

ΠΡΟΧΩΡΗΜΕΝΑ ΘΕΜΑΤΑ
ΤΕΧΝΟΛΟΓΙΑΣ ΚΑΙ ΕΦΑΡΜΟΓΩΝ
ΒΑΣΕΩΝ ΔΕΔΟΜΕΝΩΝ

ΠΡΟΓΡΑΜΜΑΤΙΣΤΙΚΗ ΕΡΓΑΣΙΑ ΓΙΑ
ΤΟ ΑΚΑΔΗΜΑΪΚΟ ΕΤΟΣ 2022-2023

ΟΜΑΔΑ 2251_3247_2886

ΖΑΧΑΚΗΣ ΧΑΡΑΛΑΜΠΟΣ, 2251

ΚΑΤΣΙΛΙΕΡΗΣ ΑΘΑΝΑΣΙΟΣ, 3247

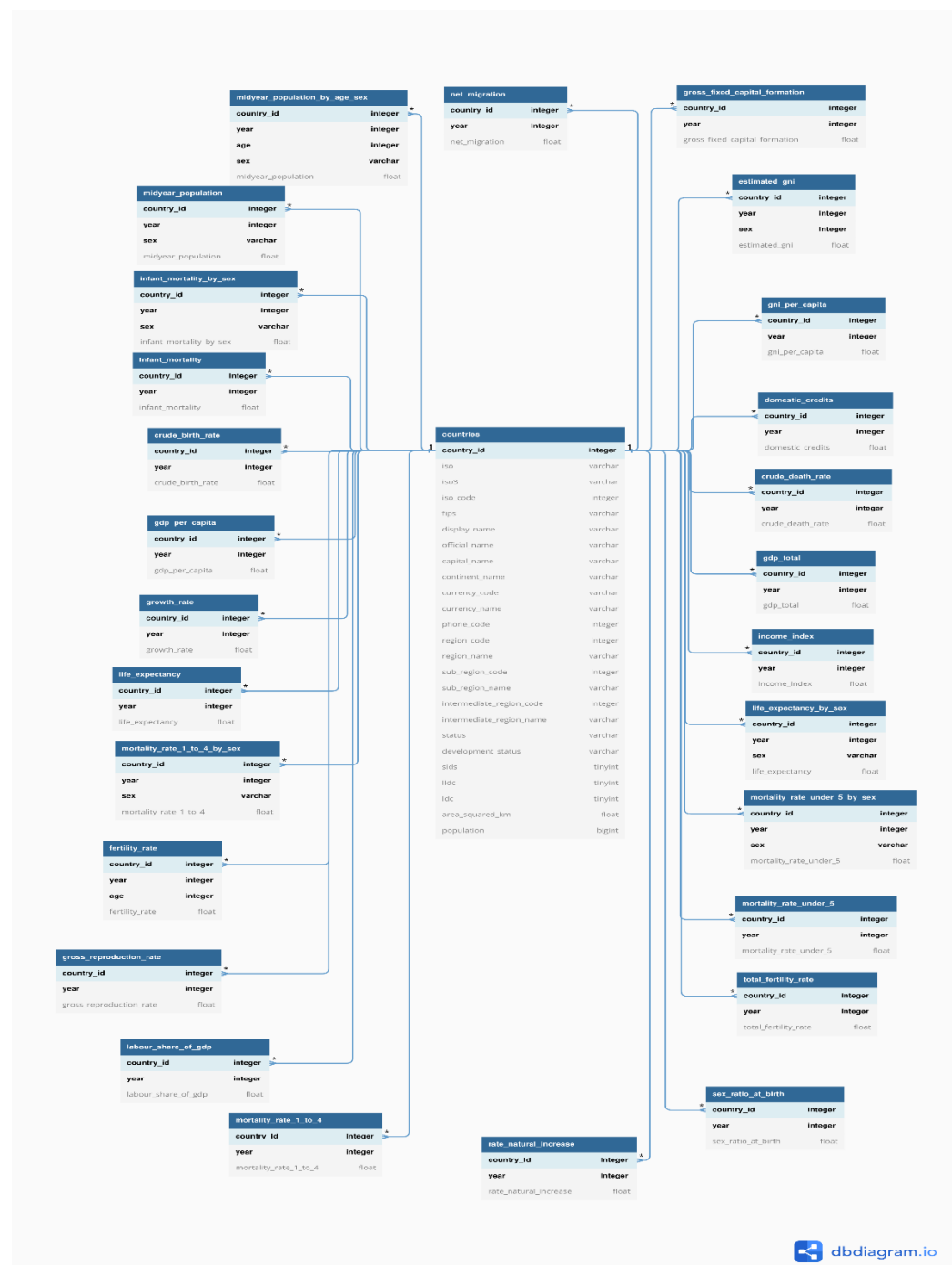
ΑΡΒΑΝΙΤΙΔΟΥ ΑΝΑΣΤΑΣΙΑ, 2886

ΤΕΛΙΚΗ ΑΝΑΦΟΡΑ

ΜΑΪΟΣ 2023

1 ΒΑΣΗ ΔΕΔΟΜΕΝΩΝ

1.1 ΣΧΕΣΙΑΚΟ ΣΧΗΜΑ ΣΕ ΛΟΓΙΚΟ ΕΠΙΠΕΔΟ



Το σχήμα με το structure της βάσης βρίσκεται στο github repository με τα etl scripts (φάκελος sql, περιέχει και backup αλλά και sql files με το structure κάθε επιμέρους table). Long story short έχουμε φέρει τη βάση σε 3nf μορφή. Στο countries table έχουμε βάλει auto increment int pk. Όλα τα stat tables έχουν foreign key στο countries.country_id. Κάθε stat table έχει

composite PK όλα τα columns του πέρα από το stat (float). Σε κάποια είναι (country_id, year), σε άλλα (country_id, year, age), σε άλλα (country_id, year, sex) και σε άλλα (country_id, year, age, sex)

1.2 ΣΧΕΣΙΑΚΟ ΣΧΗΜΑ ΣΕ ΦΥΣΙΚΟ ΕΠΙΠΕΔΟ

1.2.1 ΡΥΘΜΙΣΗ ΤΩΝ ΠΑΡΑΜΕΤΡΩΝ ΤΟΥ DBMS

- Storage Engine: InnoDB
- Memory allocation: 12GB (75% του συστήματος στο οποίο έτρεξε)

Όλα τα υπόλοιπα settings είναι default της MySQL

2 ΑΡΧΙΤΕΚΤΟΝΙΚΗ ΛΟΓΙΣΜΙΚΟΥ

2.1 ΑΡΧΙΤΕΚΤΟΝΙΚΗ ΚΑΙ ΔΟΜΗ ETL

Έχουμε φτιάξει ξεχωριστό project για το etl. Για κάθε μετρική έχουμε φτιάξει ένα model class, ένα parser class, ένα mapper class.

Sample model class:

```
@Getter
@Builder
@ETL(parserClass = FertilityRateParser.class, parseDirectory = "resources/age_specific_fertility_rates.csv", outputDirectory = "output/fertility_rates.txt")
public final class FertilityRate
{
    private Integer countryId;
    private Integer year;
    private Integer age;
    private Float fertilityRate;

    @Override
    public String toString()
    {
        return String.format("%d;%d;%d;%s", getCountryId(), getYear(), getAge(), getFertilityRate());
    }
}
```

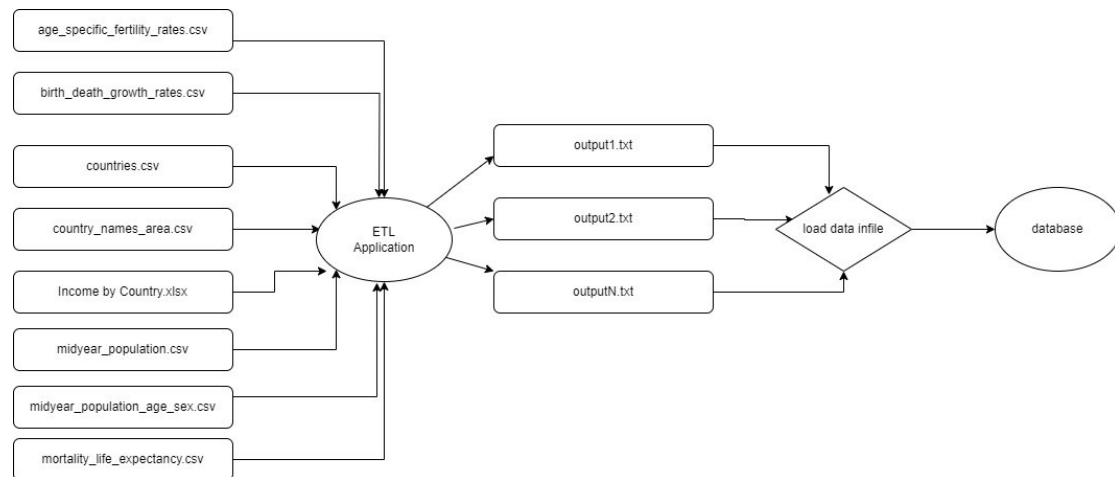
Το @Getter annotation είναι του Lombok και φτιάχνει getters για τα fields της κλάσης αυτής και το @Builder φτιάχνει inner static builder class (builder design pattern).

Κάθε entity class γράφεται σε txt αρχείο με το toString() format του.

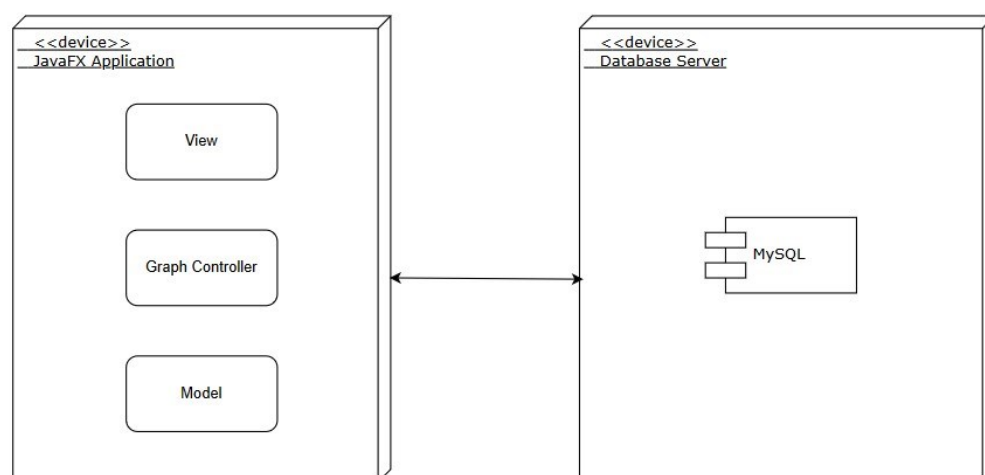
Το @ETL annotation είναι δικό μας custom annotation, κρατάμε metadata σε κάθε entity για το parser class του, το parse directory, και το output directory

```
3+ import java.lang.annotation.ElementType;
9
10 @Retention(RetentionPolicy.RUNTIME)
11 @Target(value = ElementType.TYPE)
12 public @interface ETL
13 {
14     Class<? extends Parser<?>> parserClass();
15
16     String parseDirectory();
17
18     String outputDirectory();
19 }
```

Long story short, όταν τρέξετε τη main, το application μας scanάρει σε όλο το project για @ETL annotated κλάσεις, διαβάζει τα αντίστοιχα αρχεία, τα κάνει map σε entities, και τα γράφει στο δίσκο. Στο φάκελο "sql/load data" του project έχουμε sql files με εντολές LOAD DATA INFILE που φορτώνουν τα data αυτά.



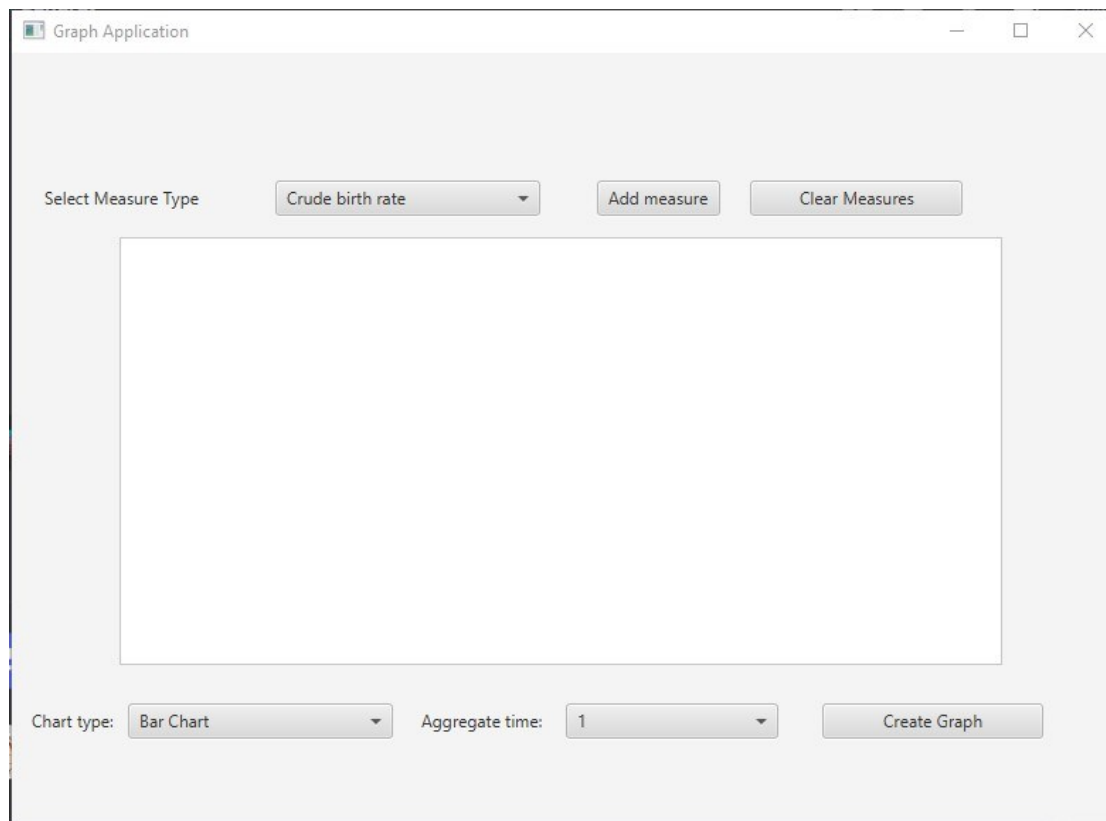
2.2 ΔΙΑΓΡΑΜΜΑΤΑ ΠΑΚΕΤΩΝ / ΥΠΟΣΥΣΤΗΜΑΤΩΝ ΚΕΝΤΡΙΚΗΣ ΕΦΑΡΜΟΓΗΣ



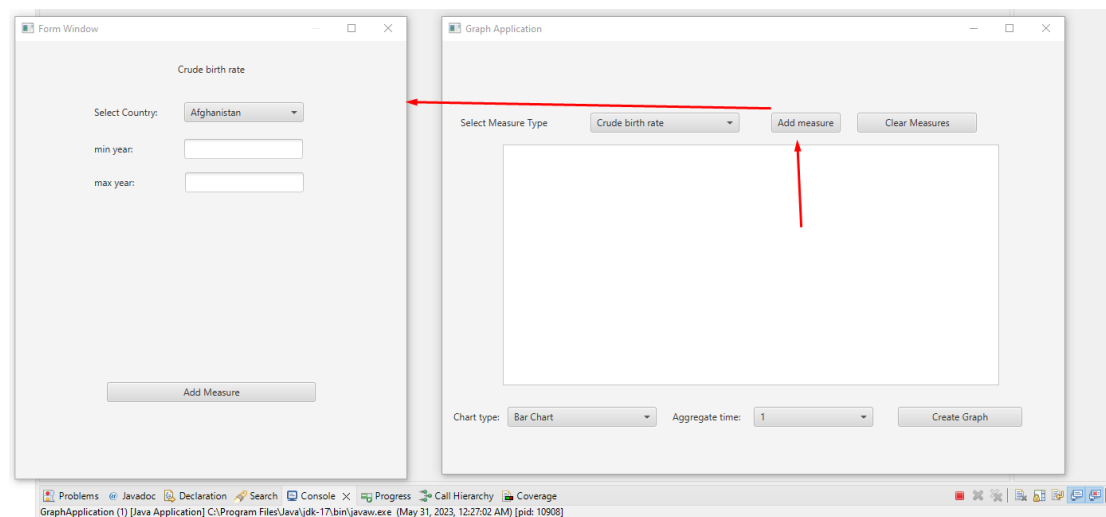
Έχουμε χρησιμοποιήσει MVC architecture στο ui application. Έχουμε χρησιμοποιήσει αρκετά design patterns όπως Factory, Singleton, Builder, Façade, Strategy, Null Object, Flyweight, Decorator. Πιο αναλυτικά τα εξηγούμε στο video.

6

3 ΥΠΟΔΕΙΓΜΑΤΑ ΕΡΩΤΗΣΕΩΝ ΚΑΙ ΑΠΑΝΤΗΣΕΩΝ

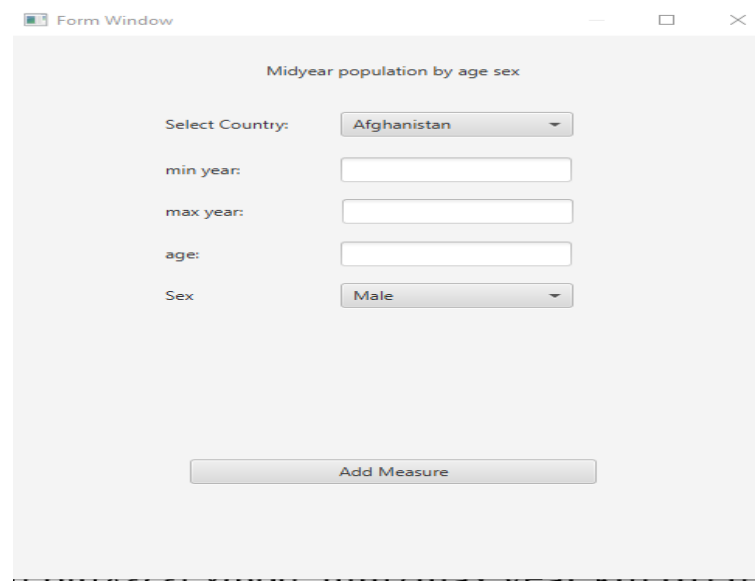


Στο dropdown menu “Select Measure Type” ο χρήστης μπορεί να διαλέξει μεταξύ 30 μετρικών. Έπειτα χρειάζεται να πατήσει το κουμπί add measure ώστε να του εμφανίσει μια φόρμα με τα πεδία που χρειάζεται να συμπληρώσει. Η φόρμα είναι δυναμική (εξαρτάται από τη μετρική)



Στη φόρμα θα πρέπει να διαλέξει χώρα, min/max year και ότι άλλο χρειαστεί, (σε άλλες φόρμες sex, σε άλλες age/sex κλπ).

Παραδείγματα από άλλα forms:



Form Window

Midyear population by age sex

Select Country: Afghanistan

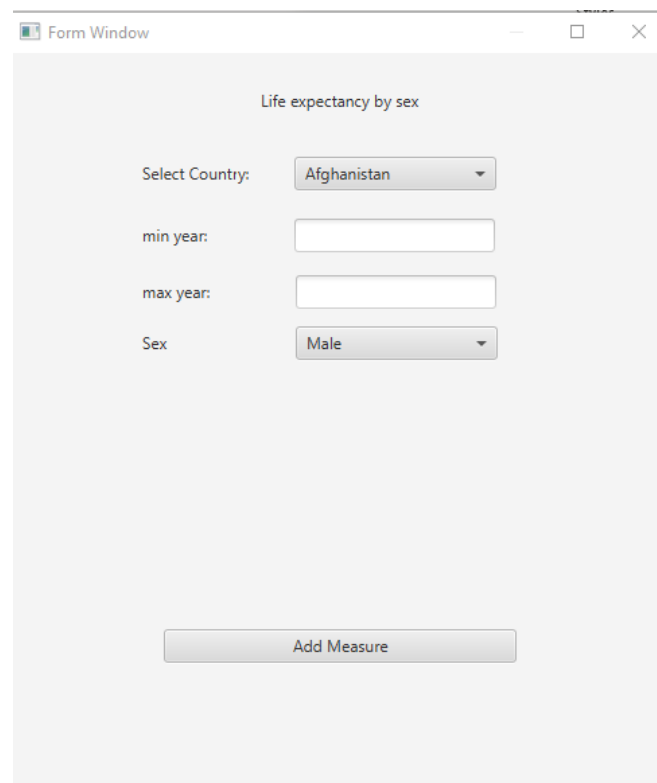
min year:

max year:

age:

Sex: Male

Add Measure



Form Window

Life expectancy by sex

Select Country: Afghanistan

min year:

max year:

Sex: Male

Add Measure

Όταν ο χρήστης πατήσει το κουμπί add measure θα προστεθεί το measure αυτό στο main window η μετρική

Graph Application

Select Measure Type: Estimated GNI Add measure Clear Measures

Afghanistan Crude birth rate, years(1991-2000)

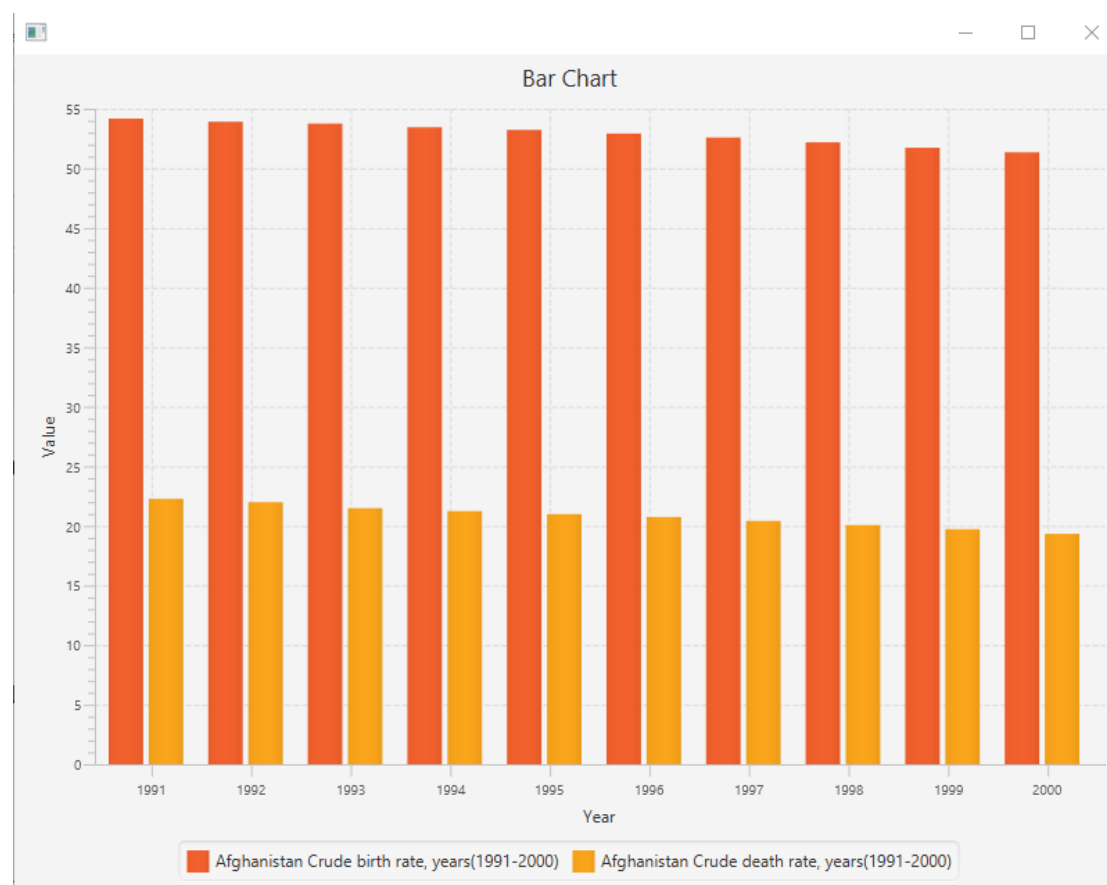
Afghanistan Crude death rate, years(1996-2010)

Chart type: Bar Chart Aggregate time: 1 Create Graph

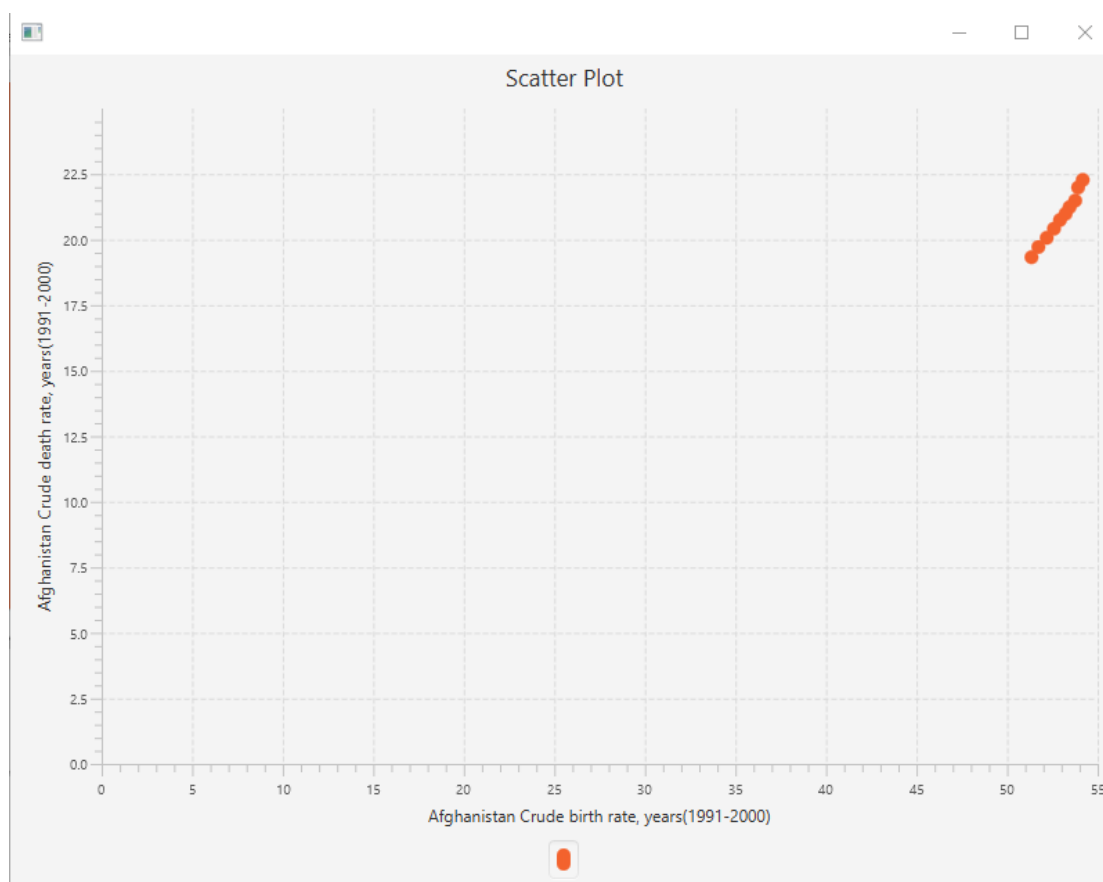
Έπειτα πρέπει να διαλέξει τον τύπο του chart το οποίο θέλει να δημιουργήσει (μεταξύ Bar Chart, Scatter Plot, Line Chart) και aggregate time (μεταξύ 1/5/10) ώστε να του κάνει display τα δεδομένα ανά 1/5/10 χρόνια. Τέλος θα πρέπει να πατήσει create graph. (To clear measures button είναι self explanatory)

Sample outputs :

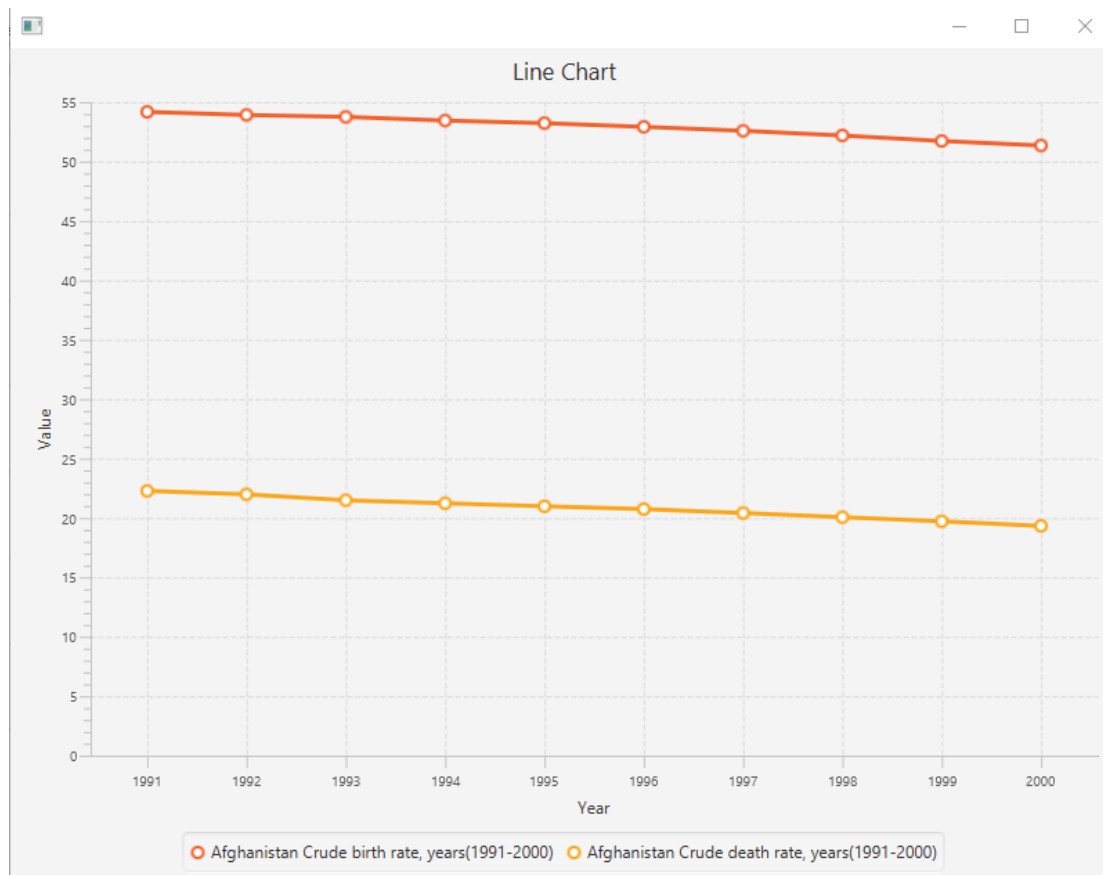
BarChart ανά χρονιά



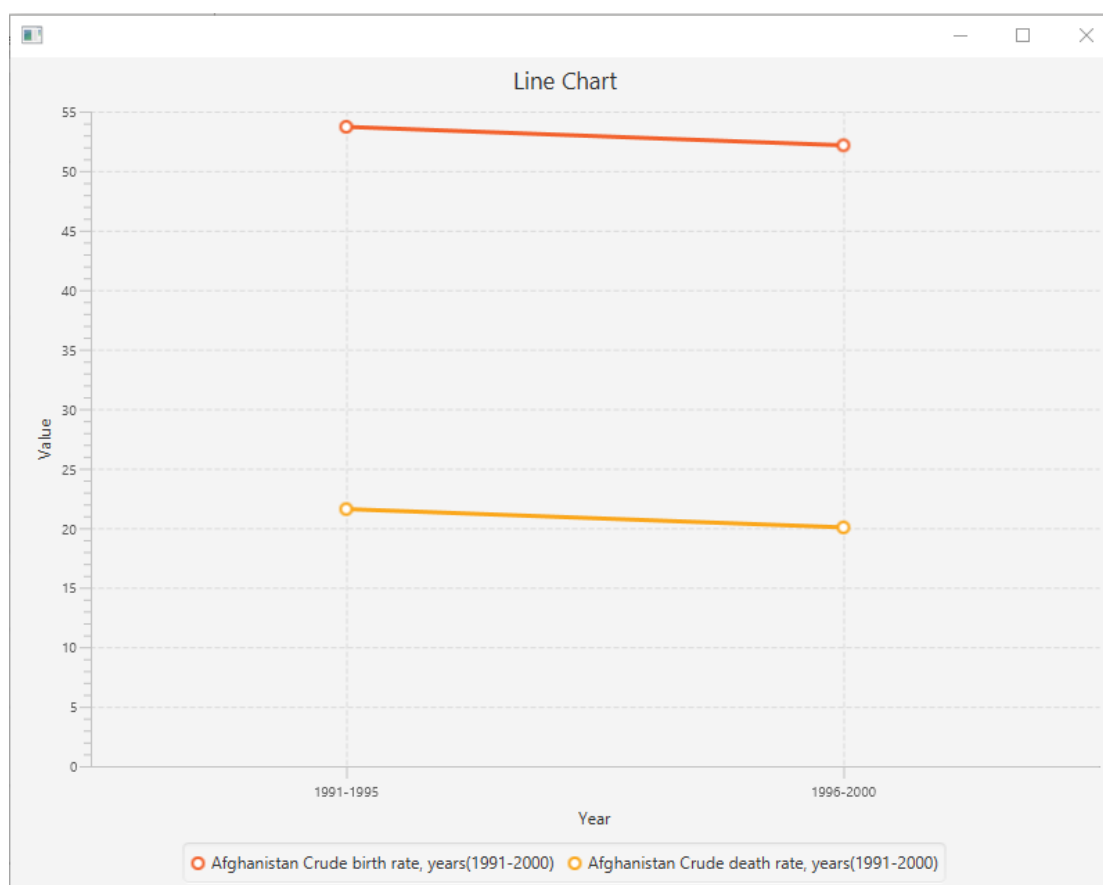
Scatter Plot ανά χρονιά



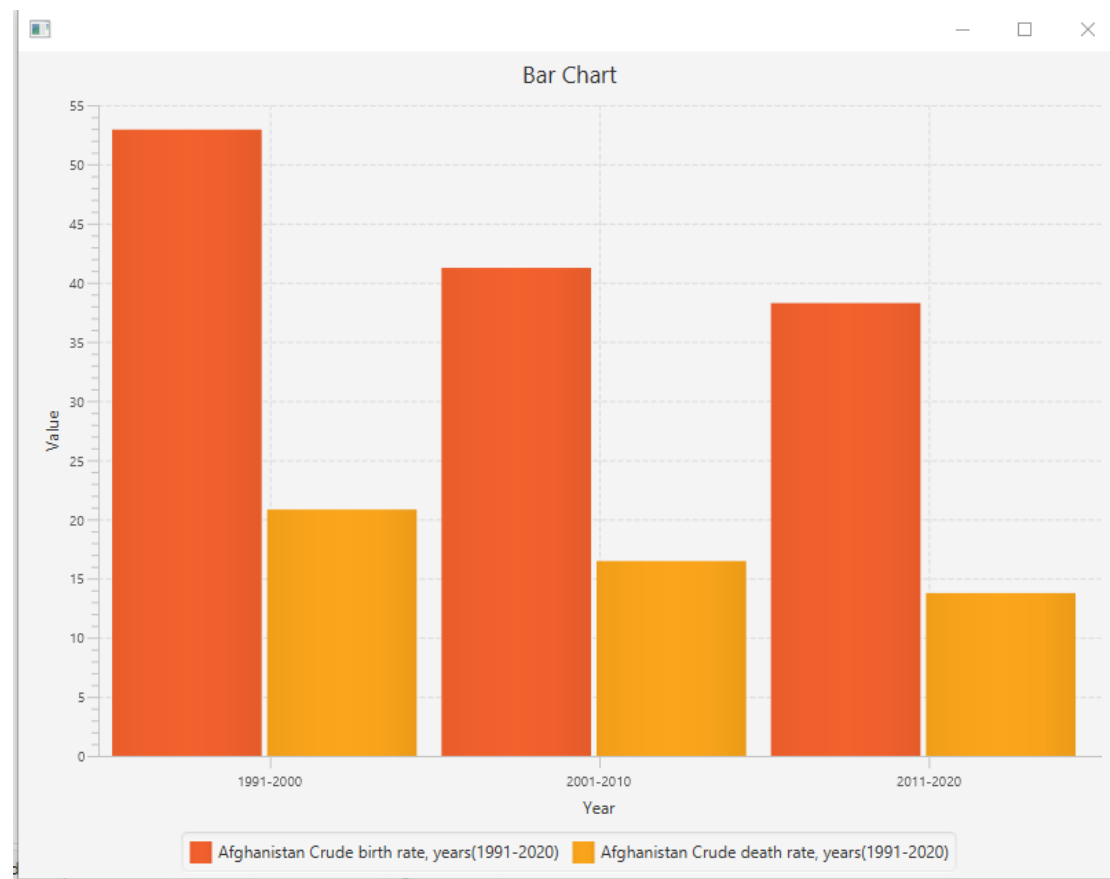
LineChart ανα χρονιά



LineChart ανά 5ετία



BarChart ανα 10ετία



- Λειτουργούν όλα τα aggregate times για όλων των ειδών τα charts.
- Στην περίπτωση του line chart πετάμε μήνυμα λάθους εάν ο χρήστης διαλέξει λιγότερες ή περισσότερες από 2 μετρικές
- Στην περίπτωση των Bar Chart/Line Chart πετάμε μήνυμα λάθους εάν δεν έχει βάλει καμία μετρική ο χρήστης στη λίστα

4 ΛΟΙΠΑ ΣΧΟΛΙΑ

- Σε περίπτωση που θέλετε να τρέξετε το πρόγραμμα θα πρέπει να κάνετε edit στο ConnectionFactory.java τα settings της db

```
package gr.uoi.cse.countrygraph.connection;  
  
import java.sql.Connection;  
  
public final class ConnectionFactory  
{  
    private static final String CONNECTION_URL = "jdbc:mysql://localhost:3306/country_stat_db?serverTimezone=UTC";  
    private static final String USERNAME = "root";  
    private static final String PASSWORD = "root";  
  
    public final Connection createConnection() throws SQLException  
    {  
        return DriverManager.getConnection(CONNECTION_URL, USERNAME, PASSWORD);  
    }  
  
    public static final ConnectionFactory getInstance()  
    {  
        return SingletonHolder.INSTANCE;  
    }  
  
    private static final class SingletonHolder  
    {  
        private static final ConnectionFactory INSTANCE = new ConnectionFactory();  
    }  
}
```

- Και στα 2 projects (etl, countrygraph) χρειάζεται JDK 17 (currently το τελευταίο LTS).
- Το etl project είναι plain java project ενώ το countrygraph είναι maven.
- Σε περίπτωση που θέλετε να το τρέξετε στο eclipse θα χρειαστεί να κάνετε setup το Lombok plugin (<https://projectlombok.org/setup/eclipse>), στο intellij εδώ και αρκετά χρόνια είναι built in.
- Σε περίπτωση που θέλετε να τρέξετε το etl project η main βρίσκεται στο package gr.uoi.cse.Main.java
- Σε περίπτωση που θέλετε να τρέξετε το countrygraph (η main βρίσκεται στο gr.uoi.cse.countrygraph.GraphApplication.java)
- Backups της βάσης (με και χωρίς data) βρίσκονται στο etl project (sql φάκελος)
- Σε περίπτωση που θέλετε να φορτώσετε τα txt data που έγιναν export από το etl πρόγραμμα, θα πρέπει να αλλάξετε paths στα sql files στο C:/path_to_etl_project/output/...

```
LOAD DATA LOCAL INFILE 'C:/Workspace/etl/output/countries.txt' IGNORE INTO TABLE countries  
FIELDS TERMINATED BY ','  
LINES TERMINATED BY '\r\n'  
(country_id, iso, iso3, iso_code, fips, display_name, official_name, capital_name, continent_name, cur)
```

5 EXTRA ΕΡΩΤΗΜΑ 2023 (ΜΕΤΡΗΣΕΙΣ)

- Εξέλιξη μεγεθών στο χρόνο

Crude Birth Rate:

Εκφράζει την αναλογία γεννήσεων μιας χώρας σε μια συγκεκριμένη χρονιά ανα 1000 κατοίκους του πληθυσμού της.

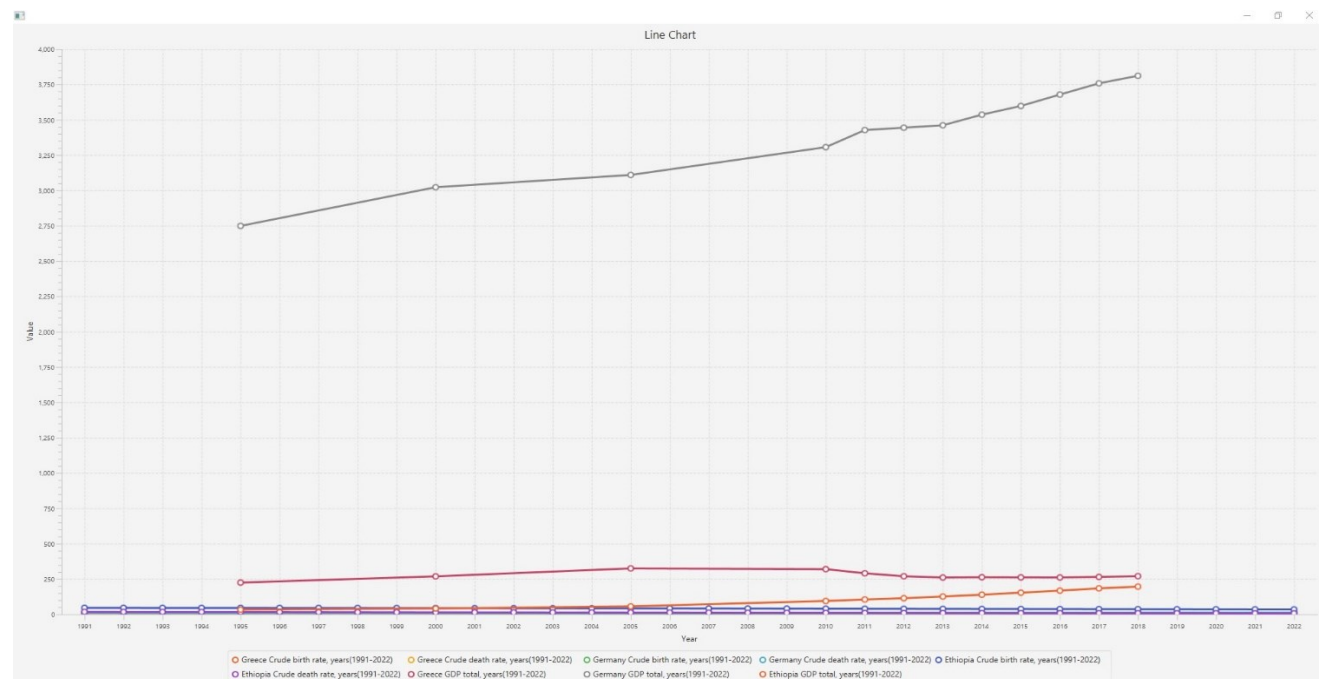
Crude Death Rate:

Εκφράζει την αναλογία θανάτων μιας χώρας σε μια συγκεκριμένη χρονιά ανά 1000 κατοίκους

GDP TOTAL:

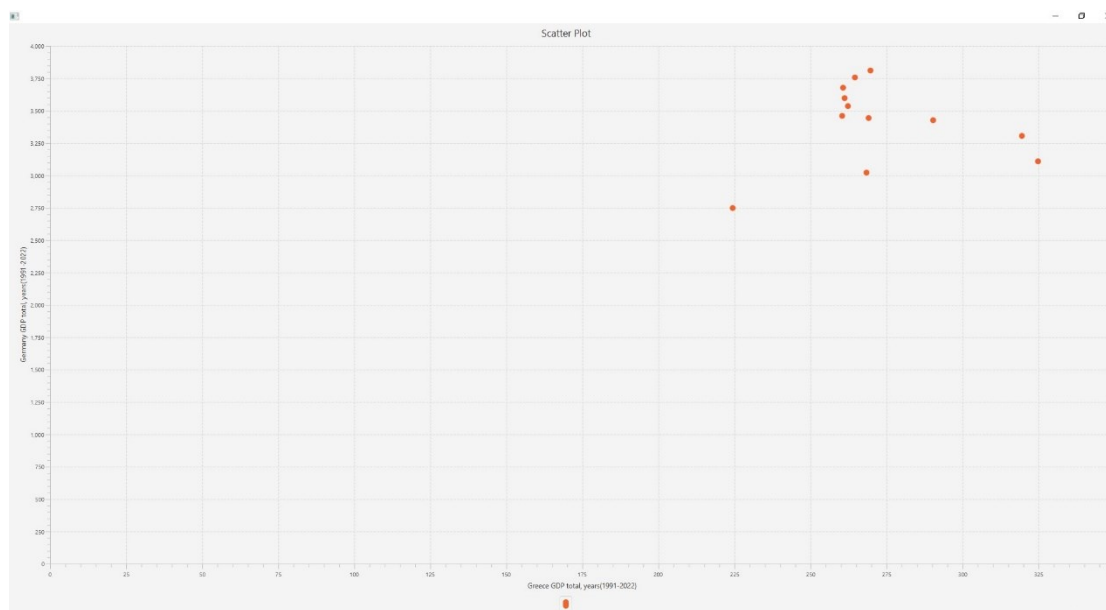
Εκφράζει το ΑΕΠ μιας χώρας για μια συγκεκριμένη χρονιά

Ενδεικτικά δείχνουμε την εξέλιξη της Ελλάδας, της Γερμανίας και της Αιθιοπίας από το 1991 μέχρι το 2022.

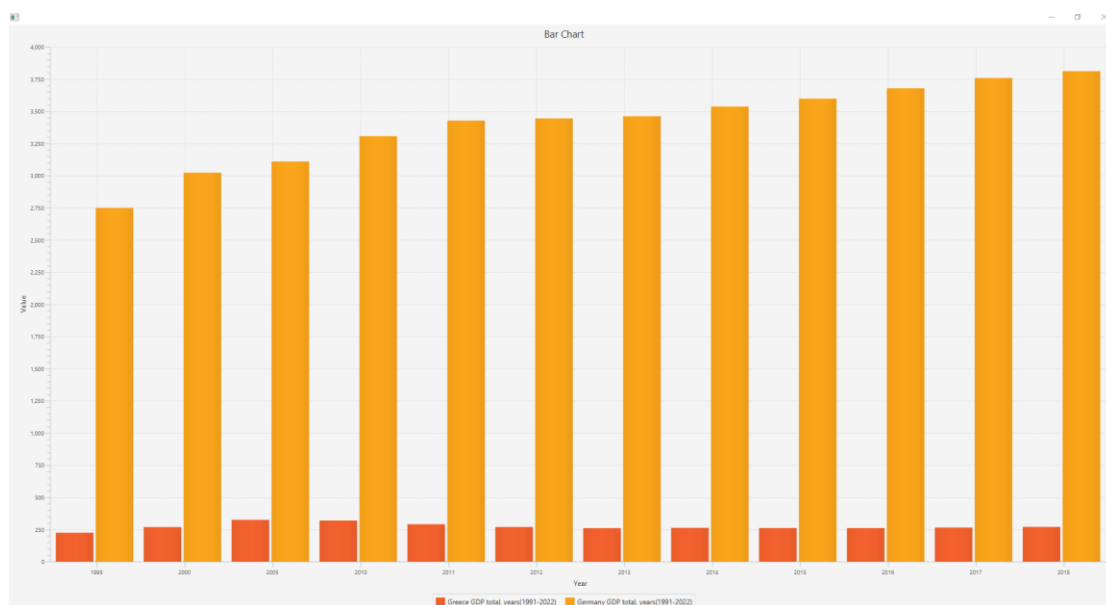


- Σύγκριση χωρών

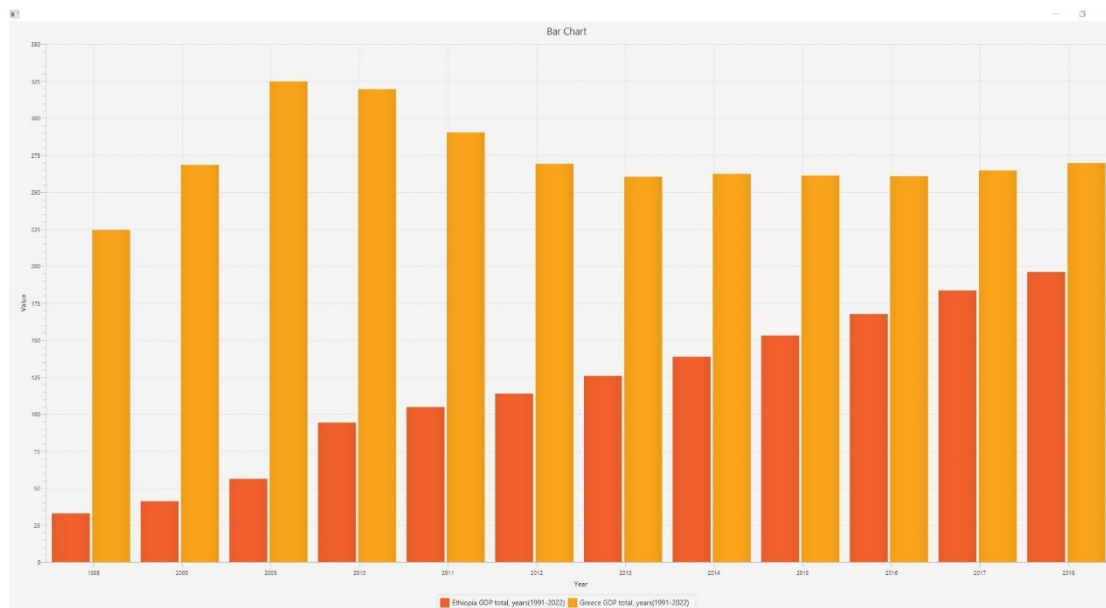
Σύγκριση ΑΕΠ Ελλάδας και Γερμανίας (1991-2022)



Στο Bar Chart φαίνεται καλύτερα η διαφορά



Σύγκριση ΑΕΠ Ελλάδας και Αιθιοπίας (1991-2022)

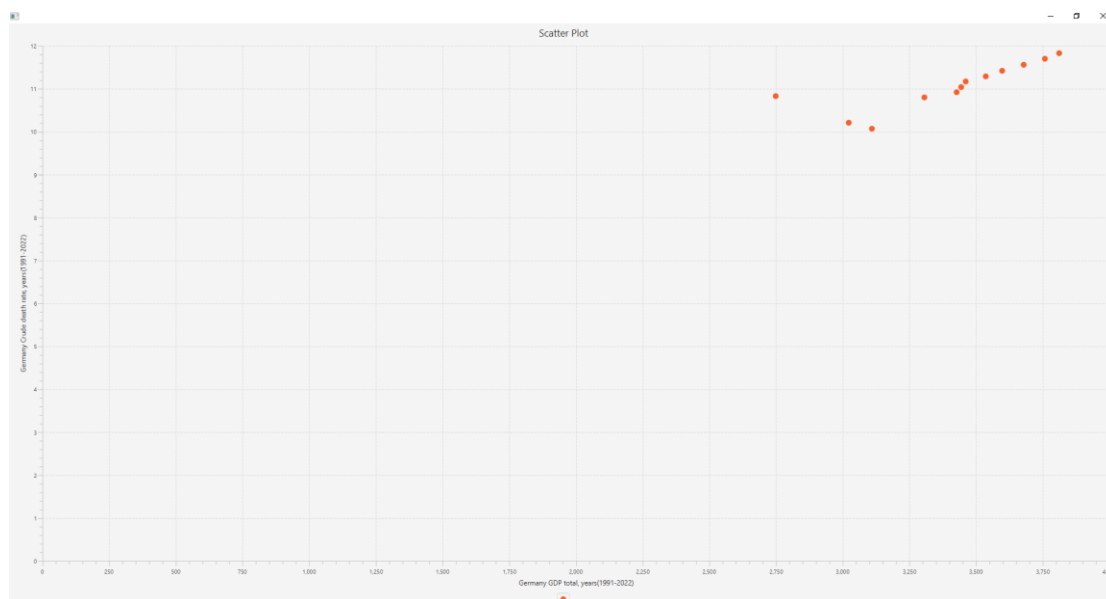


- Σύγκριση δημογραφικών – οικονομικών δεδομένων

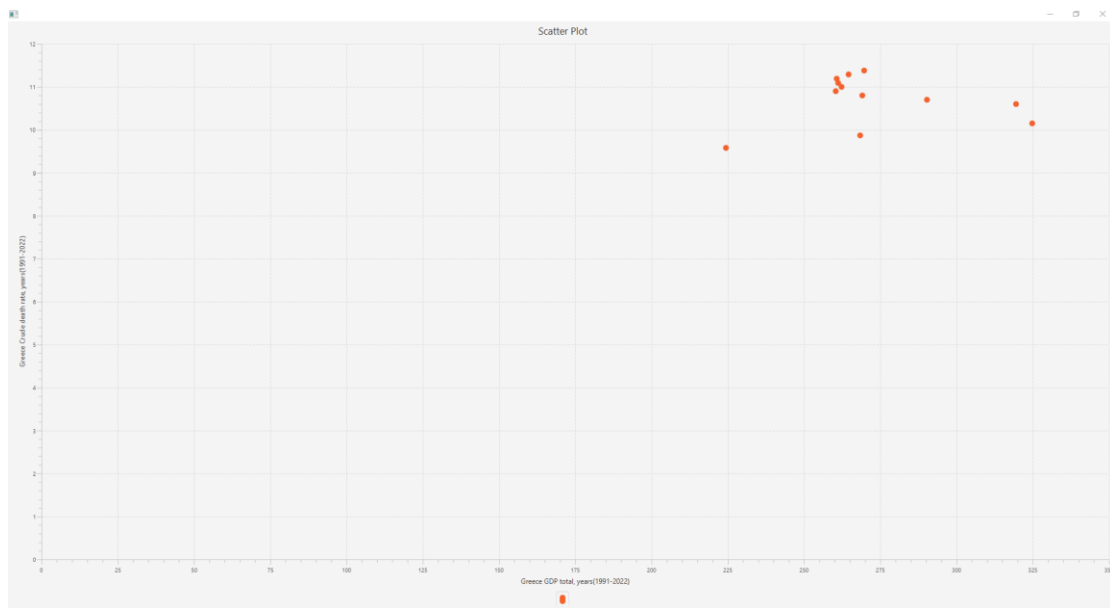
Εδώ θα συγκρίνουμε το ΑΕΠ κάθε χώρας με το ποσοστό θανάτων της.

Οι μετρήσεις γίνονται επίσης για Γερμανία, Ελλάδα και Αιθιοπία καθώς χρειαζόμαστε δεδομένα για μια ανεπτυγμένη, μια αναπτυσσόμενη και μια υποανάπτυκτη χώρα.

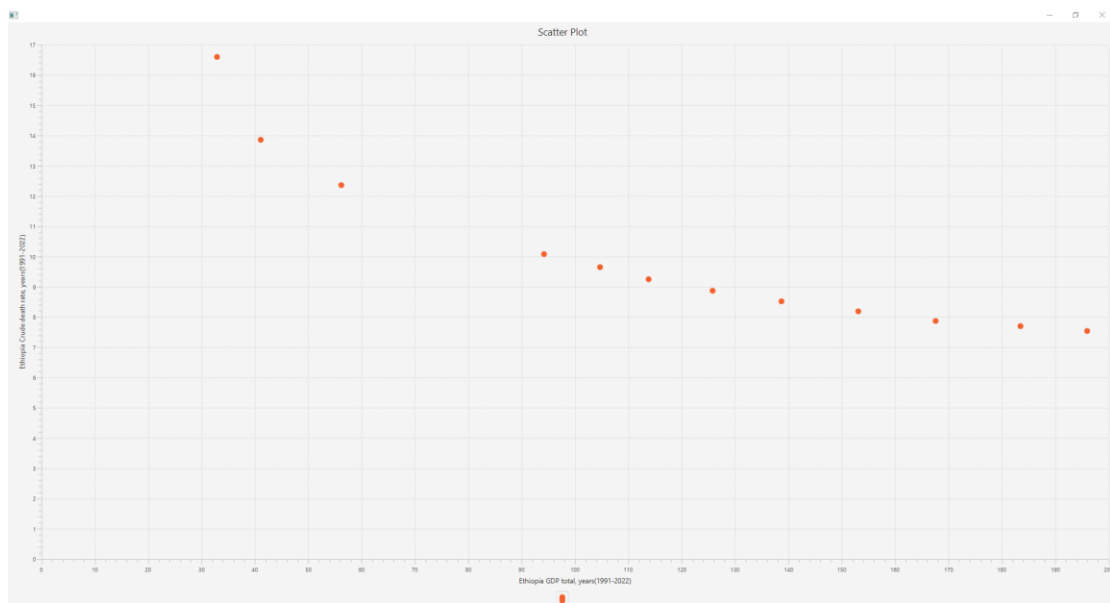
Γερμανία



Ελλάδα



Αιθιοπία



Παρατηρούμε πως όσο καλύτερη οικονομική κατάσταση έχει μια χώρα, τόσο λιγότερους θανάτους έχει ανά χρονιά