ΠΑΝΕΠΙΣΤΗΜΙΟ ΙΩΑΝΝΙΝΩΝ

ΤΜΗΜΑ ΜΗΧ. Η/Υ & ΠΛΗΡΟΦΟΡΙΚΗΣ

Προχωρημένα Θέματα Τεχνολογίας και Εφαρμογών Βάσεων Δεδομένων

ΠΡΟΓΡΑΜΜΑΤΙΣΤΙΚΗ ΕΡΓΑΣΙΑ για το ακαδημαϊκο έτος 2022-2023

Καραγιαννιδησ Χρηστοσ, 4375

Ανδρεασ τριανταφυλλοπουλοσ , 4504

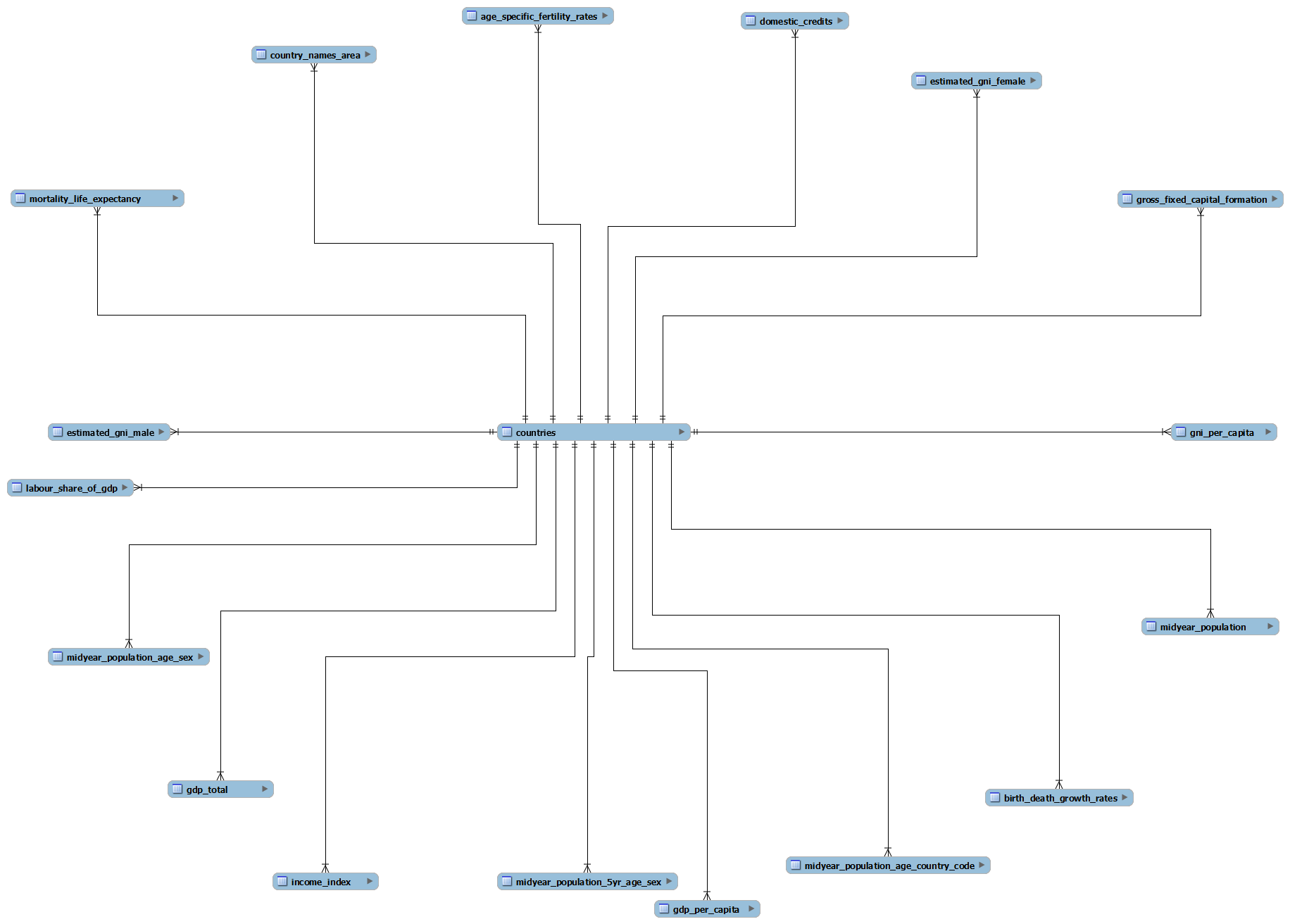
ΤΕΛΙΚΗ ΑΝΑΦΟΡΑ

ΜΑΪΟΣ 2023

# βαση δεδομενων

Στην παρούσα ενότητα περιγράφονται τα σχήματα της βάσης (ή βάσεων, αν είναι παραπάνω από μία) δεδομένων που χρησιμοποιούνται στο project.

## Σχεσιακό σχήμα σε λογικό επίπεδο



Σχήμα 1.1 Σχεσιακό σχήμα της βάσης δεδομένων του συστήματος

CREATE SCHEMA IF NOT EXISTS vaseis2;

ALTER SCHEMA `vaseis2` DEFAULT CHARACTER SET latin1 DEFAULT COLLATE latin1\_bin;

SET GLOBAL local\_infile =1;

USE vaseis2;

drop table if exists age\_specific\_fertility\_rates;

drop table if exists birth\_death\_growth\_rates;

drop table if exists country\_names\_area;

drop table if exists midyear\_population;

drop table if exists midyear\_population\_5yr\_age\_sex;

drop table if exists midyear\_population\_age\_sex;

drop table if exists mortality\_life\_expectancy;

drop table if exists midyear\_population\_age\_country\_code;

drop table if exists domestic\_credits;

drop table if exists Estimated\_GNI\_female;

drop table if exists Estimated\_GNI\_male;

drop table if exists GDP\_per\_capita;

drop table if exists GDP\_total;

drop table if exists GNI\_per\_capita;

drop table if exists Gross\_fixed\_capital\_formation;

drop table if exists Income\_Index;

drop table if exists Labour\_share\_of\_GDP;

drop table if exists countries;

CREATE TABLE IF NOT EXISTS countries (

ISO VARCHAR(2),

ISO3 VARCHAR(3),

ISO\_Code INT NOT NULL,

FIPS VARCHAR(2),

Display\_Name VARCHAR(100),

Official\_Name VARCHAR(100),

Capital VARCHAR(100),

Continent VARCHAR(100),

CurrencyCode VARCHAR(100),

CurrencyName VARCHAR(100),

Phone VARCHAR(100),

Region\_Code VARCHAR(100),

Region\_Name VARCHAR(100),

Subregion\_Code VARCHAR(100),

Subregion\_Name VARCHAR(100),

Intermediate\_Region\_Code VARCHAR(100),

Intermediate\_Region\_Name VARCHAR(100),

Status VARCHAR(100),

Developed\_or\_Developing VARCHAR(100),

Small\_Island\_Developing\_States VARCHAR(100),

Land\_Locked\_Developing\_Countries VARCHAR(100),

Least\_Developed\_Countries VARCHAR(100),

Area\_SqKm VARCHAR(100),

Population VARCHAR(100),

PRIMARY KEY (ISO\_Code)

);

CREATE TABLE IF NOT EXISTS age\_specific\_fertility\_rates (

country\_code VARCHAR(100),

country\_name VARCHAR(100),

year INT NOT NULL,

fertility\_rate\_15\_19 FLOAT,

fertility\_rate\_20\_24 FLOAT,

fertility\_rate\_25\_29 FLOAT,

fertility\_rate\_30\_34 FLOAT,

fertility\_rate\_35\_39 FLOAT,

fertility\_rate\_40\_44 FLOAT,

fertility\_rate\_45\_49 FLOAT,

total\_fertility\_rate FLOAT,

gross\_reproduction\_rate FLOAT,

sex\_ratio\_at\_birth FLOAT,

ISO\_Code INT,

PRIMARY KEY (ISO\_Code,year),

FOREIGN KEY (ISO\_Code) REFERENCES countries(ISO\_Code)

);

CREATE TABLE IF NOT EXISTS birth\_death\_growth\_rates (

country\_code VARCHAR(100),

country\_name VARCHAR(100),

year INT NOT NULL,

crude\_birth\_rate FLOAT,

crude\_death\_rate FLOAT,

net\_migration FLOAT,

rate\_natural\_increase FLOAT,

growth\_rate FLOAT,

ISO\_Code INT,

PRIMARY KEY (ISO\_Code,year),

FOREIGN KEY (ISO\_Code) REFERENCES countries(ISO\_Code)

);

CREATE TABLE IF NOT EXISTS country\_names\_area (

country\_code VARCHAR(100),

country\_name VARCHAR(100),

country\_area INT NOT NULL,

ISO\_Code INT,

PRIMARY KEY (ISO\_Code),

FOREIGN KEY (ISO\_Code) REFERENCES countries(ISO\_Code)

);

CREATE TABLE IF NOT EXISTS midyear\_population (

country\_code VARCHAR(100),

country\_name VARCHAR(100),

year INT NOT NULL,

midyear\_population VARCHAR(100),

ISO\_Code INT,

PRIMARY KEY (ISO\_Code,year),

FOREIGN KEY (ISO\_Code) REFERENCES countries(ISO\_Code)

);

CREATE TABLE IF NOT EXISTS midyear\_population\_5yr\_age\_sex (

country\_code VARCHAR(100),

country\_name VARCHAR(100),

year INT NOT NULL,

total\_flag VARCHAR(100),

starting\_age VARCHAR(100),

age\_group\_indicator VARCHAR(100),

ending\_age VARCHAR(100),

midyear\_population VARCHAR(100),

midyear\_population\_male VARCHAR(100),

midyear\_population\_female VARCHAR(100),

ISO\_Code INT,

PRIMARY KEY (ISO\_Code,year,total\_flag,starting\_age),

FOREIGN KEY (ISO\_Code) REFERENCES countries(ISO\_Code)

);

CREATE TABLE IF NOT EXISTS midyear\_population\_age\_sex (

country\_code VARCHAR(100),

country\_name VARCHAR(100),

year INT NOT NULL,

sex VARCHAR(100),

max\_age VARCHAR(100),

population\_age\_0 VARCHAR(100),

population\_age\_1 VARCHAR(100),

population\_age\_2 VARCHAR(100),

population\_age\_3 VARCHAR(100),

population\_age\_4 VARCHAR(100),

population\_age\_5 VARCHAR(100),

population\_age\_6 VARCHAR(100),

population\_age\_7 VARCHAR(100),

population\_age\_8 VARCHAR(100),

population\_age\_9 VARCHAR(100),

population\_age\_10 VARCHAR(100),

population\_age\_11 VARCHAR(100),

population\_age\_12 VARCHAR(100),

population\_age\_13 VARCHAR(100),

population\_age\_14 VARCHAR(100),

population\_age\_15 VARCHAR(100),

population\_age\_16 VARCHAR(100),

population\_age\_17 VARCHAR(100),

population\_age\_18 VARCHAR(100),

population\_age\_19 VARCHAR(100),

population\_age\_20 VARCHAR(100),

population\_age\_21 VARCHAR(100),

population\_age\_22 VARCHAR(100),

population\_age\_23 VARCHAR(100),

population\_age\_24 VARCHAR(100),

population\_age\_25 VARCHAR(100),

population\_age\_26 VARCHAR(100),

population\_age\_27 VARCHAR(100),

population\_age\_28 VARCHAR(100),

population\_age\_29 VARCHAR(100),

population\_age\_30 VARCHAR(100),

population\_age\_31 VARCHAR(100),

population\_age\_32 VARCHAR(100),

population\_age\_33 VARCHAR(100),

population\_age\_34 VARCHAR(100),

population\_age\_35 VARCHAR(100),

population\_age\_36 VARCHAR(100),

population\_age\_37 VARCHAR(100),

population\_age\_38 VARCHAR(100),

population\_age\_39 VARCHAR(100),

population\_age\_40 VARCHAR(100),

population\_age\_41 VARCHAR(100),

population\_age\_42 VARCHAR(100),

population\_age\_43 VARCHAR(100),

population\_age\_44 VARCHAR(100),

population\_age\_45 VARCHAR(100),

population\_age\_46 VARCHAR(100),

population\_age\_47 VARCHAR(100),

population\_age\_48 VARCHAR(100),

population\_age\_49 VARCHAR(100),

population\_age\_50 VARCHAR(100),

population\_age\_51 VARCHAR(100),

population\_age\_52 VARCHAR(100),

population\_age\_53 VARCHAR(100),

population\_age\_54 VARCHAR(100),

population\_age\_55 VARCHAR(100),

population\_age\_56 VARCHAR(100),

population\_age\_57 VARCHAR(100),

population\_age\_58 VARCHAR(100),

population\_age\_59 VARCHAR(100),

population\_age\_60 VARCHAR(100),

population\_age\_61 VARCHAR(100),

population\_age\_62 VARCHAR(100),

population\_age\_63 VARCHAR(100),

population\_age\_64 VARCHAR(100),

population\_age\_65 VARCHAR(100),

population\_age\_66 VARCHAR(100),

population\_age\_67 VARCHAR(100),

population\_age\_68 VARCHAR(100),

population\_age\_69 VARCHAR(100),

population\_age\_70 VARCHAR(100),

population\_age\_71 VARCHAR(100),

population\_age\_72 VARCHAR(100),

population\_age\_73 VARCHAR(100),

population\_age\_74 VARCHAR(100),

population\_age\_75 VARCHAR(100),

population\_age\_76 VARCHAR(100),

population\_age\_77 VARCHAR(100),

population\_age\_78 VARCHAR(100),

population\_age\_79 VARCHAR(100),

population\_age\_80 VARCHAR(100),

population\_age\_81 VARCHAR(100),

population\_age\_82 VARCHAR(100),

population\_age\_83 VARCHAR(100),

population\_age\_84 VARCHAR(100),

population\_age\_85 VARCHAR(100),

population\_age\_86 VARCHAR(100),

population\_age\_87 VARCHAR(100),

population\_age\_88 VARCHAR(100),

population\_age\_89 VARCHAR(100),

population\_age\_90 VARCHAR(100),

population\_age\_91 VARCHAR(100),

population\_age\_92 VARCHAR(100),

population\_age\_93 VARCHAR(100),

population\_age\_94 VARCHAR(100),

population\_age\_95 VARCHAR(100),

population\_age\_96 VARCHAR(100),

population\_age\_97 VARCHAR(100),

population\_age\_98 VARCHAR(100),

population\_age\_99 VARCHAR(100),

population\_age\_100 VARCHAR(100),

ISO\_Code INT,

PRIMARY KEY (ISO\_Code,year,sex),

FOREIGN KEY (ISO\_Code) REFERENCES countries(ISO\_Code)

);

CREATE TABLE IF NOT EXISTS mortality\_life\_expectancy (

country\_code VARCHAR(100),

country\_name VARCHAR(100),

year INT NOT NULL,

infant\_mortality VARCHAR(100),

infant\_mortality\_male VARCHAR(100),

infant\_mortality\_female VARCHAR(100),

life\_expectancy VARCHAR(100),

life\_expectancy\_male VARCHAR(100),

life\_expectancy\_female VARCHAR(100),

mortality\_rate\_under5 VARCHAR(100),

mortality\_rate\_under5\_male VARCHAR(100),

mortality\_rate\_under5\_female VARCHAR(100),

mortality\_rate\_1to4 VARCHAR(100),

mortality\_rate\_1to4\_male VARCHAR(100),

mortality\_rate\_1to4\_female VARCHAR(100),

ISO\_Code INT,

PRIMARY KEY (ISO\_Code,year),

FOREIGN KEY (ISO\_Code) REFERENCES countries(ISO\_Code)

);

CREATE TABLE IF NOT EXISTS Domestic\_credits (

country VARCHAR(100),

year\_1990 VARCHAR(100),

year\_1995 VARCHAR(100),

year\_2000 VARCHAR(100),

year\_2005 VARCHAR(100),

year\_2010 VARCHAR(100),

year\_2011 VARCHAR(100),

year\_2012 VARCHAR(100),

year\_2013 VARCHAR(100),

year\_2014 VARCHAR(100),

year\_2015 VARCHAR(100),

year\_2016 VARCHAR(100),

year\_2017 VARCHAR(100),

year\_2018 VARCHAR(100),

ISO\_Code INT,

PRIMARY KEY (ISO\_Code),

FOREIGN KEY (ISO\_Code) REFERENCES countries(ISO\_Code)

);

CREATE TABLE IF NOT EXISTS Estimated\_GNI\_female (

country VARCHAR(100),

year\_1995 VARCHAR(100),

year\_2000 VARCHAR(100),

year\_2005 VARCHAR(100),

year\_2010 VARCHAR(100),

year\_2011 VARCHAR(100),

year\_2012 VARCHAR(100),

year\_2013 VARCHAR(100),

year\_2014 VARCHAR(100),

year\_2015 VARCHAR(100),

year\_2016 VARCHAR(100),

year\_2017 VARCHAR(100),

year\_2018 VARCHAR(100),

info VARCHAR(100),

ISO\_Code INT,

PRIMARY KEY (ISO\_Code),

FOREIGN KEY (ISO\_Code) REFERENCES countries(ISO\_Code)

);

CREATE TABLE IF NOT EXISTS Estimated\_GNI\_male (

country VARCHAR(100),

year\_1995 VARCHAR(100),

year\_2000 VARCHAR(100),

year\_2005 VARCHAR(100),

year\_2010 VARCHAR(100),

year\_2011 VARCHAR(100),

year\_2012 VARCHAR(100),

year\_2013 VARCHAR(100),

year\_2014 VARCHAR(100),

year\_2015 VARCHAR(100),

year\_2016 VARCHAR(100),

year\_2017 VARCHAR(100),

year\_2018 VARCHAR(100),

info VARCHAR(100),

ISO\_Code INT,

PRIMARY KEY (ISO\_Code),

FOREIGN KEY (ISO\_Code) REFERENCES countries(ISO\_Code)

);

CREATE TABLE IF NOT EXISTS GDP\_per\_capita (

country VARCHAR(100),

year\_1990 VARCHAR(100),

year\_1995 VARCHAR(100),

year\_2000 VARCHAR(100),

year\_2005 VARCHAR(100),

year\_2010 VARCHAR(100),

year\_2011 VARCHAR(100),

year\_2012 VARCHAR(100),

year\_2013 VARCHAR(100),

year\_2014 VARCHAR(100),

year\_2015 VARCHAR(100),

year\_2016 VARCHAR(100),

year\_2017 VARCHAR(100),

year\_2018 VARCHAR(100),

info VARCHAR(100),

ISO\_Code INT,

PRIMARY KEY (ISO\_Code),

FOREIGN KEY (ISO\_Code) REFERENCES countries(ISO\_Code)

);

CREATE TABLE IF NOT EXISTS GDP\_total (

country VARCHAR(100),

year\_1990 VARCHAR(100),

year\_1995 VARCHAR(100),

year\_2000 VARCHAR(100),

year\_2005 VARCHAR(100),

year\_2010 VARCHAR(100),

year\_2011 VARCHAR(100),

year\_2012 VARCHAR(100),

year\_2013 VARCHAR(100),

year\_2014 VARCHAR(100),

year\_2015 VARCHAR(100),

year\_2016 VARCHAR(100),

year\_2017 VARCHAR(100),

year\_2018 VARCHAR(100),

info VARCHAR(100),

ISO\_Code INT,

PRIMARY KEY (ISO\_Code),

FOREIGN KEY (ISO\_Code) REFERENCES countries(ISO\_Code)

);

CREATE TABLE IF NOT EXISTS GNI\_per\_capita (

country VARCHAR(100),

year\_1990 VARCHAR(100),

year\_1991 VARCHAR(100),

year\_1992 VARCHAR(100),

year\_1993 VARCHAR(100),

year\_1994 VARCHAR(100),

year\_1995 VARCHAR(100),

year\_1996 VARCHAR(100),

year\_1997 VARCHAR(100),

year\_1998 VARCHAR(100),

year\_1999 VARCHAR(100),

year\_2000 VARCHAR(100),

year\_2001 VARCHAR(100),

year\_2002 VARCHAR(100),

year\_2003 VARCHAR(100),

year\_2004 VARCHAR(100),

year\_2005 VARCHAR(100),

year\_2006 VARCHAR(100),

year\_2007 VARCHAR(100),

year\_2008 VARCHAR(100),

year\_2009 VARCHAR(100),

year\_2010 VARCHAR(100),

year\_2011 VARCHAR(100),

year\_2012 VARCHAR(100),

year\_2013 VARCHAR(100),

year\_2014 VARCHAR(100),

year\_2015 VARCHAR(100),

year\_2016 VARCHAR(100),

year\_2017 VARCHAR(100),

year\_2018 VARCHAR(100),

info VARCHAR(100),

ISO\_Code INT,

PRIMARY KEY (ISO\_Code),

FOREIGN KEY (ISO\_Code) REFERENCES countries(ISO\_Code)

);

CREATE TABLE IF NOT EXISTS Gross\_fixed\_capital\_formation (

country VARCHAR(100),

year\_1990 VARCHAR(100),

year\_1995 VARCHAR(100),

year\_2000 VARCHAR(100),

year\_2005 VARCHAR(100),

year\_2010 VARCHAR(100),

year\_2011 VARCHAR(100),

year\_2012 VARCHAR(100),

year\_2013 VARCHAR(100),

year\_2014 VARCHAR(100),

year\_2015 VARCHAR(100),

year\_2016 VARCHAR(100),

year\_2017 VARCHAR(100),

year\_2018 VARCHAR(100),

ISO\_Code INT,

PRIMARY KEY (ISO\_Code),

FOREIGN KEY (ISO\_Code) REFERENCES countries(ISO\_Code)

);

CREATE TABLE IF NOT EXISTS Income\_Index (

country VARCHAR(100),

year\_1990 VARCHAR(100),

year\_1991 VARCHAR(100),

year\_1992 VARCHAR(100),

year\_1993 VARCHAR(100),

year\_1994 VARCHAR(100),

year\_1995 VARCHAR(100),

year\_1996 VARCHAR(100),

year\_1997 VARCHAR(100),

year\_1998 VARCHAR(100),

year\_1999 VARCHAR(100),

year\_2000 VARCHAR(100),

year\_2001 VARCHAR(100),

year\_2002 VARCHAR(100),

year\_2003 VARCHAR(100),

year\_2004 VARCHAR(100),

year\_2005 VARCHAR(100),

year\_2006 VARCHAR(100),

year\_2007 VARCHAR(100),

year\_2008 VARCHAR(100),

year\_2009 VARCHAR(100),

year\_2010 VARCHAR(100),

year\_2011 VARCHAR(100),

year\_2012 VARCHAR(100),

year\_2013 VARCHAR(100),

year\_2014 VARCHAR(100),

year\_2015 VARCHAR(100),

year\_2016 VARCHAR(100),

year\_2017 VARCHAR(100),

year\_2018 VARCHAR(100),

ISO\_Code INT,

PRIMARY KEY (ISO\_Code),

FOREIGN KEY (ISO\_Code) REFERENCES countries(ISO\_Code)

);

CREATE TABLE IF NOT EXISTS Labour\_share\_of\_GDP (

country VARCHAR(100),

year\_2000 VARCHAR(100),

year\_2005 VARCHAR(100),

year\_2010 VARCHAR(100),

year\_2011 VARCHAR(100),

year\_2012 VARCHAR(100),

year\_2013 VARCHAR(100),

year\_2014 VARCHAR(100),

year\_2015 VARCHAR(100),

year\_2016 VARCHAR(100),

year\_2017 VARCHAR(100),

year\_2018 VARCHAR(100),

ISO\_Code INT,

PRIMARY KEY (ISO\_Code),

FOREIGN KEY (ISO\_Code) REFERENCES countries(ISO\_Code)

);

CREATE TABLE IF NOT EXISTS midyear\_population\_age\_country\_code (

country\_code VARCHAR(100),

country\_name VARCHAR(100),

year INT NOT NULL,

sex VARCHAR(100),

max\_age VARCHAR(100),

population\_age\_0 VARCHAR(100),

population\_age\_1 VARCHAR(100),

population\_age\_2 VARCHAR(100),

population\_age\_3 VARCHAR(100),

population\_age\_4 VARCHAR(100),

population\_age\_5 VARCHAR(100),

population\_age\_6 VARCHAR(100),

population\_age\_7 VARCHAR(100),

population\_age\_8 VARCHAR(100),

population\_age\_9 VARCHAR(100),

population\_age\_10 VARCHAR(100),

population\_age\_11 VARCHAR(100),

population\_age\_12 VARCHAR(100),

population\_age\_13 VARCHAR(100),

population\_age\_14 VARCHAR(100),

population\_age\_15 VARCHAR(100),

population\_age\_16 VARCHAR(100),

population\_age\_17 VARCHAR(100),

population\_age\_18 VARCHAR(100),

population\_age\_19 VARCHAR(100),

population\_age\_20 VARCHAR(100),

population\_age\_21 VARCHAR(100),

population\_age\_22 VARCHAR(100),

population\_age\_23 VARCHAR(100),

population\_age\_24 VARCHAR(100),

population\_age\_25 VARCHAR(100),

population\_age\_26 VARCHAR(100),

population\_age\_27 VARCHAR(100),

population\_age\_28 VARCHAR(100),

population\_age\_29 VARCHAR(100),

population\_age\_30 VARCHAR(100),

population\_age\_31 VARCHAR(100),

population\_age\_32 VARCHAR(100),

population\_age\_33 VARCHAR(100),

population\_age\_34 VARCHAR(100),

population\_age\_35 VARCHAR(100),

population\_age\_36 VARCHAR(100),

population\_age\_37 VARCHAR(100),

population\_age\_38 VARCHAR(100),

population\_age\_39 VARCHAR(100),

population\_age\_40 VARCHAR(100),

population\_age\_41 VARCHAR(100),

population\_age\_42 VARCHAR(100),

population\_age\_43 VARCHAR(100),

population\_age\_44 VARCHAR(100),

population\_age\_45 VARCHAR(100),

population\_age\_46 VARCHAR(100),

population\_age\_47 VARCHAR(100),

population\_age\_48 VARCHAR(100),

population\_age\_49 VARCHAR(100),

population\_age\_50 VARCHAR(100),

population\_age\_51 VARCHAR(100),

population\_age\_52 VARCHAR(100),

population\_age\_53 VARCHAR(100),

population\_age\_54 VARCHAR(100),

population\_age\_55 VARCHAR(100),

population\_age\_56 VARCHAR(100),

population\_age\_57 VARCHAR(100),

population\_age\_58 VARCHAR(100),

population\_age\_59 VARCHAR(100),

population\_age\_60 VARCHAR(100),

population\_age\_61 VARCHAR(100),

population\_age\_62 VARCHAR(100),

population\_age\_63 VARCHAR(100),

population\_age\_64 VARCHAR(100),

population\_age\_65 VARCHAR(100),

population\_age\_66 VARCHAR(100),

population\_age\_67 VARCHAR(100),

population\_age\_68 VARCHAR(100),

population\_age\_69 VARCHAR(100),

population\_age\_70 VARCHAR(100),

population\_age\_71 VARCHAR(100),

population\_age\_72 VARCHAR(100),

population\_age\_73 VARCHAR(100),

population\_age\_74 VARCHAR(100),

population\_age\_75 VARCHAR(100),

population\_age\_76 VARCHAR(100),

population\_age\_77 VARCHAR(100),

population\_age\_78 VARCHAR(100),

population\_age\_79 VARCHAR(100),

population\_age\_80 VARCHAR(100),

population\_age\_81 VARCHAR(100),

population\_age\_82 VARCHAR(100),

population\_age\_83 VARCHAR(100),

population\_age\_84 VARCHAR(100),

population\_age\_85 VARCHAR(100),

population\_age\_86 VARCHAR(100),

population\_age\_87 VARCHAR(100),

population\_age\_88 VARCHAR(100),

population\_age\_89 VARCHAR(100),

population\_age\_90 VARCHAR(100),

population\_age\_91 VARCHAR(100),

population\_age\_92 VARCHAR(100),

population\_age\_93 VARCHAR(100),

population\_age\_94 VARCHAR(100),

population\_age\_95 VARCHAR(100),

population\_age\_96 VARCHAR(100),

population\_age\_97 VARCHAR(100),

population\_age\_98 VARCHAR(100),

population\_age\_99 VARCHAR(100),

population\_age\_100 VARCHAR(100),

age VARCHAR(100),

permutation\_role VARCHAR(100),

population VARCHAR(100),

ISO\_Code INT,

PRIMARY KEY (ISO\_Code,year,sex,age),

FOREIGN KEY (ISO\_Code) REFERENCES countries(ISO\_Code)

);

-- LOAD FILES

LOAD DATA INFILE "C:/Users/PC-023/Documents/GitHub/Vaseis2/countries.csv"

INTO TABLE countries

FIELDS TERMINATED BY ','

ENCLOSED BY '"'

LINES TERMINATED BY '\n'

IGNORE 1 ROWS;

LOAD DATA INFILE "C:/Users/PC-023/Documents/GitHub/Vaseis2/modified/modified\_age\_specific\_fertility\_rates.csv"

INTO TABLE age\_specific\_fertility\_rates

FIELDS TERMINATED BY ','

ENCLOSED BY '"'

LINES TERMINATED BY '\n'

IGNORE 1 ROWS;

LOAD DATA INFILE "C:/Users/PC-023/Documents/GitHub/Vaseis2/modified/modified\_birth\_death\_growth\_rates.csv"

INTO TABLE birth\_death\_growth\_rates

FIELDS TERMINATED BY ','

ENCLOSED BY '"'

LINES TERMINATED BY '\n'

IGNORE 1 ROWS;

LOAD DATA INFILE "C:/Users/PC-023/Documents/GitHub/Vaseis2/modified/modified\_country\_names\_area.csv"

INTO TABLE country\_names\_area

FIELDS TERMINATED BY ','

ENCLOSED BY '"'

LINES TERMINATED BY '\n'

IGNORE 1 ROWS;

LOAD DATA INFILE "C:/Users/PC-023/Documents/GitHub/Vaseis2/modified/modified\_midyear\_population.csv"

INTO TABLE midyear\_population

FIELDS TERMINATED BY ','

ENCLOSED BY '"'

LINES TERMINATED BY '\n'

IGNORE 1 ROWS;

LOAD DATA INFILE "C:/Users/PC-023/Documents/GitHub/Vaseis2/modified/modified\_midyear\_population\_5yr\_age\_sex.csv"

INTO TABLE midyear\_population\_5yr\_age\_sex

FIELDS TERMINATED BY ','

ENCLOSED BY '"'

LINES TERMINATED BY '\n'

IGNORE 1 ROWS;

LOAD DATA INFILE "C:/Users/PC-023/Documents/GitHub/Vaseis2/modified/modified\_midyear\_population\_age\_sex.csv"

INTO TABLE midyear\_population\_age\_sex

FIELDS TERMINATED BY ','

ENCLOSED BY '"'

LINES TERMINATED BY '\n'

IGNORE 1 ROWS;

LOAD DATA INFILE "C:/Users/PC-023/Documents/GitHub/Vaseis2/modified/modified\_midyear\_population\_age\_country\_code.csv"

INTO TABLE midyear\_population\_age\_country\_code

FIELDS TERMINATED BY ','

ENCLOSED BY '"'

LINES TERMINATED BY '\n'

IGNORE 1 ROWS;

LOAD DATA INFILE "C:/Users/PC-023/Documents/GitHub/Vaseis2/modified/modified\_mortality\_life\_expectancy.csv"

INTO TABLE mortality\_life\_expectancy

FIELDS TERMINATED BY ','

ENCLOSED BY '"'

LINES TERMINATED BY '\n'

IGNORE 1 ROWS;

LOAD DATA INFILE "C:/Users/PC-023/Documents/GitHub/Vaseis2/modified/modified\_Domestic credits.csv"

INTO TABLE Domestic\_credits

FIELDS TERMINATED BY ','

ENCLOSED BY '"'

LINES TERMINATED BY '\n'

IGNORE 1 ROWS;

LOAD DATA INFILE "C:/Users/PC-023/Documents/GitHub/Vaseis2/modified/modified\_Estimated GNI female.csv"

INTO TABLE Estimated\_GNI\_female

FIELDS TERMINATED BY ','

ENCLOSED BY '"'

LINES TERMINATED BY '\n'

IGNORE 1 ROWS;

LOAD DATA INFILE "C:/Users/PC-023/Documents/GitHub/Vaseis2/modified/modified\_Estimated GNI male.csv"

INTO TABLE Estimated\_GNI\_male

FIELDS TERMINATED BY ','

ENCLOSED BY '"'

LINES TERMINATED BY '\n'

IGNORE 1 ROWS;

LOAD DATA INFILE "C:/Users/PC-023/Documents/GitHub/Vaseis2/modified/modified\_GDP per capita.csv"

INTO TABLE GDP\_per\_capita

FIELDS TERMINATED BY ','

ENCLOSED BY '"'

LINES TERMINATED BY '\n'

IGNORE 1 ROWS;

LOAD DATA INFILE "C:/Users/PC-023/Documents/GitHub/Vaseis2/modified/modified\_GDP total.csv"

INTO TABLE GDP\_total

FIELDS TERMINATED BY ','

ENCLOSED BY '"'

LINES TERMINATED BY '\n'

IGNORE 1 ROWS;

LOAD DATA INFILE "C:/Users/PC-023/Documents/GitHub/Vaseis2/modified/modified\_GNI per capita.csv"

INTO TABLE GNI\_per\_capita

FIELDS TERMINATED BY ','

ENCLOSED BY '"'

LINES TERMINATED BY '\n'

IGNORE 1 ROWS;

LOAD DATA INFILE "C:/Users/PC-023/Documents/GitHub/Vaseis2/modified/modified\_Gross fixed capital formation.csv"

INTO TABLE Gross\_fixed\_capital\_formation

FIELDS TERMINATED BY ','

ENCLOSED BY '"'

LINES TERMINATED BY '\n'

IGNORE 1 ROWS;

LOAD DATA INFILE "C:/Users/PC-023/Documents/GitHub/Vaseis2/modified/modified\_Income Index.csv"

INTO TABLE Income\_Index

FIELDS TERMINATED BY ','

ENCLOSED BY '"'

LINES TERMINATED BY '\n'

IGNORE 1 ROWS;

LOAD DATA INFILE "C:/Users/PC-023/Documents/GitHub/Vaseis2/modified/modified\_Labour share of GDP.csv"

INTO TABLE Labour\_share\_of\_GDP

FIELDS TERMINATED BY ','

ENCLOSED BY '"'

LINES TERMINATED BY '\n'

IGNORE 1 ROWS;

## Σχεσιακό σχήμα σε φυσικο επίπεδο

Όταν θα έχετε στήσει και ρυθμίσει τη βάση δεδομένων σας, εδώ καταγράφονται και οι ρυθμίσεις σε φυσικό επίπεδο. Ενδεικτικά:

### Ρύθμιση των παραμέτρων του dbms

Τροποποιηθηκε το file ->preferences->SQL editor -> dbms connection read timeout interval από 30sec σε 1600sec και στο my.ini στη γραμμή secure-file-priv= να σβησουμε ότι ειχε μετα το ‘=’.

# αρχιτεκτονικη Λογισμικού

## Διαγράμματα ΠΑΚΕΤΩΝ / υποσυστημάτων kentρικησ εφαρμογησ

Η εφαρμογή αποτελείται από τα πακέτα:   
1)DBMS\_data\_visualization,   
2)DBMS\_data\_visualization.controller,   
3)DBMS\_data\_visualization.DAO,   
4)DBMS\_data\_visualization.services,   
5)DBMS\_data\_visualization.entities

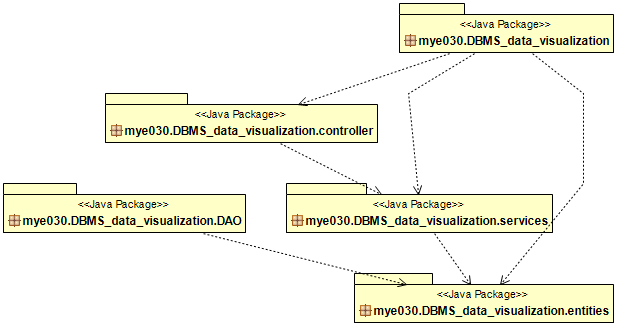
Μέσα στο DBMS\_data\_visualization βρίσκονται τα αρχεία που αφορούν το GUI και τη δημιουργία των γραφημάτων από τα δεδομένα.

Μέσα στο DBMS\_data\_visualization.controller βρίσκεται το αρχείο που είναι υπεύθυνο για το ξεκαθάρισμα των δεδομένων που αντλουνται από τη βάση μεσω του service και την τροφοδοσία τους στην κλάση που είναι υπεθυνη για τη δημιουργία των γραφημάτων.

Μέσα στο DBMS\_data\_visualization.DAO βρισκονται τα Repositories που είναι υπευθυνα για την άντληση δεδομένων από τη βάση για κάθε table της.

Μέσα στο DBMS\_data\_visualization.services βρισκονται τα αρχεία που είναι υπευθυνα για την επικοινωνία του controller με τα Repositories.

Μέσα στο DBMS\_data\_visualization.entities είναι τα αρχεία που αντιπροσωπεύουν τα tables της βάσης με τα αντίστοιχα columns.



## Python Script

Αν η ανάπτυξη γίνει αντικειμενοστρεφώς, εδώ μπαίνουν τα διαγράμματα κλάσεων + ο σχολιασμός της κεντρικής εφαρμογής. Αλλιώς μπαίνουν διαγράμματα που διευκολύνουν την κατανόηση της εσωτερικής αρχιτεκτονικής του λογισμικού (π.χ., component/ deployment diagrams / …)

Ξεκινήσαμε κατασκευάζοντας το Script για την τροποποίηση των στοιχείων που θα μπουν στη βάση, σε Python. Για την υλοποίηση του Script βασιστήκαμε στη βιβλιοθήκη pandas για την διαχείριση των δεδομένων. Δημιουργούμε ένα φάκελο “modified” στον οποίο θα μπουν τα επεξεργασμένα αρχεία. Βάζουμε όλα τα αρχεία csv xlsx που βρίσκονται στο cwd σε ένα φάκελο για να τα δείξουμε στο χρήστη και να τα επεξεργαστούμε. Δίνουμε prompt στο χρήστη να διαλέξει ποιο αρχείο θέλει να κάνει modify (ή όλα τα αρχεία) και για αυτά που θα επιλέξει καλεί τη συνάρτηση ModifyFile.  
 Η γενική ιδέα για το πως θα επεξεργαστούμε τα αρχεία είναι η εξής: Έχουμε το αρχείο countries.csv ως reference file και συσχετίζουμε όλα τα υπόλοιπα αρχεία με το column του με όνομα ISO\_Code. Για να γίνει αυτό συγκρίνουμε τα δεδομένα από τα columns τα οποία θα δώσει ο χρήστης ως input με κάποια fixed columns του αρχείου countries.csv (συγκεκριμένα ISO, ISO3, ISO\_Code, FIPS, Display\_Name, Official\_Name). Δημιουργούμε ένα pandas dataframe με τα matches κάθε column που έδωσε ο χρήστης με κάθε fixed column του αρχείου countries.csv. Για παράδειγμα (col1, ISO), (col1, ISO3)…, (col1, Official\_ Name)…, (coln, Official\_Name). Κρατάμε το dataframe με τα περισσότερα matches (πχ 85% του αρχικού αρχείου) και σε αυτό προσθέτουμε τα matches που βρέθηκαν από τα υπόλοιπα dataframes και δεν υπάρχουν ήδη στο αρχικό dataframe. Πλέον το τελικό dataframe έχει τις περισσότερες γραμμές του αρχικού αρχείου csv με ένα επιπλέον column το οποίο λέγεται ISO\_Code, το οποίο εν τέλει θα είναι foreign key στο countries όταν δημιουργήσουμε τη βάση.  
 Η ιδέα αυτή υλοποιείται στη συνάρτηση ModifyFile η οποία λειτουργεί διαφορετικά αν το αρχείο που της δοθεί είναι xlsx ή csv.

**CSV:**

Φορτώνουμε το αρχείο που θα κάνουμε Modify σε ένα dataframe. Κρατάμε τα columns που έδωσε ο χρήστης να συγκρίνουμε σε ένα πίνακα και φορτώνουμε το αρχείο countries.csv σε ένα dataframe. Κάθε ένα από τα columns που κρατήσαμε σε αυτή τη λίστα το βάζουμε σε μια από τις τέσσερεις κατηγορίες: N CHARACTERS STRING, N CHARACTERS ONLY INTS, 2 CHARACTERS, 3 CHARACTERS με βάση τις οποίες θα συγκρίνουμε αυτά τα columns με τα κατάλληλα columns του countries.csv. Δεν υπάρχει νόημα για παράδειγμα να ψάχνουμε Matches σε ένα column από integers με το column Official\_Name (string).  
 Για όλα τα συγκρίσιμα columns δημιουργούμε ένα merged dataframe όπου τα δυο columns που συγκρίνονται ταυτίζονται και βάζουμε στο τέλος το ISO\_Code της χώρας που κάναμε match. Έπειτα για όλα τα dataframes που δημιουργήθηκαν βρίσκουμε αυτό που είχε τα πιο πολλά matches σε σχέση με το σύνολο των γραμμών του csv αρχείου και το χρησιμοποιούμε σαν βάση πάνω στο οποίο θα προσθέτουμε κ άλλες γραμμές. Για όλα τα υπόλοιπα dataframes που δημιουργήθηκαν βρίσκουμε το αμέσως πιο αποτελεσματικό και αφαιρούμε από αυτό όσα rows έχουν γίνει ήδη match στο βασικό μας dataframe. Τα rows που μένουνε είναι όσα έγιναν match στο δεύτερο πιο αποτελεσματικό και δεν υπάρχουν ήδη στο βασικό μας dataframe. Κάνουμε merge αυτά τα rows με το βασικό μας dataframe και επαναλαμβάνουμε τη διαδικασία για τα υπόλοιπα dataframes (συνεπώς με κάθε merge αφού αυξάνονται τα συνολικά matches το κάθε επόμενο dataframe έχει όλο και λιγότερες γραμμές που να μην υπάρχουν ήδη στο βασικό μας dataframe). Τέλος δημιουργούμε μέσα στο φάκελο modified το νέο αρχείο modified\_name.csv, όπου name το όνομα του κάθε αρχείου που επεξεργαστήκαμε.

**XLSX:**

Το xlsx αρχείο ουσιαστικά είναι μια λίστα από csv αρχεία οπότε το κάνουμε split στα επιμέρους φύλλα του. Για όλα αυτά τα temp αρχεία ξανακαλούμε τη συνάρτηση ModifyFile η οποία όμως αυτή τη φορά θα χρησιμοποιήσει το παραπάνω part που εξηγήσαμε για τα csv αρχεία. Δηλαδή δημιουργεί τόσα modified csv αρχεία, όσα και τα φύλλα του xlsx.

Μια ενδεικτική εκτέλεση του προγράμματος είναι αυτή:

A screenshot of a computer screen

Description automatically generated with medium confidence

Τα columns country\_code και το whatever, τα οποία είναι επίτηδες λάθος, δεν θα βρουν matches οπότε δεν μας απασχολούν. Το -1 στο τέλος δείχνει ότι τελειώσαμε με την εισαγωγή των columns.

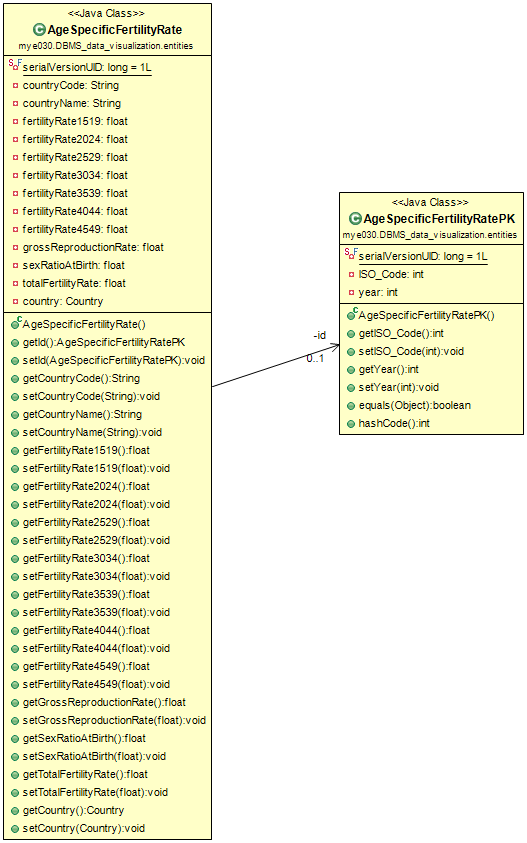
## Διαγράμματα Κλάσεων κεντρικησ εφαρμογησ

Βάση Δεδομένων

Για τη δημιουργία της βάσης χρησιμοποιήσαμε την MySQL και φτιάξαμε το αρχείο createDBscript.sql. Δημιουργείται ένα καινούριο schema με όνομα vaseis2, χρησιμοποιούμε character set latin1 και στη συνέχεια ξεκινάμε να φτιάχνουμε τα tables. Η προσέγγιση μας φτιάχνει ένα table για κάθε αρχείο csv. Στην αρχή κάνουμε drop όσα tables υπάρχουν ήδη για αποφυγή προβλημάτων. Οπότε κάθε table έχει τα columns του αντίστοιχου αρχείου. Σε κάθε πίνακα το κλειδί που σίγουρα συμμετέχει είναι το ISO\_Code και για να πετύχουμε μοναδικό κλειδί σε κάποια tables προσθέσαμε επιπλέον columns τα οποία το χαρακτηρίζουν. Στο τέλος φορτώνουμε τη βάση με δεδομένα από τα modified csv αρχεία με την LOAD DATA INFILE. Τα UML της βάσης φαίνονται στην αρχή μαζί με τον κώδικα της.

Εφαρμογή Java

Για την εφαρμογή δημιουργήθηκε ένα Maven Project πάνω στο οποίο βάλαμε το SpringBoot Framework με τα απαραίτητα dependencies για την επικοινωνία με τη βάση. Στη συνέχεια φτιάχτηκε κατάλληλα το αρχείο application.properties που χρησιμοποιείται για τη σύνδεση με τη βάση. Μετά φτιάχτηκε το package entities στο οποίο τοποθετήθηκαν όλα τα αρχεία που αντιπροσωπεύουν τα tables της βάσης και τα συνθετα (composite) κλειδιά τους τα οποία είναι απαραίτητο να φαίνονται ως ξεχωριστές κλάσεις. Για κάθε entity αν το κλειδί του είναι ένα column τότε μπαίνει από πάνω το annotation @Id ενώ αν το κλειδί του είναι σύνθετο, φτιάχνεται ένα αντικείμενο της κλάσης που το αντιπροσωπεύει και από πάνω μπαίνει ένα annotation @EmbeddedId. Επειδή το ISO\_Code είναι foreign key από το entity country, πρέπει όπου χρησιμοποιείται να μπαίνει μαζί με τα annotation @ManyToOne και @JoinColumn(name="ISO\_Code", insertable=false, updatable=false). Σε όλα τα entities έχουν φτιαχτει οι setters και οι getters για κάθε ένα από τα columns. Μέσα στο πακέτο entities έχουν φτιαχτεί και οι κλάσεις που τελειώνουν σε PK. Αυτές οι κλάσεις αντιπροσωπεύουν τους διάφορους συνδυασμούς από composite keys των entities και στην αρχή τους χρησιμοποιούν το annotation @Embeddable και επίσης όπως και τα entities έχει setters και getters για κάθε column που περιέχει.



A screenshot of a computer program

Description automatically generated with low confidence

A screen shot of a computer

Description automatically generated with medium confidence

A screenshot of a computer

Description automatically generated with medium confidence

A screenshot of a computer program

Description automatically generated with low confidence

A screenshot of a computer program

Description automatically generated with low confidence

A screenshot of a computer program

Description automatically generated with low confidence

A screenshot of a computer program

Description automatically generated with low confidence

A screenshot of a computer program

Description automatically generated with low confidence



A screenshot of a computer program

Description automatically generated with low confidence



A screenshot of a computer program

Description automatically generated with low confidence

A screenshot of a computer program

Description automatically generated with low confidence

A screenshot of a computer

Description automatically generated with low confidence

A screenshot of a computer

Description automatically generated with medium confidence

A screenshot of a computer

Description automatically generated with medium confidence

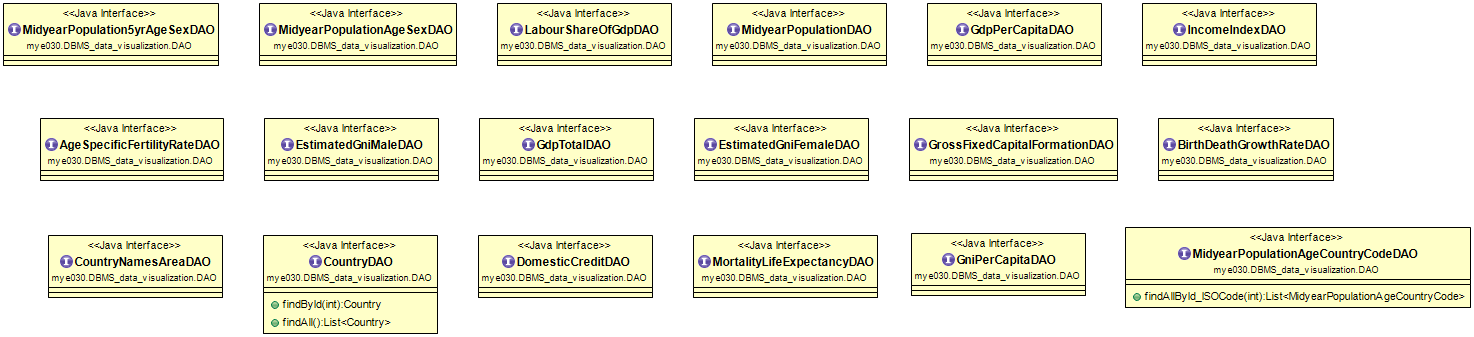
Δεν υπάρχουν φωτογραφίες για τα παρακάτω γιατί είναι πολύ μεγάλες:

MidyearPopulationAgeCountryCode

MidyearPopulationAgeSex

Στη συνέχεια φτιάχτηκε ένα πακέτο DAO που αντιπροσωπεύει τα Repositories για την επικοινωνία με κάθε table της βάσης. Σε κάθε κλάση DAO μπήκε το annotation @Repository και γίνεται extend το JpaRepository με τα κατάλληλα ορίσματα κάθε φορά. Στην πρώτη θέση μπαίνει ο τύπος του αντικειμένου που αντιπροσωπεύει το DAO και στη δεύτερη ο τύπος του κλειδιού. Η μέθοδος findAll δεν χρειάζεται να δηλωθεί υπάρχει εγγενώς μέσα στο Jpa ενώ η μέθοδος findAllById\_ISOCode πρέπει να δηλωθεί στα σημεία που την χρησιμοποιούμε.

Για τις ανάγκες του Project καλούμε όλες τις συναρτήσεις των Repositories από ένα αρχείο services και τα δεδομένα που επιστρέφονται από τα Queries τα χειρίζεται ο controller.



Έπειτα φτιάχτηκε ένα πακέτο services το οποίο περιέχει 2 κλάσεις, ένα Interface και το implementation του. Το Interface περιέχει όλες τις μεθόδους που χρησιμοποιούμε για να ανακτήσουμε αφιλτράριστα δεδομένα από τη βάση και πιο συγκεκριμένα έχει μια μέθοδο για κάθε table της βάσης. Το Implementation του στον constructor παίρνει σαν όρισμα όλα τα Repositories του project και η δουλειά του είναι να αντλεί τα δεδομένα από κάθε Repository όταν αυτά ζητούνται. Στην αρχή του service έχει μπει το annotation @Service και πάνω από κάθε μέθοδο τα annotations @Override και @Transactional.

A picture containing text, screenshot, document, font

Description automatically generated

Το πακέτο controller περιέχει μια κλάση controller η οποία χαρακτηρίζεται με το annotation @Controller. Ο constructor της κλάσεις παίρνει σαν όρισμα ένα αντικείμενο service και χρησιμοποιεί το annotation @Autowired.

Δημιουργήθηκε η μέθοδος getCountries() η οποία κάνει retrieve τα ονόματα όλων των χωρών από τη βάση και τα κρατάει για να προβληθούν στη διεπαφή με το χρήστη. Επόμενη είναι η getIndices η οποία κατασκευάζει μια λίστα για την προβολή των δεικτών στη διεπαφή με το χρήστη. Με αυτό τον τρόπο ο χρήστης μπορεί να επιλέξει τους δείκτες με βάση τους οποίους θα προβληθούν τα αποτελέσματα.

Η createGraphs παίρνει σαν ορίσματα τη λίστα με τις επιλεγμένες χώρες από τη διεπαφή με το χρήστη, τους δυο δείκτες με βάση τους οποίους θα τοποθετηθούν τα δεδομένα στα γραφήματα και τις χρονολογίες που θέλει ο χρήστης για το εύρος των οποίων πρέπει να παρουσιαστούν αποτελέσματα στο χρήστη. Μέσα στη μέθοδο κάνουμε ένα HashMap για να αντιστοιχίσουμε τους δείκτες στα Entities ώστε ανάλογα με τους δείκτες που θα επιλέξει ο χρήστης να γίνει άντληση δεδομένων από το κατάλληλο. Στη συνέχεια δημιουργούμε ένα HashMap για να αντιστοιχίσουμε τους δείκτες στα ονόματα των συναρτήσεων που χρησιμοποιούνται για την άντληση δεδομένων από τη βάση. Αν ο χρήστης δεν δώσει χρονολογίες ανατίθενται αυτόματα η ελάχιστη και η μέγιστη χρονολογία που υπάρχει στα αρχεία. Για τις χώρες που διάλεξε ο χρήστης παίρνουμε τα ISO\_Code τους από το Entity Country. Για τα δυο indices τα οποία διάλεξε ο χρήστης για τα γραφήματα καλείται η συνάρτηση getAppropriateDataFromDB η οποία επιστρέφει μια λίστα με δεδομένα που είναι κατάλληλα για να εμφανιστούν αργότερα στα γραφήματα.

Για τα Entities που προήλθαν από το xlsx αρχείο κάνουμε μια λίστα με τα χρονιά/columns και παίρνουμε όλα τα δεδομένα που υπάρχουν στη βάση για το GDPpercapita. Για κάθε item που τραβήξαμε από τη βάση και για κάθε χρόνο που έχουμε στη λίστα αν το year αυτού του item βρίσκεται εντός του range που έδωσε ο χρήστης και το ISO\_Code του εμπεριέχεται στη λίστα με τα ISO\_Codes που διάλεξε ο χρήστης, τότε γράφουμε σε μια λίστα από strings την τετράδα χώρα, year, value, column και αν η τιμή δεν είναι σκουπίδι την επιστρέφουμε.

Για τα υπόλοιπα Entities πάλι παίρνουμε όλα τα δεδομένα από τη βάση με τη findAll και για κάθε ένα από αυτά, αν το year βρίσκεται στο range που έδωσε ο χρήστης και το ISO\_Code του εμπεριέχεται στη λίστα με τα ISO\_Codes που διάλεξε ο χρήστης, τότε πάλι κρατάμε την τετράδα χώρα, year, value, column και τη βάζουμε στα φιλτραρισμένα δεδομένα που θα επιστρέψουμε.

Ειδικές περιπτώσεις αποτελούν τα πεδία “population by age” και “fertility rate by age”. Στο “population by age” επειδή παίρνουμε δεδομένα από το αρχείο που είναι 1.7GB, για κάθε μια από τις χώρες που έχει δώσει ο χρήστης καλούμε τη συνάρτηση findAllByIsoCodeMidyearPopulationAgeCountryCode ώστε να μην φορτώσουμε όλο το table με τα 1.7GB στη μνήμη και για κάθε γραμμή από αυτές που θα επιστρέψει το service πάλι παίρνουμε το ISO και το year και ελέγχουμε αν το year βρίσκεται στο range και το ISO στη λίστα με τα ISO και αν είναι βάζουμε πάλι την ίδια τετράδα χώρα, year, value, column στα filtered δεδομένα. Αν έχει ζητήσει ο χρήστης το fertility rate by age, τότε παίρνουμε όλα τα δεδομένα από το table AgeSpecificFertilityRate αυτή τη φορά προσθέτουμε, για κάθε γραμμή της οποίας το year είναι στο range και το ISO\_Code στα ISO\_Codes, επτά τετράδες, μια για κάθε εύρος του fertility rate και την αντίστοιχη τιμή της.

Τα δεδομένα αυτά επιστρέφονται με τη μορφή των τετράδων για να αξιοποιηθούν κατάλληλα στη δημιουργία των γραφημάτων από το αρχείο ChartDisplay.java.

Στο αρχείο GraphicalInterface.java έχουμε ένα private πεδιο για κάθε Repository με το annotation @Autowired. Όταν δημιουργούμε το frame του GUI περνάμε σαν όρισμα αυτά τα Repositories ώστε να τα περάσουμε σαν όρισμα στο Implementation του Service. Θέτουμε το μεγεθος του panel καθώς επίσης και το χρώμα και τη θέση του και μέσα σε αυτό προσθέτουμε τα υπόλοιπα κουμπιά και menu. Αρχικά δημιουργούμε ένα instance του DBController και μετά ένα drop down menu που περιέχει τα ονόματα όλων των χωρών από το αρχείο countries. Με το κουμπί “add country” προστίθεται η επιλεγμένη χώρα στη λίστα με τις χώρες για τις οποίες θα εμφανιστεί το γράφημα. Επίσης δημιουργούμε δυο drop down menus για τα δυο στοιχεία για τα οποία θα εμφανιστεί το γράφημα καθώς επίσης και δυο text boxes για το χρονολογικό εύρος. Τέλος έχουμε και τρία check boxes ένα για κάθε είδος γραφήματος και ένα κουμπί “delete country” για τη διαγραφή των χωρών που έχουν ήδη μπει στον πίνακα. Τέλος όταν πατιέται το κουμπί submit παίρνουμε τα στοιχεία από τα text boxes και τα drop down menus και καλούμε τη συνάρτηση createGraphs του manager για να επιστρέψει τα δεδομένα που χρειαζόμαστε σε μορφή που να είναι αξιοποιήσιμη από το εκάστοτε plot. Αναλόγως με το ποιο checkbox είναι τικαρισμένο καλείται η κατάλληλη συνάρτηση από το chartDisplay.java.

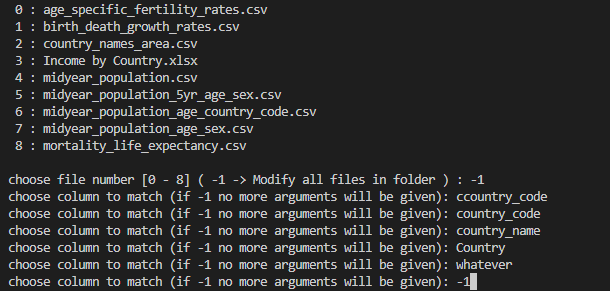
Στο αρχείο chartDisplay.java υπάρχουν 8 συναρτήσεις ,2 για τη δημιουργία του dataset για τα barchart και lineplot/scatterplot αντίστοιχα, 3 για την δημιουργία του panel για το γράφημα και άλλα 3 που όντως δημιουργούν το γράφημα. Συγκεκριμένα η συνάρτηση createScatterPlot δημιουργεί ένα frame και καλεί την scatterPlotPanel για να δημιουργήσει το γράφημα. Η scatterPlotPanel καλεί την createDefaultCategoryDatasetScatterPlot για να πάρει το κατάλληλο dataset και δημιουργεί το plot με αυτό το dataset το οποίο εμφανίζεται τελικά στην οθόνη. Το ίδιο κάνουν και οι createLineChart και createBarChart για με τις LineChartPanel και createBarChartPanel αντίστοιχα.

Τέλος η συνάρτηση createDefaultCategoryDatasetBarChart παίρνει τα δεδομένα από τον πίνακα data ως όρισμα και τα επιστρέφει με κατάλληλη μορφή ώστε να μπορεί η createBarChart να δημιουργήσει το barchart. Το ίδιο κάνει και η createDefaultCategoryDatasetScatterPlot ώστε τα δεδομένα να μπορούν να γίνουν linechart ή scatterplot.

A close-up of a computer screen

Description automatically generated with low confidence

# Υποδείγματα ερωτήσεων και απαντήσεων



# ΜΕΛΕΤΗ ΔΕΔΟΜΕΝΩΝ

A picture containing text, colorfulness, screenshot

Description automatically generated

Από το παραπάνω γράφημα παρατηρούμε ότι η κατανομή του πληθυσμού της Ελλάδας δεν αλλάζει ιδιαίτερα από το 2015 έως και το 2020 και ότι ο περισσότερος πληθυσμός της χώρας είναι από 35 με 70.

A picture containing text, screenshot, colorfulness, plot

Description automatically generated

A picture containing screenshot, rectangle, line, colorfulness

Description automatically generatedΣτην 2η εικόνα προσθέσαμε στο ίδιο γράφημα και τη κατανομή του πληθυσμού της Γερμανίας και αρχικά παρατηρούμε ότι είναι πολύ μεγαλύτερος της Ελλάδας, το οποίο είναι και αναμενόμενο. Στατιστικό ενδιαφέρον έχει και το spike στον πληθυσμό που έχει η Γερμανία στις ηλικίες 50-70 οι οποίοι αποτελούν και το μεγαλύτερο μέρος του συνολικού πληθυσμού.

A graph with red dots

Description automatically generated with low confidence

A picture containing text, screenshot, plot, line

Description automatically generatedΣτα επόμενα γραφήματα μελετήσαμε τα Crude Birth Rates και τα Crude Death Rates για την Ελλάδα από το 2000 έως το 2020 και παρατηρούμε από το barchart ότι μετα το 2010 περίπου μειωθήκαν σημαντικά οι γεννήσεις και αυξήθηκαν με τον ίδιο ρυθμό οι θάνατοι. Στα scatterplot και line plot παρατηρούμε ότι το crude death rate / crude birth rate συγκλίνει σε μια τιμή περίπου στο 1(10.0/9.7) όταν ήταν κοντά οι δείκτες για τα αντίστοιχα rates ενώ όταν απέκτησαν μια απόκλιση τότε αυξήθηκε και αυτό το ποσοστό.

A picture containing screenshot, text, line, parallel

Description automatically generated

Στο τελευταίο ενδεικτικό γράφημα βλέπουμε πως το Gross National Income (GNI) είναι καθ’ όλη τη διάρκεια της 20ετιας μεγαλύτερο για τους άντρες, παρόλο που η διαφορά έχει μικρύνει συνειδητά. Παράλληλα μπορούμε να δούμε ότι από το 2010 και μετα (περίπου από τη κρίση και μετα) υπήρχε μια σταθερή μείωση του εισοδήματος και για τα 2 φύλα.