

# Popularea Randomizata A Unei Baze De Date

—

## O Epopee

Theodor Moroianu

November 10, 2020

## **Contents**

<b>1</b>	<b>Introducere</b>	<b>3</b>
<b>2</b>	<b>Ideea din Spatele Proiectului</b>	<b>3</b>
<b>3</b>	<b>Schema Relatională</b>	<b>5</b>
<b>4</b>	<b>Codul SQL de Creare a Bazei de Date</b>	<b>6</b>
<b>5</b>	<b>Popularea Bazei de Date</b>	<b>7</b>
<b>6</b>	<b>Concluzie</b>	<b>9</b>

## 1 Introducere

În acest scurt referat o să prezint dificultățile pe care le-am întâlnit până acum în realizarea proiectului final pentru cursul de *Sisteme de Gestiune a Bazelor de Date*.

Am intalnit multe obstacole, de la instalarea si configurarea sistemului *Oracle Database Express Edition* la gasirea datelor pe care le pot adauga in baza de date si procesarea acestora.



Figure 1: Oracle 18c Express Edition

## 2 Ideea din Spatele Proiectului

M-am gandit la mai multe idei de proiect (mai multe scenarii / situatii pe care sa le modelez cu o baza de date.

Printre ele, cele care mi-au placut cel mai mult au fost:

- Gestiunea unei universitati. Aceasta cuprinde:
  - Lista cu profesorii, studentii, cursurile si salile din universitate.
  - Tabele asociative intre profesori si cursuri, studenti si cursuri, si cursuri si salile din universitate.

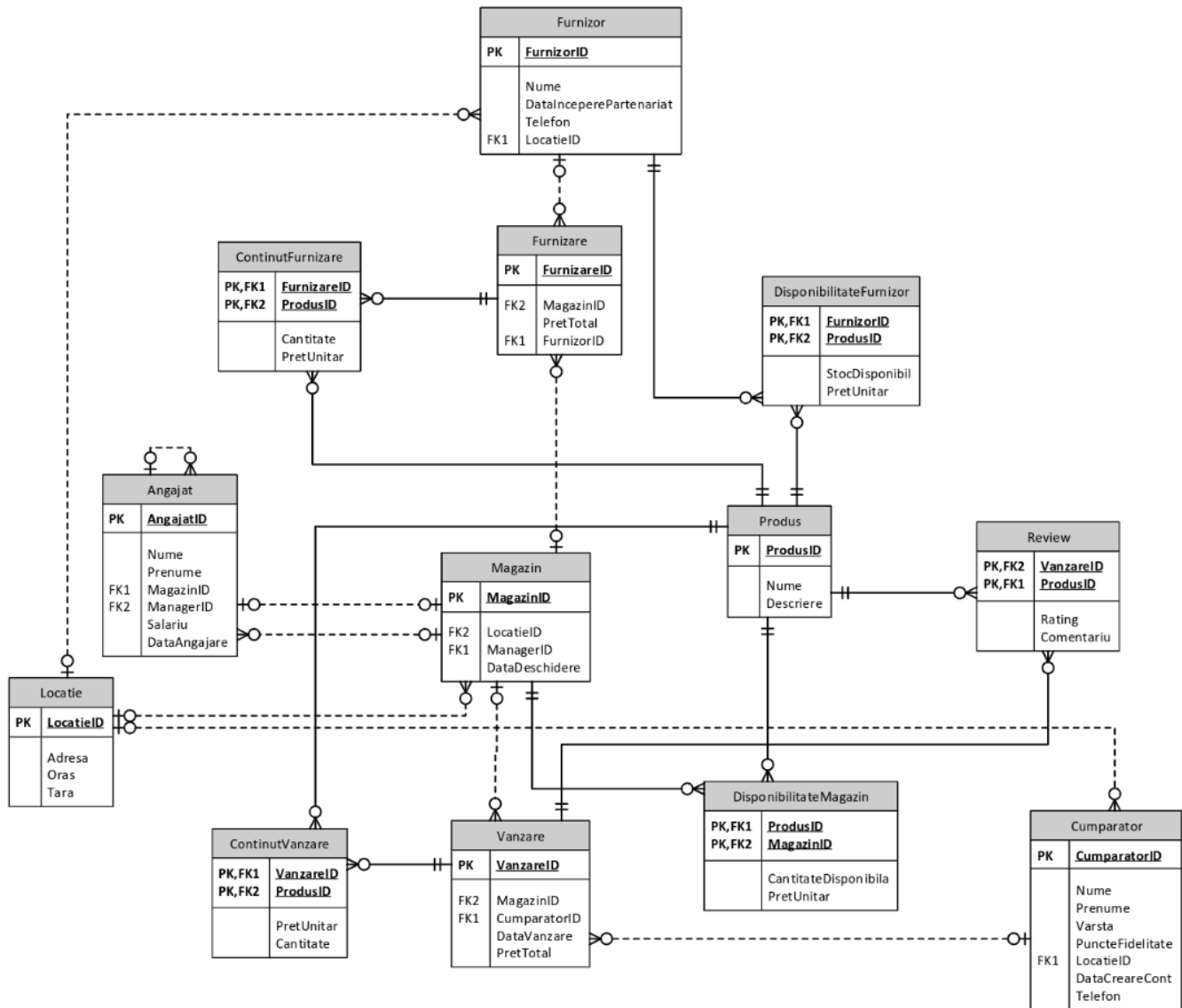
- Istoric al universitatii – ce fosti studenti are universitatea, ce cursuri a luat fiecare student, ce note a avut etc.
- Orice alte informatii care depind de universitate.
- Gestiunea unei primarii. Gestiunea primariei ar cuprinde:
  - Gestiunea functionarilor – ce job au, unde lucreaza, cine e responsabil pentru ei etc.
  - Gestiunea bugetului – O lista de potentiale achizitii, cu importanta si prioritatea lor.
  - Lista cu strazile, parcurile si casele din sector.
  - Toate autorizatiile de constructie.
  - Orice alte elemente pe care le gestioneaza primaria.
- Gestiunea unui lant de magazine. Trebuie tinute:
  - Lista de magazine (cu locatia, manager-ul, produsele etc).
  - Lista de angajati.
  - Lista de clienti si de furnizori.
  - Istoric al cumpararilor.

Dupa mai multe zile de gandire, am decis sa merg pe a 3-a varianta.  
Ideea proiectului este:

1. De-a prezenta un potential model cat mai complet pentru functionarea unui lant de magazine.
2. De-a modela diagrama  $E/R$  si schema relationala, cu lista completa de coloane, pentru a usura intelegerea bazei de date.
3. De-a implementa baza de date in  $SQL$ .
4. De-a arata cateva exemple de utilizare cu cod de  $PL/SQL$  care sa execute diferite operatii pe baza de date.
5. de-a pune la dispozitia utilizatorului un script capabil sa ne populeze tabelele cu informatii aleatoare dar plauzibile.

### 3 Schema Relatională

Schema relatională este data de modelul de mai jos:



## 4 Codul SQL de Creare a Bazei de Date

Codul de creare a bazei de date este in mare parte oarecum trivial. De exemplu, pentru a crea tablea *Vanzare*, avem urmatoarea cerere *SQL*:

```
CREATE TABLE vanzare (  
    vanzareID NUMBER PRIMARY KEY,  
    magazinID NUMBER,  
    cumparatorID NUMBER,  
    dataVanzare DATE,  
    pretTotal NUMBER NOT NULL,  
    FOREIGN KEY (magazinID) REFERENCES  
        magazin(magazinID) ON DELETE SET NULL,  
    FOREIGN KEY (cumparatorID) REFERENCES  
        cumparator(cumparatorID) ON DELETE CASCADE);
```

In schim, uitandu-ne mai atenti la cheile externe dintre tabelul *Angajat* si *Magazin*, observam ca ambii, pentru a isi satisface constrangerile de integritate, au nevoie de existenta celui alt tabel.

Pentru a rezolva acesta situatie de deadlock, declaram un tabel fara constrangerea de cheie externa, pe care o adaugam ulterior.

Astfel, avem urmatoorii pasi:

- Cream tabloul *Angajat*, fara cheia externa spre *Magazin*:

```
CREATE TABLE angajat (  
    angajatID NUMBER PRIMARY KEY,  
    nume VARCHAR2(50),  
    prenume VARCHAR2(50),  
    magazinID NUMBER,  
    managerID NUMBER,  
    salariu NUMBER,  
    dataAngajare Date,  
    FOREIGN KEY (managerID) REFERENCES  
        angajat(angajatID) ON DELETE SET NULL);
```

- Cream tabloul *Magazin*:

```
CREATE TABLE magazin (  
    magazinID NUMBER PRIMARY KEY,  
    locatieID NUMBER,  
    managerID Number,
```

```
dataDeschidere Date,
FOREIGN KEY (managerID) REFERENCES
    angajat(angajatID) ON DELETE SET NULL,
FOREIGN KEY (locatieID) REFERENCES
    locatie(locatieID) ON DELETE SET NULL);
```

- Adaugam tabelului *Angajat* constrangerea de cheie externa:

```
ALTER TABLE angajat
ADD CONSTRAINT angajatfkmagazin
FOREIGN KEY (magazinID) REFERENCES
    magazin(magazinID);
```

## 5 Popularea Bazei de Date

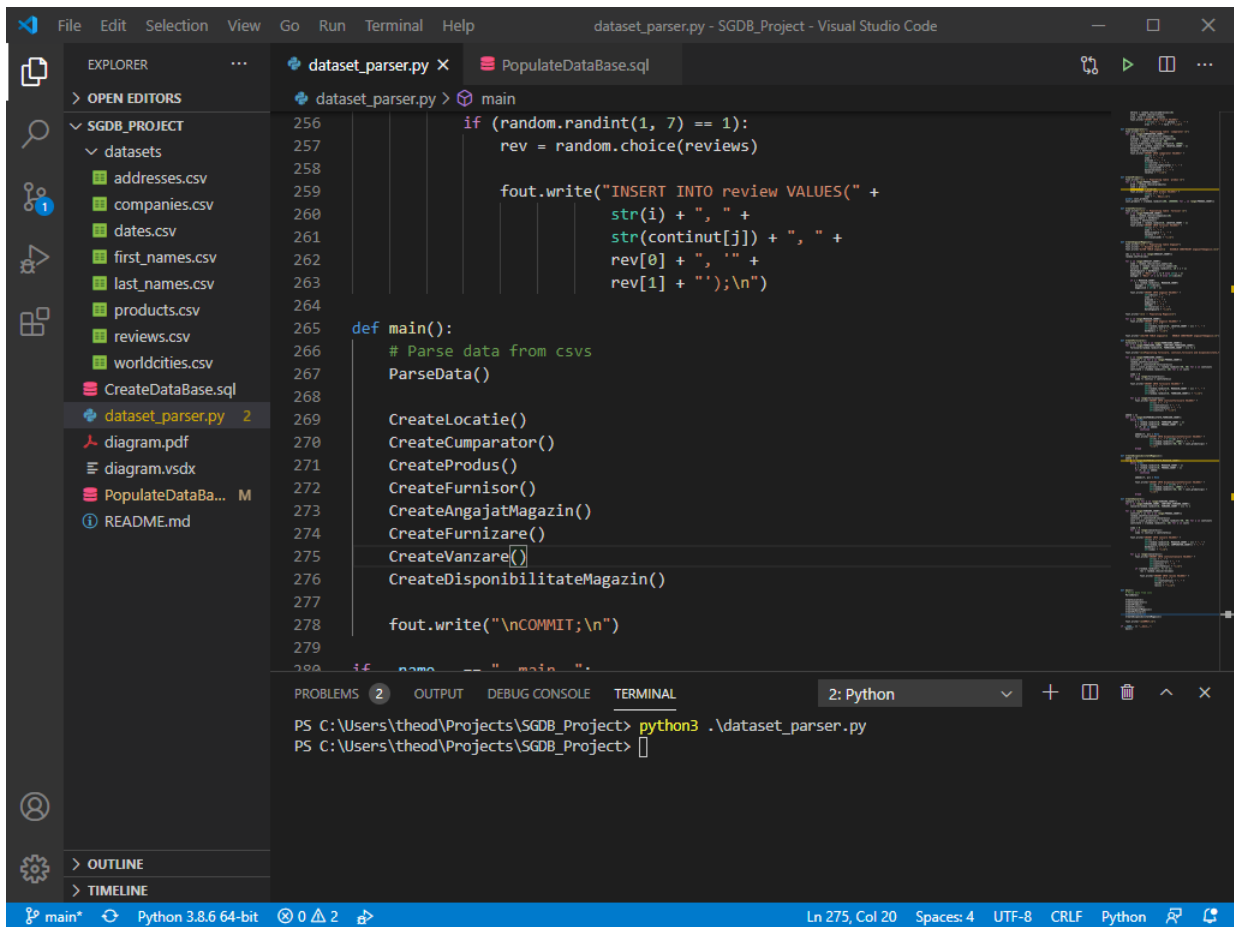
Am incercat sa caut date reale pentru modelul meu, dar evident nu am gasit nimic. Potentialele motive pentru care nu exista astfel de date "*In the wild*" sunt destul de evidente:

- Probleme legate de siguranta datelor (GDPR etc)
- Probleme legate de competitie / secretizare
- Modelele reale sunt mult mai complexe decat al meu

Totusi, nu doream sa imi populez baza de date cu "3-5 inregistrari" asa cum cerea proiectul. Asadar, am stat cateva ore pe Kaggle.com, un website cu dataseturi din aproape oricare domeniu, si mi-am salvat:

- Un dataset cu lista celor mai comune prenume din SUA
- Un dataset cu lista celor mai comune nume de familie din SUA
- Un DS cu o lista de adrese, orase si tari
- Un DS cu o lista de firme (nu doar producatori, dar macar sunt nume coerente de firme)
- O lista cu produse vandute online
- O lista cu reviewuri.

Folosind aceste date, mi-am facut un script in *Python*, care mi-a generat un script de *SQL* care populeaza baza de date cu un numar arbitrar de inregistrari, generate aleator. Pentru a nu incetini prea mult baza de date, am decis sa adaug numai 12.000 de inregistrari.



The screenshot shows the Visual Studio Code interface with the following components:

- EXPLORER:** A file tree on the left showing a project named 'SGDB\_PROJECT' with a subdirectory 'datasets' containing several CSV files (addresses.csv, companies.csv, dates.csv, first\_names.csv, last\_names.csv, products.csv, reviews.csv, worldcities.csv) and a 'CreateDataBase.sql' file. The file 'dataset\_parser.py' is selected and has a count of 2.
- EDITOR:** The main workspace shows the 'dataset\_parser.py' file. It contains a Python script with a 'main' function. The script uses 'random' to generate random data and 'fout.write' to write SQL INSERT statements. The visible code includes:

```
256 if (random.randint(1, 7) == 1):
257     rev = random.choice(reviews)
258
259     fout.write("INSERT INTO review VALUES(" +
260               str(i) + ", " +
261               str(continent[j]) + ", " +
262               rev[0] + ", " +
263               rev[1] + ");\n")
264
265 def main():
266     # Parse data from csvs
267     ParseData()
268
269     CreateLocatie()
270     CreateCumparator()
271     CreateProdus()
272     CreateFurnisor()
273     CreateAngajatMagazin()
274     CreateFurnizare()
275     CreateVanzare()
276     CreateDisponibilitateMagazin()
277
278     fout.write("\nCOMMIT;\n")
279
280 if __name__ == "__main__":
```
- TERMINAL:** At the bottom, the terminal shows the command 'python3 .\dataset\_parser.py' being executed in the directory 'C:\Users\theod\Projects\SGDB\_Project'.
- STATUS BAR:** The bottom status bar indicates 'main\*', 'Python 3.8.6 64-bit', and the current cursor position 'Ln 275, Col 20'.

Datele sunt inserate in baza de date una cate una, care probabil ca nu este cel mai eficient mod, dar este cel mai usor de implementat.



## 6 Concluzie

Acest proiect a fost pana acum unul deosebit de interesant, si prin el am invatat urmatoarele lucruri:

- Cum sa creez o schema relatională profesionala, cu ajutorul aplicatiei *MS Visio 2019*.
- Cum sa gandesc o baza de date coerenta.
- Mi-am amintit cum se fac operatiile de creare / de modificare a constrangerilor si de populare a bazelor de date Oracle.

Urmeaza sa creez si pachetele cerute in proiect, prin care sa facem diferite cereri / modificari pe baza de date, cum ar fi:

- Se adauga un client, o vanzare sau o furnizare.
- Un angajat iti da demisia, trebuie modificati subordonatii sai, sau magazinul (in cazul in care este manager).
- Un magazin se inchide. Trebuie repartizate produsele acestuia la alte magazine.

Codul sursa a generatorului, scripul *SQL* de creare si populare a bazel de date, schema relationala, dataseturile si in viitor si pachetele pentru proiect sunt disponibile la adresa aceasta:

<https://github.com/theodormoroianu/SGDB.Project>.