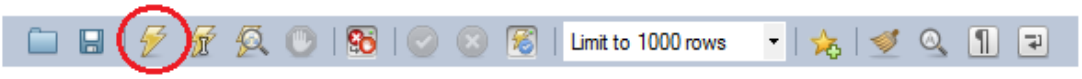
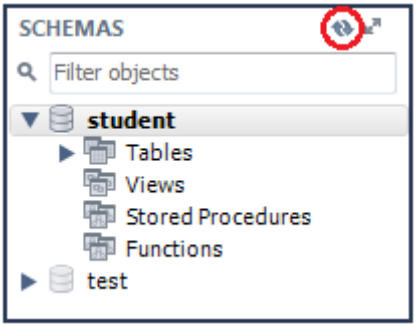
Εργαστηριακό μάθημα 5
-
Όψεις, Δικαιώματα χρηστών και Ασφάλεια ΣΒΔ

Δημιουργία της βάσης SongsDB

1. Ανοίξτε την εφαρμογή **MySQL Workbench** και συνδεθείτε στη βάση (όπως στο πρώτο εργαστήριο).
2. Αν το schema student υπάρχει ήδη διαγράψτε το. Κάνετε δεξί κλικ πάνω του, επιλέξτε "**Drop schema…**" και στο μενού που θα εμφανιστεί επιλέξτε "**Drop now**".
3. Επιλέξτε «**File -> Open SQL Script…**» για να ανοίξετε το αρχείο "***Lab5Dump.sql***" και στη συνέχεια πιέστε το κουμπί **«Execute»** (βρίσκεται στην γραμμή εργαλείων). Εναλλακτικά, μπορείτε να εκτελέσετε το script επιλέγοντας «**Query -> Execute (All or Selection)**» ή πιέζοντας Ctrl+Shift+Enter.

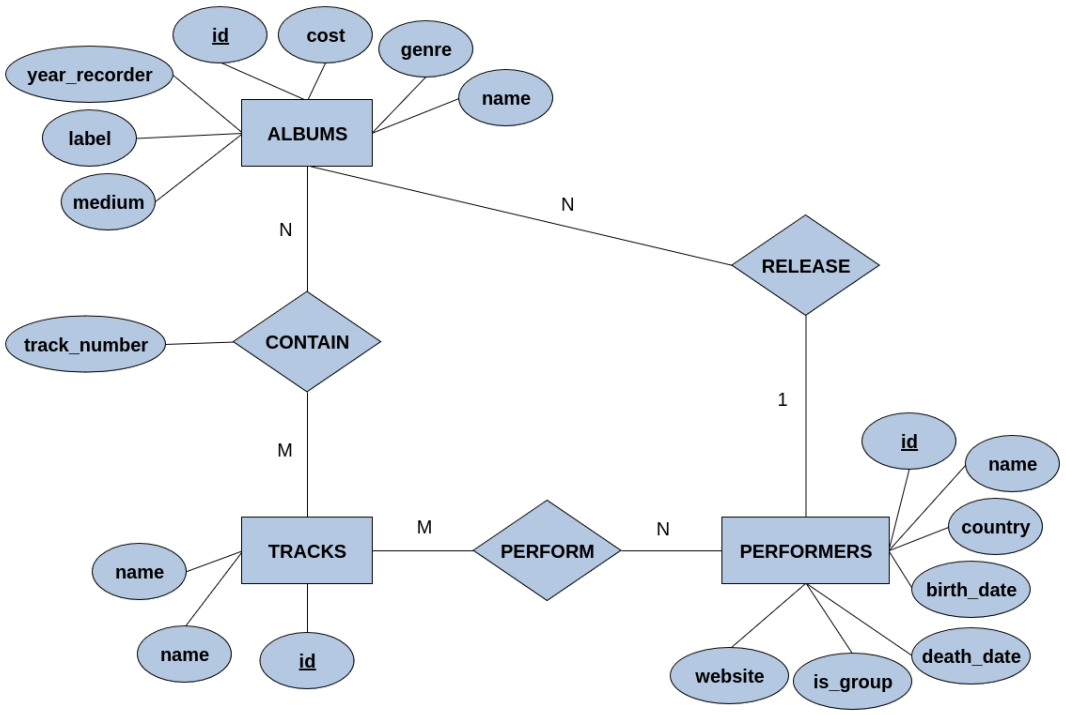


4. Ελέγξτε στο **«SCHEMAS»** (στα αριστερά του GUI) αν κάτω από το student έχουν όντως δημιουργηθεί όλοι οι πίνακες με τα ζητούμενα κλειδιά και δεδομένα. Θα χρειαστεί να κάνετε "refresh" τα SCHEMAS για να εμφανιστεί το σχήμα της ΒΔ "student".



5. Επιλέξτε «**File -> New Query Tab**» (εναλλακτικά μπορείτε να πατήσετε το κουμπί «**New Query Tab**» στην γραμμή εργαλείων Standard ή τον συνδυασμό πλήκτρων Ctrl+N).

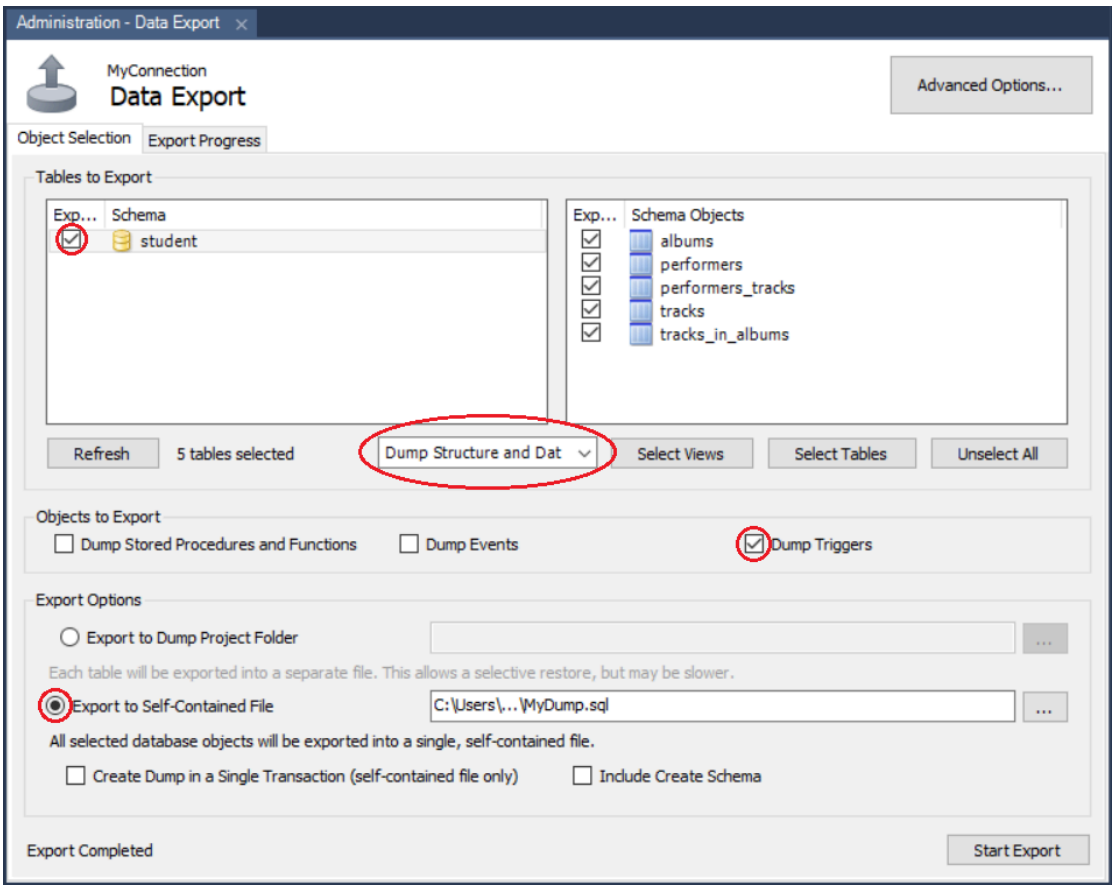
Απλοποιημένο διάγραμμα Ο/Σ της βάσης SongsDB



Εξαγωγή της ΒΔ

Από το περιβάλλον MySQL Workbench μπορείτε να πάτε στο Administration (στα αριστερά του GUI) και να επιλέξετε το Data Export.

Στη συνέχεια στην κεντρική οθόνη επιλέγετε τη βάση σας, σημειώνετε τις επιλογές Dump Structure and Data, Dump Triggers και Export to Self-Contained file (σε αυτό το σημείο επιλέγετε και που θα αποθηκευτεί το dump), και πατάτε Start Export.



Εναλλακτικά, μπορείτε να ανοίξετε τη γραμμή εντολών και να εκτελέσετε την παρακάτω εντολή:

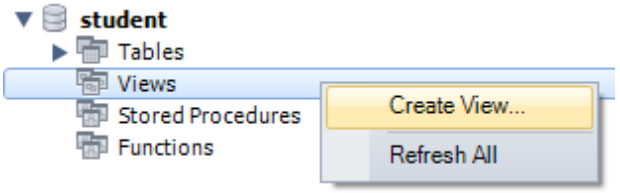
```
mysqldump -u student -p student > db-backup.sql
```

Όψεις

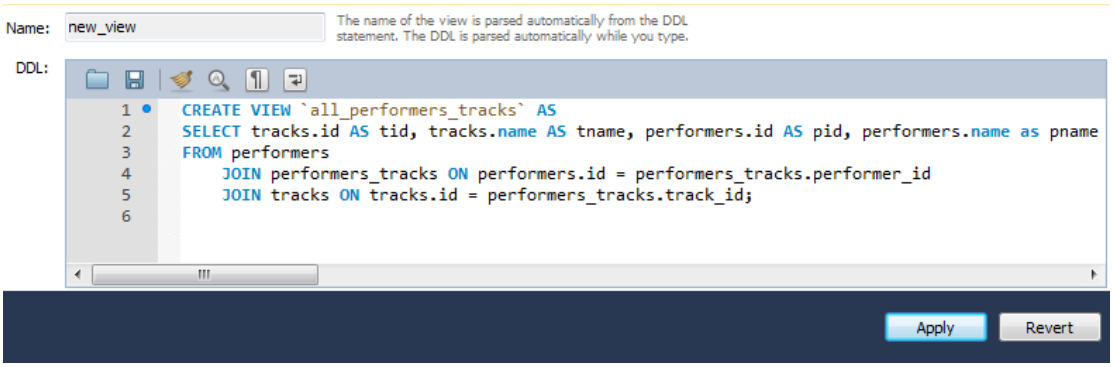
Οι **όψεις** είναι το **αποτέλεσμα συγκεκριμένων ερωτημάτων που κάνετε στη βάση**. Μια όψη μπορεί να σχετίζεται με την παρουσίαση περιορισμένων στοιχείων ή/και δεδομένων που συντίθενται από πολλούς πίνακες σε μια ομάδα χρηστών. Μπορεί, επίσης, να «μοντελοποιεί» ένα σύνθετο αλλά συχνό ερώτημα στην ΒΔ, επιτρέποντας έτσι την χρήση πιο απλών ερωτημάτων στην όψη αντί του αρχικού σύνθετου ερωτήματος.

Έστω ότι θέλουμε να δημιουργήσουμε μια όψη που να εμφανίζει τα id και τα ονόματα των τραγουδιών (tracks) και αυτών που τα ερμηνεύουν.

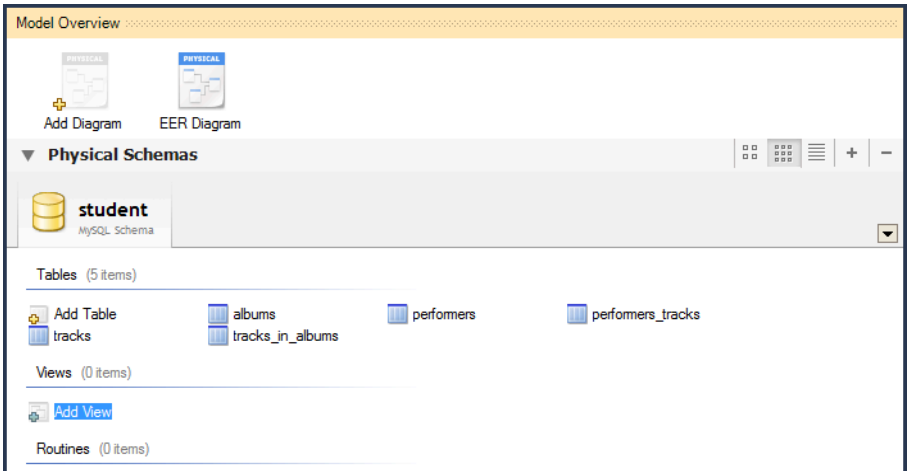
Επιλέγουμε «**Views □ Create View…**» όπως παρακάτω:



Στη συνέχεια, μπορούμε να προσδιορίσουμε το όνομα και το ερώτημα της όψης, γράφοντάς τα στον editor που εμφανίζεται στο παράθυρο και να δημιουργήσουμε την όψη επιλέγοντας το Apply.



Εναλλακτικά, μια όψη μπορεί να δημιουργηθεί και από την επιλογή Add View στο Model του MySQL Workbench:



Μπορούμε πλέον να χρησιμοποιήσουμε την όψη σε ένα οποιοδήποτε SQL ερώτημα, όπως και τους κανονικούς πίνακες. Με το παρακάτω ερώτημα, εμφανίζουμε όλα τα περιεχόμενα της όψης:

```
SELECT *
FROM all_performers_tracks
```

Όψεις και Queries

Ένα **query** είναι μια εντολή της SQL που εκτελείται δυναμικά για να ανακτήσουμε δεδομένα από πίνακες.

- Εκτελείται με το που υποβληθεί και τα αποτελέσματά του δεν αποθηκεύονται μόνιμα
- Μπορούμε να γράψουμε και να (επιχειρήσουμε να) εκτελέσουμε πολύ περίπλοκα queries

Μια **όψη (view)** είναι ένας εικονικός πίνακας που βασίζεται σε ένα προκαθορισμένο query.

- Η όψη **αποθηκεύεται στη βάση δεδομένων με ένα όνομα** και μπορεί να χρησιμοποιηθεί όπως ένας πίνακας - είναι στατική.
- Απλοποιεί την επαναχρησιμοποίηση περίπλοκων/συχνά χρησιμοποιούμενων queries.

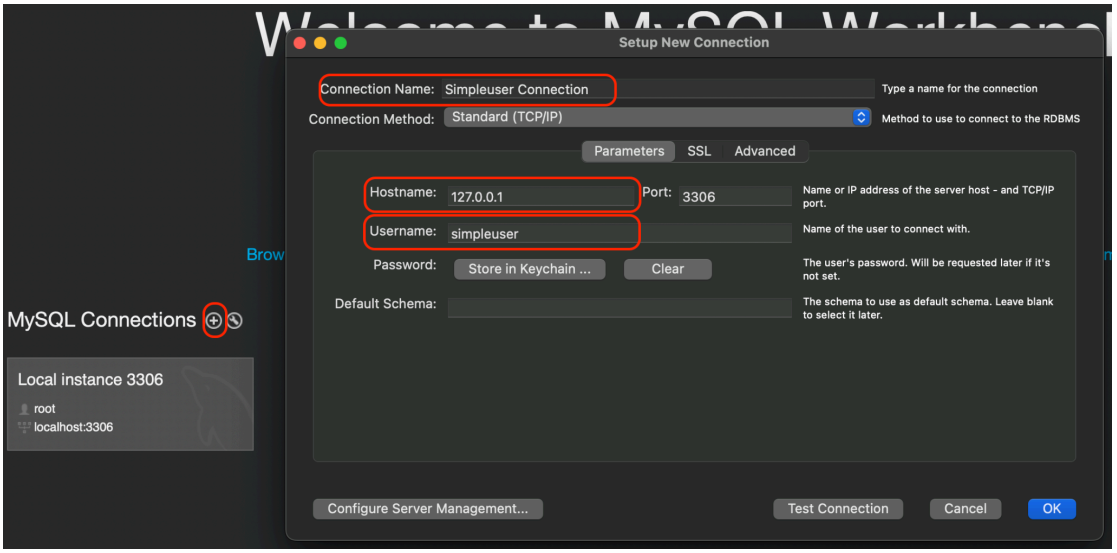
Χρήστες – Δικαιώματα

Σε ένα σενάριο χρήσης της βάσης σας, π.χ. μέσω μιας εφαρμογής Java ή μιας ιστοσελίδας, είναι απαραίτητο να οριστούν οι κατηγορίες χρηστών και τα δικαιώματά τους σε κάθε πίνακα της βάσης. Στο MySQL Workbench μπορούμε να οργανώσουμε τους χρήστες της βάσης μας όπως παρακάτω.

Σημείωση: δεν έχουν όλοι οι χρήστες δικαίωμα να δημιουργούν νέους χρήστες και να τους δίνουν δικαιώματα. Στη συνέχεια, θεωρούμε ότι έχουμε πρόσβαση στη βάση ως root, ώστε να απαντήσουμε στα παρακάτω ερωτήματα.

1. Να δημιουργηθεί ο χρήστης ***simpleuser*** με κωδικό ***simplepassword*** που θα έχει δικαίωμα μόνο να κάνει ερωτήματα select στη βάση student (και όχι να κάνει εισαγωγή, ενημέρωση ή διαγραφή δεδομένων). Ο χρήστης θα πρέπει να μπορεί να προσπελάσει απομακρυσμένα τη βάση.

```
CREATE USER 'simpleuser'@'localhost' IDENTIFIED BY 'simplepassword';
CREATE USER 'simpleuser'@'%' IDENTIFIED BY 'simplepassword';
GRANT SELECT ON student.* TO 'simpleuser'@'localhost';
GRANT SELECT ON student.* TO 'simpleuser'@'%';
```



Δοκιμάστε να προσθέσετε δεδομένα σε κάποιον πίνακα της βάσης. Για παράδειγμα εκτελέστε την παρακάτω εντολή για να προσθέσετε στον πίνακα tracks:

```
INSERT INTO tracks (name, duration) VALUES ("lost", 10.0);
```

Δοκιμάστε να πάρετε δεδομένα από οποιοδήποτε πίνακα της βάσης. Για παράδειγμα εκτελέστε την παρακάτω εντολή:

```
SELECT * FROM performers;
```

2. Να δημιουργηθεί ο χρήστης ***truser*** με κωδικό ***mypass*** που θα έχει δικαίωμα μόνο να εισάγει δεδομένα και να τα ενημερώνει, μόνο όμως μέσα από το μηχάνημα που βρίσκεται η βάση και μόνο για τον πίνακα tracks.

```
CREATE USER 'truser'@'localhost' IDENTIFIED BY 'mypass';
GRANT INSERT, UPDATE ON student.tracks TO 'truser'@'localhost';
```

Δοκιμάστε να προσθέσετε δεδομένα σε κάποιον πίνακα της βάσης. Για παράδειγμα εκτελέστε την παρακάτω εντολή για να προσθέσετε στον πίνακα tracks:

```
INSERT INTO tracks (name, duration) VALUES ("lost", 10.0);
```

Δοκιμάστε να πάρετε δεδομένα από οποιοδήποτε πίνακα της βάσης. Για παράδειγμα εκτελέστε την παρακάτω εντολή:

```
SELECT * FROM performers;
```

3. Να δημιουργηθεί ο χρήστης ***sadmin*** με κωδικό ***superpass*** που θα έχει όλα τα δικαιώματα του διαχειριστή στη βάση students, αρκεί να τα ασκεί μόνο όμως μέσα από το μηχάνημα που βρίσκεται η βάση.

```
CREATE USER 'sadmin'@'localhost' IDENTIFIED BY 'superpass';
GRANT ALL PRIVILEGES ON student.* TO 'sadmin'@'localhost';
```

Δοκιμάστε να προσθέσετε δεδομένα σε κάποιον πίνακα της βάσης. Για παράδειγμα εκτελέστε την παρακάτω εντολή για να προσθέσετε στον πίνακα tracks:

```
INSERT INTO tracks (name, duration) VALUES ("lost", 10.0);
```

Δοκιμάστε να πάρετε δεδομένα από οποιοδήποτε πίνακα της βάσης. Για παράδειγμα εκτελέστε την παρακάτω εντολή:

```
SELECT * FROM performers;
```

4. Να καταργηθεί το δικαίωμα του χρήστη ***truser*** να ενημερώνει τα δεδομένα του πίνακα tracks (ο χρήστης είχε αυτό το δικαίωμα μόνο μέσα από το μηχάνημα που βρίσκεται η βάση).

```
REVOKE UPDATE ON student.tracks FROM 'truser'@'localhost';
```

5. Δημιουργήστε τον ρόλο ***reader*** και δώστε δικαιώματα ανάγνωσης σε όλους τους πίνακες της ΒΔ. Στη συνέχεια δώστε δικαιώματα του ρόλου ***reader*** στον χρήστη ***truser***.

```
CREATE ROLE 'reader'; GRANT SELECT ON student.* TO 'reader';
```

```
GRANT 'reader' TO 'truser'@'localhost';
```

Η εν λόγω λειτουργικότητα είναι εφικτή μόνο σε νεότερες εκδόσεις της MySQL, ενώ σε παλαιότερες εκδόσεις τα δικαιώματα θα πρέπει να ορίζονται ξεχωριστά για κάθε χρήστη (περισσότερα στο https://dev.mysql.com/doc/refman/8.0/en/roles.html)

Προκειμένου να χρησιμοποιηθεί ο νέος ρόλος από τον ενδιαφερόμενο χρήστη, πρέπει να **ενεργοποιηθεί** μετά την είσοδο (log in) του χρήστη στη ΒΔ.

```
SET ROLE reader;
```

Όψεις και Ρόλοι

Μια **όψη** είναι ένας "εικονικός πίνακας" που βασίζεται σε ένα query (αποτέλεσμα ερωτήματος).

- Δεν αποθηκεύει δεδομένα, απλά τα εμφανίζει, αλλά αποθηκεύεται στη βάση δεδομένων και χρησιμοποιείται όπως ένας πίνακας
- Τα δεδομένα προκύπτουν από ένα query

Χρήση:

- Απόκρυψη περίπλοκων queries από τον χρήστη
- Περιορισμό δεδομένων που είναι ορατά στους χρήστες

Ένας **ρόλος** είναι ένας μηχανισμός διαχείρισης χρηστών και δικαιωμάτων στη βάση δεδομένων.

- Ομαδοποιούν χρήστες με κοινά δικαιώματα

Χρήση:

- Απλοποίηση διαχείρισης πρόσβασης
- Έλεγχος δικαιωμάτων

Συνδυασμός όψεων και ρόλων → Εξασφάλιση ότι οι χρήστες θα έχουν πρόσβαση μόνο στα δεδομένα που πρέπει να έχουν:

- Περιορισμός πρόσβασης μέσω όψεων, φιλτράρουμε δεδομένα
- Δίνουμε πρόσβαση στις όψεις αντί στους πλήρεις πίνακες (tables) μέσω των ρόλων → έλεγχος χρηστών που έχουν πρόσβαση σε όψεις
    - Πλέον μιλάμε για πρόσβαση σε όψεις και όχι σε πίνακες (με εξαίρεση πχ τους διαχειριστές-admins)
    - Η διαχείριση γίνεται πιο εύκολη, κάποιος μπορεί να ορίζει δικαιώματα σε ομάδες χρηστών με τον ίδιο ρόλο και όχι σε κάθε χρήστη ξεχωριστά.

6. Δημιουργήσετε τον χρήστη ***parlophone***, στον οποίο στη συνέχεια θα δώσετε το ρόλο ***parlophone_admin***, αφού πρώτα τον δημιουργήσετε. Χρησιμοποιήστε ίδιο username και password (αλλά μην το κάνετε ΠΟΤΕ σε ρεαλιστικό σενάριο).
Στη συνέχεια δημιουργήστε μια όψη με το όνομα ***parlophone_full*** που θα δίνει το δικαίωμα στον ρόλο ***parlophone_admin*** τοπικά να βλέπει και να κάνει εισαγωγή ή ενημέρωση μόνο τα albums τα οποία έχουν εκδοθεί από την Parlophone (είτε μόνη της, είτε σε σύμπραξη με άλλη δισκογραφική). Θα πρέπει να εμφανίζονται το όνομα του άλμπουμ, το κόστος του και το όνομα του καλλιτέχνη.
Αντίστοιχα, να γίνει το ίδιο με τον χρήστη ***capitol*** και τη δισκογραφική Capitol.

Οι εντολές για την πρώτη περίπτωση (parlophone) είναι:

```
CREATE VIEW `parlophone_full` AS
SELECT albums.name, albums.cost, performers.name as performer
FROM albums
        LEFT JOIN performers ON albums.performer_id = performers.id
WHERE
        albums.label LIKE '%parlophone%'
```

```
CREATE USER 'parlophone'@'localhost' IDENTIFIED BY 'parlophone';
CREATE ROLE 'parlophone_admin';
GRANT ALL PRIVILEGES ON student.parlophone_full TO
'parlophone_admin';
GRANT  'parlophone_admin' TO 'parlophone'@'localhost';
```

Οι εντολές για την δεύτερη περίπτωση (capitol) είναι:

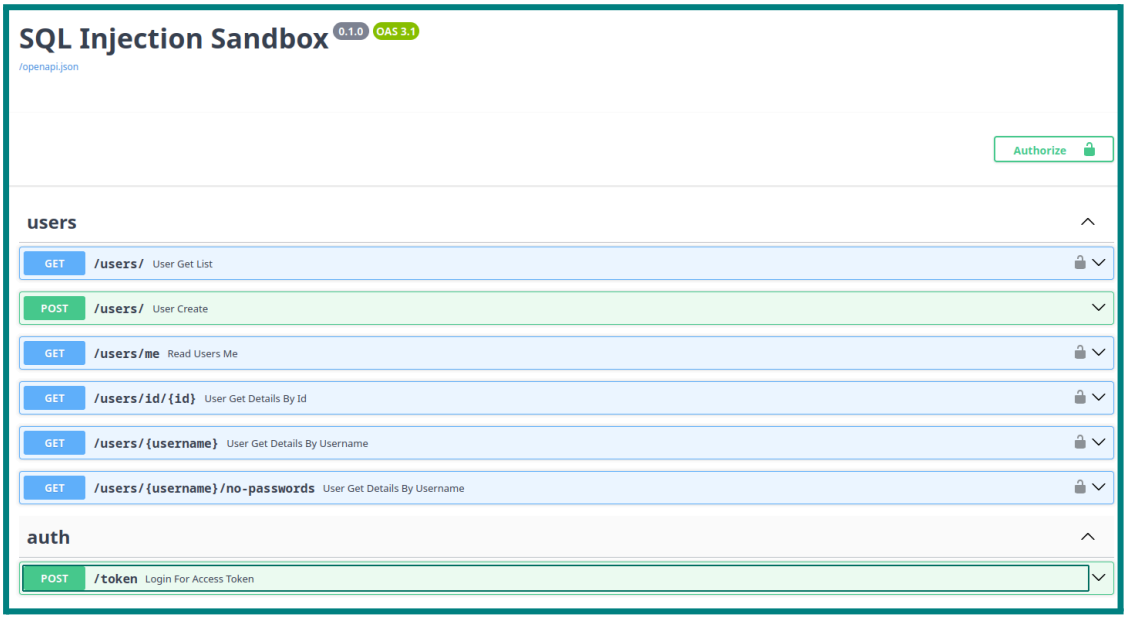
```
CREATE VIEW `capitol_full` AS
SELECT albums.name, albums.cost, performers.name as performer
FROM albums
        LEFT JOIN performers ON albums.performer_id = performers.id
WHERE
        albums.label LIKE '%capitol%'
```

```
CREATE USER 'capitol'@'localhost' IDENTIFIED BY 'capitol';
CREATE ROLE 'capitol_admin';
GRANT ALL PRIVILEGES ON student.capitol_full TO 'capitol_admin';
GRANT  'capitol_admin' TO 'capitol';
```

Σημείωση: Principle of least privilege, κάθε χρήστης πρέπει να έχει πρόσβαση στα δεδομένα, υποδομές και εφαρμογές, τα οποία έχει **ανάγκη** προκειμένου να κάνει τη δουλειά του.

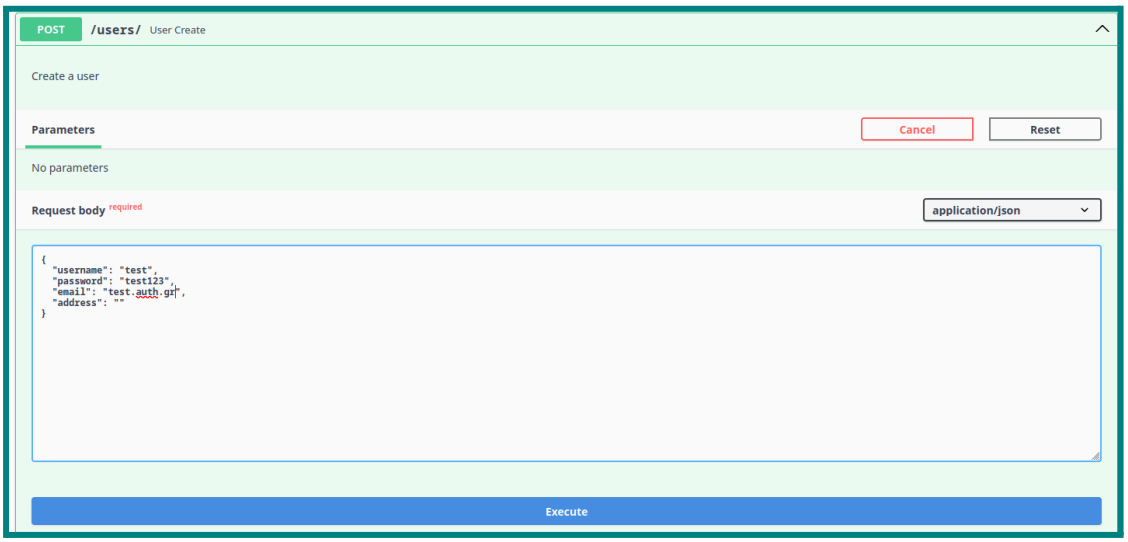
SQL Injection - SQLi

Συνδεθείτε στο στον ιστότοπο issel10.ee.auth.gr/docs.

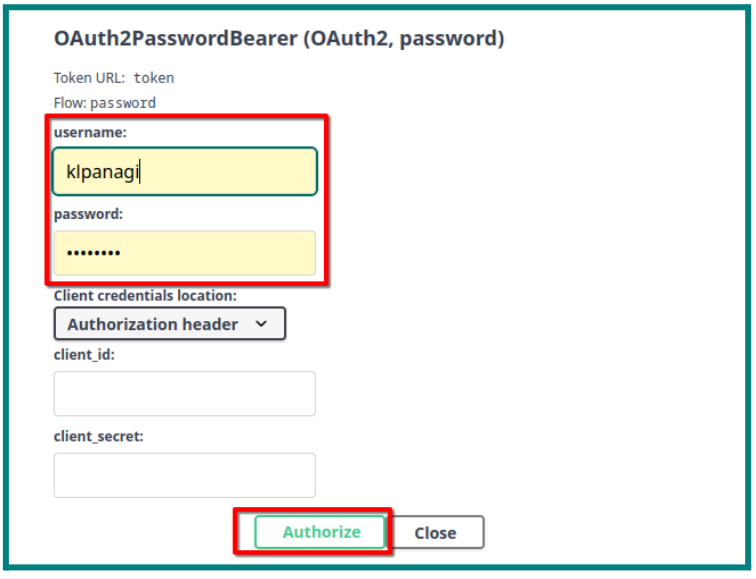


Το παραπάνω είναι το REST API μιας απλής εφαρμογής που περιέχει μόνο 1 πίνακα, τον **users**. Στον πίνακας αυτό αποθηκεύεται η πληροφορία των εγγεγραμμένων χρηστών.

Για να δημιουργήσετε λογαριασμό, χρησιμοποιήστε το τερματικό **POST /users/**.



Πατήστε το Execute για να εγγραφείτε στο ΣΒΔ. Στη συνέχεια επιλέγοντας Authorize (πάνω δεξιά) και χρησιμοποιώντας τα στοιχεία που χρησιμοποιήσατε για την εγγραφή, μπορείτε να συνδεθείτε και να πάρετε πρόσβαση στις υπόλοιπες υπηρεσίες.



7. Ελέγξτε τα παρακάτω ερωτήματα (χωρίς τα μονά 'εισαγωγικά') και δείτε σε κάθε περίπτωση τι συμβαίνει:

α) /users/{id}

```
id: 8
```

Το σύστημα επιστρέφει τα στοιχεία του performer με id 8.

β) /users/{username}

```
username: Kylie Minogue
```

Το σύστημα επιστρέφει τα στοιχεία του performer με όνομα Kylie Minogue.

γ) /users/{id}

```
id: 0 or 1=1
```

Το σύστημα επιστρέφει τα στοιχεία όλων των performers.

δ) /users/{username}

```
Name = " or ""="
```

Το σύστημα επιστρέφει τα στοιχεία όλων των performers.

8. Προσπαθήστε να κάνετε DROP τη ΒΔ (DROP Database…)

Δίνονται οι παρακάτω χρήσιμες πληροφορίες για να βρείτε το vulnerability

- API Language: Python 3.10
- Database: MariaDB 10.3.27