ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ
ΣΧΟΛΗ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ
ΕΡΓΑΣΤΗΡΙΟ ΣΥΣΤΗΜΑΤΩΝ ΒΑΣΕΩΝ ΓΝΩΣΕΩΝ ΚΑΙ ΔΕΔΟΜΕΝΩΝ
Ακ. έτος 2022-2023, 6ο εξάμηνο, ΣΗΜΜΥ
Βάσεις Δεδομένων
Εξαμηνιαία Εργασία

ΔΙΚΤΥΟ ΣΧΟΛΙΚΩΝ ΒΙΒΛΙΟΘΗΚΩΝ

ΟΜΑΔΑ 11:

Βιτζηλαίος Γιώργος (03116672 – george.vitzilaios@hotmail.com)

Αλέξανδρος Παπαδάκος (03118860 – alexpaa15@gmail.com)

Περιεχόμενα Εργασίας

Όνομα Ενότητας Σελίδα

- Περιγραφή Εργασίας

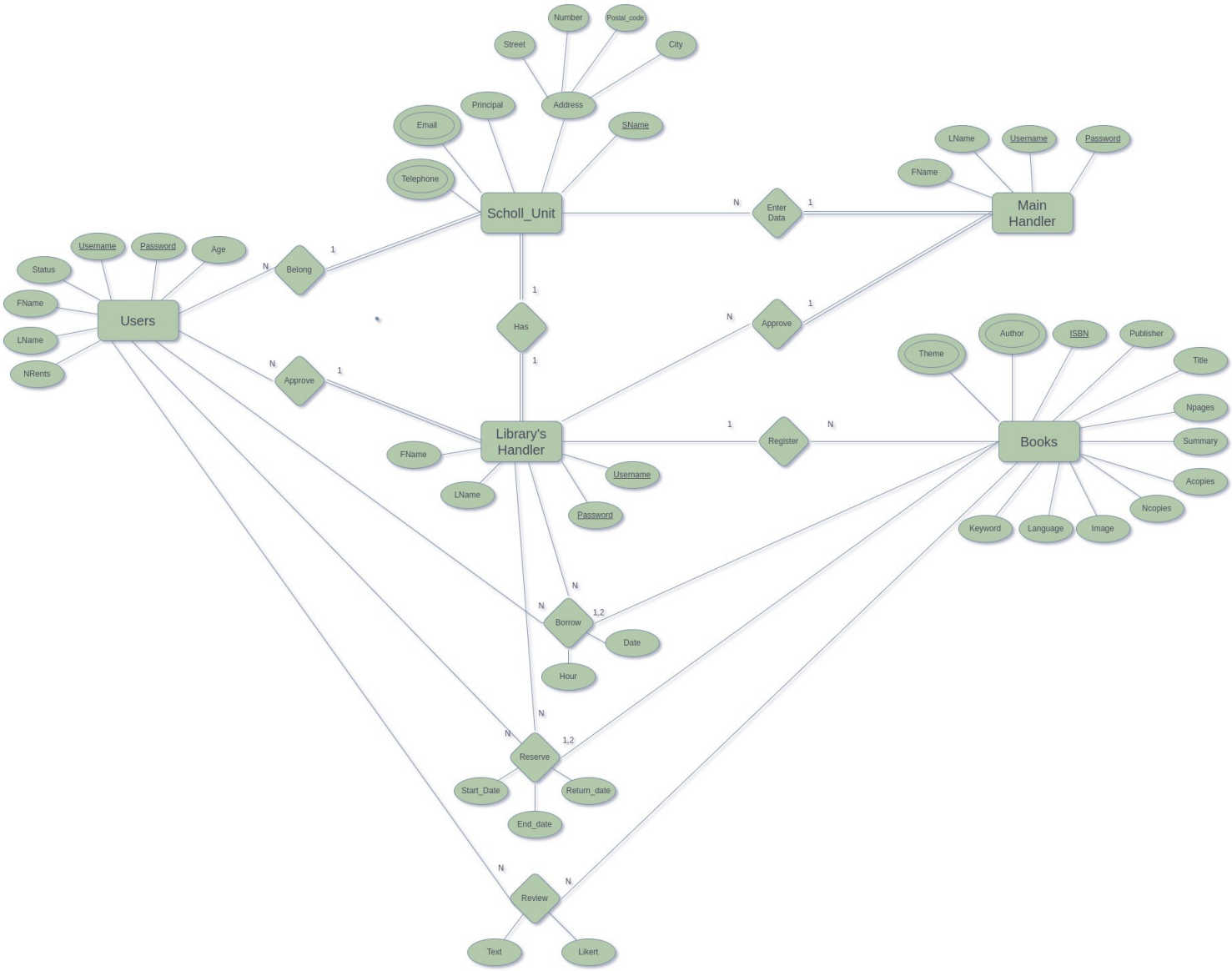
Η παρούσα εξαμηνιαία εργασία αναφέρεται στη δημιουργία μιας εφαρμογής με ΒΔ για την υποστήριξη λειτουργιών για βιβλιοθήκες σχολικών μονάδων σε Python 3 με χρήση του Flask web framework και της MySQL. Η εφαρμογή επιτρέπει σε όλες τις σχολικές μονάδες της χώρας να μπορούν να γραφτούν στο σύστημα, να καταχωρήσουν χρήστες, να καταχωρήσουν τα βιβλία που έχουν στην κατοχή τους στην εκάστοτε βιβλιοθήκη, να μπορούν να επεξεργαστούν γρήγορα και αποδοτικά όλους τους δανεισμούς, τις κρατήσεις, τα στοιχεία των βιβλίων κ.α. Για την ορθή λειτουργία της εφαρμογής απαιτείται η ανάπτυξη μίας ΒΔ σε MySQL που θα περιέχει τις απαιτούμενες πληροφορίες ώστε να είναι δυνατές οι ενέργειες που περιγράφονται παραπάνω.

Πιό συγκεκριμένα η εφαρμογή μας:

- Επιτρέπει την είσοδο στο σύστημα με την χρήση username και password. Ανάλογα με την ιδιότητα του χρήστη, έχει και τα ανάλογα δικαιώματα. Οι κατηγορίες χρηστών της εφαρμογής είναι 3(Διαχειριστής Συστήματος, Διαχειριστής εκάστοτε Σχολικής Μονάδας, Απλός Χρήστης).
- Ο Διαχειριστής του συστήματος έχει την δυνατότητα να καταχωρήσει καινούριες σχολικές μονάδες και τον αντίστοιχο Διαχειριστή της βιβλιοθήκης τους. Έχει επίσης την δυνατότητα να αντλήσει πληροφορίες από το σύστημα όπως την παρουσίαση λίστας με συνολικό αριθμό δανεισμών ανά σχολείο, να βρεί τους συγγραφείς που τα βιβλία τους δεν έχουν τύχει δανεισμού κ.α.
- Ο Διαχειριστής της βιβλιοθήκης της εκάστοτε σχολικής μονάδας. Διαχειρίζεται τα βιβλία (προσθέτει – αφαιρεί) για το σχολείο που ανήκει, έχουν επίσης την συνολική ευθύνη για τους δανεισμούς και τις κρατήσεις στην βιβλιοθήκη του σχολείου τους, δηλαδή να καταχωρούν τους δανεισμούς, τις επιστροφές, να ενημερώνουν αν υπάρχουν καθυστερήσεις τους χρήστες κ.α. Με βάση τα παραπάνω μπορούν και αυτοί να βλέπουν όλα τα βιβλία της βιβλιοθήκης που ανήκουν, ποιοί έχουν καθυστερήσει την επιστροφή όπως και τον μέσο όρο αξιολογήσεων ανά δανειζόμενο και κατηγορία.
- Ο απλός χρήστης(καθηγητές και μαθητές). Ο απλός χρήστης καταχωρείται στο σύστημα με την έγκριση του διαχειριστή της εκάστοτε σχολικής μονάδας. Οι χρήστες μπορούν επίσης να δουν τα βιβλία της βιβλιοθήκης του σχολείου τους, να κάνουν κράτηση ενός βιβλίου για μία εβδομάδα, να δανειστούν βιβλία αλλά και να τα αξιολογήσουν. Μπορούν επίσης να έχουν πρόσβαση σε λίστα που περιέχει τα βιβλία που έχουν δανειστεί.

- ER Diagram

Παρακάτω παρουσιάζεται το er διάγραμμα της βάσης δεδομένων μας.



Κάποια σχόλια πάνω στο er:

- Έχουμε πάρει ως διαφορετικές οντότητες τους διάφορους χρήστες ώστε να μπορούμε να υλοποιήσουμε με μεγαλύτερη ευκολία στην βάση μας τις διάφορες λειτουργίες που εκτελεί ο καθένας

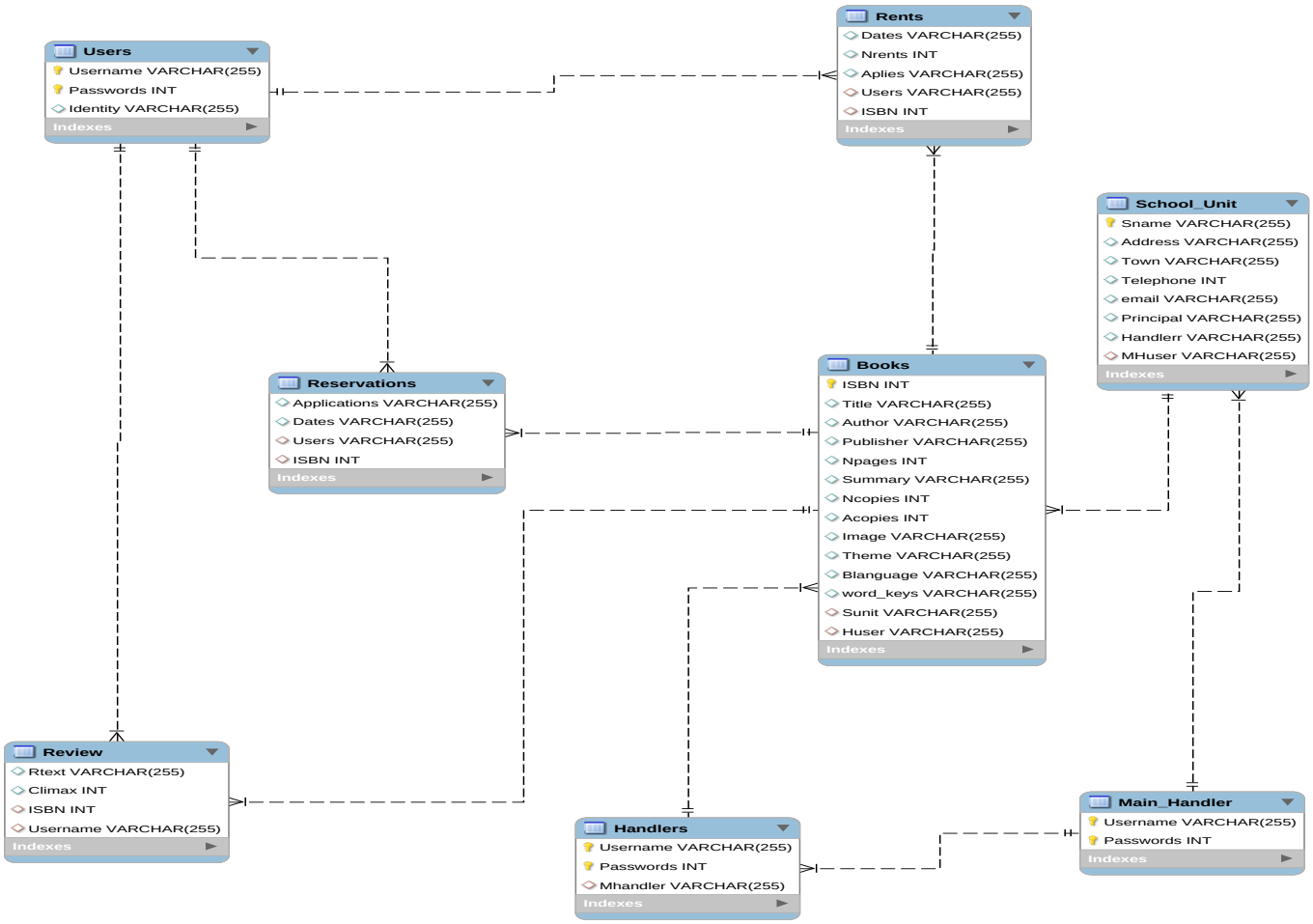
- Οντότητα αποτελεί επίσης η σχολική μονάδα και τα βιβλία.

- Τα διάφορα χαρακτηριστίκα των οντοτήτων που δίνονται από την εκφώνηση της εργασίας φαίνονται στο σχήμα με μικρούς κύκλους συνδεμένους με απλές γραμμές στις οντότητες. Κάποια χαρακτηριστικά οντοτήτων έχουν παραπάνω από δύο τιμές επομένως συμβολίζονται με διπλό κύκλο.

- Τις σχέσεις οντοτήτων τις συμβολίζουμε με ρόμβο και έχουν να κάνουν με τις σχέσεις που συνδέουν τα βιβλία με τους χρήστες (αν τα δανείζονται, αν κάνουν κράτηση ή αν τα αξιολογούν).

- Συμβολίζουμε επίσης με σχέσεις οντοτήτων τις σχέσεις που συνδέουν τους διάφορους χρήστες μεταξύ τους. Για παράδειγμα ο κύριος διαχειριστής καταχωρεί τους διαχειριστές των σχολικών μονάδων και αυτό φαίνεται με την σχέση approve.

- Σχεσιακό Διάγραμμα



Κάποια σχόλια πάνω στο relation schema

- Κάθε (strong) entity αντιστοιχεί σε έναν πίνακα με τα attributes του και το primary key του.

- Οι σχέσεις του δανεισμού, της κράτησης και της αξιολόγησης γίνονται και αυτές πίνακες μιας και είναι σχέσεις N:M.

- Βάζουμε τις συσχετίσεις με βάση αν η κάθε οντότητα από το σύνολο οντοτήτων μπορεί να συσχετιστεί ως “πολλά σε ένα” ή “ένα σε πολλά” ή “πολλά σε πολλά”. Πχ μία σχολική μονάδα μπορεί να έχει μόνο έναν διαχειριστή βιβλιοθήκης και κάθε διαχειριστής βιβλιοθήκης μπορεί να ανήκει μόνο σε ένα σχολείο.

- Θέτουμε τα Foreign Keys ανάλογα με τις συχετίσεις που περιγράφουμε παραπάνω.

- DDL – DML Script

Όσο αναφορά τους πίνακες έχουμε τα παρακάτω. Βλέπουμε τα αντίστοιχα attributes, τις δηλώσεις των τύπων για κάθε attribute, όπως και επίσης τα χαρακτηριστικά που δεν μπορούν να είναι null:

```
CREATE DATABASE SCHOOL;

use SCHOOL;

CREATE TABLE School_Unit (
Sname varchar(255) not null,
Address varchar(255) not null,
Town varchar(255) not null,
Telephone varchar(13) not null,
email varchar(255) not null,
Principal varchar(255) not null,
Handlerr varchar(255) not null,
PRIMARY KEY (Sname)
);


show tables;
show columns from School_Unit;

CREATE TABLE Books (
ISBN varchar(255) not null,
Title varchar(255) not null,
Author varchar(255) not null,
Publisher varchar(255) not null,
Npages int(10) not null,
Summary varchar(255) ,
Ncopies int(10) not null,
Acopies int(10) not null,
Image varchar(255),
Theme varchar(255) not null,
Blanguage varchar(255),
word_keys varchar(255),
PRIMARY KEY (ISBN)
```

```
);


show columns from Books;

CREATE TABLE Main_Handler(
Username varchar(255),
Passwords int(10),
Firstn varchar(255),
Lastn varchar(255),
PRIMARY KEY (Username, Passwords)
);

show columns from Main_Handler;



CREATE TABLE Handlers(
Username varchar(255),
Passwords int(10),
Firstname varchar(255),
Lastname varchar(255),
PRIMARY KEY (Username, Passwords)
);



CREATE TABLE Users(
Username varchar(255),
Passwords int(10),
isteacher bool,
name varchar(255),
surname varchar(255),
PRIMARY KEY (Username, Passwords)
);


show tables;

CREATE TABLE Review(
```

```
Rtext varchar(255),
Climax int(10),
ISBN varchar(255),
Username varchar(255),
FOREIGN KEY (ISBN) references Books(ISBN),
FOREIGN KEY (Username) references Users(Username)
);
#isteacher
CREATE TABLE Rents(
StartD datetime,
Returnd datetime,
due_d datetime,
Users varchar(255),
ISBN varchar(255),
S_unit varchar(255),
FOREIGN KEY (Users) REFERENCES Users(Username),
FOREIGN KEY (ISBN) REFERENCES Books(ISBN),
foreign key(S_unit) references School_Unit(Sname)
);

CREATE TABLE Reservations(
Applications varchar(255),
SDay datetime,
Ddat datetime,
Usern varchar(255),
ISBN varchar(255),
foreign key (Usern) references Users(Username),
foreign key (ISBN) references Books(ISBN)
);


ALTER TABLE School_Unit
ADD CONSTRAINT H_user
FOREIGN KEY (Handlerr) REFERENCES Handlers(Username);

ALTER TABLE School_Unit
ADD COLUMN MHuser varchar(255);

ALTER TABLE School_Unit
ADD CONSTRAINT FK_MH
```

```
FOREIGN KEY (MHuser) REFERENCES Main_Handler(Username);



ALTER TABLE Books
ADD COLUMN Sunit varchar(255);

ALTER TABLE Books
ADD CONSTRAINT FK_Sunit
FOREIGN KEY (Sunit) REFERENCES School_Unit(Sname);

ALTER TABLE Books
ADD COLUMN Huser varchar(255);

ALTER TABLE Books
ADD CONSTRAINT FK_Huser
FOREIGN KEY (Huser) REFERENCES Handlers(Username);


ALTER TABLE Handlers
ADD COLUMN Sname varchar(255);

ALTER TABLE Handlers
ADD CONSTRAINT Sname_users
FOREIGN KEY (Sname) REFERENCES School_Unit(Sname);


ALTER TABLE Handlers
ADD COLUMN Mhandler varchar(255);

ALTER TABLE Handlers
ADD CONSTRAINT FK_Mhandler
FOREIGN KEY (Mhandler) REFERENCES Main_Handler(Username);

alter table Users add index (isteacher);
```

Αντίστοιχα για τα indexes(Τα χρησιμοποιούμε για να γίνουν πιο γρήγορα τα αντίστοιχα queries):

```
use SCHOOL;

create index idx_name1
on Main_Handler(Username);

create index idx_name2
on Users(Username);

create index idx_age
on Users(Age);

create index idx_name3
on Handlers(Username);

create index idx_isbn
on Books(ISBN);

create index idx_theme
on Books(Theme);

create index idx_author
on Books(Author);
```

Τέλος για τα constraints έχουμε:

```
USE SCHOOL;

SHOW TABLES;
ALTER TABLE Handlers add constraint unique (Username);

ALTER TABLE Main_Handler add constraint unique (Username, Passwords);

ALTER TABLE Users add constraint unique (Username, Passwords);
```

```
ALTER TABLE Books add constraint unique(ISBN);

ALTER TABLE Rents add constraint check(StartD < due_d);

ALTER TABLE School_Unit add constraint unique (Sname);

ALTER TABLE Review add constraint check (Climax < 10);

alter table Users add constraint check (
(isteacher = true AND Nrents <=1) or
(isteacher = false and Nrents <=2)
);
```

- USER MANUAL

Η εφαρμογή αναπτύχθηκε σε Python 3 με χρήση του MySQL driver. Επιλέξαμε αυτή τη λύση διότι είναι προγραμματιστικά εύκολη και μας προσφέρει μεγάλη ευελιξία. Για τη διαχείρηση της βάσης χρησιμοποιήθηκε το MySQL Workbench. Για το γραφικό περιβάλλον (frontend) έχουμε χρησιμοποιήσει HTML. Η εφαρμογή αποτελείται από σελίδες (views) οι οποίες χειρίζονται GET και POST requests. Συγκεκριμένα οι σελίδες που φορτώνονται προκειμένου να δείξουν στοιχεία υλοποιούν GET ενώ αυτές που διαχειρίζονται δεδομένα φορμών διαχειρίζονται μεθόδους POST. Οι πληροφορίες περνάνε μέσα από τις φόρμες οι οποίες γίνονται submit (POST request) και εν συνεχεία αναλαμβάνει η Python για να τρέξει τα queries που χρειάζονται στη βάση και να άλλες λειτουργίες του backend. Εκεί γίνεται επίσης χειρισμός εξαιρέσεων από την Python για διάφορες εξαιρέσεις και την εμφάνιση καταλλήλων μηνυμάτων λάθους στο χρήστη. Ο connector της βάσης αρχικοποιείται με την έναρξη του application, συνδέεται στη βάση και μετά από αυτό δημιουργούμε έναν cursor ο οποίος είναι υπεύθυνος για το τρέξιμο των queries και την λήψη πληροφοριών (fetch) από τη βάση σε περίπτωση που κάνουμε ερωτήσεις (π.χ. select) ή commit σε περίπτωση που επιρεάζουμε τα περιεχόμενα της βάσης. Για να διασφαλιστεί η απαιτούμενη ασφάλεια από λανθασμένες ενέργειες υπάρχουν έλεγχοι τόσο στο frontend (στις φόρμες), στο backend (conditions).

- Εγκατάσταση Εφαρμογής σε Linux

1. Εγκαθιστούμε τον MySQL Server στο σύστημα μας με τη βοήθεια των εντολών
   sudo apt-get update && sudo apt-get install mysql-server && mysql_secure_installation.
   Στην παραπάνω διαδικασία θα ερωτηθούμε να θέσουμε κάποιους κωδικούς για να έχουμε πρόσβαση στον MySQL server. Αν θέλουμε να τον ξεκινήσουμε τρέχουμε
   sudo service mysql start
   Μπορούμε τέλος να κάνουμε login στον MySQL server με την ακόλουθη εντολή
   mysql -u {username} -p
   Αφού βεβαιωθούμε ότι ο MySQL server δουλεύει σωστά, μπορούμε να προχωρήσουμε στην διαδικασία δημιουργίας της βάσης.

2. Κάνουμε login στον MySQL server και δημιουργούμε τη βάση
   Create database SCHOOL;
   Use SCHOOL;
   Αντίστοιχα δίνονται οι πίνακες σε αρχείο sql όπως και τα indexes και τα constraints. Ανοίγοντας τα παραπάνω στο workbench και κάνοντας execute μπορούμε να δημιουργήσουμε την βάση. Αντίστοιχα χρησιμοποιούμε ένα αρχείο για τα insertions ώστε να βάλουμε τα δεδομένα που δημιουργήσαμε στην βάση. Το συγκεκριμένο αρχείο έχει ΕΝΔΕΙΚΤΙΚΑ κάποια insertions και δεν έχει όλες τις καταχωρήσεις δεδομένων. Για να δούμε ότι λειτουργεί η βάση μας αποδοτικά κάναμε και άλλα insertions αλλά manual.

3. Έχοντας στήσει τη βάση, προχωράμε στο στήσιμο του περιβάλλοντος πάνω στο οποίο τρέχει η εφαρμογή που έχουμε υλοποιήσει. Σε ένα τερματικό εγκαθιστούμε τα απαραίτητα πακέτα
   sudo apt-get install python3 python3-dev python3-flask python3-pip python3-mysql.connector

4. Έπειτα εκτελούμε την εντολή run flask στο τερματικό με το directory στο αρχείο app.py και στα Templates που πρέπει να βρίσκονται στον ίδιο φάκελο.

- Git Repo Εφαρμογής

https://github.com/alexppapp/Databases23
gh repo clone alexppapp/Databases23