

ΠΡΟΧΩΡΗΜΕΝΑ ΘΕΜΑΤΑ
ΤΕΧΝΟΛΟΓΙΑΣ ΚΑΙ ΕΦΑΡΜΟΓΩΝ
ΒΑΣΕΩΝ ΔΕΔΟΜΕΝΩΝ

ΠΡΟΓΡΑΜΜΑΤΙΣΤΙΚΗ ΕΡΓΑΣΙΑ ΓΙΑ
ΤΟ ΑΚΑΔΗΜΑΪΚΟ ΕΤΟΣ 2019-2020

ΟΜΑΔΑ ???

LIANOS GEORGIOS, 1706

KRANAS KONSTANTINOS, 2278

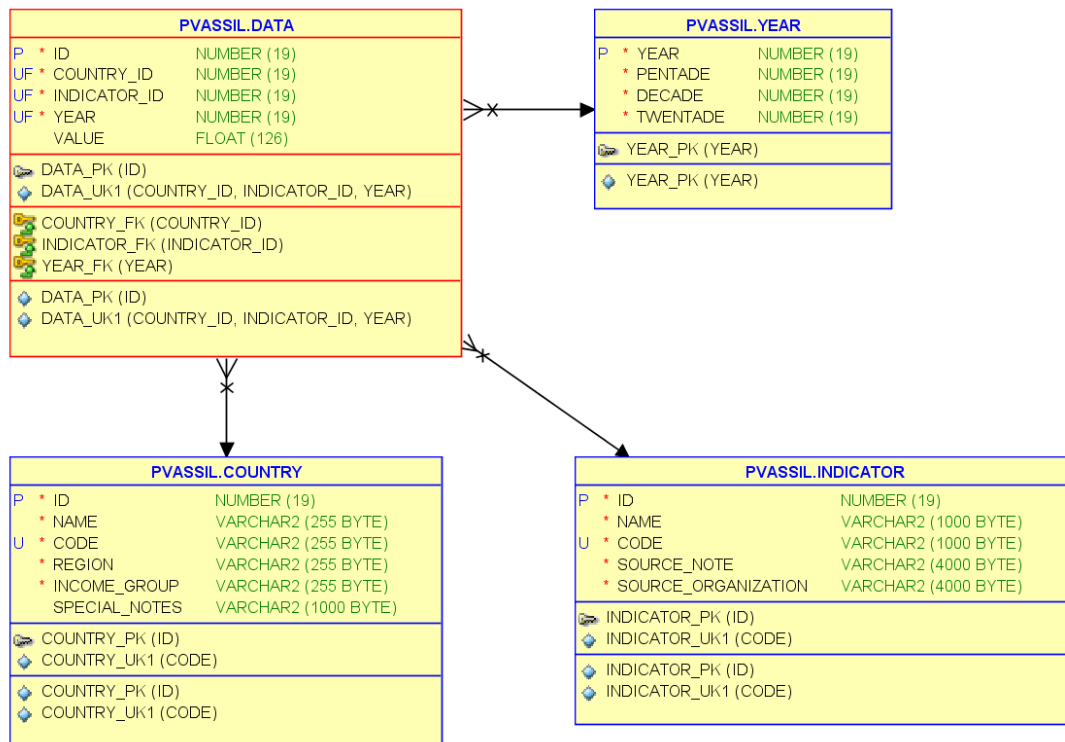
ΤΕΛΙΚΗ ΑΝΑΦΟΡΑ

ΙΣΤΟΡΙΚΟ ΠΡΟΗΓΟΥΜΕΝΩΝ ΕΚΔΟΣΕΩΝ

Ημερομηνία	Έκδοση	Περιγραφή	Συγγραφέας
2020/06/04	1.0	Final	Κ. Κωνσταντίνος, Λ. Γεώργιος

1 ΒΑΣΗ ΔΕΔΟΜΕΝΩΝ

1.1 ΣΧΕΣΙΑΚΟ ΣΧΗΜΑ ΣΕ ΛΟΓΙΚΟ ΕΠΙΠΕΔΟ



Σύμβολο στο Σχεσιακό Σχήμα	Ιδιότητα
Κλειδί	Primary Key
Κλειδί με βελάκι	Foreign Key
Ρόμβος (3 ^η γραμμή)	Unique Field
Ρόμβος (Τελευταία γραμμή)	Index

1.2 ΣΧΕΣΙΑΚΟ ΣΧΗΜΑ ΣΕ ΦΥΣΙΚΟ ΕΠΙΠΕΔΟ

1.2.1 ΡΥΘΜΙΣΗ ΤΩΝ ΠΑΡΑΜΕΤΡΩΝ ΤΟΥ DBMS

Η Oracle χρησιμοποιεί Multitenant Architecture ([1.1 About the Multitenant Architecture](#)).

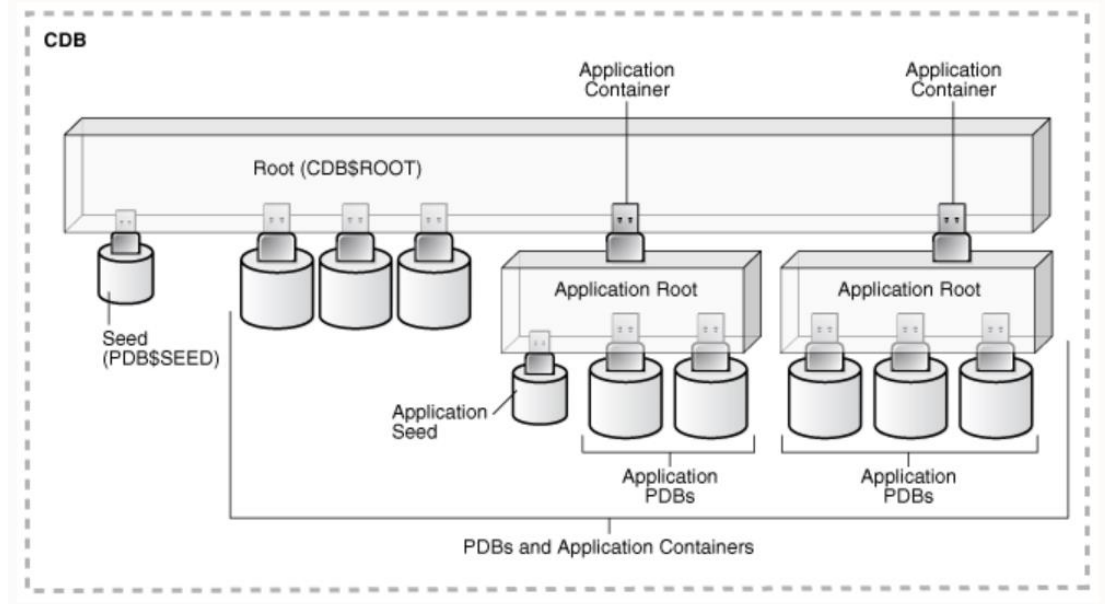
Περίληπτικά, έχει ένα ακριβώς Container Database (CDB) ως root, το οποίο περιέχει πολλές Pluggable Database (PDB). Οι PDBs περιέχουν τα schemas(users). Κατά την εγκατάσταση δημιουργείται αυτόματα ένα PDB, η XEPDB1. Ένας CDB Administrator μπορεί να δημιουργήσει περισσότερα από ένα PDBs, ώστε να το διαχειρίζονται διαφορετικοί PDB Administrators.

Γιατί:

- Configure once, for all
- PDBs share resources
- PDBs data are isolated
- PDBs can be unplugged and plugged to a different server

Πιο αναλυτικά ([1.2 Benefits of the Multitenant Architecture](#))

Figure 1-1 Containers in a CDB



Στα πλαίσια του Project μας αρκεί η XEPDB1, και πρέπει να δημιουργήσουμε ένα καινούριο schema(user). Από την γραμμή εντολών συνδεόμαστε ως sysdba στη sqlplus, αλλάζουμε το container ώστε να συνδεθούμε στην XEPDB1, δημιουργούμε τον user nvassil, και τέλος, του δίνουμε τα απαραίτητα δικαιώματα.

Βήματα δημιουργίας καινούριου schema(user):

1. cmd> sqlplus sys as sysdba
 - 1.1. *enter admin password (set on database installation)
2. SQL> show pdbs; (optional: shows current pdbs)
3. SQL> alter session set container=xepdb1;
4. SQL> create user myuser identified by mypassword;
5. SQL> grant all privileges to myuser; (not the best practice, but the easiest one)

Για ευκολία όλα τα credentials στο project είναι nvassil/pvassil.

Αφού δημιουργήσαμε τη βάση μας, μπορούμε πλέον να συνδεθούμε σε αυτήν κατευθείαν από τερματικό με την εντολή:

```
cmd> sqlplus nvassil/pvassil@localhost:1521/xepdb
```

Πιο γενικά:

```
cmd> sqlplus myuser/mypass@server:port/pdb
```

Με αυτή την εντολή δημιουργούμε νέα σύνδεση στον sqldeveloper, το γραφικό περιβάλλον της Oracle.

1.2.2 ΡΥΘΜΙΣΗ ΤΟΥ ΦΥΣΙΚΟΥ ΣΧΗΜΑΤΟΣ ΤΗΣ ΒΑΣΗΣ ΔΕΔΟΜΕΝΩΝ

Κατά τον ορισμό των primary και unique keys των πινάκων δημιουργούνται αυτόματα και αντίστοιχα indexes.

Επίσης, κατά τον ορισμό auto incremented πεδίων, δημιουργούνται αυτόματα αντίστοιχα triggers και sequences. Τα triggers ενεργοποιούνται σε εντολές insert και παίρνουν το αμέσως επόμενο νούμερο των αντίστοιχων sequences. Τα νούμερα των sequences έχουν ένα εύρος τιμών, με δυνατότητα κύκλου αν ξεπεραστεί η μέγιστη τιμή και cached τιμών για πολλαπλά insert. Στα πλαίσια του Project, μιας και μία φορά τροφοδοτούμε δεδομένα δεν μας ενδιαφέρουν, απλά αναφέρονται.

Οι πίνακες COUNTRY, INDICATOR και YEAR είναι lookup tables.

Όλοι οι πίνακες εκτός του YEAR έχουν auto incremented το πεδίο id ως primary key. Εφόσον κάθε χρονιά υπάρχει ακριβώς μία φορά, μπορεί να υπάρξει ως primary key στον πίνακα YEAR, και έτσι τέθηκε.

1.2.3 ΡΥΘΜΙΣΗ ΑΣΦΑΛΕΙΑΣ

Export:

Μπορεί να μεταφερθεί όλη η PDB σε άλλο σημείο και κάθε schema θα διατηρεί τα user/pass που είχε.

Εκτός του backup σηκώνοντας όλη την PDB, μπορούμε να κρατήσουμε backup μόνο το schema που θέλουμε με την εντολή:

```
cmd> exp userid=myuser/mypass@//server:port/pdb owner=(myuser) file=myfile.dmp
```

όπου myuser και mypass τα στοιχεία που θέσαμε κατά τη δημιουργία του schema.

Το πεδίο mypass μπορεί να παραλειφθεί από την εντολή, αλλά ζητείται αμέσως μετά.

Import:

Εάν έχουμε το *.dmp αρχείο ενός exp backup, μπορούμε να τρέξουμε την εντολή:

```
cmd> imp userid=myuser/mypass@server:port/xepdb1 file=myfile.dmp  
fromuser=myuser touser=myuser
```

Το Oracle Apex (γραφιστικό περιβάλλον ανάπτυξης εφαρμογής) μας δίνει τη δυνατότητα δημιουργίας χρηστών που θα αλληλεπιδρούν με την εφαρμογή (developers, end users), οπότε θα το δούμε αργότερα.

2 ΑΡΧΙΤΕΚΤΟΝΙΚΗ ΛΟΓΙΣΜΙΚΟΥ

2.1 ΑΡΧΙΤΕΚΤΟΝΙΚΗ ΚΑΙ ΔΟΜΗ ETL

Directory:

```
/Project Scripts/ETL
  /data
  /node_modules/oracledb
  /transformed
  transform_countries.js
  transform_data.js
  transform_indicators.js
  transform_years.js
```

Στο φάκελο /data βρίσκονται τα CSV αρχεία (unzipped) από όλες τις χώρες που θέλουμε. Τα scripts διαβάζουν τα αρχεία από τον φάκελο /data και μετά την επεξεργασία δημιουργούν το αντίστοιχο αρχείο CSV στον φάκελο /transformed.

Σημείωση: Τα τελικά αρχεία CSV που θα φορτώσουμε στην εφαρμογή δεν περιέχουν headers. Δηλαδή η πρώτη σειρά περιέχει δεδομένα. Αυτό δεν μας ενοχλεί καθώς ο Data Import Wizard μας δίνει την επιλογή να φορτώσουμε header-less αρχεία και manually να επιλέξουμε τις στήλες που αντιπροσωπεύουν. [Example: Loading Data into a Table](#)

Σε όλα τα scripts, η λογική τους είναι απλή.

Κάθε script διαβάζει τα αντίστοιχα αρχεία που θέλει.

Script	Starts with
transform_countries.js	Metadata_Country
transform_indicators.js	Metadata_Indicator
transform_data.js	API
transform_years.js	API

Φορτώνουμε το περιεχόμενο του κάθε αρχείου σε ένα object datastring.

Θέλουμε να αποφύγουμε τυχόν «σκουπίδια» που να υπάρχουν πριν τους headers (πχ τα API έχουν τελευταίας ενημέρωσης πριν τα δεδομένα που θέλουμε), άρα κάθε script κάνει slice σε ένα αντίστοιχο κοινό χαρακτηριστικό των δεδομένων του.

Script	Slice to
transform_countries.js	Country Code
transform_indicators.js	Indicator Code
transform_data.js	API
transform_years.js	1960

Υστερα, διατρέχει το datastring και σε κάθε « , » που βρίσκει προσauξάνει έναν πίνακα dataArray με το περιεχόμενο που υπήρχε μεταξύ των τελευταίων « ” ».

Γνωρίζουμε εκ των προτέρων ότι οι Countries έχουν 5 στήλες, άρα κάθε γραμμή στο αρχείο θα έχει 5 θέσεις στον dataArray, συγκεκριμένα η i-οστή γραμμή θα έχει τις θέσεις


i+0, i+1, i+2, i+3, i+4. Με ένα απλό mapping των στηλών βρίσκουμε με ποια σειρά τοποθετήθηκαν στον πίνακα και δημιουργούμε καινούρια γραμμή στο αρχείο /transformed/country.csv.

Το ίδιο συμβαίνει και για τους Indicators, με την διαφορά ότι έχουμε 4 στήλες.

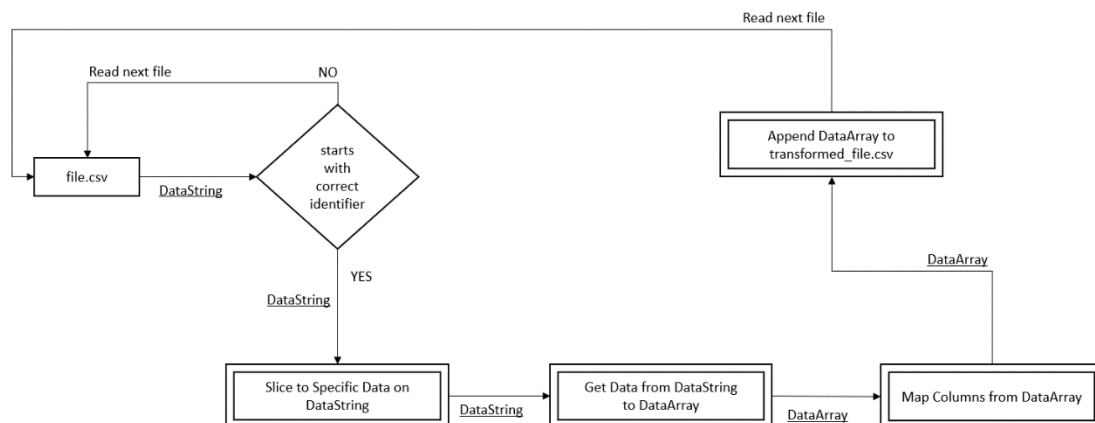
Για τις χρονίες, αφού φορτώσει τα περιεχόμενα του αρχείου στον datastring, ξεκινά να το διατρέχει από τη χρονιά 1960 που γνωρίζουμε ότι ξεκινάν όλες οι μετρήσεις, και όσο βρίσκει αριθμό προσauξάνει τον dataarray.

Όσον αφορά το transform_data.js, είναι σημαντικό να έχουμε φορτώσει τα δεδομένα των country, indicator, years πινάκων στη βάση. Στο ξεκίνημα του, το transform_data.js τραβάει από τη βάση τα id και code των countries και indicators. Ύστερα, για κάθε αρχείο, διατρέχει τις γραμμές του, και σε κάθε γραμμή, εάν η 1^η της στήλη αντιστοιχεί σε κωδικό χώρας που σήκωσε από την βάση και η 3^η της στήλη σε κωδικό indicator, τότε η γραμμή είναι έγκυρη και από την 5^η στήλη και μέχρι το τέλος της γραμμής προσauξάνει τον dataarray με τη μορφή: "ccode, icode, year, value". Στο τέλος κάθε αρχείου γράφει τον dataarray στο αρχείο /transformed/data.csv και συνεχίζει στο επόμενο api αρχείο.

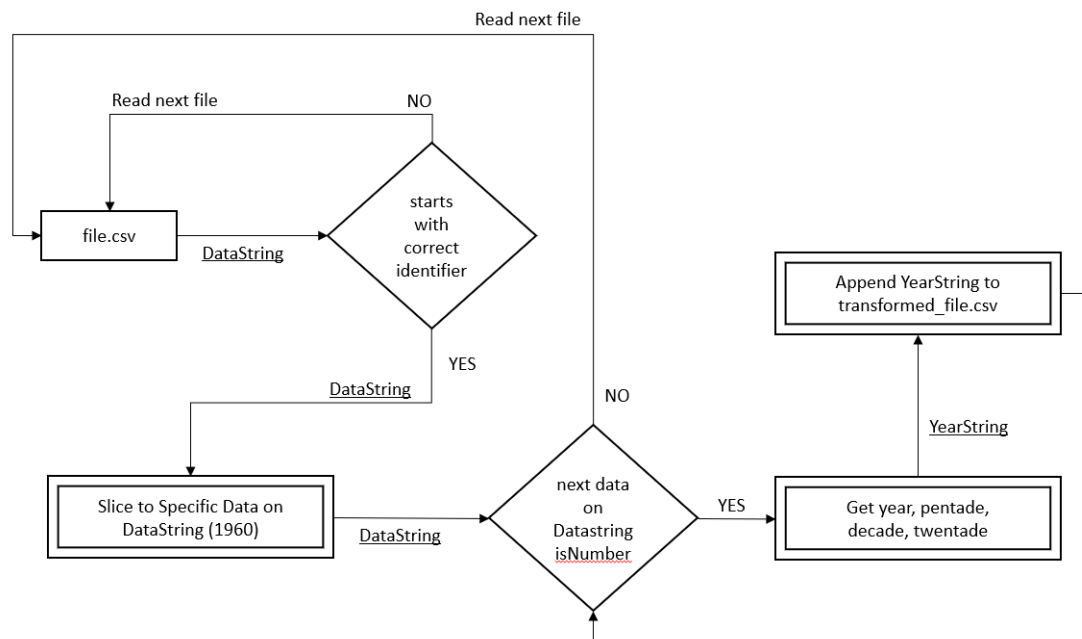
Flow Charts:

Με  στα σχήματα συμβολίζονται συναρτήσεις/απλοί μετασχηματισμοί πάνω σε data.

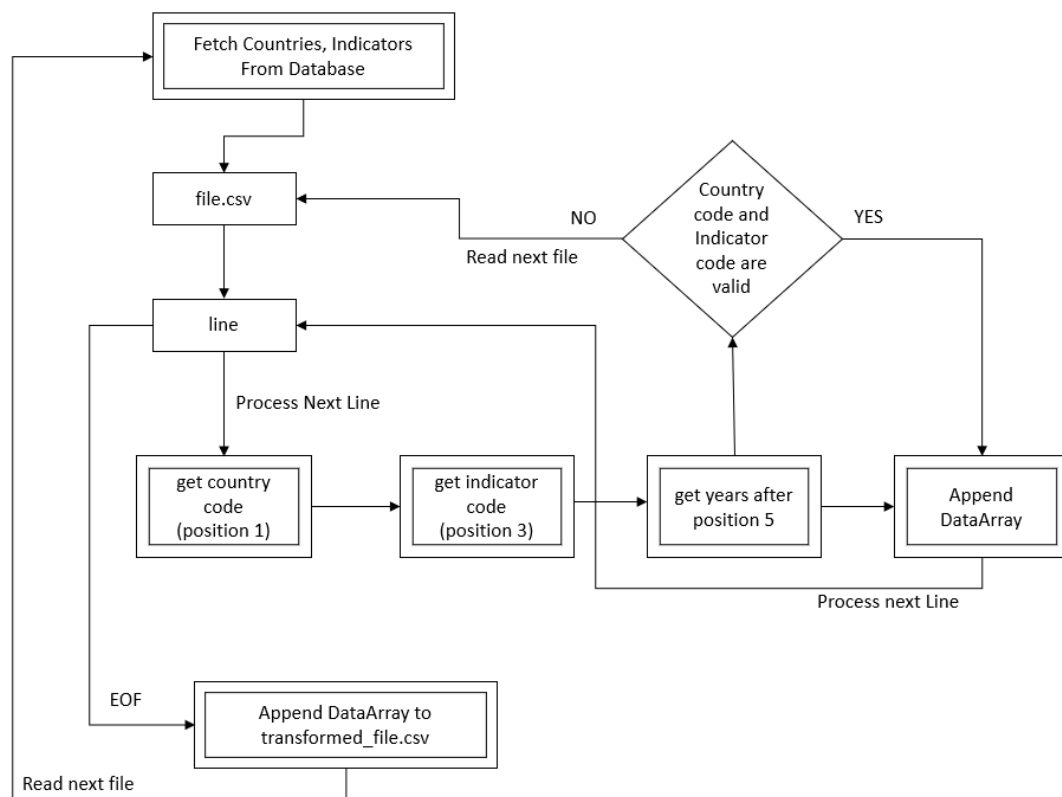
transform_countries & transform_indicators:



transform_years:



transform_data:

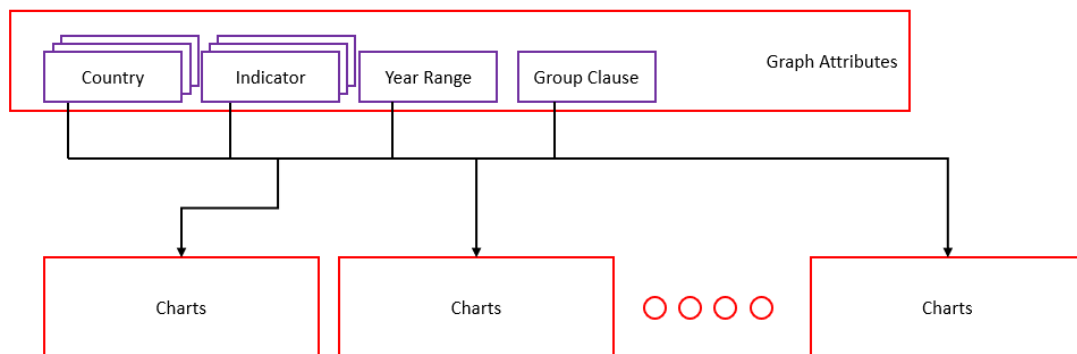


2.2 ΔΙΑΓΡΑΜΜΑΤΑ ΠΑΚΕΤΩΝ / ΥΠΟΣΥΣΤΗΜΑΤΩΝ ΚΕΝΤΡΙΚΗΣ ΕΦΑΡΜΟΓΗΣ

Στο Oracle Apex, η εφαρμογή αποτελείται από κάποιες σελίδες. Κάθε σελίδα περιέχει components της μορφής Regions, Items, Buttons.

Έχουμε Regions για Line Chart Και Bar Chart, και ένα Graph Attributes Region το οποίο περιέχει τα Items που τροφοδοτούν τα queries. Συγκεκριμένα περιέχει λίστες από countries, indicators και year_from – year_to, και checkboxes για κανονικό query ή groupby.

Ενδεικτικά:



Με μωβ στο σχήμα είναι τα Items, ενώ με κόκκινο τα Regions.

Η εφαρμογή έχει υλοποιηθεί με σκοπό να είναι One Page, έτσι με το που ο χρήστης κάνει συνδεθεί μεταφέρεται αυτομάτως στη σελίδα με τα γραφήματα.

Η Oracle έχει κάνει ήδη τη δουλειά για εμάς και μας παρέχει login page για τους χρήστες που έχουμε δημιουργήσει, κάθε developer έχει και end user account για testing.

Οι λίστες Country, Indicator και YearFrom, YearTo τροφοδοτούνται από κάποιες καθολικές λίστες List of Values (LOV) της εφαρμογής. Οι LOV μπορούν να είναι στατικές ή δυναμικές(SQL queries), με σκοπό να υλοποιούνται μία φορά και να χρησιμοποιούνται από πολλά αντικείμενα της εφαρμογής.

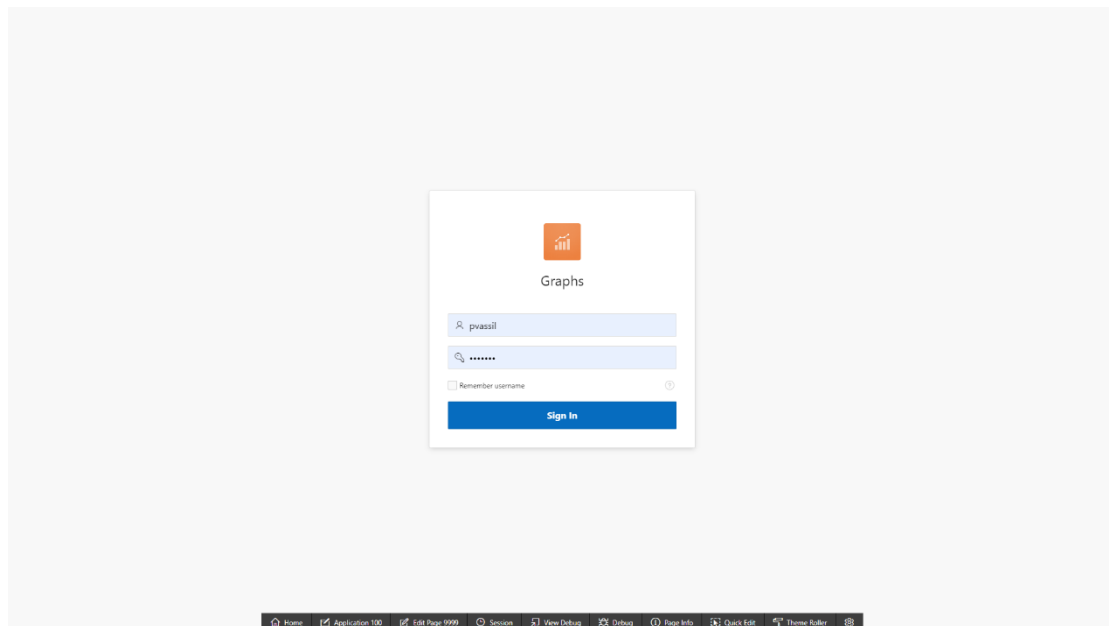
LOV_COUNTRY:

```
select NAME as d, ID as r
from COUNTRY
```

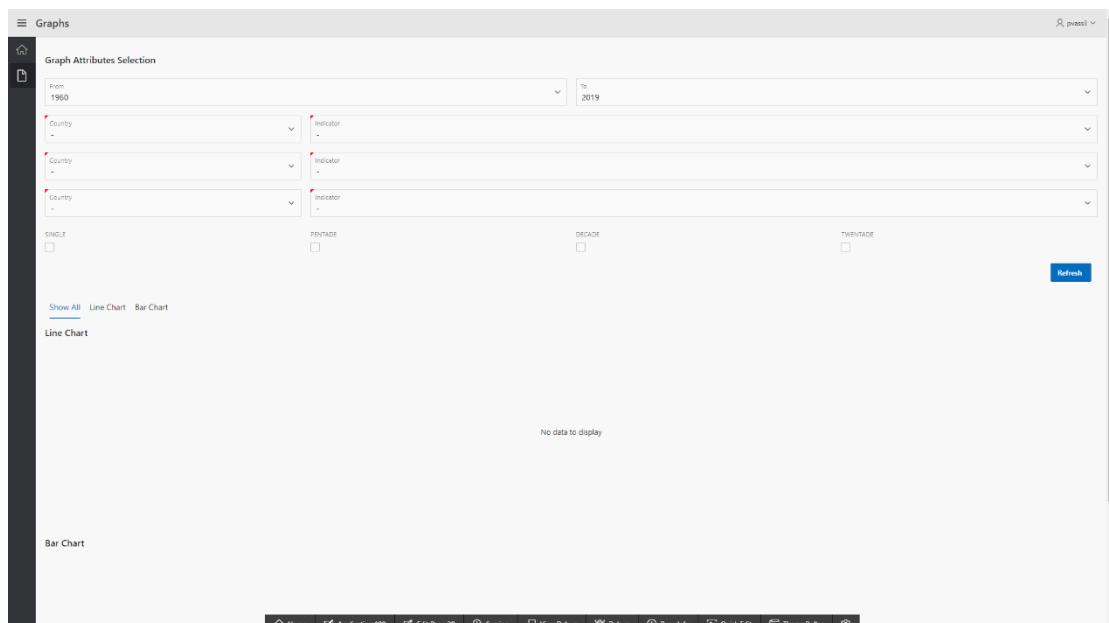
Η LOV_COUNTRY επιστρέφει το όνομα μιας χώρας ως display και το id της ως return. Τα ίδια ισχύουν και για τις LOV_INDICATOR και LOV_YEAR. Έτσι, γνωρίζουμε ποια id θέλουμε από τον πίνακα DATA.

3 ΥΠΟΔΕΙΓΜΑΤΑ ΕΡΩΤΗΣΕΩΝ ΚΑΙ ΑΠΑΝΤΗΣΕΩΝ

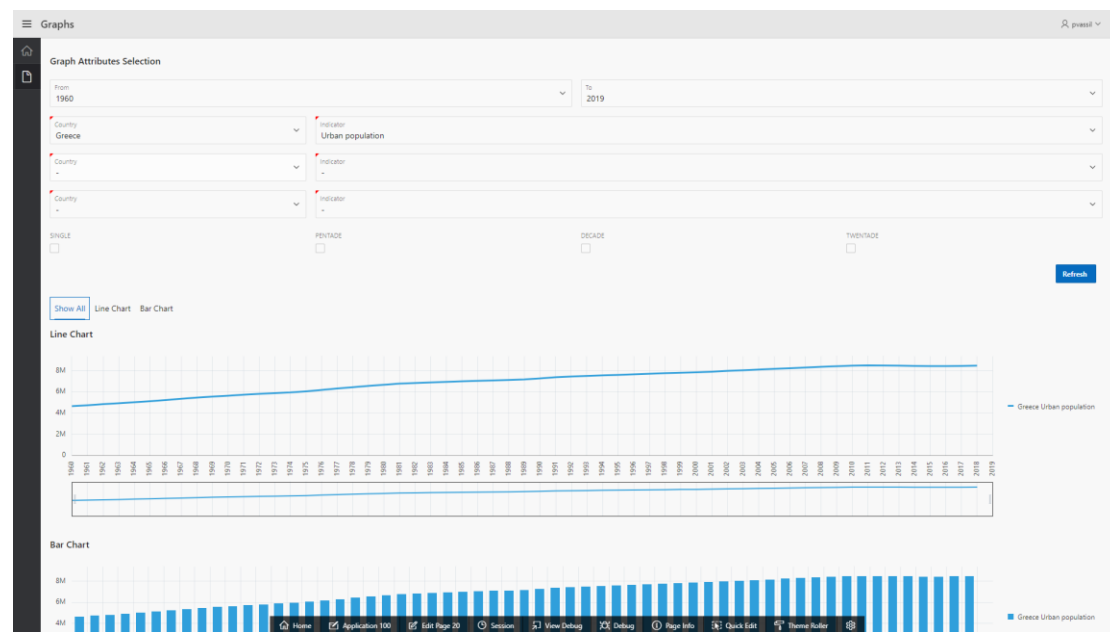
Login Page:



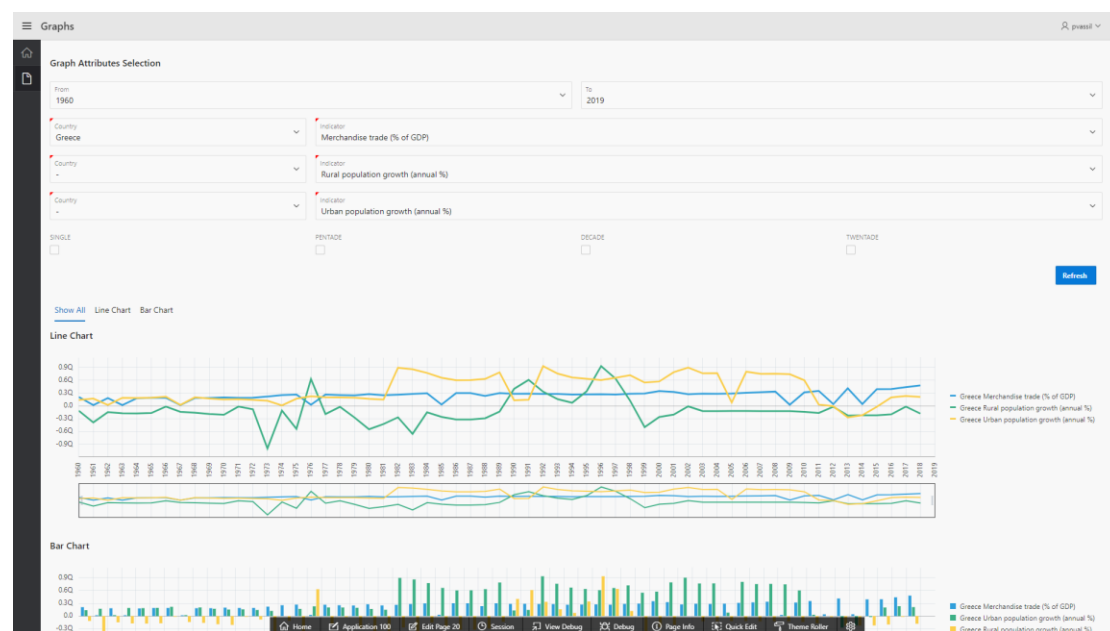
Main Page:



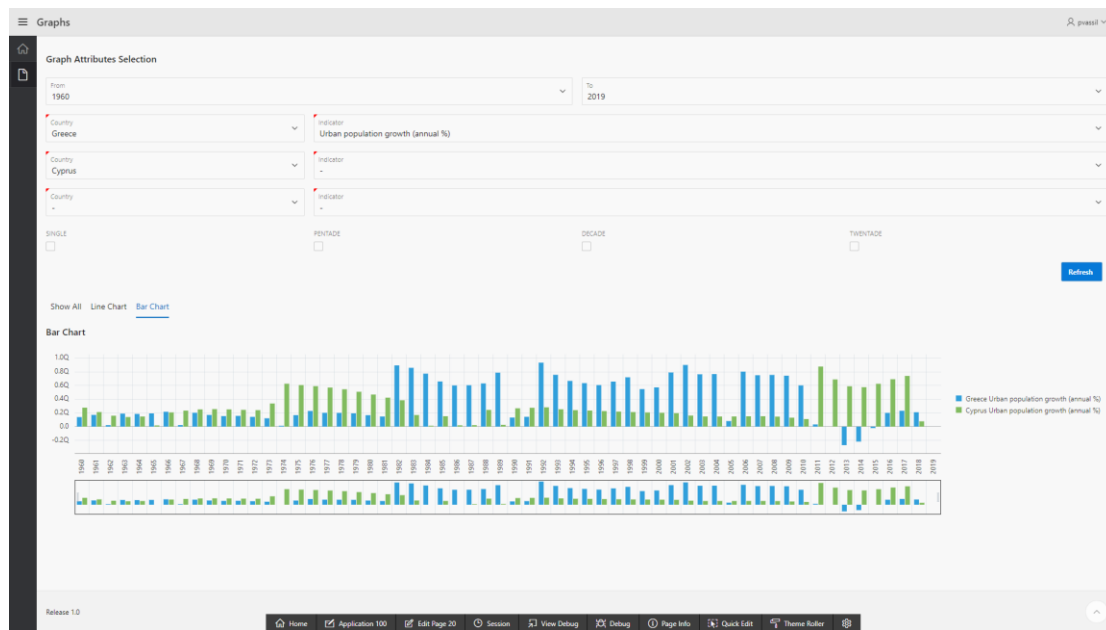
1 Country 1 Indicator Query:



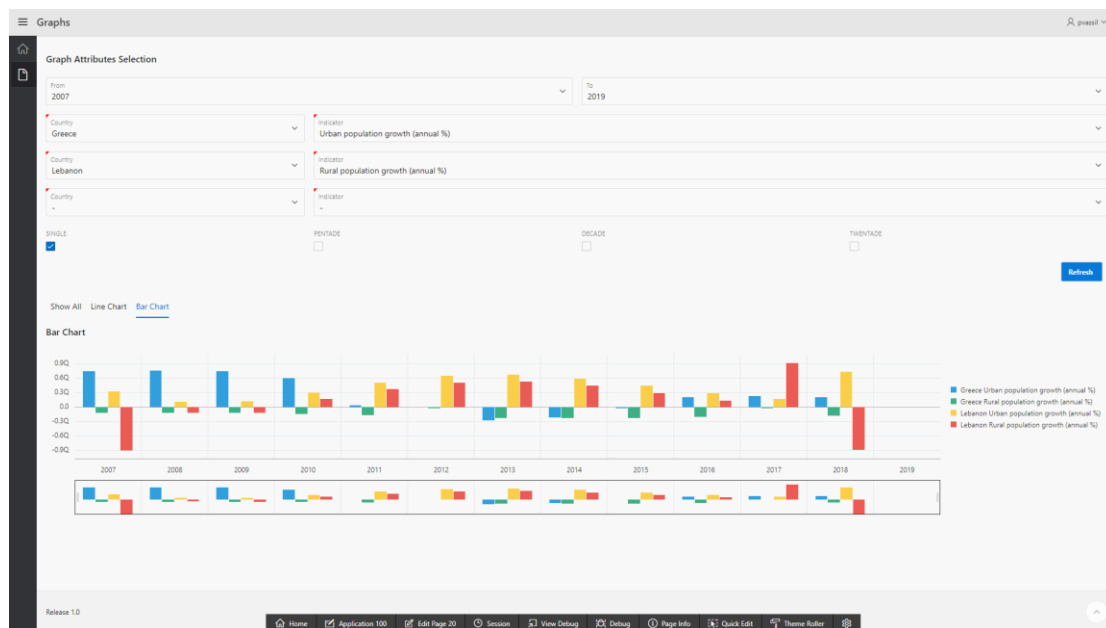
1 Country 3 Indicators Query:



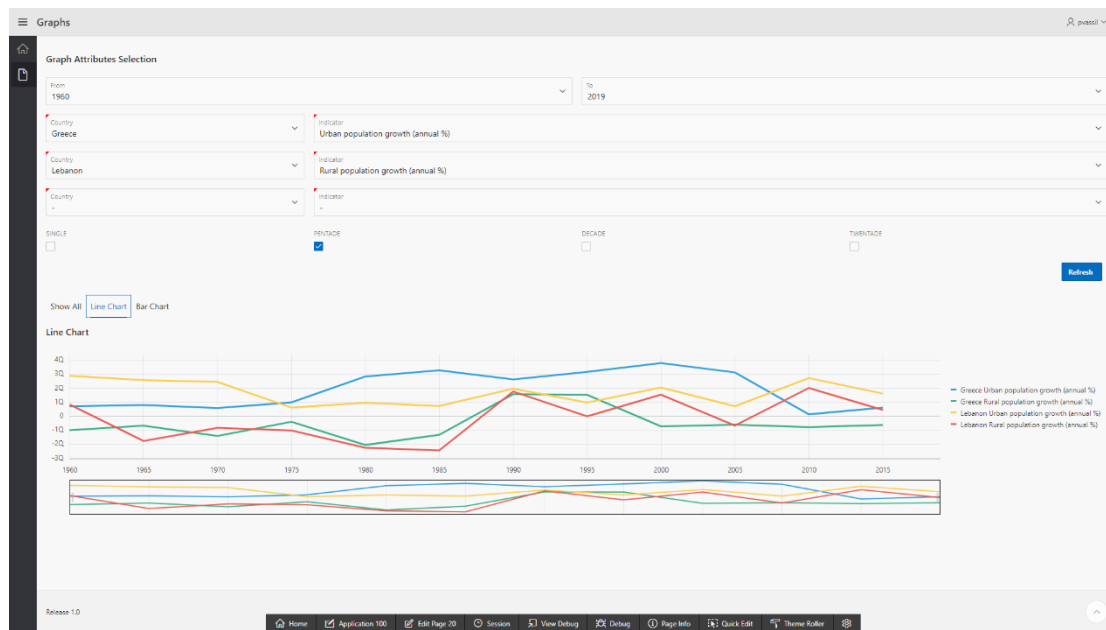
2 Countries 1 Indicator Bar Chart:



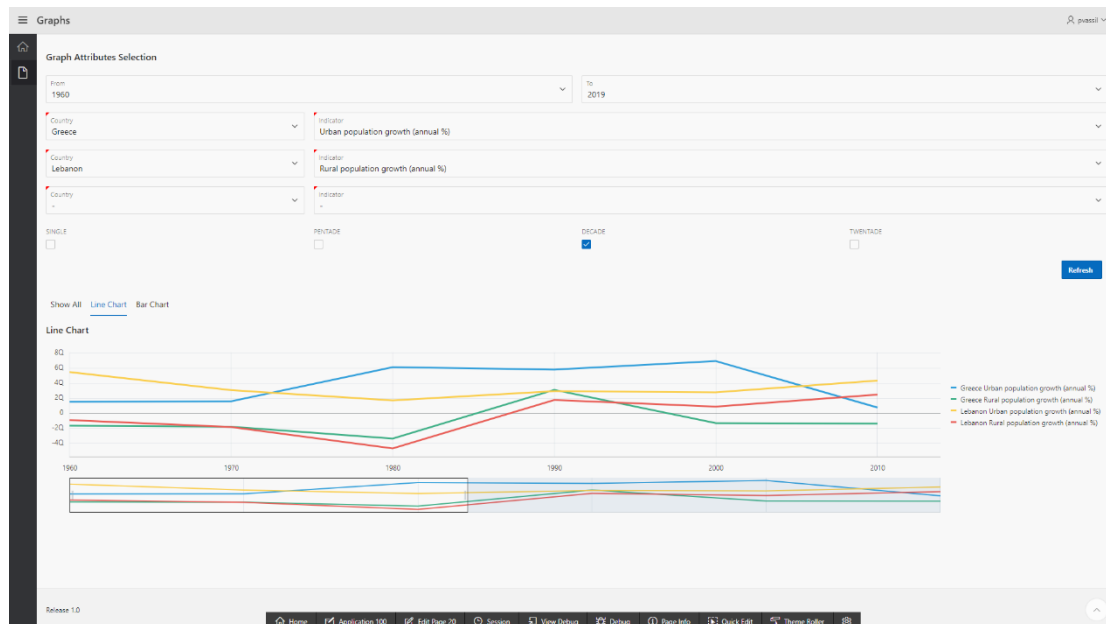
2 Countries 2 Indicators Bar Ranged(2007-2019):



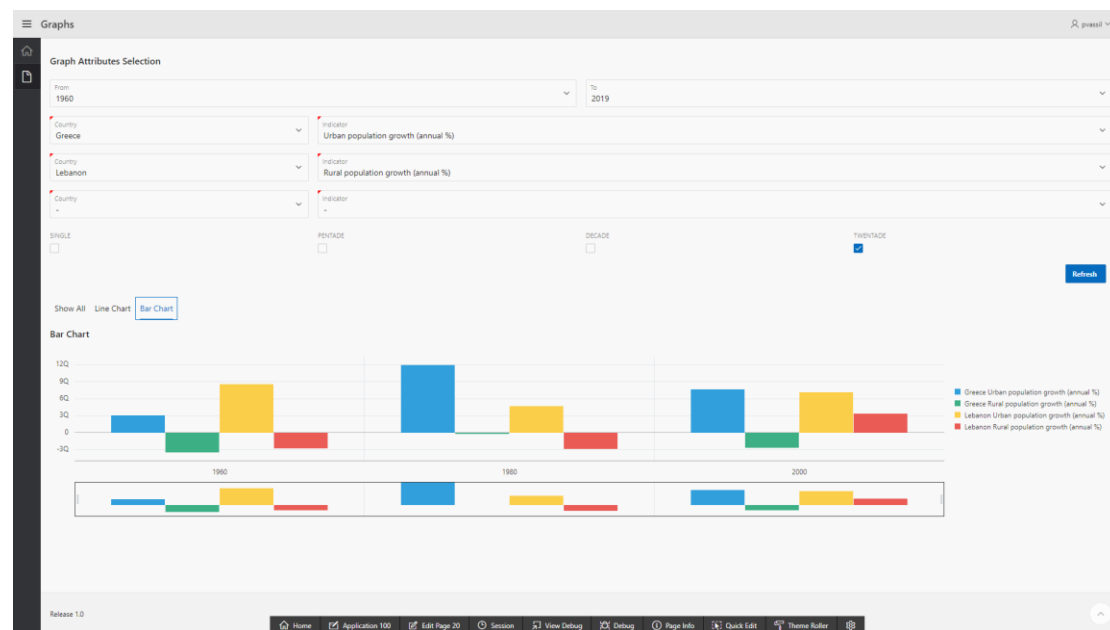
2 Countries 2 Indicators Pentade GroupBy 5:



2 Countries 2 Indicators Pentade GroupBy 10:



2 Countries 2 Indicators Pentade GroupBy 20 BAR:



4 ΤΕΚΜΗΡΙΩΣΗ ΚΑΙ ΛΟΙΠΑ ΣΧΟΛΙΑ

Το Oracle Apex παρέχει πολλές δυνατότητες τις οποίες αλλιώς θα έπρεπε να κάνουμε manually, όπως διαχείριση χρηστών, προσθήκη βιβλιοθηκών για τα γραφήματα κλπ.

Παρόλα αυτά όμως, έχει και μειονεκτήματα. Συγκεκριμένα δεν είναι αρκετά ευέλικτο, και δεν μπορούμε εύκολα να έχουμε γράφημα με όσες χώρες και όσους indicators θέλουμε.