

ΠΡΟΧΩΡΗΜΕΝΑ ΘΕΜΑΤΑ
ΤΕΧΝΟΛΟΓΙΑΣ ΚΑΙ ΕΦΑΡΜΟΓΩΝ
ΒΑΣΕΩΝ ΔΕΔΟΜΕΝΩΝ

ΠΡΟΓΡΑΜΜΑΤΙΣΤΙΚΗ ΕΡΓΑΣΙΑ ΓΙΑ ΤΟ
ΑΚΑΔΗΜΑΪΚΟ ΈΤΟΣ **2020-2021**

ΟΜΑΔΑ

ΔΗΜΗΤΡΗΣ ΚΟΛΙΑΤΟΣ, 3252

ΓΙΩΡΓΟΣ ΓΙΑΤΣΟΣ, 3202

ΙΩΑΝΝΗΣ ΚΟΥΤΡΟΥΔΗΣ, 3258

ΤΕΛΙΚΗ ΑΝΑΦΟΡΑ

ΜΑΪΟΣ 2021

Το πρόβλημα που αφορά την προγραμματιστική άσκηση (project) του μαθήματος είναι η οπτικοποίηση δεδομένων. Στόχος των τεχνικών οπτικοποίησης που υλοποιήσαμε είναι να δώσουν στο χρήστη την πληροφορία με τρόπο που αναδεικνύει οπτικά ιδιότητες, τάσεις και πρότυπα που βρίσκονται κρυμμένα στα δεδομένα. Ο κύριος λόγος της ενασχόλησής μας με την υλοποίηση του project είναι η αξιοποίηση μεγάλου όγκου δεδομένων και η παρουσίαση αυτού με τρόπο διαδραστικό και ενδιαφέρον για το χρήστη.

Για την εργασία ήταν απαραίτητη η χρήση της ιστοθεσίας <https://www.gapminder.org/data/> από την οποία διαλέξαμε 10 δείκτες της υποκατηγορίας Work, δύο κοινωνικούς δείκτες αναφοράς και δύο επιπλέον δείκτες με στοιχεία αναφοράς που διαλέξαμε εμείς. Συγκεκριμένα, οι 10 δείκτες της υποκατηγορίας Work είναι οι εξής:

1. Female industry workers (% of female employment)
2. Female service workers (% of female employment)
3. Female long term unemployment rate (%)
4. Male industry workers (% of male employment)
5. Male service workers (% of male employment)
6. Male long term unemployment rate (%)
7. Industry workers (% of employment)
8. Service workers (% of employment)
9. Long term unemployment rate (%)
10. Manufacturing employment as a percentage of total employment (%)

Οι δύο κοινωνικοί δείκτες αναφοράς είναι οι εξής:

1. Happiness score (WHR)
2. Human Development Index (HDI)

Και τέλος οι δύο δείκτες που διαλέξαμε ώστε να κάνουμε αναλύσεις αργότερα είναι οι παρακάτω:

1. Industry (% of GDP)
2. Services (% of GDP)

Αφού στήσαμε τη MySQL & MySQL Workbench στο μηχάνημά μας και κατεβάσαμε τα αρχεία με τα παραπάνω δεδομένα του οργανισμού Gapminder σχεδιάσαμε τη μορφή της βάσης μας. Έπειτα, υλοποιήσαμε transformation και loading scripts που μετατρέπουν τα εισερχόμενα αρχεία σε αρχεία φόρτωσης δεδομένων και τα φορτώνουν στη βάση μας αντίστοιχα. Στη συνέχεια, αξιοποιούμε τα δεδομένα για την εξαγωγή συμπερασμάτων με τη χρήση ερωτήσεων που αργότερα θα επιλέξει ο χρήστης μέσω μιας ιστοσελίδας. Η οπτικοποίηση των δεδομένων γίνεται σε διάφορες μορφές διαγραμμάτων. Τέλος, στην ιστοσελίδα έχουμε μενού επιλογής και χρησιμοποιούμε γραφικό τρόπο αλληλεπίδρασης ώστε να πάρουμε από το χρήστη τις πληροφορίες που θα οδηγήσουν στις αναπαραστάσεις που επιθυμεί να δει.

ΙΣΤΟΡΙΚΟ ΠΡΟΗΓΟΥΜΕΝΩΝ ΕΚΔΟΣΕΩΝ

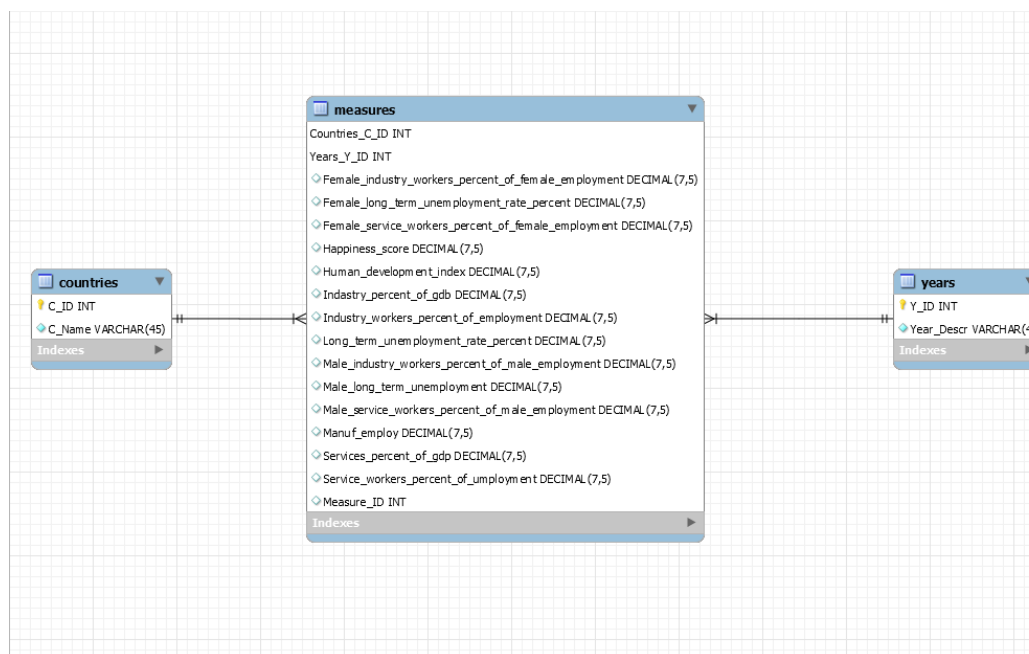
| Ημερομηνία | Έκδοση | Περιγραφή | Συγγραφέας |
|------------|--------|---|------------|
| 2021/03/25 | 0.1 | Δημιουργία και φόρτωση βάσης (Φάση I) | Ομάδα |
| 2021/04/22 | 0.2 | Αρχική έκδοση της εργασίας (Φάσεις I, II, III) | Ομάδα |
| 2021/05/27 | 1.0 | Τελική έκδοση της εργασίας (Φάσεις I, II, III) | Ομάδα |

1 ΒΑΣΗ ΔΕΔΟΜΕΝΩΝ

Για το μέγεθος και την πολυπλοκότητα των δεδομένων της βάσης μας χρειαστήκαμε ένα σχήμα μιας και είχαμε να κάνουμε με τρεις τύπους πληροφοριών χώρες, χρονολογίες και μετρήσεις.

1.1 ΣΧΕΣΙΑΚΟ ΣΧΗΜΑ ΣΕ ΛΟΓΙΚΟ ΕΠΙΠΕΔΟ

Το σχεσιακό σχήμα της βάσης δεδομένων αναπαρίσταται στο παρακάτω σχήμα:



Στο σχήμα αυτό διακρίνουμε τους πίνακες που χρησιμοποιούμε κατά την διάρκεια του project. Δημιουργούνται τρεις πίνακες ως εξής:

1. Countries

```
CREATE TABLE IF NOT EXISTS projectDB.Countries (  
    C_ID INT NOT NULL AUTO_INCREMENT,  
    C_Name VARCHAR(45) NOT NULL,  
    PRIMARY KEY (C_ID),  
    UNIQUE INDEX C_Name_UNIQUE (C_Name ASC) )  
ENGINE = InnoDB
```

2. Years

```
CREATE TABLE IF NOT EXISTS projectDB.Years (  
    Y_ID INT NOT NULL AUTO_INCREMENT,  
    Year_Descr VARCHAR(4) NOT NULL,  
    PRIMARY KEY (Y_ID),  
    UNIQUE INDEX Year_Descr_UNIQUE (Year_Descr ASC) )  
ENGINE = InnoDB
```

3. Measures

```
CREATE TABLE IF NOT EXISTS projectDB.Measures (  
    Countries_C_ID INT NOT NULL,  
    Years_Y_ID INT NOT NULL,  
    Female_industry_workers_percent_of_female_employment DECIMAL(7,5)  
    NULL,  
    Female_long_term_unemployment_rate_percent DECIMAL(7,5) NULL,  
    Female_service_workers_percent_of_female_employment DECIMAL(7,5)  
    NULL,  
    Happiness_score DECIMAL(7,5) NULL,  
    Human_development_index DECIMAL(7,5) NULL,  
    Industry_percent_of_gdp DECIMAL(7,5) NULL,  
    Industry_workers_percent_of_employment DECIMAL(7,5) NULL,  
    Long_term_unemployment_rate_percent DECIMAL(7,5) NULL,
```

```
Male_industry_workers_percent_of_male_employment DECIMAL(7,5) NULL,  
Male_long_term_unemployment DECIMAL(7,5) NULL,  
Male_service_workers_percent_of_male_employment DECIMAL(7,5) NULL,  
Manuf_employ DECIMAL(7,5) NULL,  
Services_percent_of_gdp DECIMAL(7,5) NULL,  
Service_workers_percent_of_unemployment DECIMAL(7,5) NULL,  
Measure_ID INT NULL,  
INDEX fk_Happiness_Countries_idx (Countries_C_ID ASC) ,  
INDEX fk_Happiness_Years1_idx (Years_Y_ID ASC) ,  
PRIMARY KEY (Countries_C_ID, Years_Y_ID),  
CONSTRAINT fk_Happiness_Countries  
FOREIGN KEY (Countries_C_ID)  
REFERENCES projectDB.Countries (C_ID)  
ON DELETE NO ACTION  
ON UPDATE NO ACTION,  
CONSTRAINT fk_Happiness_Years1  
FOREIGN KEY (Years_Y_ID)  
REFERENCES projectDB.Years (Y_ID)  
ON DELETE NO ACTION  
ON UPDATE NO ACTION)  
ENGINE = InnoDB
```

1.2 ΣΧΕΣΙΑΚΟ ΣΧΗΜΑ ΣΕ ΦΥΣΙΚΟ ΕΠΙΠΕΔΟ

Οι αλλαγές που κάναμε σε φυσικό επίπεδο είναι ότι τροποποιήσαμε τον περιορισμό γραμμών για να χωρέσει τα δεδομένα μας.

1.2.1 ΡΥΘΜΙΣΗ ΤΩΝ ΠΑΡΑΜΕΤΡΩΝ ΤΟΥ DBMS

Χρησιμοποιήσαμε για μηχανή αποθήκευσης(storage engine) για το DBMS μας την InnoDB η οποία ουσιαστικά αποτελεί την by default μηχανή αποθήκευσης.

1.2.2 ΡΥΘΜΙΣΗ ΤΟΥ ΦΥΣΙΚΟΥ ΣΧΗΜΑΤΟΣ ΤΗΣ ΒΑΣΗΣ ΔΕΔΟΜΕΝΩΝ

Όσον αφορά τον ορισμό πιθανών ευρετηρίων(indexes) και όψεων(views) δεν κάναμε κάποια τροποποίηση και επειδή δεν αντιμετωπίσαμε κάποιο ιδιαίτερο εμπόδιο δεν χρειάστηκε να

κάνουμε καμία αλλαγή στο μέγεθος είτε στο σχήμα των πινάκων και επομένως τα αφήσαμε όλα στις default καταστάσεις τους.

1.2.3 ΡΥΘΜΙΣΗ ΑΣΦΑΛΕΙΑΣ

Αρχικά, με την εγκατάσταση της MySQL στο μηχάνημά μας έπρεπε να ορίσουμε έναν κωδικό. Έπειτα, λόγω του μεγάλου όγκου δεδομένων που χρειάστηκε να φορτώσουμε στη βάση μας με τη χρήση της εντολής `LOAD DATA LOCAL INFILE` έπρεπε να αλλάξουμε την τιμή `set global local_infile` σε 1. Ενώ το αρχείο προορισμού μας `final.csv` έπρεπε να είναι στο προστατευόμενο φάκελο προορισμού που έχει θέσει ως προκαθορισμένο η MySQL, κατά την εγκατάστασή της. Ο έλεγχος αυτού έγινε με την εντολή `mysql>show variables like "secure_file_priv";`.

2 ΑΡΧΙΤΕΚΤΟΝΙΚΗ ΛΟΓΙΣΜΙΚΟΥ

2.1 ΑΡΧΙΤΕΚΤΟΝΙΚΗ ΚΑΙ ΔΟΜΗ ETL

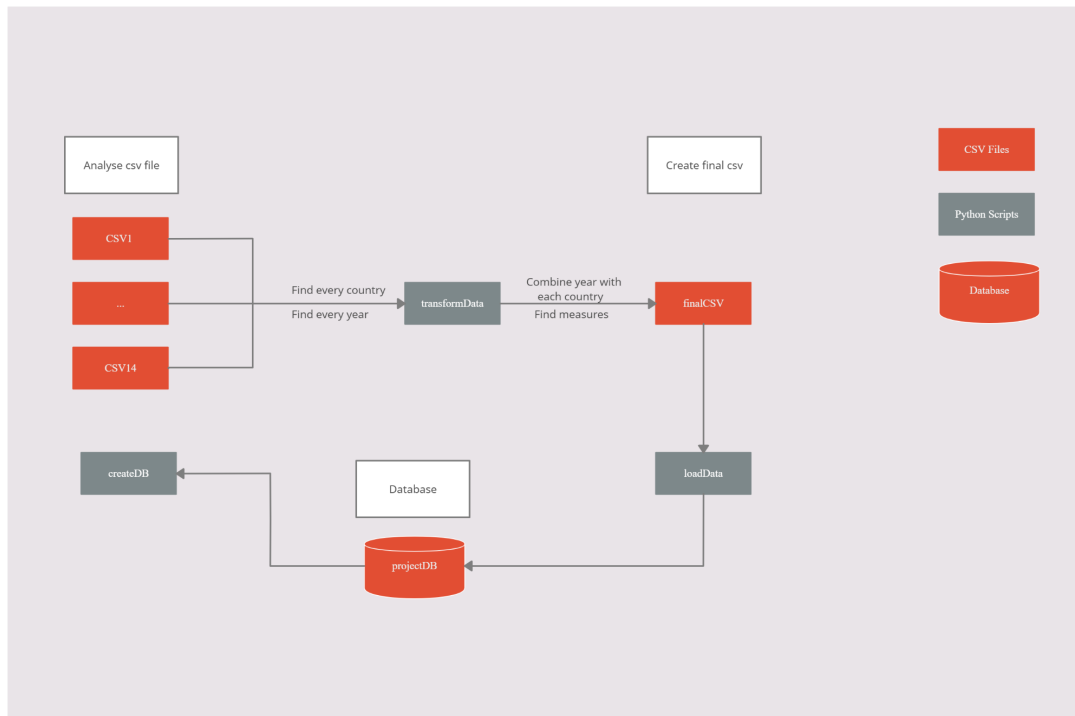
Αφού κατεβάσουμε τα csv αρχεία που θα χρησιμοποιήσουμε από την ιστοσελίδα Gapminder για τη φόρτωση των δεδομένων τους στη βάση μας υλοποιούμε τρία Python scripts.

Το πρώτο ονομάζεται `createDB.py` στο οποίο συνδεόμαστε μέσω του `python database connector` στη βάση δίνοντας τα κατάλληλα `host`, `user` και `password` για να συνδεθούμε στο σέρβερ μας. Εκτός από αυτή τη λειτουργία στο script αυτό δημιουργούμε και τους τρεις πίνακές μας.

Η σχεδιαστική λύση που σκεφτήκαμε για την υλοποίηση της βάσης μας αφορά τρεις πίνακες όπως προαναφέραμε. Οι δύο από αυτούς είναι τύπου `lookup` πίνακα και έχουν να κάνουν με τις χώρες και τα έτη ο καθένας ξεχωριστά. Ο τρίτος πίνακάς μας είναι τύπου `fact` και περιέχει πεδία για κάθε ένα δείκτη. Η χρήση των `lookup` πινάκων γίνεται με σκοπό τη σύνδεση κάθε δείκτη με μια συγκεκριμένη χρονιά και χώρα. Η υλοποίηση αυτή μπορεί να είναι πιο αποδοτική, αλλά δεν είναι χρήσιμη για τη συγκεκριμένη περίπτωση που έχουμε μόνο ένα `primary key` και ένα επιπλέον πεδίο στους `lookup` πίνακες, αφού θα μπορούσαμε να χρησιμοποιήσουμε μόνο έναν πίνακα. Επιπρόσθετα, επιλέξαμε να βάλουμε όλους τους δείκτες μέσα σε έναν πίνακα αντί να δημιουργήσουμε για τον καθένα δείκτη έναν ξεχωριστό πίνακα. Με αυτόν τον τρόπο μπορεί να αποκτούμε έναν πιο αργό σύστημα διαχείρισης της βάσης μας, όμως στην περίπτωσή μας εξαιτίας του όγκου δεδομένων μας προτιμήσαμε τη μικρότερη πολυπλοκότητα στις ερωτήσεις σχεσιακής άλγεβρας.

Το δεύτερο script ονομάζεται `transformData.py` και σε αυτό βρίσκουμε και παίρνουμε από κάθε csv αρχείο όλες τις χώρες και όλα τα έτη και τα ταξινομούμε. Συνδυάζοντάς τα και ταυτόχρονα βρίσκοντας για τον καθένα συνδυασμό (χώρα-έτος) την αντίστοιχη τιμή από κάθε δείκτη δημιουργούμε το τελικό μας csv αρχείο (`final`).

Το τρίτο script μας ονομάζεται loadData.py και αφού ξανασυνδεθούμε στη βάση μας μέσω του Python database connector μας φορτώνουμε τα δεδομένα στον fact πίνακά μας με τη βοήθεια του final.csv αρχείου μας μέσω της εντολής της MySQL **LOAD DATA LOCAL INFILE**. Ενώ με τους δύο βοηθητικούς πίνακες υποδοχής του script μας, που περιέχουν τις χώρες και τα έτη ταξινομημένα, φορτώνουμε τους lookup πίνακες της βάσης μας.



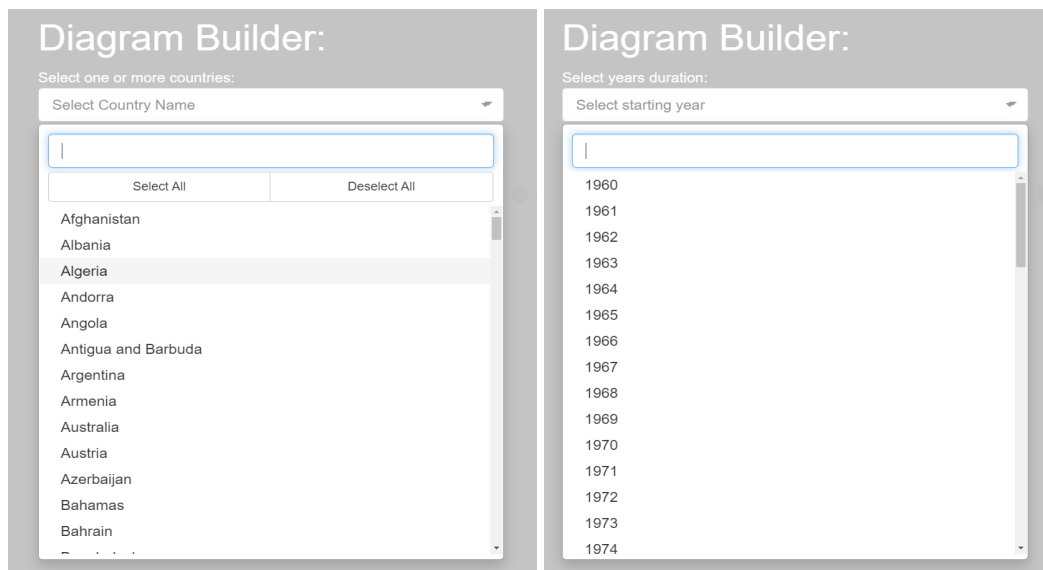
2.2 ΔΙΑΓΡΑΜΜΑΤΑ ΠΑΚΕΤΩΝ / ΥΠΟΣΥΣΤΗΜΑΤΩΝ ΚΕΝΤΡΙΚΗΣ ΕΦΑΡΜΟΓΗΣ

Στην υλοποίηση της κεντρικής μας εφαρμογής χρησιμοποιήσαμε Python για την τροποποίηση των δεδομένων μας σε κατάλληλη μορφή, ώστε με τη χρήση του micro web framework Flask να ανταλλάσσουμε δεδομένα με την ιστοσελίδα μας. Με τη βοήθεια του framework Flask παίρνουμε τις πληροφορίες που επιλέγει-ζητά ο χρήστης από την ιστοσελίδα μας και έπειτα υλοποιούμε τις ερωτήσεις αυτές στη βάση μας με τη βοήθεια του Python database connector. Στη συνέχεια, αφού πάρουμε τις απαντήσεις μας από τη βάση, τις επεξεργαζόμαστε καταλλήλως και τις στέλνουμε στην ιστοσελίδα μας. Εξαιτίας αυτής της υλοποίησης δεν υπάρχουν διαγράμματα πακέτων και κλάσεων που αφορούν τη high-level αρχιτεκτονική.

3 ΥΠΟΔΕΙΓΜΑΤΑ ΕΡΩΤΗΣΕΩΝ ΚΑΙ ΑΠΑΝΤΗΣΕΩΝ

Μέσω του παρακάτω κομματιού κώδικα ρωτάμε τη βάση μας για όλες τις επιλογές χωρών και ετών που έπειτα τις απεικονίζουμε στην ιστοσελίδα μέσω dropbox.

```
@app.route('/DiagramBuilder')
def countries():
    cur = mysql.connection.cursor(MySQLdb.cursors.DictCursor)
    cur.execute("SELECT * FROM countries ORDER BY C_Name")
    countries = cur.fetchall()
    cur.execute("SELECT * FROM years ORDER BY Year_Descr")
    years = cur.fetchall()
    return render_template('diagramBuilder.html',countries=countries,years=years)
```



Μέσω του παρακάτω κώδικα παίρνουμε αρχικά τις επιλογές του χρήστη από την ιστοσελίδα για τη σχεδίαση του διαγράμματός μας.

```
@app.route('/diagram', methods=['GET', 'POST'])
def getSelectedCountries():
    if request.method == 'POST':
        countries = request.form.getlist('C_Name')
        year1 = request.form['Year_Descr']
        year2 = request.form['Year_Descr2']
        measureId = request.form.getlist('measure')
        measureType = []
        for id in measureId:
            measureType.append(idToSelection(id))
        diagramId = request.form['diagram']
        timePeriodId = request.form['time_period']
```


Αφού έχουμε πλέον τις επιλογές του, η μέθοδος `prepareDataForCharts` εκτελεί ερώτηση (query) στη βάση, όπως φαίνεται στην εικόνα παρακάτω. Στη συνέχεια, μαζεύει τις τιμές των δεδομένων που παίρνουμε ως απαντήσεις από τη βάση και τα τροποποιούμε αναλόγως κάθε φορά με τις απαιτήσεις του χρήστη. Με τη μέθοδο `timePeriodChanges` τα ομαδοποιούμε σύμφωνα με το χρονικό εύρος που μας ζητήθηκε. Ενώ, με τη μέθοδο `findTextFromID` μετατρέπουμε τα `id` που έχει κάθε χώρα, δείκτης και έτος στην κατάλληλη τους ονομασία. Με τις `json.dumps` μεθόδους στέλνουμε πληροφορίες στην ιστοσελίδα μας. Τέλος, μέσω του `diagramId` στη `return` μέθοδό μας καλούμε κάθε φορά την αντίστοιχη `html` σελίδα που έχουμε για το κάθε διάγραμμα.

```
timePeriodId = request.POST.get('time_period')
finalList = prepareDataForCharts(countries, year1, year2, measureType, diagramId)
TPC = timePeriodChanges(finalList, timePeriodId)
data = findTextFromID(TPC, measureType, diagramId)
measureName = titleFixer(measureType)
data = json.dumps(data)
measureName = json.dumps(measureName)
return diagram(data, measureName, diagramId)

query = '''
... SELECT * FROM measures WHERE Countries_C_ID = %s AND Years_Y_ID = %s
...
values = (c_id,y_id)
cur.execute(query,values)
```

4 ΤΕΚΜΗΡΙΩΣΗ ΚΑΙ ΛΟΙΠΑ ΣΧΟΛΙΑ

Μέσω της υλοποίησης της εργασίας μάθαμε να διαχειριζόμαστε και να πειραματιζόμαστε πάνω σε βάση δεδομένων για μεγάλο ή μικρό όγκο δεδομένων. Με τη χρήση `scripts` αυτοματοποιήσαμε τη διαδικασία αυτή και μπορέσαμε να δημιουργήσουμε μία βάση στα μέτρα των δεδομένων μας. Τέλος, φτιάξαμε την ιστοσελίδα μας ώστε το `front-end` της εφαρμογής μας να επικοινωνεί με το σέρβερ μας, να προετοιμάζει τις επιλογές που δίνει ο χρήστης και με τις κατάλληλες ερωτο-απαντήσεις να του επιστρέφουμε τα αποτελέσματα που ζήτησε και στη μορφή διαγραμμάτων που διάλεξε.

Μία δυσκολία που αντιμετωπίσαμε ήταν ο συνδυασμός δύο δεικτών σε διάγραμμα `scatter plot` μιας και δεν καταφέραμε την κατάλληλη προετοιμασία για τη σωστή οπτικοποίηση των δεδομένων μας.

Επίσης, στην περίπτωση πολλαπλών χωρών και πολλαπλών δεικτών στα διαγράμματα μπορεί να μην εμφανίζονται καθαροί και όλοι οι συνδυασμοί.

Επιπλέον, σε περίπτωση που τερματίσει απότομα το πρόγραμμά μας προτείνουμε να κάνετε `refresh` και να ξαναπροσπαθήσετε αν και έχουμε προσπαθήσει με περιορισμούς να αποτρέψουμε κάτι τέτοιο.

ΣΑΣ ΕΥΧΑΡΙΣΤΟΥΜΕ ΠΟΛΥ!