Baza de date

« HappyCoffee »

**CUPRINS**

0. [TEMA](#Tema)

1. [Descrierea modelului real](#DescriereA)

2. [Prezentarea constrângerilor](#Prezentarea)

3. [Descrierea entităților](#DescriereE)

4. [Descrierea relațiilor](#DescriereR)

5. [Descrierea atributelor](#DescriereA)

6. [Realizarea diagramei entitate-relație](#RealizareaDER)

7. [Realizarea diagramei conceptuale](#RealizareDC)

8. [Enumerarea schemelor relaționale](#Enumerare)

9. [Realizarea normalizării](#RealizareNormalizare)

9.1 [FN1](#FN1)

9.2  [FN2](#FN2)

9.3 [FN3](#FN3)

10[. Crearea unei secvențe](#CreareSecvente)

11. [Crearea tabelelor în SQL și inserarea de date](#CreareTabele)

12.  [Formulați în limbaj natural și implementați 5 cereri SQL complexe](#Cereri)

13.  [Implementarea a 3 operații de actualizare și de suprimare a datelor utilizând subcereri](#StergereUpdate)

1. **Tema**

HappyCoffee este o cafenea care dorește să implementeze un sistem de baze de date pentru a gestiona comenzile clienților, stocurile de produse și informațiile despre clienți. Sistemul trebuie să fie capabil să înregistreze comenzile zilnice, să gestioneze inventarul de produse și să mențină un istoric al clienților fideli.

1. **Descrierea modelului real, a utilității acestuia și a regulilor de funcționare.**

Aceasta baza de date va gestiona toate aspectele operationale ce tin de cafeneaua „HappyCoffee”, incluzând plasarea și procesarea comenzilor, gestionarea clienților și angajaților, înregistrarea plăților și administrarea stocurilor de produse. Fiecare dintre entitățile principale din baza de date - Comenzi, Bauturi, Clienti, Plati, Angajati, Zile\_lucrate și Produs - joacă un rol esențial în funcționarea zilnică a cafenelei.

Fiecare comandă plasată va conține clientul este identificat unic și se înregistrează cu numele și adresa de email, asigurând o experiență personalizată și urmărirea fidelității acestuia, angajatul, fiecare având un rol specific, sunt asociați cu comenzile pe care le procesează și zilele lucrate, detaliate în entitatea Zile\_lucrate cu datele și orele de început și sfârșit ale turelor, conținutul acesteia cu toate detaliile specifice și plățile care sunt documentate cu dată și metodă de plată utilizată, oferind transparență financiară și posibilitatea de raportare precisă.

1. **Prezentarea constrângerilor (restricții, reguli) impuse asupra modelului.**

Adresele de email pentru clienți și angajați trebuie să fie unice.

Identificatorii (ID-uri) pentru toate entitățile trebuie să fie unici (ID\_Comanda, ID\_Bauturi, ID\_Clienti, ID\_plata, ID\_Angajat, ID\_Zile\_lucrate, ID\_Produs).

Fiecare comandă trebuie să fie asociată cu un client existent (ID\_Clienti) și un angajat existent (ID\_Angajati).

Fiecare conținut de comandă (ID\_Bauturi) trebuie să aparțină unei comenzi existente (ID\_Comanda).

Fiecare plată trebuie să fie asociată cu o comandă existentă.

Prețul produselor (pret\_produs) și totalul comenzii (total) trebuie să fie valori pozitive.

Cantitatea produselor (cantitate) trebuie să fie un număr întreg pozitiv.

Data plății (plata\_data), data comenzii (data) și ziua lucrată (ziua\_data) trebuie să fie date valide și nu pot fi în viitor.

Ora de început (zi\_inceput) trebuie să fie mai mică decât ora de final (zi\_final) pentru zilele lucrate.

Nu se pot șterge produse care sunt parte din conținutul comenzilor existente.

Fiecare angajat trebuie să aibă un rol definit (de exemplu: barista, casier).

1. **Descrierea entităților, incluzând precizarea cheii primare.**

Comenzi: Identifică fiecare comandă plasată în cafenea, având *ID\_Comanda* ca si cheie primară. Aceasta entitate stochează informații esențiale pentru fiecare comandă, cum ar fi identificatorul conținutului comenzii, identificatorul angajatului care a procesat comanda și identificatorul clientului care a plasat comanda.

Bauturi: Detaliază conținutul fiecărei comenzi, cu *ID\_Bauturi* ca si cheie primară. Această entitate include informații despre data la care a fost plasată comanda și totalul sumei pentru comanda respectivă, oferind detalii specifice despre fiecare element al comenzii.

Clienti: Reprezintă clienții cafenelei, fiecare client fiind identificat unic prin *ID\_Clienti* care este cheia primară. Această entitate stochează informații despre clienți, inclusiv numele și adresa de email, facilitând urmărirea și gestionarea relațiilor cu clienții.

Plati: Înregistrează detaliile plăților, având *ID\_plata* ca si cheie primară. Această entitate conține informații despre data plății și metoda de plată folosită, asigurând transparența și gestionarea eficientă a tranzacțiilor financiare.

Angajati: Gestionează informațiile despre angajații cafenelei, cu *ID\_Angajat* ca si cheie primară. Această entitate include detalii despre rolul angajatului, adresa de email și numărul de telefon, facilitând gestionarea resurselor umane și alocarea sarcinilor.

Zile\_lucrate: Înregistrează zilele lucrate de angajați, cu *ID\_Zile\_lucrate* ca si cheie primară. Această entitate conține informații despre data zilei lucrate, ora de început și ora de final a turelor, ajutând la gestionarea programului de lucru al angajaților.

Produs: Administrează informațiile despre produsele disponibile, fiecare produs având *ID\_Produs* ca si cheie primară. Această entitate include detalii despre numele produsului, prețul și cantitatea disponibilă, facilitând gestionarea stocurilor și asigurarea disponibilității produselor.

1. **Descrierea relațiilor, incluzând precizarea cardinalității acestora.**

BAUTURI\_contin\_PRODUS = relație dintre entitățile BAUTURI și PRODUS (ce produse sunt incluse în conținutul comenzilor). O bautura poate contine mai multe produse și un produs poate fi continut de mai multe bauturi. Relația are cardinalitatea minimă 1 : 1 și cardinalitatea maximă m : n.

CLIENTI\_plaseaza\_COMENZI = relație dintre entitățile CLIENTI și COMENZI (ce comenzi sunt plasate de clienți). Un client poate plasa mai multe comenzi dar trebuie să plaseze cel puțin o comandă pentru a fi înregistrat. Relația are cardinalitatea minimă 1 : 1 și cardinalitatea maximă 1 : n.

ANGAJATI\_fac\_COMENZI = relație dintre entitățile ANGAJATI și COMENZI (ce comenzi sunt procesate de angajați). Un angajat poate face mai multe comenzi dar nu trebuie să faca cel puțin o comandă pentru a fi înregistrat. Relația are cardinalitatea minimă 1 : 0 și cardinalitatea maximă 1 : n.

COMENZI\_include\_BAUTURI = relație dintre entitățile COMENZI și BAUTURI (ce conținuturi de comenzi sunt asociate cu comenzi). O comandă poate include mai multe bauturi dar trebuie să includă cel puțin o bautura pentru a fi înregistrată. Relația are cardinalitatea minimă 1 : 1 și cardinalitatea maximă 1 : n.

CLIENTI\_efectueaza\_PLATI = relație dintre entitățile CLIENTI și PLATI (ce plăți sunt efectuate de clienți). Un client poate efectua mai multe plăți dar trebuie să efectueze cel puțin o plată pentru a fi înregistrat. Relația are cardinalitatea minimă 1 : 1 și cardinalitatea maximă 1 : n.

ANGAJATI\_au\_ZILE\_LUCRATE = relație dintre entitățile ANGAJATI și ZILE\_LUCRATE (ce zile lucrate sunt asociate cu angajați). Un angajat poate avea mai multe zile lucrate dar trebuie să aibă cel puțin o zi lucrată pentru a fi înregistrat. Relația are cardinalitatea minimă 1 : 1 și cardinalitatea maximă m : n.

1. **Descrierea atributelor, incluzând tipul de date și eventualele constrângeri, valori implicite, valori posibile ale atributelor.**

***Comenzi:***

ID\_Comanda: INT, cheie primară.

ID\_Bauturi: INT, referință la Bauturi (ID\_Bautura), NOT NULL.

ID\_Angajati: INT, referință la Angajati (ID\_Angajat), NOT NULL.

ID\_Clienti: INT, referință la Clienti (ID\_Clienti), NOT NULL.

***Bauturi:***

ID\_Bautura: INT, cheie primară.

nume: VARCHAR(255), UNIQUE.

tip: VARCHAR(255), NOT NULL.

pret: DECIMAL(10, 2), NOT NULL, valoare implicită 0.00, trebuie să fie pozitiv.

**Clienti:**

ID\_Clienti: INT, cheie primară.

nume: VARCHAR(100), NOT NULL.

email: VARCHAR(255), NOT NULL, trebuie să fie unic, format valid de email.

**Plati:**

ID\_plata: INT, cheie primară.

plata\_data: DATE, NOT NULL.

plata\_metoda: VARCHAR(50), NOT NULL, valori posibile: ('card', 'numerar', 'online').

Suma: DECIMAL(10,2),NOT NULL valoare implicită 0.00, trebuie să fie pozitiv.

**Angajati:**

ID\_Angajat: INT, cheie primară.

rol: VARCHAR(50), NOT NULL, valori posibile: ('barista', 'casier', 'manager').

nume: VARCHAR(255) , UNIQUE.

email: VARCHAR(255), NOT NULL, trebuie să fie unic, format valid de email.

telefon: VARCHAR(15),UNIQUE, format valid de telefon.

**Zile\_lucrate:**

ID\_Zile\_lucrate: INT, cheie primară.

ziua\_data: DATE, NOT NULL.

zi\_inceput: timestamp NOT NULL ,

zi\_final: timestamp NOT NULL ,Trebuie sa fie mai mare decat zi\_inceput

**Produse:**

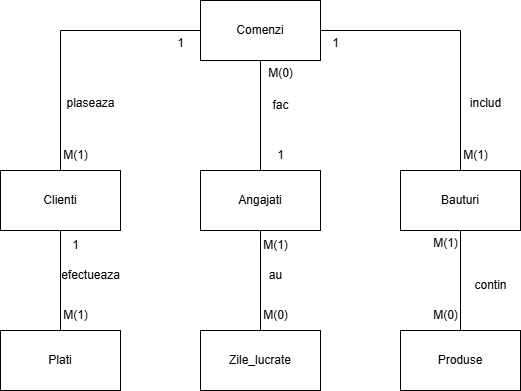
ID\_Produs: INT, cheie primară.

nume\_produs: VARCHAR(255), NOT NULL.

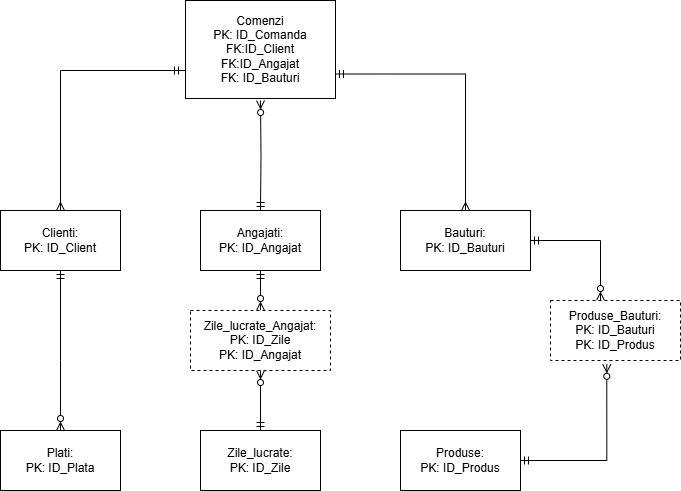
pret\_produs: DECIMAL(10, 2), NOT NULL, valoare implicită 0.00, trebuie să fie pozitiv.

cantitate: INT, NOT NULL, valoare implicită 0, trebuie să fie pozitiv.

1. **Realizarea diagramei entitate-relație corespunzătoare descrierii de la punctele 3-5.**



1. **Realizarea diagramei conceptuale corespunzătoare diagramei entitate-relație proiectate la punctul 6. Diagrama conceptuală obținută trebuie să conțină minimum 7 tabele (fără considerarea subentităților), dintre care cel puțin un tabel asociativ.**



1. **Enumerarea schemelor relaționale corespunzătoare diagramei conceptuale proiectate la punctul 7.**

***Comenzi:***

ID\_Comanda: INT, cheie primară.

ID\_Bauturi: INT, referință la Bauturi (ID\_Bautura), NOT NULL.

ID\_Angajati: INT, referință la Angajati (ID\_Angajat), NOT NULL.

ID\_Clienti: INT, referință la Clienti (ID\_Clienti), NOT NULL.

***Bauturi:***

ID\_Bautura: INT, cheie primară.

nume: VARCHAR(100), NOT NULL.

tip: VARCHAR(100), NOT NULL.

pret: DECIMAL(10, 2), NOT NULL, valoare implicită 0.00, trebuie să fie pozitiv.

**Clienti:**

ID\_Clienti: INT, cheie primară.

nume: VARCHAR(100), NOT NULL.

email: VARCHAR(100), NOT NULL, trebuie să fie unic, format valid de email.

**Plati:**

ID\_plata: INT, cheie primară.

plata\_data: DATE, NOT NULL.

plata\_metoda: VARCHAR(50), NOT NULL, valori posibile: ('card', 'numerar', 'online').

**Angajati:**

ID\_Angajat: INT, cheie primară.

rol: VARCHAR(50), NOT NULL, valori posibile: ('barista', 'casier', 'manager').

email: VARCHAR(100), NOT NULL, trebuie să fie unic, format valid de email.

telefon: VARCHAR(15), NOT NULL, format valid de telefon.

**Zile\_lucrate:**

ID\_Zile\_lucrate: INT, cheie primară.

ziua\_data: DATE, NOT NULL.

zi\_inceput timestamp NOT NULL ,

zi\_final timestamp NOT NULL ,trebuie sa fie mai mare decat zi\_inceput

**Produse:**

ID\_Produs: INT, cheie primară.

nume\_produs: VARCHAR(100), NOT NULL.

pret\_produs: DECIMAL(10, 2), NOT NULL, valoare implicită 0.00, trebuie să fie pozitiv.

cantitate: INT, NOT NULL, valoare implicită 0, trebuie să fie pozitiv.

**Produse\_Bauturi:**

ID\_Produs: INT.

ID\_Bauturi: INT.

Zile\_lucrate\_Angajat:

ID\_Angajat: INT.

ID\_Zile\_lucrate: INT.

1. **Realizarea normalizării până la forma normală 3 (FN1-FN3).**

**FN1:**

**In cadrul bazei mele de date , voi prezenta urmatorul exemplu: bautura cu identificatorul 3 este facuta din produsele 1,2 si 3.**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ID\_Bautura | ID\_Produs | Nume\_produs | Pret\_produs | cantitate |
| 1 | 1,3 | zahar | 12.99 | 200 |
| 2 | 4,5 | cafea | 20.99 | 100 |
| 3 | 1,2,3 | vanilie | 32.99 | 100 |

**Pentru a rezolva aceasta problema vom crea tabelul asociativ Produse\_Bauturi care respete restrictiile de unicitate si aduce baza de date in FN1:**

|  |  |
| --- | --- |
| **ID\_Bautura** | **ID\_Produs** |
| **1** | **1** |
| **1** | **3** |
| **2** | **4** |
| **2** | **5** |
| **3** | **1** |
| **3** | **2** |
| **3** | **3** |

**FN2:**

**In cazul bazei mele de date voi utiliza urmatorul exemplu: tabelele Angajati si Zile\_lucrate. Pentru a vedea in ce zile au lucrat mai multi angajati ar trebui sa exista o relatie cu cardinalitate m : n .**

|  |  |  |  |
| --- | --- | --- | --- |
| **ID\_Zile\_lucrate** | **Ziua\_data** | **ID\_Angajat** | **Rol** |
| **1** | **13/02/2024** | **1** | **Barista** |
| **1** | **26/01/2024** | **4** | **Barista** |
| **2** | **14/05/2024** | **5** | **Manager** |
| **2** | **06/06/2023** | **5** | **Casier** |
| **3** | **18/02/2023** | **6** | **Casier** |
| **3** | **10/01/2023** | **6** | **Barista** |
| **3** | **04/04/2024** | **6** | **Barista** |

**Pentru a se afla in FN2 , tabelul trebuie sa fie in FN1(toate atributele sunt atomice), dar mai mult, pentru tabelele cu cheie compusa , fiecare atribut non-cheie trebuie sa fie dependent de intreaga cheie compusa, nu doar de o parte a acesteia.In exemplul de mai jos se poate observa ca numele entitatilor Zile\_lucrate si Angajati este dependent de propriul identificator:**

|  |  |
| --- | --- |
| **ID\_Zile\_lucrate** | **ID\_Angajat** |
| **1** | **1** |
| **1** | **4** |
| **2** | **5** |
| **2** | **5** |
| **3** | **6** |
| **3** | **6** |
| **3** | **6** |

**FN3:**

**In cazul bazei mele de date voi utiliza un exemplu folosind tabela Clienti.Aceasta contine atributele : ID\_Client,nume si email.In acest moment, tabela se afla atat in FN1 , cat si in FN2. Atributele nume si emal depind direct de cheia primara ID\_Client, dar atributele plata\_data si plata\_metoda nu depind in mod direct de cheia primara.**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **ID\_Client** | **nume** | **email** | **Plata\_data** | **Plata\_metoda** |
| **1** | **Marian** | **marian@gmail.com** | **13/03/2024** | **card** |
| **2** | **Maria** | **maria@gmail.com** | **13/03/2024** | **card** |
| **3** | **Andrei** | **andrei@gmail.com** | **14/01/2024** | **cash** |
| **4** | **Ion** | **ion@gmail.com** | **03/03/2024** | **cash** |

**Pentru a aduce acest tabel la FN3 vom separa atributele legate de plati de tabel Clienti si vom crea tabela Plati. Tabel Clienti va primi un foreign\_key ID\_plata care va face legatura cu atributele platilor.**

**Clienti:**

|  |  |  |  |
| --- | --- | --- | --- |
| **ID\_Client** | **nume** | **email** | **ID\_Plata** |
| **1** | **Marian** | **marian@gmail.com** | **1** |
| **2** | **Maria** | **maria@gmail.com** | **1** |
| **3** | **Andrei** | **andrei@gmail.com** | **2** |
| **4** | **Ion** | **ion@gmail.com** | **3** |

**Plati:**

|  |  |  |
| --- | --- | --- |
| **Id\_Plata** | **Plata\_data** | **Plata\_metoda** |
| **1** | **13/03/2024** | **card** |
| **2** | **14/01/2024** | **cash** |
| **3** | **03/03/2024** | **card** |

1. **Crearea unei secvențe ce va fi utilizată în inserarea înregistrărilor în tabele (punctul 11).**

CREATE SEQUENCE PRODUSE\_SEQ START WITH 1;

CREATE SEQUENCE BAUTURI\_SEQ START WITH 1;

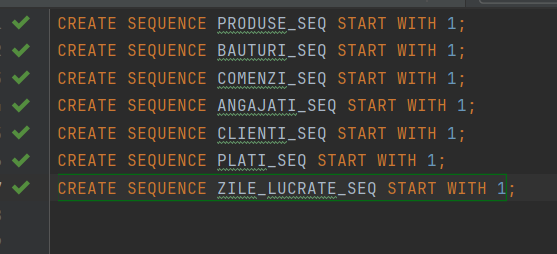
CREATE SEQUENCE COMENZI\_SEQ START WITH 1;

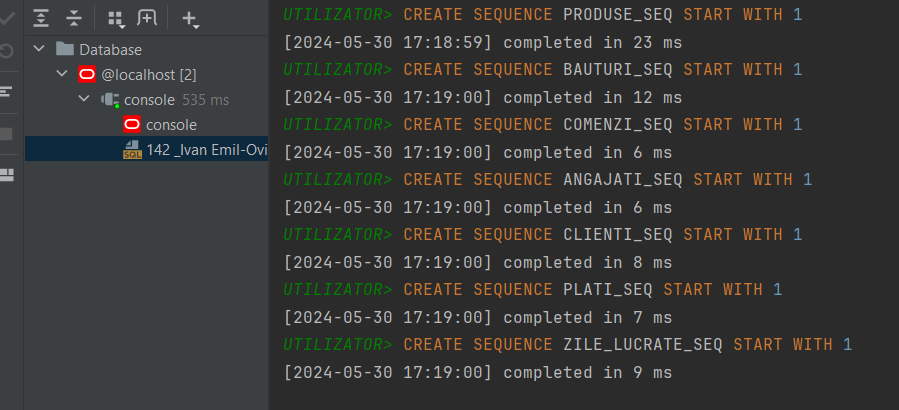
CREATE SEQUENCE ANGAJATI\_SEQ START WITH 1;

CREATE SEQUENCE CLIENTI\_SEQ START WITH 1;

CREATE SEQUENCE PLATI\_SEQ START WITH 1;

CREATE SEQUENCE ZILE\_LUCRATE\_SEQ START WITH 1;



****

1. **Crearea tabelelor în SQL și inserarea de date coerente în fiecare dintre acestea (minimum 5 înregistrări în fiecare tabel neasociativ; minimum 10 înregistrări în tabelele asociative; maxim 30 de înregistrări în fiecare tabel).**

CREATE TABLE Bauturi (

ID\_Bautura INT PRIMARY KEY,

nume VARCHAR(255) UNIQUE,

tip VARCHAR(255) NOT NULL,

pret DECIMAL(10, 2) NOT NULL

);

CREATE TABLE Plati (

ID\_Plata INT PRIMARY KEY,

data DATE NOT NULL,

metoda VARCHAR(255) NOT NULL,

suma DECIMAL(10, 2) NOT NULL

);

CREATE TABLE Clienti (

ID\_Clienti INT PRIMARY KEY,

nume VARCHAR(255) UNIQUE,

email VARCHAR(255) UNIQUE ,

ID\_Plata INT NOT NULL ,

FOREIGN KEY (ID\_Plata) REFERENCES Plati(ID\_Plata)

);

CREATE TABLE Angajati (

ID\_Angajat INT PRIMARY KEY,

rol VARCHAR(255) NOT NULL,

nume VARCHAR(255) UNIQUE,

email VARCHAR(255) UNIQUE ,

telefon VARCHAR(255) UNIQUE

);

CREATE TABLE Zile\_lucrate (

ID\_Zile\_lucrate INT PRIMARY KEY,

ziua\_data DATE NOT NULL,

zi\_inceput VARCHAR(255) NOT NULL,

zi\_final VARCHAR(255) NOT NULL ,

ID\_Angajat INT,

FOREIGN KEY (ID\_Angajat) REFERENCES Angajati(ID\_Angajat)

);

CREATE TABLE Produs (

ID\_Produs INT PRIMARY KEY,

nume VARCHAR(255)UNIQUE ,

pret DECIMAL(10, 2)NOT NULL,

cantitate INT NOT NULL,

ID\_Bautura INT,

FOREIGN KEY (ID\_Bautura) REFERENCES bAUTURI(ID\_Bautura)

);

CREATE TABLE Comenzi (

ID\_Comanda INT PRIMARY KEY,

ID\_Bauturi INT,

ID\_Angajati INT,

ID\_Clienti INT,

FOREIGN KEY (ID\_Bauturi) REFERENCES Bauturi(ID\_Bautura),

FOREIGN KEY (ID\_Angajati) REFERENCES Angajati(ID\_Angajat),

FOREIGN KEY (ID\_Clienti) REFERENCES Clienti(ID\_Clienti)

);

CREATE TABLE Bauturi\_Produse (

ID\_Bautura\_Produs INT PRIMARY KEY,

ID\_Bautura INT,

ID\_Produs INT,

FOREIGN KEY (ID\_Bautura) REFERENCES Bauturi(ID\_Bautura),

FOREIGN KEY (ID\_Produs) REFERENCES Produs(ID\_Produs)

);

CREATE TABLE Angajati\_Zile\_lucrate (

ID\_Angajati\_Zile\_lucrate INT PRIMARY KEY,

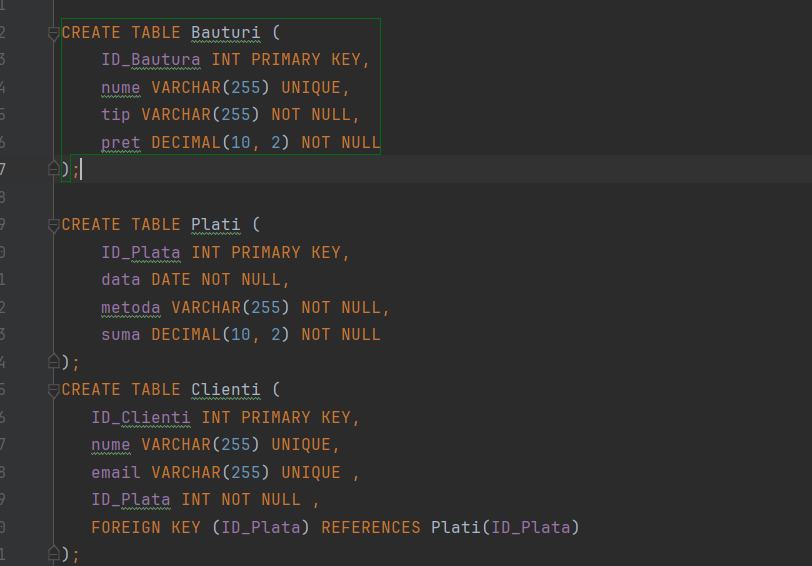
ID\_Angajat INT,

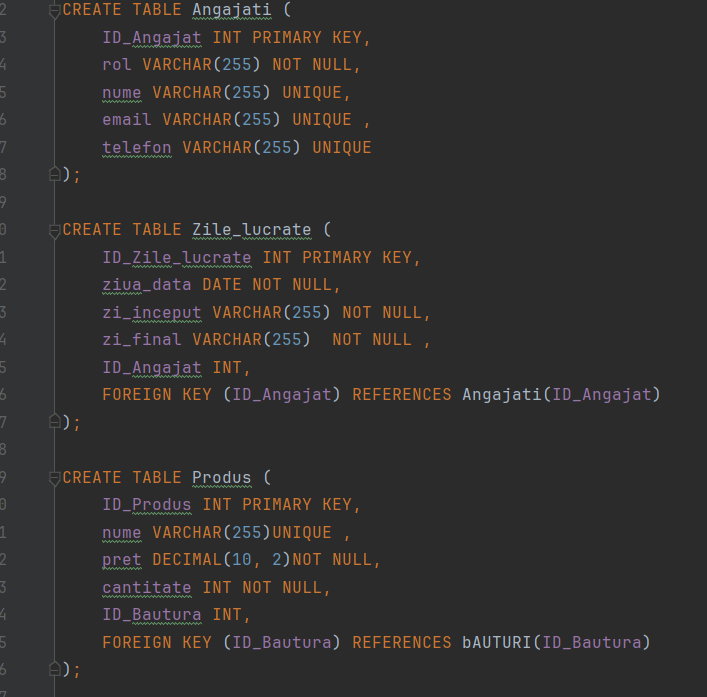
ID\_Zile\_lucrate INT,

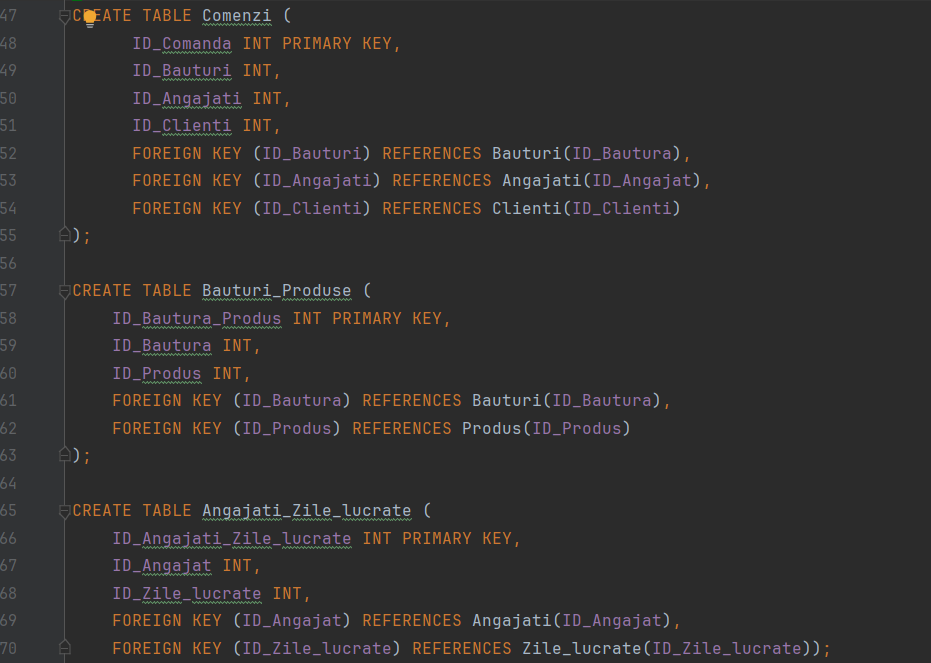
FOREIGN KEY (ID\_Angajat) REFERENCES Angajati(ID\_Angajat),

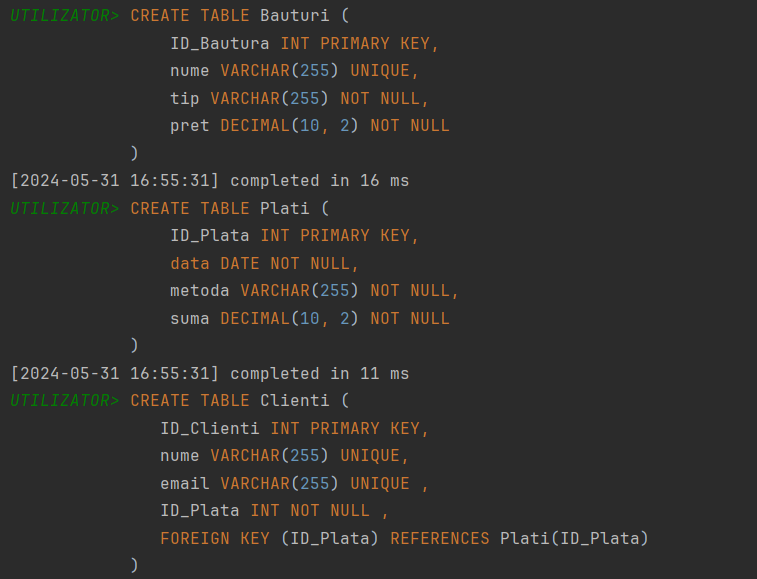
FOREIGN KEY (ID\_Zile\_lucrate) REFERENCES Zile\_lucrate(ID\_Zile\_lucrate)

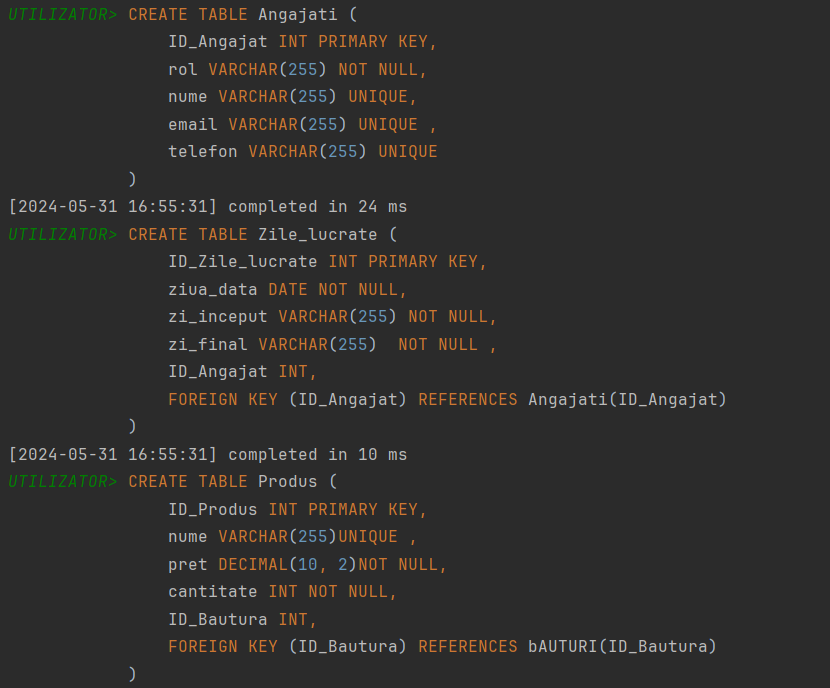
);

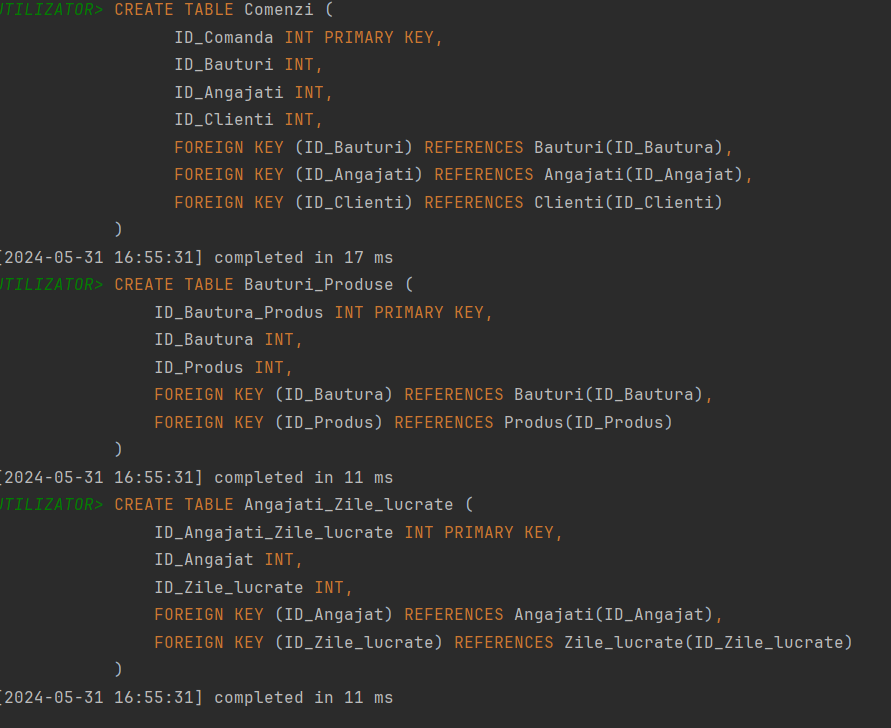


****

****

****

****

****

**Inserari:**

--Inserare in tabelul "Bauturi":

INSERT INTO Bauturi (ID\_Bautura, nume,tip, pret) VALUES (1, 'Caffe Latte', 'Cafea', 25.99);

INSERT INTO Bauturi (ID\_Bautura, nume,tip, pret) VALUES (2, 'Cappuccino', 'Cafea', 20.99);

INSERT INTO Bauturi (ID\_Bautura, nume,tip, pret) VALUES (3, 'Espresso', 'Cafea', 15.99);

INSERT INTO Bauturi (ID\_Bautura, nume,tip, pret) VALUES (4, 'Americano', 'Cafea', 18.99);

INSERT INTO Bauturi (ID\_Bautura, nume,tip, pret) VALUES (5, 'Macchiato', 'Cafea', 22.99);

INSERT INTO Bauturi (ID\_Bautura, nume,tip, pret) VALUES (6, 'Ceai verde', 'Ceai', 30.99);

INSERT INTO Bauturi (ID\_Bautura, nume,tip, pret) VALUES (7, 'Ceai negru', 'Ceai', 25.99);

INSERT INTO Bauturi (ID\_Bautura, nume,tip, pret) VALUES (8, 'Ceai de fructe', 'Ceai', 35.99);

--Inserare in tabelul "Plati":

INSERT INTO Plati (ID\_Plata,data, metoda, suma) VALUES (1,to\_date('2024-03-14','YYYY-MM-DD'), 'Card',57.99);

INSERT INTO Plati (ID\_Plata,data, metoda, suma) VALUES (2,to\_date('2024-03-14','YYYY-MM-DD') , 'Cash', 45.99);

INSERT INTO Plati (ID\_Plata,data, metoda, suma) VALUES (3,to\_date('2023-04-23','YYYY-MM-DD') , 'PayPal', 35.99);

INSERT INTO Plati (ID\_Plata,data, metoda, suma) VALUES (4, to\_date('2024-02-01','YYYY-MM-DD') , 'Transfer bancar', 38.99);

INSERT INTO Plati (ID\_Plata,data, metoda, suma) VALUES (5, to\_date('2024-01-16','YYYY-MM-DD'), 'Card', 42.99);

--Inserare in tabelul "Clienti":

INSERT INTO Clienti (ID\_Clienti, nume, email, ID\_Plata) VALUES (1, 'Ion Popescu', 'ion.popescu@example.com', 1);

INSERT INTO Clienti (ID\_Clienti, nume, email, ID\_Plata) VALUES (2, 'Maria Ionescu', 'maria.ionescu@example.com', 2);

INSERT INTO Clienti (ID\_Clienti, nume, email, ID\_Plata) VALUES (3, 'Vasile Georgescu', 'vasile.georgescu@example.com', 3);

INSERT INTO Clienti (ID\_Clienti, nume, email, ID\_Plata) VALUES (4, 'Ana Marinescu', 'ana.marinescu@example.com', 4);

INSERT INTO Clienti (ID\_Clienti, nume, email, ID\_Plata) VALUES (5, 'Gheorghe Popa', 'gheorghe.popa@example.com', 5);

--Inserare in tabelul "Angajati":

INSERT INTO Angajati (ID\_Angajat, rol, nume, email, telefon) VALUES (1, 'Brista', 'Maria Vasilescu', 'maria1@example.com', '0712345678');

INSERT INTO Angajati (ID\_Angajat, rol, nume, email, telefon) VALUES (2, 'Barista', 'Ana Benim', 'ana2@example.com', '0723456789');

INSERT INTO Angajati (ID\_Angajat, rol, nume, email, telefon) VALUES (3, 'Barista', 'Matei Ivan', 'matei3@example.com', '0734567890');

INSERT INTO Angajati (ID\_Angajat, rol, nume, email, telefon) VALUES (4, 'Manager', 'Darius Hornet', 'darius4@example.com', '0745678901');

INSERT INTO Angajati (ID\_Angajat, rol, nume, email, telefon) VALUES (5, 'Casier', 'Irina Dumitru', 'irina5@example.com', '0756789012');

INSERT INTO Angajati (ID\_Angajat, rol, nume, email, telefon) VALUES (6, 'Casier', 'Stefan Curechiriu', 'stefan6@example.com', '0767890123');

--Inserare in tabelul "Zile\_lucrate":

INSERT INTO Zile\_lucrate (ID\_Zile\_lucrate, ziua\_data, zi\_inceput, zi\_final, ID\_Angajat) VALUES (1,to\_date('2024-01-01','YYYY-MM-DD') , '09:00', '17:00', 1);

INSERT INTO Zile\_lucrate (ID\_Zile\_lucrate, ziua\_data, zi\_inceput, zi\_final, ID\_Angajat) VALUES (2, to\_date('2024-01-02','YYYY-MM-DD'), '09:00', '17:00', 2);

INSERT INTO Zile\_lucrate (ID\_Zile\_lucrate, ziua\_data, zi\_inceput, zi\_final, ID\_Angajat) VALUES (3, to\_date('2024-01-03','YYYY-MM-DD'), '09:00', '17:00', 3);

INSERT INTO Zile\_lucrate (ID\_Zile\_lucrate, ziua\_data, zi\_inceput, zi\_final, ID\_Angajat) VALUES (4, to\_date('2024-01-04','YYYY-MM-DD'), '09:00', '17:00', 4);

INSERT INTO Zile\_lucrate (ID\_Zile\_lucrate, ziua\_data, zi\_inceput, zi\_final, ID\_Angajat) VALUES (5, to\_date('2024-01-05','YYYY-MM-DD'), '09:00', '17:00', 5);

--Inserare in tabelul "Produs":

INSERT INTO Produs (ID\_Produs, nume, pret, cantitate, ID\_Bautura) VALUES (1, 'cafea', 10.00, 50, 1);

INSERT INTO Produs (ID\_Produs, nume, pret, cantitate, ID\_Bautura) VALUES (2, 'ceai verde', 8.00, 30, 2);

INSERT INTO Produs (ID\_Produs, nume, pret, cantitate, ID\_Bautura) VALUES (3, 'vanilie', 12.00, 20, 3);

INSERT INTO Produs (ID\_Produs, nume, pret, cantitate, ID\_Bautura) VALUES (4, 'caramel', 5.00, 100, 4);

INSERT INTO Produs (ID\_Produs, nume, pret, cantitate, ID\_Bautura) VALUES (5, 'lapte', 15.00, 40, 5);

INSERT INTO Produs (ID\_Produs, nume, pret, cantitate, ID\_Bautura) VALUES (6, 'zahar', 3.00, 200, 6);

--Inserare in tabelul "Comenzi":

INSERT INTO Comenzi (ID\_Comanda, ID\_Bauturi, ID\_Angajati, ID\_Clienti) VALUES (1, 1, 1, 1);

INSERT INTO Comenzi (ID\_Comanda, ID\_Bauturi, ID\_Angajati, ID\_Clienti) VALUES (2, 2, 2, 2);

INSERT INTO Comenzi (ID\_Comanda, ID\_Bauturi, ID\_Angajati, ID\_Clienti) VALUES (3, 3, 3, 3);

INSERT INTO Comenzi (ID\_Comanda, ID\_Bauturi, ID\_Angajati, ID\_Clienti) VALUES (4, 4, 4, 4);

INSERT INTO Comenzi (ID\_Comanda, ID\_Bauturi, ID\_Angajati, ID\_Clienti) VALUES (5, 5, 5, 5);

--Inserare in tabelul "Bauturi\_Produse":

INSERT INTO Bauturi\_Produse (ID\_Bautura\_Produs, ID\_Bautura, ID\_Produs) VALUES (1, 1, 1);

INSERT INTO Bauturi\_Produse (ID\_Bautura\_Produs, ID\_Bautura, ID\_Produs) VALUES (2, 2, 2);

INSERT INTO Bauturi\_Produse (ID\_Bautura\_Produs, ID\_Bautura, ID\_Produs) VALUES (3, 3, 3);

INSERT INTO Bauturi\_Produse (ID\_Bautura\_Produs, ID\_Bautura, ID\_Produs) VALUES (4, 4, 4);

INSERT INTO Bauturi\_Produse (ID\_Bautura\_Produs, ID\_Bautura, ID\_Produs) VALUES (5, 5, 5);

INSERT INTO Bauturi\_Produse (ID\_Bautura\_Produs, ID\_Bautura, ID\_Produs) VALUES (6, 1, 2);

INSERT INTO Bauturi\_Produse (ID\_Bautura\_Produs, ID\_Bautura, ID\_Produs) VALUES (7, 2, 3);

INSERT INTO Bauturi\_Produse (ID\_Bautura\_Produs, ID\_Bautura, ID\_Produs) VALUES (8, 3, 4);

INSERT INTO Bauturi\_Produse (ID\_Bautura\_Produs, ID\_Bautura, ID\_Produs) VALUES (9, 4, 5);

INSERT INTO Bauturi\_Produse (ID\_Bautura\_Produs, ID\_Bautura, ID\_Produs) VALUES (10, 5, 1);

--inserare in tabelul "Angajati\_Zile\_lucrate":

INSERT INTO Angajati\_Zile\_lucrate (ID\_Angajati\_Zile\_lucrate, ID\_Angajat, ID\_Zile\_lucrate)

VALUES (1, 1, 1);

INSERT INTO Angajati\_Zile\_lucrate (ID\_Angajati\_Zile\_lucrate, ID\_Angajat, ID\_Zile\_lucrate)

VALUES (2, 2, 2);

INSERT INTO Angajati\_Zile\_lucrate (ID\_Angajati\_Zile\_lucrate, ID\_Angajat, ID\_Zile\_lucrate)

VALUES (3, 3, 3);

INSERT INTO Angajati\_Zile\_lucrate (ID\_Angajati\_Zile\_lucrate, ID\_Angajat, ID\_Zile\_lucrate)

VALUES (4, 4, 4);

INSERT INTO Angajati\_Zile\_lucrate (ID\_Angajati\_Zile\_lucrate, ID\_Angajat, ID\_Zile\_lucrate)

VALUES (5, 5, 5);

INSERT INTO Angajati\_Zile\_lucrate (ID\_Angajati\_Zile\_lucrate, ID\_Angajat, ID\_Zile\_lucrate)

VALUES (6, 1, 2);

INSERT INTO Angajati\_Zile\_lucrate (ID\_Angajati\_Zile\_lucrate, ID\_Angajat, ID\_Zile\_lucrate)

VALUES (7, 2, 3);

INSERT INTO Angajati\_Zile\_lucrate (ID\_Angajati\_Zile\_lucrate, ID\_Angajat, ID\_Zile\_lucrate)

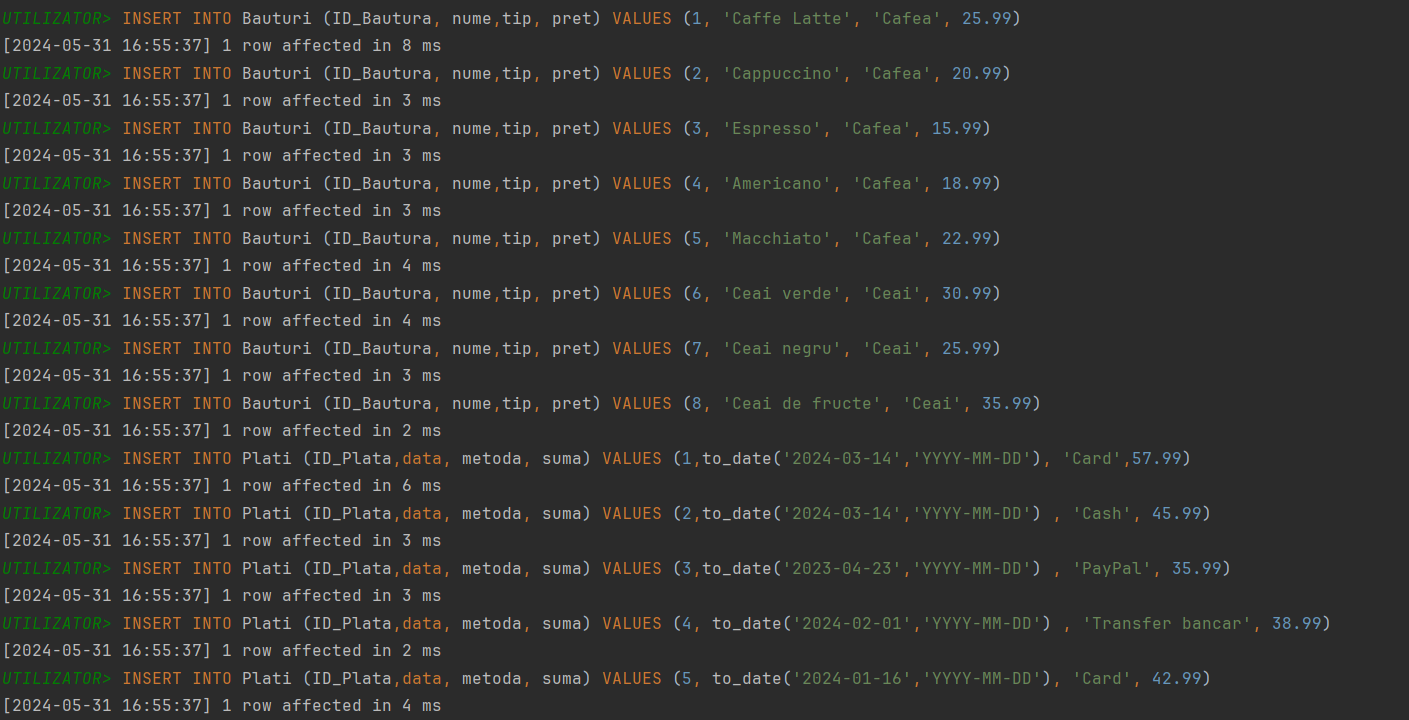
VALUES (8, 3, 4);

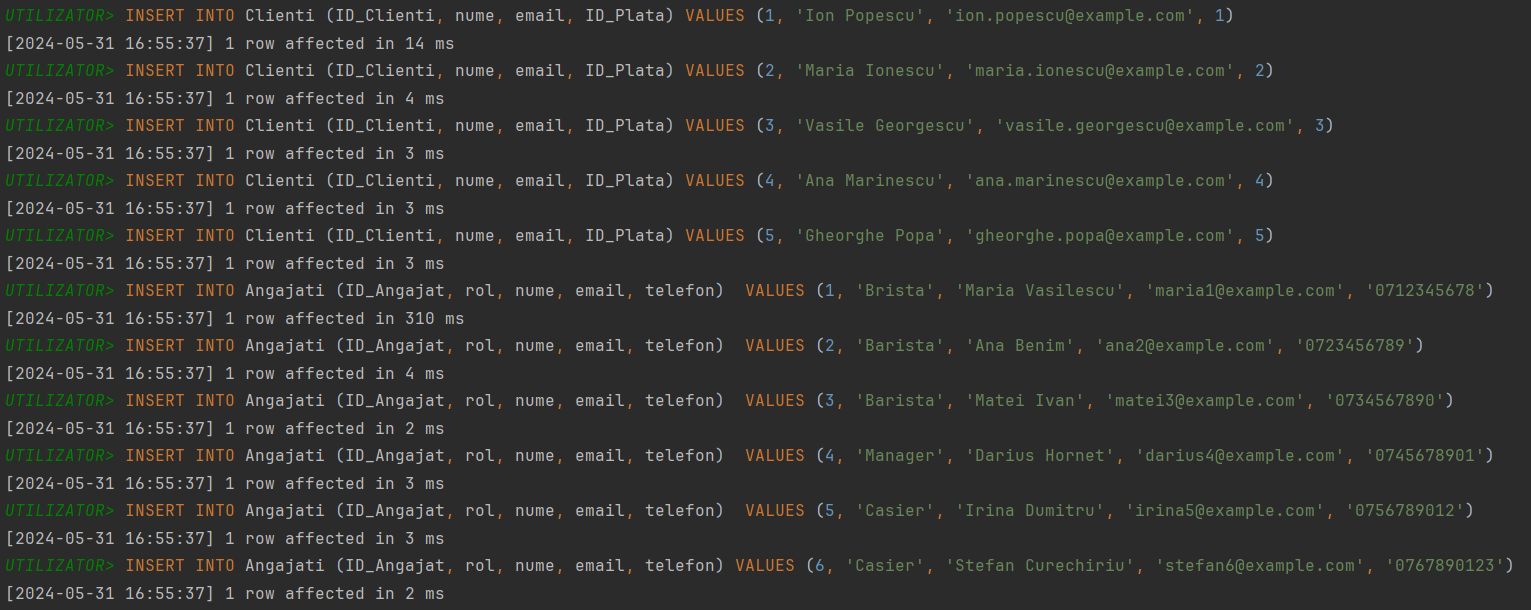
INSERT INTO Angajati\_Zile\_lucrate (ID\_Angajati\_Zile\_lucrate, ID\_Angajat, ID\_Zile\_lucrate)

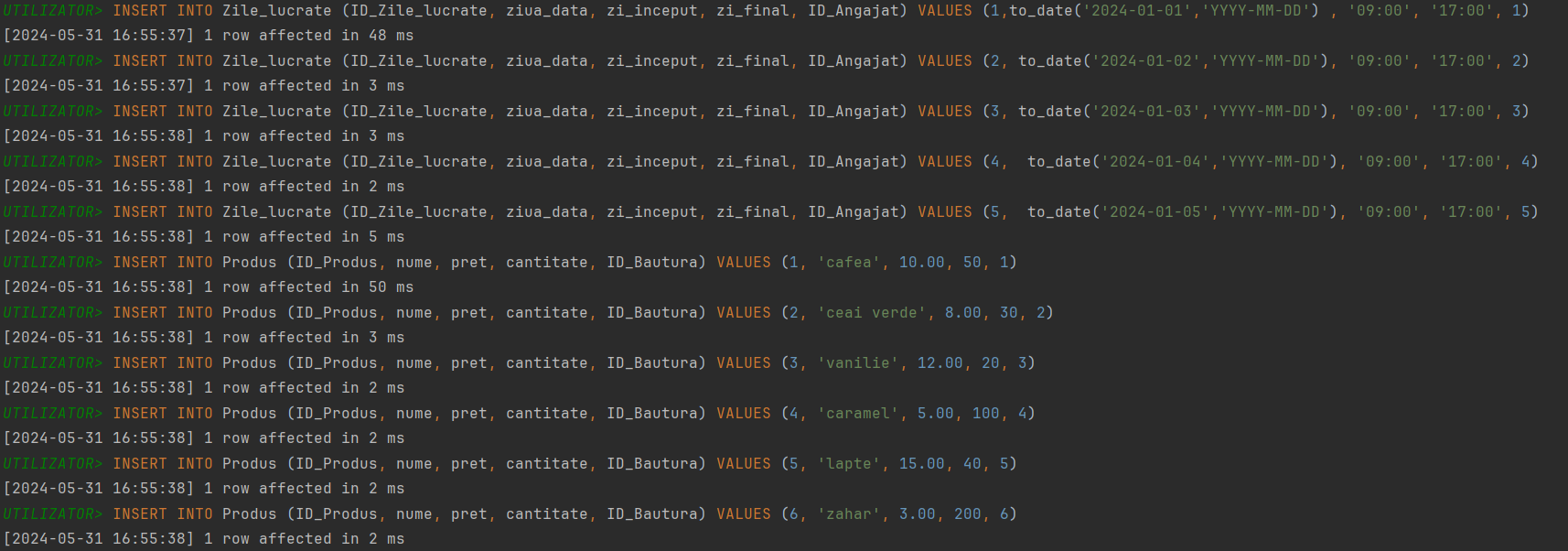
VALUES (9, 4, 5);

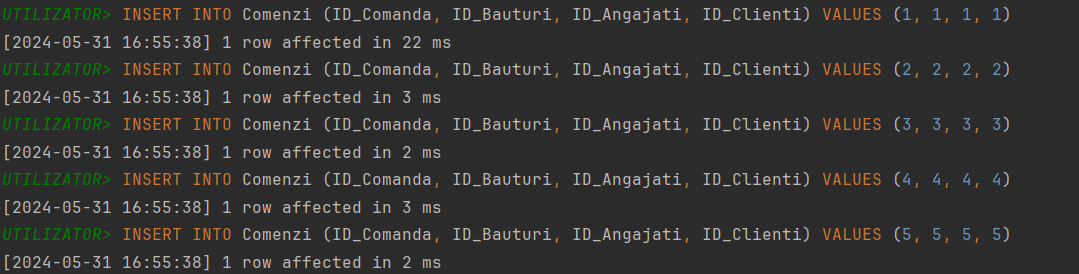
INSERT INTO Angajati\_Zile\_lucrate (ID\_Angajati\_Zile\_lucrate, ID\_Angajat, ID\_Zile\_lucrate)

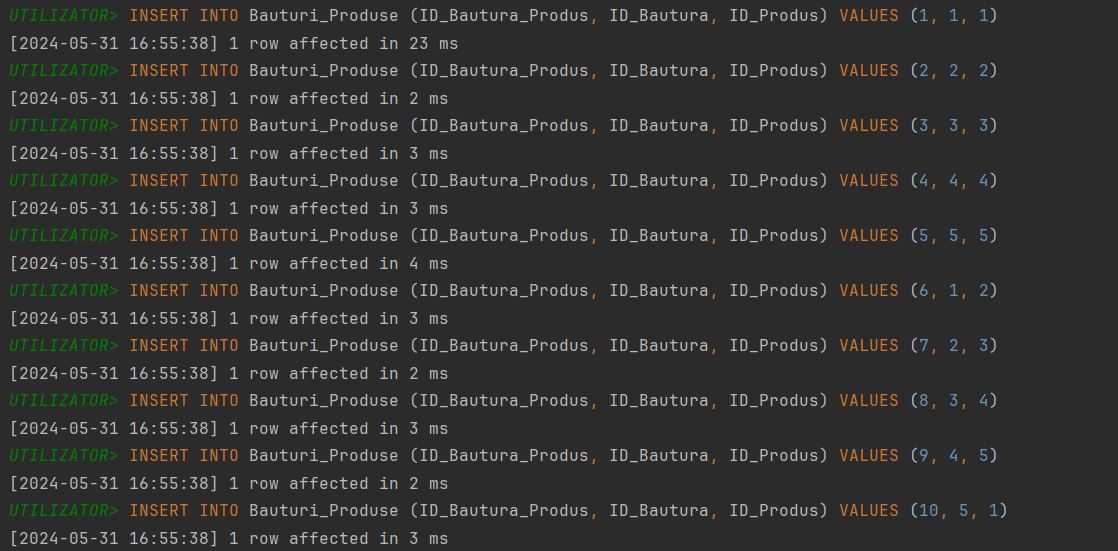
VALUES (10, 5, 1);

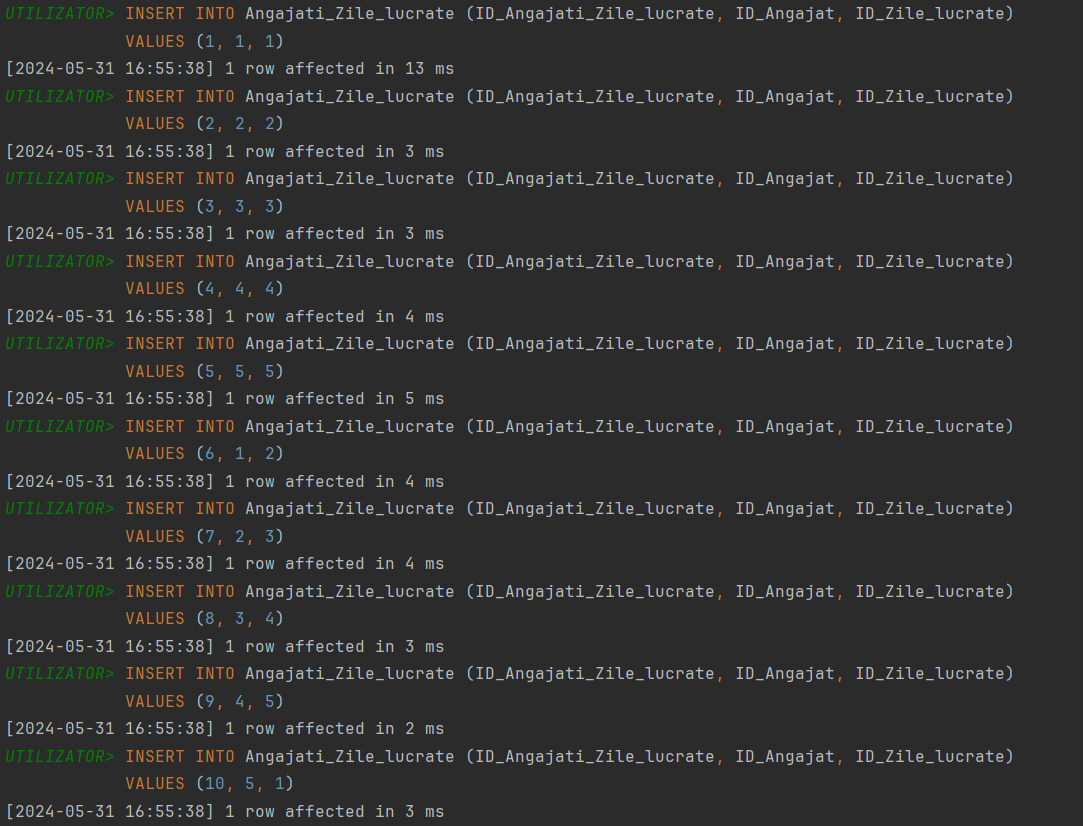
****

****

****

****

****

****

1. **Formulați în limbaj natural și implementați 5 cereri SQL complexe ce vor utiliza, în ansamblul lor, următoarele elemente:**

1.Afisati numele clienților, numele angajaților și suma totală cheltuită de fiecare client, în cazul în care suma totală cheltuită de client depășește media sumelor plătite. Rezultatele trebuie grupate în funcție de numele clienților și numele angajaților și sortate în ordine descrescătoare după suma totală cheltuită.

**SELECT**

**c.nume AS NUME\_Client,**

**a.nume AS NUME\_ANGAJAT,**

**SUM(p.suma) AS Total\_Cheltuit**

**FROM**

**Clienti c**

**JOIN**

**Comenzi co ON c.ID\_Clienti = co.ID\_Client**

**JOIN**

**Angajati a ON co.ID\_Angajat = a.ID\_Angajat**

**JOIN**

**Plati p ON c.ID\_Plata = p.ID\_Plata**

**GROUP BY**

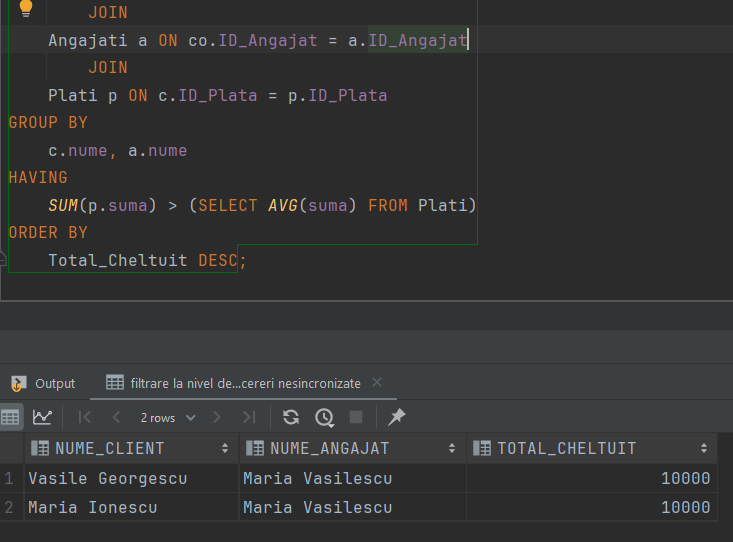
**c.nume, a.nume**

**HAVING**

**SUM(p.suma) > (SELECT AVG(suma) FROM Plati)**

**ORDER BY**

**Total\_Cheltuit DESC;**

****

**2.** Afișează numele băuturilor și numărul total de comenzi pentru fiecare băutură. Rezultatele trebuie să fie ordonate în ordine descrescătoare după numărul total de comenzi.

**SELECT**

**b.nume AS bautura\_name,**

**COUNT(co.ID\_Comanda) AS total\_orders**

**FROM**

**(SELECT ID\_Bautura, nume FROM Bauturi) b**

**LEFT JOIN**

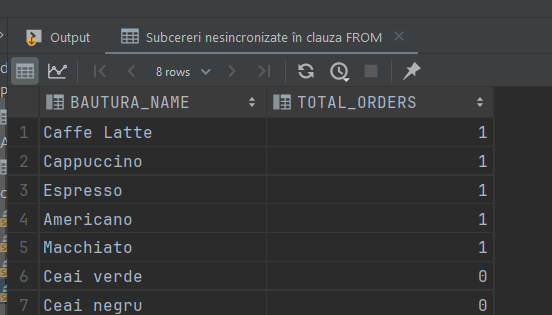
**Comenzi co ON b.ID\_Bautura = co.ID\_Bautura**

**GROUP BY**

**b.nume**

**ORDER BY**

**total\_orders DESC;**

****

**3.** Calculează numărul total de clienți pentru fiecare rol de angajat și afișează numai acele roluri pentru care numărul total de clienți este mai mare decât media numărului total de clienți pentru toate rolurile de angajat.

**SELECT a.rol, COUNT(DISTINCT c.ID\_Clienti) AS Total\_Clienti**

**FROM Angajati a**

**JOIN Comenzi co ON a.ID\_Angajat = co.ID\_Angajat**

**JOIN Clienti c ON co.ID\_Client = c.ID\_Clienti**

**GROUP BY a.rol**

**HAVING COUNT(DISTINCT c.ID\_Clienti) > (**

**SELECT AVG(Clienti\_Numar)**

**FROM (**

**SELECT COUNT(DISTINCT c1.ID\_Clienti) AS Clienti\_Numar**

**FROM Angajati a1**

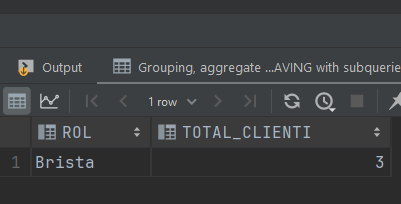
**JOIN Comenzi co1 ON a1.ID\_Angajat = co1.ID\_Angajat**

**JOIN Clienti c1 ON co1.ID\_Client = c1.ID\_Clienti**

**GROUP BY a1.rol**

**)**

**);**

****

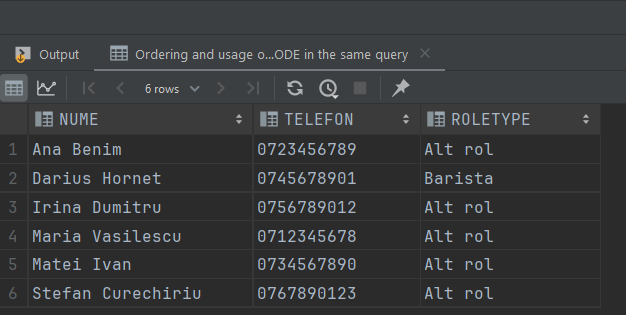
**4.** Afișează numele angajaților împreună cu numărul de telefon, sau un mesaj corespunzător dacă angajatul nu are număr de telefon. De asemenea, atribuie fiecărui angajat un tip de rol, indicând 'Barista' pentru rolurile 'Manager', în timp ce pentru alte roluri se indică 'Alt rol'. Rezultatele sunt sortate alfabetic după numele angajaților.

SELECT a.nume, NVL(a.telefon, 'Fara numar de telefon') AS Telefon,

DECODE(a.rol, 'Manager', 'Barista', 'Alt rol') AS RoleType

FROM Angajati a

ORDER BY a.nume;



**5.** Afișează detalii despre angajați, incluzând numele, emailul (în litere mari), o formă prescurtată a numelui, ani de experiență în domeniu, data ultimei lucrări, și un indicator al volumului de comenzi, împărțit în 'Mare', 'Mediu' sau 'Mic', bazat pe numărul total de comenzi. Rezultatele sunt grupate după angajați și sortate.

**WITH TotalComenzi AS (**

**SELECT a.ID\_Angajat, COUNT(co.ID\_Comanda) AS Total\_Comenzi**

**FROM Angajati a**

**JOIN Comenzi co ON a.ID\_Angajat = co.ID\_Angajat**

**GROUP BY a.ID\_Angajat**

**)**

**SELECT a.nume,**

**UPPER(a.email) AS EMAIL,**

**SUBSTR(a.nume, 1, 3) AS NUMEPRESCURTAT,**

**EXTRACT(YEAR FROM SYSDATE) - EXTRACT(YEAR FROM MIN(zl.ziua\_data)) AS ANILUCARTI,**

**TO\_CHAR(MAX(zl.ziua\_data), 'YYYY-MM-DD') AS ULTIMAZILUCRATA,**

**CASE**

**WHEN tc.Total\_Comenzi > 100 THEN 'Mare'**

**WHEN tc.Total\_Comenzi BETWEEN 50 AND 100 THEN 'Mediu'**

**ELSE 'Mic'**

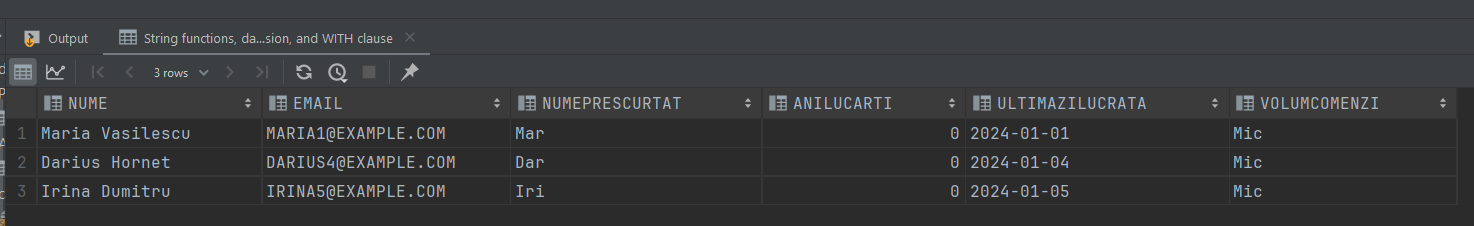
**END AS VOLUMCOMENZI**

**FROM Angajati a**

**JOIN Zile\_lucrate zl ON a.ID\_Angajat = zl.ID\_Angajat**

**JOIN TotalComenzi tc ON a.ID\_Angajat = tc.ID\_Angajat**

**GROUP BY a.ID\_Angajat, a.nume, a.email, tc.Total\_Comenzi;**

****

1. **Implementarea a 3 operații de actualizare și de suprimare a datelor utilizând subcereri.**

**Șterge toate comenzile pentru clienții care au plătit în numerar și au o sumă mai mare de 100 de unități, folosind o subinterogare pentru a identifica clienții și plățile lor în numerar cu sume mai mari de 100 de lei**

**DELETE FROM Comenzi**

**WHERE ID\_Client IN (**

**SELECT ID\_Clienti**

**FROM Clienti**

**WHERE ID\_Plata IN (**

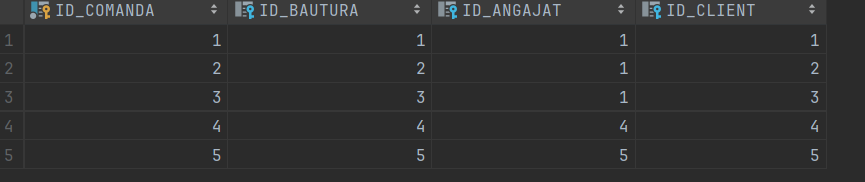
**SELECT ID\_Plata**

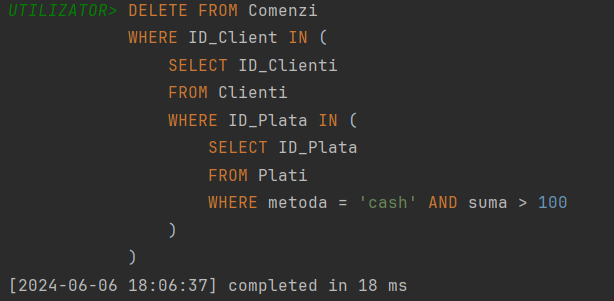
**FROM Plati**

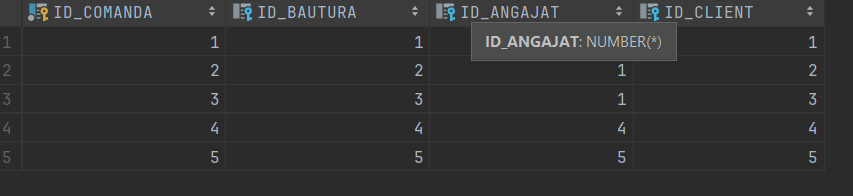
**WHERE metoda = 'cash' AND suma > 100**

**)**

**);**

****

****

****

Actualizează numele produselor care sunt in băuturi populare în comenzile clienților care au efectuat plăți cu sume mai mari de 1000 de unități, adăugând "Popular - " în fața numelui fiecărui produs, utilizând o subinterogare pentru a identifica comenzile, clienții și plățile respective.

**UPDATE Produs**

**SET nume = CONCAT('Popular - ', nume)**

**WHERE ID\_Produs IN (**

**SELECT ID\_Produs**

**FROM Produs**

**WHERE ID\_Bautura IN (**

**SELECT ID\_Bautura**

**FROM Comenzi**

**WHERE ID\_Client IN (**

**SELECT ID\_Clienti**

**FROM Clienti**

**WHERE ID\_Plata IN (**

**SELECT ID\_Plata**

**FROM Plati**

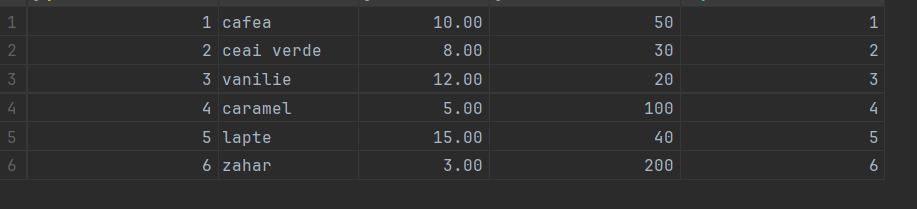
**WHERE suma > 1000**

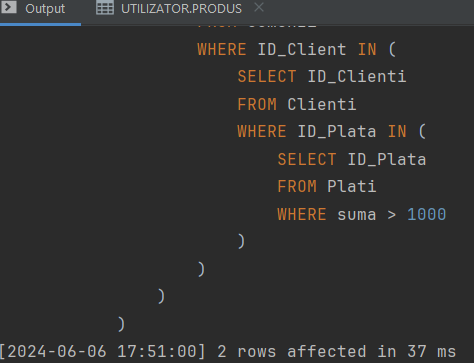
**)**

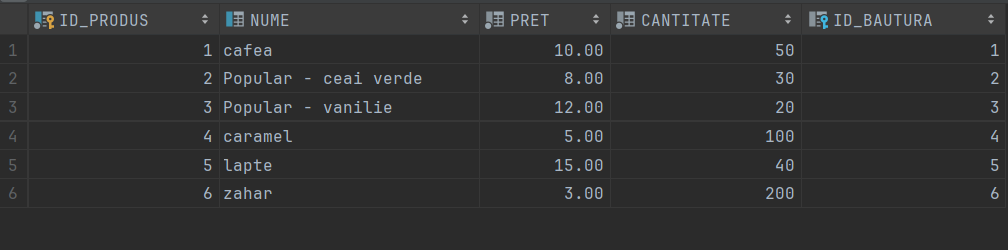
**)**

**)**

**);**

****

****

****

Dacă un produs are mai mult de un client care a comandat acest produs, cantitatea este dublată. Dacă un produs are exact trei clienți care l-au comandat, prețul este redus cu 20%. Acest lucru se aplică doar produselor care au mai mult de un client sau exact trei clienți care le-au comandat.

**UPDATE Produs**

**SET**

**cantitate =**

**CASE**

**WHEN ID\_Produs IN (**

**SELECT ID\_Produs**

**FROM (**

**SELECT ID\_Produs, COUNT(DISTINCT ID\_Client) AS nr\_clienti**

**FROM Comenzi**

**GROUP BY ID\_Produs**

**HAVING COUNT(DISTINCT ID\_Client) > 1**

**) Produse\_Mai\_Multe\_Clienti**

**) THEN cantitate \* 2**

**ELSE cantitate**

**END,**

**pret =**

**CASE**

**WHEN ID\_Produs IN (**

**SELECT ID\_Produs**

**FROM (**

**SELECT ID\_Produs, COUNT(DISTINCT ID\_Client) AS nr\_clienti**

**FROM Comenzi**

**GROUP BY ID\_Produs**

**HAVING COUNT(DISTINCT ID\_Client) = 3**

**) Produse\_Trei\_Clienti**

**) THEN pret \* 0.8**

**ELSE pret**

**END**

**WHERE ID\_Produs IN (**

**SELECT ID\_Produs**

**FROM (**

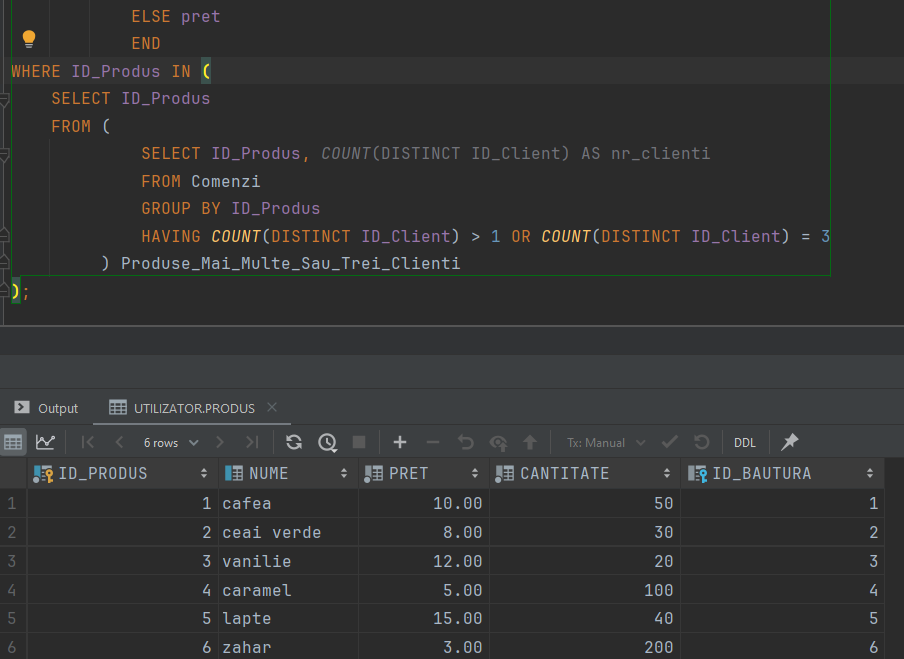
**SELECT ID\_Produs, COUNT(DISTINCT ID\_Client) AS nr\_clienti**

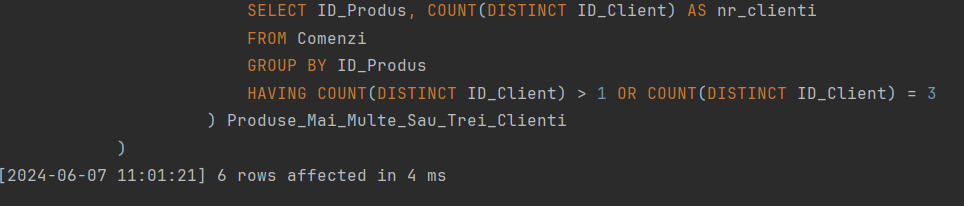
**FROM Comenzi**

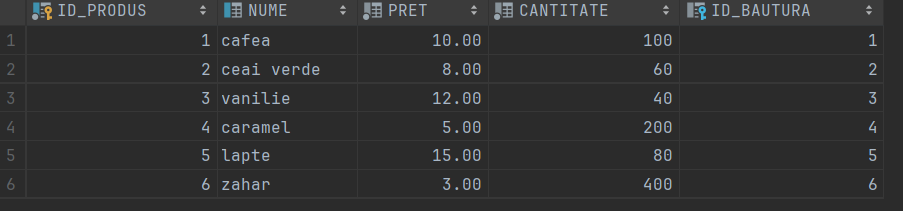
**GROUP BY ID\_Produs**

**HAVING COUNT(DISTINCT ID\_Client) > 1 OR COUNT(DISTINCT ID\_Client) = 3**

**) Produse\_Mai\_Multe\_Sau\_Trei\_Clienti**

**);**

****

****