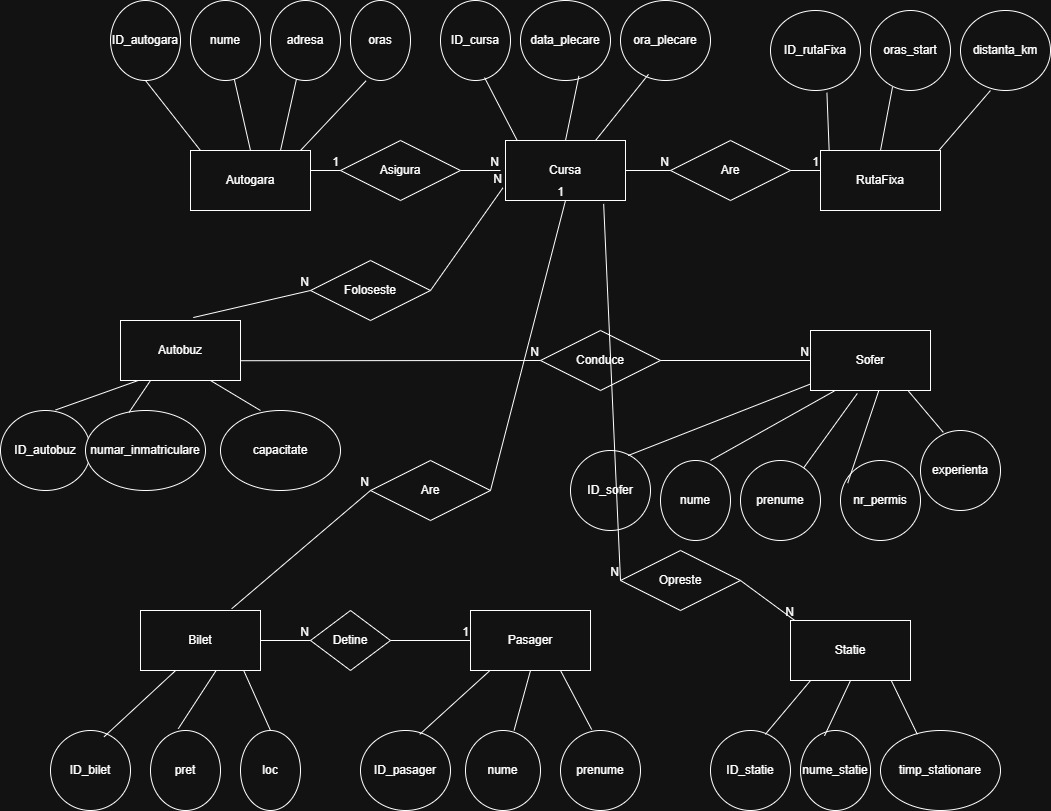
Bejoiu Roxana-Georgiana

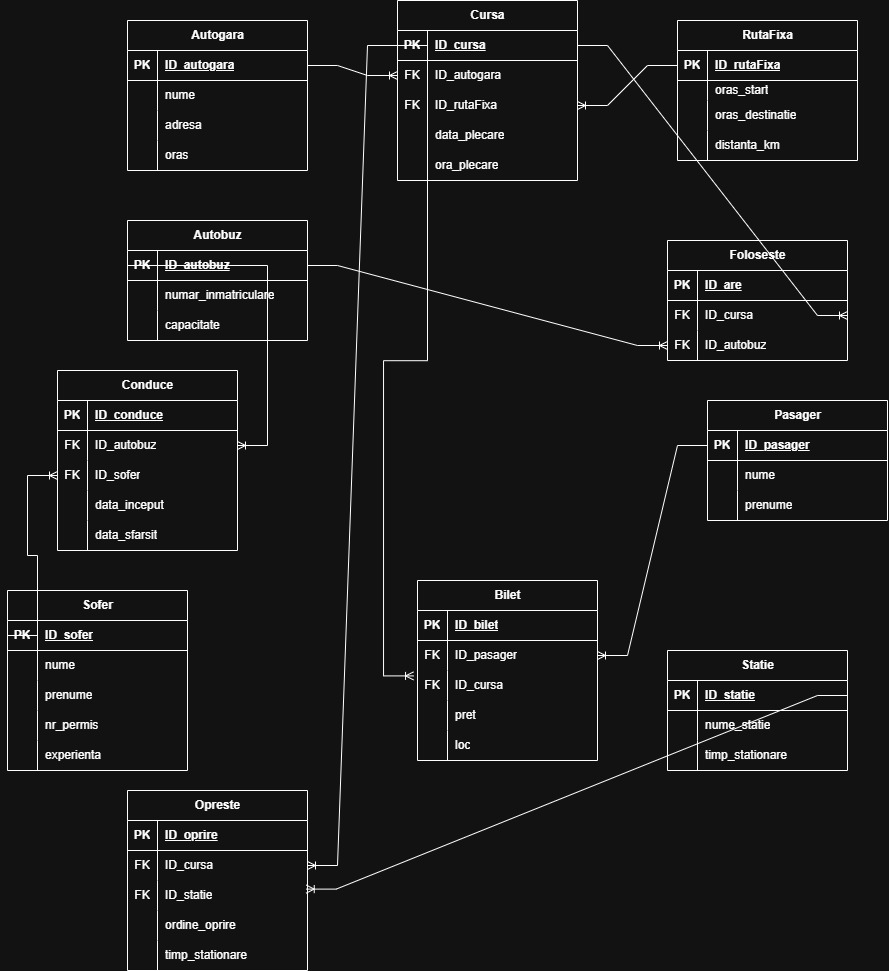
CR3.1A

Proiect BD – Autogara

1. (1p)Modelul ER entitate – legatura. Prezentarea entitatilor si a legaturilor dintre acestea, impreuna cu cardinaliatea lor. Baza de date trebuie sa contina minimum 7 entitati.



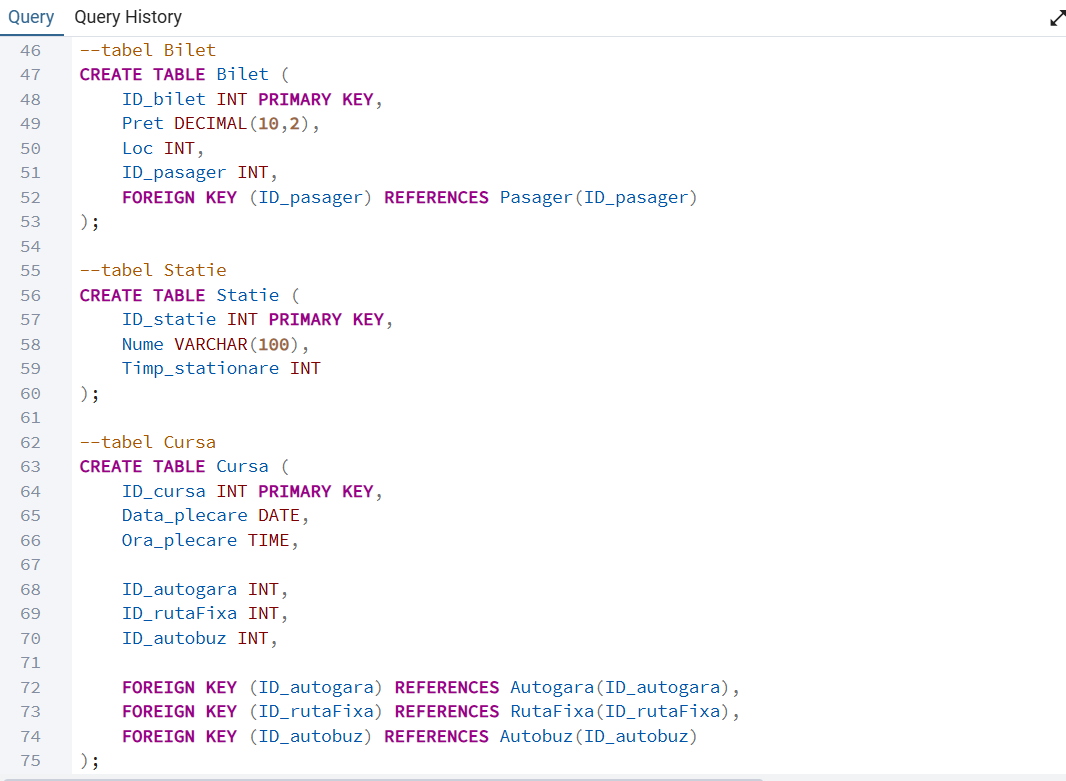
1. (1p)Schema relationala (stabilirea relatiilor, atributelor, cheilor primare, constrangerilor, legaturilor dintre relatii - chei externe)

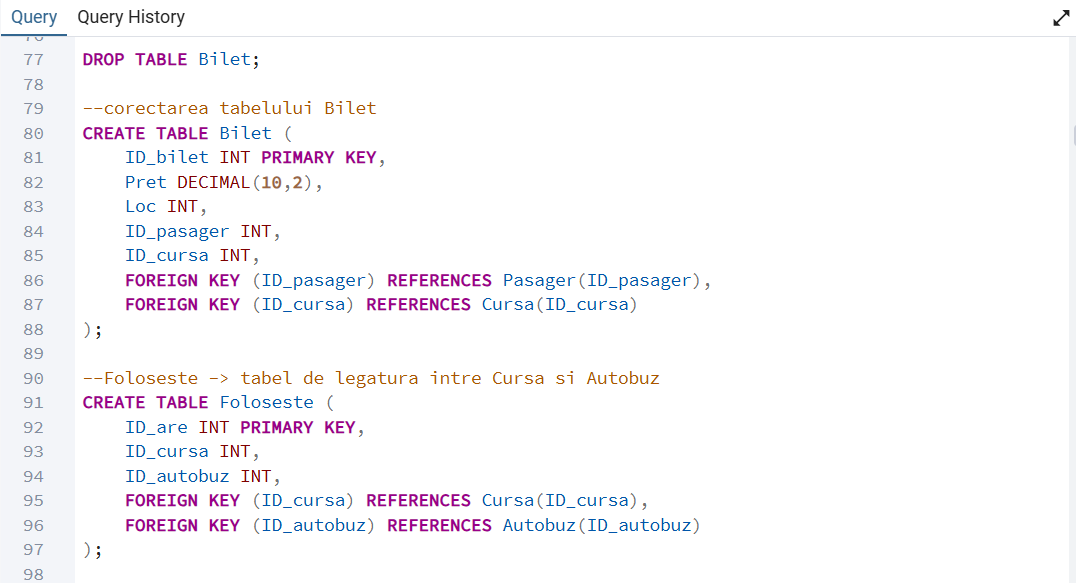


3. (1.5p)Implementarea modelului relational folosind SQL – crearea tabelelor, constrangerilor + Operatii de manipulare a relatiilor/tabelelor – inserare, modificare, stergere.



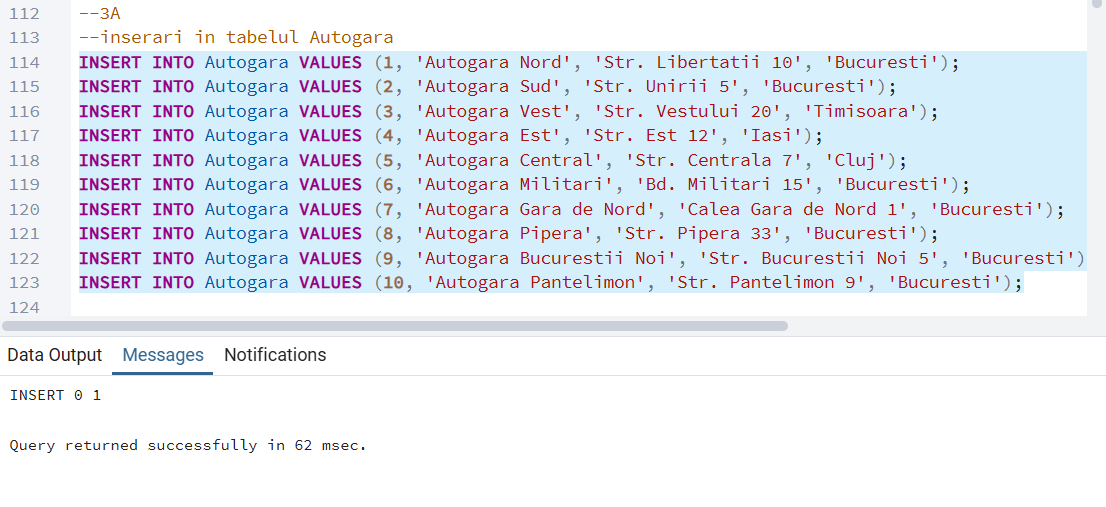


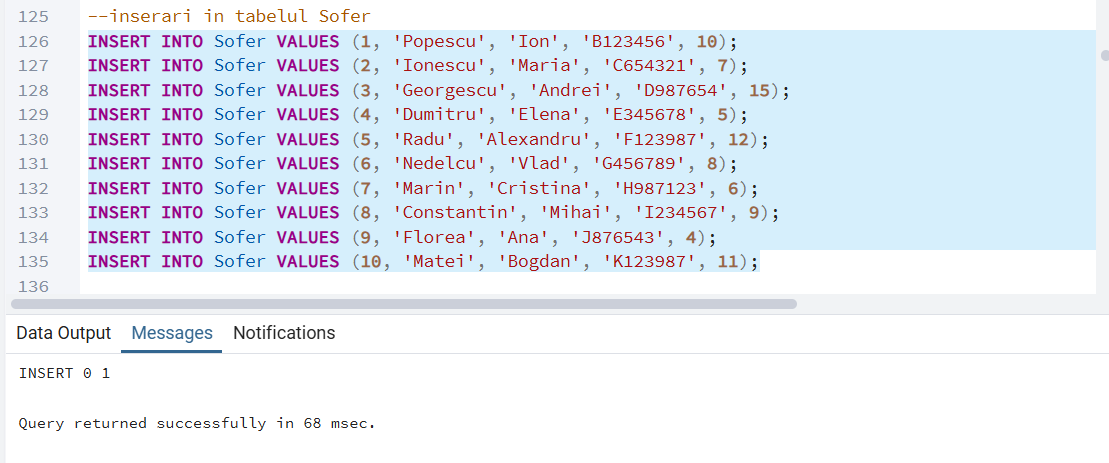


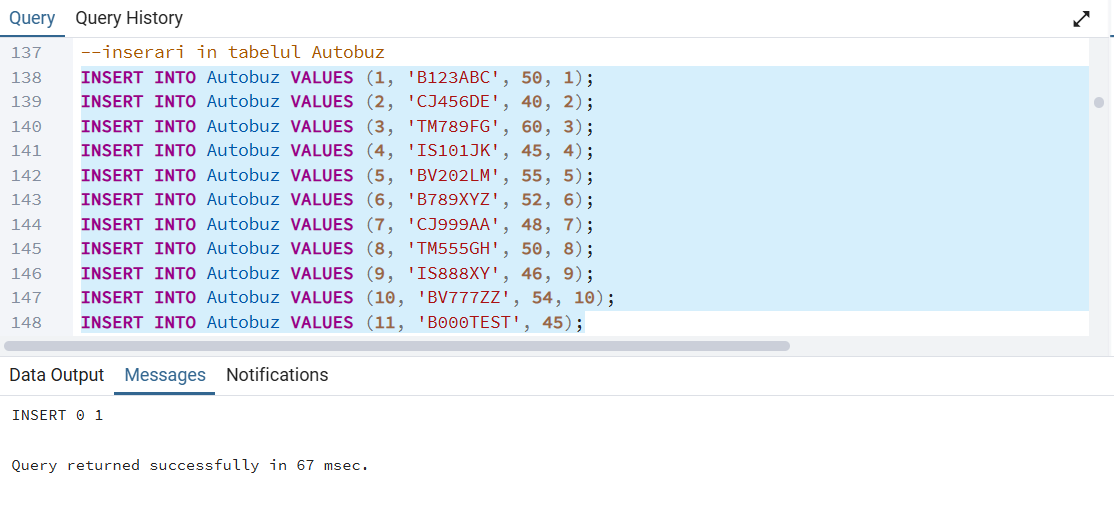


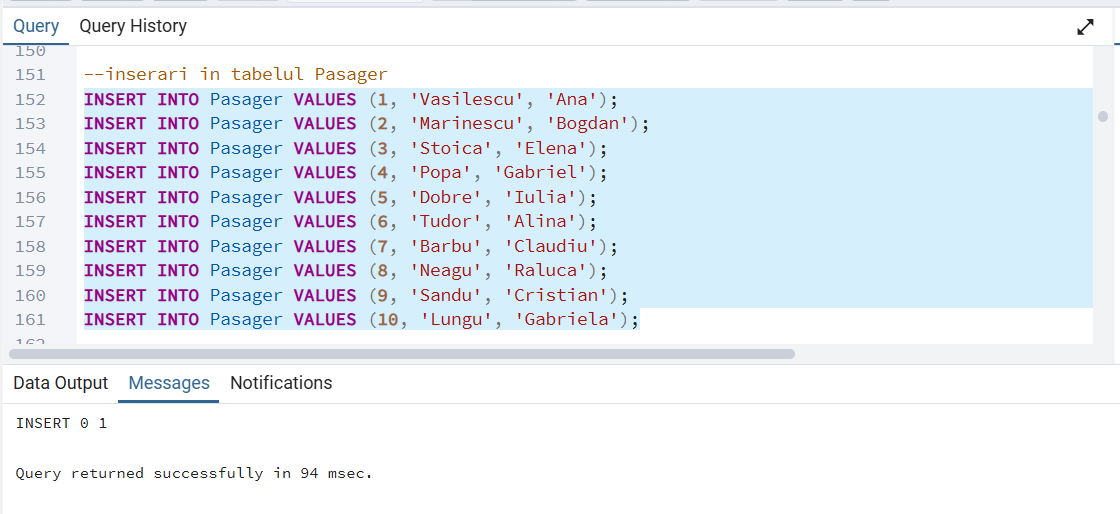


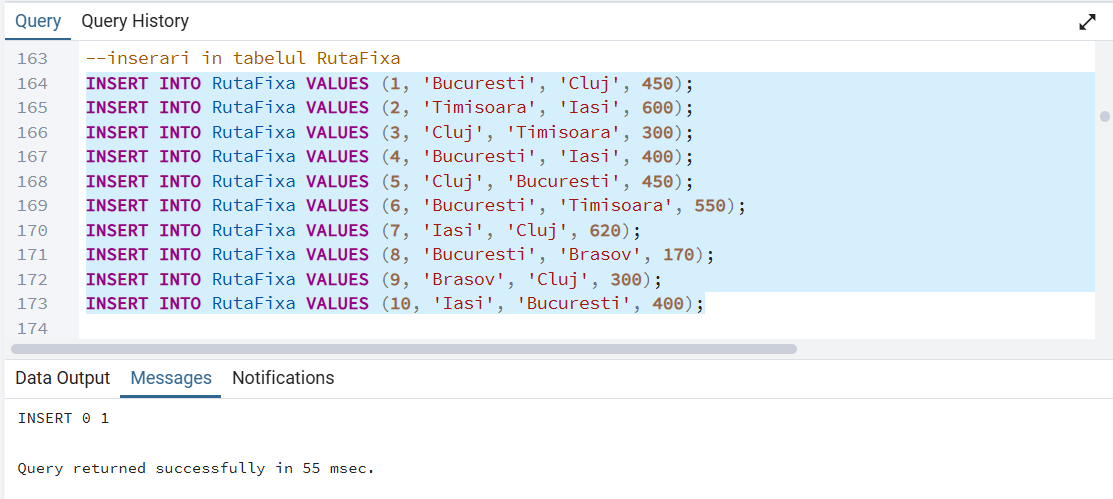
1. Fiecare tabel va contine cel putin 5 inregistrari.

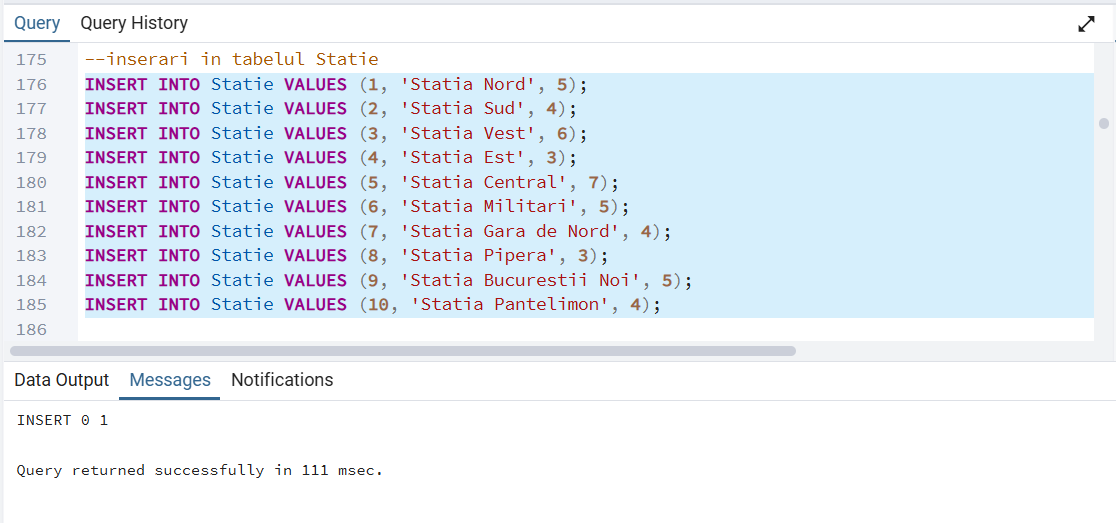


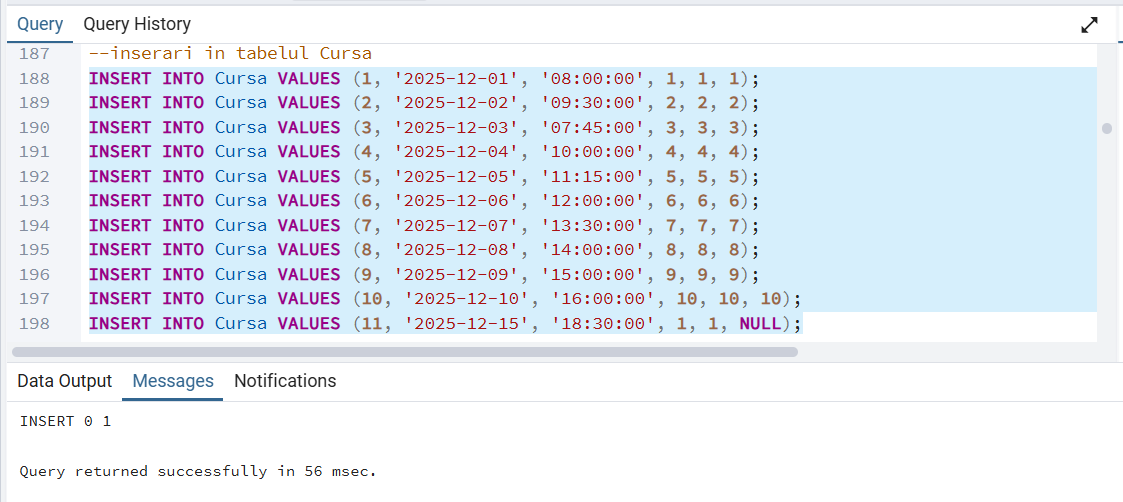


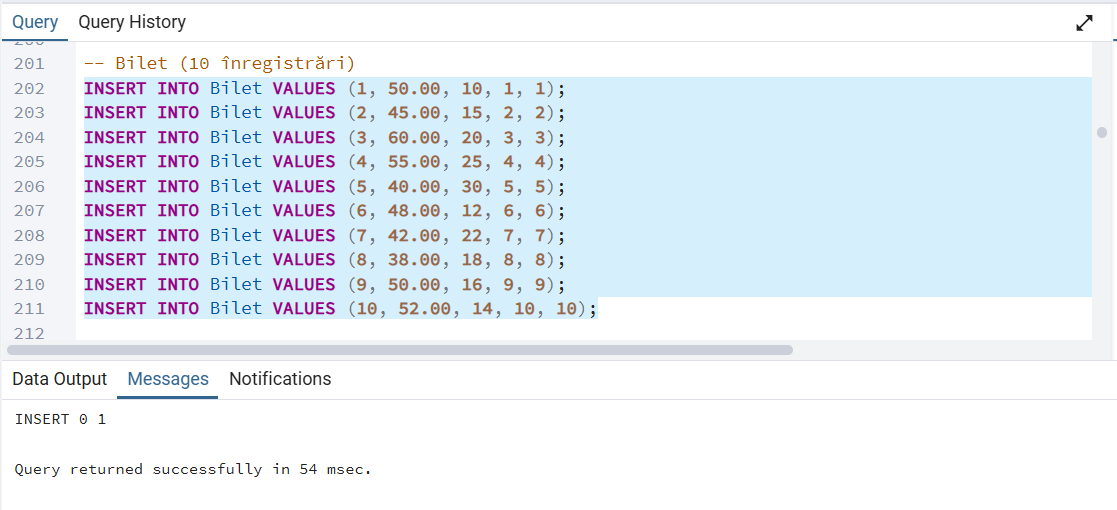


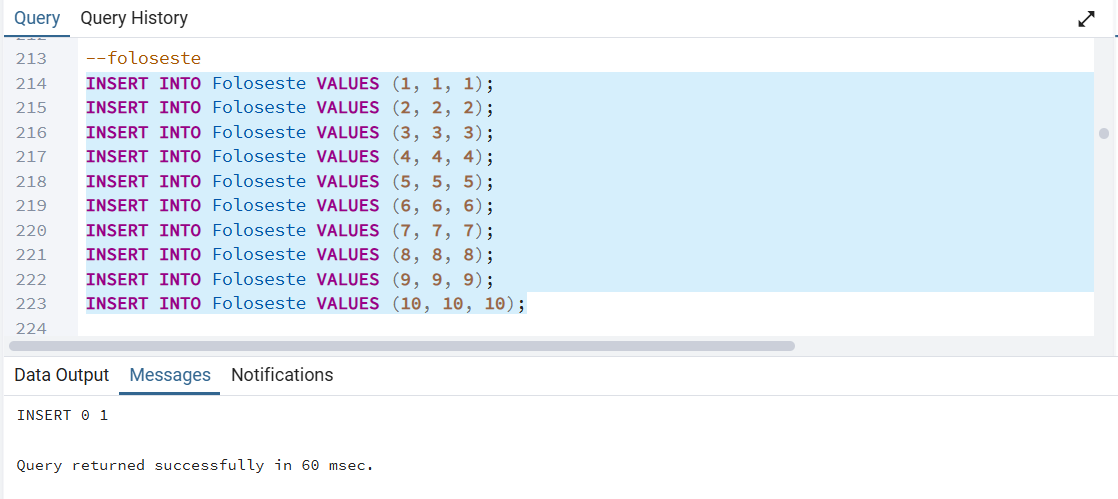


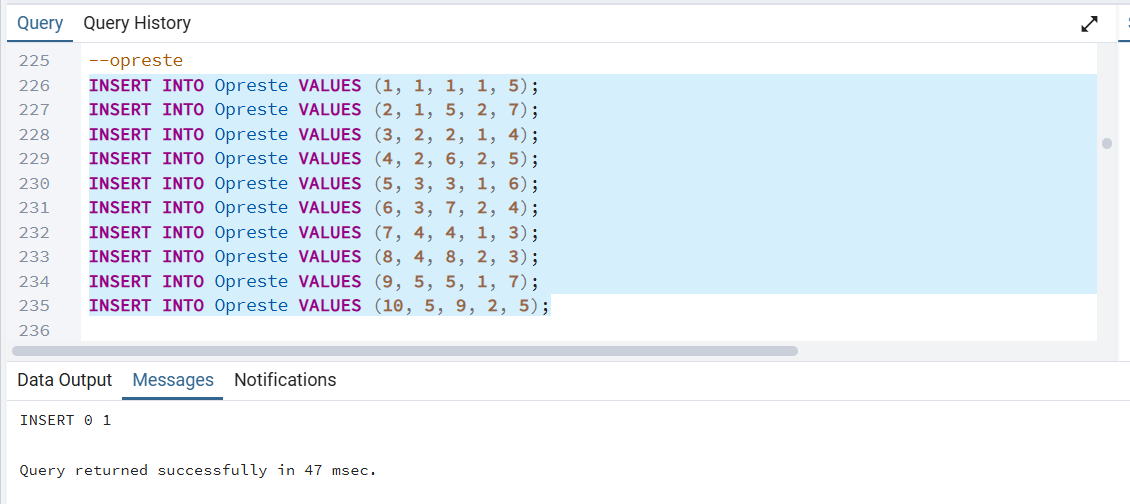




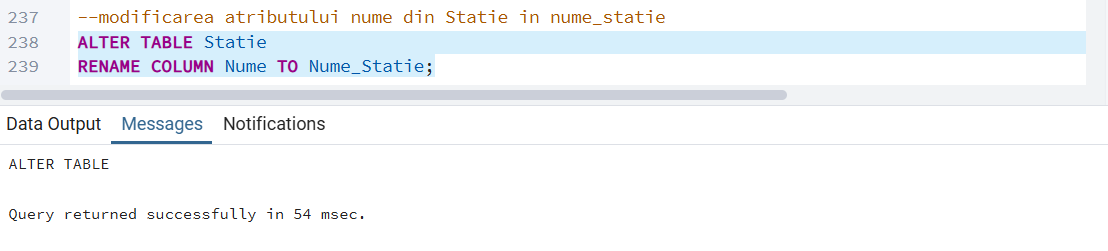




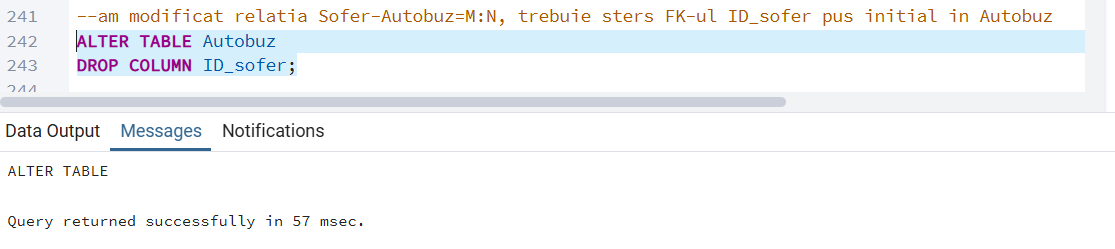




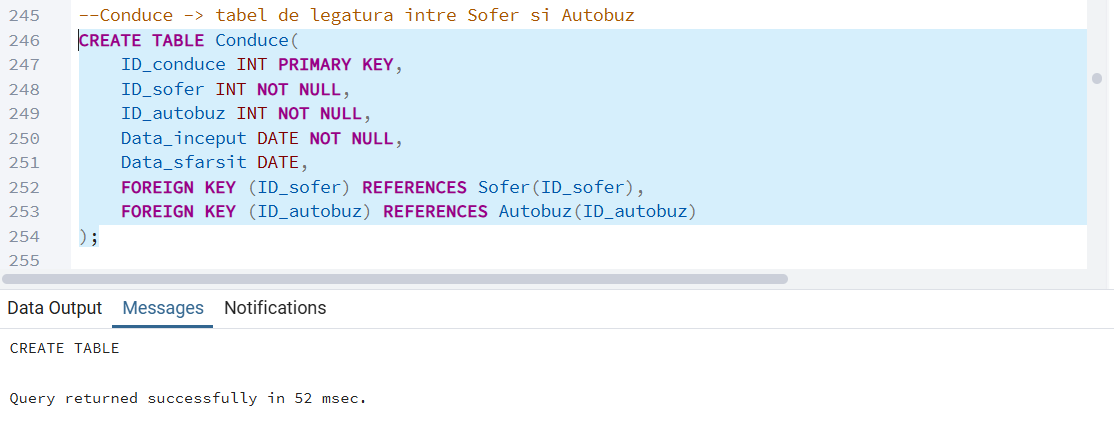
Se poate observa ca inserarile au fost facute cu succes.



Am facut o modificare a denumirii atributului pentru a fi mai explicita.

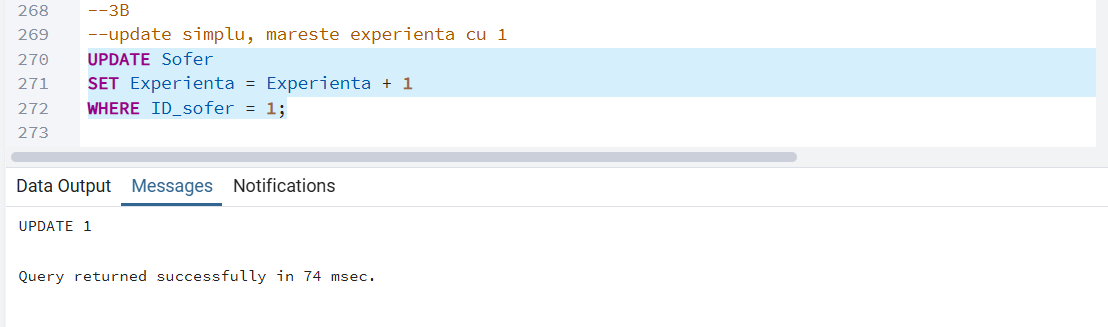


Prima data am facut relatia Sofer-Autobuz 1:N, dar aceasta nu era in tocmai corecta si am modificat stergand ID\_sofer din tabelul Autobuz si adaugand tabelul de legatura – Conduce.

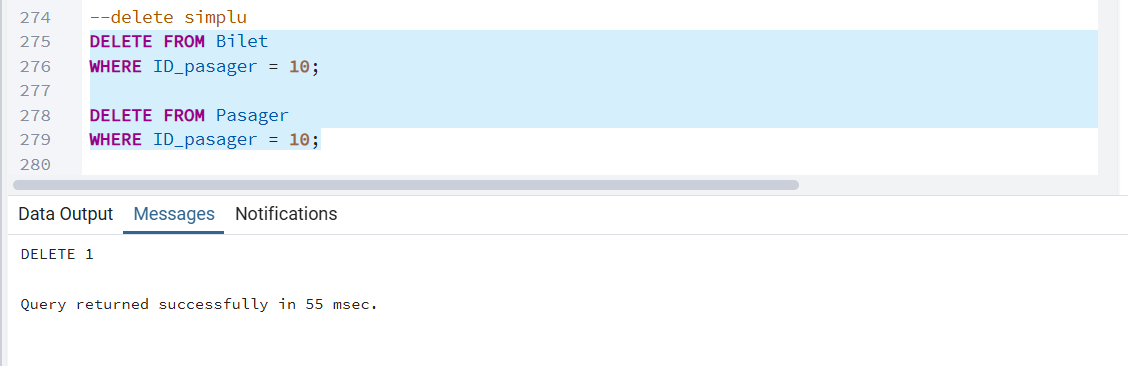


La acest subpunct am creat toate tabelele conform modelului ER si a schemei relationare create.

1. Se vor da exemple de operatii de actualizare si stergere a datelor.

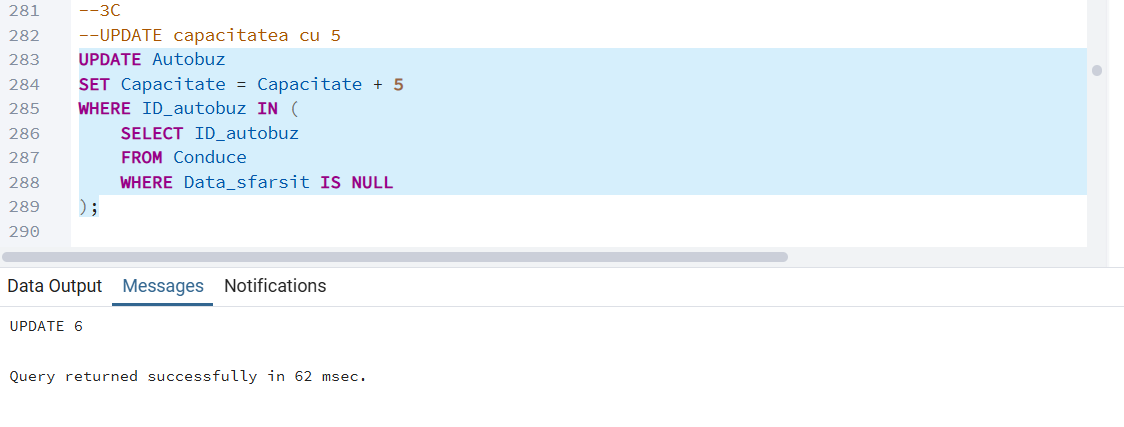


Exemplu de update in tabelul Sofer, am incrementat experienta actual inserata cu 1.

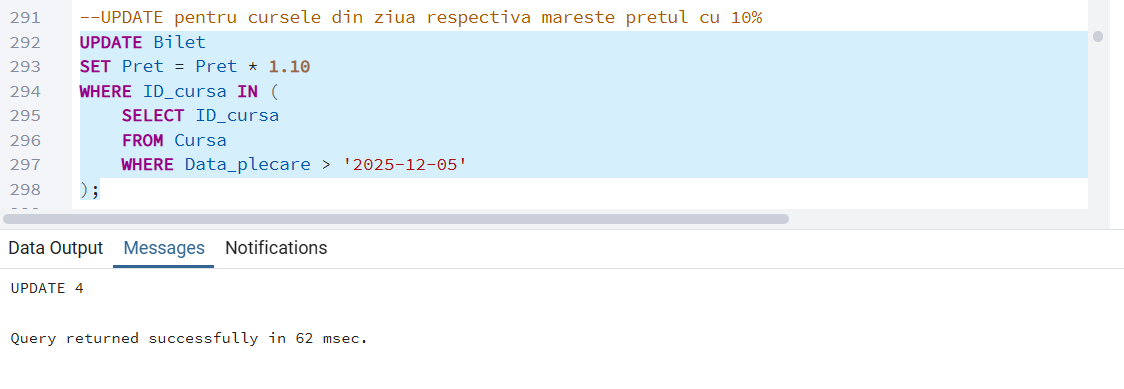


Exemplu de delete in tabelele Bilet si Pasager unde ID\_pasager=10

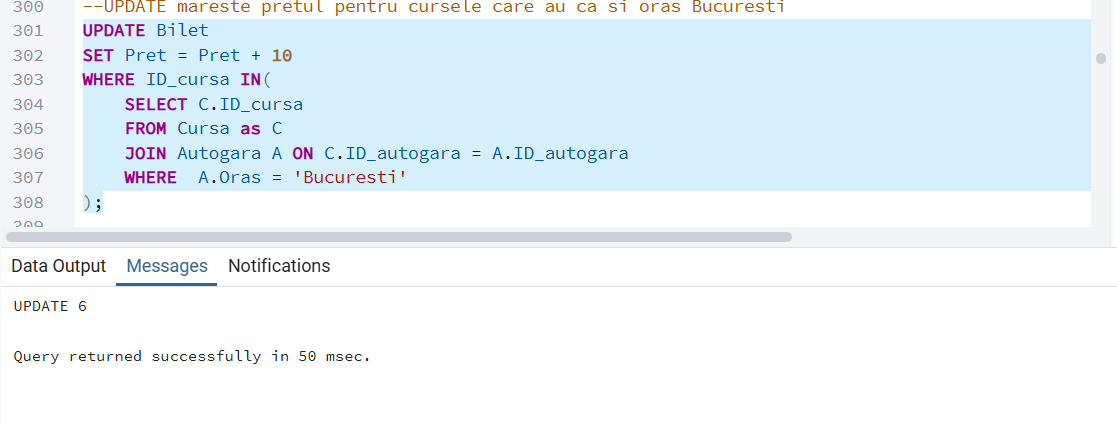
1. Implementarea a minimum 3 operații de actualizare sau stergere a datelor utilizând subinterogari



Am facut un update a atributului Capacitate in tabela, acolo unde ID\_autobuz se regaseste in tabela de legatura Conduce unde Data\_sfarsit IS NULL, adica soferul conduce in continuare autobuzul respectiv.



Am realizat un update al pretului din tabelul Bilet, acolo unde Data\_plecare > '2025-12-05'



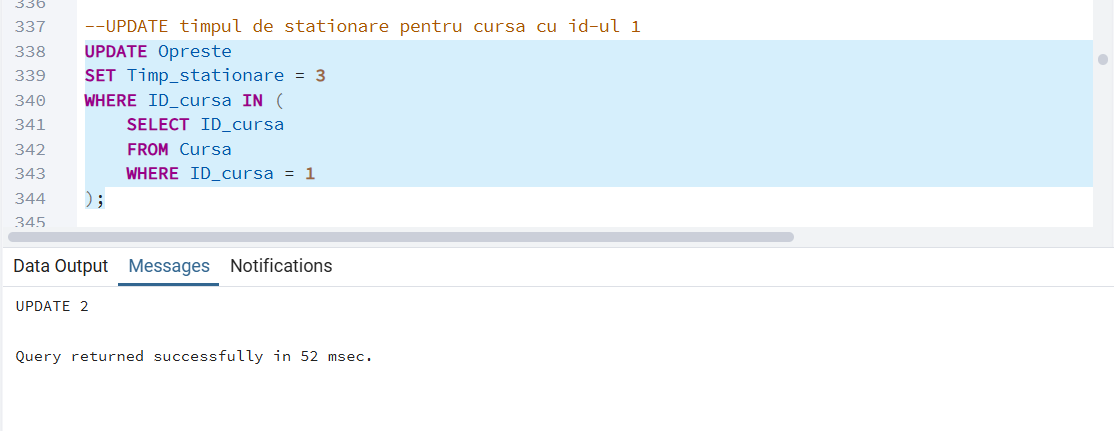
Am realizat un update al pretului din tabelul Bilet, orasul in care se afla Autogara este Bucuresti.



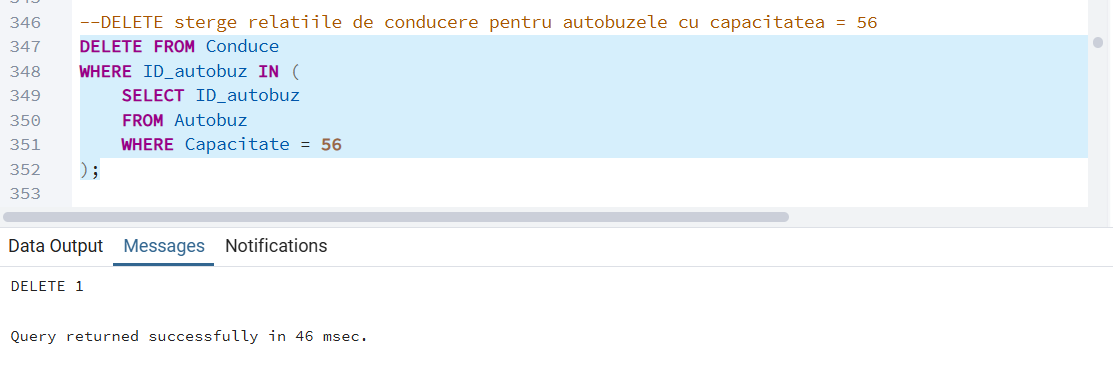
Update care mareste experienta doar pentru soferi care conduc un autobuz cu capacitatea mai mare de 50 de locuri.



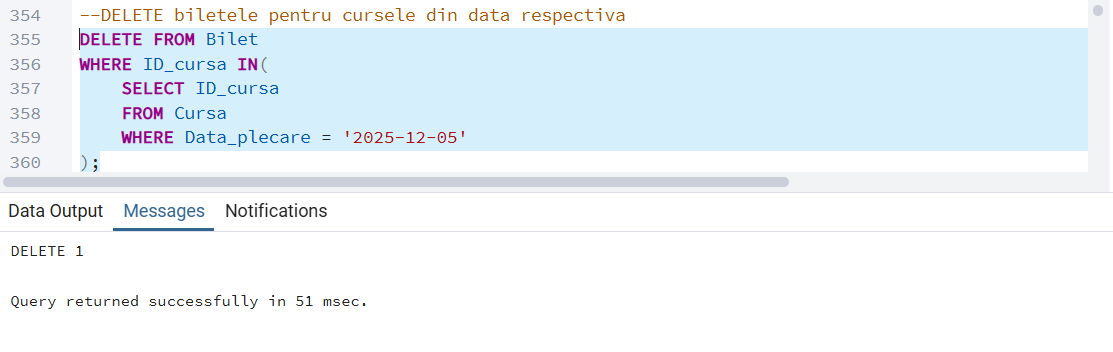
Update care creste capacitatea doar pentru autobuzele carea au ca si oras – Bucuresti.

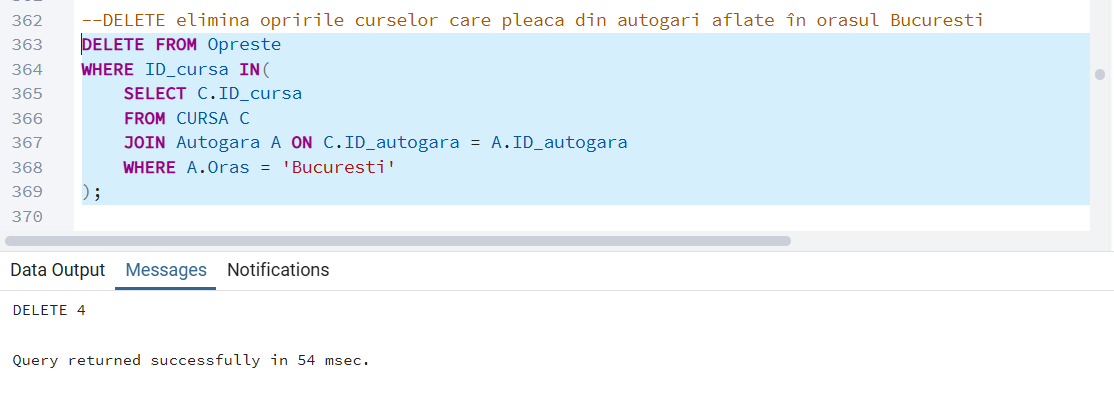


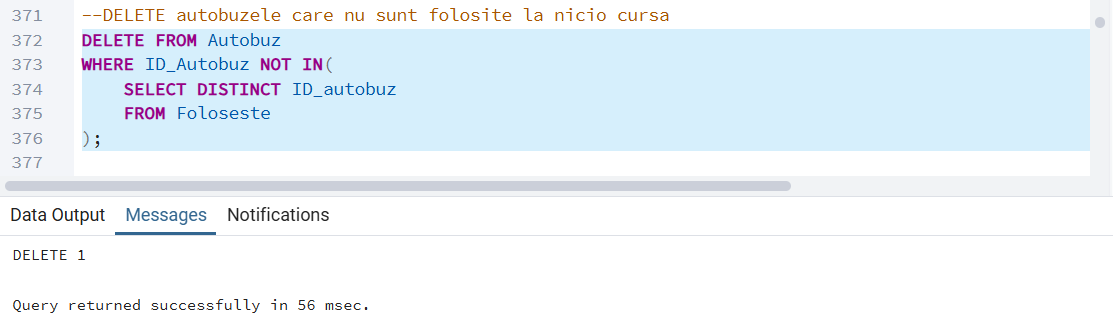
Update pentru a modifica timpul de stationare pentru cursa cu update-ul 1.



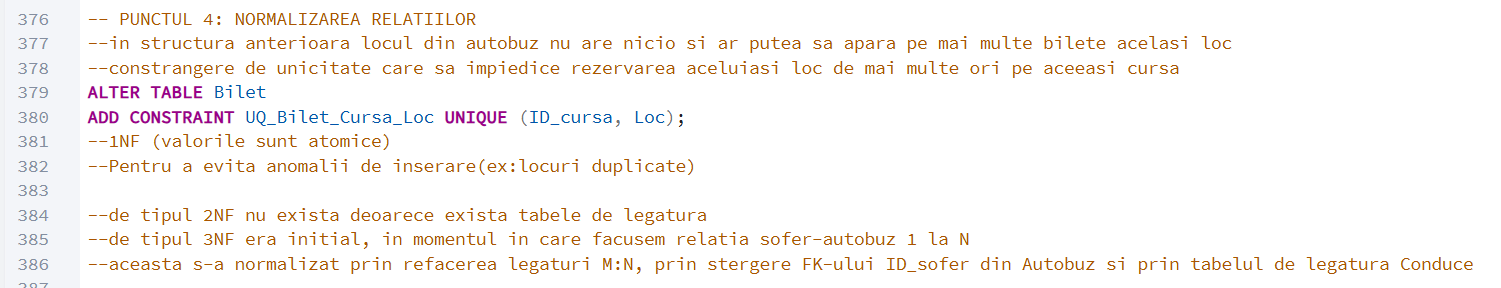
Delete care sterge din tabelul de legatura Conduce autobuzele care au capacitatea de 56 de locuri.







4.(0.5p)Normalizarea relatiilor. Pe baza de date aleasa, se vor da exemple care nu respecta formele normale.

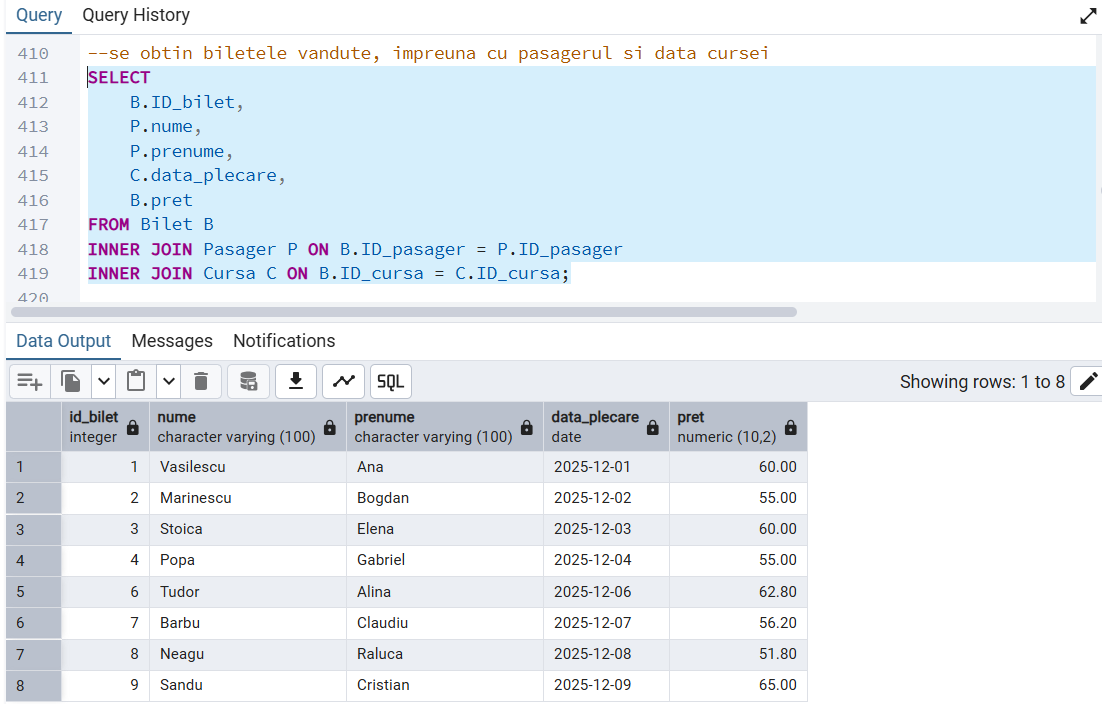


5.(3)Interogari

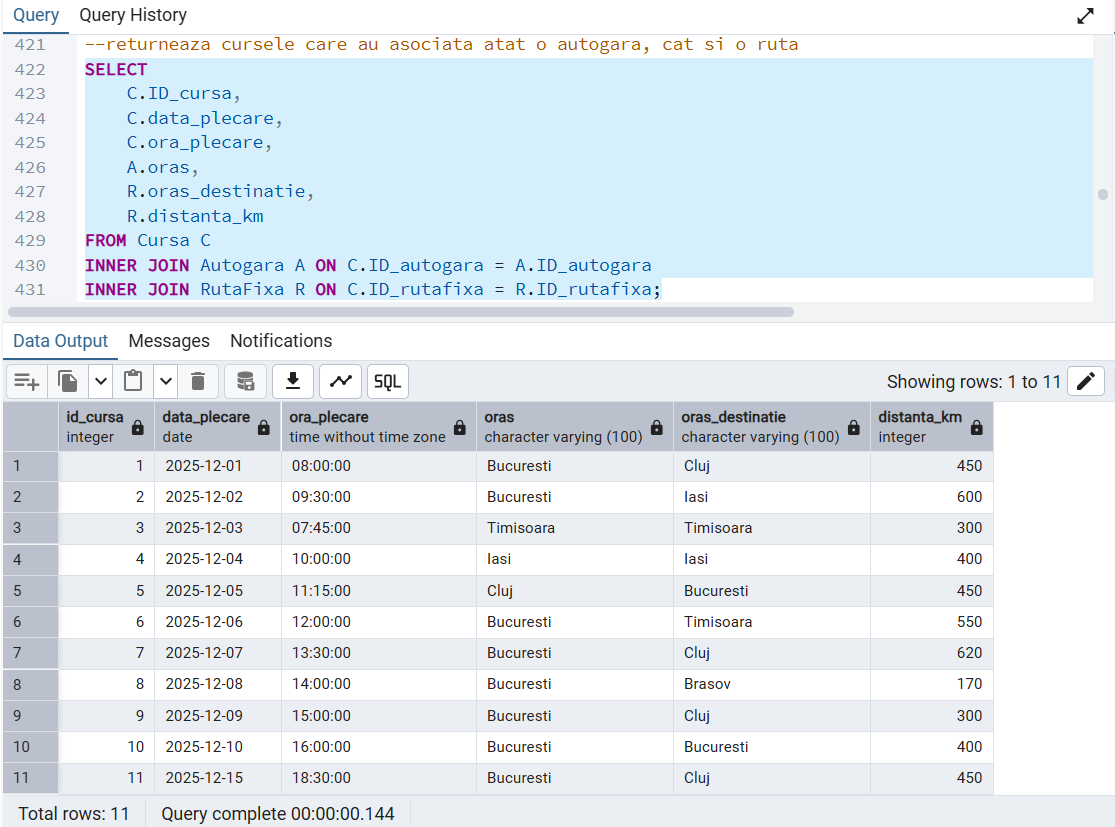
a. Minim 4 interogari cu operatii de jonctiune (inner join+outer join)

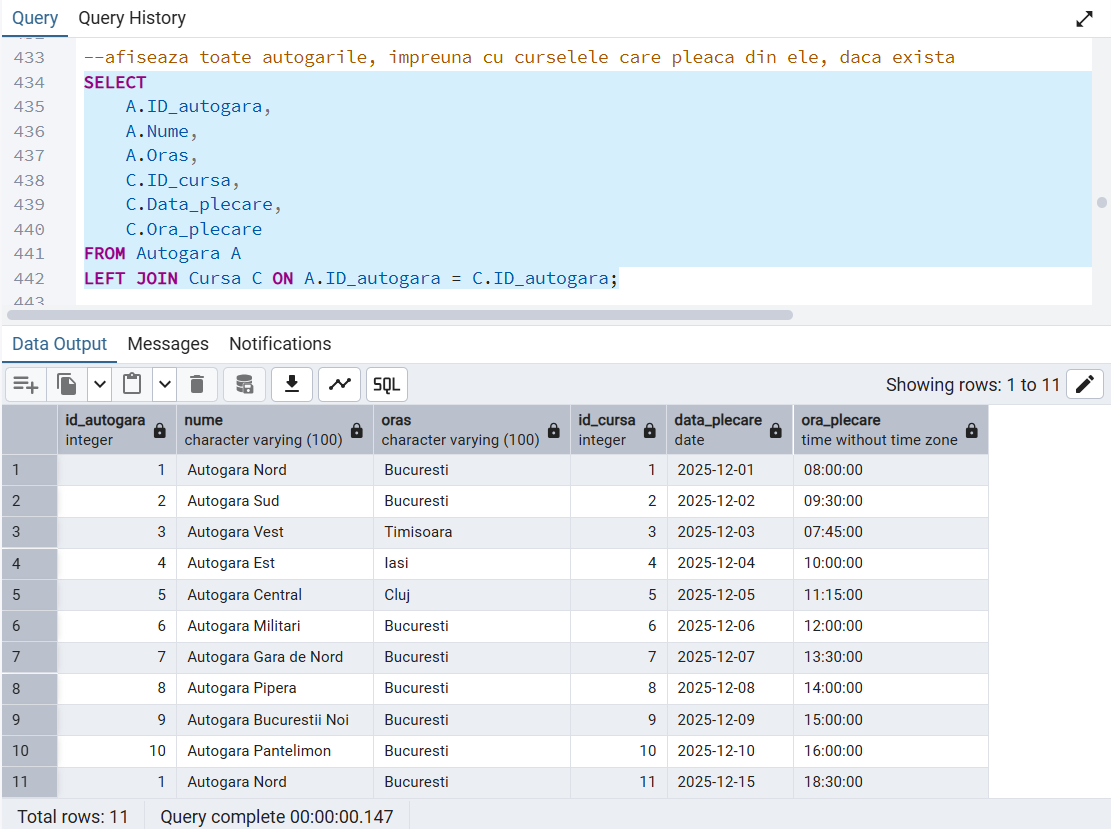


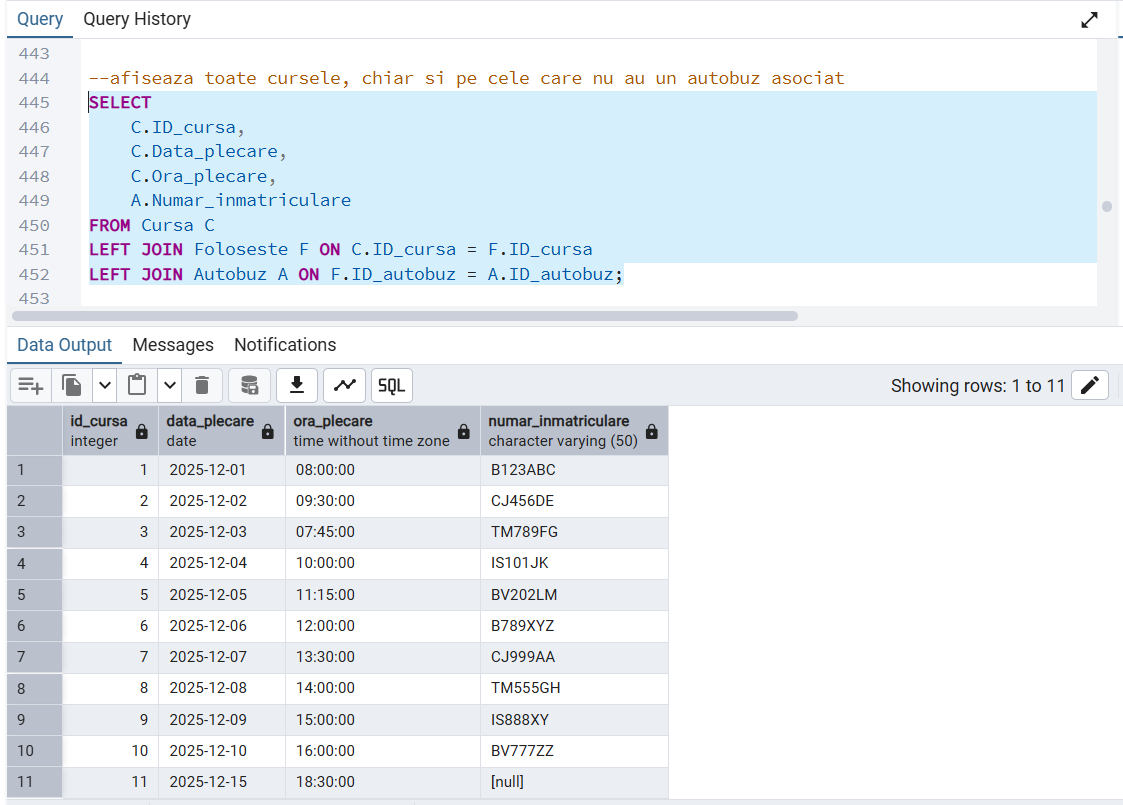
Am selectat atributele pe care doresc sa le afisez din tabelele Cursa si Autobuz, iar prin INNER JOIN am facut o legatura prin atributul ID\_autobuz.



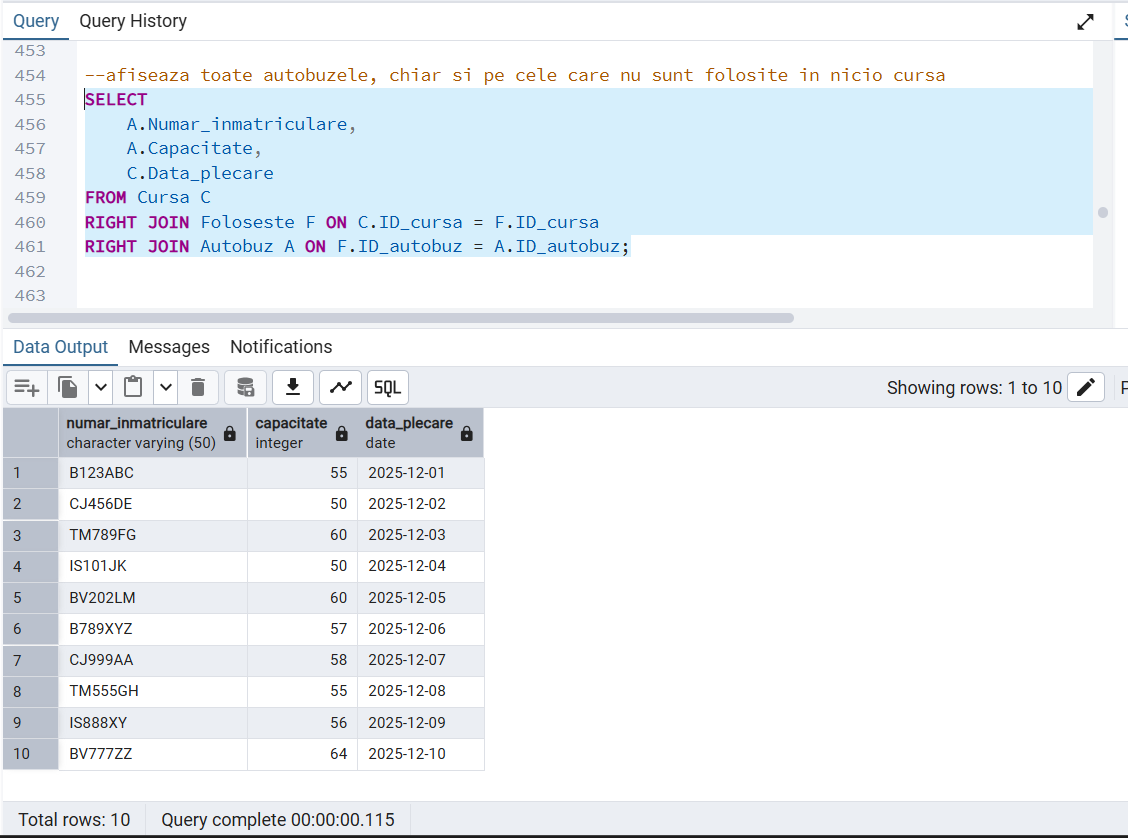
Prin INNER JOIN am creat o legatura intre Pasager si Bilet prin ID\_pasager, asociind fiecarui pasager un bilet si o legatura intre Bilet si Cursa prin ID\_cursa, atribuind fiecarui bilet o cursa corespunzatoare.





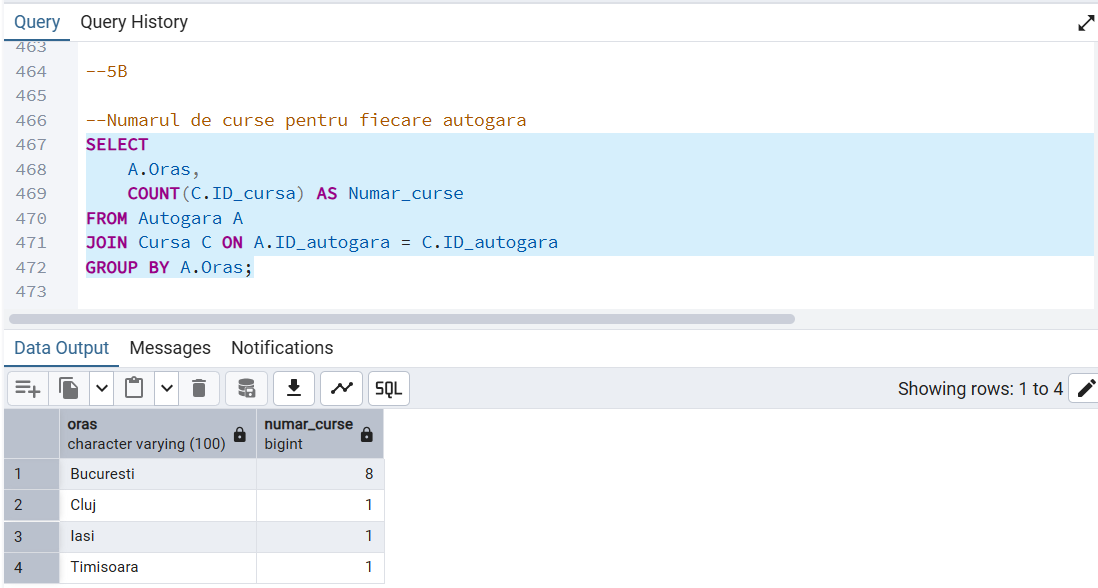


Am utilizat LEFT JOIN, care leaga cursele cu autobuzele. Aceasta asigura ca toate cursele apar în rezultat, chiar daca nu au autobuz asociat. Cel de al doilea LEFT intre tabela **Foloseste** si tabela **Autobuz**, pentru a prelua numarul de inmatriculare al autobuzului care face cursa.

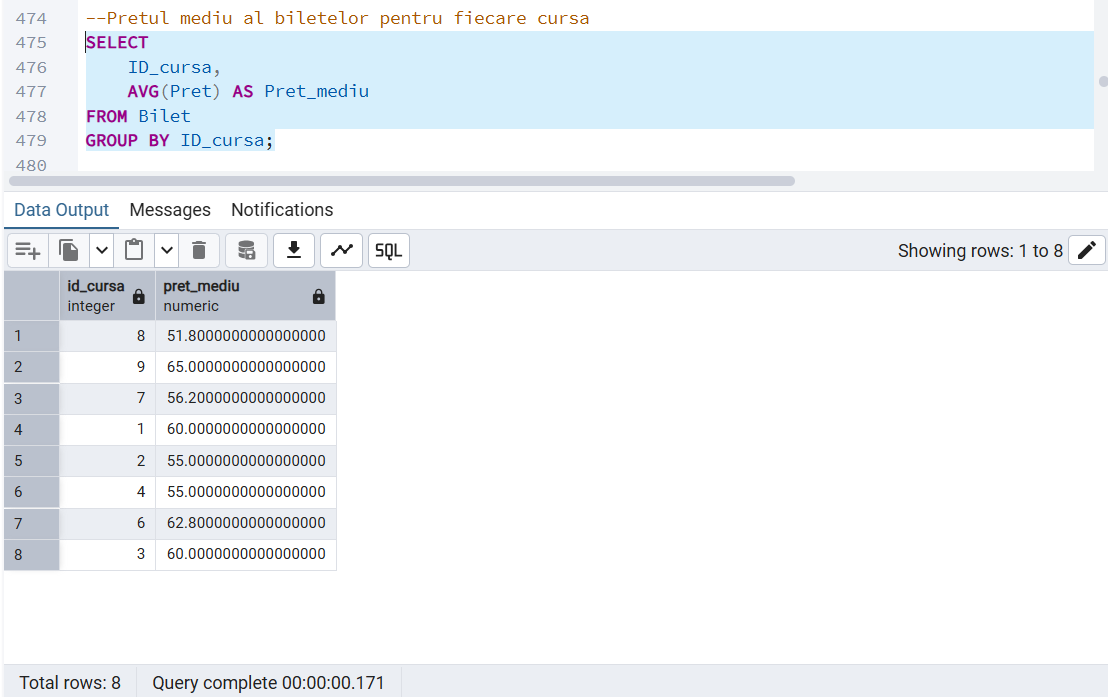


Toate inregistrarile din **Foloseste** vor aparea in rezultat, chiar dacă nu exista o cursa corespunzatoare in tabela **Cursa** (cum ar fi o cursa stearsa sau necompletata).

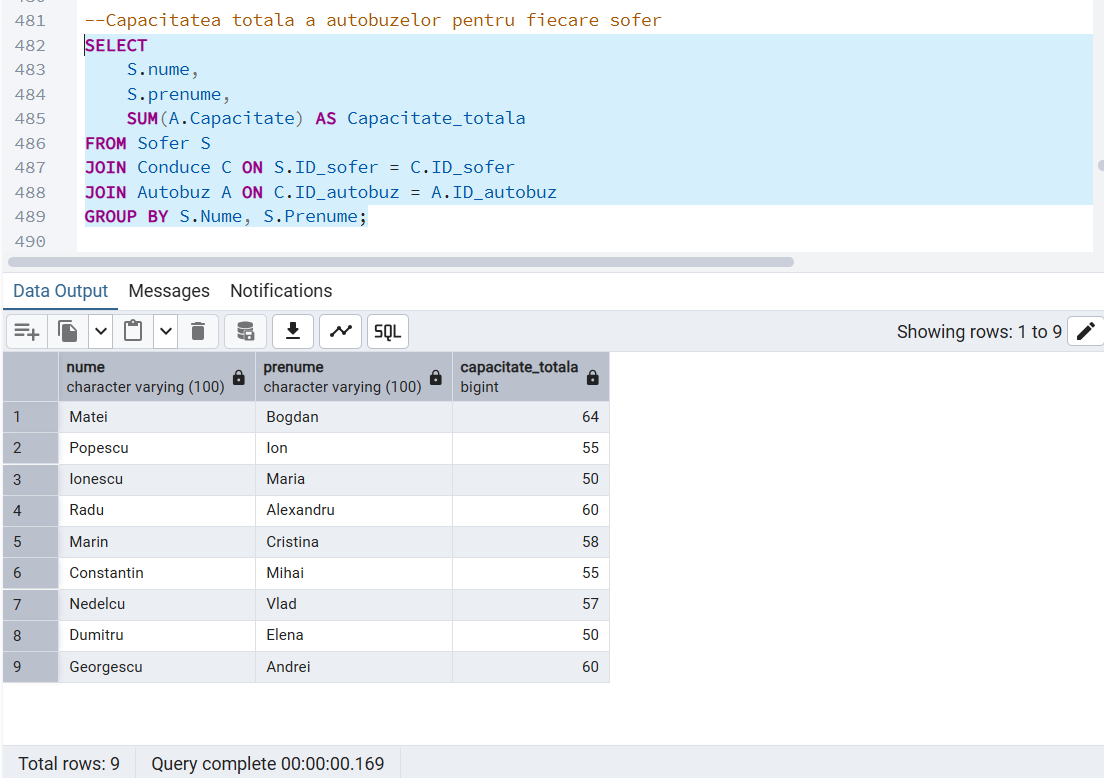
b. Minim 4 interogari care sa contina functii de agregare si Group by



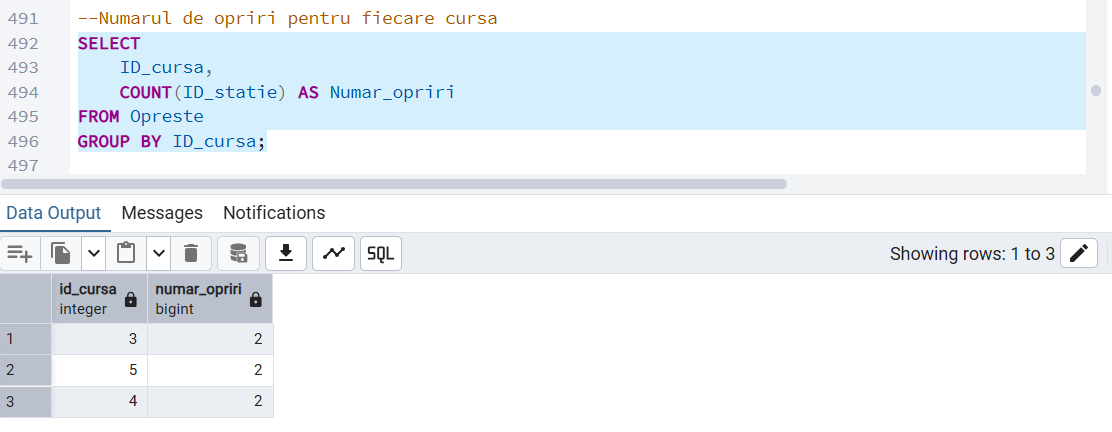
Am folosit Count pentru a contoriza numarul total de curse care pleaca din oricare dintre autogarile din acel oras.

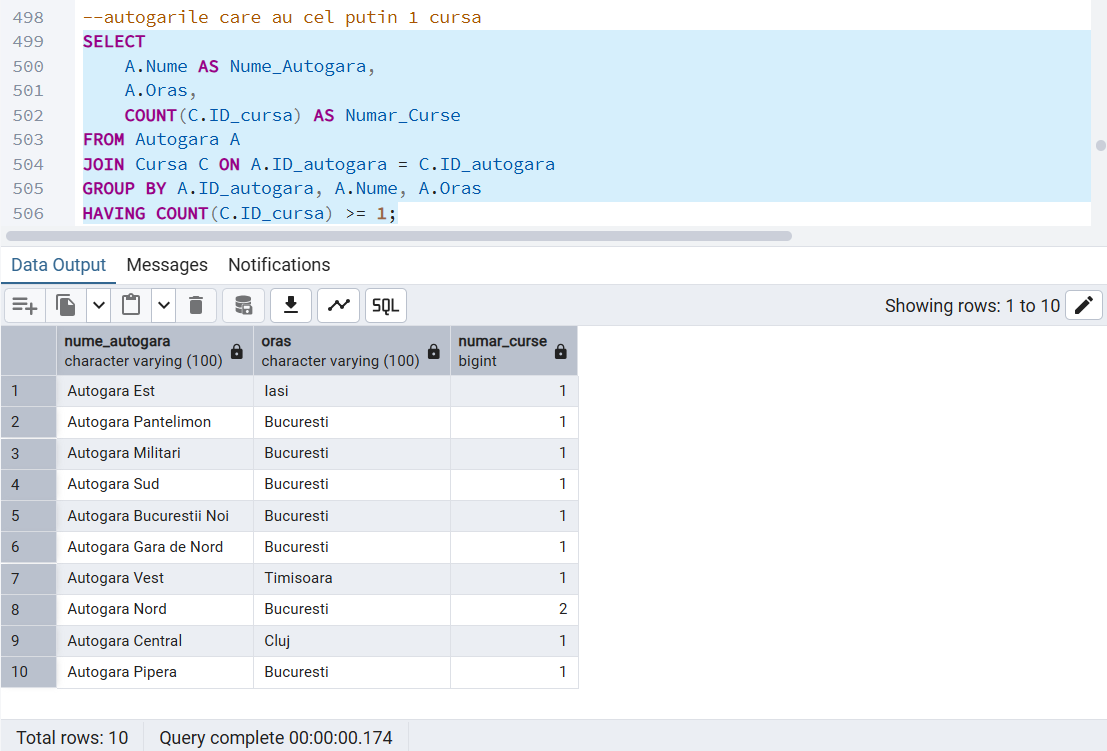


Am folosit AVG pentru a calcula pretul mediu al tuturor biletelor vandute pentru aceea cursa.



SUM - pentru fiecare sofer, calculeaza suma totala a capacitatii autobuzelor pe care le conduce.

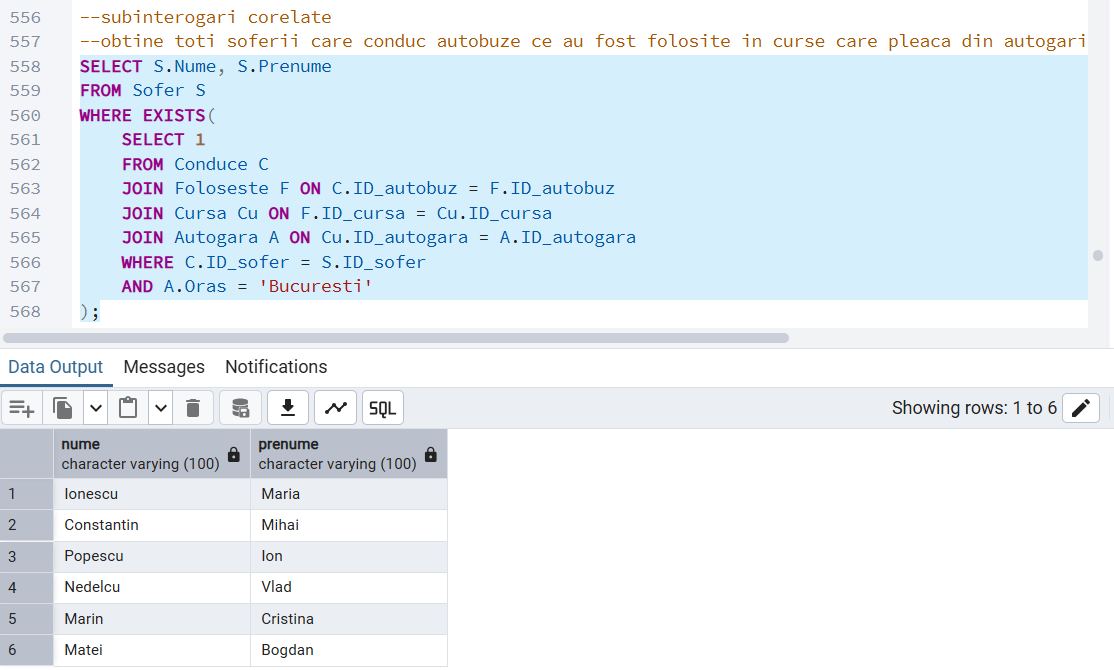




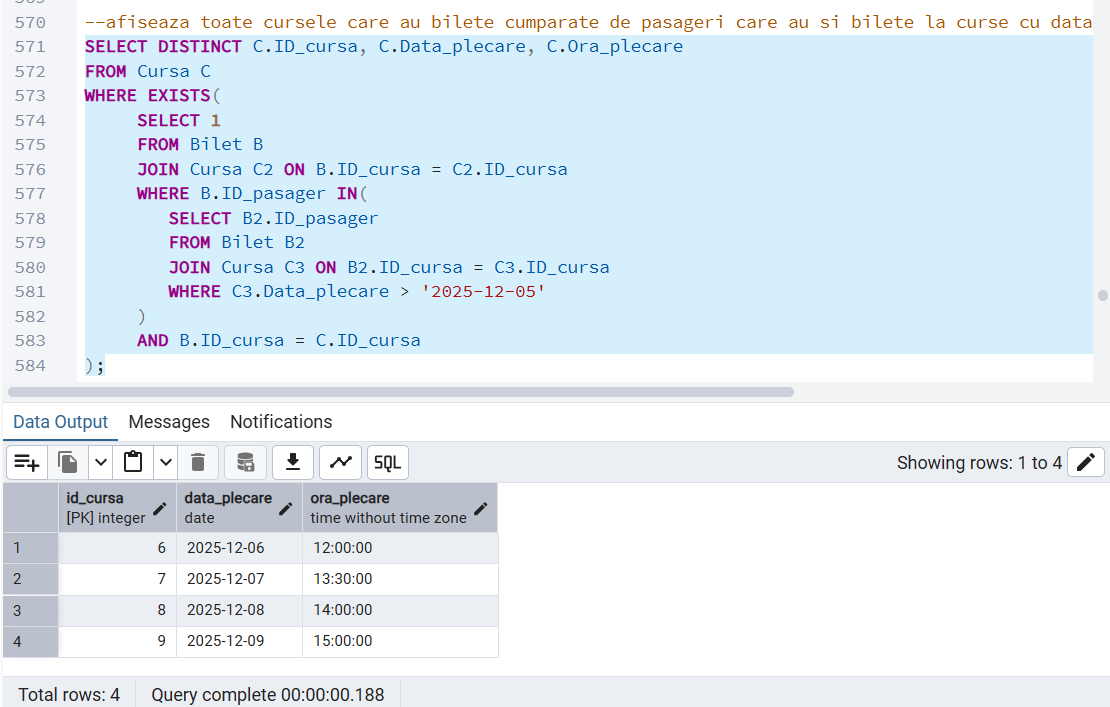
Returneaza lista autogarilor impreuna cu orasul lor si numarul de curse care pleaca din fiecare si sunt afisate doar autogarile care au cel putin o cursa.

c. Minim 4 exemple de subinterogari (IN , ANY, Exists, etc)

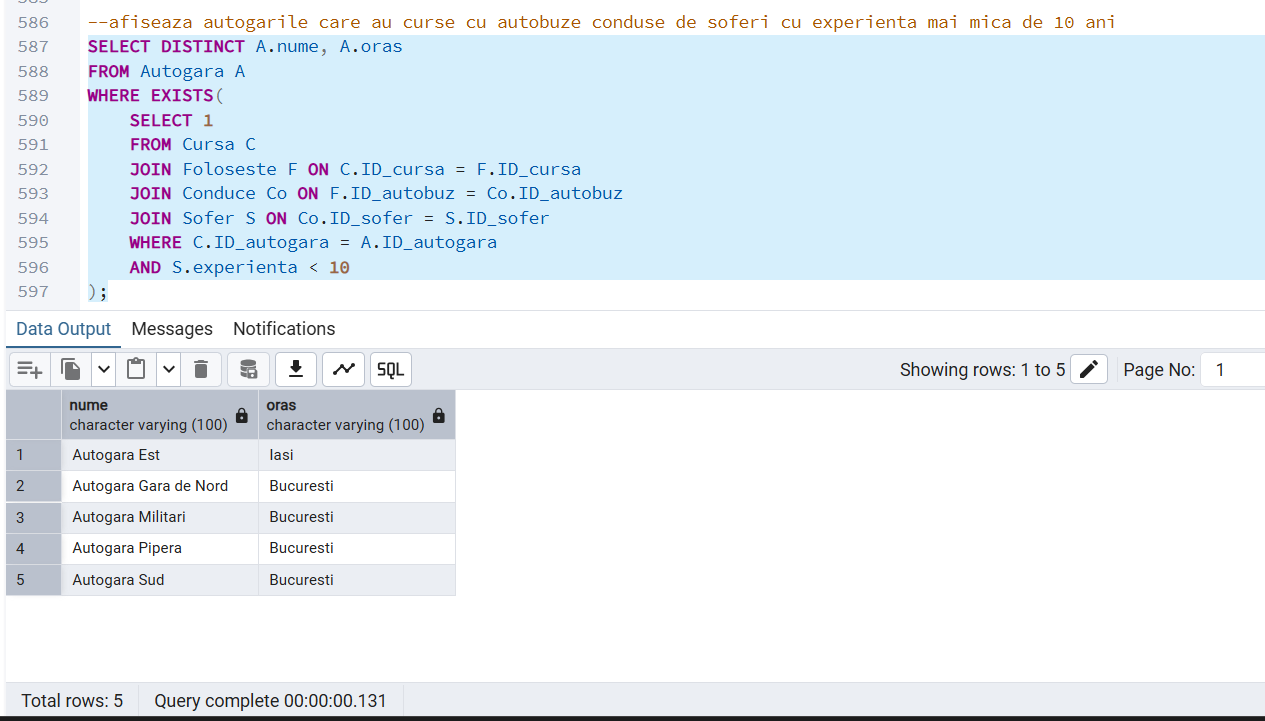
i. Subinterogari corelate în care intervin cel puțin 3 tabele

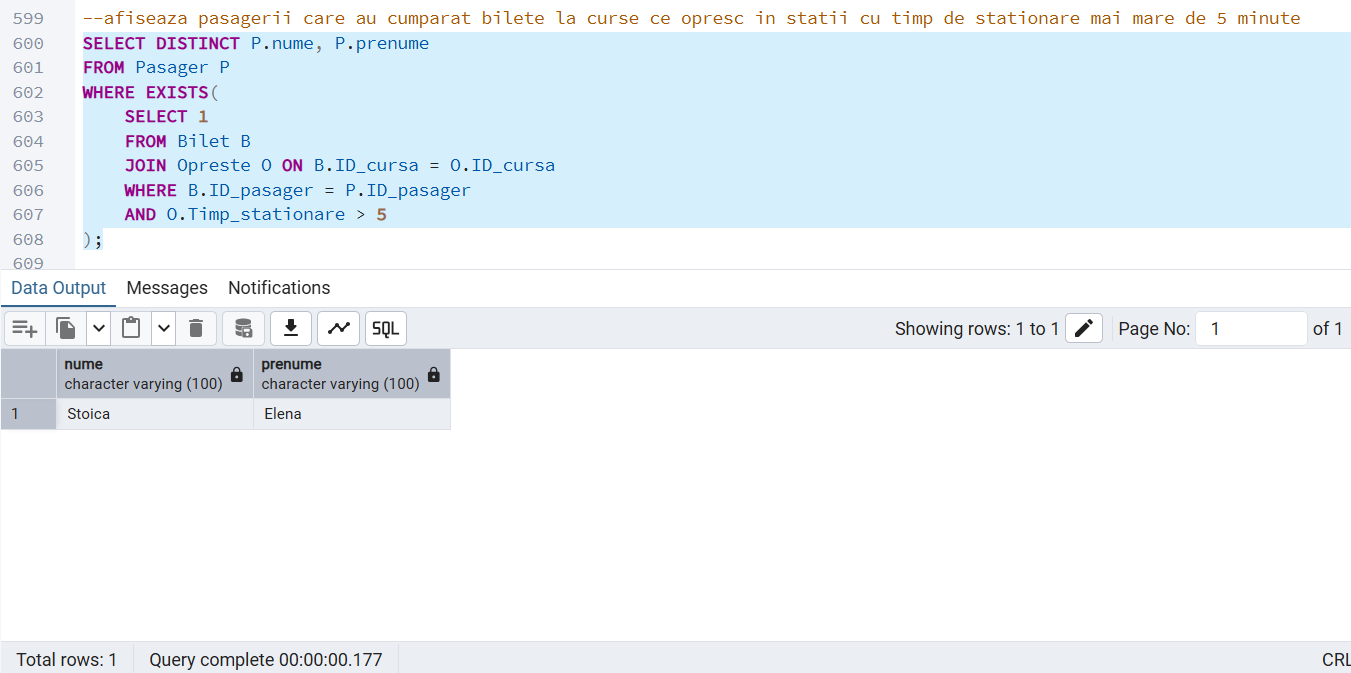


Interogarea returneaza numele si prenumele soferilor care conduc autobuze folosite in curse ce pleaca din autogari situate in Bucuresti.



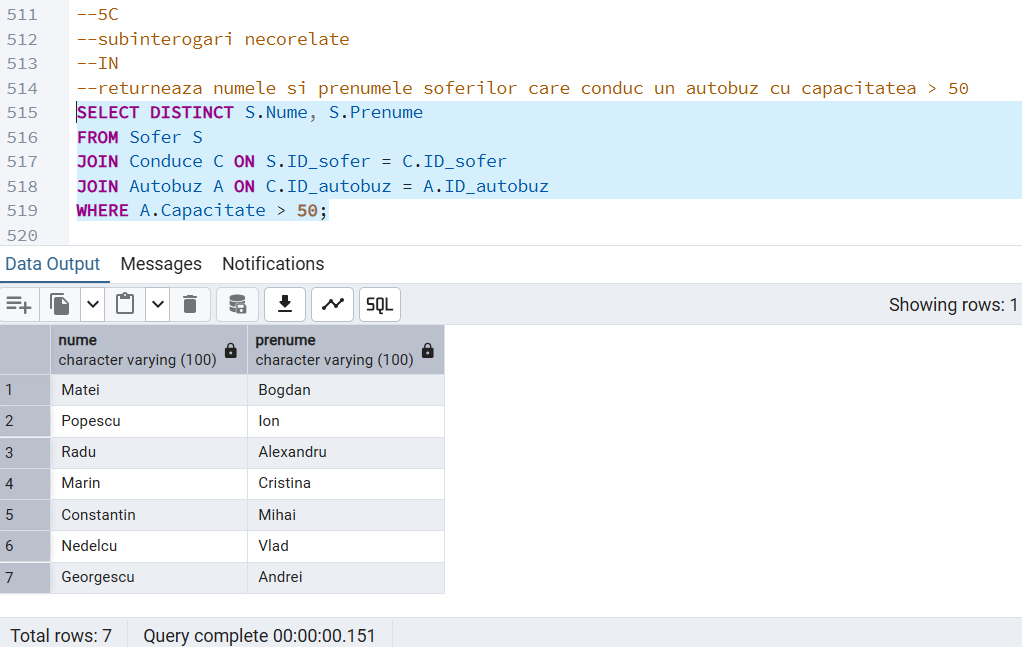
Interogarea gaseste toti pasagerii care au bilete pentru curse dupa 5 decembrie 2025 si am aflat toate cursele la care acesti pasageri au cumparat bilete (indiferent de data cursei).

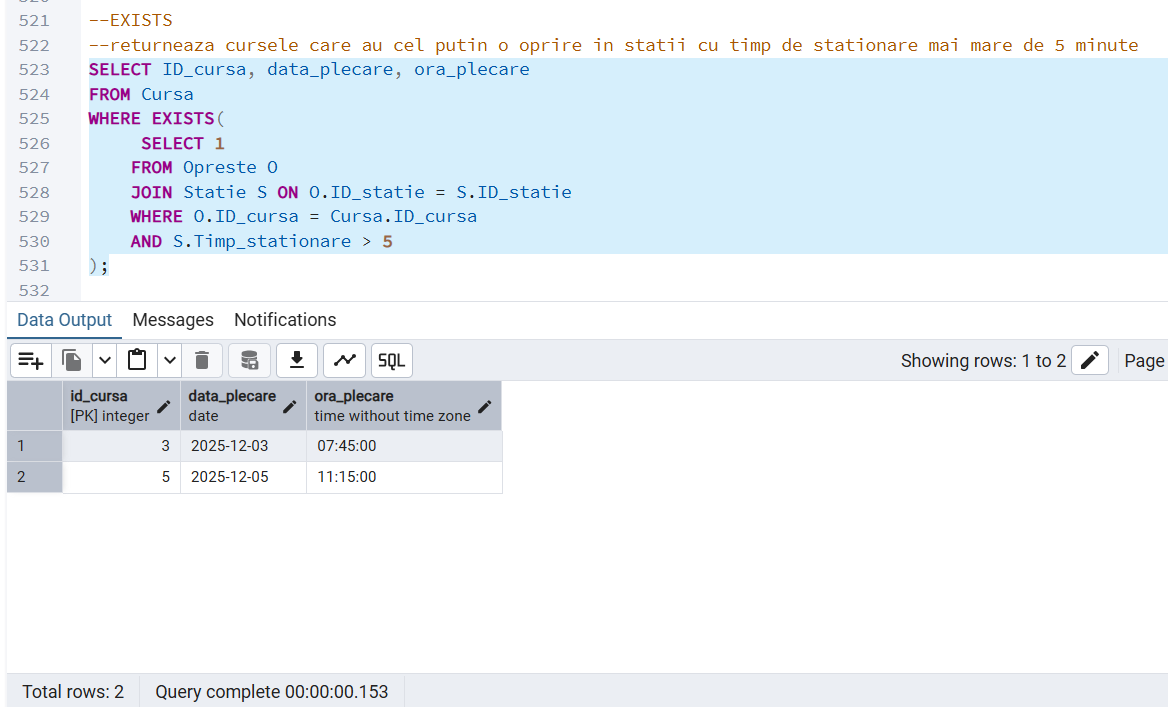


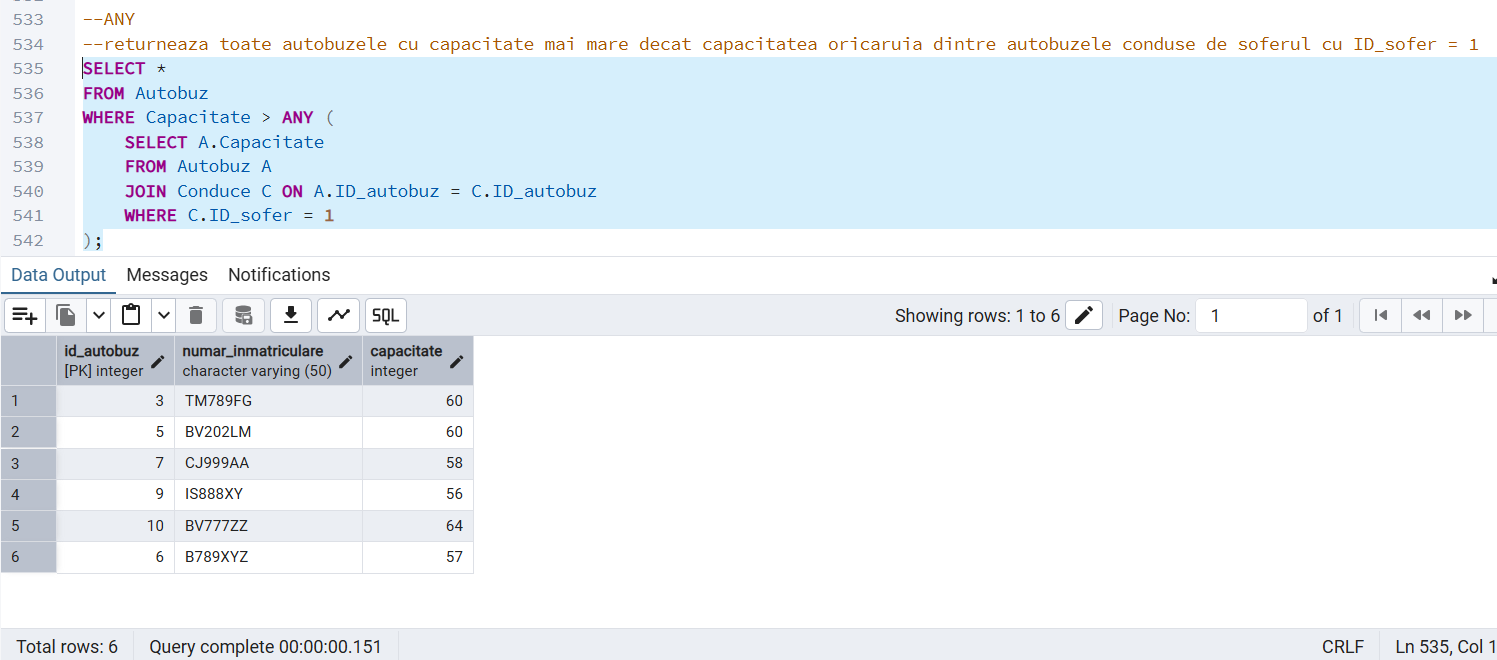


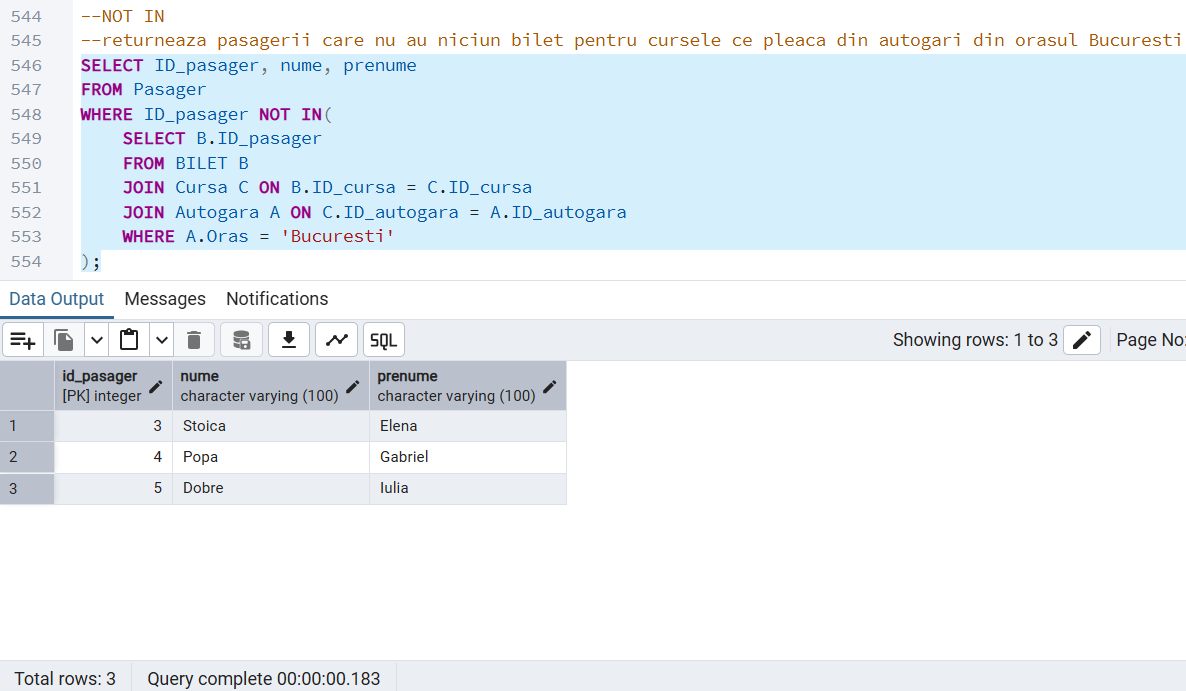
Afiseaza pasageri care au bilete la cel putin o cursa care are un timp de stationare de 5 minute.

ii. Subinterogari necorelate în care intervin cel puțin 3 tabele

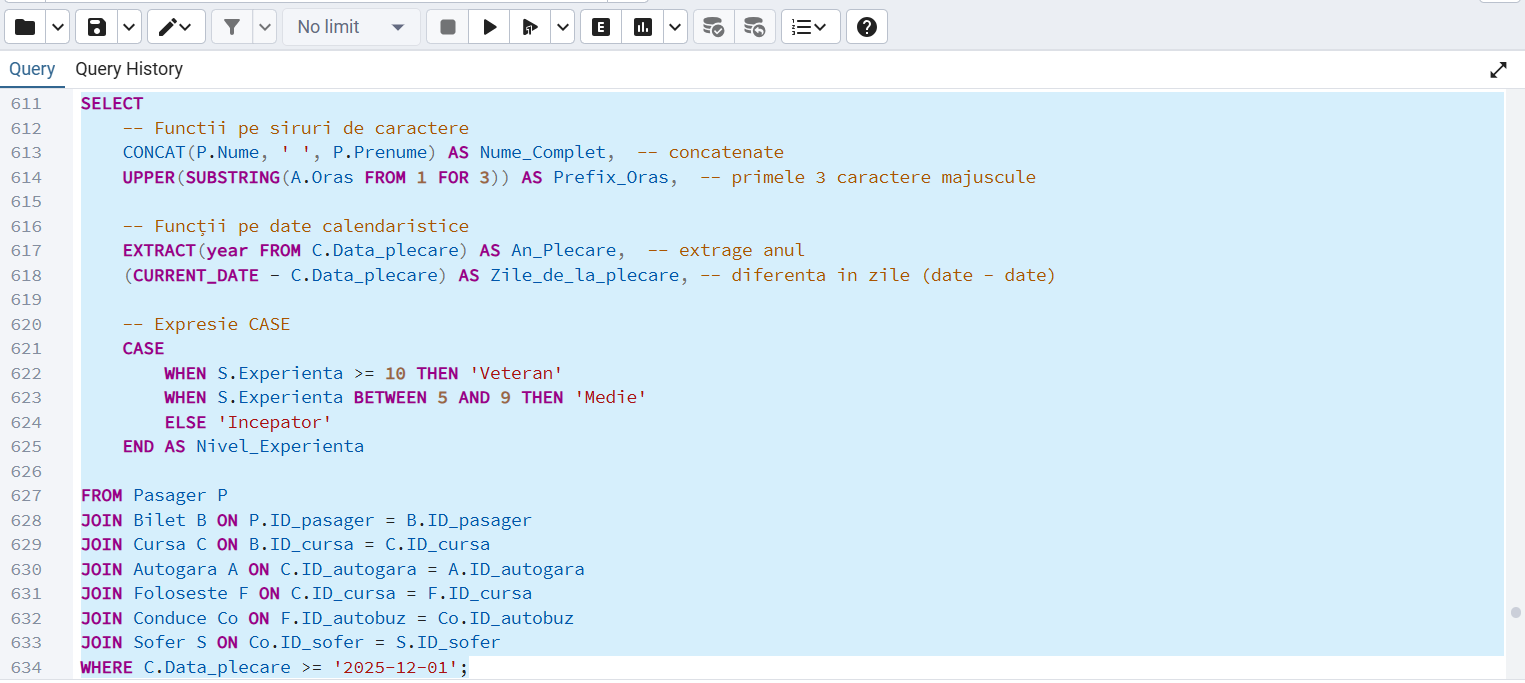








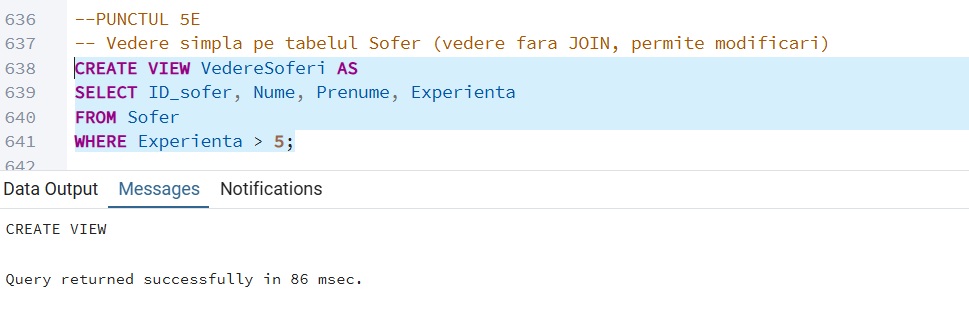
1. Utilizarea a cel puțin 2 funcții pe șiruri de caractere, 2 funcții pe date calendaristice, si a cel puțin unei expresii CASE.

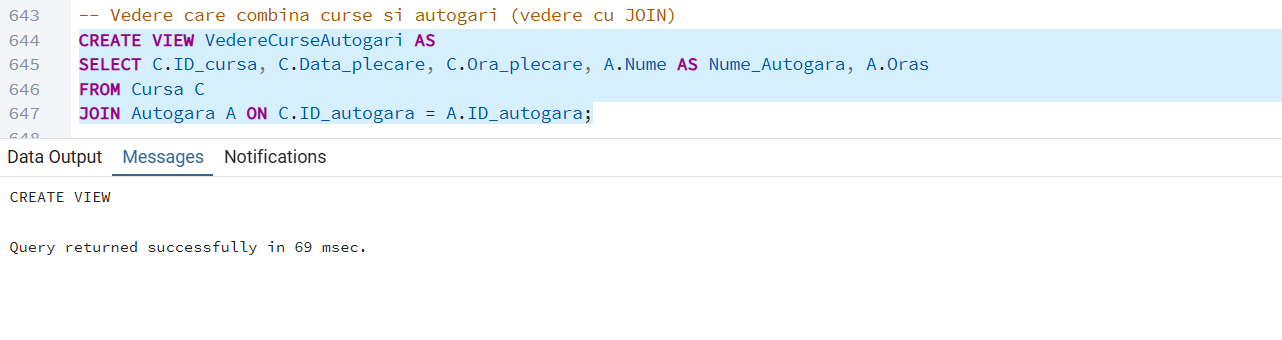




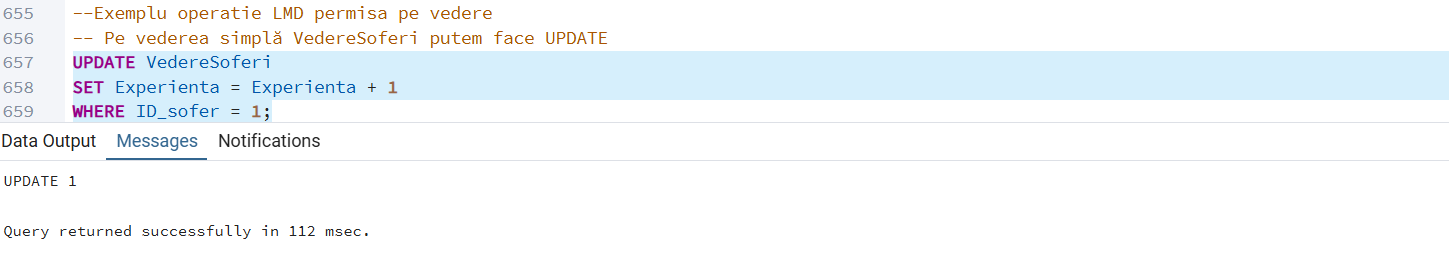
e. Minim 3 vederi.

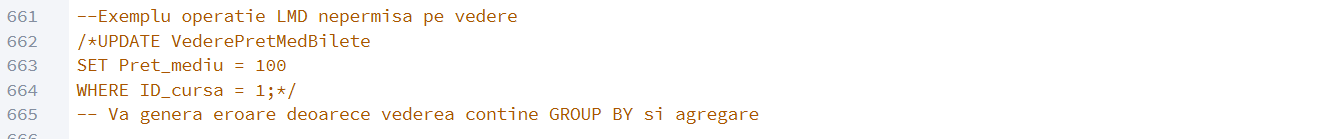
i. Dați un exemplu de operație LMD permisă pe vedere și un exemplu de operație LMD nepermisă.



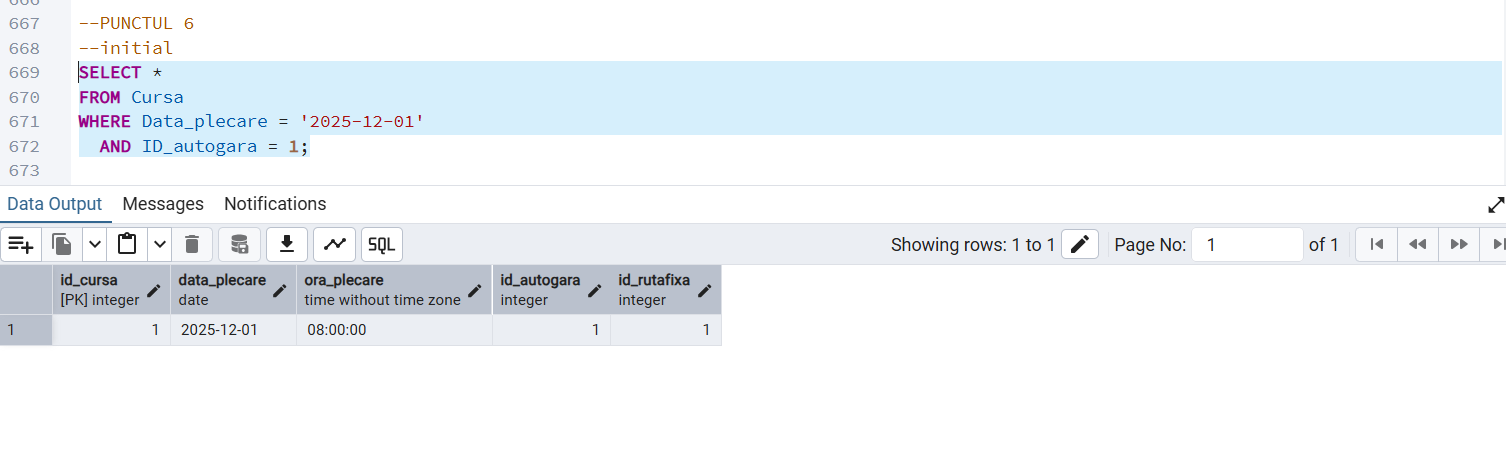


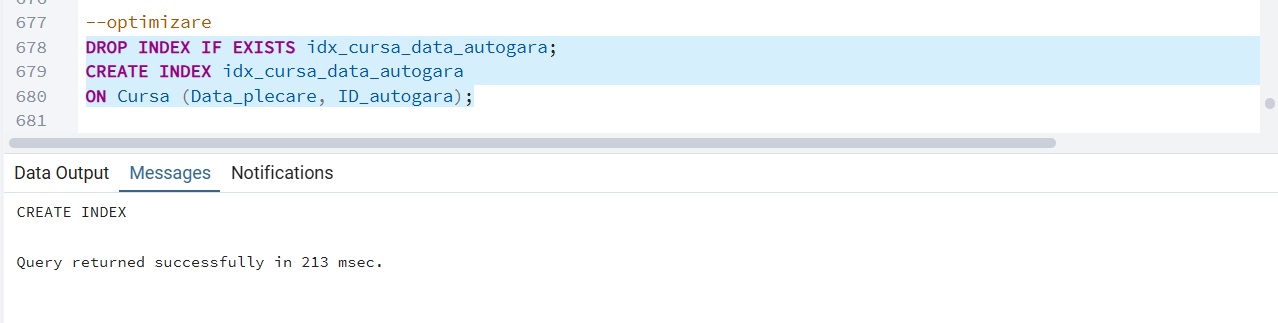






6. (0.5p)Crearea unui index care să optimizeze o interogare cu 2 criterii de cautare.





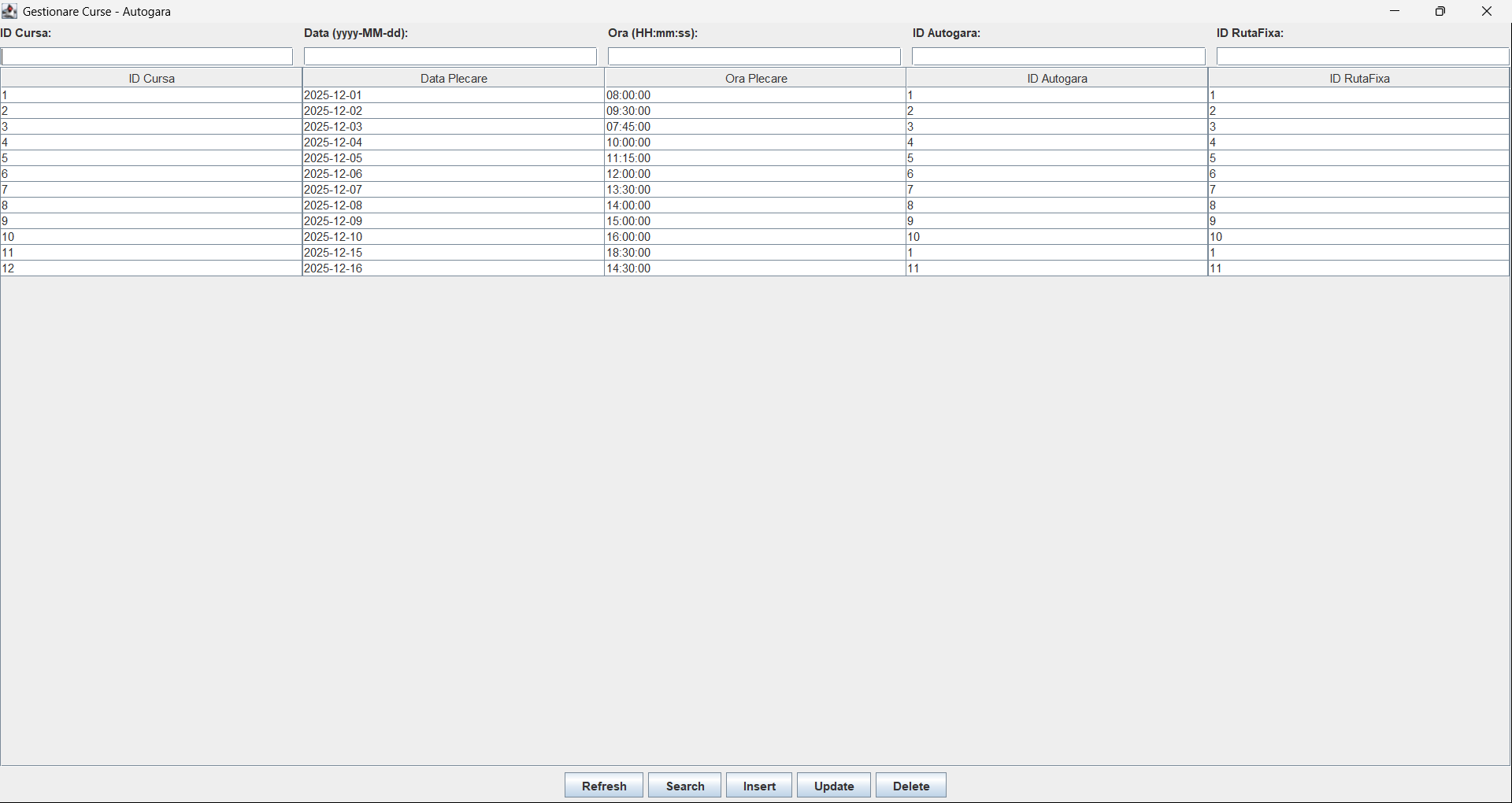
7. (1.5p)Implementarea unei aplicatii client care sa permita:

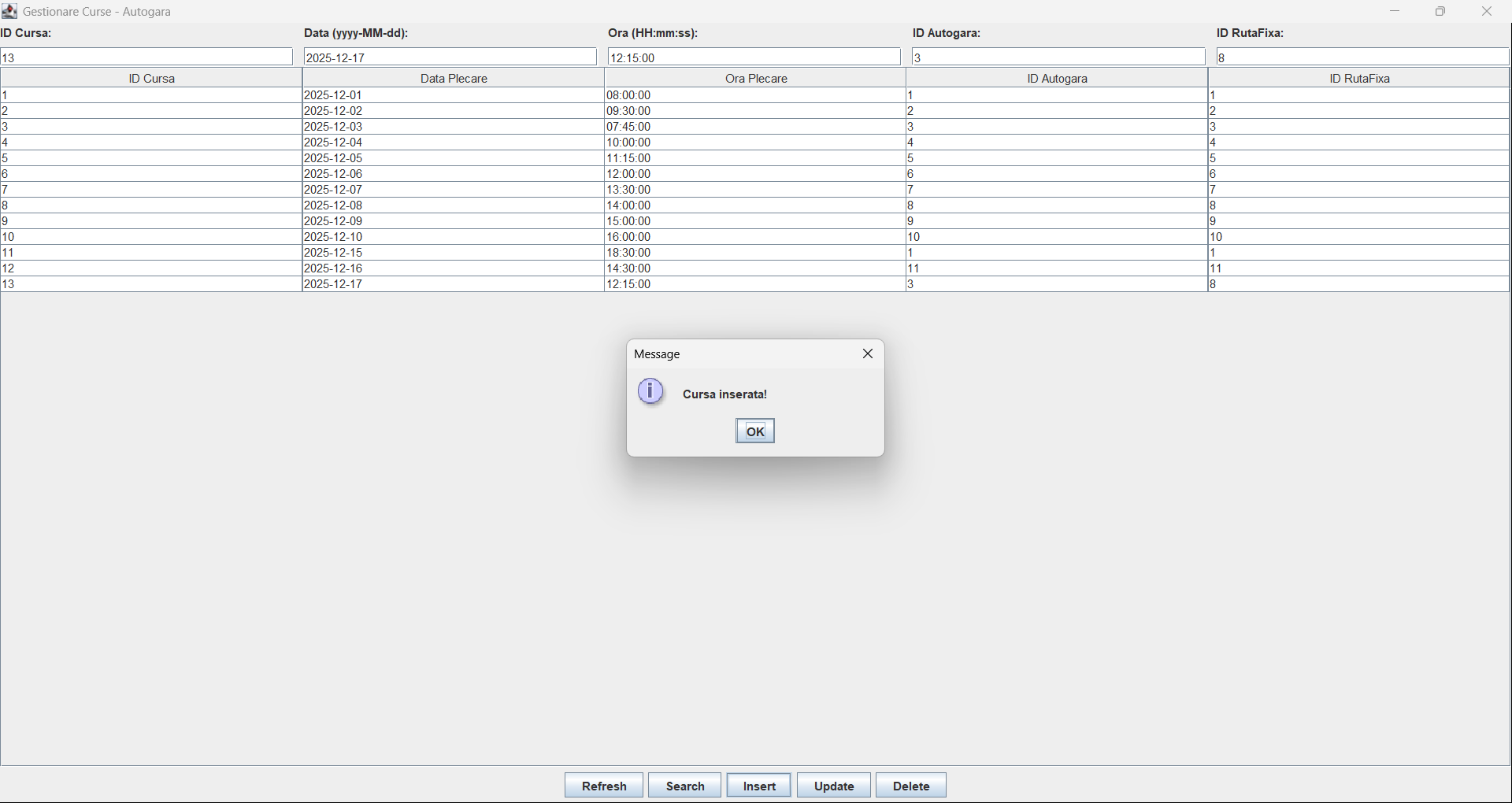
a. Operatii de manipulare a datelor in baza de date (update, insert, delete)

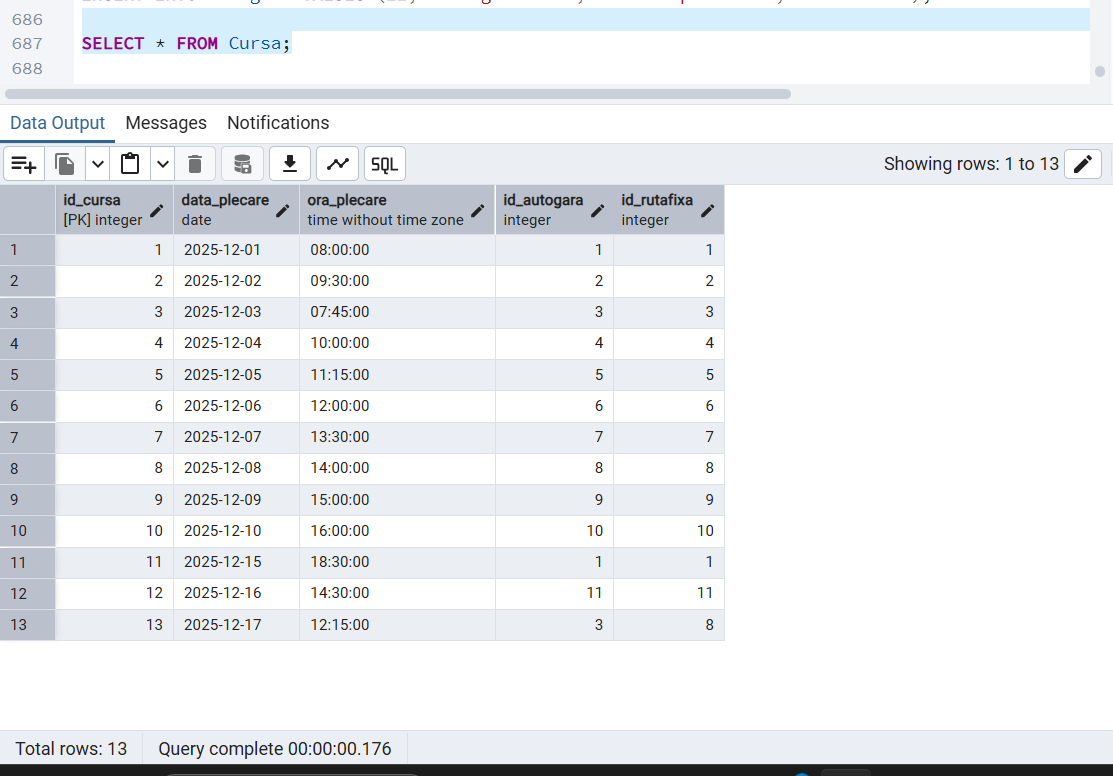
Am facut o interfata grafica pentru doua entitati: autogara si cursa.

**CURSA**

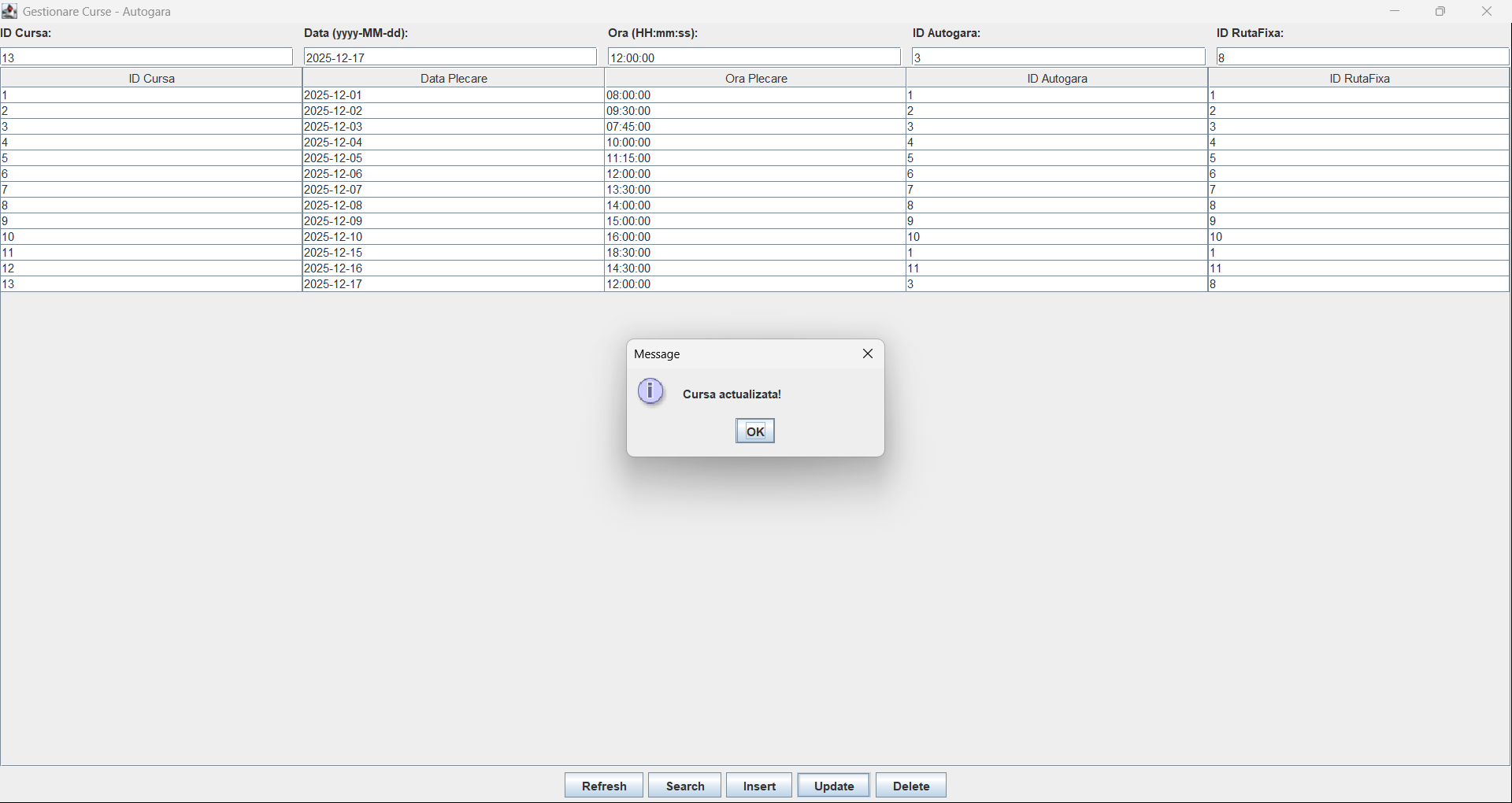
INSERT

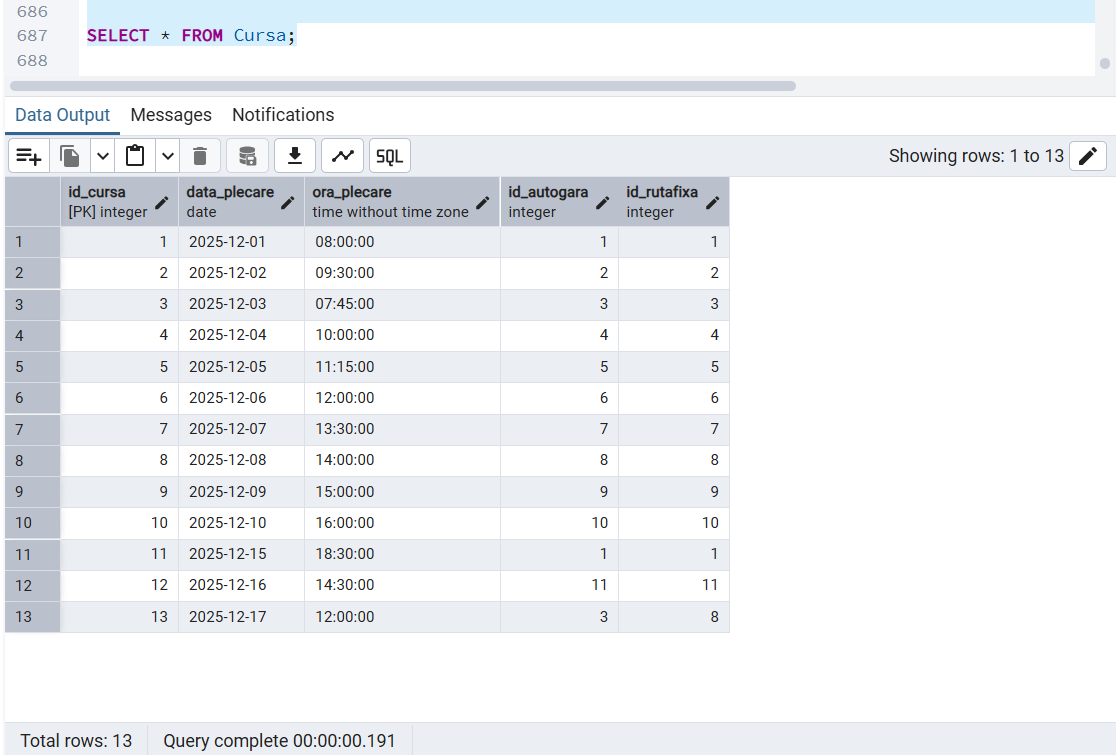




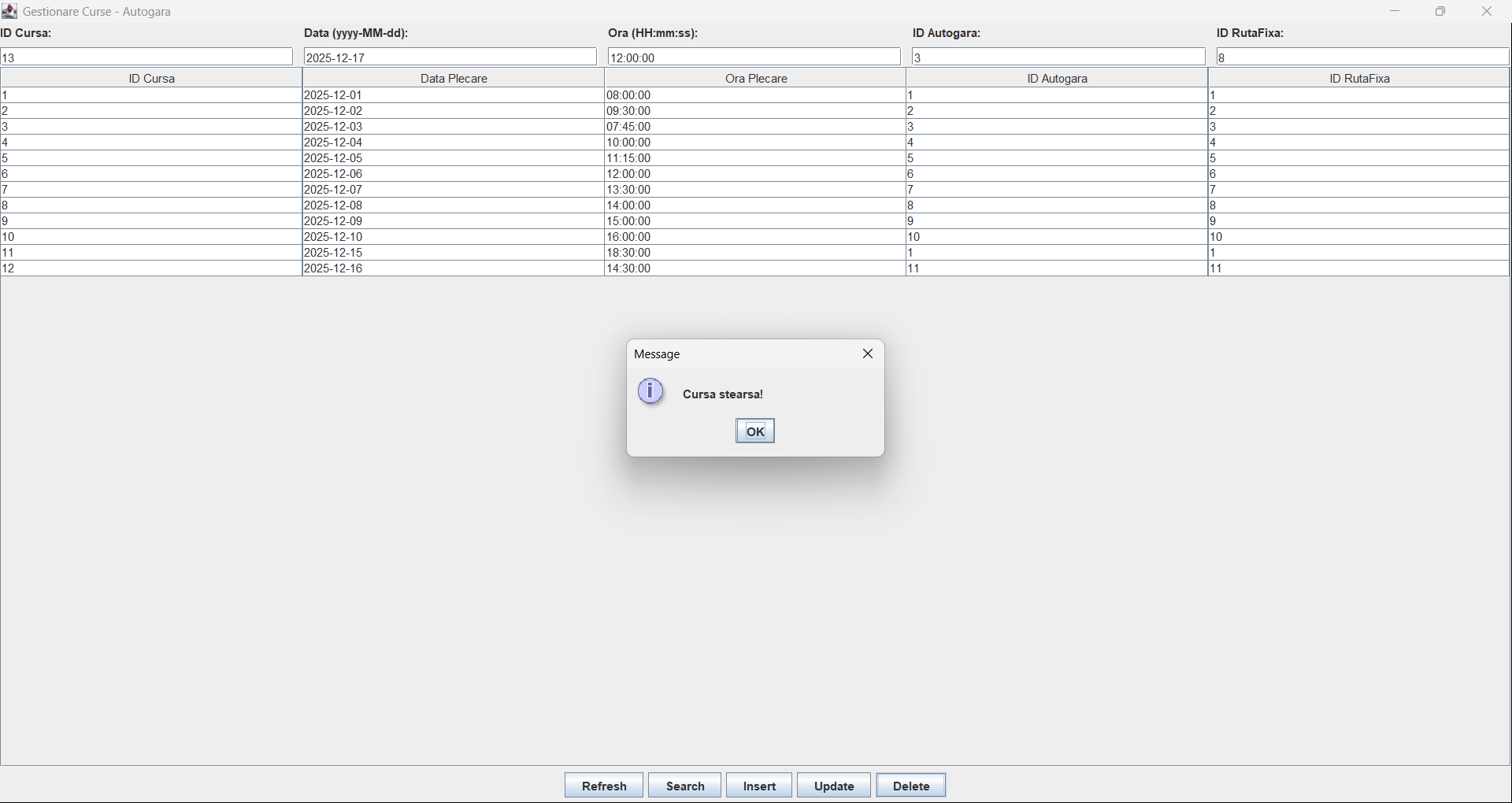


UPDATE - ora





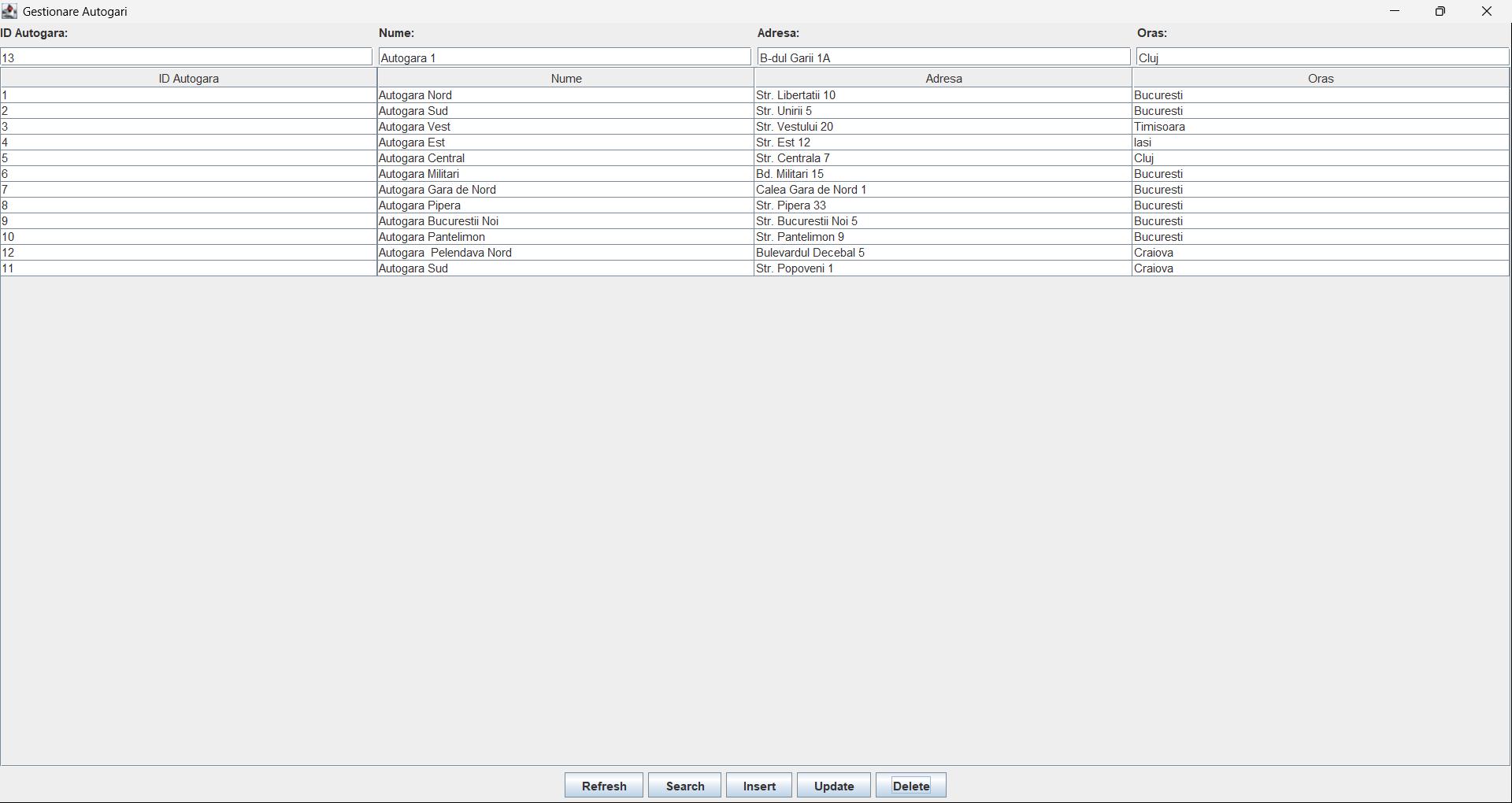
DELETE

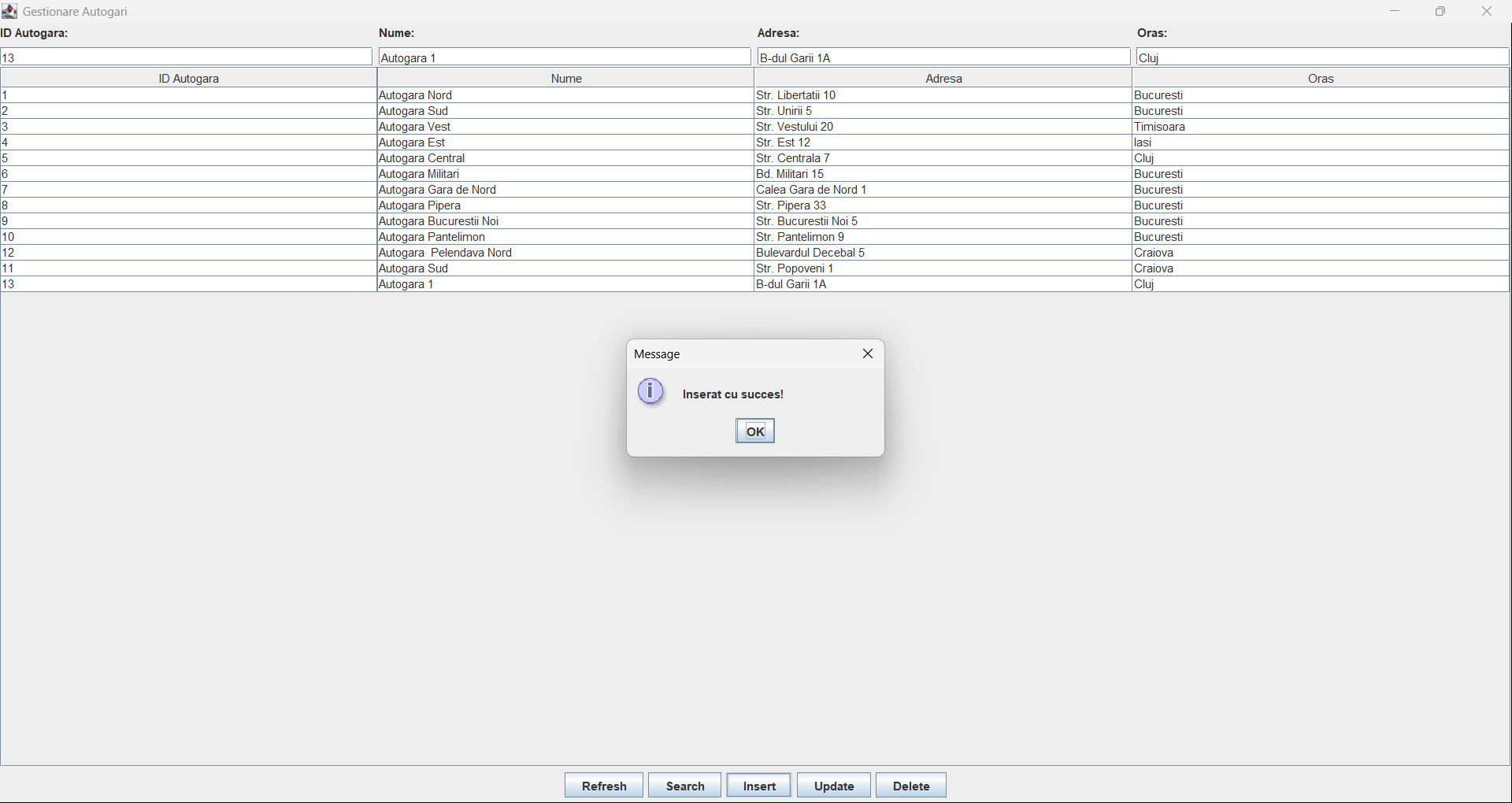


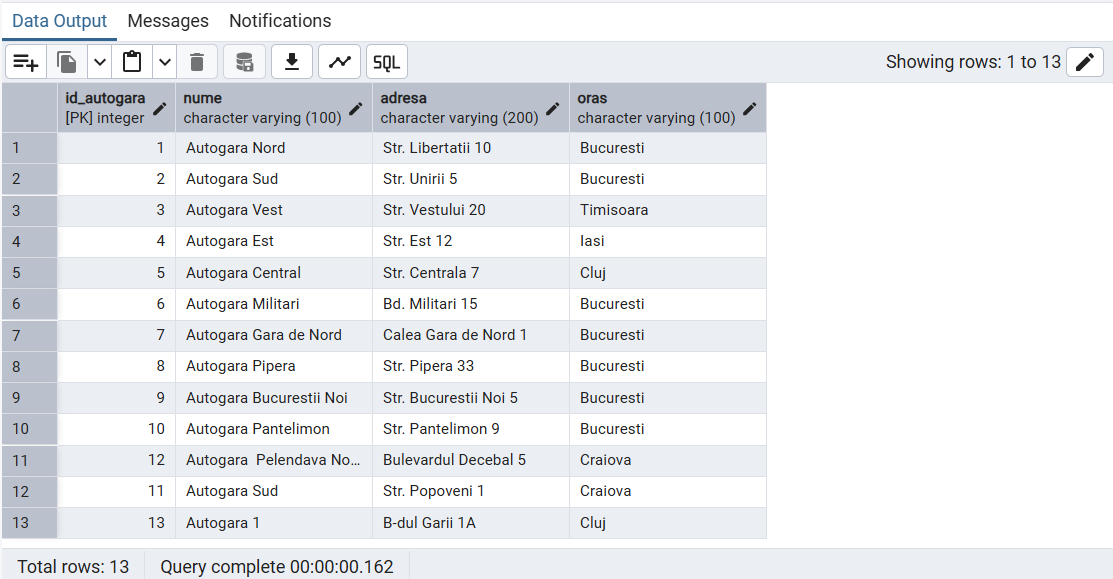


**AUTOGARA**

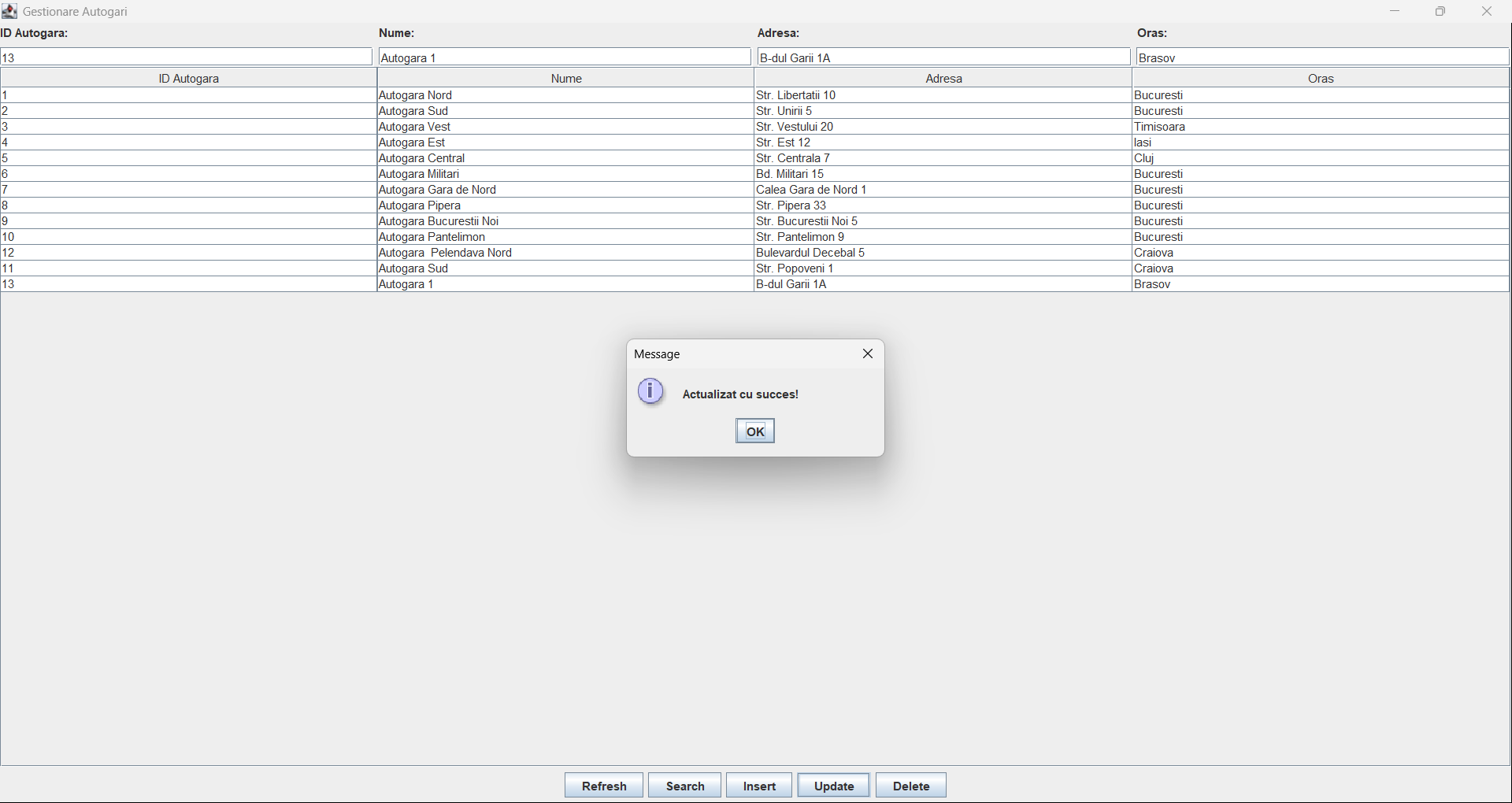
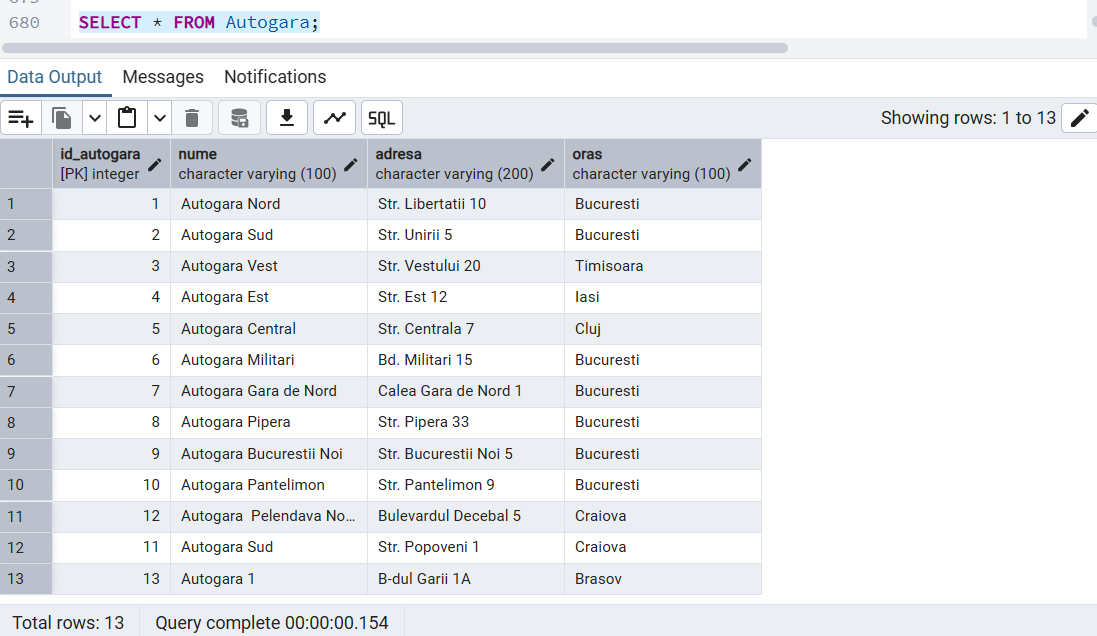
INSERT



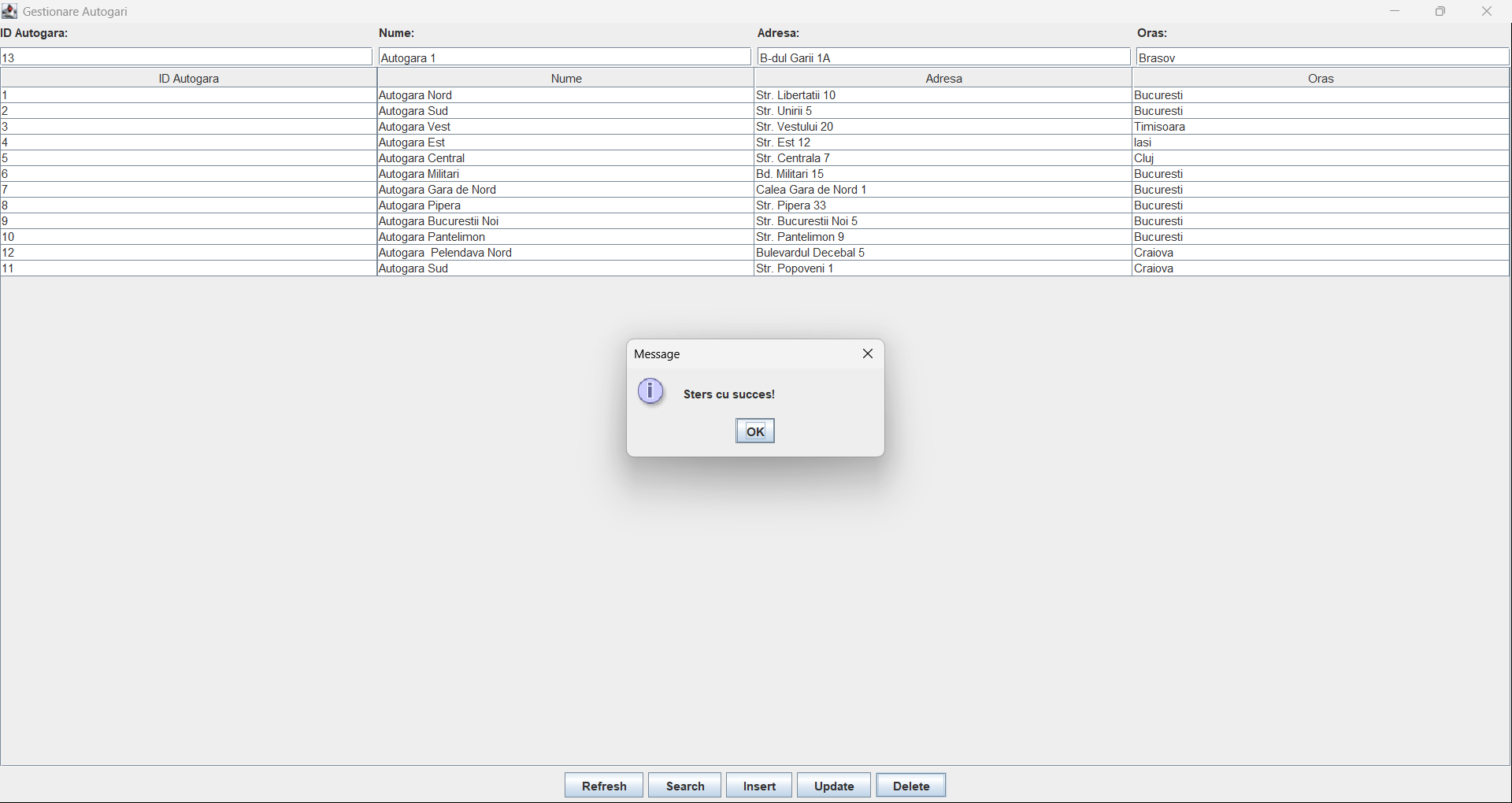


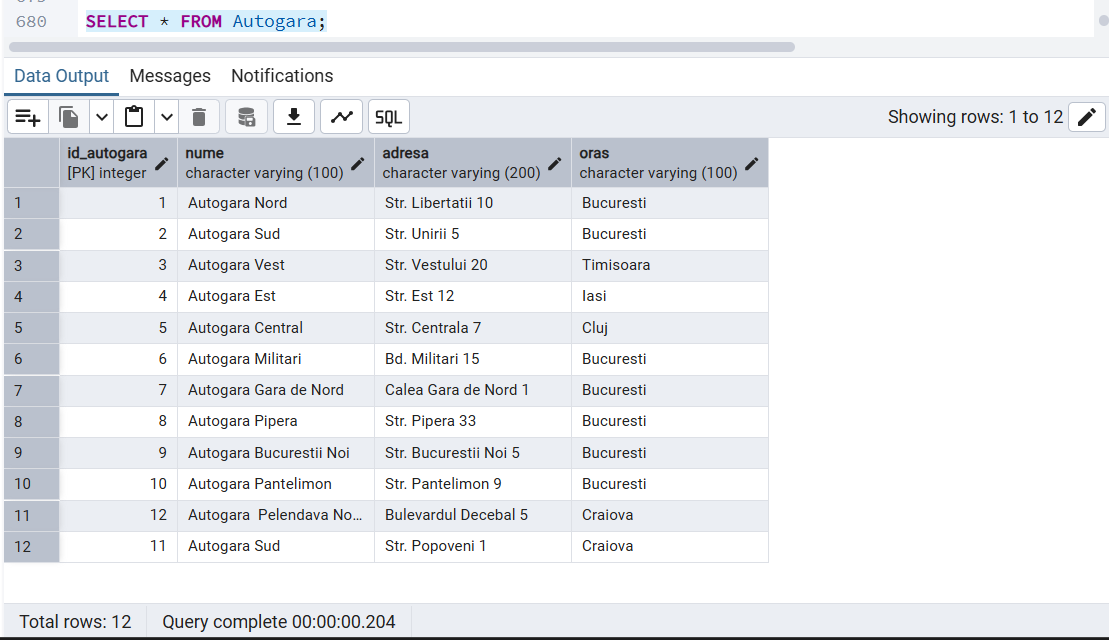


UPDATE – oras

* 
* 

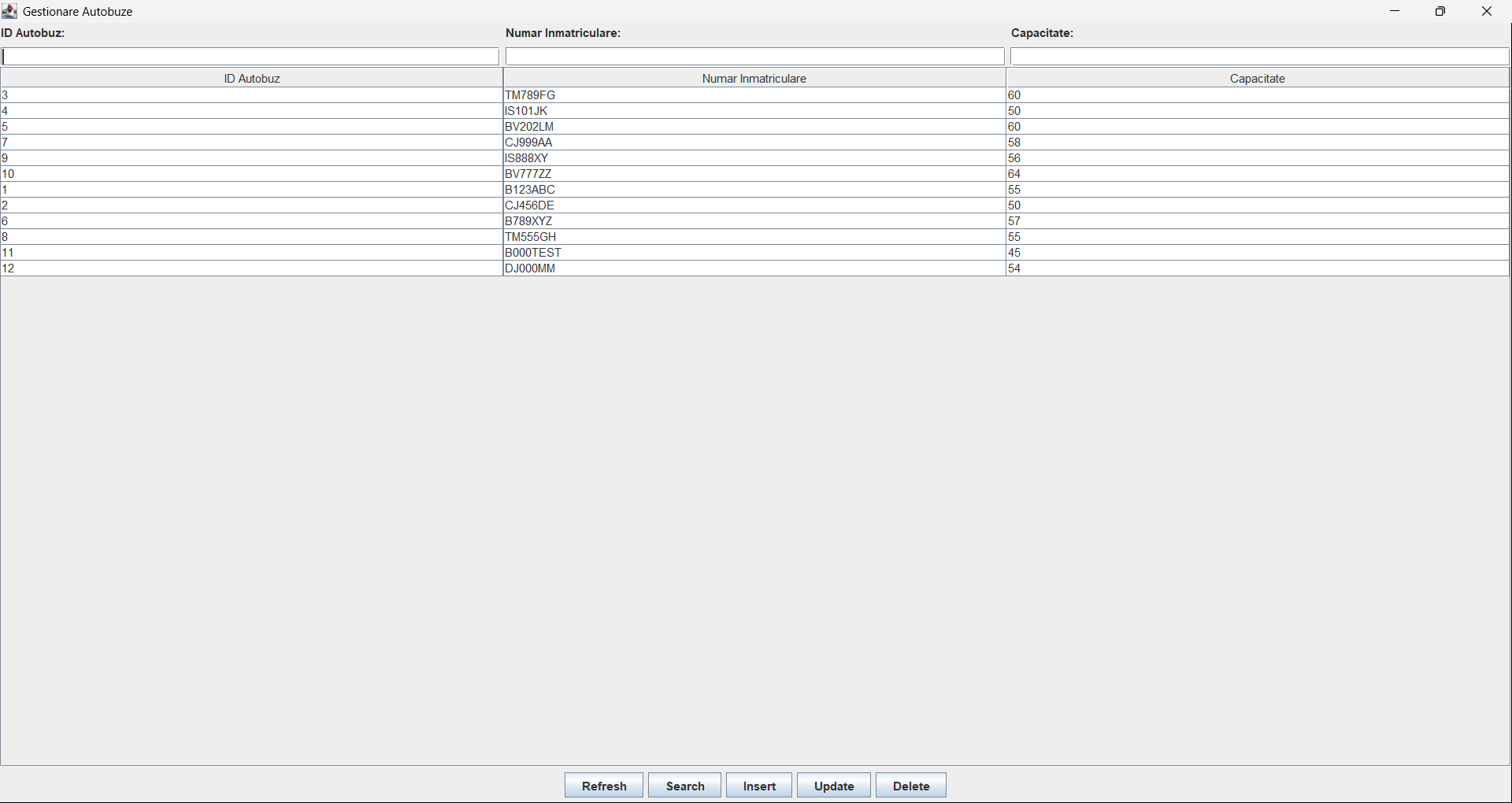
DELETE

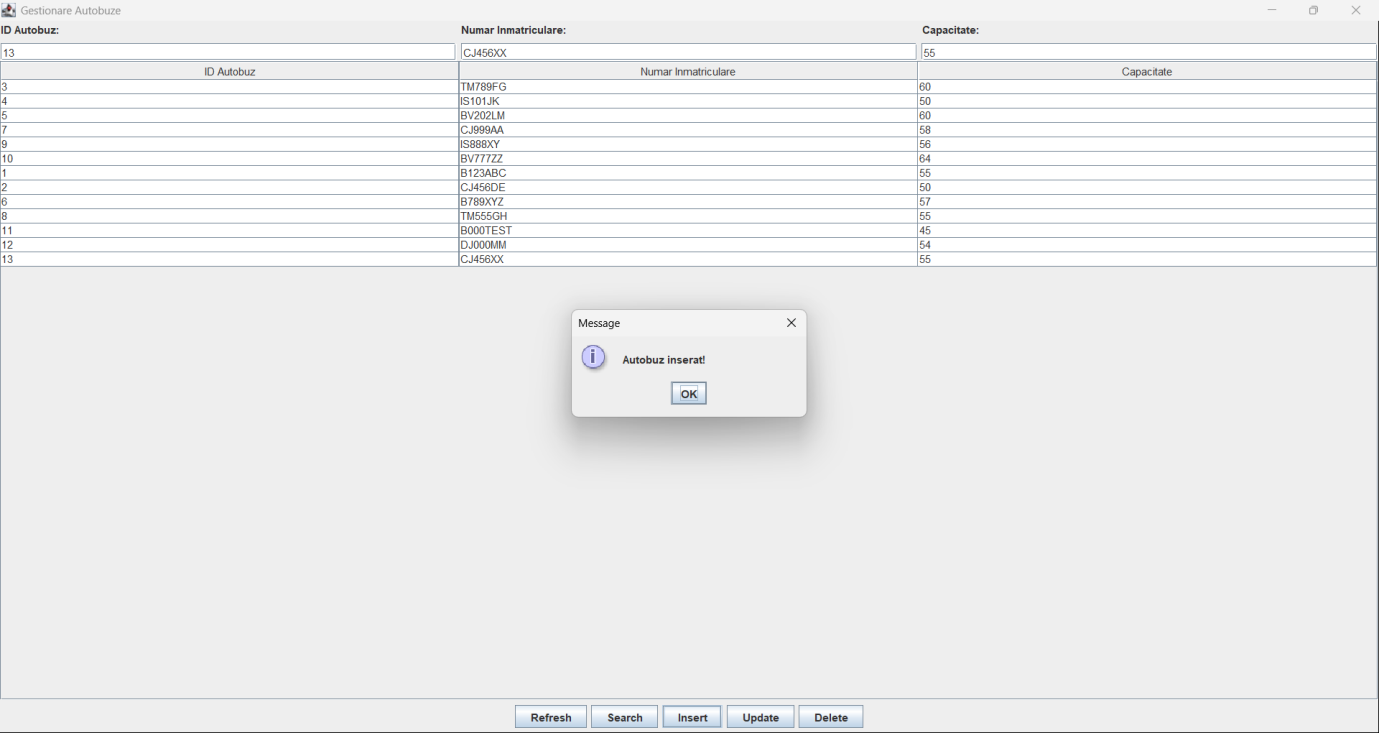


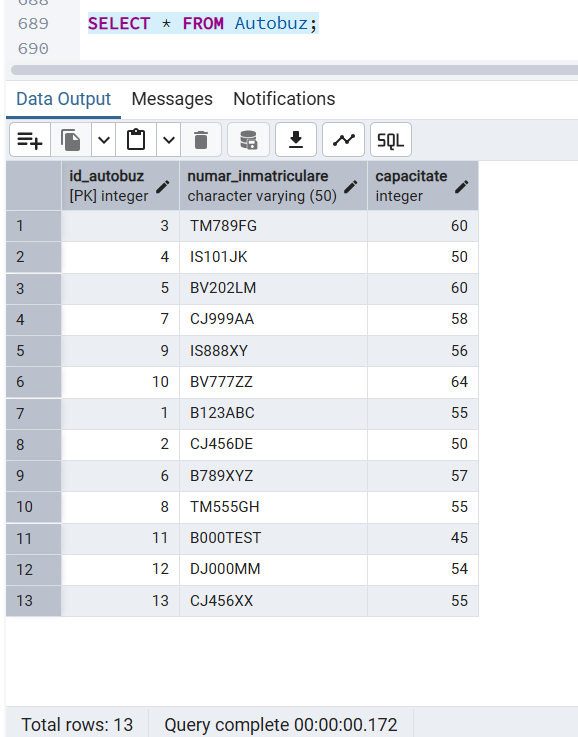


**AUTOBUZ**

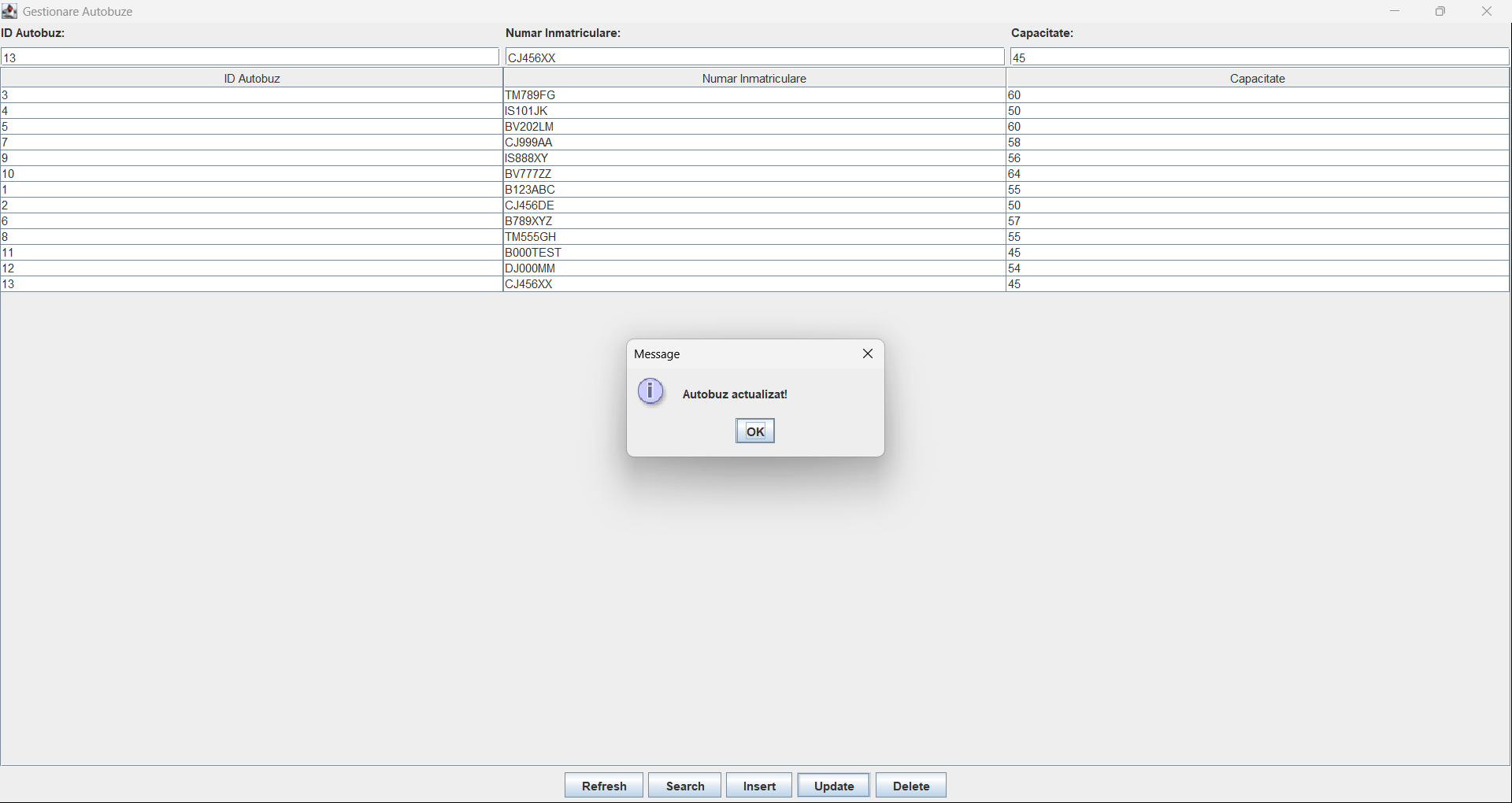
INSERT

****

****

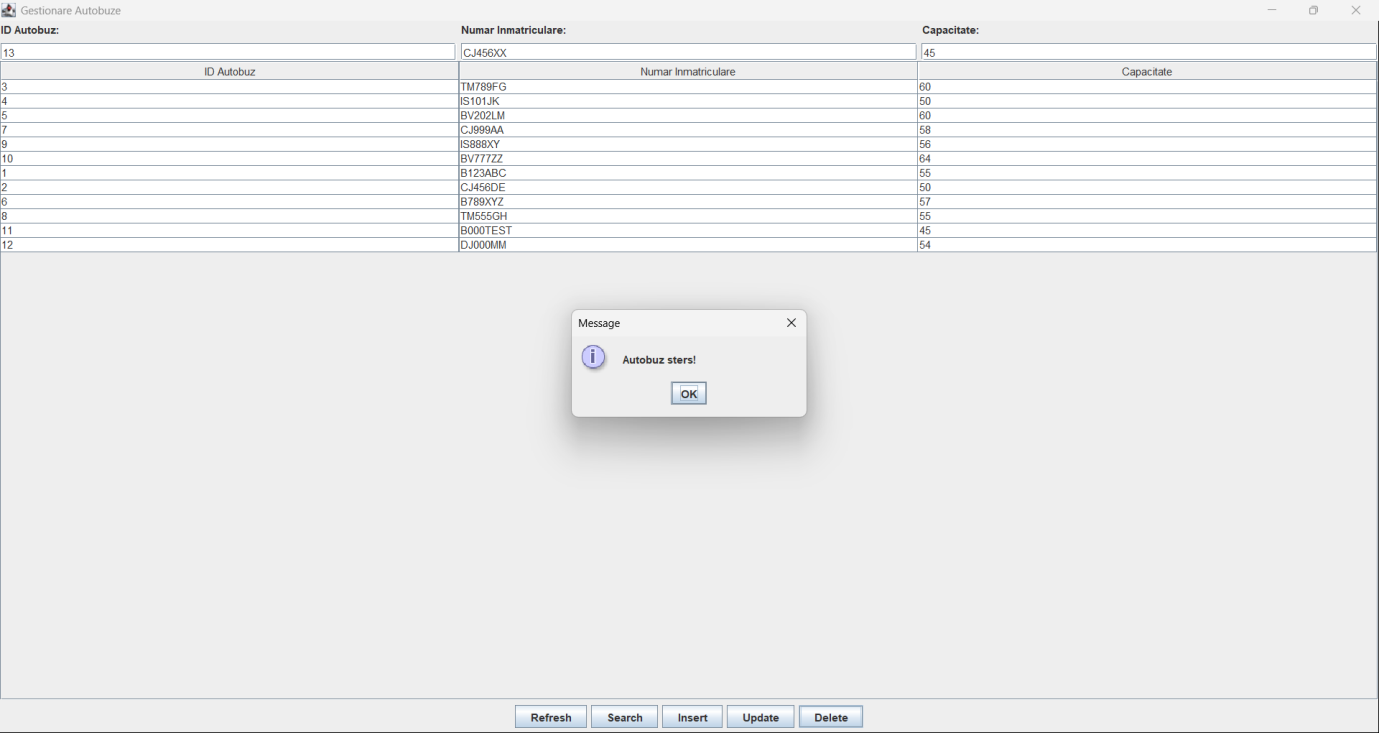
****

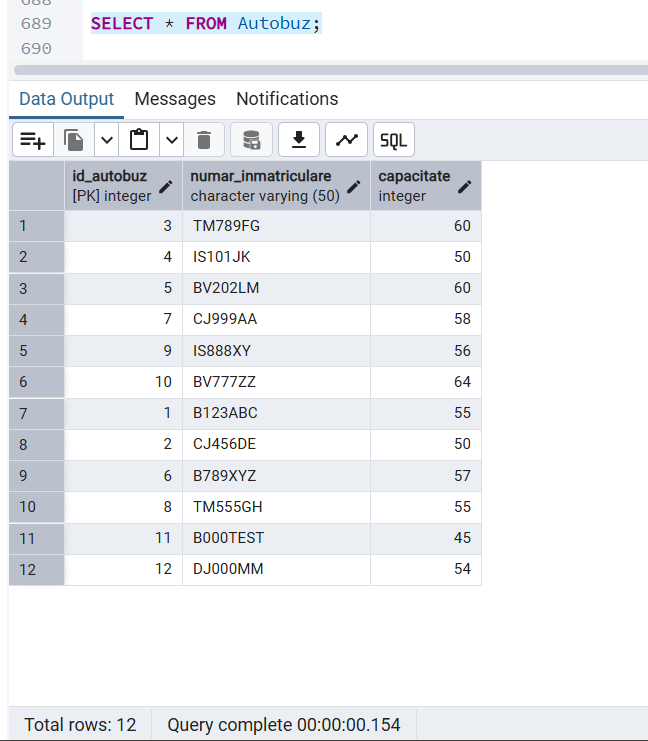
UPDATE – capacitate



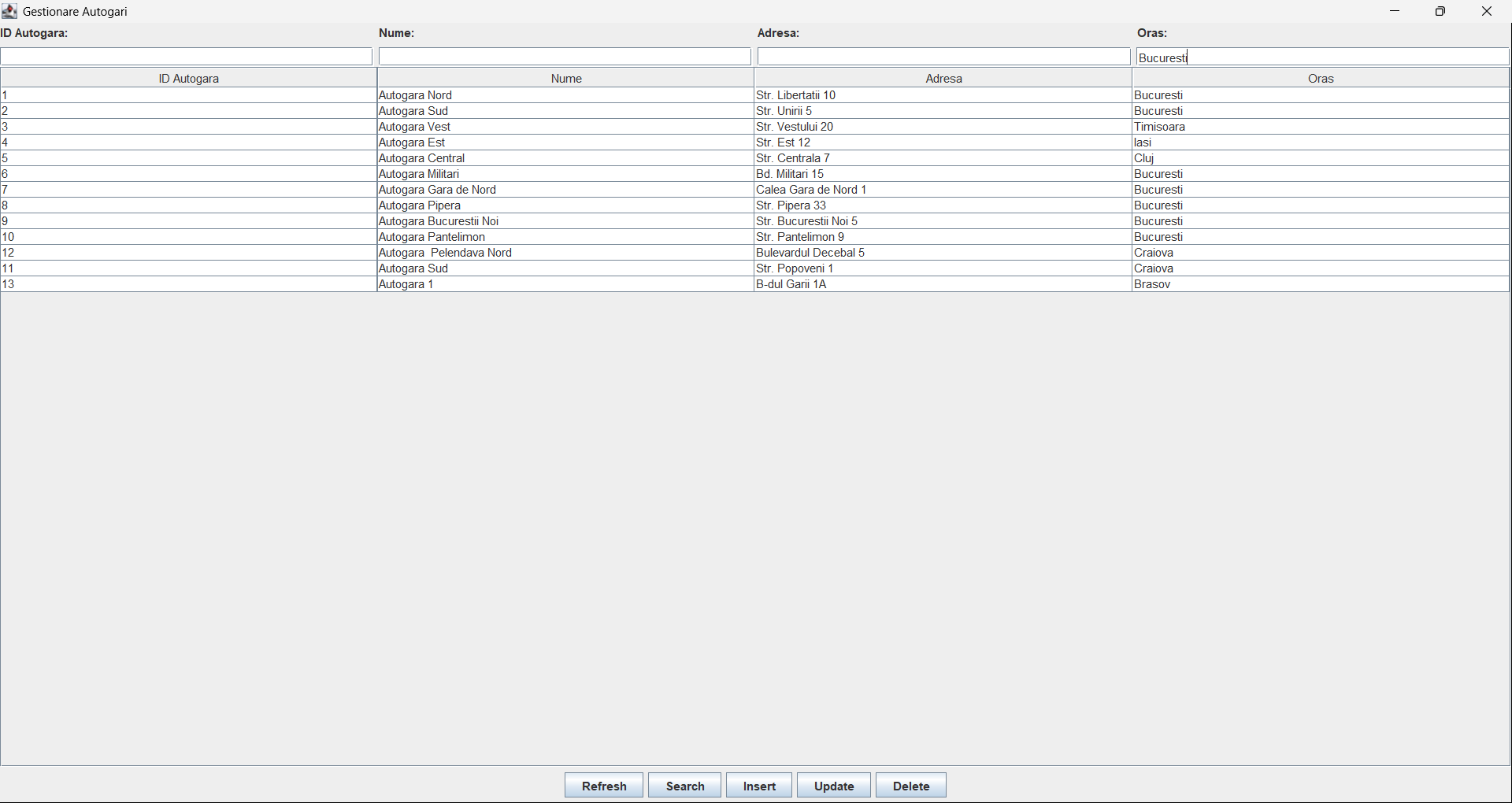


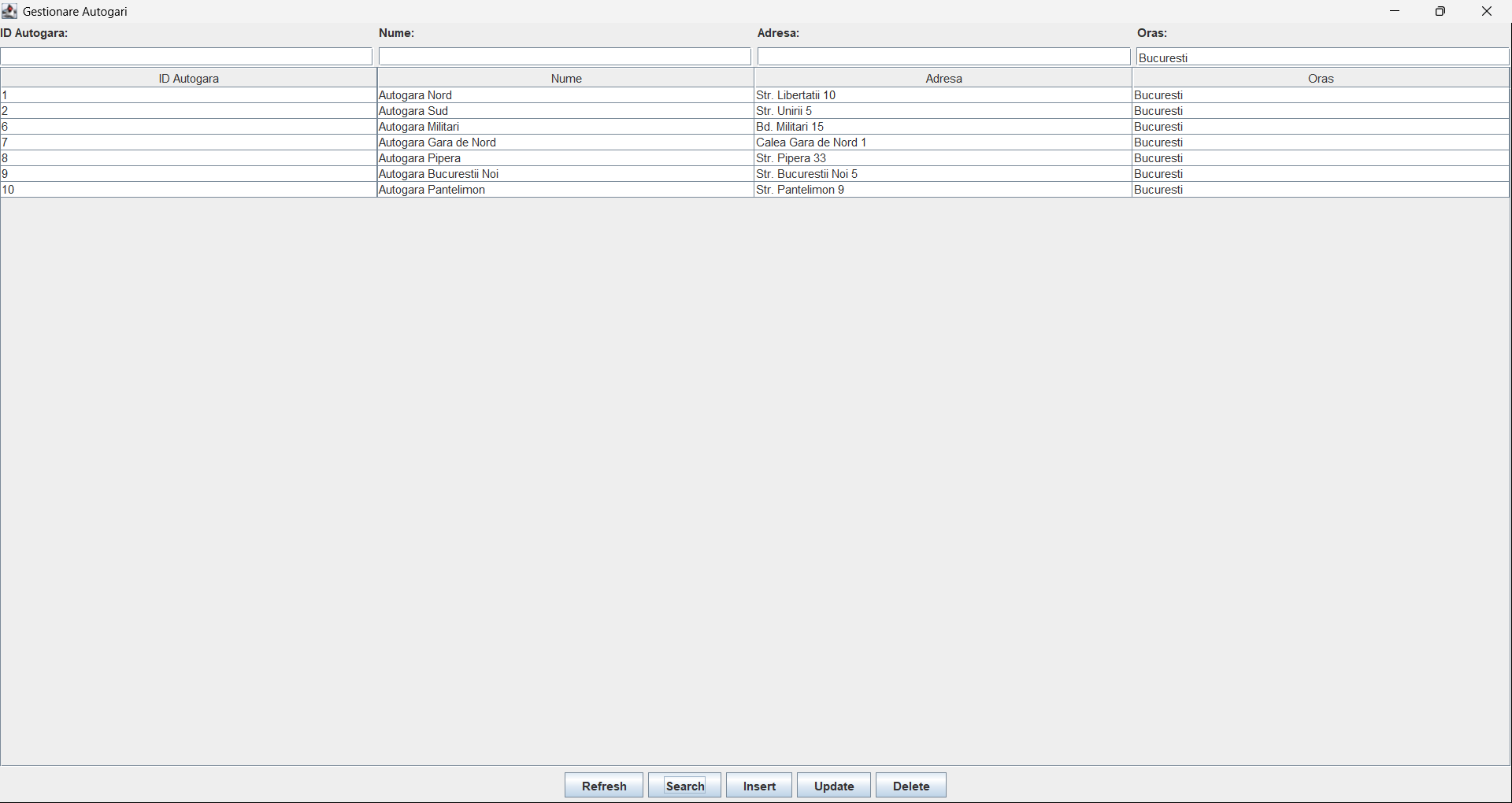
DELETE



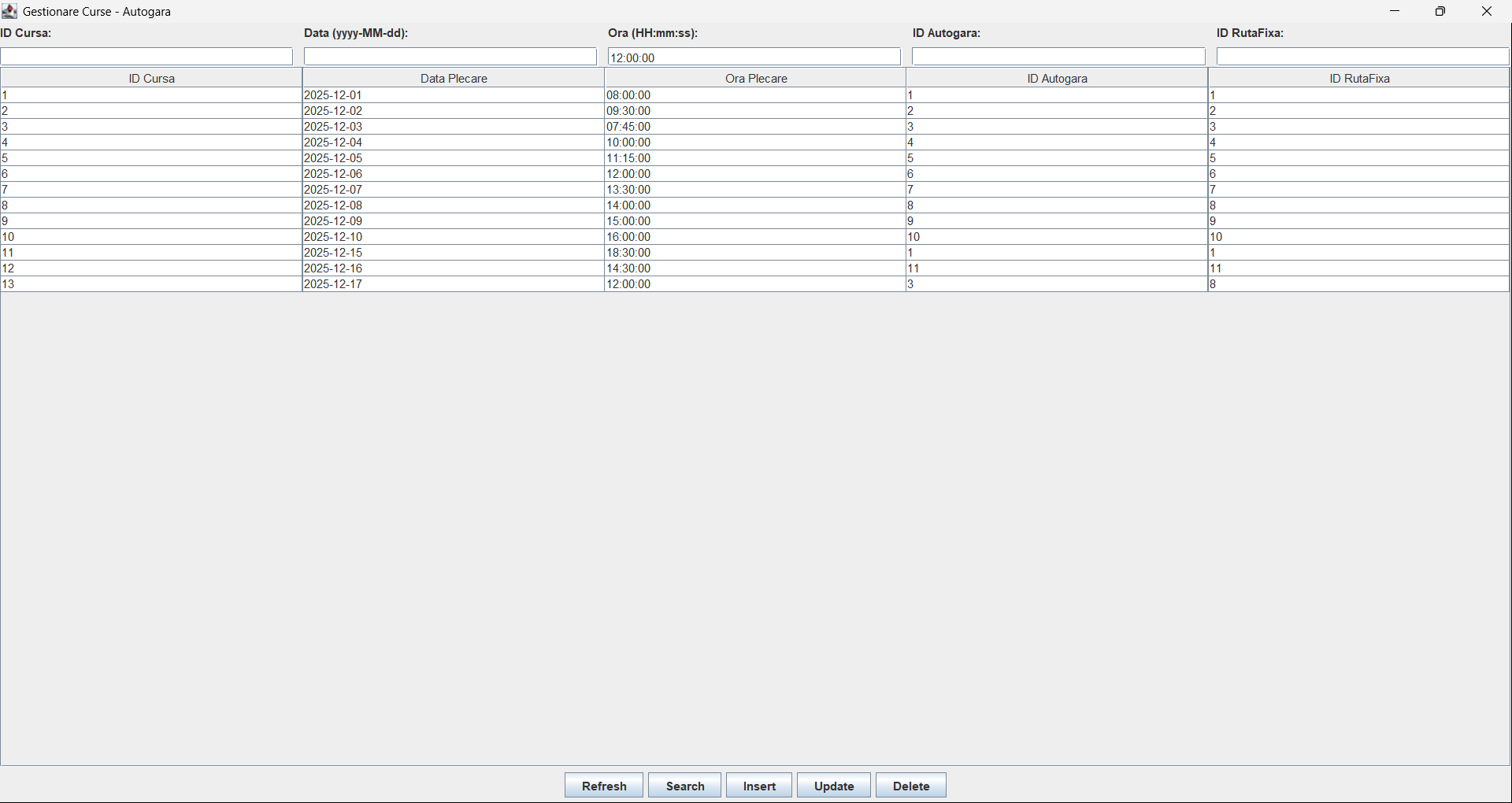


b. Operatii de cautare (dupa unul sau mai multe criterii) a datelor in baza de date

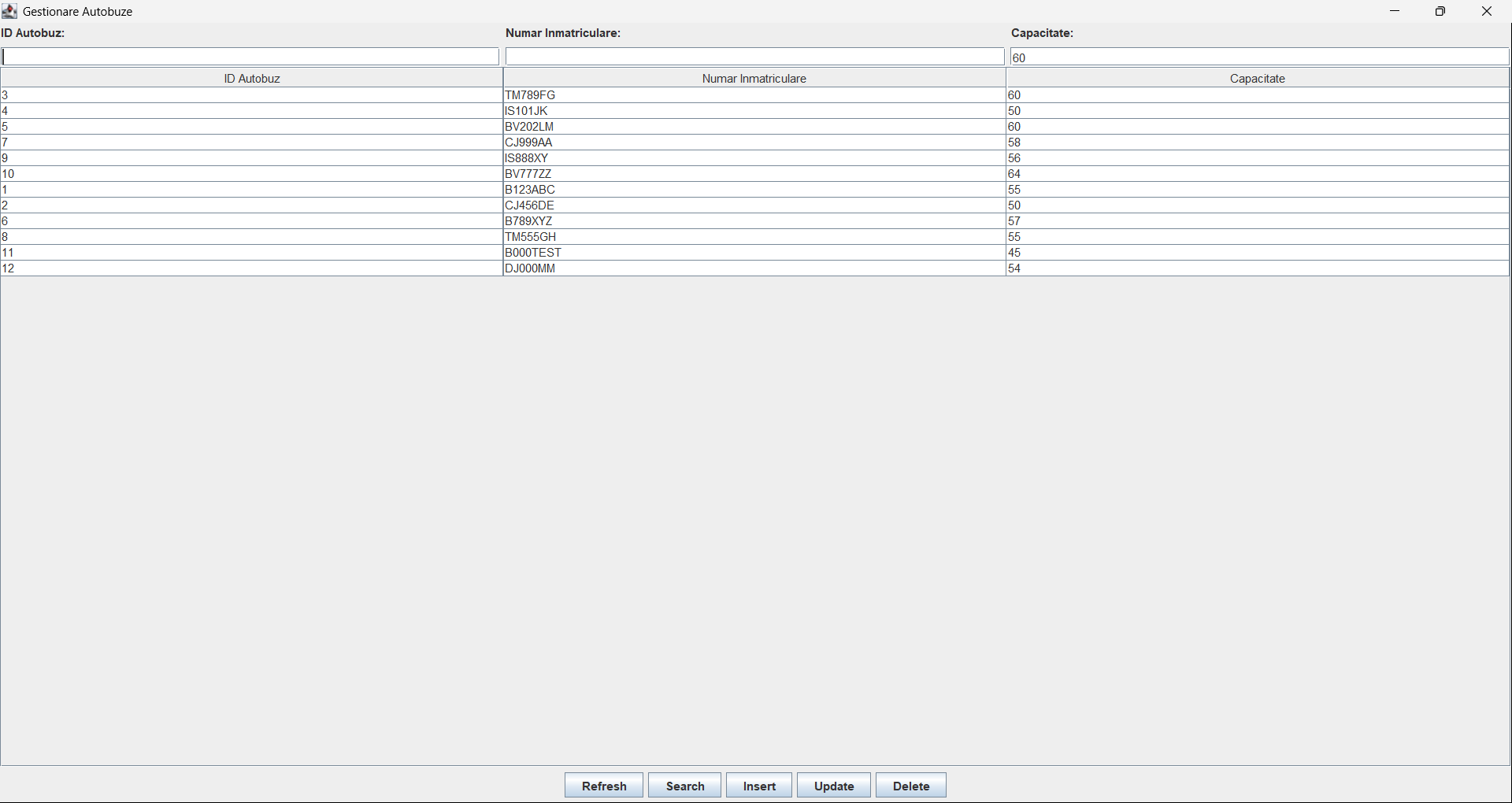
**Autogara – cautare dupa oras**  


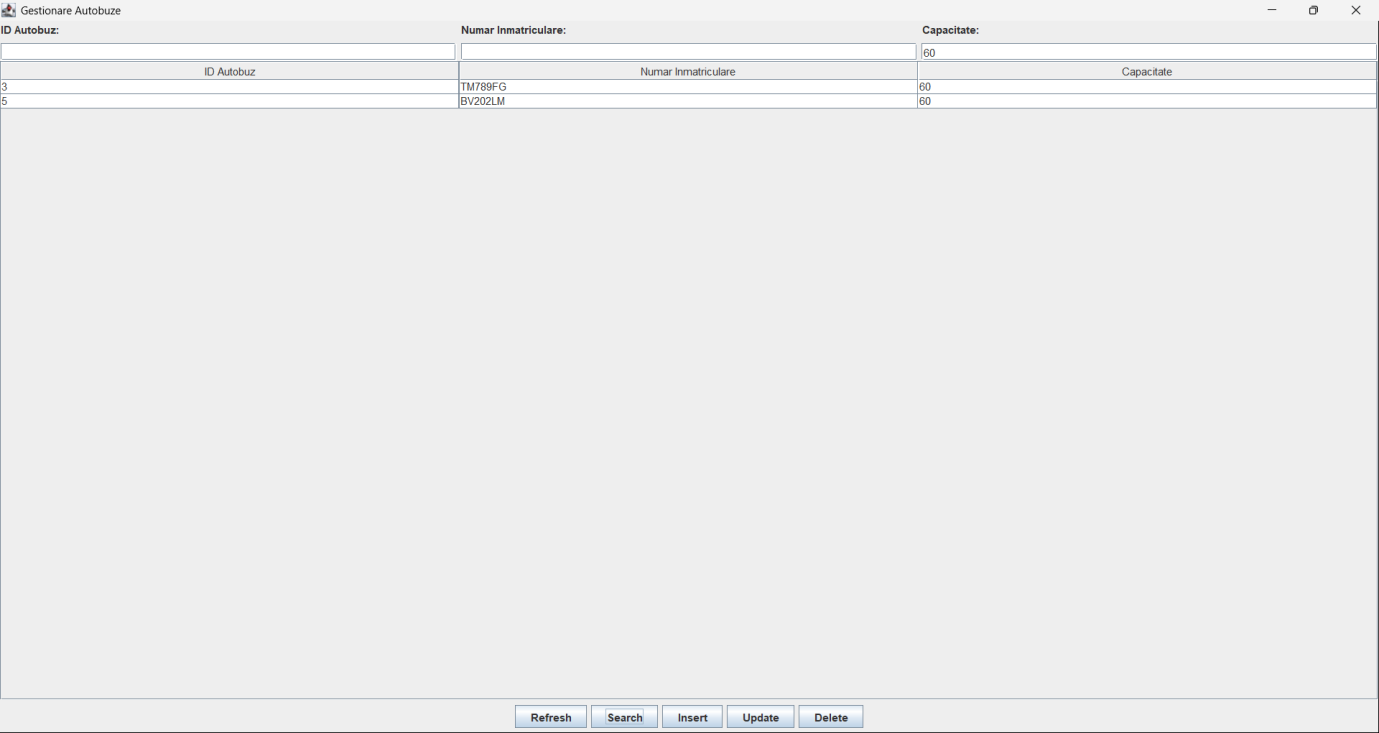


Cursa – cautare dupa data



**Autobuz – cautare dupa autobuz**





***Codul SQL al proiectului***

--crearea bazei de date

CREATE DATABASE ProiectAutogara;

--PUNCTUL 3

--tabel Autogara

CREATE TABLE Autogara (

ID\_autogara INT PRIMARY KEY,

Nume VARCHAR(100),

Adresa VARCHAR(200),

Oras VARCHAR(100)

);

--tabel RutaFixa

CREATE TABLE RutaFixa (

ID\_rutaFixa INT PRIMARY KEY,

Oras\_start VARCHAR(100),

Oras\_destinatie VARCHAR(100),

Distanta\_km INT

);

--tabel Sofer

CREATE TABLE Sofer (

ID\_sofer INT PRIMARY KEY,

Nume VARCHAR(100),

Prenume VARCHAR(100),

Nr\_permis VARCHAR(50),

Experienta INT

);

--tabel Autobuz

CREATE TABLE Autobuz (

ID\_autobuz INT PRIMARY KEY,

Numar\_inmatriculare VARCHAR(50),

Capacitate INT,

ID\_sofer INT,

FOREIGN KEY (ID\_sofer) REFERENCES Sofer(ID\_sofer)

);

--tabel Pasager

CREATE TABLE Pasager (

ID\_pasager INT PRIMARY KEY,

Nume VARCHAR(100),

Prenume VARCHAR(100)

);

--tabel Bilet

CREATE TABLE Bilet (

ID\_bilet INT PRIMARY KEY,

Pret DECIMAL(10,2),

Loc INT,

ID\_pasager INT,

FOREIGN KEY (ID\_pasager) REFERENCES Pasager(ID\_pasager)

);

--tabel Statie

CREATE TABLE Statie (

ID\_statie INT PRIMARY KEY,

Nume VARCHAR(100),

Timp\_stationare INT

);

--tabel Cursa

CREATE TABLE Cursa (

ID\_cursa INT PRIMARY KEY,

Data\_plecare DATE,

Ora\_plecare TIME,

ID\_autogara INT,

ID\_rutaFixa INT,

ID\_autobuz INT,

FOREIGN KEY (ID\_autogara) REFERENCES Autogara(ID\_autogara),

FOREIGN KEY (ID\_rutaFixa) REFERENCES RutaFixa(ID\_rutaFixa),

FOREIGN KEY (ID\_autobuz) REFERENCES Autobuz(ID\_autobuz)

);

DROP TABLE Bilet;

--corectarea tabelului Bilet

CREATE TABLE Bilet (

ID\_bilet INT PRIMARY KEY,

Pret DECIMAL(10,2),

Loc INT,

ID\_pasager INT,

ID\_cursa INT,

FOREIGN KEY (ID\_pasager) REFERENCES Pasager(ID\_pasager),

FOREIGN KEY (ID\_cursa) REFERENCES Cursa(ID\_cursa)

);

--Foloseste -> tabel de legatura intre Cursa si Autobuz

CREATE TABLE Foloseste (

ID\_are INT PRIMARY KEY,

ID\_cursa INT,

ID\_autobuz INT,

FOREIGN KEY (ID\_cursa) REFERENCES Cursa(ID\_cursa),

FOREIGN KEY (ID\_autobuz) REFERENCES Autobuz(ID\_autobuz)

);

--Opreste -> tabel de legatura intre Cursa si Statie

CREATE TABLE Opreste (

ID\_oprire INT PRIMARY KEY,

ID\_cursa INT,

ID\_statie INT,

Ordine\_oprire INT,

Timp\_stationare INT,

FOREIGN KEY (ID\_cursa) REFERENCES Cursa(ID\_cursa),

FOREIGN KEY (ID\_statie) REFERENCES Statie(ID\_statie)

);

--3A

--inserari in tabelul Autogara

INSERT INTO Autogara VALUES (1, 'Autogara Nord', 'Str. Libertatii 10', 'Bucuresti');

INSERT INTO Autogara VALUES (2, 'Autogara Sud', 'Str. Unirii 5', 'Bucuresti');

INSERT INTO Autogara VALUES (3, 'Autogara Vest', 'Str. Vestului 20', 'Timisoara');

INSERT INTO Autogara VALUES (4, 'Autogara Est', 'Str. Est 12', 'Iasi');

INSERT INTO Autogara VALUES (5, 'Autogara Central', 'Str. Centrala 7', 'Cluj');

INSERT INTO Autogara VALUES (6, 'Autogara Militari', 'Bd. Militari 15', 'Bucuresti');

INSERT INTO Autogara VALUES (7, 'Autogara Gara de Nord', 'Calea Gara de Nord 1', 'Bucuresti');

INSERT INTO Autogara VALUES (8, 'Autogara Pipera', 'Str. Pipera 33', 'Bucuresti');

INSERT INTO Autogara VALUES (9, 'Autogara Bucurestii Noi', 'Str. Bucurestii Noi 5', 'Bucuresti');

INSERT INTO Autogara VALUES (10, 'Autogara Pantelimon', 'Str. Pantelimon 9', 'Bucuresti');

--inserari in tabelul Sofer

INSERT INTO Sofer VALUES (1, 'Popescu', 'Ion', 'B123456', 10);

INSERT INTO Sofer VALUES (2, 'Ionescu', 'Maria', 'C654321', 7);

INSERT INTO Sofer VALUES (3, 'Georgescu', 'Andrei', 'D987654', 15);

INSERT INTO Sofer VALUES (4, 'Dumitru', 'Elena', 'E345678', 5);

INSERT INTO Sofer VALUES (5, 'Radu', 'Alexandru', 'F123987', 12);

INSERT INTO Sofer VALUES (6, 'Nedelcu', 'Vlad', 'G456789', 8);

INSERT INTO Sofer VALUES (7, 'Marin', 'Cristina', 'H987123', 6);

INSERT INTO Sofer VALUES (8, 'Constantin', 'Mihai', 'I234567', 9);

INSERT INTO Sofer VALUES (9, 'Florea', 'Ana', 'J876543', 4);

INSERT INTO Sofer VALUES (10, 'Matei', 'Bogdan', 'K123987', 11);

--inserari in tabelul Autobuz

INSERT INTO Autobuz VALUES (1, 'B123ABC', 50, 1);

INSERT INTO Autobuz VALUES (2, 'CJ456DE', 40, 2);

INSERT INTO Autobuz VALUES (3, 'TM789FG', 60, 3);

INSERT INTO Autobuz VALUES (4, 'IS101JK', 45, 4);

INSERT INTO Autobuz VALUES (5, 'BV202LM', 55, 5);

INSERT INTO Autobuz VALUES (6, 'B789XYZ', 52, 6);

INSERT INTO Autobuz VALUES (7, 'CJ999AA', 48, 7);

INSERT INTO Autobuz VALUES (8, 'TM555GH', 50, 8);

INSERT INTO Autobuz VALUES (9, 'IS888XY', 46, 9);

INSERT INTO Autobuz VALUES (10, 'BV777ZZ', 54, 10);

INSERT INTO Autobuz VALUES (11, 'B000TEST', 45);

--inserari in tabelul Pasager

INSERT INTO Pasager VALUES (1, 'Vasilescu', 'Ana');

INSERT INTO Pasager VALUES (2, 'Marinescu', 'Bogdan');

INSERT INTO Pasager VALUES (3, 'Stoica', 'Elena');

INSERT INTO Pasager VALUES (4, 'Popa', 'Gabriel');

INSERT INTO Pasager VALUES (5, 'Dobre', 'Iulia');

INSERT INTO Pasager VALUES (6, 'Tudor', 'Alina');

INSERT INTO Pasager VALUES (7, 'Barbu', 'Claudiu');

INSERT INTO Pasager VALUES (8, 'Neagu', 'Raluca');

INSERT INTO Pasager VALUES (9, 'Sandu', 'Cristian');

INSERT INTO Pasager VALUES (10, 'Lungu', 'Gabriela');

--inserari in tabelul RutaFixa

INSERT INTO RutaFixa VALUES (1, 'Bucuresti', 'Cluj', 450);

INSERT INTO RutaFixa VALUES (2, 'Timisoara', 'Iasi', 600);

INSERT INTO RutaFixa VALUES (3, 'Cluj', 'Timisoara', 300);

INSERT INTO RutaFixa VALUES (4, 'Bucuresti', 'Iasi', 400);

INSERT INTO RutaFixa VALUES (5, 'Cluj', 'Bucuresti', 450);

INSERT INTO RutaFixa VALUES (6, 'Bucuresti', 'Timisoara', 550);

INSERT INTO RutaFixa VALUES (7, 'Iasi', 'Cluj', 620);

INSERT INTO RutaFixa VALUES (8, 'Bucuresti', 'Brasov', 170);

INSERT INTO RutaFixa VALUES (9, 'Brasov', 'Cluj', 300);

INSERT INTO RutaFixa VALUES (10, 'Iasi', 'Bucuresti', 400);

--inserari in tabelul Statie

INSERT INTO Statie VALUES (1, 'Statia Nord', 5);

INSERT INTO Statie VALUES (2, 'Statia Sud', 4);

INSERT INTO Statie VALUES (3, 'Statia Vest', 6);

INSERT INTO Statie VALUES (4, 'Statia Est', 3);

INSERT INTO Statie VALUES (5, 'Statia Central', 7);

INSERT INTO Statie VALUES (6, 'Statia Militari', 5);

INSERT INTO Statie VALUES (7, 'Statia Gara de Nord', 4);

INSERT INTO Statie VALUES (8, 'Statia Pipera', 3);

INSERT INTO Statie VALUES (9, 'Statia Bucurestii Noi', 5);

INSERT INTO Statie VALUES (10, 'Statia Pantelimon', 4);

--inserari in tabelul Cursa

INSERT INTO Cursa VALUES (1, '2025-12-01', '08:00:00', 1, 1, 1);

INSERT INTO Cursa VALUES (2, '2025-12-02', '09:30:00', 2, 2, 2);

INSERT INTO Cursa VALUES (3, '2025-12-03', '07:45:00', 3, 3, 3);

INSERT INTO Cursa VALUES (4, '2025-12-04', '10:00:00', 4, 4, 4);

INSERT INTO Cursa VALUES (5, '2025-12-05', '11:15:00', 5, 5, 5);

INSERT INTO Cursa VALUES (6, '2025-12-06', '12:00:00', 6, 6, 6);

INSERT INTO Cursa VALUES (7, '2025-12-07', '13:30:00', 7, 7, 7);

INSERT INTO Cursa VALUES (8, '2025-12-08', '14:00:00', 8, 8, 8);

INSERT INTO Cursa VALUES (9, '2025-12-09', '15:00:00', 9, 9, 9);

INSERT INTO Cursa VALUES (10, '2025-12-10', '16:00:00', 10, 10, 10);

INSERT INTO Cursa VALUES (11, '2025-12-15', '18:30:00', 1, 1, NULL);

-- Bilet (10 înregistrări)

INSERT INTO Bilet VALUES (1, 50.00, 10, 1, 1);

INSERT INTO Bilet VALUES (2, 45.00, 15, 2, 2);

INSERT INTO Bilet VALUES (3, 60.00, 20, 3, 3);

INSERT INTO Bilet VALUES (4, 55.00, 25, 4, 4);

INSERT INTO Bilet VALUES (5, 40.00, 30, 5, 5);

INSERT INTO Bilet VALUES (6, 48.00, 12, 6, 6);

INSERT INTO Bilet VALUES (7, 42.00, 22, 7, 7);

INSERT INTO Bilet VALUES (8, 38.00, 18, 8, 8);

INSERT INTO Bilet VALUES (9, 50.00, 16, 9, 9);

INSERT INTO Bilet VALUES (10, 52.00, 14, 10, 10);

--foloseste

INSERT INTO Foloseste VALUES (1, 1, 1);

INSERT INTO Foloseste VALUES (2, 2, 2);

INSERT INTO Foloseste VALUES (3, 3, 3);

INSERT INTO Foloseste VALUES (4, 4, 4);

INSERT INTO Foloseste VALUES (5, 5, 5);

INSERT INTO Foloseste VALUES (6, 6, 6);

INSERT INTO Foloseste VALUES (7, 7, 7);

INSERT INTO Foloseste VALUES (8, 8, 8);

INSERT INTO Foloseste VALUES (9, 9, 9);

INSERT INTO Foloseste VALUES (10, 10, 10);

--opreste

INSERT INTO Opreste VALUES (1, 1, 1, 1, 5);

INSERT INTO Opreste VALUES (2, 1, 5, 2, 7);

INSERT INTO Opreste VALUES (3, 2, 2, 1, 4);

INSERT INTO Opreste VALUES (4, 2, 6, 2, 5);

INSERT INTO Opreste VALUES (5, 3, 3, 1, 6);

INSERT INTO Opreste VALUES (6, 3, 7, 2, 4);

INSERT INTO Opreste VALUES (7, 4, 4, 1, 3);

INSERT INTO Opreste VALUES (8, 4, 8, 2, 3);

INSERT INTO Opreste VALUES (9, 5, 5, 1, 7);

INSERT INTO Opreste VALUES (10, 5, 9, 2, 5);

--modificarea atributului nume din Statie in nume\_statie

ALTER TABLE Statie

RENAME COLUMN Nume TO Nume\_Statie;

--am modificat relatia Sofer-Autobuz=M:N, trebuie sters FK-ul ID\_sofer pus initial in Autobuz

ALTER TABLE Autobuz

DROP COLUMN ID\_sofer;

--Conduce -> tabel de legatura intre Sofer si Autobuz

CREATE TABLE Conduce(

ID\_conduce INT PRIMARY KEY,

ID\_sofer INT NOT NULL,

ID\_autobuz INT NOT NULL,

Data\_inceput DATE NOT NULL,

Data\_sfarsit DATE,

FOREIGN KEY (ID\_sofer) REFERENCES Sofer(ID\_sofer),

FOREIGN KEY (ID\_autobuz) REFERENCES Autobuz(ID\_autobuz)

);

INSERT INTO Conduce (ID\_conduce, ID\_sofer, ID\_autobuz, Data\_inceput, Data\_sfarsit) VALUES

(1, 1, 1, '2024-01-01', '2024-12-31'),

(2, 2, 2, '2024-03-15', NULL),

(3, 3, 3, '2024-05-10', '2024-08-30'),

(4, 4, 4, '2024-07-01', NULL),

(5, 5, 5, '2024-09-01', NULL),

(6, 6, 6, '2024-10-01', '2024-11-30'),

(7, 7, 7, '2024-01-20', NULL),

(8, 8, 8, '2024-02-10', '2024-05-15'),

(9, 9, 9, '2024-04-01', NULL),

(10, 10, 10, '2024-06-01', NULL);

--3B

--update simplu, mareste experienta cu 1

UPDATE Sofer

SET Experienta = Experienta + 1

WHERE ID\_sofer = 1;

--delete simplu

DELETE FROM Bilet

WHERE ID\_pasager = 10;

DELETE FROM Pasager

WHERE ID\_pasager = 10;

--3C

--UPDATE capacitatea cu 5

UPDATE Autobuz

SET Capacitate = Capacitate + 5

WHERE ID\_autobuz IN (

SELECT ID\_autobuz

FROM Conduce

WHERE Data\_sfarsit IS NULL --adica soferul conduce in continuare autobuzul respectiv

);

--UPDATE pentru cursele din ziua respectiva mareste pretul cu 10%

UPDATE Bilet

SET Pret = Pret \* 1.10

WHERE ID\_cursa IN (

SELECT ID\_cursa

FROM Cursa

WHERE Data\_plecare > '2025-12-05'

);

--UPDATE mareste pretul pentru cursele care au ca si oras Bucuresti

UPDATE Bilet

SET Pret = Pret + 10

WHERE ID\_cursa IN(

SELECT C.ID\_cursa

FROM Cursa as C

JOIN Autogara A ON C.ID\_autogara = A.ID\_autogara

WHERE A.Oras = 'Bucuresti'

);

--mareste experienta pentru soferii care conduc autobuze cu capacitatea > 50

UPDATE Sofer

Set Experienta =Experienta + 1

WHERE ID\_sofer IN(

SELECT ID\_sofer

FROM Conduce

WHERE ID\_autobuz IN (

SELECT ID\_autobuz

FROM Autobuz

WHERE Capacitate > 50

)

);

--UPDATE creste capacitatea autobuzelor care au ca si oras Bucuresti

UPDATE Autobuz

Set Capacitate = Capacitate + 5

WHERE ID\_autobuz IN(

SELECT ID\_autobuz

FROM Cursa

WHERE ID\_cursa IN (

SELECT C.ID\_cursa

FROM Cursa C

JOIN Autogara A ON C.ID\_autogara = A.ID\_autogara

WHERE A.Oras = 'Bucuresti'

)

);

--UPDATE timpul de stationare pentru cursa cu id-ul 1

UPDATE Opreste

SET Timp\_stationare = 3

WHERE ID\_cursa IN (

SELECT ID\_cursa

FROM Cursa

WHERE ID\_cursa = 1

);

--DELETE sterge relatiile de conducere pentru autobuzele cu capacitatea = 56

DELETE FROM Conduce

WHERE ID\_autobuz IN (

SELECT ID\_autobuz

FROM Autobuz

WHERE Capacitate = 56

);

--DELETE biletele pentru cursele din data respectiva

DELETE FROM Bilet

WHERE ID\_cursa IN(

SELECT ID\_cursa

FROM Cursa

WHERE Data\_plecare = '2025-12-05'

);

--DELETE elimina opririle curselor care pleaca din autogari aflate în orasul Bucuresti

DELETE FROM Opreste

WHERE ID\_cursa IN(

SELECT C.ID\_cursa

FROM CURSA C

JOIN Autogara A ON C.ID\_autogara = A.ID\_autogara

WHERE A.Oras = 'Bucuresti'

);

--DELETE autobuzele care nu sunt folosite la nicio cursa

DELETE FROM Autobuz

WHERE ID\_Autobuz NOT IN(

SELECT DISTINCT ID\_autobuz

FROM Foloseste

);

-- PUNCTUL 4: NORMALIZAREA RELATIILOR

--in structura anterioara locul din autobuz nu are nicio si ar putea sa apara pe mai multe bilete acelasi loc

--constrangere de unicitate care sa impiedice rezervarea aceluiasi loc de mai multe ori pe aceeasi cursa

ALTER TABLE Bilet

ADD CONSTRAINT UQ\_Bilet\_Cursa\_Loc UNIQUE (ID\_cursa, Loc);

--1NF (valorile sunt atomice)

--Pentru a evita anomalii de inserare(ex:locuri duplicate)

--de tipul 2NF nu exista deoarece exista tabele de legatura

--de tipul 3NF era initial, in momentul in care facusem relatia sofer-autobuz 1 la N

--aceasta s-a normalizat prin refacerea legaturi M:N, prin stergere FK-ului ID\_sofer din Autobuz si prin tabelul de legatura Conduce

--Relatia Cursa nu respecta initial forma normala a treia

--atributul ID\_autobuz introducea o dependenta tranzitiva intre cheia primara si atributele autobuzului

--de tipul 3NF era initial, in momentul in care facusem relatia cursa-autobuz 1 la N

--aceasta s-a normalizat prin refacerea legaturi M:N, prin stergere FK-ului ID\_autobuz din Cursa si prin tabelul de legatura Foloseste

ALTER TABLE Cursa

DROP COLUMN ID\_autobuz;

--PUNCTUL 5

--A

--returneaza doar cursele care au o autogara asociata

SELECT

C.ID\_cursa,

C.Data\_plecare,

C.Ora\_plecare,

A.Nume AS Autogara,

A.Oras

FROM Cursa C

INNER JOIN Autogara A

ON C.ID\_autogara = A.ID\_autogara;

--se obtin biletele vandute, impreuna cu pasagerul si data cursei

SELECT

B.ID\_bilet,

P.nume,

P.prenume,

C.data\_plecare,

B.pret

FROM Bilet B

INNER JOIN Pasager P ON B.ID\_pasager = P.ID\_pasager

INNER JOIN Cursa C ON B.ID\_cursa = C.ID\_cursa;

--returneaza cursele care au asociata atat o autogara, cat si o ruta

SELECT

C.ID\_cursa,

C.data\_plecare,

C.ora\_plecare,

A.oras,

R.oras\_destinatie,

R.distanta\_km

FROM Cursa C

INNER JOIN Autogara A ON C.ID\_autogara = A.ID\_autogara

INNER JOIN RutaFixa R ON C.ID\_rutafixa = R.ID\_rutafixa;

--afiseaza toate autogarile, impreuna cu curselele care pleaca din ele, daca exista

SELECT

A.ID\_autogara,

A.Nume,

A.Oras,

C.ID\_cursa,

C.Data\_plecare,

C.Ora\_plecare

FROM Autogara A

LEFT JOIN Cursa C ON A.ID\_autogara = C.ID\_autogara;

--afiseaza toate cursele, chiar si pe cele care nu au un autobuz asociat

SELECT

C.ID\_cursa,

C.Data\_plecare,

C.Ora\_plecare,

A.Numar\_inmatriculare

FROM Cursa C

LEFT JOIN Foloseste F ON C.ID\_cursa = F.ID\_cursa

LEFT JOIN Autobuz A ON F.ID\_autobuz = A.ID\_autobuz;

--afiseaza toate autobuzele, chiar si pe cele care nu sunt folosite in nicio cursa

SELECT

A.Numar\_inmatriculare,

A.Capacitate,

C.Data\_plecare

FROM Cursa C

RIGHT JOIN Foloseste F ON C.ID\_cursa = F.ID\_cursa

RIGHT JOIN Autobuz A ON F.ID\_autobuz = A.ID\_autobuz;

--5B

--Numarul de curse pentru fiecare autogara

SELECT

A.Oras,

COUNT(C.ID\_cursa) AS Numar\_curse

FROM Autogara A

JOIN Cursa C ON A.ID\_autogara = C.ID\_autogara

GROUP BY A.Oras;

--Pretul mediu al biletelor pentru fiecare cursa

SELECT

ID\_cursa,

AVG(Pret) AS Pret\_mediu

FROM Bilet

GROUP BY ID\_cursa;

--Capacitatea totala a autobuzelor pentru fiecare sofer

SELECT

S.nume,

S.prenume,

SUM(A.Capacitate) AS Capacitate\_totala

FROM Sofer S

JOIN Conduce C ON S.ID\_sofer = C.ID\_sofer

JOIN Autobuz A ON C.ID\_autobuz = A.ID\_autobuz

GROUP BY S.Nume, S.Prenume;

--Numarul de opriri pentru fiecare cursa

SELECT

ID\_cursa,

COUNT(ID\_statie) AS Numar\_opriri

FROM Opreste

GROUP BY ID\_cursa;

--autogarile care au cel putin 1 cursa

SELECT

A.Nume AS Nume\_Autogara,

A.Oras,

COUNT(C.ID\_cursa) AS Numar\_Curse

FROM Autogara A

JOIN Cursa C ON A.ID\_autogara = C.ID\_autogara

GROUP BY A.ID\_autogara, A.Nume, A.Oras

HAVING COUNT(C.ID\_cursa) >= 1;--filtram sa includem doar autogarile care au cel putin o cursa.

--5C

--subinterogari necorelate

--IN

--returneaza numele si prenumele soferilor care conduc un autobuz cu capacitatea > 50

SELECT DISTINCT S.Nume, S.Prenume

FROM Sofer S

JOIN Conduce C ON S.ID\_sofer = C.ID\_sofer

JOIN Autobuz A ON C.ID\_autobuz = A.ID\_autobuz

WHERE A.Capacitate > 50;

--EXISTS

--returneaza cursele care au cel putin o oprire in statii cu timp de stationare mai mare de 5 minute

SELECT ID\_cursa, data\_plecare, ora\_plecare

FROM Cursa

WHERE EXISTS(

SELECT 1

FROM Opreste O

JOIN Statie S ON O.ID\_statie = S.ID\_statie

WHERE O.ID\_cursa = Cursa.ID\_cursa

AND S.Timp\_stationare > 5

);

--ANY

--returneaza toate autobuzele cu capacitate mai mare decat capacitatea oricaruia dintre autobuzele conduse de soferul cu ID\_sofer = 1

SELECT \*

FROM Autobuz

WHERE Capacitate > ANY (

SELECT A.Capacitate

FROM Autobuz A

JOIN Conduce C ON A.ID\_autobuz = C.ID\_autobuz

WHERE C.ID\_sofer = 1

);

--NOT IN

--returneaza pasagerii care nu au niciun bilet pentru cursele ce pleaca din autogari din orasul Bucuresti

SELECT ID\_pasager, nume, prenume

FROM Pasager

WHERE ID\_pasager NOT IN(

SELECT B.ID\_pasager

FROM BILET B

JOIN Cursa C ON B.ID\_cursa = C.ID\_cursa

JOIN Autogara A ON C.ID\_autogara = A.ID\_autogara

WHERE A.Oras = 'Bucuresti'

);

--subinterogari corelate

--obtine toti soferii care conduc autobuze ce au fost folosite in curse care pleaca din autogari din Bucuresti

SELECT S.Nume, S.Prenume

FROM Sofer S

WHERE EXISTS(

SELECT 1 --returneaza cel putin un rand

FROM Conduce C

JOIN Foloseste F ON C.ID\_autobuz = F.ID\_autobuz

JOIN Cursa Cu ON F.ID\_cursa = Cu.ID\_cursa

JOIN Autogara A ON Cu.ID\_autogara = A.ID\_autogara

WHERE C.ID\_sofer = S.ID\_sofer

AND A.Oras = 'Bucuresti'

);

--afiseaza toate cursele care au bilete cumparate de pasageri care au si bilete la curse cu data de plecare dupa 2025-12-05

SELECT DISTINCT C.ID\_cursa, C.Data\_plecare, C.Ora\_plecare

FROM Cursa C

WHERE EXISTS(

SELECT 1

FROM Bilet B

JOIN Cursa C2 ON B.ID\_cursa = C2.ID\_cursa

WHERE B.ID\_pasager IN(

SELECT B2.ID\_pasager

FROM Bilet B2

JOIN Cursa C3 ON B2.ID\_cursa = C3.ID\_cursa

WHERE C3.Data\_plecare > '2025-12-05'

)

AND B.ID\_cursa = C.ID\_cursa

);

--afiseaza autogarile care au curse cu autobuze conduse de soferi cu experienta mai mica de 10 ani

SELECT DISTINCT A.nume, A.oras

FROM Autogara A

WHERE EXISTS(

SELECT 1

FROM Cursa C

JOIN Foloseste F ON C.ID\_cursa = F.ID\_cursa

JOIN Conduce Co ON F.ID\_autobuz = Co.ID\_autobuz

JOIN Sofer S ON Co.ID\_sofer = S.ID\_sofer

WHERE C.ID\_autogara = A.ID\_autogara

AND S.experienta < 10

);

--afiseaza pasagerii care au cumparat bilete la curse ce opresc in statii cu timp de stationare mai mare de 5 minute

SELECT DISTINCT P.nume, P.prenume

FROM Pasager P

WHERE EXISTS(

SELECT 1

FROM Bilet B

JOIN Opreste O ON B.ID\_cursa = O.ID\_cursa

WHERE B.ID\_pasager = P.ID\_pasager

AND O.Timp\_stationare > 5

);

--5D

--Afiseaza informatii despre pasageri care au bilete la curse dupa 1 decembrie 2025

SELECT

-- Functii pe siruri de caractere

CONCAT(P.Nume, ' ', P.Prenume) AS Nume\_Complet, -- concatenate

UPPER(SUBSTRING(A.Oras FROM 1 FOR 3)) AS Prefix\_Oras, -- primele 3 caractere majuscule

-- Funcții pe date calendaristice

EXTRACT(year FROM C.Data\_plecare) AS An\_Plecare, -- extrage anul

(CURRENT\_DATE - C.Data\_plecare) AS Zile\_de\_la\_plecare, -- diferenta in zile (date - date)

-- Expresie CASE

CASE --

WHEN S.Experienta >= 10 THEN 'Veteran'

WHEN S.Experienta BETWEEN 5 AND 9 THEN 'Medie'

ELSE 'Incepator'

END AS Nivel\_Experienta

FROM Pasager P

JOIN Bilet B ON P.ID\_pasager = B.ID\_pasager

JOIN Cursa C ON B.ID\_cursa = C.ID\_cursa

JOIN Autogara A ON C.ID\_autogara = A.ID\_autogara

JOIN Foloseste F ON C.ID\_cursa = F.ID\_cursa

JOIN Conduce Co ON F.ID\_autobuz = Co.ID\_autobuz

JOIN Sofer S ON Co.ID\_sofer = S.ID\_sofer

WHERE C.Data\_plecare >= '2025-12-01';

--Legaturi:pasagerii de biletele lor, biletele de curse, cursele de autogari, apoi de autobuze si, in final, de soferii care le conduc

--PUNCTUL 5E

-- Vedere simpla pe tabelul Sofer (vedere fara JOIN, permite modificari)

CREATE VIEW VedereSoferi AS

SELECT ID\_sofer, Nume, Prenume, Experienta

FROM Sofer

WHERE Experienta > 5;

-- Vedere care combina curse si autogari (vedere cu JOIN)

CREATE VIEW VedereCurseAutogari AS

SELECT C.ID\_cursa, C.Data\_plecare, C.Ora\_plecare, A.Nume AS Nume\_Autogara, A.Oras

FROM Cursa C

JOIN Autogara A ON C.ID\_autogara = A.ID\_autogara;

-- Vedere cu agregare bilete per cursa (nu permite modificari)

CREATE VIEW VederePretMedBilete AS

SELECT ID\_cursa, AVG(Pret) AS Pret\_mediu

FROM Bilet

GROUP BY ID\_cursa;

--Exemplu operatie LMD permisa pe vedere

-- Pe vederea simplă VedereSoferi putem face UPDATE

UPDATE VedereSoferi

SET Experienta = Experienta + 1

WHERE ID\_sofer = 1;

--Exemplu operatie LMD nepermisa pe vedere

/\*UPDATE VederePretMedBilete

SET Pret\_mediu = 100

WHERE ID\_cursa = 1;\*/

-- Va genera eroare deoarece vederea contine GROUP BY si agregare

--PUNCTUL 6

--Crearea unui index care sa optimizeze o interogare cu 2 criterii de cautare

--initial

SELECT \*

FROM Cursa

WHERE Data\_plecare = '2025-12-01'

AND ID\_autogara = 1;

--optimizare

DROP INDEX IF EXISTS idx\_cursa\_data\_autogara;

CREATE INDEX idx\_cursa\_data\_autogara

ON Cursa (Data\_plecare, ID\_autogara);

SELECT \* FROM Autogara;

DELETE FROM Autogara

WHERE ID\_autogara = 11;

INSERT INTO Autogara VALUES (11, 'Autogara Sud', 'Str. Popoveni 1', 'Bucuresti');

SELECT \* FROM Cursa;

SELECT \* FROM Autobuz;

INSERT INTO RutaFixa VALUES (11, 'Craiova', 'Bucuresti', 230);

INSERT INTO Autobuz VALUES (12, 'DJ000MM', 54);

SELECT \* FROM Autogara WHERE Oras = 'Bucuresti';

SELECT \* FROM Cursa WHERE Data\_plecare = '2025-12-01';

SELECT column\_name FROM information\_schema.columns WHERE table\_name = 'cursa';

SELECT \* FROM Autogara;

SELECT \* FROM Sofer;

SELECT \* FROM Autobuz;

-- Eliminăm constrângerea veche

ALTER TABLE bilet

DROP CONSTRAINT bilet\_id\_pasager\_fkey;

-- O recreăm cu regula de ștergere în cascadă

ALTER TABLE bilet

ADD CONSTRAINT bilet\_id\_pasager\_fkey

FOREIGN KEY (id\_pasager)

REFERENCES pasager(id\_pasager)

ON DELETE CASCADE;

-- Eliminăm constrângerea pentru cursă

ALTER TABLE bilet

DROP CONSTRAINT bilet\_id\_cursa\_fkey;

-- O recreăm cu regula de ștergere în cascadă

ALTER TABLE bilet

ADD CONSTRAINT bilet\_id\_cursa\_fkey

FOREIGN KEY (id\_cursa)

REFERENCES cursa(id\_cursa)

ON DELETE CASCADE;

DELETE FROM pasager WHERE id\_pasager = 1;

SELECT \* FROM bilet WHERE id\_pasager = 1;

Link catre github: https://github.com/GeorgianaBejoiu/proiectBD