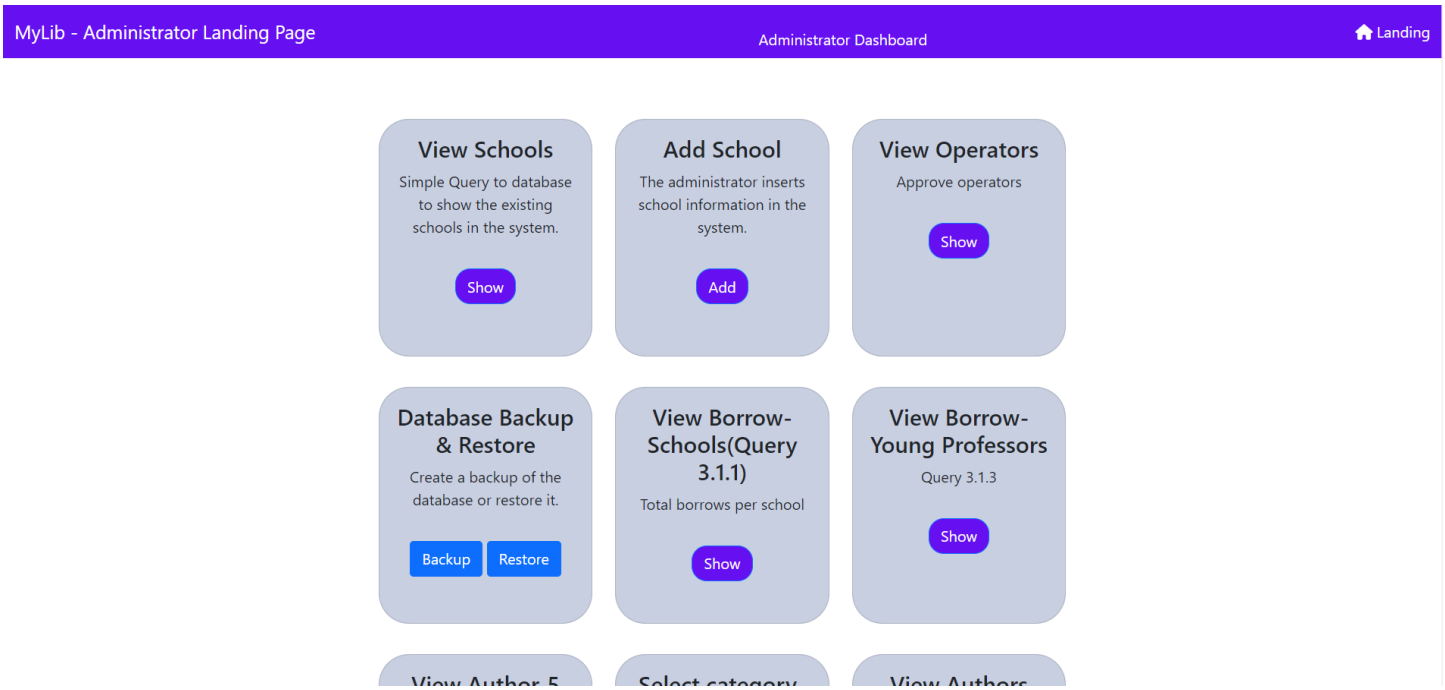
User Manual:

Ο επισκέπτης της σελίδας μπορεί να κάνει εγγραφή στην φόρμα του signup ή να συνδεθεί χρησιμοποιώντας username και password. Επίσης, μπορεί να αλλάξει τον κωδικό εάν το επιθυμεί. Κατά το login, πραγματοποιείται το κατάλληλο authentication και authorization ανάλογα με τις δυνατότητες που έχει το κάθε user_type.



Για παράδειγμα, εάν συνδεθούμε σαν διαχειριστής, θα βλέπουμε το παρακάτω layout με τις δυνατότητες που μας παρέχει η σελίδα, σύμφωνα με την εκφώνηση της άσκησης, δηλαδή για view, update, create και delete δυνατότητες πάνω στις αντίστοιχες οντότητες. Για παράδειγμα, ο διαχειριστής μπορεί να κάνει view τα αποτελέσματα των queries που το αφορούν(3.1.1-3.1.7) και να τα φιλτράρει με κάποια κριτήρια αναζήτησης μέσω drop down list. Επίσης, μπορεί να καταχωρίσει τα στοιχεία των νέων σχολικών μονάδων(Add School), να εγκρίνει τους χειριστές(εντός του View operators) και να κάνει backup/restore στη βάση.

Αντίστοιχα layouts υπάρχουν και για τα υπόλοιπα είδη χρηστών.



Αναλυτικά βήματα εγκατάστασης εφαρμογής και σύνδεσμος για το git repo:

Για την ανάπτυξη της εφαρμογής χρησιμοποιήθηκε MySQL για την υλοποίηση της ΒΔ, NodeJS για το server side και HTML για το client side της εφαρμογής.

1. Ανοίγουμε στο MySQL Workbench τα sql scipts που είναι μέσα στον φάκελο sql_code. Συγκεκριμένα τα create_db, dml, viewscreation και indexcreation. Δημιουργούμε τη βάση, κάνουμε insert τα δεδομένα στους πίνακες και δημιουργούμε τα Views και τα ευρετήρια.

2. Επεξεργαζόμαστε το .env.localhost.txt εντός του φακέλου app, σύμφωνα με τις οδηγίες παρακάτω:

- Ανοίγουμε το .env.localhost.txt
- Αλλάζουμε το περιεχόμενο του για να δουλεύει στο σύστημά μας

  .env.localhost content:

  SERVER_PORT=3000

  DB_HOST=localhost

  DB_PORT=3306

  DB_USER=dbuser

  DB_PASS=dbpass

  DB=db-name

- Κάνουμε Save us, αφαιρώντας την κατάληξη .txt και επιλέγουμε All files στο Save as types.

3. Στη γραμμή εντολών κατευθυνόμαστε στο directory του app με cd command.

4. Τρέχουμε την εντολή npm install για να εγκαταστήσουμε όλα τα dependencies που περιέχονται στο package.json.

5. Κάνουμε npm start για την εκκίνηση του Server.

6. Επισκεπτόμαστε την σελίδα http://localhost:PORT/, όπου PORT ό,τι έχουμε γράψει στο SERVER_PORT στο .env.localhost.

7. Παρατήρηση: προκειμένου να μπορεί να κάνει backup/restore ο administrator, χρειάζεται να έχετε επιβεβαιώσει ότι υπάρχει το MySQL\MySQL Server X.X\bin Directory στο PATH στις μεταβλητές περιβάλλοντος του συστήματός σας. Εάν δεν υπάρχει, χρειάζεται να το προσθέσετε για τη συγκεκριμένη λειτουργία του administrator.

Git repo link: ntua-el19050/Databases-Project - GitHub

.