Proiect

-Baze de Date-

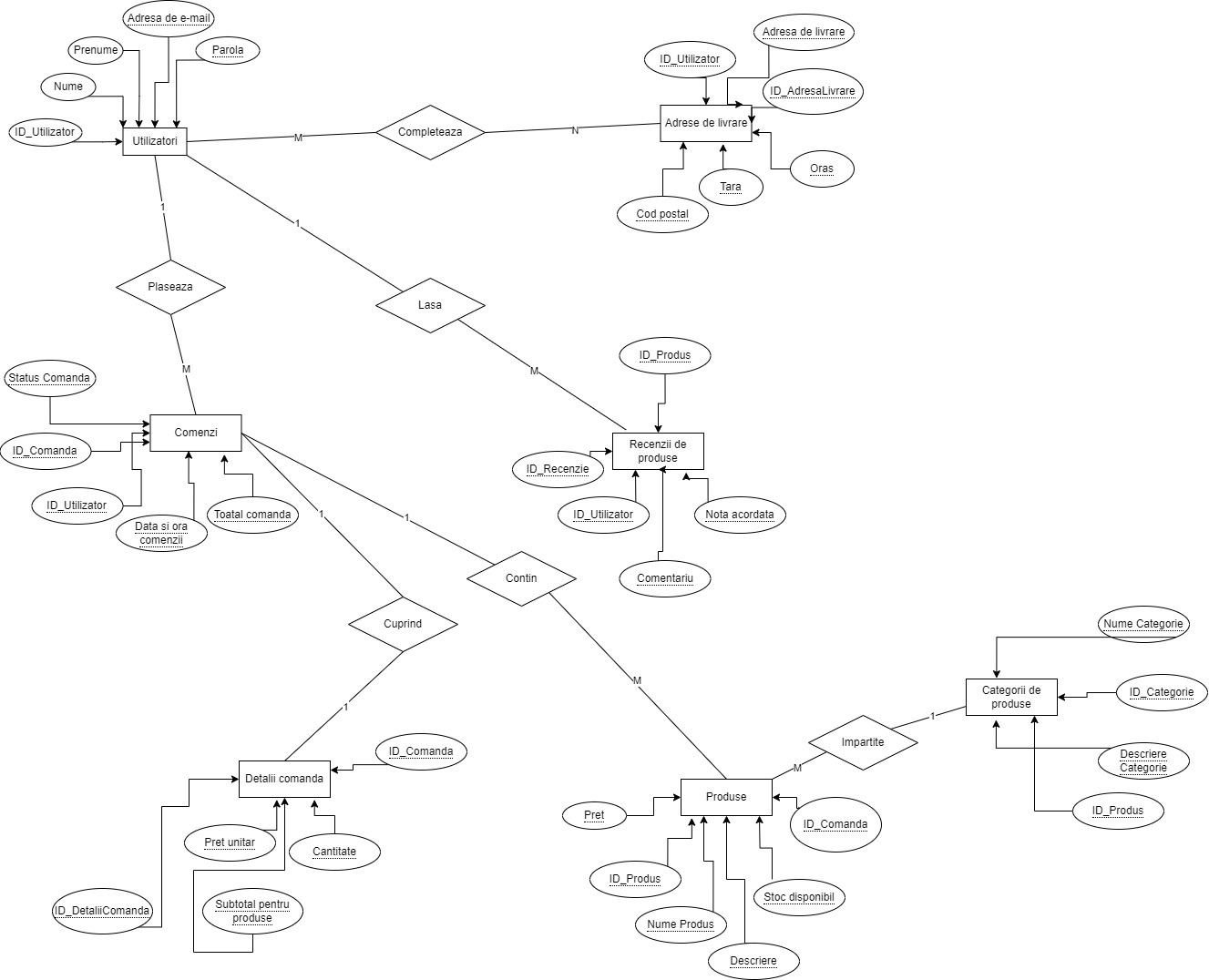
Safciu Alexandru Radu

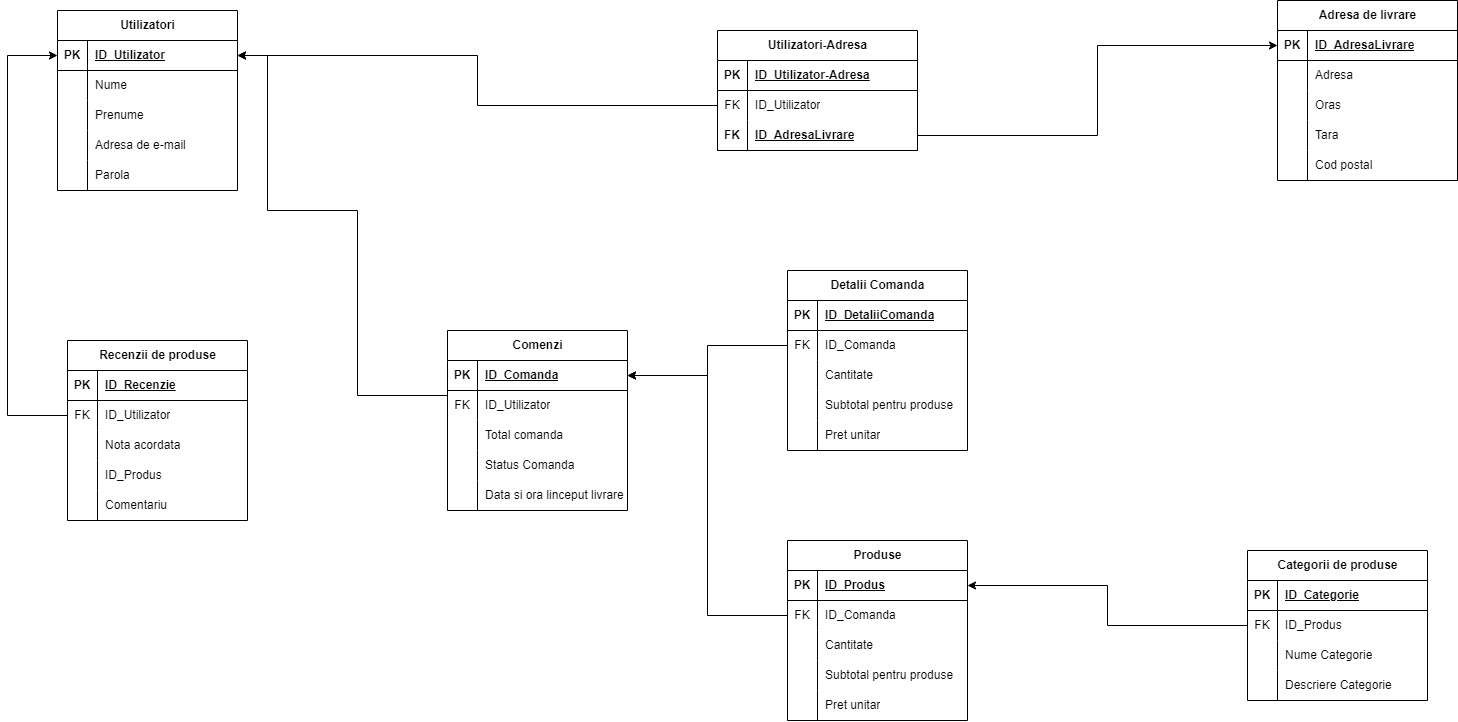
**Descriere tema**

Pentru acest tema de casa am proiectat si implementat o baza de date care descrie un magazine online simplu.

Magazinul are utilizatori care pot comanda produse din diferite categorii care sa fie livrate la domiciliu. De asemenea acestia pot lasa recenzii la produsele pe care le au cumparat.

**Diagrama ER**

****

**Schema Relationala**

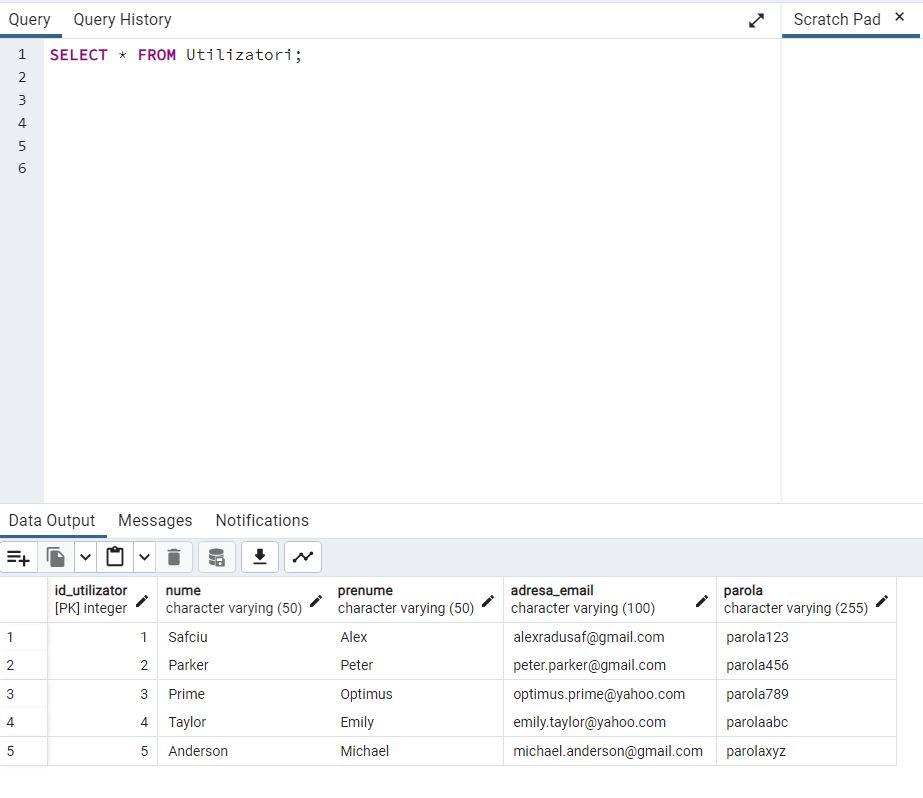
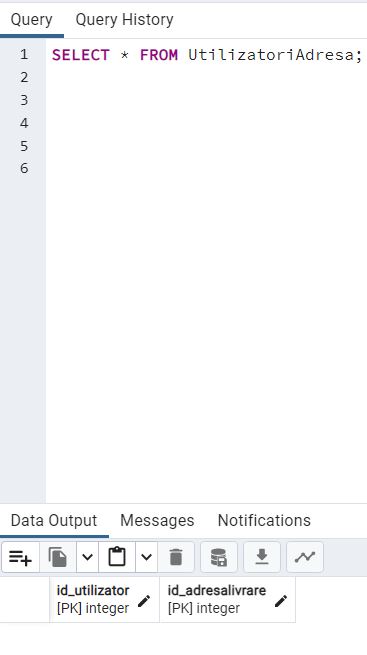
**Crearea tabelelor**

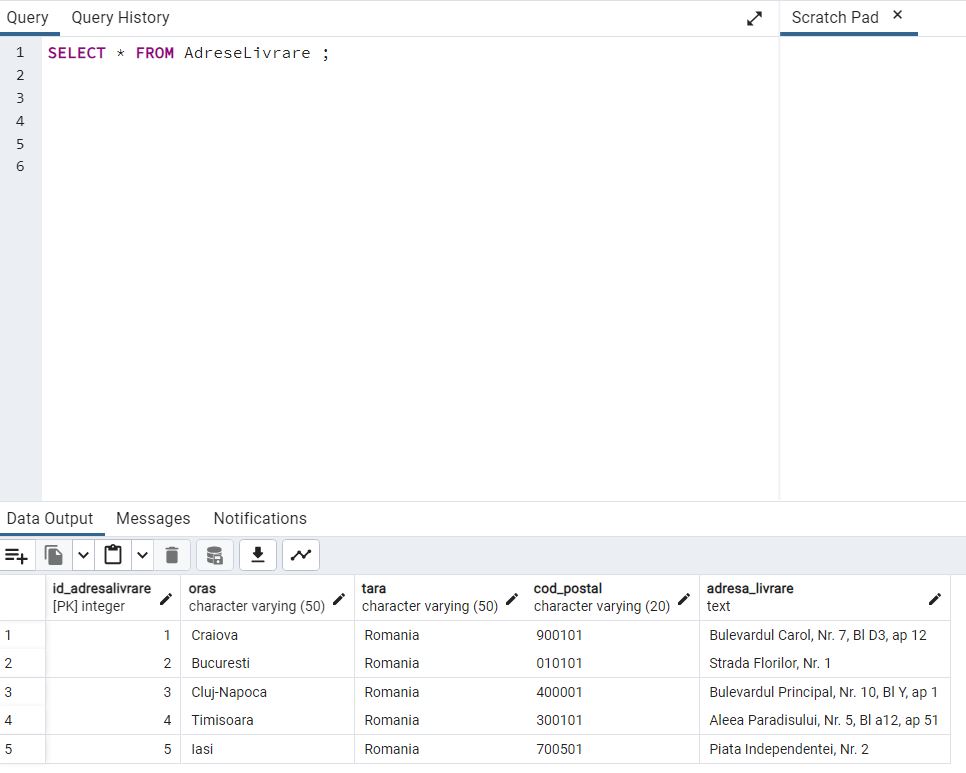
Am creat tabelele descrise in diagrama ER si am pus constrangerile si relatiile dintre tabele cu ajutorul freign keys regasite in Schema Relationala.

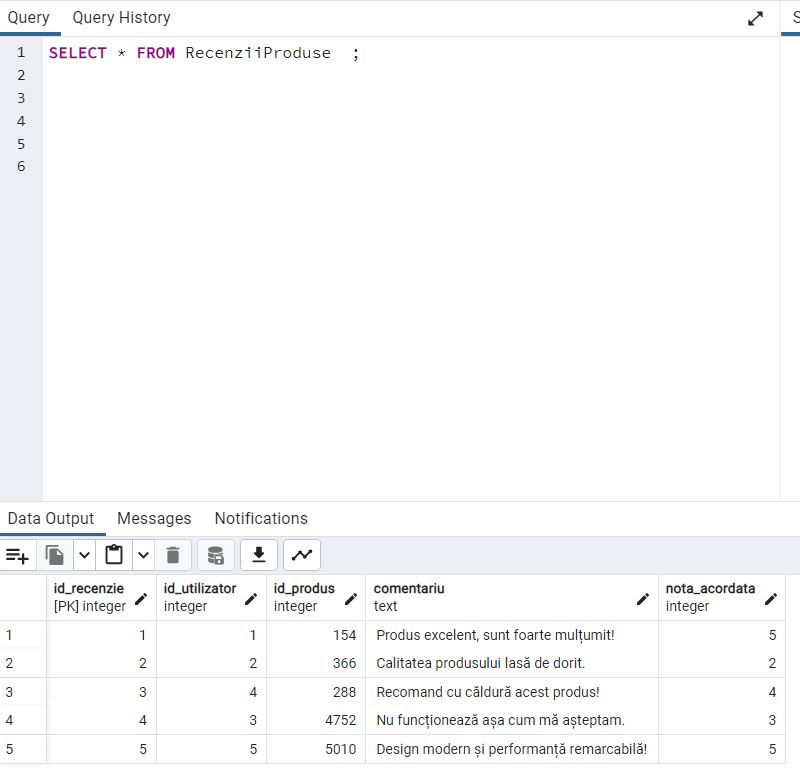
Query-urile pentru crearea tabelelor este [pe github-ul meu](https://github.com/AlexSafciu/Proiect-baze-de-date/blob/main/Creare%20Tabele)

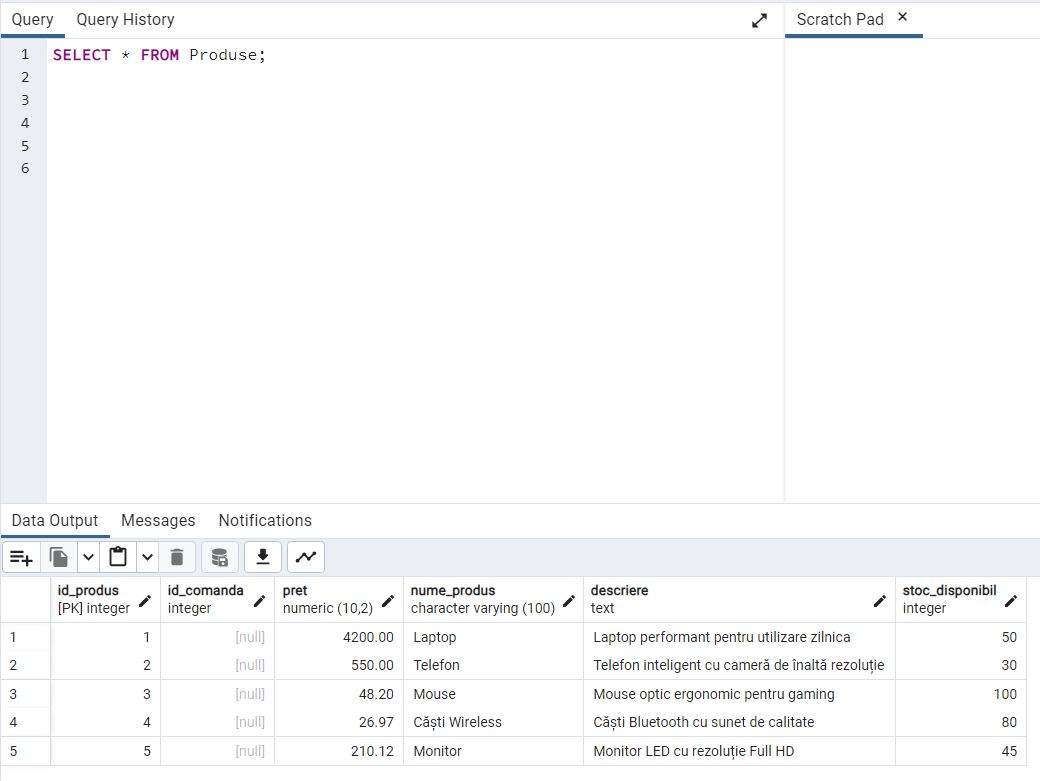
**Inserarea Datelor**

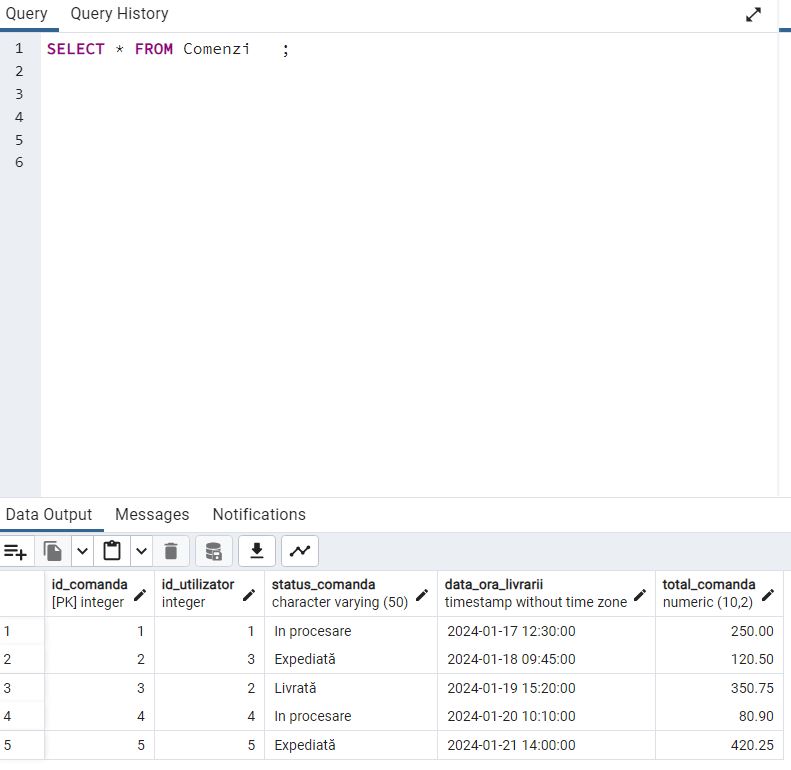
Am continuat prin a popula tabelele, codul este tot [pe github-ul meu](https://github.com/AlexSafciu/Proiect-baze-de-date/blob/main/Inserare%20date%20in%20tabele).

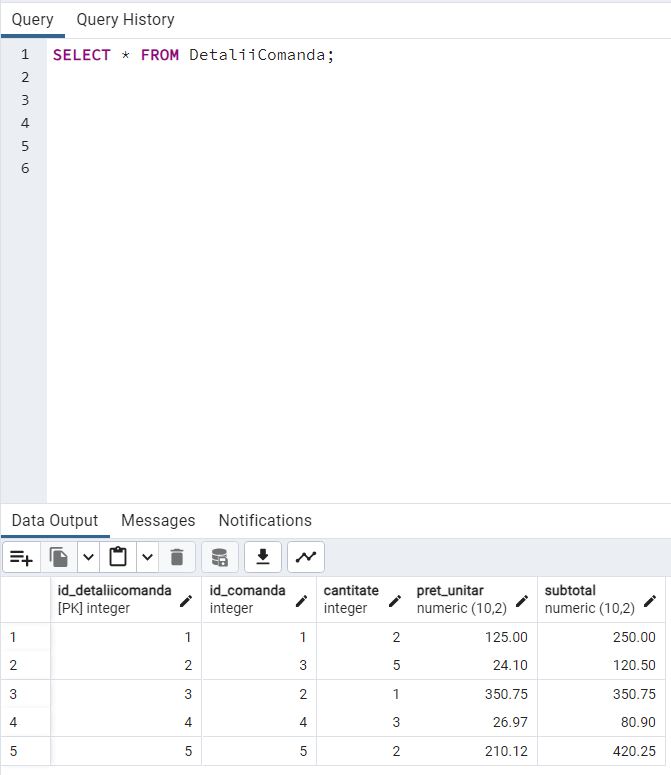


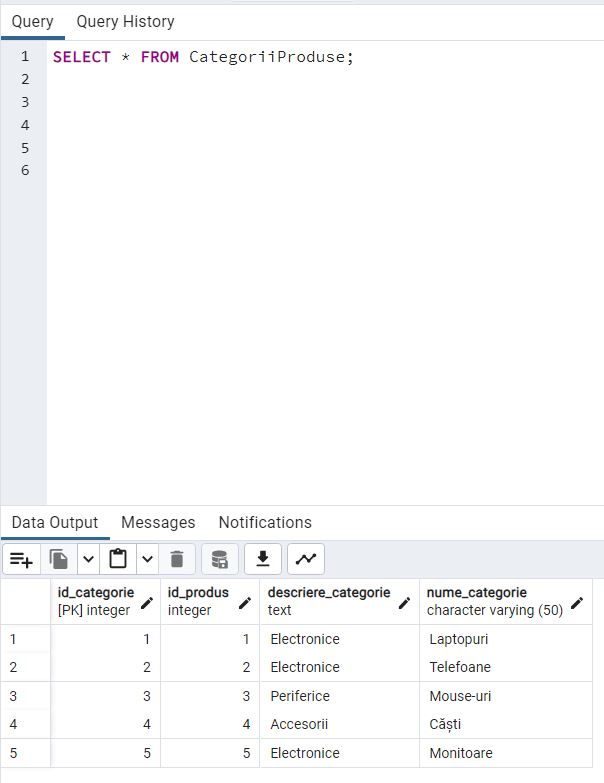










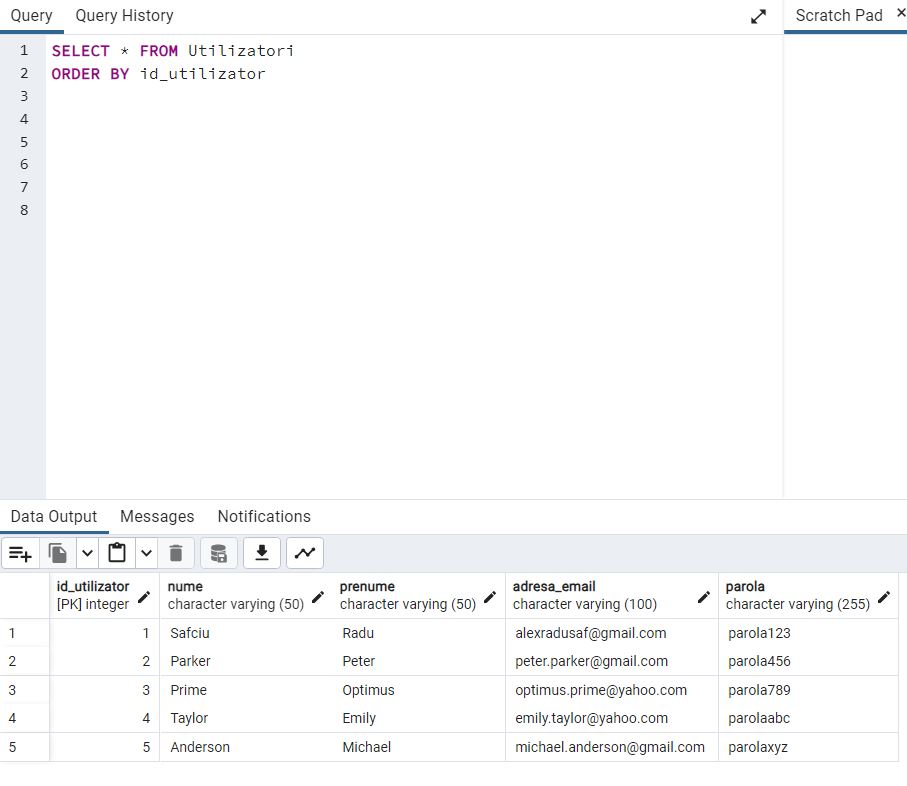


**Exemple operatii modificare date**

UPDATE Utilizatori

SET Prenume = 'Radu'

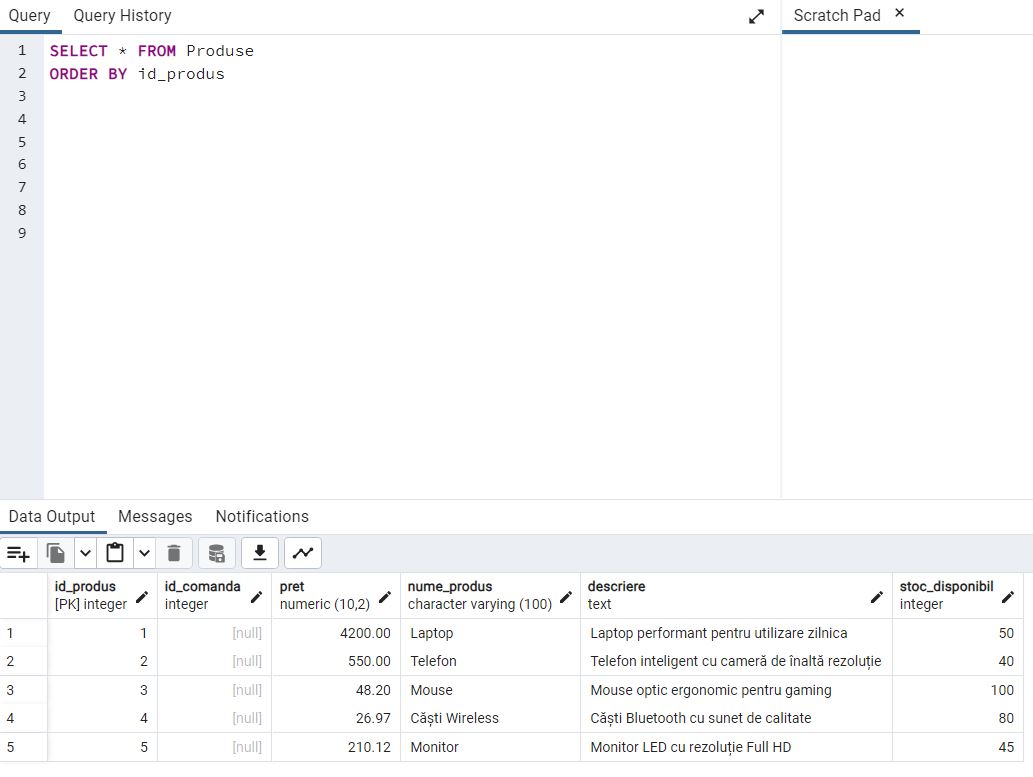
WHERE ID\_Utilizator = 1;



UPDATE Produse

SET Stoc\_Disponibil = Stoc\_Disponibil + 10

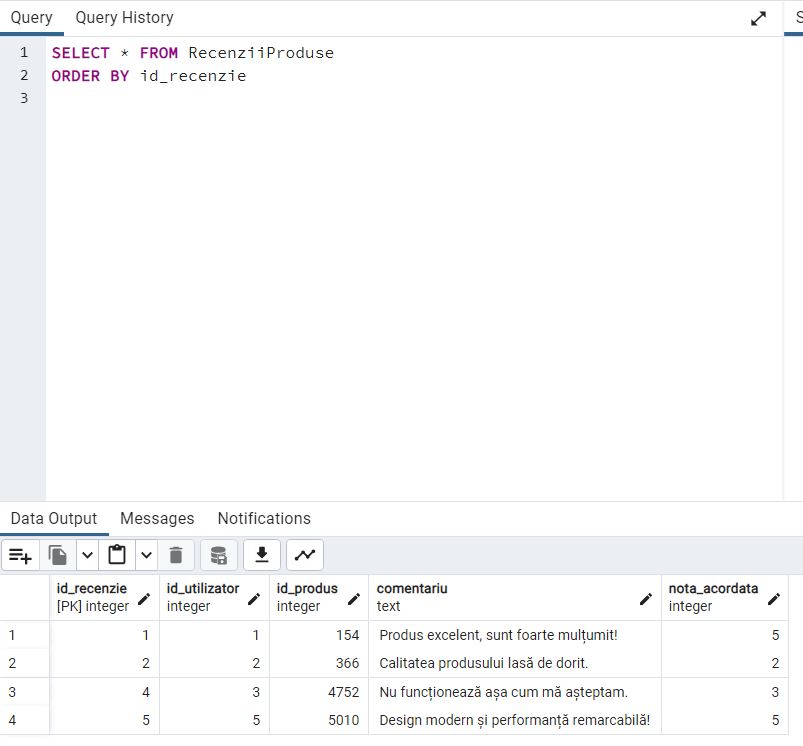
WHERE ID\_Produs = 2;



**Exemple operatii stergere date**

DELETE FROM RecenziiProduse

WHERE ID\_Recenzie = 3;



(id\_produs din aceasta imagine este gresit, l-am corectat dar nu am mai facut ss-uri pentru problemele prin care deja trecusem.)

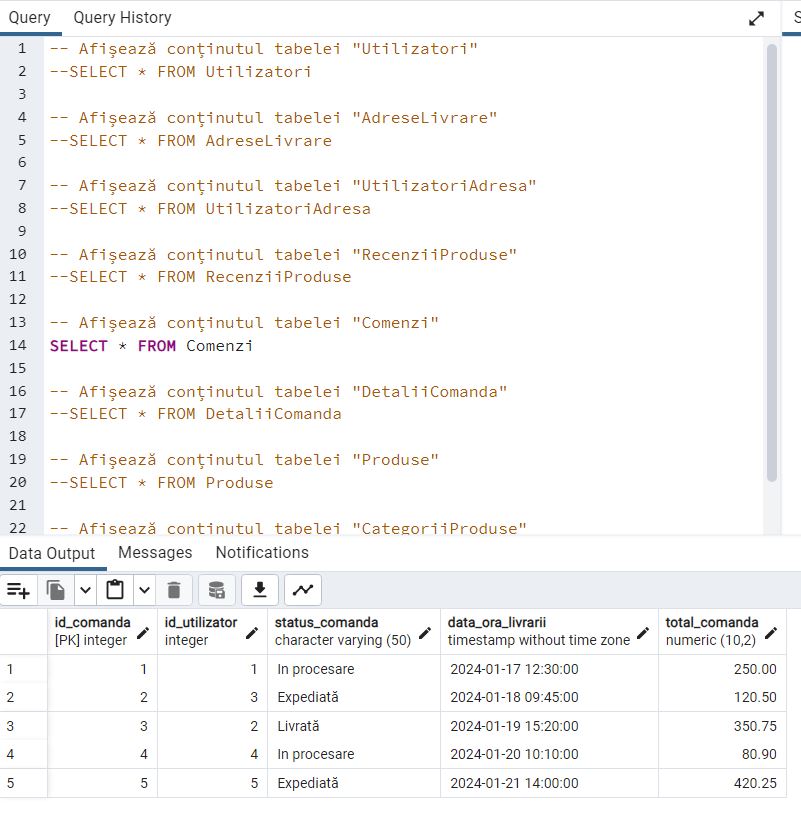
**Exemple de operații de modificare sau ștergere folosind subinterogări:**

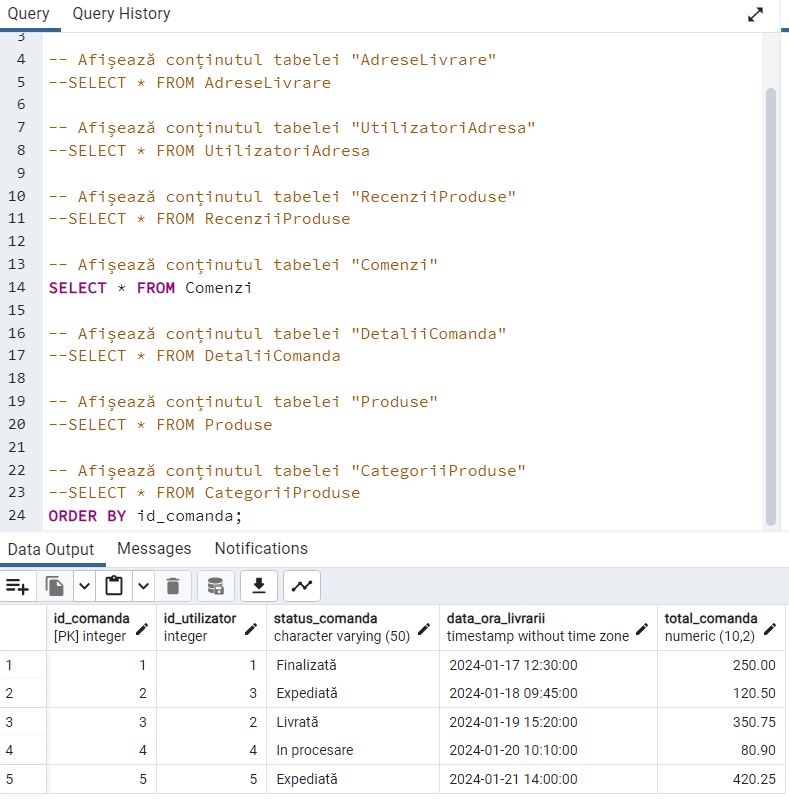
Actualizare - Actualizarea statusului comenzii la 'Finalizată' pentru un utilizator anume:

UPDATE Comenzi

SET Status\_Comanda = 'Finalizată'

WHERE ID\_Utilizator = (SELECT ID\_Utilizator FROM Utilizatori WHERE Nume = 'Safciu' AND Prenume = 'Radu');



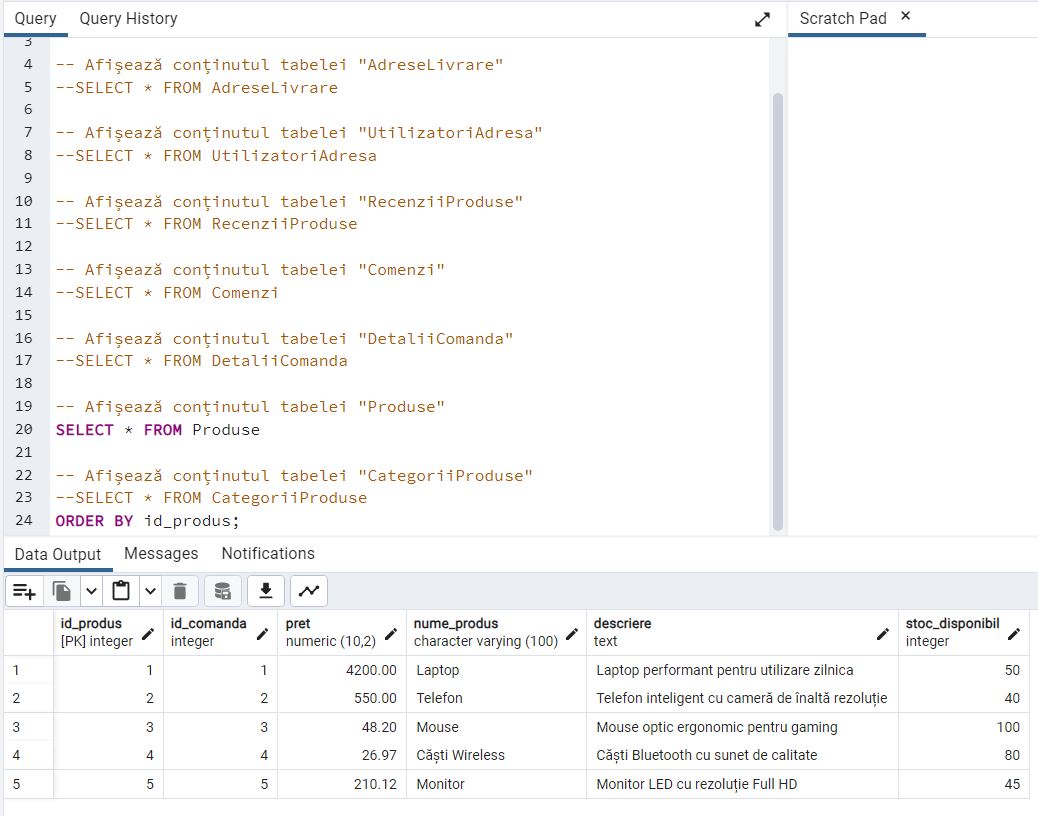


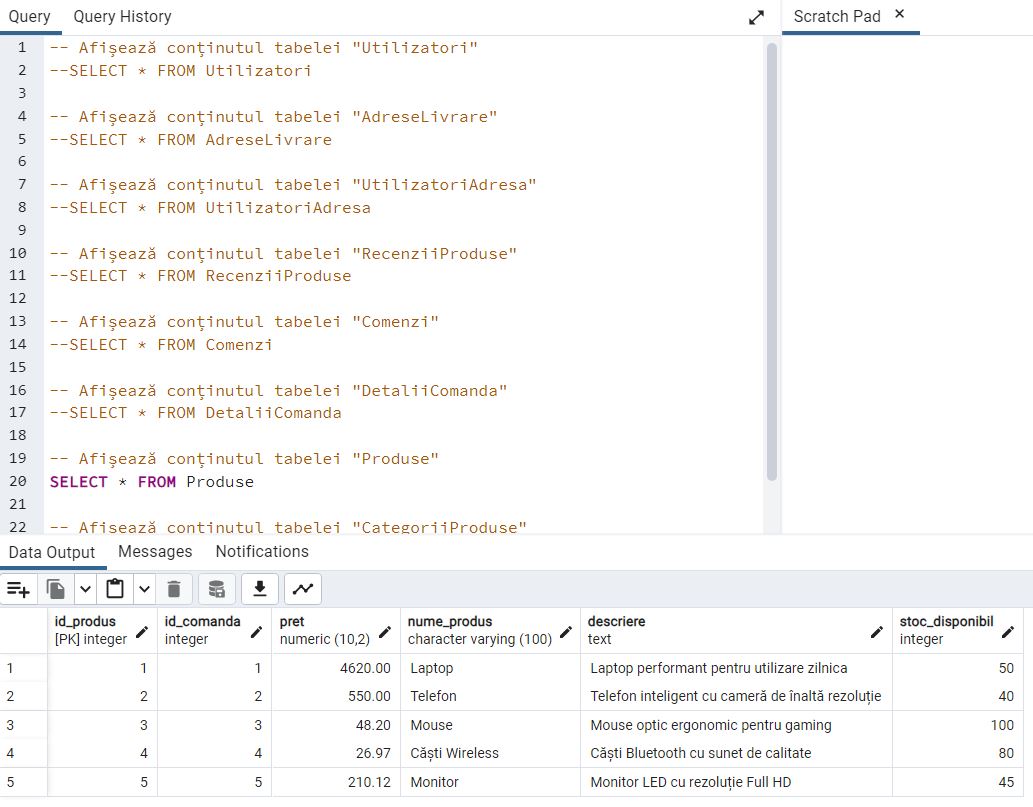
Actualizare - Actualizarea prețului produselor cu o anumită categorie:

UPDATE Produse

SET Pret = Pret \* 1.1

WHERE ID\_Produs IN (SELECT ID\_Produs FROM CategoriiProduse WHERE Nume\_Categorie = 'Laptopuri');

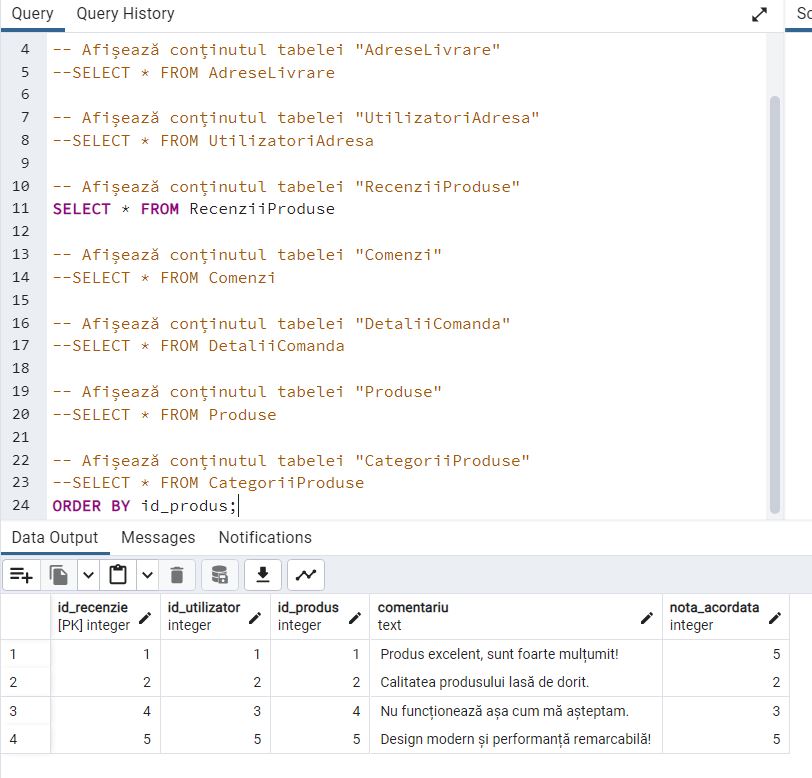


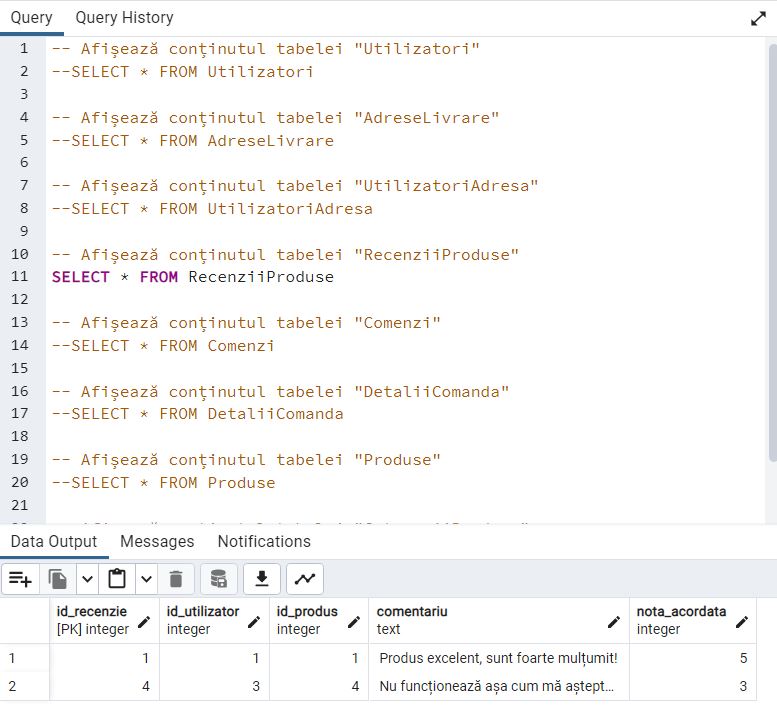


Ștergere - Ștergerea tuturor recenziilor asociate produselor cu stoc disponibil mai mic de 49:

DELETE FROM RecenziiProduse

WHERE ID\_Produs IN (SELECT ID\_Produs FROM Produse WHERE Stoc\_Disponibil < 49);





**Normalizarea relatiilor prezente**

Normalizarea este procesul de proiectare al bazelor de date pentru a minimiza redundanța și a asigura consistența datelor. Există mai multe forme normale, iar fiecare urmărește anumite reguli de organizare a datelor pentru a preveni anumite anomalii.

O situație care nu respectă 1NF ar putea fi dacă ai avea o coloană într-o tabelă care conține o listă de valori separate de virgule, cum ar fi o listă de adrese într-un singur câmp.

-- Nu respectă 1NF

CREATE TABLE Utilizatori (

ID\_Utilizator SERIAL PRIMARY KEY,

Nume VARCHAR(50) NOT NULL,

Prenume VARCHAR(50) NOT NULL,

Adrese\_Livrare VARCHAR(255) NOT NULL -- Exemplu de câmp cu mai multe valori

);

O situație care nu respectă 2NF ar putea fi dacă ai o cheie primară compusă și există dependențe parțiale.

-- Nu respectă 2NF

CREATE TABLE DetaliiComanda (

ID\_DetaliiComanda SERIAL PRIMARY KEY,

ID\_Comanda INT,

Nume\_Produs VARCHAR(100) NOT NULL,

Pret\_Unitar DECIMAL(10, 2),

FOREIGN KEY (ID\_Comanda) REFERENCES Comenzi(ID\_Comanda)

);

În acest caz, Nume\_Produs și Pret\_Unitar depind doar de o parte a cheii primare (ID\_Comanda).

O situație care nu respectă 3NF ar putea fi dacă există dependențe tranzitive.

-- Nu respectă 3NF

CREATE TABLE Produse (

ID\_Produs SERIAL PRIMARY KEY,

ID\_Comanda INT,

Nume\_Produs VARCHAR(100) NOT NULL,

Descriere TEXT,

Stoc\_Disponibil INT,

Pret DECIMAL(10, 2),

FOREIGN KEY (ID\_Comanda) REFERENCES Comenzi(ID\_Comanda)

);

Pret depinde de ID\_Comanda, dar și de Nume\_Produs. Într-o formă normală corectă, Pret ar trebui să depindă doar de cheia primară.

Din pacate, prima oara cand am citit cerinca credeam ca trebuie sa facem una din cele 3 normalizari, asa ca acest cod este in baza de date:

Creează o nouă tabelă pentru informații despre produse:

CREATE TABLE InformatiiProduse (

ID\_Produs SERIAL PRIMARY KEY,

Nume\_Produs VARCHAR(100) NOT NULL,

Descriere TEXT,

Stoc\_Disponibil INT

);

Actualizează tabela Produse pentru a include doar cheile primare necesare și adaugă o cheie străină către InformatiiProduse:

ALTER TABLE Produse

ADD COLUMN ID\_InfoProdus INT,

ADD FOREIGN KEY (ID\_InfoProdus) REFERENCES InformatiiProduse(ID\_Produs);

Transferă datele relevante din Produse la InformatiiProduse:

INSERT INTO InformatiiProduse (ID\_Produs, Nume\_Produs, Descriere, Stoc\_Disponibil)

SELECT ID\_Produs, Nume\_Produs, Descriere, Stoc\_Disponibil

FROM Produse;

Elimină coloanele redundante din Produse:

ALTER TABLE Produse

DROP COLUMN Nume\_Produs,

DROP COLUMN Descriere,

DROP COLUMN Stoc\_Disponibil;

Cu aceste modificări, am separat informațiile despre produse într-o nouă tabelă InformatiiProduse, iar tabela Produse conține acum doar cheile străine necesare. Astfel, baza de date ar trebui să respecte a doua formă normală (2NF).

**Interogari:**

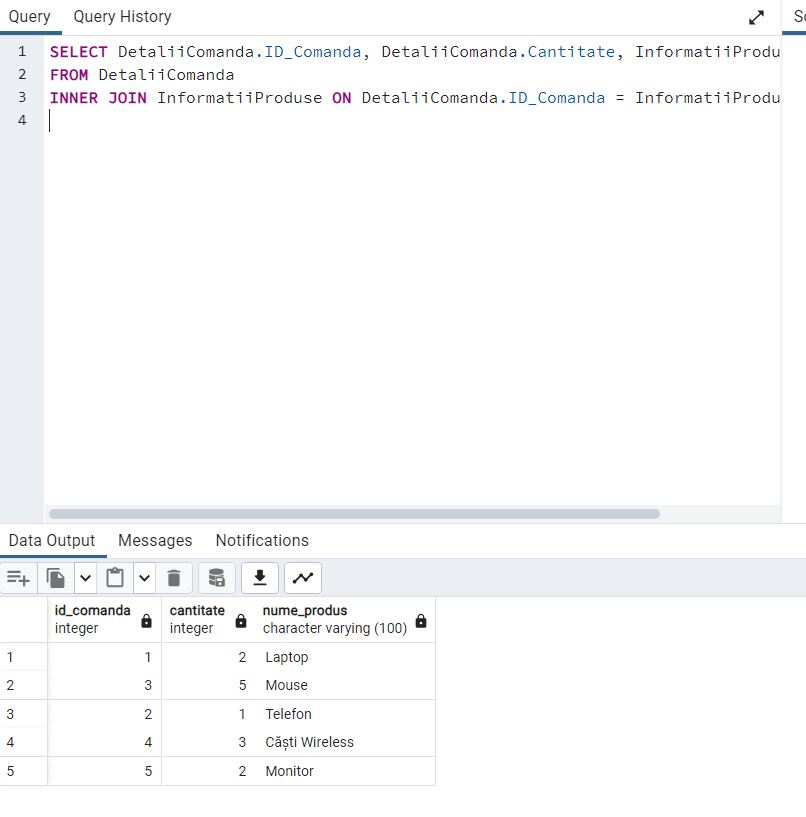
**Exemple interogari innerjoin+outerjoin:**

*Inner Join - Obține informații despre produsele dintr-o comandă:*

SELECT DetaliiComanda.ID\_Comanda, DetaliiComanda.Cantitate, InformatiiProduse.Nume\_Produs

FROM DetaliiComanda

INNER JOIN InformatiiProduse ON DetaliiComanda.ID\_Comanda = InformatiiProduse.ID\_Produs;



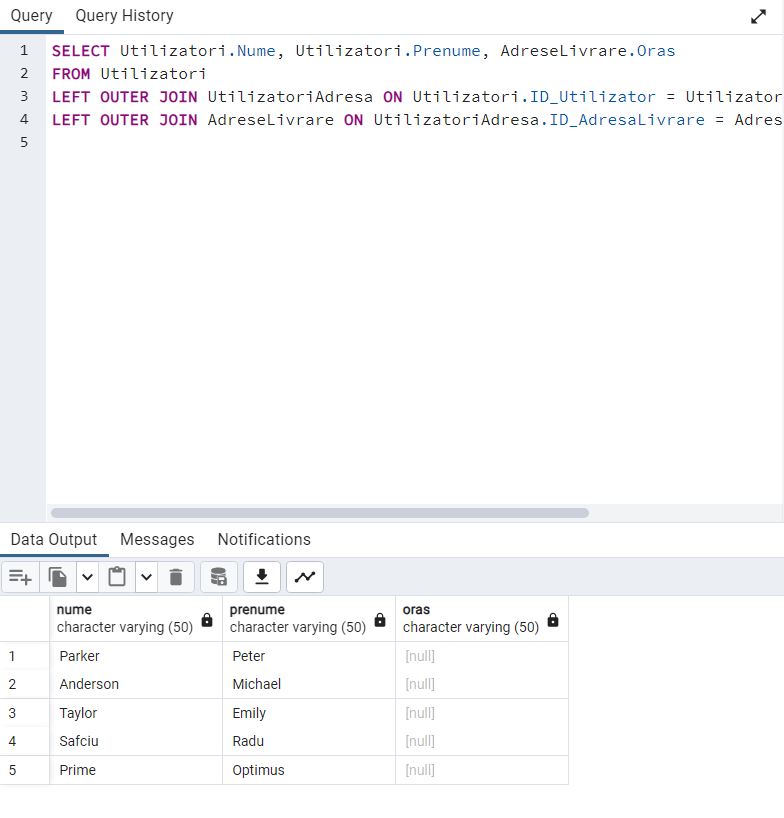
*Left Outer Join - Afișează toate adresele de livrare și utilizatorii asociați (chiar și cei fără adrese de livrare):*

SELECT Utilizatori.Nume, Utilizatori.Prenume, AdreseLivrare.Oras

FROM Utilizatori

LEFT OUTER JOIN UtilizatoriAdresa ON Utilizatori.ID\_Utilizator = UtilizatoriAdresa.ID\_Utilizator

LEFT OUTER JOIN AdreseLivrare ON UtilizatoriAdresa.ID\_AdresaLivrare = AdreseLivrare.ID\_AdresaLivrare;



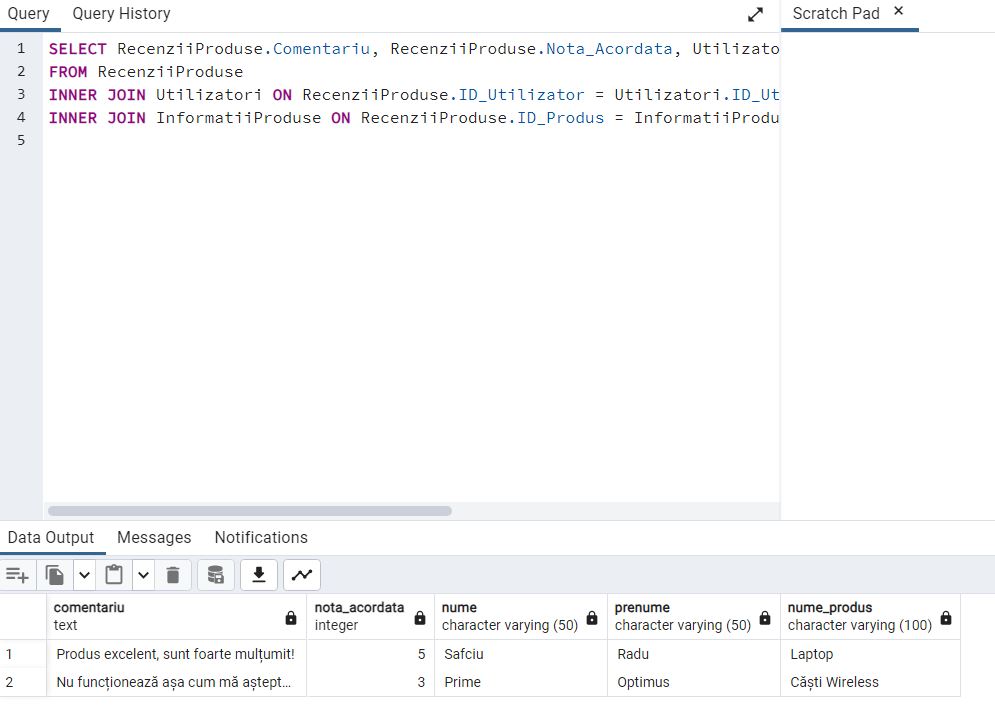
*Inner Join - Obține recenziile și numele utilizatorilor care le-au scris, împreună cu informații despre produse:*

SELECT RecenziiProduse.Comentariu, RecenziiProduse.Nota\_Acordata, Utilizatori.Nume, Utilizatori.Prenume, InformatiiProduse.Nume\_Produs

FROM RecenziiProduse

INNER JOIN Utilizatori ON RecenziiProduse.ID\_Utilizator = Utilizatori.ID\_Utilizator

INNER JOIN InformatiiProduse ON RecenziiProduse.ID\_Produs = InformatiiProduse.ID\_Produs;

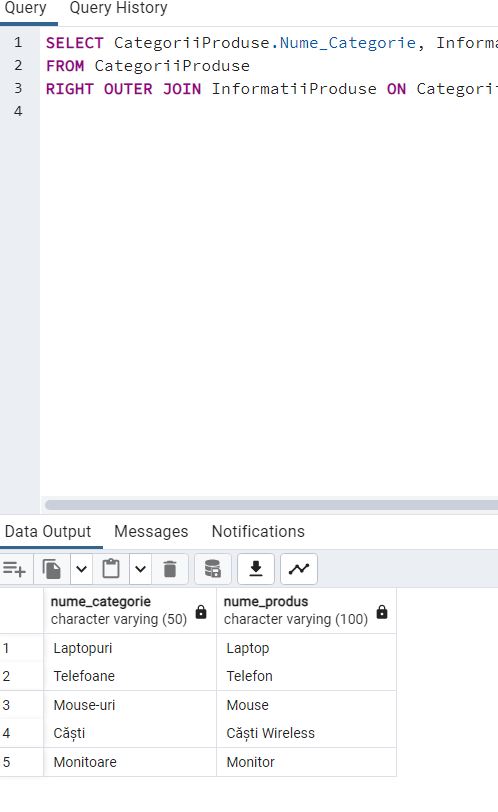


*Right Outer Join - Afișează toate categoriile de produse și produsele asociate (chiar și cele fără categorii), împreună cu informații despre produse:*

SELECT CategoriiProduse.Nume\_Categorie, InformatiiProduse.Nume\_Produs

FROM CategoriiProduse

RIGHT OUTER JOIN InformatiiProduse ON CategoriiProduse.ID\_Produs = InformatiiProduse.ID\_Produs;



**Exemple de interogări cu funcții de agregare și GROUP BY:**

*Afișează suma totală a comenzilor pentru fiecare utilizator:*

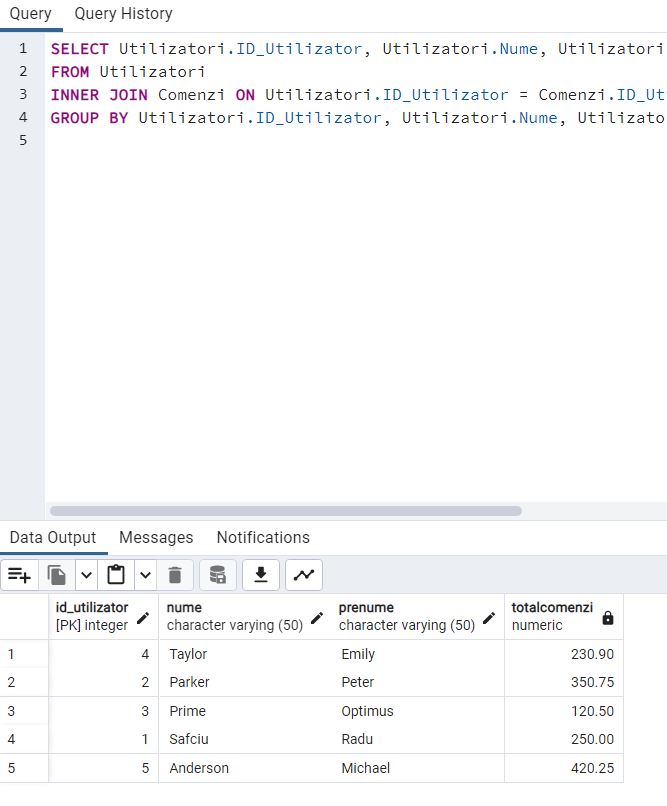
*(am adaugat inca o comanda la id utiliz 4)*

SELECT RecenziiProduse.ID\_Produs, InformatiiProduse.Nume\_Produs, AVG(RecenziiProduse.Nota\_Acordata) AS MedieNota

FROM RecenziiProduse

INNER JOIN InformatiiProduse ON RecenziiProduse.ID\_Produs = InformatiiProduse.ID\_Produs

GROUP BY RecenziiProduse.ID\_Produs, InformatiiProduse.Nume\_Produs;



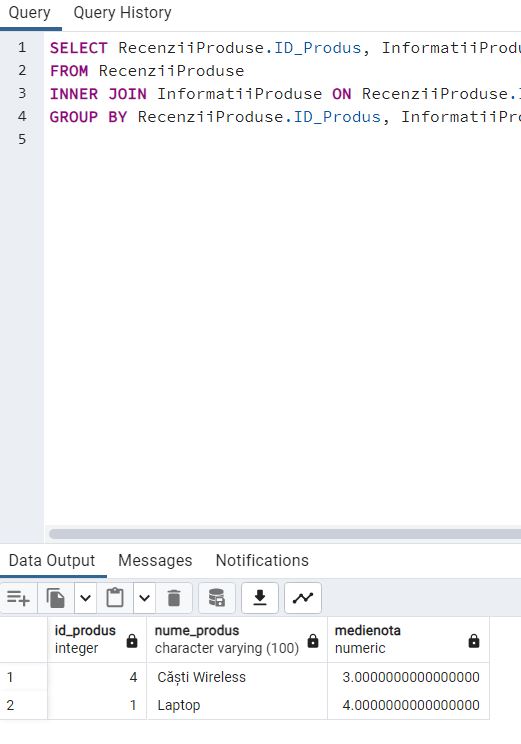
*Afișează media notelor acordate în recenzii pentru fiecare produs:*

SELECT RecenziiProduse.ID\_Produs, InformatiiProduse.Nume\_Produs, AVG(RecenziiProduse.Nota\_Acordata) AS MedieNota

FROM RecenziiProduse

INNER JOIN InformatiiProduse ON RecenziiProduse.ID\_Produs = InformatiiProduse.ID\_Produs

GROUP BY RecenziiProduse.ID\_Produs, InformatiiProduse.Nume\_Produs;



*Afișează cantitatea totală de produse comandate pentru fiecare categorie:*

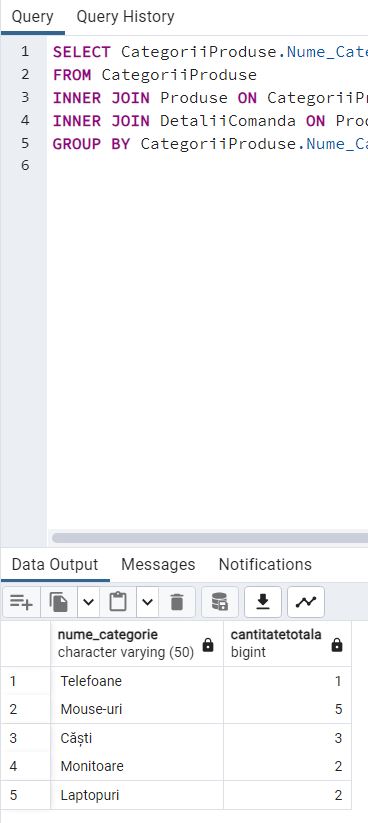
SELECT CategoriiProduse.Nume\_Categorie, SUM(DetaliiComanda.Cantitate) AS CantitateTotala

FROM CategoriiProduse

INNER JOIN Produse ON CategoriiProduse.ID\_Produs = Produse.ID\_Produs

INNER JOIN DetaliiComanda ON Produse.ID\_Comanda = DetaliiComanda.ID\_Comanda

GROUP BY CategoriiProduse.Nume\_Categorie;



**Exemple de subinterogări corelate:**

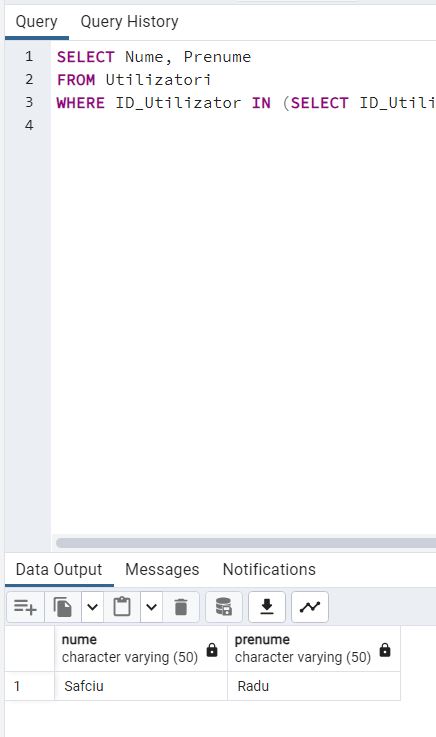
*Exemplu cu IN:*

Această subinterogare verifică utilizatorii care au comenzi finalizate.

SELECT Nume, Prenume

FROM Utilizatori

WHERE ID\_Utilizator IN (SELECT ID\_Utilizator FROM Comenzi WHERE Status\_Comanda = 'Finalizată');



*Exemplu cu ANY:*

*Această interogare returnează numele și prenumele utilizatorilor care au plasat comenzi cu o cantitate mai mare de 2 pentru orice produs, implicând astfel cele trei tabele menționate.*

SELECT Nume, Prenume

FROM Utilizatori

WHERE ID\_Utilizator IN (

SELECT DISTINCT ID\_Utilizator

FROM Comenzi

WHERE ID\_Comanda IN (

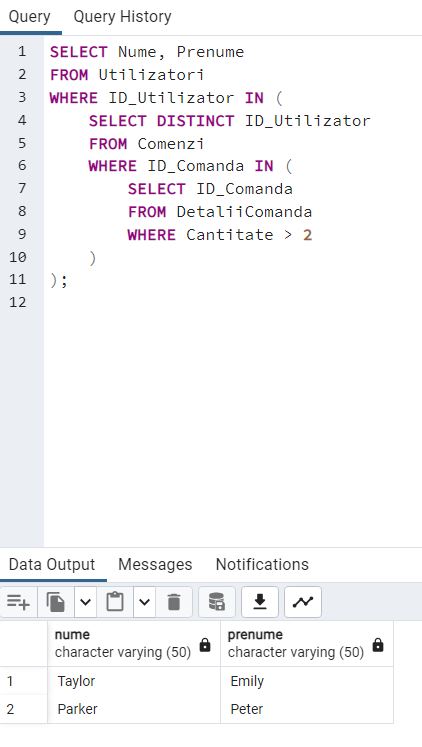
SELECT ID\_Comanda

FROM DetaliiComanda

WHERE Cantitate > 2

)

);



*Exemplu cu EXISTS:*

*Această interogare returnează numele și prenumele utilizatorilor care au comenzi cu o cantitate mai mare de 2 pentru oricare dintre produsele comandate, implicând cele trei tabele menționate.*

SELECT Nume, Prenume

FROM Utilizatori U

WHERE EXISTS (

SELECT 1

FROM Comenzi C

WHERE EXISTS (

SELECT 1

FROM DetaliiComanda DC

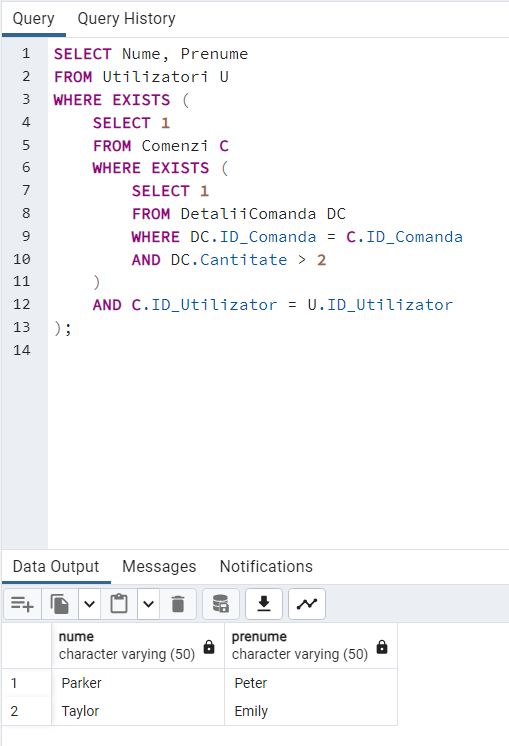
WHERE DC.ID\_Comanda = C.ID\_Comanda

AND DC.Cantitate > 2

)

AND C.ID\_Utilizator = U.ID\_Utilizator

);



*Exemplu cu ALL:*

*Această interogare verifică dacă utilizatorul are toate comenzile finalizate cu un total mai mare de 200 și, în același timp, are toate detaliile comenzilor cu o cantitate mai mare de 1.*

SELECT Nume

FROM Utilizatori

WHERE ID\_Utilizator = ALL (

SELECT ID\_Utilizator

FROM Comenzi

WHERE Status\_Comanda = 'Finalizată' AND Total\_Comanda > 200

AND ID\_Utilizator = ALL (

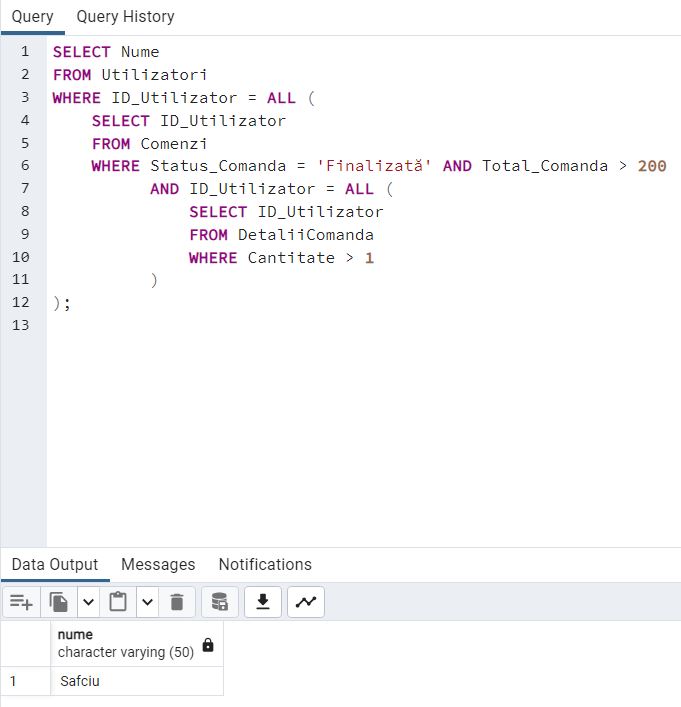
SELECT ID\_Utilizator

FROM DetaliiComanda

WHERE Cantitate > 1

)

);



**Exemple de subinterogări necorelate:**

*IN:*

*Se selectează numele utilizatorilor care au plasat comenzi și au lăsat recenzii la produse.*

SELECT Nume

FROM Utilizatori

WHERE ID\_Utilizator IN (

SELECT DISTINCT ID\_Utilizator

FROM Comenzi

) AND ID\_Utilizator IN (

SELECT DISTINCT ID\_Utilizator

FROM RecenziiProduse

);



*ANY:*

*Se selectează numele categoriilor de produse care au cel puțin un produs cu un stoc disponibil mai mare de 10 unități.*

SELECT Nume\_Categorie

FROM CategoriiProduse

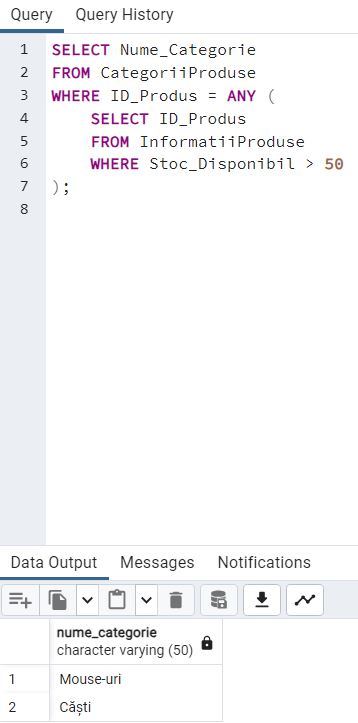
WHERE ID\_Produs = ANY (

SELECT ID\_Produs

FROM InformatiiProduse

WHERE Stoc\_Disponibil > 10

);



*EXISTS:*

*Se selectează numele produselor care fac parte din categoria "Electronice" și nu au fost comandate în cantități mai mari de 5.*

SELECT Nume\_Produs

FROM InformatiiProduse

WHERE EXISTS (

SELECT 1

FROM CategoriiProduse

WHERE Descriere\_Categorie LIKE 'Electronice%'

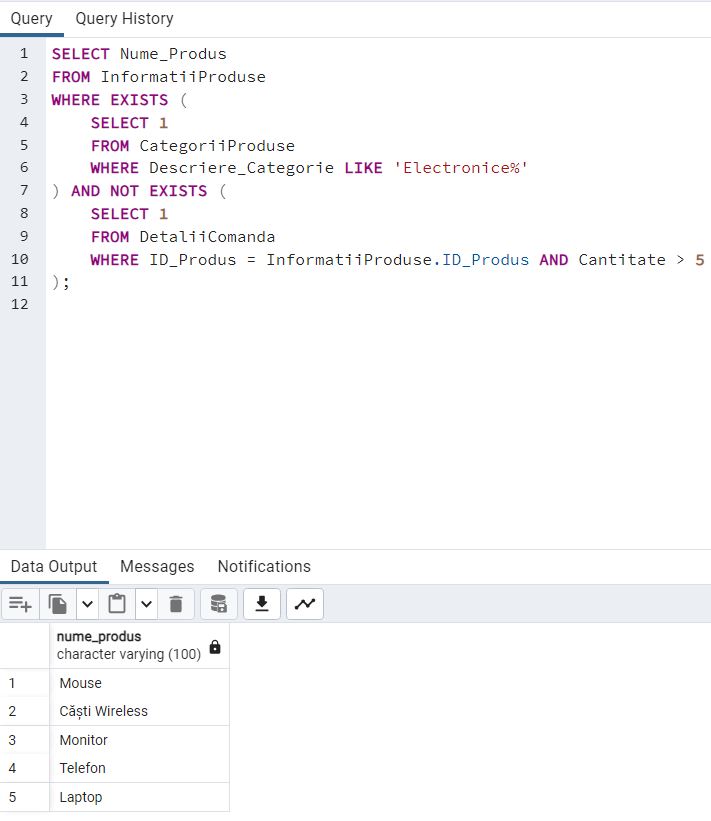
) AND NOT EXISTS (

SELECT 1

FROM DetaliiComanda

WHERE ID\_Produs = InformatiiProduse.ID\_Produs AND Cantitate > 5

);



*ALL:*

*Se selectează numele utilizatorilor care au plasat toate comenzile finalizate și au dat note mai mari de 3 în toate recenziile lor.*

SELECT Nume

FROM Utilizatori

WHERE ID\_Utilizator = ALL (

SELECT ID\_Utilizator

FROM Comenzi

WHERE Status\_Comanda = 'Finalizată'

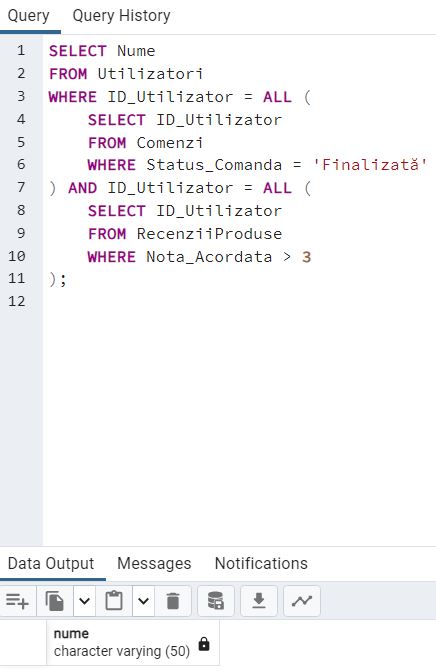
) AND ID\_Utilizator = ALL (

SELECT ID\_Utilizator

FROM RecenziiProduse

WHERE Nota\_Acordata > 3

);

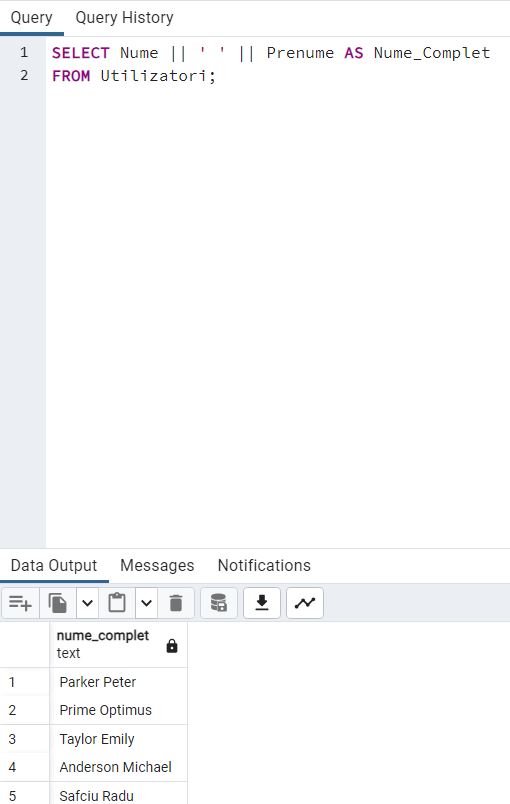


**Exemple de interogări folosind funcții pe șiruri de caractere:**

*Exemplu de concatenare a numelui și prenumelui utilizatorilor:*

SELECT Nume || ' ' || Prenume AS Nume\_Complet

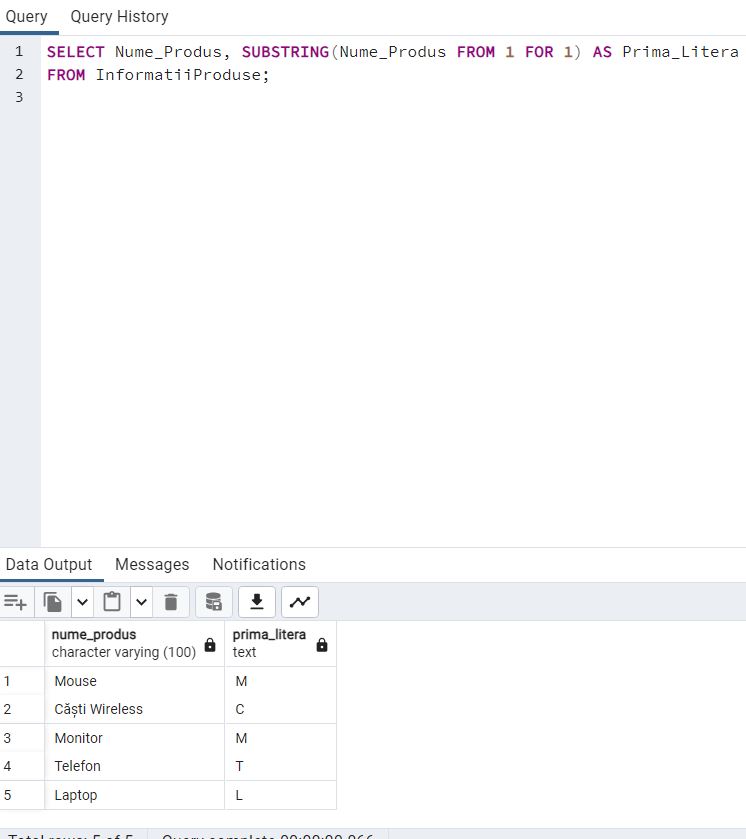
FROM Utilizatori;



*Exemplu de extragere a primei litere din numele produselor:*

SELECT Nume\_Produs, SUBSTRING(Nume\_Produs FROM 1 FOR 1) AS Prima\_Litera

FROM InformatiiProduse;

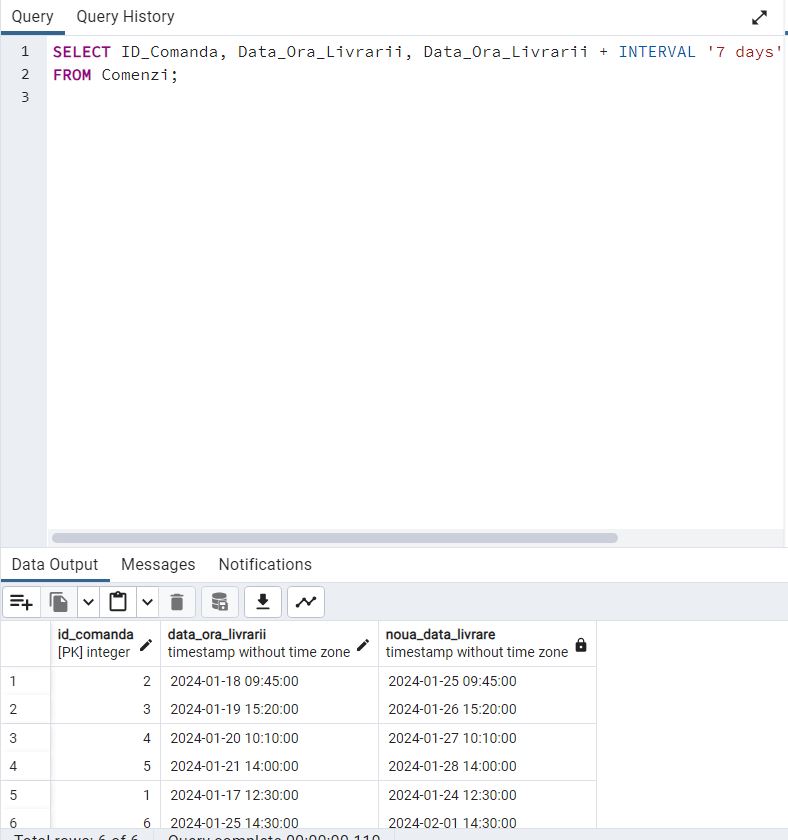


**Exemple de interogări folosind funcții pe date calendaristice:**

*Exemplu de adăugare a 7 zile la data livrării comenzilor:*

SELECT ID\_Comanda, Data\_Ora\_Livrarii, Data\_Ora\_Livrarii + INTERVAL '7 days' AS Noua\_Data\_Livrare

FROM Comenzi;



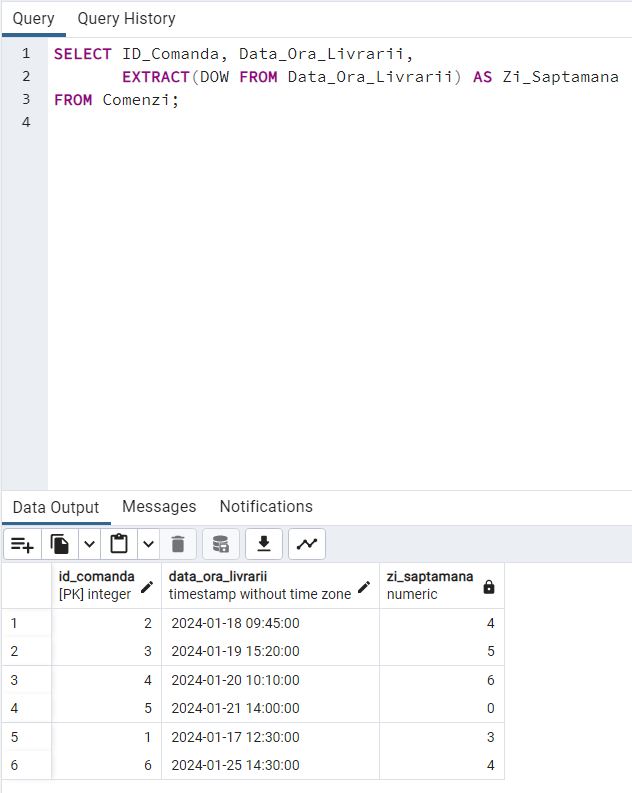
*Calcularea zilei săptămânii în care a fost înregistrată fiecare recenzie:*

*Această interogare utilizează funcția EXTRACT(DOW FROM Data\_Inregistrare), care extrage ziua săptămânii (0 pentru duminică, 1 pentru luni, etc.) din data de înregistrare a recenziilor. Rezultatul este afișat în coloana "Zi\_Saptamana".*

SELECT ID\_Comanda, Data\_Ora\_Livrarii,

EXTRACT(DOW FROM Data\_Ora\_Livrarii) AS Zi\_Saptamana

FROM Comenzi;



**Exemple de interogări folosind CASE:**

*Exemplu de utilizare a expresiei CASE pentru a evalua starea comenzilor:*

SELECT ID\_Comanda, Status\_Comanda,

CASE

WHEN Status\_Comanda = 'Finalizată' THEN 'Comanda Finalizata'

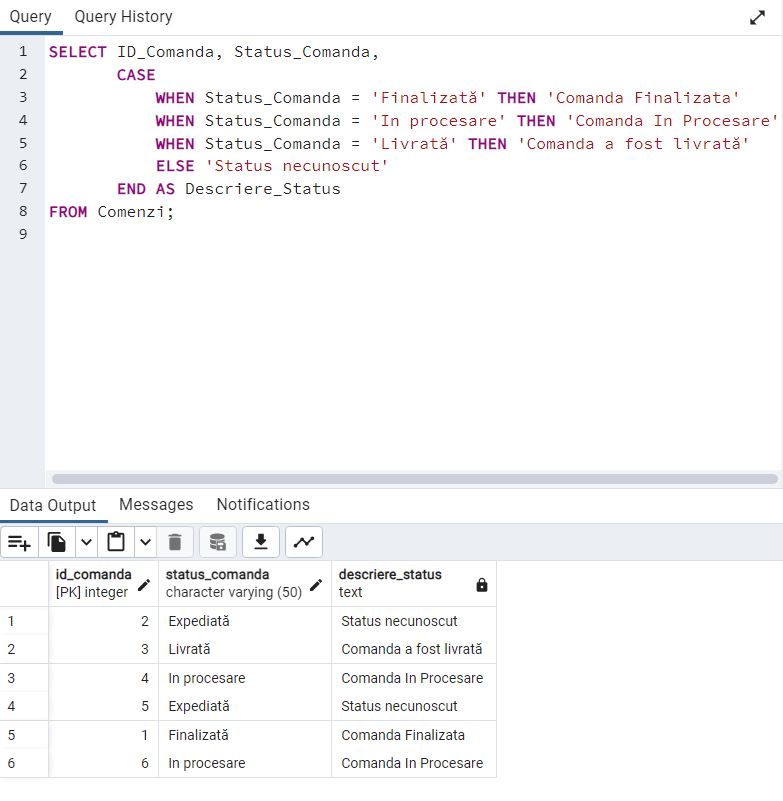
WHEN Status\_Comanda = 'In procesare' THEN 'Comanda In Procesare'

WHEN Status\_Comanda = 'Livrată' THEN 'Comanda a fost livrată'

ELSE 'Status necunoscut'

END AS Descriere\_Status

FROM Comenzi;



*Exemplu de utilizare a expresiei CASE pentru a clasifica produsele în funcție de stoc:*

SELECT Nume\_Produs, Stoc\_Disponibil,

CASE

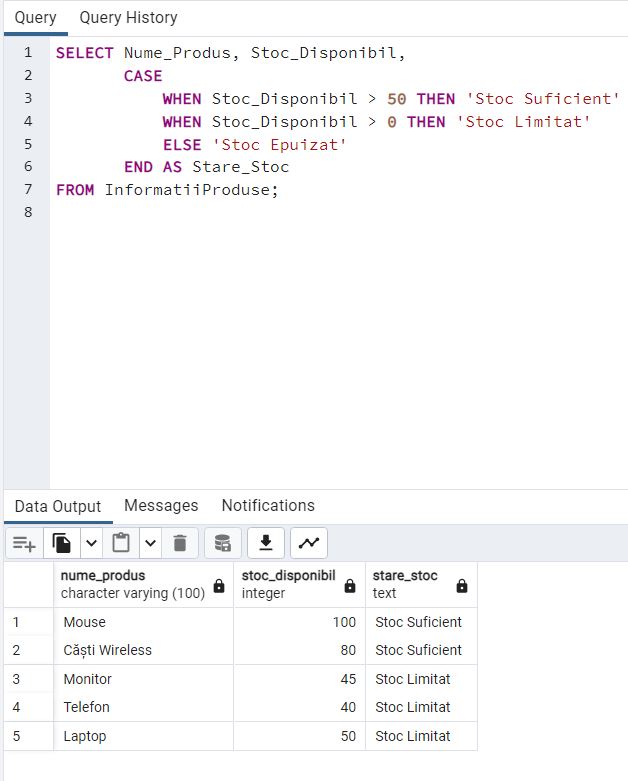
WHEN Stoc\_Disponibil > 50 THEN 'Stoc Suficient'

WHEN Stoc\_Disponibil > 0 THEN 'Stoc Limitat'

ELSE 'Stoc Epuizat'

END AS Stare\_Stoc

FROM InformatiiProduse;



**Vederi:**

*View pentru Lista Comenzilor:*

CREATE VIEW ListaComenzi AS

SELECT Comenzi.ID\_Comanda, Utilizatori.Nume, Utilizatori.Prenume, Comenzi.Status\_Comanda

FROM Comenzi

JOIN Utilizatori ON Comenzi.ID\_Utilizator = Utilizatori.ID\_Utilizator;

SELECT \* FROM ListaComenzi;--PERMISA

UPDATE ListaComenzi

SET Status\_Comanda = 'Finalizată'--NEPERMISA

WHERE ID\_Comanda = 1;

*View pentru Produsele în Stoc:*

CREATE VIEW ProduseInStoc AS

SELECT Nume\_Produs, Stoc\_Disponibil

FROM InformatiiProduse

WHERE Stoc\_Disponibil > 0;

UPDATE ProduseInStoc

SET Stoc\_Disponibil = Stoc\_Disponibil - 1 --PERMISA

WHERE Nume\_Produs = 'Laptop';

INSERT INTO ProduseInStoc (Nume\_Produs, Stoc\_Disponibil) --NEPERMISA

VALUES ('Cameră Foto', 5);

*View pentru Detaliile Recenziilor:*

CREATE VIEW DetaliiRecenziiComenzi AS

SELECT

DC.ID\_DetaliiComanda,

C.ID\_Comanda,

C.ID\_Utilizator,

U.Nume AS Nume\_Utilizator,

U.Prenume AS Prenume\_Utilizator,

C.Status\_Comanda,

C.Data\_Ora\_Livrarii,

C.Total\_Comanda,

RP.ID\_Recenzie,

RP.Comentariu,

RP.Nota\_Acordata

FROM

DetaliiComanda DC

JOIN Comenzi C ON DC.ID\_Comanda = C.ID\_Comanda

JOIN RecenziiProduse RP ON C.ID\_Utilizator = RP.ID\_Utilizator

JOIN Utilizatori U ON C.ID\_Utilizator = U.ID\_Utilizator;

SELECT \* FROM DetaliiRecenziiComenzi;-PERMISA

-- Exemplu de încercare de actualizare a vederii (nepermis)

UPDATE DetaliiRecenziiComenzi

SET Status\_Comanda = 'Finalizată'

WHERE ID\_Comanda = 1;

-- Exemplu de încercare de inserare în vedere (nepermis)

INSERT INTO DetaliiRecenziiComenzi (ID\_Comanda, Status\_Comanda)

VALUES (6, 'Noua Comanda');

-- Exemplu de încercare de ștergere din vedere (nepermis)

DELETE FROM DetaliiRecenziiComenzi

WHERE ID\_Comanda = 2;

**Index:**

*Crearea unui index compus pe coloanele utilizate în condițiile de căutare*

CREATE INDEX idx\_AdreseLivrare\_Comenzi

ON AdreseLivrare (Oras)

INCLUDE (Cod\_Postal);

*Include Cod\_Postal în index, dar nu este parte din cheia de căutare.*

*Acest index, numit idx\_AdreseLivrare\_Comenzi, acoperă coloana Oras din tabela AdreseLivrare și include coloana Cod\_Postal în index, dar nu este parte a cheii de căutare. Prin includerea coloanei Cod\_Postal în index, putem îmbunătăți performanța pentru situațiile în care această coloană este inclusă în SELECT sau în condiții de filtrare, fără a o face o cheie de căutare strictă.*