PROIECT BAZE DE DATE

PROFESOR, STUDENT,

VASILE SILVIU-LAURENȚIU RĂCĂȘEAN IVONA-MARIA,

GRUPA 251

BUCUREȘTI

2020

BAZA DE DATE PENTRU GESTIUNEA UNEI COFETĂRII



PROFESOR, STUDENT,

VASILE SILVIU-LAURENȚIU RĂCĂȘEAN IVONA-MARIA,

GRUPA 251

BUCUREȘTI

2020

CUPRINS:

1. Introducere;
2. Prezentarea modelului și a regulilor acestuia;
3. Diagrama entitate-relație;
4. Descrierea entităților, atributelor, cheilor, relațiilor și a cardinalităților pentru fiecare tabelă;
5. Diagrama conceptuală;
6. Descrierea constrângerilor de integritate, schemele relaționale, crearea tabelelor (inclusiv a constrângerilor), introducere date:

* Tabela CLIENT
* Tabela COMANDĂ
* Tabela PRODUS
* Tabela DESCRIERE COMANDĂ
* Tabela PERSONAL
* Tabela ISTORIC
* Tabela INGREDIENT
* Tabela REȚETĂ
* Tabela FURNIZOR
* Tabela FACTURĂ FISCALĂ
* Tabela DETALIU FACTURĂ

1. Introducere

Care este motivația mea de a-mi alege ca temă gestiunea unei cofetării? Am un răspuns destul de simplu, și anume: pasiunea mea de a face dulciuri și prăjituri, dar și slăbiciunea de a le mânca. Pe lângă acest fapt, consider că este destul de interesant să descopăr cum funcționează o astfel de afacere în ziua de azi.

1. Prezentarea modelului din lumea reală și a regulilor acestuia

Ca în orice cofetărie obișnuită există un personal care se ocupă de buna funcționare a acesteia. Mai precis, de producerea unor prăjituri și torturi iubite de toată lumea. Clienții pot veni în cofetărie să cumpere pe loc propria porție de serotonină, dar pot lăsa și o comandă pe care o vor ridica mai târziu. Pentru acest lucru, clienții trebuie notați, altfel ne-am încurca în comenzi.

Să nu uităm că pentru a face aceste produse, avem nevoie atât de rețetele lor exacte, dar și de cele mai bune ingrediente, pe care le procurăm de la prietenii noștri furnizori. Bineînțeles că trebuie să reținem și sumele pe care le plătim, păstrând toate facturile fiscale.

Pentru a ușura munca, ne putem folosi de baza de date ce am creat-o mai jos.

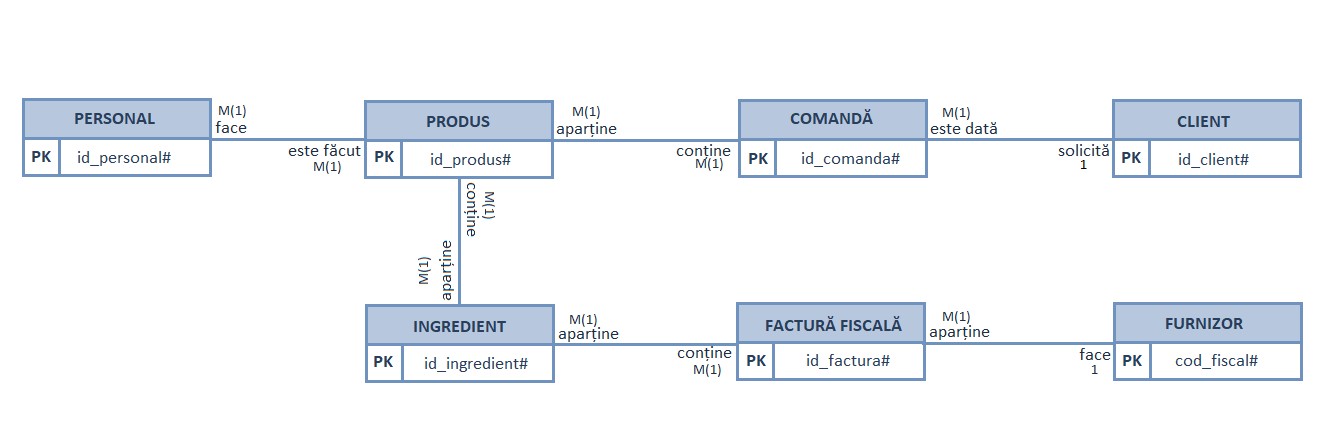
Această bază de date reține informații despre cofăterie, fiecare angajat ce lucrează în ea (fie cofetari, contabili sau îngrijitori), fiecare produs vândut (prăjituri și torturi), inclusiv rețetele acestora, precum și furnizorii de ingrediente necesare pentru producție. Baza de date reține și informații despre clienții care vin și dau comenzi.

Scopul creării acestei baze de date este acela de a putea ține evidența tuturor comenzilor date de către clienți, de produsele și ingredientele necesare creării acestora și a facturilor fiscale dintre cofetărie și furnizori.

Ca reguli avem următoarele afirmații:

* Nu toti angajații din personal sunt cofetari (de ex: contabil sau femeie de serviciu), iar aceștia vor avea cheia externă nulă în tabela ISTORIC;
* Toți cofetarii din personal vor avea un istoric ce reține produsul/produsele pe care le-au făcut;
* Fiecare produs are o rețetă unică ce conține mai multe ingrediente;
* Pentru fiecare achiziționare de ingrediente de la furnizori, avem nevoie de o factură fiscală pentru a ține evidența acestora;
* Sunt reținute în baza de date doar comenzile ce vor fi făcute pe termen mai lung (de exemplu: un client dorește ca în data de 10.01.2021 să ia un tort de ciocolată);
* Trebuie să reținem câteva date despre clientul care dă o comandă, pentru a avea o garanție la ridicarea acesteia.

3. Diagrama entitate-relație



4. Descrierea entităților, atributelor, cheilor, relațiilor și a cardinalităților pentru fiecare tabelă

* PERSONAL :
* reține următoarele informații despre angajații cofetăriei: id-ul angajatului (id\_personal INT(11)), numele (nume VARCHAR(20)), prenumele (prenume VARCHAR(20)), meseria pe care o are (meserie VARCHAR(20)), adresa de domiciliu (adresa VARCHAR(40)) și numărul de telefon (telefon INT(11));
* cheia primară a tabelei este “id\_personal”, fiecare angajat are un id unic;
* personalul este cel care se ocupă de producerea prăjiturilor: un cofetar poate să facă mai multe produse, iar un produs poate să fie făcut de mai mulți cofetari, rezultând o cardinalitate maximă **many-many** cu tabela “PRODUS”;
* relația PERSONAL face PRODUS are cardinalitatea minimă **one-one**, deoarece un cofetar trebuie să facă cel puțin un produs, iar un produs trebuie să fie făcut de către cel puțin un cofetar.
* PRODUS :
* tabela PRODUS reține date despre toate produsele create în cofetărie;
* reține următoarele informații despre produsele cofetăriei: id\_produs (identificator unic pentru fiecare produs, cheie primară, INT(11)), denumirea acestuia (denumire VARCHAR(20)) și prețul (preț FLOAT(5,2));
* un produs poate să conțină unul sau mai multe ingrediente, iar un ingredient poate să aparțină mai multor produse, rezultând o cardinalitate maximă **many-many** cu tabela INGREDIENT;
* un produs trebuie să conțină cel puțin un ingredient, iar un ingredient trebuie să aparțină cel puțin unui produs, rezultând o cardinalitate minimă **one-one** cu tabela INGREDIENT;
* un produs poate să fie făcut de unul sau mai mulți cofetari, iar un cofetar poate să facă unul sau mai multe produse, rezultând o cardinalitate maximă **many-many** cu tabela PRODUS;
* un produs trebuie să fie făcut de către cel puțin un cofetar, iar un cofetar trebuie să facă cel puțin un produs, rezultând o cardinalitate minimă **one-one** cu tabela PRODUS;
* un produs poate să aparțină uneia sau mai multor comenzi, iar o comanda poate să conțină unul sau mai multe produse, rezultând o cardinalitate maximă **many-many** cu tabela COMANDĂ;
* un produs trebuie să aparțină cel puțin unei comenzi, iar o comandă trebuie să conțină cel puțin un produs, rezultând o cardinalitate minima **one-one** cu tabela COMANDĂ.
* COMANDĂ :
* în tabela COMANDĂ sunt reținute toate comenzile solicitate, ce vor fi făcute în viitor;
* reține următoarele date despre fiecare comandă dată de către clienți: id\_comanda (identificator unic al comenzilor, cheie primară, INT(11)) și data la care clientul dorește să ridice comanda (data DATE) ;
* o comandă poate să conțină unul sau mai multe produse, iar un produs poate să aparțină uneia sau mai multor comenzi, rezultând o cardinalitate maximă **many-many** cu tabela PRODUS;
* o comandă trebuie să conțină cel puțin un produs, iar un produs trebuie să aparțină cel puțin unei comenzi, rezultând o cardinalitate minimă **one-one** cu tabela PRODUS;
* o comandă poate să fie dată de către un singur client, iar un client poate să solicite una sau mai multe comenzi, rezultând o cardinalitate maximă **many-one** cu tabela CLIENT;
* o comandă trebuie să fie dată de un singur client, iar un client trebuie sa solicite cel puțin o comandă, rezultând o cardinalitate minimă **one-one** cu tabela CLIENT.
* CLIENT :
* tabela CLIENT este creată pentru a avea o garanție la ridicarea comenzilor;
* avem nevoie de următoarele informații despre clienți: id\_client (identificator unic, cheie primară, INT(11)), numele clientului (nume VARCHAR(20)) și numărul de telefon pentru orice eventualitate (telefon INT(11));
* un client poate să solicite una sau mai multe comenzi, iar o comandă poate să fie dată de un singur client, rezultând o cardinalitate maximă **many-one** cu tabela COMANDĂ;
* un client trebuie să solicite cel puțin o comandă, iar o comandă trebuie să fie dată de către un singur client, rezultând o cardinalitate minimă **one-one** cu tabela COMANDĂ.
* INGREDIENT :
* tabela INGREDIENT reține toate ingredientele necesare pentru producerea prăjiturilor și a torturilor;
* reține informații despre ingredientele necesare producerii lor: id\_ingredient (identificator unic, cheie primară INT(11)) și denumirea lor (denumire VARCHAR(20));
* un ingredient poate să aparțină unuia sau mai multor produse, iar un produs poate să conțină unul sau mai multe ingredient, rezultând o cardinalitate maximă **many-many** cu tabela PRODUS;
* un ingredient trebuie să aparțină cel puțin unui produs, iar un produs trebuie să conțină cel puțin un ingredient, rezultând o cardinalitate minimă **one-one** cu tabela PRODUS;
* un ingredient poate să aparțină uneia sau mai multor facturi fiscale, iar o factură fiscală poate să conțină unul sau mai multe produse, rezultând o cardinalitate maximă **many-many** cu tabela FACTURĂ FISCALĂ;
* un ingredient trebuie să aparțină cel puțin unei facturi fiscale, iar o factură fiscală trebuie să conțină cel puțin un ingredient, rezultând o cardinalitate minima one-one cu tabela FACTURĂ FISCALĂ;
* FACTURĂ FISCALĂ :
* tabela FACTURĂ FISCALĂ este creată pentru a memora detalii despre fiecare achiziție de ingrediente de la furnizori;
* reține date despre facturile dintre cofetărie și furnizori: id\_factură (identificator unic, cheie primară, INT) și data în care se face achiziția (data DATE);
* o factură fiscală poate să conțină unul sau mai multe ingrediente, iar un ingredient poate să aparțină uneia sau mai multor facturi fiscale, rezultând o cardinalitate maximă **many-many** cu tabela INGREDIENT;
* o factură fiscală trebuie să conțină cel puțin un ingredient, iar un ingredient trebuie să aparțină cel puțin unei facturi, rezultând o cardinalitate minimă **one-one** cu tabela INGREDIENT;
* o factură fiscală poate să aparțină unui singur furnizor, iar un furnizor poate face una sau mai multe facturi fiscale, rezultând o cardinalitate maximă **many-one** cu tabela FURNIZOR;
* o factură fiscală trebuie să aparțină unui singur furnizor, iar un furnizor trebuie să facă cel puțin o factură fiscal, rezultând o cardinalitate minima **one-one** cu tabela FURNIZOR;
* FURNIZOR :
* reține informații despre furnizorii cofetăriei: cod\_fiscal (identificator unic, cheie primară, INT), numele furnizorului (denumire VARCHAR(20)) și numărul de telefon (telefon INT(11));
* un furnizor poate să facă una sau mai multe facturi, iar o factură poate să aparțină unui singur furnizor, rezultând o cardinalitate maximă **many-one** cu tabela FACTURĂ FISCALĂ;
* un furnizor trebuie să facă cel puțin o factură fiscală, iar o factură fiscală trebuie să aparțină unui singur furnizor, rezultând o cardinalitate minima **one-one** cu tabela FACTURĂ FISCALĂ.

5. Diagrama conceptuală:

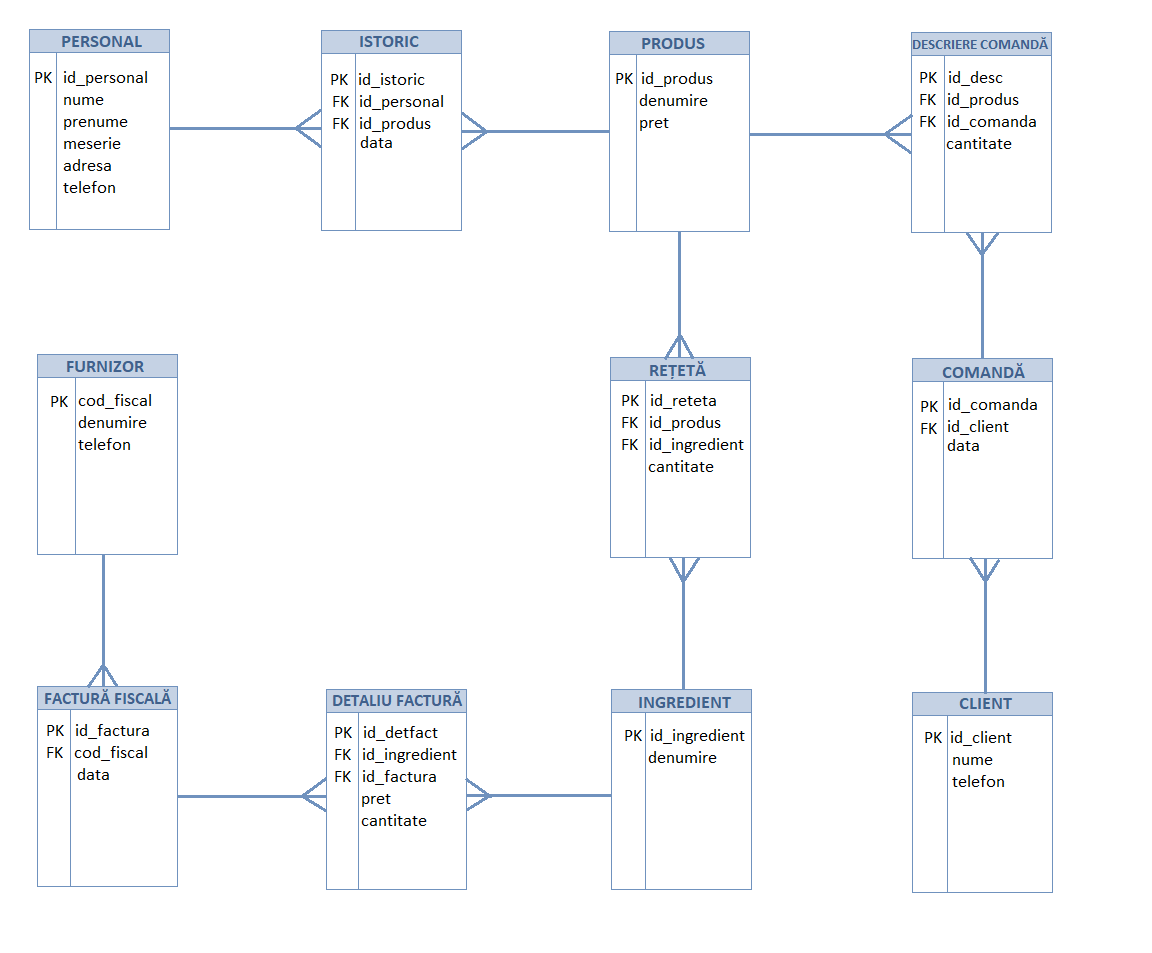
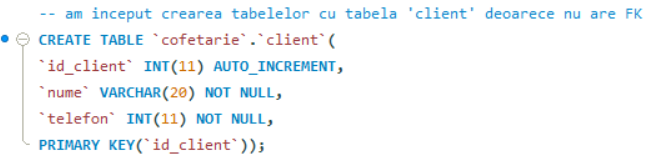


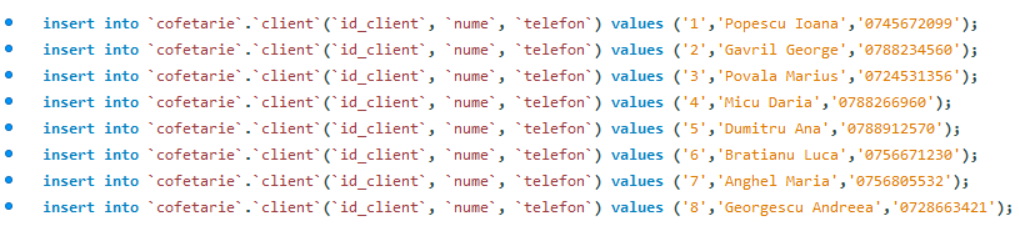
Diagrama conceptuală, față de diagrama entitate-relație, are în minus relațiile și cardinalitățile, aparând, în schimb, cheile externe și tabelele asociative. Tabele asociative: ISTORIC, REȚETĂ, DETALIU FACTURĂ și DESCRIERE COMANDĂ sunt menite să rezolve relațiile Many to Many. Am ales să implementez și să populez modelul de mai sus în MySQL Workbench.

Tabela CLIENT:

Tabela CLIENT are următoarele constrângeri: id\_istoric este PRIMARY KEY având clauza AUTO\_INCREMENT, iar câmpurile ”nume” și ”telefon” nu pot fi nule.



Inserarea datelor în tabela CLIENT cu INSERT EXPLICIT:



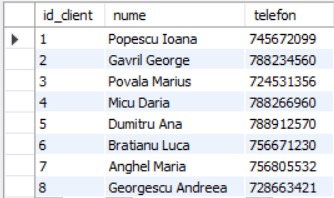
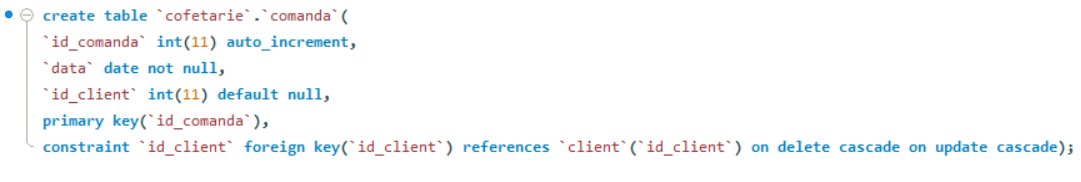
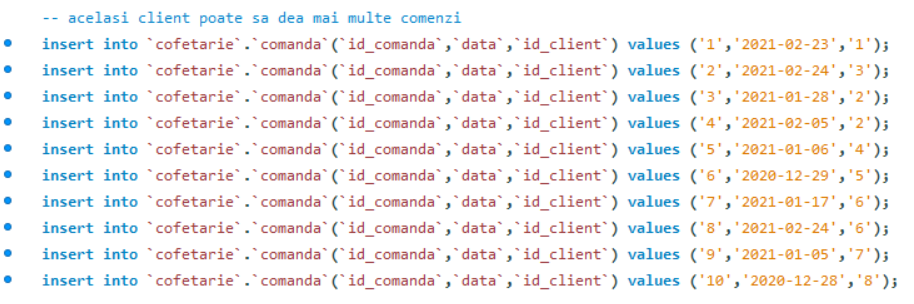


Tabela COMANDĂ:

În această tabela avem următoarele constrângeri: id\_comanda (PRIMARY KEY cu tipul de date INT(11) AUTO\_INCREMENT), data (cu tipul de date DATE, nu poate fi nulă), iar deoarece relația dintre tabelele COMANDĂ și CLIENT este many-one, cheia primara id\_client din tabela CLIENT devine cheie externă pentru tabela COMANDĂ. Constrangerea FOREIGN KEY de pe coloana id\_client are clauzele ON DELETE CASCADE si ON UPDATE CASCADE pentru a permite ștergerea și actualizarea unui comenzi odată cu modificarea datelor din tabela CLIENT.



Inserarea datelor în tabela COMANDĂ cu INSERT EXPLICIT:



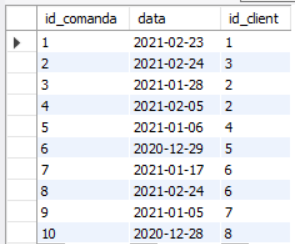
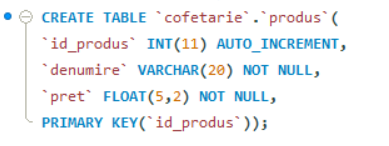


Tabela PRODUS:

Tabela PRODUS are următoarele constrângeri: id\_produs este PRIMARY KEY având clauza AUTO\_INCREMENT, iar câmpurile ”denumire” și ”preț” nu pot fi nule.



Inserarea datelor în tabela PRODUS cu INSERT EXPLICIT:

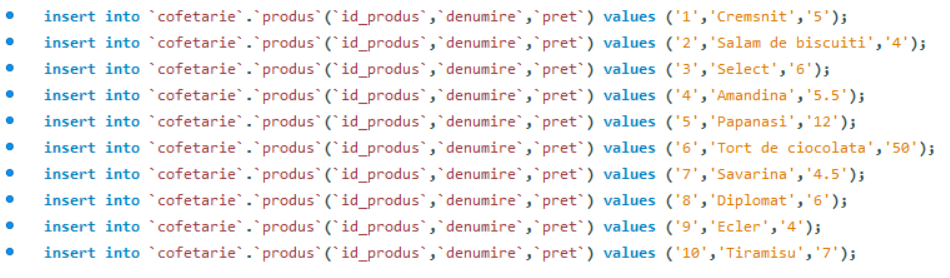
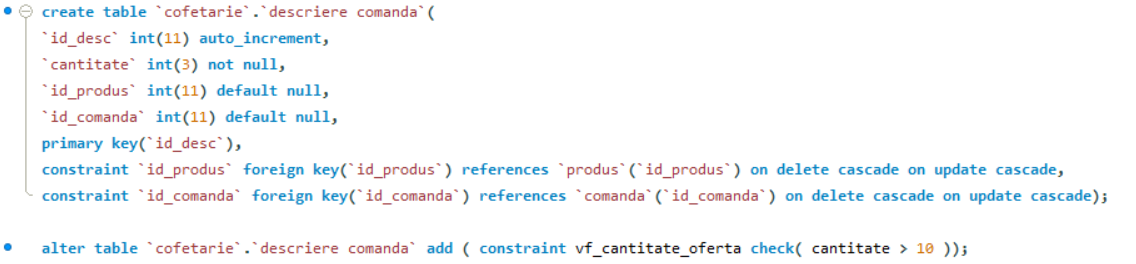




Tabela DESCRIERE COMANDĂ:

Pentru a rezolva relația Many to Many dintre tabelele PRODUS și COMANDĂ, am creat tabela DESCRIERE COMANDĂ cu următoarele câmpuri: id\_desc (identificatorul unic cu tipul de date INT(11)) și cantitate (numărul de produse comandate, fiind un INT(3)) și următoarele constrângeri: id\_desc este cheie primară, cantitatea nu poate avea o valoare nulă deoarece trebuie să știm numărul de produse comandate, iar id\_produs și id\_comanda devin chei externe în această tabelă. Constrângerile FOREIGN KEY de pe coloana id\_personal și id\_comanda au clauzele ON DELETE CASCADE si ON UPDATE CASCADE pentru a permite ștergerea și actualizarea unei descrieri odată cu modificarea datelor din cele două tabele: COMANDĂ și PRODUS. Am introdus și constrângerea CHECK: în cazul în care un client dă o comandă cu un număr de produse mai mare decât 10, primește o reducere.



Inserarea datelor în tabela DESCRIERE COMANDĂ cu INSERT EXPLICIT:



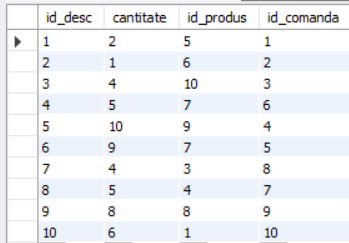
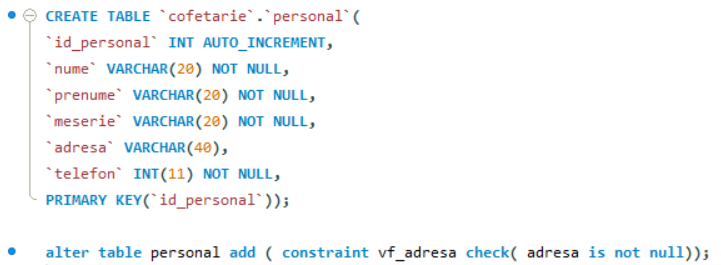


Tabela PERSONAL:

Tabela PERSONAL are următoarele constrângeri: id\_personal este PRIMARY KEY având clauza AUTO\_INCREMENT, iar câmpurile ”nume”, ”prenume”, ”meserie”, și ”telefon” nu pot fi nule deoarece aceste informații despre angajați sunt necesare. Pentru câmpul ”adresa”, care poate fi nul, am pus o constrângere CHECK care va verifica dacă adresa este introdusă sau nu.



Inserarea datelor în tabela PERSONAL cu INSERT EXPLICIT:



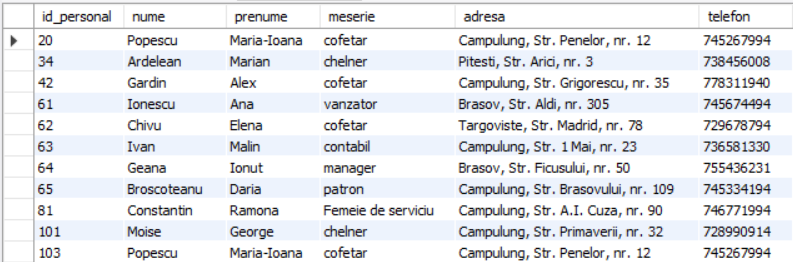
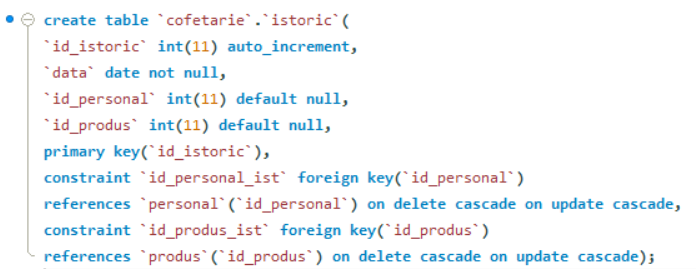


Tabela ISTORIC:

Pentru a rezolva relația Many to Many dintre tabelele PRODUS și PERSONAL, am creat tabela ISTORIC cu următoarele câmpuri: id\_istoric (identificatorul unic cu tipul de date INT(11)) și data (data producerii produsului, cu tipul de date DATE) și următoarele constrângeri: id\_istoric este cheie primară, data nu poate avea o valoare nulă, iar id\_produs și id\_personal devin chei externe în această tabelă. Constrangerile FOREIGN KEY de pe coloana id\_personal și id\_produs au clauzele ON DELETE CASCADE si ON UPDATE CASCADE pentru a permite ștergerea și actualizarea unui istoric odată cu modificarea datelor din cele doua tabele: PERSONAL și PRODUS.



Inserarea datelor în tabela ISTORIC cu INSERT EXPLICIT:

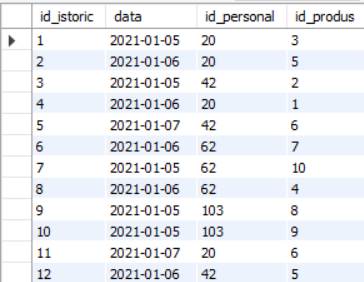
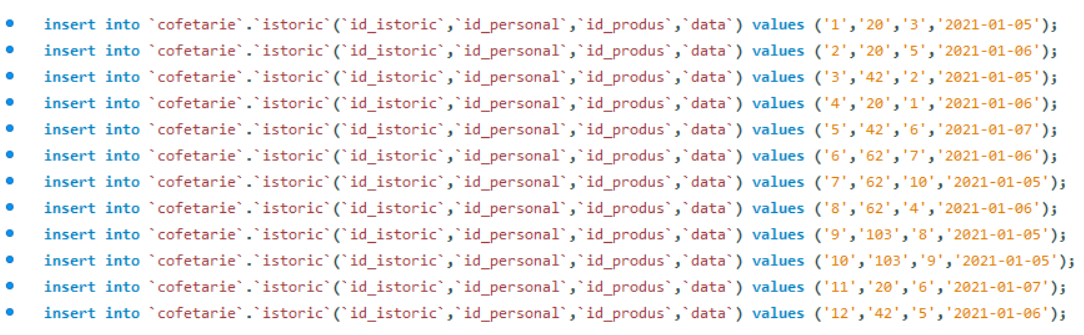
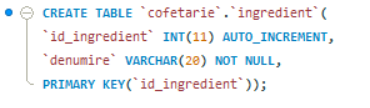
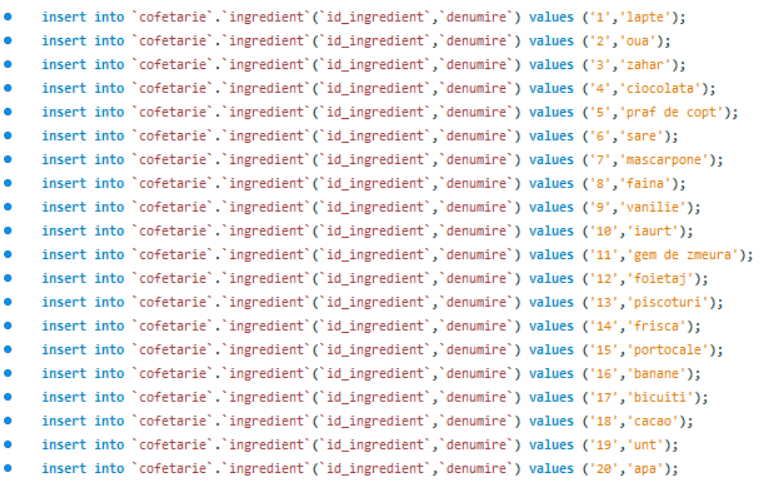


Tabela INGREDIENT:

Tabela INGREDIENT are următoarele constrângeri: id\_ingredient este PRIMARY KEY având clauza AUTO\_INCREMENT, iar câmpul ”denumire” nu poate fi nul deoarece este necesar să știm numele ingredientului.



Inserarea datelor în tabela INGREDIENT cu INSERT EXPLICIT:



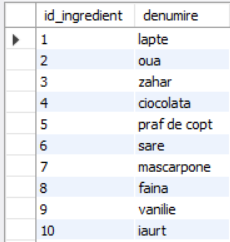
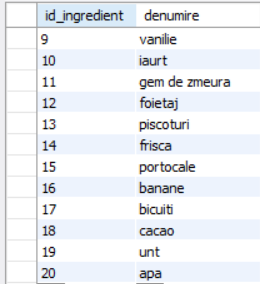
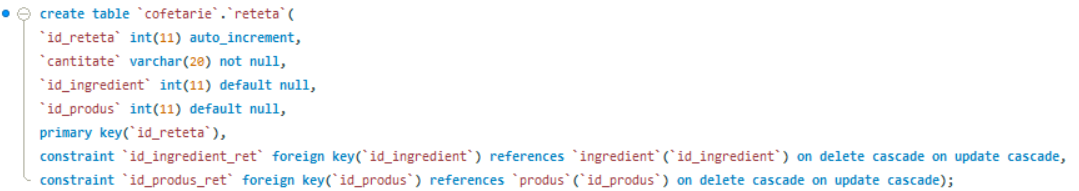
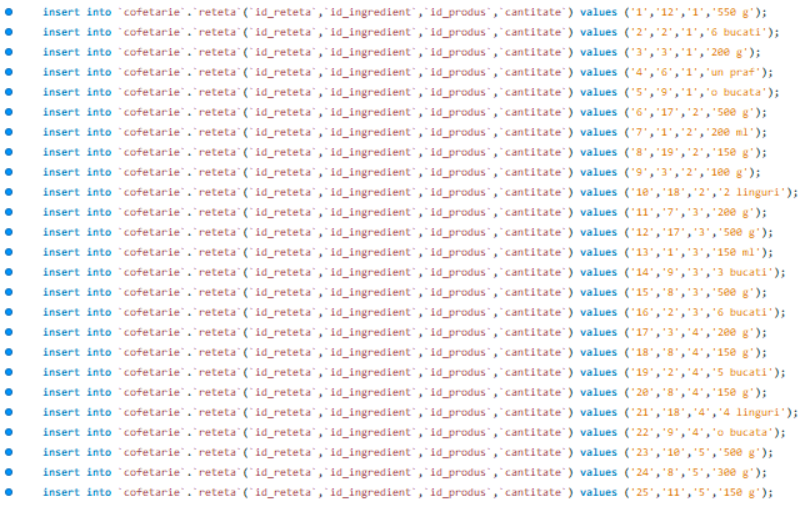
 

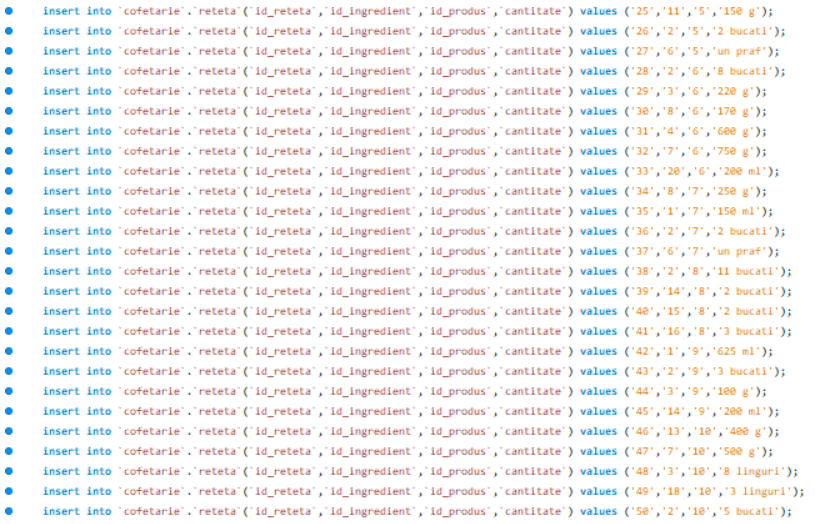
Tabela REȚETĂ

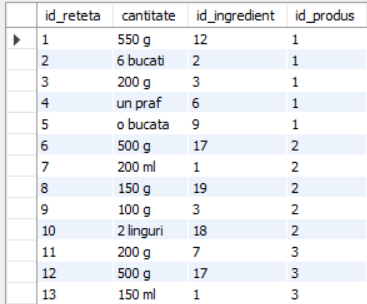
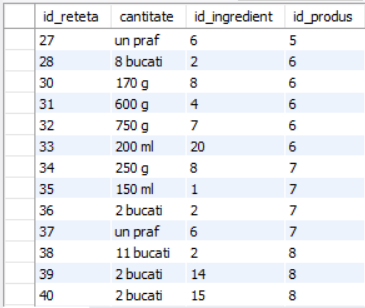
Pentru a rezolva relația Many to Many dintre tabelele PRODUS și INGREDIENT, am creat tabela RETETA cu următoarele câmpuri: id\_reteta (identificatorul unic al unei rețete cu tipul de date INT(11)) și cantitate (cantitatea pentru fiecare produs ce urmează să fie folosit la producerea unei prăjituri, tipul de date VARCHAR(20)) și următoarele constrângeri: id\_reteta este cheie primară, cantitatea nu poate avea o valoare nulă, iar id\_produs și id\_ingredient devin chei externe în această tabelă. Constrangerile FOREIGN KEY de pe coloana id\_ingredient și id\_produs au clauzele ON DELETE CASCADE si ON UPDATE CASCADE pentru a permite ștergerea și actualizarea unei rețete odată cu modificarea datelor din cele două tabele: INGREDIENT și PRODUS.



Inserarea datelor în tabela REȚETĂ cu INSERT EXPLICIT:





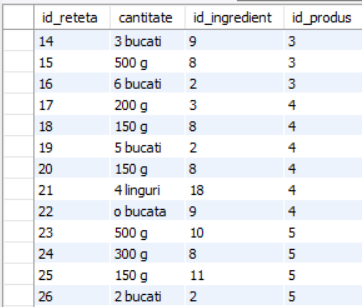
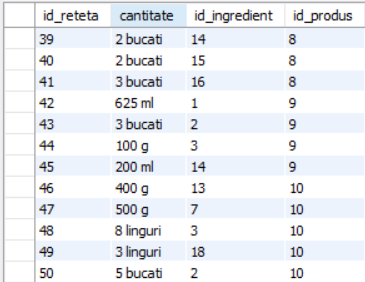
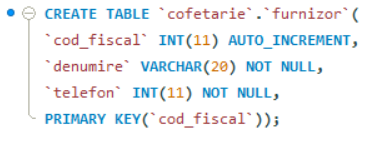
 

Tabela FURNIZOR:

Tabela FURNIZOR are următoarele constrângeri: cod\_fiscal este PRIMARY KEY având clauza AUTO\_INCREMENT, iar câmpurile ”denumire” și ”telefon” nu pot fi nule deoarece este necesar să știm aceste informații despre furnizori.



Inserarea datelor în tabela FURNZIOR cu INSERT EXPLICIT:

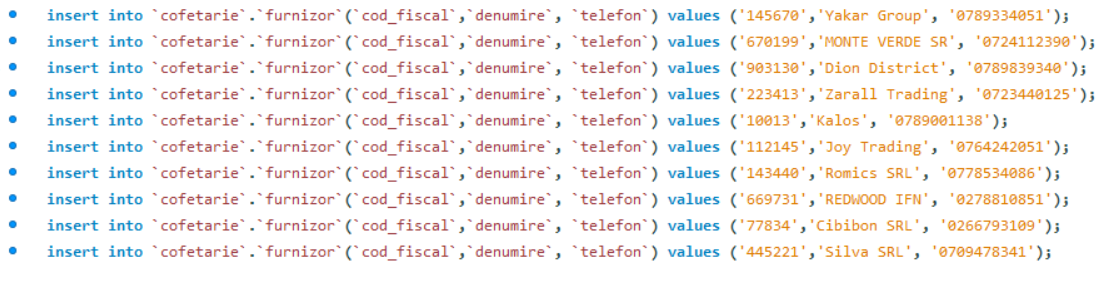
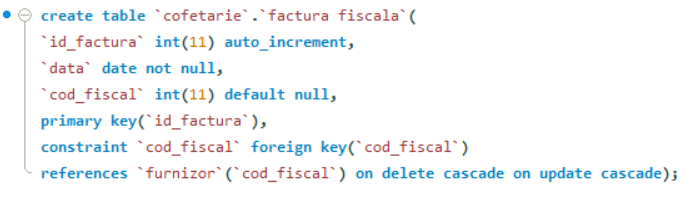


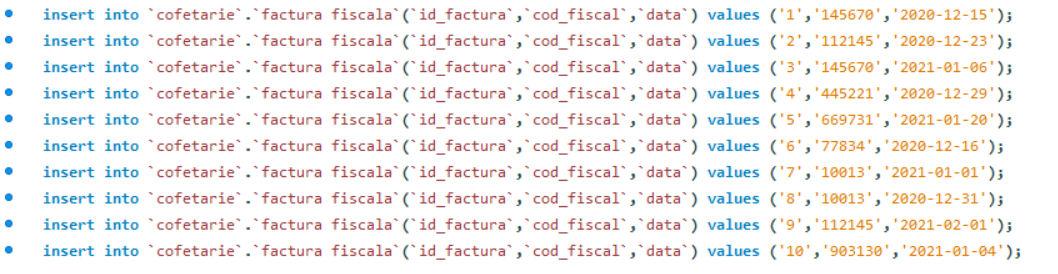


Tabela FACTURĂ FISCALĂ:

În această tabelă avem următoarele constrângeri: id\_factura (PRIMARY KEY cu tipul de date INT(11) AUTO\_INCREMENT), data (cu tipul de date DATE, nu poate fi nulă), iar deoarece relația dintre tabelele FURNIZOR și FACTURĂ FISCALĂ este many-one, cheia primară din tabela FURNIZOR devine cheie externă pentru tabela FACTURĂ FISCALĂ. Constrângerea FOREIGN KEY de pe coloana cod\_fiscal are clauzele ON DELETE CASCADE si ON UPDATE CASCADE pentru a permite ștergerea și actualizarea unui facturi odată cu modificarea datelor din tabela FURNIZOR.



Inserarea datelor în tabela FACTURĂ FISCALĂ cu INSERT EXPLICIT:



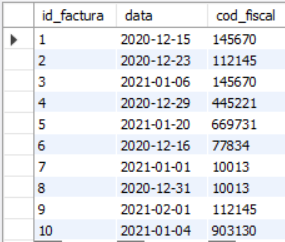
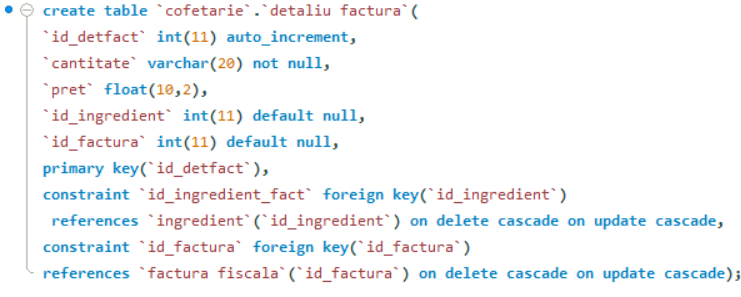


Tabela DETALIU FACTURĂ:

Pentru a rezolva relația Many to Many dintre tabelele FACTURĂ FISCALĂ și INGREDIENT, am creat tabela DETALIU FACTURĂ cu următoarele câmpuri: id\_detfact (identificatorul unic pentru fiecare descriere a facturilor, cu tipul de date INT(11)), cantitate (cantitatea procurată a unui produs, cu tipul de date VARCHAR(20)) și preț (prețul produsului, cu tipul de date FLOAT(10,2)) și următoarele constrângeri: id\_detfact este cheie primară, cantitatea și prețul nu pot avea valori nule, iar id\_ingredient și id\_factura devin chei externe în această tabelă. Constrângerile FOREIGN KEY de pe coloana id\_ingredient și id\_factura au clauzele ON DELETE CASCADE si ON UPDATE CASCADE pentru a permite ștergerea și actualizarea descrierii facturii odată cu modificarea datelor din cele două tabele: FACTURĂ FISCALĂ și INGREDIENT.



Inserarea datelor în tabela DETALIU FACTURĂ cu INSERT EXPLICIT:

