

Proiect Baze de Date

Gestiune Magazin Online

Măciucă Emma-Iulia

Grupa 141

Cuprins	2
1. Descrierea modelului real, a utilității acestuia și a regulilor de funcționare.	3
2. Prezentarea constrângerilor (restricții, reguli) impuse asupra modelului.	4
3. Descrierea entităților, incluzând precizarea cheii primare.	7
4. Descrierea relațiilor, incluzând precizarea cardinalității acestora.	10
5. Descrierea atributelor, incluzând tipul de date și eventualele constrângerile, valori implicite, valori posibile ale atributelor.	13
6. Realizarea diagramei entitate-relație corespunzătoare descrierii de la punctele 3-5.	21
7. 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.	21
8. Enumerarea schemelor relaționale corespunzătoare diagramei conceptuale proiectate la punctul 7.	22
9. Realizarea normalizării până la forma normală 3 (FN1-FN3).	23
10. Crearea unei secvențe ce va fi utilizată în inserarea înregistrărilor în tabele (punctul 11).....	26
11. 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 associative; maxim 30 de înregistrări în fiecare tabel).	28
12. Formulați în limbaj natural și implementați 5 cereri SQL complexe ce vor utiliza, în ansamblul lor, următoarele elemente:	66
13. Implementarea a 3 operații de actualizare și de suprimare a datelor utilizând subcereri.	76
14. Crearea unei vizualizări complexe. Dați un exemplu de operație LMD permisă pe vizualizarea respectivă și un exemplu de operație LMD nepermisă.....	79
15. Formulați în limbaj natural și implementați în SQL: o cerere ce utilizează operația outer-join pe minimum 4 tabele, o cerere ce utilizează operația division și o cerere care implementează analiza top-n. Observație: Cele 3 cereri sunt diferite de cererile de la exercițiul 12.	82
16. Optimizarea unei cereri, aplicând regulile de optimizare ce derivă din proprietățile operatorilor algebrei relaționale. Cererea va fi exprimată prin expresie algebraică, arbore algebraic și limbaj (SQL), atât anterior cât și ulterior optimizării.	86
17. a. Realizarea normalizării BCNF, FN4, FN5.....	87
b. Aplicarea denormalizării, justificând necesitatea acesteia.	87

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

Tema

Tema proiectului este o bază de date realizată pentru o aplicație de gestiune a unui magazin online, cu scopul de a facilita interacțiunea utilizatorilor și a angajaților cu funcționalitățile magazinului online.

Descriere

Baza de date este folositoare pentru a monitoriza și stoca detalii atât despre produsele disponibile, cât și despre utilizatori și istoricul de comenzi plasate în cadrul magazinului, precum și informații privind livrarea și plata.

Entitatea Utilizator stochează informații importante despre utilizatorii aplicatiei, precum date de contact și de autentificare. Entitatea Comanda urmărește comenzi plasate de utilizatori, iar entitatile Livrare și Plata retin informații despre metodele de livrare și plata ale unei comenzi. De asemenea, entitățile Curier și Masina_Livrare rețin informații suplimentare despre livrari. Entitatea Card stochează datele cardului cu care se efectuează plata, iar Adresa adresa unde se livrează comanda și se facturează chitanta de plată. Entitatile Cos_Cumpărături și Lista_Dorinte reprezintă funcționalități universale unui magazin online, unde utilizatorii pot retine produsele preferate. În final, entitatile Produs și Categorie stochează informații despre produsele disponibile în magazin.

Utilitate

Utilitatea acestei baze de date este de a îmbunătăți gestionarea stocului de produse din magazin și a comenzi plasate de către administratori, dar și pentru a ajuta utilizatorii înregistrati să își urmărească istoricul de comenzi plasate, datele personale vizibile din contul de utilizator, produsele favorite din lista de dorințe, dar și produsele adăugate în coșul de cumpărături.

Informațiile stocate în baza de date sunt folositoare atât pentru a asigura acces rapid și eficient la informații și a monitoriza performanța vânzărilor, cât și pentru personalizarea experienței utilizatorilor și menținerea securității și confidențialității datelor.

Functionalități

Funcționalitățile bazei de date includ gestiunea produselor, adăugarea, actualizarea și ștergerea informațiilor despre produse și categorii și a utilizatorilor, administrarea datelor personale ale utilizatorilor, inclusiv înregistrarea și autentificarea utilizatorilor.

De asemenea, aplicatia permite monitorizarea comenzi plasate, inclusiv stările comenzi și detaliile aferente și administrarea informațiilor despre metodele de plată și livrare, inclusiv detaliu despre curieri și vehicule de livrare.

In plus, coșul de cumpărături și lista de dorințe permite utilizatorilor să adauge și să gestioneze produsele din coșul de cumpărături și lista de dorințe.

Concluzie

În concluzie, baza de date proiectată pentru gestionarea unui magazin online facilitează optimizarea operațiunilor magazinului și îmbunătățirea experienței utilizatorilor. Prin stocarea și gestionarea eficientă a datelor despre produse, utilizatori, comenzi, plăți și livrări, baza de date asigură o funcționare optimă a magazinului online și permite luarea de decizii informate bazate pe date precise și actualizate.

2. Prezentarea constrângерilor (restricții, reguli) impuse asupra modelului.

1. Tabela UTILIZATOR

- id_utilizator este cheie primară și trebuie să fie unică pentru fiecare înregistrare din tabel
- nume trebuie să aibă lungimea maxima de 255 de caractere
- prenume trebuie să aibă lungimea maxima de 255 de caractere
- email trebuie să aibă lungimea maxima de 255 de caractere, să fie o valoare unică pentru fiecare înregistrare din tabel, de tip NOT NULL
- parola trebuie să aibă lungimea maxima de 255 de caractere, să fie o valoare unică pentru fiecare înregistrare din tabel, de tip NOT NULL
- numar_telefon trebuie să aibă o lungime de 10 caractere
- data_nastere trebuie să fie o valoare de tipul data

2. Tabela COMANDA

- id_comanda este cheie primară și trebuie să fie unică pentru fiecare înregistrare din tabel
- total_plata trebuie să fie o valoare pozitivă, de tip NOT NULL
- stare_comanda trebuie să fie una dintre valorile “în curs de procesare”, “efectuată”, “în curs de pregatire”
- data_comanda trebuie să fie o valoare de tipul data
- id_utilizator este cheie externă către tabela Utilizator și trebuie să se refere la un id_utilizator existent
- id_livrare este cheie externă către tabela Livrare, care trebuie să se refere la un id_livrare existent și să fie unică pentru fiecare înregistrare din tabel

3. Tabela PLATA

- id_plata este cheie primară și trebuie să fie unică pentru fiecare înregistrare din tabel
- valoare trebuie să fie o valoare pozitivă, de tip NOT NULL
- data_plata trebuie să fie o valoare de tipul data
- stare trebuie să fie una dintre valorile “în curs de procesare”, “efectuată”, “respinsă”
- id_card este cheie externă către tabela Card și trebuie să se refere la un id_card existent
- id_comanda este cheie externă către tabela Comanda și trebuie să se refere la un id_comanda existent

4. Tabela CARD

- id_card este cheie primară și trebuie să fie unică pentru fiecare înregistrare din tabel
- numar_card trebuie să aibă o lungime de 16 caractere și să fie o valoare de tip NOT NULL, unică
- nume_proprietar trebuie să aibă lungimea maxima de 255 de caractere
- CVV trebuie să aibă o lungime de 3 sau 4 caractere
- data_expirare trebuie să fie o valoare de tipul data
- id_utilizator este cheie externă către tabela Utilizator și trebuie să se refere la un id_utilizator existent

5. Tabela LIVRARE

- id_livrare este cheie primară și trebuie să fie unică pentru fiecare înregistrare din tabel
- metoda_livrare trebuie să aibă lungimea maxima de 255 de caractere
- pret_livrare trebuie să fie un număr pozitiv
- stare trebuie să fie una dintre valorile “în curs de procesare”, “efectuata”, “în curs de livrare”
- data_livrare trebuie să fie o valoare de tip data NOT NULL
- id_adresa este cheie externă către tabela Adresa și trebuie să se refere la un id_adresa existent
- id_curier este cheie externă către tabela Curier și trebuie să se refere la un id_curier existent
- id_masina este cheie externă către tabela Masina_Livrare și trebuie să se refere la un id_masina existent

6. Tabela ADRESA

- id_adresa este cheie primară și trebuie să fie unică pentru fiecare înregistrare din tabel
- strada trebuie să aibă o lungime maxima de 255 de caractere și să fie o valoare de tip NOT NULL
- oras trebuie să aibă o lungime maxima de 255 de caractere și să fie o valoare de tip NOT NULL
- tara trebuie să aibă o lungime maxima de 255 de caractere și să fie o valoare de tip NOT NULL

- cod_zip trebuie sa aibă o lungime maxima de 10 caractere si sa fie o valoare de tip NOT NULL

7. Tabela LISTA_DORINTE

- id_lista este cheie primară si trebuie sa fie unică pentru fiecare înregistrare din tabel
- id_utilizator este cheie externă către tabela Utilizator si trebuie sa se refere la un id_utilizator existent
- denumire_lista trebuie sa aibă o lungime maxima de 255 de caractere si sa fie o valoare de tip NOT NULL

8. Tabela COS_CUMPARATURI

- id_cos este cheie primară si trebuie sa fie unică pentru fiecare înregistrare din tabel
- id_utilizator este cheie externă către tabela Utilizator, care trebuie sa se refere la un id_utilizator existent si sa fie unică pentru fiecare înregistrare din tabel
- total_cos trebuie sa fie o valoare pozitiva

9. Tabela PRODUS

- id_produs este cheie primară si trebuie sa fie unică pentru fiecare înregistrare din tabel
- denumire_produs trebuie sa aibă lungimea maxima de 255 de caractere
- pret trebuie sa fie o valoare NOT NULL, pozitivă
- stoc trebuie sa fie o valoare pozitiva
- id_categorie este cheie externă către tabela Categorie si trebuie sa se refere la un id_categorie existent

10. Tabela CATEGORIE

- id_categorie este cheie primară si trebuie sa fie unică pentru fiecare înregistrare din tabel
- denumire_categoria trebuie sa aibă lungimea maxima de 255 de caractere si sa nu fie o valoare nula

11. Tabela PRODUS_COMANDA

- id_produs, id_comanda - formează cheia primară a tabelului, trebuie sa fie unică pentru fiecare înregistrare din tabel. Id_produs si id_comanda fac referire la cheile primare ale tabelelor Produs și Comanda

12. Tabela PRODUS_LISTA

- id_producător, id_listă - formează cheia primară a tabelului, trebuie să fie unică pentru fiecare înregistrare din tabel. Id_producător și id_listă fac referire la cheile primare ale tabelelor Producător și Listă_Dorinte

13. Tabela PRODUS_COS

- id_producător, id_cos - formează cheia primară a tabelului, trebuie să fie unică pentru fiecare înregistrare din tabel. Id_producător și id_cos fac referire la cheile primare ale tabelelor Producător și Cos_Cumpărături

14. Tabela CURIER

- id_curier este cheie primară și trebuie să fie unică pentru fiecare înregistrare din tabel
- nume trebuie să aibă lungimea maxima de 255 de caractere și să fie NOT NULL
- prenume trebuie să aibă lungimea maxima de 255 de caractere și să fie NOT NULL
- salariu trebuie să fie o valoare numerică, pozitivă

15. Tabela MASINA_LIVRARE

- id_masina este cheie primară și trebuie să fie unică pentru fiecare înregistrare din tabel
- model_masina trebuie să aibă lungimea maxima de 255 de caractere
- numar_inmatriculare trebuie să aibă lungimea maxima de 255 de caractere și să fie NOT NULL

3. Descrierea entităților, incluzând precizarea cheii primare.

1. Entitatea UTILIZATOR

Tabela UTILIZATOR stochează informații despre utilizatorii magazinului. Fiecare înregistrare este identificată în mod unic prin cheia primară id_utilizator și conține informații despre numele și prenumele utilizatorului, email-ul și parola folosite la înregistrare, un număr de telefon și data nașterii. Informațiile sunt utile atât pentru utilizatori, pentru a putea vedea datele personale și a le modifica, cât și pentru administratorii magazinului, pentru a putea contacta și a identifica utilizatorii magazinului online.

2. Entitatea COMANDĂ

Tabela COMANDA stochează detaliile comenziilor plasate în cadrul magazinului. Înregistrările sunt identificate prin cheia primară id_comanda și contin informații precum totalul de plată, data la care s-a plasat comanda, id-ul utilizatorului care a plasat comanda prin cheia externă id_utilizator care conectează tabela COMANDA la tabela UTILIZATOR și detaliile despre livrare, date de cheia externă id_livrare, care creează legătură cu tabela LIVRARE. Aceasta entitate este utilă pentru a ajuta utilizatorii să își accesize istoricul de comenzi și pentru a tine evidența tuturor comenziilor plasate în magazin.

3. Entitatea **PLATA**

Tabela PLATA conține informații despre metoda de plată a unei comenzi. Înregistrările sunt identificate prin cheia primară id_plata și stochează detalii despre valoarea platii, data la care s-a efectuat plată și starea în care se află, cardul asociat platii, ce leagă tabela cu entitatea CARD prin cheia externă id_card și comanda pentru care s-a efectuat plată prin cheia externă id_comanda, ce realizează legătură cu entitatea COMANDA. Entitatea este importantă pentru administratorii afacerii, pentru a păstra istoricul platilor făcute în cadrul magazinului.

4. Entitatea **CARD**

Tabela CARD stochează informațiile unui card bancar asociat unui utilizator, legătură cu entitatea UTILIZATOR realizată prin cheia externă id_utilizator. Tabela conține detalii standard ale unui card, precum numărul cardului, codul CVV, numele proprietarului și data de expirare. Înregistrările sunt identificate prin cheia primară id_card.

5. Entitatea **LIVRARE**

Tabela LIVRARE conține detalii despre livrarea unei comenzi. Cheia primară id_livrare asigura identificarea înregistrărilor. Entitatea stochează metoda prin care se efectuează livrarea, costul, starea și data estimativă livrării, dar și adresa la care se va livra coletul, data de cheia externă id_adresa, ce leagă tabela de entitatea ADRESA. În plus, cunoaștem detalii despre curierul și mașina cu care se va livra comanda, prin cheile externe id_curier și id_masina. Utilitatea acestei entități constă în asigurarea unui proces eficient și sigur de a livra rapid comenziile clienților.

6. Entitatea **ADRESA**

Tabela ADRESA stochează detalii importante despre adresa unde se va efectua livrarea unei comenzi. Înregistrările tablei sunt identificate prin cheia primară id_adresa și conțin informații precum strada, orașul, țara și codul ZIP al adresei.

7. Entitatea **LISTA_DORINTE**

Tabela LISTA_DORINTE exemplifică o funcționalitate importantă a unui magazin online, accea de a salva produsele favorite într-o lista de dorințe. Un utilizator poate crea mai multe

liste de dorințe în contul personal, atribuind un nume pentru fiecare. Înregistrările sunt identificate în mod unic prin cheia primară id_listă, iar cheia externe id_utilizator asigură legătură cu tabela UTILIZATOR.

8. Entitatea **COS_CUMPARATURI**

Tabela COS_CUMPARATURI asigură o altă funcționalitate a unui magazin online, accea de a adăuga produsele dorințe în coșul de cumpărături, urmând ca la final să se plaseze o comandă. Tabela stochează cheia primară id_cos, cheia externă id_utilizator care leagă tabela de entitatea UTILIZATOR și totalul de plată.

9. Entitatea **PRODUS**

Tabela PRODUS conține informații despre produsele disponibile în magazin. Fiecare produs este identificat în mod unic prin cheia primară id_produs, iar tabela stochează denumirea produsului, prețul, numărul de articole în stoc și categoria din care face parte, data de cheia externă id_categorie, care leagă tabela PRODUS de tabela CATEGORIE. Aceasta entitate este importantă pentru gestionarea eficientă a produselor disponibile în magazin.

10. Entitatea **CATEGORIE**

Entitatea CATEGORIE stochează principalele categorii din care fac parte produsele din magazin, pentru a ajuta la gestiunea produselor de către angajați și pentru a facilita navigarea prin articolele disponibile. Tabela asigură identificarea înregistrărilor prin cheia primară id_categorie și conține denumirea categoriei.

11. Entitatea **PRODUS_COMANDA**

Tabela PRODUS_COMANDA este un tabel intermediar dat de relația many-to-many dintre tabelele PRODUS și COMANDA și stochează produsele conținute într-o comandă. Aceasta conține 2 chei externe, id_produs și id_comanda, care fac legătură cu tabelul PRODUS și COMANDA, iar cheia primară este compusă din cele 2 atrbute.

12. Entitatea **PRODUS_LISTA**

Tabela PRODUS_LISTA constituie un tabel intermediar într-o relație many-to-many între tabelele PRODUS și LISTA și stochează produsele dintr-o lista de dorințe creată de un utilizator. Tabela conține 2 chei externe, id_produs și id_listă, care fac legătură cu tabelul PRODUS și LISTA, iar cheia primară este compusă din cele 2 atrbute.

13. Entitatea **PRODUS_COS**

Tabela PRODUS_COS este un tabel intermediar dat de relația many-to-many dintre tabelele PRODUS și COS_CUMPARATURI și stochează produsele din coșul de cumpărături al unui

utilizator. Tabela conține 2 chei externe, id_produs și id_cos, care fac legătură cu tabelul PRODUS și COS, iar cheia primară este compusă din cele 2 attribute.

14. Entitatea **CURIER**

Tabela CURIER care stochează informații despre curierii care efectuează livrările, pentru a ajuta la gestionarea mai eficientă și mai sigură a comenziilor, în special în cazul posibilelor probleme întâmpinate în procesul de livrare. Aceasta conține id-ul unic al unui curier, numele, prenumele, cât și salariul.

15. Entitatea **MASINA_LIVRARE**

Tabela MASINA_LIVRARE conține informații despre mașinile cu care se efectuează livrările și stochează id-ul mașinii, modelul și numărul de înmatriculare. Aceasta entitate oferă siguranță unui colet aflat în curs de livrare, știind exact unde se află din momentul în care pleacă din depozitul furnizorului pana ajunge la destinatar.

4. Descrierea relațiilor, incluzând precizarea cardinalității acestora.

- Tabela UTILIZATOR are o relație “one-to-many” cu tabela CARD, deoarece unui utilizator i se pot asocia mai multe carduri (sau niciunul), dar un card este asociat unui singur utilizator. Așadar cardinalitatea este 1:M(0)
- Tabela UTILIZATOR are o relație “one-to-many” cu tabela COMANDA, deoarece un utilizator poate plasa mai multe comenzi (sau niciuna), dar o comandă este asociată unui singur utilizator (obligatoriu). Așadar cardinalitatea este 1:M(0)
- Tabela UTILIZATOR are o relație “one-to-many” cu tabela LISTA_DORINTE, deoarece un utilizator poate avea mai multe liste de dorințe (sau niciuna), în funcție de preferințele sale, dar o lista de dorințe este asociată unui singur utilizator (obligatoriu). Așadar cardinalitatea este 1:M(0)
- Tabela UTILIZATOR are o relație “one-to-one” cu tabela COS_CUMPARATURI, deoarece un utilizator poate avea un singur cos de cumpărături aferent contului de utilizator, iar un cos de cumpărături este asociat unui singur utilizator. Așadar cardinalitatea este 1:1
- Tabela COMANDA are o relație de tip “one-to-one” cu tabela LIVRARE, deoarece o livrare este asociată cu o comandă unică, iar o comandă poate fi livrată o singura dată. Așadar cardinalitatea este 1:1
- Tabela COMANDA are o relație de tip “one-to-one” cu tabela UTILIZATOR, relație descrisă în secțiunea acesteia
- Tabela COMANDA are o relație de tip “one-to-many” cu tabela PLATA, deoarece o plată corespunde unei singure comenzi, dar la o comandă pot fi încercate mai multe plăti, în caz ca una dintre acestea eșuează. Așadar cardinalitatea este 1:M(1)

- Tabela COMANDA se află într-o relație de tip “many-to-many” cu tabela PRODUSE, încât o comandă poate conține mai multe produse (cel puțin unul), iar un produs poate fi comandat de mai multe ori (sau niciodată). Așadar, cardinalitatea este M(0):M(1). Petru a gestiona relația “many-to-many” se va implementa tabelul intermediu PRODUS_COMANDA
- Tabela PLATA are o relație de tip “many-to-one” cu tabela COMANDA, relate descrisa in secțiunea acesteia
- Tabela PLATA are o relație de tip “many-to-one” cu tabela CARD, deoarece o plată se poate efectua cu un singur card, dar un card poate efectua mai multe plăti(cel puțin una). Așadar cardinalitatea este M(1):1
- Tabela CARD are o relație de tip “one-to-many” cu tabela PLATA, relate descrisa in secțiunea acesteia
- Tabela CARD are o relație de tip “many-to-one” cu tabela UTILIZATOR, relate descrisa in secțiunea acesteia
- Tabela LIVRARE are o relație are o relație de tip “one-to-one” cu tabela COMANDA, relatie descrisa in secțiunea acesteia
- Tabela LIVRARE are o relație de tip “many-to-one” cu tabela ADRESA, deoarece o livrare se face la o singura adresă, dar la o adresă se pot face mai multe livrări (sau niciuna), livrările având comenzi diferite. Prin urmare, cardinalitatea este M(0):1
- Tabela LIVRARE are o relație de tip “many-to-one” cu tabela CURIER, deoarece o livrare este efectuată de un singur curier, dar un curier poate afectua mai multe livrări (sau niciuna). Prin urmare, cardinalitatea este M(0):1
- Tabela LIVRARE are o relație de tip “many-to-one” cu tabela MASINA_LIVRARE, deoarece o livrare se efectuează cu o singura mașină, dar o mașină poate realiza mai multe livrări (sau niciuna). Prin urmare, cardinalitatea este M(0):1
- Tabela ADRESA are o relație de tip “one-to-many” cu tabela LIVRARE, relatia descrisa in secțiunea acesteia
- Tabela LISTA_DORINTE se află într-o relație de tip “many-to-one” cu tabela UTILIZATOR, relatia descrisa in secțiunea acesteia
- Tabela LISTA_DORINTE se află într-o relație de tip “many-to-many” cu tabela PRODUS, încât o lista de dorințe poate conține mai multe produse (sau niciunul), iar un produs poate fi adăugat în mai multe liste de dorințe diferite (sau niciuna). Așadar, cardinalitatea este M(0):M(0). Petru a gestiona relația “many-to-many” se va implementa tabelul intermediu PRODUS_LISTA
- Tabela COS_CUMPARATURI se află într-o relație de tip “many-to-one” cu tabela UTILIZATOR, relatia descrisa in secțiunea acesteia

- Tabela COS_CUMPARATURI se afla într-o relație de tip “many-to-many” cu tabela PRODUS, încât un cos de cumpărături poate conține mai multe produse (sau niciunul), iar un produs poate fi adăugat în mai multe coșuri de cumpărături diferite (sau niciuna). Așadar, cardinalitatea este M(0):M(0). Petru a gestiona relația “many-to-many” se va implementa tabelul intermediu PRODUS_COS
- Tabela PRODUS se afla într-o relație de tip “many-to-many” cu tabela COMANDA, relație descrisă în secțiunea acesteia
- Tabela PRODUS se afla într-o relație de tip “many-to-many” cu tabela LISTA_DORINTE, relație descrisă în secțiunea acesteia
- Tabela PRODUS se afla într-o relație de tip “many-to-many” cu tabela COS_CUMPARATURI, relație descrisă în secțiunea acesteia
- Tabela PRODUS se afla într-o relație de tip “many-to-one” cu tabela CATEGORIE, deoarece un produs face parte dintr-o singură categorie de articole, dar o categorie poate conține mai multe produse (sau niciunul). Așadar cardinalitatea este M(0):1
- Tabela CATEGORIE se afla într-o relație de tip “one-to-many” cu tabela PRODUS, relație descrisă în secțiunea acesteia
- Tabela PRODUS_COMANDA este un tabel intermediu, utilizat pentru gestionarea relației “many-to-many” dintre tabelele PRODUS și COMANDA. Tabelul are o relație “many-to-one” atât cu tabela PRODUS, cât și cu tabela COMANDA, ce indică faptul că un produs din PRODUS_COMANDA este asociat unei comenzi unice. Cardinalitatea este M(1):1, cu tabela COMANDA și M(0):1 cu tabela PRODUS
- Tabela PRODUS_LISTA este un tabel intermediu, utilizat pentru gestionarea relației “many-to-many” dintre tabelele PRODUS și LISTA_DORINTE. Tabelul are o relație “many-to-one” atât cu tabela PRODUS, cât și cu tabela LISTA_DORINTE, ce indică faptul că un produs din PRODUS_LISTA este asociat unei liste unice. În ambele cazuri, cardinalitatea este M(0):1
- Tabela PRODUS_COS este un tabel intermediu, utilizat pentru gestionarea relației “many-to-many” dintre tabelele PRODUS și COS_CUMPARATURI. Tabelul are o relație “many-to-one” atât cu tabela PRODUS, cât și cu tabela COS_CUMPARATURI, ce indică faptul că un produs din PRODUS_COS este asociat unei liste unice. În ambele cazuri, cardinalitatea este M(0):1
- Tabela CURIER are o relație de tip “one-to-many” cu tabela LIVRARE, relație descrisă în secțiunea acesteia
- Tabela MASINA_LIVRARE are o relație de tip “one-to-many” cu tabela LIVRARE, relație descrisă în secțiunea acesteia

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

1. Tabela UTILIZATOR

- id_utilizator (varchar(255)): cheie primară
- nume (varchar(255)): reprezintă numele de familie al utilizatorului, are lungimea maxima de 255 de caractere
- prenume (varchar(255)): reprezintă prenumele utilizatorului, are lungimea maxima de 255 de caractere
- email (varchar(255)): reprezintă adresa de mail cu care se înregistrează utilizatorul, de lungime maxima de 255 de caractere, unica, de tip NOT NULL
- parola (varchar(255)): reprezintă parola aleasa la înregistrare, de lungime maxima de 255 de caractere, de tip NOT NULL
- numar_telefon (varchar(10)): reprezinta numărul de telefon al utilizatorului, trebuie sa aibă lungimea fixa de 10 caractere
- data_nastere(date): data de naștere a utilizatorului, stocata sub forma de dată

COLUMN_NAME	DATA_TYPE	NULLABLE	DATA_DEFAULT	COLUMN_ID	COMMENTS
1 ID_UTILIZATOR	NUMBER(38,0)	No	"UTILIZATOR"."UTILIZATOR_SEQ"."NEXTVAL"	1	(null)
2 NUME	VARCHAR2(255 BYTE)	Yes	(null)	2	(null)
3 PRENUME	VARCHAR2(255 BYTE)	Yes	(null)	3	(null)
4 EMAIL	VARCHAR2(255 BYTE)	No	(null)	4	(null)
5 PAROLA	VARCHAR2(255 BYTE)	No	(null)	5	(null)
6 NUMAR_TELEFON	VARCHAR2(10 BYTE)	Yes	(null)	6	(null)
7 DATA_NASTERE	DATE	Yes	(null)	7	(null)

2. Tabela COMANDA

- id_comanda: cheie primară
- total_plata (number(10,2)): reprezintă totalul de plata al unei comenzi si stochează o valoare numerica pozitiva, cu 2 zecimale, de tip NOT NULL
- stare_comanda (varchar(255)): reprezintă starea în care se afla comanda și trebuie sa fie una dintre valorile “în curs de procesare”, “efectuată”, “in curs de pregatire”
- data_comanda (date): data plasarii comenzii, stocata sub forma de data
- id_utilizator este cheie externă către tabela Utilizator
- id_livrare este cheie externă către tabela Livrare

The screenshot shows the Oracle SQL Developer interface with the title bar "Oracle SQL Developer : Table UTILIZATOR.COMANDA@tutorial". The menu bar includes File, Edit, View, Navigate, Run, Team, Tools, Window, and Help. Below the menu is a toolbar with various icons. The main area shows a connection named "tutorial" and a table named "COMANDA". The table structure is displayed in a grid with columns: COLUMN_NAME, DATA_TYPE, NULLABLE, DATA_DEFAULT, COLUMN_ID, and COMMENTS. The data rows are:

COLUMN_NAME	DATA_TYPE	NULLABLE	DATA_DEFAULT	COLUMN_ID	COMMENTS
1 ID_COMANDA	NUMBER(38,0)	No	"UTILIZATOR"."COMANDA_SEQ"."NEXTVAL"	1	(null)
2 TOTAL_PLATA	NUMBER(10,2)	No	(null)	2	(null)
3 STARE_COMANDA	VARCHAR2(255 BYTE)	Yes	(null)	3	(null)
4 DATA_COMANDA	DATE	Yes	(null)	4	(null)
5 ID_UTILIZATOR	NUMBER(38,0)	Yes	(null)	5	(null)
6 ID_LIVRARE	NUMBER(38,0)	Yes	(null)	6	(null)

3. Tabela PLATA

- id_plata: cheie primă
- valoare (number(10,2)): reprezintă totalul de plată și stochează o valoare numerică pozitivă, cu 2 zecimale, de tip NOT NULL
- data_plata (date): data efectuării platii, stocată sub forma de dată
- stare (varchar(255)): reprezintă starea în care se află plată și trebuie să fie una dintre valorile “în curs de procesare”, “efectuată”, “respinsă”
- id_card este cheie externă către tabela Card
- id_comanda este cheie externă către tabela Comanda

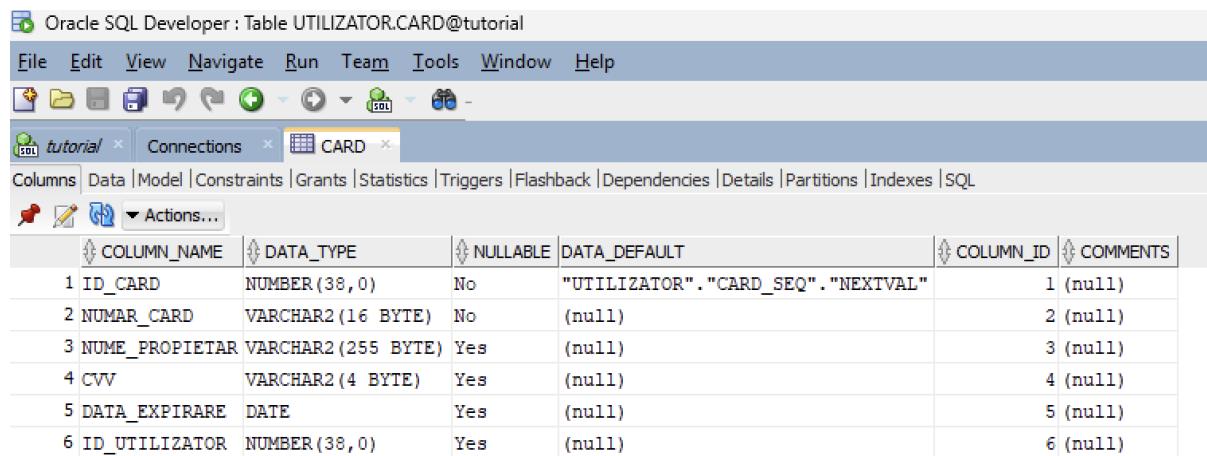
The screenshot shows the Oracle SQL Developer interface with the title bar "Oracle SQL Developer : Table UTILIZATOR.PLATA@tutorial". The menu bar includes File, Edit, View, Navigate, Run, Team, Tools, Window, and Help. Below the menu is a toolbar with various icons. The main area shows a connection named "tutorial" and a table named "PLATA". The table structure is displayed in a grid with columns: COLUMN_NAME, DATA_TYPE, NULLABLE, DATA_DEFAULT, COLUMN_ID, and COMMENTS. The data rows are:

COLUMN_NAME	DATA_TYPE	NULLABLE	DATA_DEFAULT	COLUMN_ID	COMMENTS
1 ID_PLATA	NUMBER(38,0)	No	"UTILIZATOR"."PLATA_SEQ"."NEXTVAL"	1	(null)
2 VALOARE	NUMBER(10,2)	No	(null)	2	(null)
3 DATA_PLATA	DATE	Yes	(null)	3	(null)
4 STARE	VARCHAR2(255 BYTE)	Yes	(null)	4	(null)
5 ID_CARD	NUMBER(38,0)	Yes	(null)	5	(null)
6 ID_COMANDA	NUMBER(38,0)	Yes	(null)	6	(null)

4. Tabela CARD

- id_card: cheie primă
- numar_card (varchar(16)): reprezintă numărul cardului, de lungime fixă de 16 caractere, valoare de tip NOT NULL, unică
- nume_proprietar (varchar(255)): reprezintă numele proprietarului cardului, are lungimea maximă de 255 de caractere

- CVV (varchar(255)): reprezintă codul de securitate de pe spatele cardului, are o lungime de 3 sau 4 caractere
- data_expirare (date): data expirării cardului, stocată sub forma de dată
- id_utilizator este cheie externă către tabela Utilizator



The screenshot shows the Oracle SQL Developer interface with the title bar "Oracle SQL Developer : Table UTILIZATOR.CARD@tutorial". The menu bar includes File, Edit, View, Navigate, Run, Team, Tools, Window, and Help. Below the menu is a toolbar with various icons. The main window shows a connection named "tutorial" and a table named "CARD". The table structure is displayed in a grid with the following columns:

COLUMN_NAME	DATA_TYPE	NULLABLE	DATA_DEFAULT	COLUMN_ID	COMMENTS
1 ID_CARD	NUMBER(38,0)	No	"UTILIZATOR"."CARD_SEQ"."NEXTVAL"	1	(null)
2 NUMAR_CARD	VARCHAR2(16 BYTE)	No	(null)	2	(null)
3 NUME_PROPIETAR	VARCHAR2(255 BYTE)	Yes	(null)	3	(null)
4 CVV	VARCHAR2(4 BYTE)	Yes	(null)	4	(null)
5 DATA_EXPIRARE	DATE	Yes	(null)	5	(null)
6 ID_UTILIZATOR	NUMBER(38,0)	Yes	(null)	6	(null)

5. Tabela LIVRARE

- id_livrare: cheie primară
- metoda_livrare (varchar(255)): reprezintă metoda prin care se va efectua livrarea, are lungimea maxima de 255 de caractere
- pret_livrare (number(10,2)): reprezintă cât costa livrarea și stochează o valoare numerică pozitiva, cu 2 zecimale
- stare (varchar(255)): reprezintă starea în care se află livrarea și trebuie să fie una dintre valorile “în curs de procesare”, “efectuata”, “în curs de livrare”
- data_livrare (date): data la care se va efectua livrarea, stocată sub forma de dată, NOT NULL
- id_adresa este cheie externă către tabela Adresa

The screenshot shows the Oracle SQL Developer interface with the title bar "Oracle SQL Developer : Table UTILIZATOR.LIVRARE@tutorial". The menu bar includes File, Edit, View, Navigate, Run, Team, Tools, Window, and Help. Below the menu is a toolbar with various icons. The main window shows a connection named "tutorial" and a table named "LIVRARE". The "Columns" tab is selected, displaying the following table structure:

COLUMN_NAME	DATA_TYPE	NULLABLE	DATA_DEFAULT	COLUMN_ID	COMMENTS
1 ID_LIVRARE	NUMBER(38,0)	No	"UTILIZATOR"."LIVRARE_SEQ"."NEXTVAL"	1	(null)
2 METODA_LIVRARE	VARCHAR2(255 BYTE)	No	(null)	2	(null)
3 PRET_LIVRARE	NUMBER(10,2)	Yes	(null)	3	(null)
4 STARE	VARCHAR2(255 BYTE)	No	(null)	4	(null)
5 DATA_LIVRARE	DATE	Yes	(null)	5	(null)
6 ID_ADRESA	NUMBER(38,0)	Yes	(null)	6	(null)
7 ID_CURIER	NUMBER(38,0)	Yes	(null)	7	(null)
8 ID_MASINA	NUMBER(38,0)	Yes	(null)	8	(null)

6. Tabela ADRESA

- id_adresa: cheie primară
- strada (varchar(255)): reprezintă strada adresei la care se va efectua livrarea, are lungimea maxima de 255 de caractere, NOT NULL
- oras(varchar(255)): reprezintă orasul adresei la care se va efectua livrarea, are lungimea maxima de 255 de caractere, NOT NULL
- tara (varchar(255)): reprezintă tara adresei la care se va efectua livrarea, are lungimea maxima de 255 de caractere, NOT NULL
- cod_zip (varchar(10)): reprezintă codul ZIP al adresei la care se va efectua livrarea, are lungimea maxima de 10 caractere, NOT NULL

The screenshot shows the Oracle SQL Developer interface with the title bar "Oracle SQL Developer : Table UTILIZATOR.ADRESA@tutorial". The menu bar includes File, Edit, View, Navigate, Run, Team, Tools, Window, and Help. Below the menu is a toolbar with various icons. The main window shows a connection named "tutorial" and a table named "ADRESA". The "Columns" tab is selected, displaying the following table structure:

COLUMN_NAME	DATA_TYPE	NULLABLE	DATA_DEFAULT	COLUMN_ID	COMMENTS
1 ID_ADRESA	NUMBER(38,0)	No	"UTILIZATOR"."ADRESA_SEQ"."NEXTVAL"	1	(null)
2 STRADA	VARCHAR2(255 BYTE)	No	(null)	2	(null)
3 ORAS	VARCHAR2(255 BYTE)	No	(null)	3	(null)
4 TARA	VARCHAR2(255 BYTE)	No	(null)	4	(null)
5 COD_ZIP	VARCHAR2(10 BYTE)	No	(null)	5	(null)

7. Tabela LISTA_DORINTE

- id_lista: cheie primară
- id_utilizator este cheie externă către tabela Utilizator

- denumire_lista (varchar(255)): reprezintă denumirea listei de dorințe, are lungimea maxima de 255 de caractere, NOT NULL

The screenshot shows the Oracle SQL Developer interface with the title bar "Oracle SQL Developer : Table UTILIZATOR.LISTA_DORINTE@tutorial". The menu bar includes File, Edit, View, Navigate, Run, Team, Tools, Window, and Help. Below the menu is a toolbar with various icons. The main area shows a connection named "tutorial" and a table named "LISTA_DORINTE". The table structure is displayed in a grid:

COLUMN_NAME	DATA_TYPE	NULLABLE	DATA_DEFAULT	COLUMN_ID	COMMENTS
1 ID_LISTA	NUMBER(38,0)	No	"UTILIZATOR"."LISTA_DORINTE_SEQ"."NEXTVAL"	1	(null)
2 ID_UTILIZATOR	NUMBER(38,0)	Yes	(null)	2	(null)
3 DENUMIRE_LISTA	VARCHAR2(255 BYTE)	No	(null)	3	(null)

8. Tabela COS_CUMPARATURI

- id_cos: cheie primară
- id_utilizator este cheie externă către tabela Utilizator
- total_cos (number(10,2)): reprezintă totalul produselor din cos și stochează o valoare numerică pozitivă, cu 2 zecimale

The screenshot shows the Oracle SQL Developer interface with the title bar "Oracle SQL Developer : Table UTILIZATOR.COS_CUMPARATURI@tutorial". The menu bar includes File, Edit, View, Navigate, Run, Team, Tools, Window, and Help. Below the menu is a toolbar with various icons. The main area shows a connection named "tutorial" and a table named "COS_CUMPARATURI". The table structure is displayed in a grid:

COLUMN_NAME	DATA_TYPE	NULLABLE	DATA_DEFAULT	COLUMN_ID	COMMENTS
1 ID_COS	NUMBER(38,0)	No	"UTILIZATOR"."COS_CUMPARATURI_SEQ"."NEXTVAL"	1	(null)
2 ID_UTILIZATOR	NUMBER(38,0)	Yes	(null)	2	(null)
3 TOTAL_COS	NUMBER(10,2)	Yes	(null)	3	(null)

9. Tabela PRODUS

- id_produs: cheie primară
- denumire_produs (varchar(255)): reprezintă denumirea produsului, are lungimea maxima de 255 de caractere
- pret (number(10,2)): reprezintă cât costa produsul și stochează o valoare numerică pozitivă, cu 2 zecimale, NOT NULL
- stoc (number(10)): reprezintă număr de articole în stoc și stochează o valoare numerică pozitivă întreagă
- id_categorie este cheie externă către tabela Categorie

The screenshot shows the Oracle SQL Developer interface with the title bar "Oracle SQL Developer : Table UTILIZATOR.PRODUS@tutorial". The menu bar includes File, Edit, View, Navigate, Run, Team, Tools, Window, and Help. The toolbar has icons for New Connection, Open Connection, Save, Undo, Redo, and others. The connections tab shows "tutorial" is selected. The table structure for "PRODUS" is displayed in the main pane. The columns are:

COLUMN_NAME	DATA_TYPE	NULLABLE	DATA_DEFAULT	COLUMN_ID	COMMENTS
1 ID_PRODUS	NUMBER(38,0)	No	"UTILIZATOR"."PRODUS_SEQ"."NEXTVAL"	1	(null)
2 DENUMIRE_PRODUS	VARCHAR2(255 BYTE)	Yes	(null)	2	(null)
3 PRET	NUMBER(10,2)	No	(null)	3	(null)
4 STOC	NUMBER(10,0)	Yes	(null)	4	(null)
5 ID_CATEGORIE	NUMBER(38,0)	Yes	(null)	5	(null)

10. Tabela CATEGORIE

- id_categorie: cheie primară
- denumire_categorie(varchar(255)): reprezintă denumirea categoriei, are lungimea maxima de 255 de caractere, NOT NULL

The screenshot shows the Oracle SQL Developer interface with the title bar "Oracle SQL Developer : Table UTILIZATOR.CATEGORIE@tutorial". The menu bar includes File, Edit, View, Navigate, Run, Team, Tools, Window, and Help. The toolbar has icons for New Connection, Open Connection, Save, Undo, Redo, and others. The connections tab shows "tutorial" is selected. The table structure for "CATEGORIE" is displayed in the main pane. The columns are:

COLUMN_NAME	DATA_TYPE	NULLABLE	DATA_DEFAULT	COLUMN_ID	COMMENTS
1 ID_CATEGORIE	NUMBER(38,0)	No	"UTILIZATOR"."CATEGORIE_SEQ"."NEXTVAL"	1	(null)
2 DENUMIRE_CATEGORIE	VARCHAR2(255 BYTE)	No	(null)	2	(null)

11. Tabela PRODUS_COMANDA

- id_produs, id_comanda: formează cheia primară a tabelului. Id_produs și id_comanda fac referire la cheile primare ale tabelelor Produs și Comanda, sunt chei externe în tabel

The screenshot shows the Oracle SQL Developer interface with the title bar "Oracle SQL Developer : Table UTILIZATOR.PRODUS_COMANDA@tutorial". The menu bar includes File, Edit, View, Navigate, Run, Team, Tools, Window, and Help. The toolbar has icons for New Connection, Open Connection, Save, Undo, Redo, and others. The connections tab shows "tutorial" is selected. The table structure for "PRODUS_COMANDA" is displayed in the main pane. The columns are:

COLUMN_NAME	DATA_TYPE	NULLABLE	DATA_DEFAULT	COLUMN_ID	COMMENTS
1 ID_PRODUS	NUMBER(38,0)	No	(null)	1	(null)
2 ID_COMANDA	NUMBER(38,0)	No	(null)	2	(null)

12. Tabela PRODUS_LISTA

- id_produs, id_lista: formează cheia primară a tabelului. Id_produs și id_comanda fac referire la cheile primare ale tabelelor Produs și Lista_Dorinte, sunt chei externe în tabel

The screenshot shows the Oracle SQL Developer interface with the title bar "Oracle SQL Developer : Table UTILIZATOR.PRODUS_LISTA@tutorial". The menu bar includes File, Edit, View, Navigate, Run, Team, Tools, Window, and Help. Below the menu is a toolbar with various icons. The main area shows a connection named "tutorial" and a table named "PRODUS_LISTA". The table structure is displayed in a grid with columns: COLUMN_NAME, DATA_TYPE, NULLABLE, DATA_DEFAULT, COLUMN_ID, and COMMENTS. Two rows are present:

COLUMN_NAME	DATA_TYPE	NULLABLE	DATA_DEFAULT	COLUMN_ID	COMMENTS
1 ID_PRODUS	NUMBER (38,0)	No	(null)	1	(null)
2 ID_LISTA	NUMBER (38,0)	No	(null)	2	(null)

13. Tabela PRODUS_COS

- id_produs, id_cos: formează cheia primară a tabelului. Id_produs și id_comanda fac referire la cheile primare ale tabelelor Produs și Cos_Cumparaturi, sunt chei externe în tabel

The screenshot shows the Oracle SQL Developer interface with the title bar "Oracle SQL Developer : Table UTILIZATOR.PRODUS_COS@tutorial". The menu bar includes File, Edit, View, Navigate, Run, Team, Tools, Window, and Help. Below the menu is a toolbar with various icons. The main area shows a connection named "tutorial" and a table named "PRODUS_COS". The table structure is displayed in a grid with columns: COLUMN_NAME, DATA_TYPE, NULLABLE, DATA_DEFAULT, COLUMN_ID, and COMMENTS. Two rows are present:

COLUMN_NAME	DATA_TYPE	NULLABLE	DATA_DEFAULT	COLUMN_ID	COMMENTS
1 ID_PRODUS	NUMBER (38,0)	No	(null)	1	(null)
2 ID_COS	NUMBER (38,0)	No	(null)	2	(null)

14. Tabela CURIER

- id_curier: cheie primară
- nume (varchar(255)): reprezintă numele de familie al curierului, are lungimea maxima de 255 de caractere

- prenume (varchar(255)): reprezintă prenumele curierului, are lungimea maxima de 255 de caractere
- salariu (number(10,2)): reprezintă salariul curierului și stochează o valoare numerică pozitiva, cu 2 zecimale

The screenshot shows the Oracle SQL Developer interface with the title bar "Oracle SQL Developer : Table UTILIZATOR.CURIER@tutorial". Below the title bar is a menu bar with File, Edit, View, Navigate, Run, Team, Tools, Window, and Help. Under the Tools menu, there is a "Connections" tab which is selected, showing "tutorial" and "CURIER". The main area displays the table structure for "CURIER". The "Columns" tab is selected, showing the following columns:

COLUMN_NAME	DATA_TYPE	NULLABLE	DATA_DEFAULT	COLUMN_ID	COMMENTS
1 ID_CURIER	NUMBER(38, 0)	No	"UTILIZATOR"."CURIER_SEQ"."NEXTVAL"	1	(null)
2 NUME	VARCHAR2(255 BYTE)	No	(null)	2	(null)
3 PRENUME	VARCHAR2(255 BYTE)	No	(null)	3	(null)
4 SALARIU	NUMBER(10, 2)	Yes	(null)	4	(null)

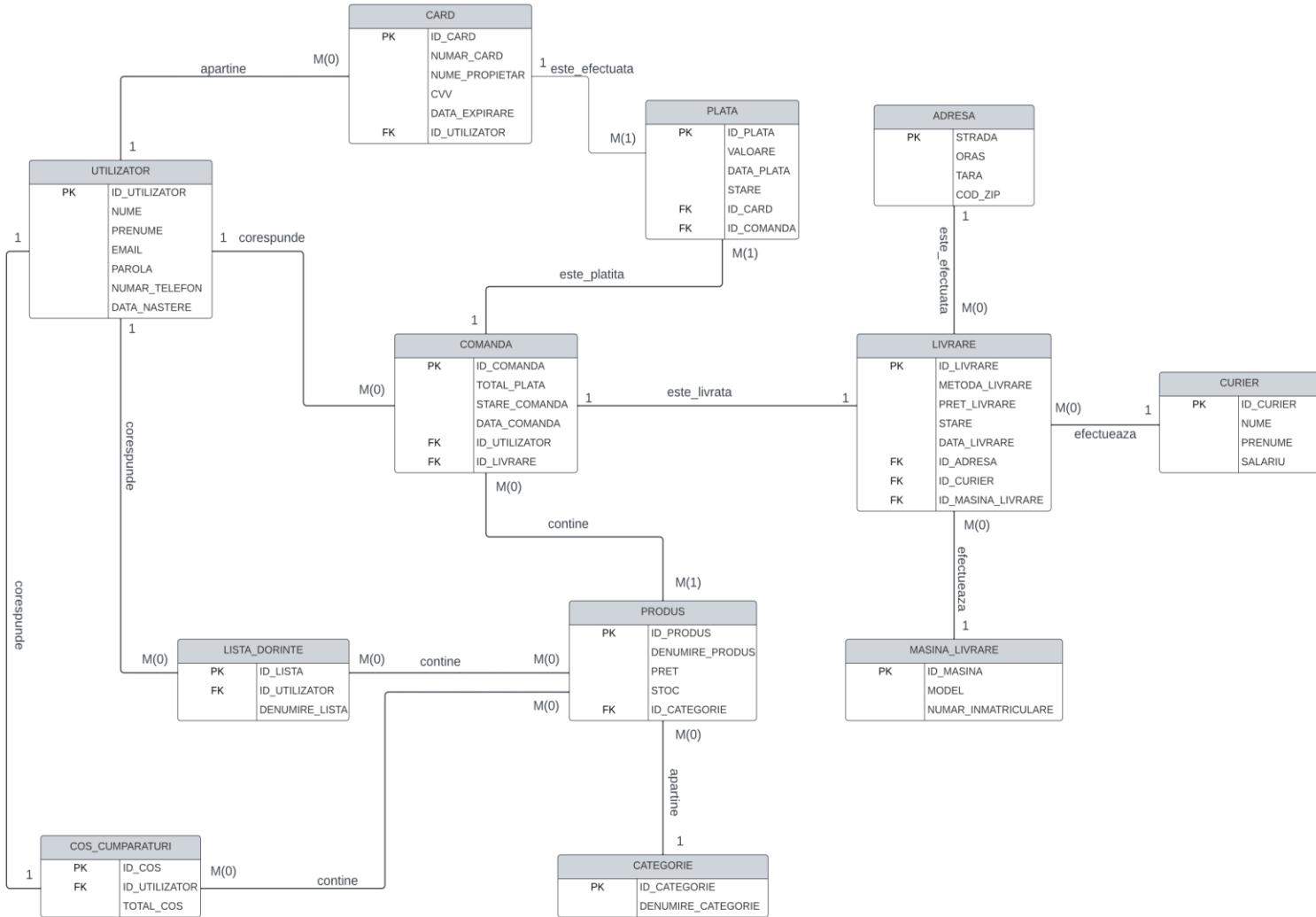
15. Tabela **MASINA_LIVRARE**

- id_masina: cheie primară
- model_masina(varchar(255)): reprezintă modelul mașinii, are lungimea maxima de 255 de caractere
- numar_inmatriculare(varchar(255)): reprezintă numărul de înmatriculare al masinii, are lungimea maxima de 255 de caractere, NOT NULL

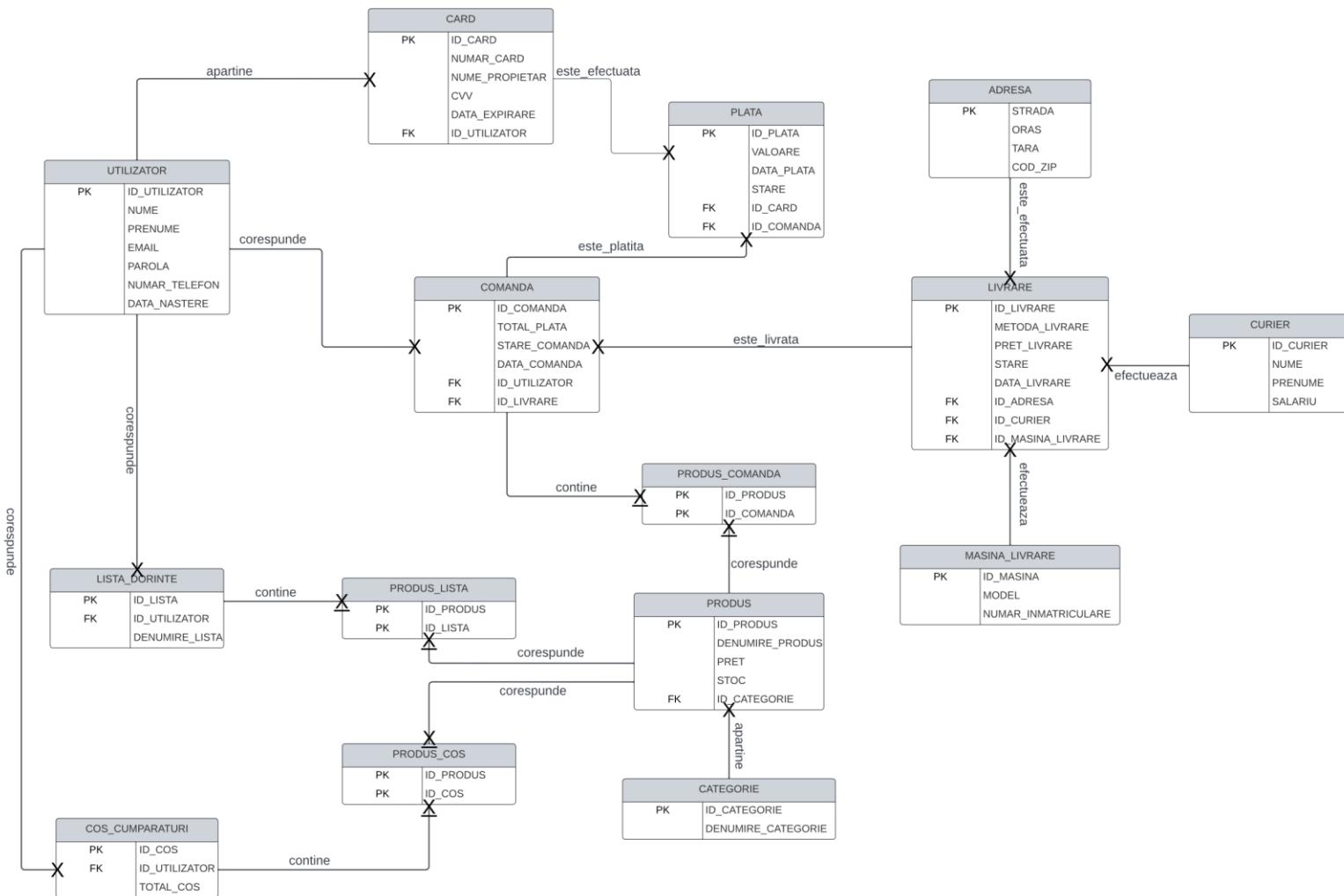
The screenshot shows the Oracle SQL Developer interface with the title bar "Oracle SQL Developer : Table UTILIZATOR.MASINA_LIVRARE@tutorial". Below the title bar is a menu bar with File, Edit, View, Navigate, Run, Team, Tools, Window, and Help. Under the Tools menu, there is a "Connections" tab which is selected, showing "tutorial" and "MASINA_LIVRARE". The main area displays the table structure for "MASINA_LIVRARE". The "Columns" tab is selected, showing the following columns:

COLUMN_NAME	DATA_TYPE	NULLABLE	DATA_DEFAULT	COLUMN_ID	COMMENTS
1 ID_MASINA	NUMBER(38, 0)	No	"UTILIZATOR"."MASINA_LIVRARE_SEQ"."NEXTVAL"	1	(null)
2 MODEL_MASINA	VARCHAR2(255 BYTE)	Yes	(null)	2	(null)
3 NUMAR_INMATRICULARE	VARCHAR2(255 BYTE)	No	(null)	3	(null)

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



7. 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.



8. Enumerarea schemelor relaționale corespunzătoare diagramei conceptuale proiectate la punctul 7.

In continuare sunt enumerate schemele relationare corespunzătoare diagramei conceptuale.

1. Tabela UTILIZATOR (id_utilizator, nume, prenume, email, parola, numar_telefon, data_nastere)
2. Tabela COMANDA (id_comanda, total_plata, stare_comanda, data_comanda, id_utilizator, id_livrare)
3. Tabela PLATA (id_plata, valoare, data_plata, stare, id_card, id_comanda)
4. Tabela CARD (id_card, numar_card, nume_proprietar, CVV, data_expirare, id_utilizator)
5. Tabela LIVRARE (id_livrare, metoda_livrare, pret_livrare, stare, data_livrare, id_adresa, id_curier, id_masina)
6. Tabela ADRESA (id_adresa, strada, oraș, tara, cod_zip)
7. Tabela LISTA_DORINTE (id_lista, id_utilizator, denumire_lista)
8. Tabela COS_CUMPARATURI (id_cos, id_utilizator, total_cos)

9. Tabela PRODUS (id_produs, denumire_produs, preț, stoc, id_categorie)
10. Tabela CATEGORIE (id_categorie, denumire_categorie)
11. Tabela PRODUS_COMANDA (id_produs, id_comanda)
12. Tabela PRODUS_LISTA (id_produs, id_lista)
13. Tabela PRODUS_COS (id_produs, id_cos)
14. Tabela CURIER(id_curier, nume, prenume, salariu)
15. Tabela MASINA_LIVRARE(id_masina, model, numar_inmatriculare)

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

Normalizari

Exemplu Non-FN1: Tabela Produs si Cos

O relație se află în FN1 dacă fiecărui atribut care o compune îi corespunde o valoare care nu se poate descompune. Un exemplu de forma NON-FN1 poate apărea în tabelul Cos, în care o comanda poate conține mai multe produse.

id_cos	id_produs
100	111/4623, 234/0987, 764/2894
101	129/2821, 729/3422
102	2894/7684, 8682/9783

Pentru a transforma tabelul în forma FN1 putem construi tabelul Produs_Cos, care determină corespondența dintre o comandă și produsele care o alcătuiesc. Cheia primară este compusă din id-ul coșului și id-ul produsului.

id_cos	id_produs
100	111/4623
100	234/0987
100	764/2894
101	129/2821
101	729/3422
102	2894/7684
102	8682/9783

Exemplu Non-FN2: Tabela Produs si Comanda

O relatie se afla in FN2 daca și numai daca se afla deja în FN1, iar fiecare atribut care nu participă la cheia primară este dependent de întreaga cheie primară.

Pentru a exemplifica acest lucru vom lucra cu tabela Produs_Comanda, care se afla in FN1. Cheia primară din aceasta tabela este compusă din id_produs și id_comanda.

In aceasta tabela, atributele data_comanda și denumire_produs nu alcătuiesc cheia primară, deci trebuie sa depindă de întreaga cheie. Acest criteriu nu este îndeplinit, deoarece se observa dependența dintre id_comanda și data_comanda, respectiv id_produs și denumire_produs.

id_comanda	data_comanda	id_produs	denumire_produs
2541	31-04-2023	111/4623	Tricou alb
7536	23-01-2024	234/0987	Rochie lungă
1006	10-12-2023	764/2894	Geanta de umăr
2680	03-10-2022	129/2821	Pantaloni

Pentru a realiza trecerea in FN2, atributul data_comanda trebuie sa se afle doar în tabela Comanda, iar denumire_produs doar în tabela Produs, astfel eliminam aceste dependente parțiale.

id_comanda	id_produs
2541	111/4623
7536	234/0987
1006	764/2894
2680	129/2821

Exemplu Non-FN3: Tabela Produs si Categorie

O relatie se afla în forma normală FN3 daca se află în forma FN2 și fiecărui atribut care nu este cheie primară, depinde de cheie, de întreaga cheie și numai de cheie. Putem exemplifica acest lucru prin intermediul tabelei Produs.

Tabela conținea inițial atributele id_produs, denumire_produs, preț, stoc, id_categorie, denumire_categorie.

Atributele denumire_produs, preț și stoc depind direct de cheia primară, dar denumire_categorie depinde de alt atribut al tabelei, id_categorie.

id_produs	id_produs	id_produs	id_produs	id_categorie	denumire_categorie
234/0987	Tricou albastru	55.50	10	C123	Tricou
111/4623	Pantaloni formali	120.00	6	C809	Pantaloni
8903/8743	Camasa de in	110.70	10	C1178	Camasa
892/1893	Geaca de iarna	199.99	35	G8923	Geaca

Pentru a aduce relația în FN3 vom crea o nouă tabelă, Categorie, unde vom introduce cele 2 atribute. Vom înlocui cele 2 atribute din tabela Produse cu o cheie strânsă, id_categorie, creând o relație de tip one-to-many, deoarece o categorie poate conține mai multe produse, dar un produs poate face parte dintr-o singură categorie

id_produs	id_produs	id_produs	id_produs
234/0987	Tricou albastru	55.50	10
111/4623	Pantaloni formali	120.00	6
8903/8743	Camasa de in	110.70	10
892/1893	Geaca de iarna	199.99	35

id_categorie	denumire_categorie
C123	Tricou
C809	Pantaloni
C1178	Camasa
G8923	Geaca

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

Cod sursa

```
CREATE SEQUENCE UTILIZATOR_SEQ START WITH 1;  
CREATE SEQUENCE PRODUS_SEQ START WITH 1;  
CREATE SEQUENCE CATEGORIE_SEQ START WITH 1;  
CREATE SEQUENCE COMANDA_SEQ START WITH 1;  
CREATE SEQUENCE LIVRARE_SEQ START WITH 1;  
CREATE SEQUENCE ADRESA_SEQ START WITH 1;  
CREATE SEQUENCE PLATA_SEQ START WITH 1;  
CREATE SEQUENCE CARD_SEQ START WITH 1;  
CREATE SEQUENCE COS_CUMPARATURI_SEQ START WITH 1;  
CREATE SEQUENCE LISTA_DORINTE_SEQ START WITH 1;  
CREATE SEQUENCE CURIER_SEQ START WITH 1;  
CREATE SEQUENCE MASINA_LIVRARE_SEQ START WITH 1;
```

PrintScreen

Oracle SQL Developer : tutorial

File Edit View Navigate Run Source Team Tools Window Help

Connections Welcome Page tutorial

Worksheet Query Builder

```
CREATE SEQUENCE UTILIZATOR_SEQ START WITH 1;
CREATE SEQUENCE PRODUS_SEQ START WITH 1;
CREATE SEQUENCE CATEGORIE_SEQ START WITH 1;
CREATE SEQUENCE COMANDA_SEQ START WITH 1;
CREATE SEQUENCE LIVRARE_SEQ START WITH 1;
CREATE SEQUENCE ADRESA_SEQ START WITH 1;
CREATE SEQUENCE PLATA_SEQ START WITH 1;
CREATE SEQUENCE CARD_SEQ START WITH 1;
CREATE SEQUENCE COS_CUMPARATURI_SEQ START WITH 1;
CREATE SEQUENCE LISTA_DORINTE_SEQ START WITH 1;
CREATE SEQUENCE CURIER_SEQ START WITH 1;
CREATE SEQUENCE MASINA_LIVRARE_SEQ START WITH 1;
```

Query Result Script Output

Sequence UTILIZATOR_SEQ created.

Sequence PRODUS_SEQ created.

Sequence CATEGORIE_SEQ created.

Sequence COMANDA_SEQ created.

Sequence LIVRARE_SEQ created.

Sequence ADRESA_SEQ created.

Sequence PLATA_SEQ created.

```
Sequence CARD_SEQ created.  
  
Sequence COS_CUMPARATURI_SEQ created.  
  
Sequence LISTA_DORINTE_SEQ created.  
  
Sequence CURIER_SEQ created.  
  
Sequence MASINA_LIVRARE_SEQ created.
```

11. 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).

Utilizator

```
CREATE TABLE UTILIZATOR (  
    ID_UTILIZATOR INT DEFAULT UTILIZATOR_SEQ.NEXTVAL,  
    NUME VARCHAR(255),  
    PRENUME VARCHAR(255),  
    EMAIL VARCHAR(255) NOT NULL,  
    PAROLA VARCHAR(255) NOT NULL,  
    NUMAR_TELEFON VARCHAR(10) CONSTRAINT VERIFICARE_TELEFON CHECK  
        ( LENGTH(NUMAR_TELEFON) = 10 ),  
    DATA_NASTERE DATE,  
    CONSTRAINT CHEIE_PRIMARA_UTILIZATOR PRIMARY KEY (ID_UTILIZATOR),  
    CONSTRAINT EMAIL_UNIC UNIQUE (EMAIL)  
);
```

```
INSERT INTO UTILIZATOR (NUME, PRENUME, EMAIL, PAROLA,  
    NUMAR_TELEFON, DATA_NASTERE) VALUES  
    ('Maciuca', 'Emma', 'emmamaciuca@gmail.com', 'parola2024', '0712345678', '12-FEB-2004');
```

```
INSERT INTO UTILIZATOR (NUME, PRENUME, EMAIL, PAROLA,  
NUMAR_TELEFON, DATA_NASTERE) VALUES
```

```
('Popescu', 'Andreea-Ioana', 'popescuandreea@gmail.com', 'abcpasola', '0757828352' , '14-  
APR-2000');
```

```
INSERT INTO UTILIZATOR (NUME, PRENUME, EMAIL, PAROLA,  
NUMAR_TELEFON, DATA_NASTERE) VALUES
```

```
('Ion', 'Andrei', 'ionandrei@gmail.com', 'contmagazin', '0717025379' , '20-MAR-1990');
```

```
INSERT INTO UTILIZATOR (NUME, PRENUME, EMAIL, PAROLA,  
NUMAR_TELEFON, DATA_NASTERE) VALUES
```

```
('Andreescu', 'Maria', 'mariaandr@gmail.com', '123456789', '0720953487' , '09-MAY-2001');
```

```
INSERT INTO UTILIZATOR (NUME, PRENUME, EMAIL, PAROLA,  
NUMAR_TELEFON, DATA_NASTERE) VALUES
```

```
('Maciuca', 'Sara', 'maciucasara@gmail.com', 'sara1998', '0709348726' , '26-MAY-1998');
```

```
INSERT INTO UTILIZATOR (NUME, PRENUME, EMAIL, PAROLA,  
NUMAR_TELEFON, DATA_NASTERE) VALUES
```

```
('Marin', 'Iuliana', 'mariniuliana@gmail.com', 'iulianamarin12', '0746892537' , '12-JUL-1999');
```

```
INSERT INTO UTILIZATOR (NUME, PRENUME, EMAIL, PAROLA,  
NUMAR_TELEFON, DATA_NASTERE) VALUES
```

```
('Pop', 'Daria', 'dariap56@gmail.com', 'fructabcd', '0756293641' , '05-JUN-2002');
```

```
INSERT INTO UTILIZATOR (NUME, PRENUME, EMAIL, PAROLA,  
NUMAR_TELEFON, DATA_NASTERE) VALUES
```

```
('Popa', 'Mara', 'marapopa@gmail.com', 'maraa2103', '0747935728' , '21-MAR-2000');
```

```
INSERT INTO UTILIZATOR (NUME, PRENUME, EMAIL, PAROLA,  
NUMAR_TELEFON, DATA_NASTERE) VALUES
```

```
('Mares', 'Darius', 'maresdarius@gmail.com', 'guh4ngj5nkf!', '0767937563' , '08-JUN-2003');
```

```
INSERT INTO UTILIZATOR (NUME, PRENUME, EMAIL, PAROLA,  
NUMAR_TELEFON, DATA_NASTERE) VALUES
```

```
('Balan', 'Ioana', 'balan_ioana@gmail.com', 'ppp273ffs', '0763927455', '15-APR-1989');
```

```
INSERT INTO UTILIZATOR (NUME, PRENUME, EMAIL, PAROLA,  
NUMAR_TELEFON, DATA_NASTERE) VALUES
```

```
('Popescu', 'Marian', 'mariannpopescu@gmail.com', 'parola_magazin', '0765223465', '11-JUN-  
1980');
```

```
SELECT * FROM UTILIZATOR;
```

The screenshot shows the Oracle SQL Developer interface. The top menu bar includes File, Edit, View, Navigate, Run, Source, Team, Tools, Window, Help. The toolbar has icons for New Connection, Home, Welcome Page, and tutorial. The Connections panel shows a connection named 'tutorial'. The main area is a Worksheet titled 'Query Builder' containing the following SQL code:

```
CREATE TABLE UTILIZATOR (
    ID_UTILIZATOR INT DEFAULT UTILIZATOR_SEQ.NEXTVAL,
    NUME VARCHAR(15),
    PRENUME VARCHAR(15),
    EMAIL VARCHAR(255) NOT NULL,
    PAROLA VARCHAR(255) NOT NULL,
    NUMAR_TELEFON CHAR(10) CONSTRAINT VERIFICARE_TELEFON CHECK ( LENGTH(NUMAR_TELEFON) = 10 ),
    DATA_NASTERE DATE,
    CONSTRAINT C1_ID_UTILIZATOR PRIMARY KEY (ID_UTILIZATOR),
    CONSTRAINT EMAIL_UNIQUE UNIQUE (EMAIL)
);

INSERT INTO UTILIZATOR (NUME, PRENUME, EMAIL, PAROLA, NUMAR_TELEFON, DATA_NASTERE) VALUES
('Marian', 'Ioana', 'balan_ioana@gmail.com', 'ppp273ffs', '0763927455', '15-APR-1989');

INSERT INTO UTILIZATOR (NUME, PRENUME, EMAIL, PAROLA, NUMAR_TELEFON, DATA_NASTERE) VALUES
('Popescu', 'Andreea-Ioana', 'popescuandreea@gmail.com', 'amparola', '9765223465', '11-JUN-1980');

INSERT INTO UTILIZATOR (NUME, PRENUME, EMAIL, PAROLA, NUMAR_TELEFON, DATA_NASTERE) VALUES
('Popescu', 'Maria', 'mariannpopescu@gmail.com', 'parola_magazin', '0765223465', '11-JUN-1980');

Sequence UTILIZATOR_SEQ created.
```

The bottom status bar shows the number of rows inserted (1 row inserted twice), the current time (9:57 AM), and the date (6/7/2024). The system tray indicates it's 73°F and sunny.

```

-- Oracle SQL Developer / tutorial
File Edit View Navigate Run Source Team Tools Window Help
Connections Welcome Page tutorial
Worksheet Query Builder
-- Insertion of 11 rows into UTILIZATOR table
INSERT INTO UTILIZATOR (NUME, PRENUME, EMAIL, PAROLA, NUMAR_TELEFON, DATA_NASTERE) VALUES
('Andreeescu', 'Maria', 'mariaandr@gmail.com', '123456789', '0720953487', '09-MAY-2001');
INSERT INTO UTILIZATOR (NUME, PRENUME, EMAIL, PAROLA, NUMAR_TELEFON, DATA_NASTERE) VALUES
('Maciuca', 'Sara', 'maciucasara@gmail.com', 'sara1998', '0709348726', '26-MAY-1998');
INSERT INTO UTILIZATOR (NUME, PRENUME, EMAIL, PAROLA, NUMAR_TELEFON, DATA_NASTERE) VALUES
('Maria', 'Iuliana', 'mariniuliana@gmail.com', 'iulianamarin12', '0746892537', '12-JUL-1999');
INSERT INTO UTILIZATOR (NUME, PRENUME, EMAIL, PAROLA, NUMAR_TELEFON, DATA_NASTERE) VALUES
('Pop', 'Daria', 'dariaPop56@gmail.com', 'fructabcd', '0756293641', '05-JUN-2002');
INSERT INTO UTILIZATOR (NUME, PRENUME, EMAIL, PAROLA, NUMAR_TELEFON, DATA_NASTERE) VALUES
('Pop', 'Mara', 'marapop@gmail.com', 'maraaa2103', '0747935728', '21-MAR-00');
INSERT INTO UTILIZATOR (NUME, PRENUME, EMAIL, PAROLA, NUMAR_TELEFON, DATA_NASTERE) VALUES
('Mares', 'Darius', 'maresdarius@gmail.com', 'guh4ngj5nkf!', '0767937563', '08-JUN-03');
INSERT INTO UTILIZATOR (NUME, PRENUME, EMAIL, PAROLA, NUMAR_TELEFON, DATA_NASTERE) VALUES
('Balan', 'Ioana', 'balan_ioana@gmail.com', 'ppp273ffs', '0763927455', '15-APR-1969');
INSERT INTO UTILIZATOR (NUME, PRENUME, EMAIL, PAROLA, NUMAR_TELEFON, DATA_NASTERE) VALUES
('Popescu', 'Marian', 'mariannopescu@gmail.com', 'parola_magazin', '0765223465', '11-JUN-1980');

SELECT * FROM UTILIZATOR;

```

Script Output | Query Result | Task completed in 0.568 seconds

1 row inserted.
1 row inserted.

Script Output | Query Result | All Rows Fetched: 11 in 0.086 seconds

ID_UTILIZATOR	NUME	PRENUME	EMAIL	PAROLA	NUMAR_TELEFON	DATA_NASTERE
1	Maciuca	Emma	emmamaciuca@gmail.com	parola2024	0712345678	12-FEB-04
2	Popescu	Andreea-Ioana	popescuandreea@gmail.com	abcparola	0757828352	14-APR-00
3	Ion	Andrei	ionandrei@gmail.com	contmagazin	0717025379	20-MAR-90
4	Andreeescu	Maria	mariaandr@gmail.com	123456789	0720953487	09-MAY-01
5	Maciuca	Sara	maciucasara@gmail.com	sara1998	0709348726	26-MAY-98
6	Marin	Iuliana	mariniuliana@gmail.com	iulianamarin12	0746892537	12-JUL-99
7	Pop	Daria	dariaPop56@gmail.com	fructabcd	0756293641	05-JUN-02
8	Popa	Mara	marapop@gmail.com	maraaa2103	0747935728	21-MAR-00
9	Mares	Darius	maresdarius@gmail.com	guh4ngj5nkf!	0767937563	08-JUN-03
10	Balan	Ioana	balan_ioana@gmail.com	ppp273ffs	0763927455	15-APR-89
11	Popescu	Marian	mariannopescu@gmail.com	parola_magazin	0765223465	11-JUN-80

Adresa

CREATE TABLE ADRESA (

```

ID_ADRESA INT DEFAULT ADRESA_SEQ.NEXTVAL,
STRADA VARCHAR2(255) NOT NULL,
ORAS VARCHAR2(255) NOT NULL,
TARA VARCHAR2(255) NOT NULL,
COD_ZIP VARCHAR2(10) NOT NULL,

```

CONSTRAINT CHEIE_PRIMARA_ADRESA PRIMARY KEY (ID_ADRESA)
);

INSERT INTO ADRESA (STRADA, ORAS, TARA, COD_ZIP) VALUES ('BULEVARDUL TOMIS 287', 'CONSTANTA', 'ROMANIA', '900407');

INSERT INTO ADRESA (STRADA, ORAS, TARA, COD_ZIP) VALUES ('STRADA VASILE PARVAN 2', 'BUCURESTI', 'ROMANIA', '901294');

INSERT INTO ADRESA (STRADA, ORAS, TARA, COD_ZIP) VALUES ('BULEVARDUL ALEXANDRU LAPUSNEANU 13', 'CONSTANTA', 'ROMANIA', '900352');

INSERT INTO ADRESA (STRADA, ORAS, TARA, COD_ZIP) VALUES ('STRADA DEZROBIRII 100', 'CONSTANTA', 'ROMANIA', '900294');

INSERT INTO ADRESA (STRADA, ORAS, TARA, COD_ZIP) VALUES ('BULEVARDUL CAROL I NR.12', 'BUCURESTI', 'ROMANIA', '010292');

INSERT INTO ADRESA (STRADA, ORAS, TARA, COD_ZIP) VALUES ('STRADA MATEI BASARAB NR.67A', 'GALATI', 'ROMANIA', '800921');

INSERT INTO ADRESA (STRADA, ORAS, TARA, COD_ZIP) VALUES ('SPLAIUL UNIRII NR.312 BLOC T12 SC.C', 'BUCURESTI', 'ROMANIA', '010734');

INSERT INTO ADRESA (STRADA, ORAS, TARA, COD_ZIP) VALUES ('STRADA CASTANIILOR NR.1 BLOC A2 SC.B AP.6', 'BRASOV', 'ROMANIA', '500752');

INSERT INTO ADRESA (STRADA, ORAS, TARA, COD_ZIP) VALUES ('STRADA FAGETULUI NR.9', 'BUCURESTI', 'ROMANIA', '010184');

INSERT INTO ADRESA (STRADA, ORAS, TARA, COD_ZIP) VALUES ('STRADA MARULUI NR.92', 'BUCURESTI', 'ROMANIA', '010793');

The screenshot shows the Oracle SQL Developer interface. The top menu bar includes File, Edit, View, Navigate, Run, Source, Team, Tools, Window, Help. Below the menu is a toolbar with icons for New Connection, Welcome Page, and Tutorial. The main workspace is titled 'Worksheet' and contains a query builder with the following SQL script:

```
INSERT INTO ADRESA (STRADA, GRAD, TARA, COD_ZIP) VALUES ('STRADA VASILE PAROH 2', 'BOUCURESTI', 'ROMANIA', '901284');

INSERT INTO ADRESA (STRADA, GRAD, TARA, COD_ZIP) VALUES ('BULEVARDUL ALEXANDRU LAPOINEANU 13', 'CONSTANTIA', 'ROMANIA', '900352');

INSERT INTO ADRESA (STRADA, GRAD, TARA, COD_ZIP) VALUES ('STRADA DEGRONIRII 100', 'CONSTANTIA', 'ROMANIA', '900294');

INSERT INTO ADRESA (STRADA, GRAD, TARA, COD_ZIP) VALUES ('BULEVARDUL CAROL I NR.12', 'BOUCURESTI', 'ROMANIA', '010282');

INSERT INTO ADRESA (STRADA, GRAD, TARA, COD_ZIP) VALUES ('STRADA MATEI BASARAB NR.67A', 'GLATI', 'ROMANIA', '900921');

INSERT INTO ADRESA (STRADA, GRAD, TARA, COD_ZIP) VALUES ('SPALEUL UNIRII NR.312 BLOC T12 SC.C', 'BOUCURESTI', 'ROMANIA', '010734');

INSERT INTO ADRESA (STRADA, GRAD, TARA, COD_ZIP) VALUES ('STRADA CANTEMIR NR.1 BLOC A2 SC.B AP.6', 'BRASOV', 'ROMANIA', '500732');

INSERT INTO ADRESA (STRADA, GRAD, TARA, COD_ZIP) VALUES ('STRADA FAGETUTII NR.9', 'BOUCURESTI', 'ROMANIA', '010184');

INSERT INTO ADRESA (STRADA, GRAD, TARA, COD_ZIP) VALUES ('STRADA MARULUII NR.93', 'BOUCURESTI', 'ROMANIA', '010793');

...;
```

The 'Script Output' tab shows the results of the execution:

```
1 row inserted.

1 row inserted.
```

The status bar at the bottom right indicates 'Line 136 Column 1 | Insert | Modified | Windows'. The system tray shows the date and time as '6/7/2024 10:04 AM'. A weather icon in the bottom left corner shows '73°F Sunny'.

The screenshot shows the 'Query Result' tab in Oracle SQL Developer with the following data:

ID_ADRESA	STRADA	ORAS	TARA	COD_ZIP
1	1 BULEVARDUL TOMIS 287	CONSTANTA	ROMANIA	900407
2	2 STRADA VASILE PARVAN 2	BUCURESTI	ROMANIA	901294
3	3 BULEVARDUL ALEXANDRU LAPUSNEANU 13	CONSTANTA	ROMANIA	900352
4	4 STRADA DEZROBIRII 100	CONSTANTA	ROMANIA	900294
5	5 BULEVARDUL CAROL I NR.12	BUCURESTI	ROMANIA	010292
6	6 STRADA MATEI BASARAB NR.67A	GALATI	ROMANIA	800921
7	7 SPLAIUL UNIRII NR.312 BLOC T12 SC.C	BUCURESTI	ROMANIA	010734
8	8 STRADA CASTANILOR NR.1 BLOC A2 SC.B AP.6	BRASOV	ROMANIA	500752
9	9 STRADA FAGETULUI NR.9	BUCURESTI	ROMANIA	010184
10	10 STRADA MARULUI NR.92	BUCURESTI	ROMANIA	010793

Curier

```
CREATE TABLE CURIER(
    ID_CURIER INT DEFAULT CURIER_SEQ.NEXTVAL,
    NUME VARCHAR2(255) NOT NULL,
    PRENUME VARCHAR2(255)NOT NULL,
    SALARIU NUMBER(10,2),
    CONSTRAINT PK_CURIER PRIMARY KEY (ID_CURIER)
);
```

```
INSERT INTO CURIER (NUME, PRENUME, SALARIU) VALUES ('ION', 'ALEXANDRU', 2800);
```

```
INSERT INTO CURIER (NUME, PRENUME, SALARIU) VALUES ('POPESCU', 'MARIA', 3200);
```

```
INSERT INTO CURIER (NUME, PRENUME, SALARIU) VALUES ('IONESCU', 'GEORGE', 2900);
```

```
INSERT INTO CURIER (NUME, PRENUME, SALARIU) VALUES ('STANESCU', 'ANA', 3100);
```

```
INSERT INTO CURIER (NUME, PRENUME, SALARIU) VALUES ('DOBRESCU', 'MIHAI', 2700);
```

```
INSERT INTO CURIER (NUME, PRENUME, SALARIU) VALUES ('MARINESCU',  
'ANDREI', 3300);
```

```
INSERT INTO CURIER (NUME, PRENUME, SALARIU) VALUES ('ALEXANDRU',  
'RADU', 3400);
```

```
INSERT INTO CURIER (NUME, PRENUME, SALARIU) VALUES ('STEFANESCU',  
'CRISTINA', 2800);
```

```
INSERT INTO CURIER (NUME, PRENUME, SALARIU) VALUES ('IONESCU', 'ADINA',  
3500);
```

```
INSERT INTO CURIER (NUME, PRENUME, SALARIU) VALUES ('RADU', 'FLORIN',  
3500);
```

```
INSERT INTO CURIER (NUME, PRENUME, SALARIU) VALUES ('STEFANESCU',  
'VLAD', 3600);
```

```
INSERT INTO CURIER (NUME, PRENUME, SALARIU) VALUES ('CONSTANTINESCU', 'MARIUS', 3700);
```

```
select * from curier;
```

The screenshot shows the Oracle SQL Developer interface. The top menu bar includes File, Edit, View, Navigate, Run, Source, Team, Tools, Window, and Help. The Connections tab shows a single connection named "tutorial". The main workspace has two tabs: "Worksheet" and "Query Builder". The "Worksheet" tab contains the following SQL code:

```
CREATE TABLE CURIER  
(ID_CURIER INT GENERATED BY DEFAULT CURIER_SEQ.NEXTVAL,  
NUME NVARCHAR2(55) NOT NULL,  
PRENUME VARCHAR2(55) NOT NULL,  
SALARIU NUMBER(10,2)  
CONSTRAINT PK_CURIER PRIMARY KEY (ID_CURIER)  
);  
  
INSERT INTO CURIER (NUME, PRENUME, SALARIU) VALUES ('ION', 'ALEXANDRU', 2800);  
INSERT INTO CURIER (NUME, PRENUME, SALARIU) VALUES ('POPOESCU', 'MARIUS', 3200);  
INSERT INTO CURIER (NUME, PRENUME, SALARIU) VALUES ('IONESCU', 'ADINA', 3500);  
INSERT INTO CURIER (NUME, PRENUME, SALARIU) VALUES ('STEFANESCU', 'CRISTINA', 2800);  
INSERT INTO CURIER (NUME, PRENUME, SALARIU) VALUES ('RADU', 'FLORIN', 3500);  
INSERT INTO CURIER (NUME, PRENUME, SALARIU) VALUES ('DOROBESCU', 'MIRAL', 2700);  
INSERT INTO CURIER (NUME, PRENUME, SALARIU) VALUES ('MARINESCU', 'ANDREI', 3300);  
INSERT INTO CURIER (NUME, PRENUME, SALARIU) VALUES ('VLAD', 'RADU', 3400);  
INSERT INTO CURIER (NUME, PRENUME, SALARIU) VALUES ('STEFANESCU', 'RADU', 3400);  
INSERT INTO CURIER (NUME, PRENUME, SALARIU) VALUES ('IONESCU', 'CRISTINA', 2800);  
INSERT INTO CURIER (NUME, PRENUME, SALARIU) VALUES ('RADU', 'FLORIN', 3500);  
INSERT INTO CURIER (NUME, PRENUME, SALARIU) VALUES ('STEFANESCU', 'VLAD', 3600);  
INSERT INTO CURIER (NUME, PRENUME, SALARIU) VALUES ('CONSTANTINESCU', 'MARIUS', 3700);  
select * from curier;
```

The "Script Output" tab shows the results of the table creation and the execution of the INSERT ALL statement. It displays the message "Table CURIER created." followed by seven "1 row inserted." messages, one for each inserted row. The "Task completed in 0.41 seconds" message is also present.

ID_CURIER	NUME	PRENUME	SALARIU
1	1 ION	ALEXANDRU	2800
2	2 POPESCU	MARIA	3200
3	3 IONESCU	GEORGE	2900
4	4 STANESCU	ANA	3100
5	5 DOBRESCU	MIHAI	2700
6	6 MARINESCU	ANDREI	3300
7	7 ALEXANDRU	RADU	3400
8	8 STEFANESCU	CRISTINA	2800
9	9 IONESCU	ADINA	3500
10	10 RADU	FLORIN	3500
11	11 STEFANESCU	VLAD	3600
12	12 CONSTANTINESCU	MARIUS	3700

Livrare

```

CREATE TABLE MASINA_LIVRARE(
    ID_MASINA INT DEFAULT MASINA_LIVRARE_SEQ.NEXTVAL,
    MODEL_MASINA VARCHAR2(255),
    NUMAR_INMATRICULARE VARCHAR2(255)NOT NULL,
    CONSTRAINT PK_MASINA PRIMARY KEY (ID_MASINA)
);

INSERT INTO MASINA_LIVRARE (MODEL_MASINA, NUMAR_INMATRICULARE)
VALUES ('Mercedes-Benz Sprinter', 'B-123-ABC');

INSERT INTO MASINA_LIVRARE (MODEL_MASINA, NUMAR_INMATRICULARE)
VALUES ('Ford Transit', 'BV-456-DEF');

INSERT INTO MASINA_LIVRARE (MODEL_MASINA, NUMAR_INMATRICULARE)
VALUES ('Fiat Ducato', 'B-789-GHI');

INSERT INTO MASINA_LIVRARE (MODEL_MASINA, NUMAR_INMATRICULARE)
VALUES ('Volkswagen Crafter', 'CT-101-JKL');

INSERT INTO MASINA_LIVRARE (MODEL_MASINA, NUMAR_INMATRICULARE)
VALUES ('Renault Master', 'B-202-MNO');

```

```
INSERT INTO MASINA_LIVRARE (MODEL_MASINA, NUMAR_INMATRICULARE)
VALUES ('Peugeot Boxer', 'GL-303-PQR');
```

```
INSERT INTO MASINA_LIVRARE (MODEL_MASINA, NUMAR_INMATRICULARE)
VALUES ('Iveco Daily', 'B-404-STU');
```

```
INSERT INTO MASINA_LIVRARE (MODEL_MASINA, NUMAR_INMATRICULARE)
VALUES ('Citroën Jumper', 'CT-505-VWX');
```

```
INSERT INTO MASINA_LIVRARE (MODEL_MASINA, NUMAR_INMATRICULARE)
VALUES ('Opel Movano', 'B-606-YZA');
```

```
INSERT INTO MASINA_LIVRARE (MODEL_MASINA, NUMAR_INMATRICULARE)
VALUES ('Nissan NV400', 'B-707-BCD');
```

```
INSERT INTO MASINA_LIVRARE (MODEL_MASINA, NUMAR_INMATRICULARE)
VALUES ('Hyundai H350', 'BV-808-EFG');
```

```
INSERT INTO MASINA_LIVRARE (MODEL_MASINA, NUMAR_INMATRICULARE)
VALUES ('MAN TGE', 'CT-909-HIJ');
```

```
select * from masina_livrare;
```

The screenshot shows the Oracle SQL Developer interface. The top menu bar includes File, Edit, View, Navigate, Run, Source, Team, Tools, Window, and Help. The Connections tab shows a connection named 'tutorial'. The Worksheet tab contains the following SQL code:

```
--Tabelul MASINA_LIVRARE
CREATE TABLE MASINA_LIVRARE(
    ID_MASINA INT DEFAULT MASINA_LIVRARE_SEQ.NEXTVAL,
    MODEL_MASINA VARCHAR(255),
    NUMAR_INMATRICULARE VARCHAR(255) NOT NULL,
    CONSTRAINT PK_MASINA PRIMARY KEY (ID_MASINA)
);

INSERT INTO MASINA_LIVRARE (MODEL_MASINA, NUMAR_INMATRICULARE) VALUES ('Mercedes-Benz Sprinter', 'B-123-ABC');
INSERT INTO MASINA_LIVRARE (MODEL_MASINA, NUMAR_INMATRICULARE) VALUES ('Ford Transit', 'BV-456-DEF');
INSERT INTO MASINA_LIVRARE (MODEL_MASINA, NUMAR_INMATRICULARE) VALUES ('Fiat Ducato', 'B-789-GHI');
INSERT INTO MASINA_LIVRARE (MODEL_MASINA, NUMAR_INMATRICULARE) VALUES ('Volkswagen Crafter', 'CT-101-JKL');
INSERT INTO MASINA_LIVRARE (MODEL_MASINA, NUMAR_INMATRICULARE) VALUES ('Renault Master', 'BV-234-MNO');
INSERT INTO MASINA_LIVRARE (MODEL_MASINA, NUMAR_INMATRICULARE) VALUES ('Peugeot Boxer', 'GL-303-PQR');
INSERT INTO MASINA_LIVRARE (MODEL_MASINA, NUMAR_INMATRICULARE) VALUES ('Iveco Daily', 'B-404-STU');
INSERT INTO MASINA_LIVRARE (MODEL_MASINA, NUMAR_INMATRICULARE) VALUES ('Citroën Jumper', 'CT-505-VWX');
INSERT INTO MASINA_LIVRARE (MODEL_MASINA, NUMAR_INMATRICULARE) VALUES ('Opel Movano', 'B-606-YZA');
INSERT INTO MASINA_LIVRARE (MODEL_MASINA, NUMAR_INMATRICULARE) VALUES ('Nissan NV400', 'B-707-BCD');
INSERT INTO MASINA_LIVRARE (MODEL_MASINA, NUMAR_INMATRICULARE) VALUES ('Hyundai H350', 'BV-808-EFG');
INSERT INTO MASINA_LIVRARE (MODEL_MASINA, NUMAR_INMATRICULARE) VALUES ('MAN TGE', 'CT-909-HIJ');
```

The Script Output window shows the message "Table MASINA_LIVRARE created." and "1 row inserted." repeated nine times, corresponding to the nine rows inserted in the table. The Query Result window is empty. The status bar at the bottom right shows "10:10 AM 6/7/2024".

The screenshot shows a database query results window in Oracle SQL Developer. The window title is 'Query Result'. It displays 12 rows of data from a table named 'MASINA'. The columns are labeled 'ID_MASINA', 'MODEL_MASINA', and 'NUMAR_INMATRICULARE'. The data includes various car models and their license plate numbers.

ID_MASINA	MODEL_MASINA	NUMAR_INMATRICULARE
1	1 Mercedes-Benz Sprinter	B-123-ABC
2	2 Ford Transit	BV-456-DEF
3	3 Fiat Ducato	B-789-GHI
4	4 Volkswagen Crafter	CT-101-JKL
5	5 Renault Master	B-202-MNO
6	6 Peugeot Boxer	GL-303-PQR
7	7 Iveco Daily	B-404-STU
8	8 Citroën Jumper	CT-505-VWX
9	9 Opel Movano	B-606-YZA
10	10 Nissan NV400	B-707-BCD
11	11 Hyundai H350	BV-808-EFG
12	12 MAN TGE	CT-909-HIJ

Livrare

```

CREATE TABLE LIVRARE (
    ID_LIVRARE INT DEFAULT LIVRARE_SEQ.NEXTVAL,
    METODA_LIVRARE VARCHAR2(255) NOT NULL,
    PRET_LIVRARE NUMBER(10,2),
    STARE VARCHAR2(255) NOT NULL,
    DATA_LIVRARE DATE,
    ID_ADRESA INT,
    ID_CURIER INT,
    ID_MASINA INT,
    CONSTRAINT PK_UTILIZATOR PRIMARY KEY (ID_LIVRARE),
    CONSTRAINT FK_LIVRARE_ADRESA FOREIGN KEY(ID_ADRESA) REFERENCES
    ADRESA(ID_ADRESA),
    CONSTRAINT FK_LIVRARE_CURIER FOREIGN KEY(ID_CURIER) REFERENCES
    CURIER(ID_CURIER),
    CONSTRAINT FK_LIVRARE_MASINA FOREIGN KEY(ID_MASINA) REFERENCES
    MASINA_LIVRARE(ID_MASINA),
    CONSTRAINT STARE_CHECK CHECK (STARE IN ('IN CURS DE PROCESARE',
    'EFECTUATA', 'IN CURS DE LIVRARE'))
);

```

```
CONSTRAINT PRET_LIVRARE_CHECK CHECK (PRET_LIVRARE >= 0)
);
```

```
INSERT INTO LIVRARE (metoda_livrare, pret_livrare, stare, data_livrare, id_adresa, id_curier, id_masina) VALUES ('CURIER', 12.90 , 'EFECTUATA', '12-MAY-2024', 1,2,7);
```

```
INSERT INTO LIVRARE (metoda_livrare, pret_livrare, stare, data_livrare, id_adresa, id_curier, id_masina) VALUES ('EASYBOX',6.90 , 'EFECTUATA', '15-MAY-2024', 4,3,10);
```

```
INSERT INTO LIVRARE (metoda_livrare, pret_livrare, stare, data_livrare, id_adresa, id_curier, id_masina) VALUES ('EASYBOX',6.90 , 'EFECTUATA', '15-MAY-2024', 4,3,10);
```

```
INSERT INTO LIVRARE (metoda_livrare, pret_livrare, stare, data_livrare, id_adresa, id_curier, id_masina) VALUES ('CURIER', 16.99 , 'INCURS DELIVRARE', '20-MAY-2024', 9,1,5);
```

```
INSERT INTO LIVRARE (metoda_livrare, pret_livrare, stare, data_livrare, id_adresa, id_curier, id_masina) VALUES ('FANBOX', 10 , 'EFECTUATA', '20-FEB-2024', 8,5,1);
```

```
INSERT INTO LIVRARE (metoda_livrare, pret_livrare, stare, data_livrare, id_adresa, id_curier, id_masina) VALUES ('CURIER', 16.99 , 'EFECTUATA', '02-DEC-2023', 10,7,11);
```

```
INSERT INTO LIVRARE (metoda_livrare, pret_livrare, stare, data_livrare, id_adresa, id_curier, id_masina) VALUES ('SAMEDAY',0 , 'IN CURS DE PROCESARE', '17-MAY-2024', 7,10,2);
```

```
INSERT INTO LIVRARE (metoda_livrare, pret_livrare, stare, data_livrare, id_adresa, id_curier, id_masina) VALUES ('CURIER', 19.99 , 'EFECTUATA', '16-APR-2024', 6,1,5);
```

```
INSERT INTO LIVRARE (metoda_livrare, pret_livrare, stare, data_livrare, id_adresa, id_curier, id_masina) VALUES ('CURIER', 16.99 , 'INCURS DE LIVRARE', '26-JUN-2023', 2,6,3);
```

```
INSERT INTO LIVRARE (metoda_livrare, pret_livrare, stare, data_livrare, id_adresa, id_curier, id_masina) VALUES ('EASYBOX',0 , 'IN CURS DE PROCESARE', '16-MAY-2024', 9,11,8);
```

```
INSERT INTO LIVRARE (metoda_livrare, pret_livrare, stare, data_livrare, id_adresa, id_curier, id_masina) VALUES ('CURIER', 14.99, 'EFECTUATA', '30-MAR-2024', 5, 4, 12);
```

```
INSERT INTO LIVRARE (metoda_livrare, pret_livrare, stare, data_livrare, id_adresa, id_curier, id_masina) VALUES ('FANBOX',10.50 , 'EFECTUATA', '22-MAR-2024', 3, 6, 4);
```

```
INSERT INTO LIVRARE (metoda_livrare, pret_livrare, stare, data_livrare, id_adresa, id_curier, id_masina) VALUES ('CURIER', 19.99, 'INCURS DE LIVRARE', '01-JUN-2024', 2, 12, 9);
```

```
SELECT * FROM LIVRARE;
```


The screenshot shows a database query results window in Oracle SQL Developer. The title bar says "Script Output" and "Query Result". Below the title bar, there are icons for refresh, print, and exit, followed by the text "SQL | All Rows Fetched: 13 in 0.116 seconds". The main area displays a table with 13 rows of data. The columns are labeled: ID_LIVRARE, METODA_LIVRARE, PRET_LIVRARE, STARE, DATA_LIVRARE, ID_ADRESA, ID_CURIER, and ID_MASINA. The data includes various delivery methods like CURIER, EASYBOX, and FANBOX, with prices ranging from 6.9 to 19.99 and dates from 12-MAY-24 to 01-JUN-24.

ID_LIVRARE	METODA_LIVRARE	PRET_LIVRARE	STARE	DATA_LIVRARE	ID_ADRESA	ID_CURIER	ID_MASINA
1	1 CURIER	12.9	EFFECTUATA	12-MAY-24	1	2	7
2	2 EASYBOX	6.9	EFFECTUATA	15-MAY-24	4	3	10
3	3 EASYBOX	6.9	EFFECTUATA	15-MAY-24	4	3	10
4	4 CURIER	16.99	IN CURS DE LIVRARE	20-MAY-24	9	1	5
5	5 FANBOX	10	EFFECTUATA	20-FEB-24	8	5	1
6	6 CURIER	16.99	EFFECTUATA	02-DEC-23	10	7	11
7	7 SAMEDAY	0	IN CURS DE PROCESARE	17-MAY-24	7	10	2
8	8 CURIER	19.99	EFFECTUATA	16-APR-24	6	1	5
9	9 CURIER	16.99	IN CURS DE LIVRARE	26-JUN-23	2	6	3
10	10 EASYBOX	0	IN CURS DE PROCESARE	16-MAY-24	9	11	8
11	11 CURIER	14.99	EFFECTUATA	30-MAR-24	5	4	12
12	12 FANBOX	10.5	EFFECTUATA	22-MAR-24	3	6	4
13	13 CURIER	19.99	IN CURS DE LIVRARE	01-JUN-24	2	12	9

Comanda

```

CREATE TABLE COMANDA (
    ID_COMANDA INT DEFAULT COMANDA_SEQ.NEXTVAL,
    TOTAL_PLATA NUMBER(10,2) NOT NULL,
    STARE_COMANDA VARCHAR2(255),
    DATA_COMANDA DATE,
    ID_UTILIZATOR INT,
    ID_LIVRARE INT UNIQUE,
    CONSTRAINT CHEIE_PRIMARA_COMANDA PRIMARY KEY (ID_COMANDA),
    CONSTRAINT TOTAL_PLATA_CHECK CHECK (TOTAL_PLATA >= 0),
    CONSTRAINT FK_COMANDA_UTILIZATOR FOREIGN KEY(ID_UTILIZATOR)
        REFERENCES UTILIZATOR(ID_UTILIZATOR),
    CONSTRAINT FK_COMANDA_LIVRARE FOREIGN KEY(ID_LIVRARE)
        REFERENCES LIVRARE(ID_LIVRARE),
    CONSTRAINT STARE_COMANDA_CHECK CHECK (STARE_COMANDA IN ('IN
CURS DE PROCESARE', 'EFFECTUATA', 'IN CURS DE PREGATIRE'))
);

```

```

INSERT INTO COMANDA (TOTAL_PLATA, STARE_COMANDA, DATA_COMANDA,
ID_UTILIZATOR, ID_LIVRARE) VALUES(100,'EFFECTUATA','23-FEB-2023', 1,1);

```

INSERT INTO COMANDA (TOTAL_PLATA, STARE_COMANDA, DATA_COMANDA, ID_UTILIZATOR, ID_LIVRARE) VALUES(106.9,'IN CURS DE PREGATIRE','10-MAY-2024', 3,2);

INSERT INTO COMANDA (TOTAL_PLATA, STARE_COMANDA, DATA_COMANDA, ID_UTILIZATOR, ID_LIVRARE) VALUES(237.6,'IN CURS DE PREGATIRE','9-MAY-2024', 5,3);

INSERT INTO COMANDA (TOTAL_PLATA, STARE_COMANDA, DATA_COMANDA, ID_UTILIZATOR, ID_LIVRARE) VALUES(129.5,'IN CURS DE PREGATIRE','15-MAY-2024',6 ,10);

INSERT INTO COMANDA (TOTAL_PLATA, STARE_COMANDA, DATA_COMANDA, ID_UTILIZATOR, ID_LIVRARE) VALUES (68.4, 'EFECTUATA', '23-FEB-2023', 1, 13);

INSERT INTO COMANDA (TOTAL_PLATA, STARE_COMANDA, DATA_COMANDA, ID_UTILIZATOR, ID_LIVRARE) VALUES (126.9,'IN CURS DE PREGATIRE','10-MAY-2024', 2, 12);

INSERT INTO COMANDA (TOTAL_PLATA, STARE_COMANDA, DATA_COMANDA, ID_UTILIZATOR, ID_LIVRARE) VALUES (117.6, 'IN CURS DE PREGATIRE', '9-MAY-2024', 3, 11);

INSERT INTO COMANDA (TOTAL_PLATA, STARE_COMANDA, DATA_COMANDA, ID_UTILIZATOR, ID_LIVRARE) VALUES (66.98, 'IN CURS DE PREGATIRE', '15-MAY-2024', 4, 4);

INSERT INTO COMANDA (TOTAL_PLATA, STARE_COMANDA, DATA_COMANDA, ID_UTILIZATOR, ID_LIVRARE) VALUES (62.5, 'EFECTUATA', '20-FEB-2024', 6, 6);

INSERT INTO COMANDA (TOTAL_PLATA, STARE_COMANDA, DATA_COMANDA, ID_UTILIZATOR, ID_LIVRARE) VALUES (216.98, 'EFECTUATA', '02-DEC-2023', 7, 7);

```
INSERT INTO COMANDA (TOTAL_PLATA, STARE_COMANDA, DATA_COMANDA,  
ID_UTILIZATOR, ID_LIVRARE) VALUES (116.98, 'IN CURS DE PROCESARE', '20-  
MAY-2024', 5, 5);
```

```
INSERT INTO COMANDA (TOTAL_PLATA, STARE_COMANDA, DATA_COMANDA,  
ID_UTILIZATOR, ID_LIVRARE) VALUES (250.68, 'IN CURS DE PROCESARE', '17-  
MAY-2024', 8, 8);
```

```
INSERT INTO COMANDA (TOTAL_PLATA, STARE_COMANDA, DATA_COMANDA,  
ID_UTILIZATOR, ID_LIVRARE) VALUES (346.48, 'EFECTUATA', '16-APR-2024', 9, 9);
```

```
SELECT * FROM COMANDA;
```

The screenshot displays two instances of Oracle SQL Developer running on the same machine, illustrating a race condition or concurrent execution issue. Both sessions show the same sequence of operations:

- Session 1 (Top):
 - Creates the table `COMANDA` with constraints.
 - Inserts 6 rows into `COMANDA`. The first row has `ID_PLATA = 100`, `STARE_COMANDA = 'IN CURS DE PREGATIRE'`, and `ID_UTILIZATOR = 1`.
 - Inserts 5 rows into `LIVRARE` with values ranging from `104.9` to `129.5`.
- Session 2 (Bottom):
 - Creates the table `COMANDA` with constraints.
 - Inserts 6 rows into `COMANDA`. The first row has `ID_PLATA = 106.9`, `STARE_COMANDA = 'IN CURS DE PREGATIRE'`, and `ID_UTILIZATOR = 3`.
 - Inserts 5 rows into `LIVRARE` with values ranging from `117.6` to `126.9`.

Both sessions complete successfully, indicating that the race condition did not affect the final state of the database.

Script Output | Query Result | SQL | All Rows Fetched: 13 in 0.194 seconds

ID_COMANDA	TOTAL_PLATA	STARE_COMANDA	DATA_COMANDA	ID_UTILIZATOR	ID_LIVRARE
1	1	100 EFECTUATA	23-FEB-23	1	1
2	2	106.9 IN CURS DE PREGATIRE	10-MAY-24	3	2
3	3	237.6 IN CURS DE PREGATIRE	09-MAY-24	5	3
4	4	129.5 IN CURS DE PREGATIRE	15-MAY-24	6	10
5	5	68.4 EFECTUATA	23-FEB-23	1	13
6	6	126.9 IN CURS DE PREGATIRE	10-MAY-24	2	12
7	7	117.6 IN CURS DE PREGATIRE	09-MAY-24	3	11
8	8	66.98 IN CURS DE PREGATIRE	15-MAY-24	4	4
9	9	62.5 EFECTUATA	20-FEB-24	6	6
10	10	216.98 EFECTUATA	02-DEC-23	7	7
11	11	116.98 IN CURS DE PROCESARE	20-MAY-24	5	5
12	12	250.68 IN CURS DE PROCESARE	17-MAY-24	8	8
13	13	346.48 EFECTUATA	16-APR-24	9	9

Card

```
CREATE TABLE CARD (
    ID_CARD INT DEFAULT CARD_SEQ.NEXTVAL,
    NUMAR_CARD VARCHAR2(16) NOT NULL,
    NUME_PROPIETAR VARCHAR2(255),
    CVV VARCHAR2(4),
    DATA_EXPIRARE DATE,
    ID_UTILIZATOR INT,
    CONSTRAINT VERIFICARE_NUMAR_CARD CHECK ( LENGTH(NUMAR_CARD)=16),
    CONSTRAINT VERIFICARE_CVV CHECK ( LENGTH(CVV)=3 or LENGTH(CVV)=4),
    CONSTRAINT CHEIE_PRIMARA_CARD PRIMARY KEY (ID_CARD),
    CONSTRAINT NUMAR_CARD_UNIC UNIQUE (NUMAR_CARD)
);
```

```
INSERT INTO CARD (numar_card, nume_propietar, CVV, data_expirare, id_utilizator)
VALUES ('0000111122223333','EMMA MACIUCA', '123', '01-FEB-2026', 1);
```

```
INSERT INTO CARD (numar_card, nume_propietar, CVV, data_expirare, id_utilizator)
VALUES ('1800789243671234','IULIANA MARIN', '812', '01-MAR-2028', 6);
```

```
INSERT INTO CARD (numar_card, nume_propietar, CVV, data_expirare, id_utilizator)
VALUES ('8902489019003532','POPESCU ANDREEA IOANA', '602', '01-APR-2029', 2);

INSERT INTO CARD (numar_card, nume_propietar, CVV, data_expirare, id_utilizator)
VALUES ('2310649119007413','ION ANDREI', '241', '01-OCT-2025', 3);

INSERT INTO CARD (numar_card, nume_propietar, CVV, data_expirare, id_utilizator)
VALUES ('1475148418001975','ANDREESCU MARIA', '207', '01-NOV-2026', 4);

INSERT INTO CARD (numar_card, nume_propietar, CVV, data_expirare, id_utilizator)
VALUES ('3485120719001298','MACIUCA SARA', '109', '01-DEC-2029', 5);

INSERT INTO CARD (numar_card, nume_propietar, CVV, data_expirare, id_utilizator)
VALUES ('4723699318001431','POP DARIA', '192', '01-JAN-2030', 7);

INSERT INTO CARD (numar_card, nume_propietar, CVV, data_expirare, id_utilizator)
VALUES ('2367646716001239','MARES DARIUS', '276', '01-APR-2025',9);

INSERT INTO CARD (numar_card, nume_propietar, CVV, data_expirare, id_utilizator)
VALUES ('6892163417001465','POPA MARA', '225', '01-MAY-2027', 8);

INSERT INTO CARD (numar_card, nume_propietar, CVV, data_expirare, id_utilizator)
VALUES ('1354235715003687','BALAN IOANA', '913', '01-JUL-2024', 10);

INSERT INTO CARD (numar_card, nume_propietar, CVV, data_expirare, id_utilizator)
VALUES ('1565611319003575','BALAN IOANA', '637', '01-DEC-2029', 10);
```

```
SELECT * FROM CARD;
```

Oracle SQL Developer tutorial

File Edit View Navigate Run Source Team Tools Window Help

Connections Welcome Page Autoral

Worksheet Query Builder

```
-- TABELA CARD
CREATE TABLE CARD (
    ID_CARD NUMBER(10) NOT NULL,
    CARD_SEQ NUMBER(10),
    NUMAR_CARD VARCHAR2(16) NOT NULL,
    NUME_PROPRIETAR VARCHAR2(255),
    CVV VARCHAR2(4),
    DATA_EXPIRARE DATE,
    ID_UTILIZATOR INT,
    CONSTRAINT VERIFICARE_NUMAR_CARD CHECK ( LENGTH(NUMAR_CARD) = 16 ),
    CONSTRAINT VERIFICARE_CVV CHECK ( LENGTH(CVV)=3 or LENGTH(CVV)=4),
    CONSTRAINT CHEIE_PRIMARA_CARD PRIMARY KEY (ID_CARD),
    CONSTRAINT NUMAR_CARD_UNIC UNIQUE (NUMAR_CARD)
);

INSERT INTO CARD (numar_card, nume_propietar, CVV, data_expirare, id_utilizator) VALUES ('0000111122223333', 'EMIL MACHIUCA', '123', '01-FEB-2026', 1);
INSERT INTO CARD (numar_card, nume_propietar, CVV, data_expirare, id_utilizator) VALUES ('180079243671234', 'IULIANA MARIS', '812', '01-MAR-2028', 6);
INSERT INTO CARD (numar_card, nume_propietar, CVV, data_expirare, id_utilizator) VALUES ('2310649119007412', 'IOAN ANDREI', '241', '01-OCT-2029', 2);
INSERT INTO CARD (numar_card, nume_propietar, CVV, data_expirare, id_utilizator) VALUES ('2310649119007412', 'IOAN ANDREI', '241', '01-OCT-2029', 3);
INSERT INTO CARD (numar_card, nume_propietar, CVV, data_expirare, id_utilizator) VALUES ('1475148418001975', 'ANDREEESCU MARIA', '207', '01-NOV-2024', 4);
INSERT INTO CARD (numar_card, nume_propietar, CVV, data_expirare, id_utilizator) VALUES ('0485120719001298', 'MACHIUCA SARA', '109', '01-DEC-2029', 5);
INSERT INTO CARD (numar_card, nume_propietar, CVV, data_expirare, id_utilizator) VALUES ('4723693180014131', 'POP DARIU', '182', '01-JAN-2030', 7);
INSERT INTO CARD (numar_card, nume_propietar, CVV, data_expirare, id_utilizator) VALUES ('238764716001239', 'MARES DARUS', '276', '01-APR-2025', 9);
INSERT INTO CARD (numar_card, nume_propietar, CVV, data_expirare, id_utilizator) VALUES ('6892163417001485', 'PORE MARA', '225', '01-MAY-2027', 6);
INSERT INTO CARD (numar_card, nume_propietar, CVV, data_expirare, id_utilizator) VALUES ('1354255715003687', 'BALAN IOANA', '913', '01-NOV-2024', 10);
INSERT INTO CARD (numar_card, nume_propietar, CVV, data_expirare, id_utilizator) VALUES ('1565611319003575', 'BALAN IOANA', '637', '01-DEC-2029', 10);

Table CARD created.

1 row inserted.

1 row inserted.
```

Script Output X Query Result X Task completed in 0.494 seconds

73°F Sunny 10:27 AM 6/7/2024

Oracle SQL Developer tutorial

File Edit View Navigate Run Source Team Tools Window Help

Connections Welcome Page Autoral

Worksheet Query Builder

```
-- TABELA CARD
CREATE TABLE CARD (
    ID_EXPIRARE DATE,
    ID_UTILIZATOR INT,
    CONSTRAINT VERIFICARE_NUMAR_CARD CHECK ( LENGTH(NUMAR_CARD) = 16 ),
    CONSTRAINT VERIFICARE_CVV CHECK ( LENGTH(CVV)=3 or LENGTH(CVV)=4),
    CONSTRAINT CHEIE_PRIMARA_CARD PRIMARY KEY (ID_CARD),
    CONSTRAINT NUMAR_CARD_UNIC UNIQUE (NUMAR_CARD)
);

INSERT INTO CARD (numar_card, nume_propietar, CVV, data_expirare, id_utilizator) VALUES ('0000111122223333', 'EMIL MACHIUCA', '123', '01-FEB-2026', 1);
INSERT INTO CARD (numar_card, nume_propietar, CVV, data_expirare, id_utilizator) VALUES ('180079243671234', 'IULIANA MARIS', '812', '01-MAR-2028', 6);
INSERT INTO CARD (numar_card, nume_propietar, CVV, data_expirare, id_utilizator) VALUES ('2310649119007412', 'IOAN ANDREI', '241', '01-OCT-2029', 2);
INSERT INTO CARD (numar_card, nume_propietar, CVV, data_expirare, id_utilizator) VALUES ('2310649119007412', 'IOAN ANDREI', '241', '01-OCT-2029', 3);
INSERT INTO CARD (numar_card, nume_propietar, CVV, data_expirare, id_utilizator) VALUES ('1475148418001975', 'ANDREEESCU MARIA', '207', '01-NOV-2024', 4);
INSERT INTO CARD (numar_card, nume_propietar, CVV, data_expirare, id_utilizator) VALUES ('0485120719001298', 'MACHIUCA SARA', '109', '01-DEC-2029', 5);
INSERT INTO CARD (numar_card, nume_propietar, CVV, data_expirare, id_utilizator) VALUES ('4723693180014131', 'POP DARIU', '182', '01-JAN-2030', 7);
INSERT INTO CARD (numar_card, nume_propietar, CVV, data_expirare, id_utilizator) VALUES ('238764716001239', 'MARES DARUS', '276', '01-APR-2025', 9);
INSERT INTO CARD (numar_card, nume_propietar, CVV, data_expirare, id_utilizator) VALUES ('6892163417001485', 'PORE MARA', '225', '01-MAY-2027', 6);
INSERT INTO CARD (numar_card, nume_propietar, CVV, data_expirare, id_utilizator) VALUES ('1354255715003687', 'BALAN IOANA', '913', '01-NOV-2024', 10);
INSERT INTO CARD (numar_card, nume_propietar, CVV, data_expirare, id_utilizator) VALUES ('1565611319003575', 'BALAN IOANA', '637', '01-DEC-2029', 10);
SELECT * FROM CARD;
```

Script Output X Query Result X Task completed in 0.484 seconds

1 row inserted.

73°F Sunny 10:27 AM 6/7/2024

Script Output x | Query Result x

SQL | All Rows Fetched: 12 in 0.045 seconds

	ID_CARD	NUMAR_CARD	NUME_PROPIETAR	CVV	DATA_EXPIRARE	ID_UTILIZATOR	
1	1 0000111122223333	EMMA MACIUCA		123	01-FEB-26	1	
2	2 1800789243671234	IULIANA MARIN		812	01-MAR-28	6	
3	3 8902489019003532	POPESCU ANDREEA IOANA		602	01-APR-29	2	
4	4 2310649119007413	ION ANDREI		241	01-OCT-25	3	
5	5 1475148418001975	ANDREESCU MARIA		207	01-NOV-26	4	
6	6 3485120719001298	MACIUCA SARA		109	01-DEC-29	5	
7	7 4723699318001431	POP DARIA		192	01-JAN-30	7	
8	8 2367646716001239	MARES DARIUS		276	01-APR-25	9	
9	9 6892163417001465	POPA MARA		225	01-MAY-27	8	
10	10 1354235715003687	BALAN IOANA		913	01-JUL-24	10	
11	11 1565611319003575	BALAN IOANA		637	01-DEC-29	10	
12	12 8243186518006295	POPESCU ANDREEA IOANA		265	01-APR-28	2	

Plata

```

CREATE TABLE PLATA (
    ID_PLATA INT DEFAULT PLATA_SEQ.NEXTVAL,
    VALOARE NUMBER(10,2) NOT NULL,
    DATA_PLATA DATE,
    STARE VARCHAR2(255),
    ID_CARD INT,
    ID_COMANDA INT,
    CONSTRAINT CHEIE_PRIMARA_PLATA PRIMARY KEY (ID_PLATA),
    CONSTRAINT VALOARE_CHECK CHECK (VALOARE >= 0),
    CONSTRAINT FK_PLATA_CARD FOREIGN KEY(ID_CARD) REFERENCES CARD(ID_CARD),
    CONSTRAINT FK_PLATA_COMANDA FOREIGN KEY(ID_COMANDA) REFERENCES COMANDA(ID_COMANDA),
    CONSTRAINT STARE_PLATA_CHECK CHECK (STARE IN ('IN CURS DE PROCESARE', 'EFECTUATA', 'RESPINSA'))
);

```

INSERTINTO PLATA (VALOARE,DATA_PLATA,STARE,ID_CARD,ID_COMANDA)
VALUES(100,'23-FEB-2023','RESPINSA',1,1);

INSERTINTO PLATA (VALOARE,DATA_PLATA,STARE,ID_CARD,ID_COMANDA)
VALUES(129.5,'15-MAY-2024','EFFECTUATA',2,4);

INSERTINTO PLATA (VALOARE,DATA_PLATA,STARE,ID_CARD,ID_COMANDA)
VALUES(106.9,'10-MAY-2024','EFFECTUATA',4,2);

INSERTINTO PLATA (VALOARE,DATA_PLATA,STARE,ID_CARD,ID_COMANDA)
VALUES(126.9,'10-MAY-2024','RESPINSA',3,6);

INSERTINTO PLATA (VALOARE,DATA_PLATA,STARE,ID_CARD,ID_COMANDA)
VALUES(126.9,'10-MAY-2024','EFFECTUATA',12,6);

INSERTINTO PLATA (VALOARE,DATA_PLATA,STARE,ID_CARD,ID_COMANDA)
VALUES(237.6,'09-MAY-2024','EFFECTUATA',6,3);

INSERTINTO PLATA (VALOARE,DATA_PLATA,STARE,ID_CARD,ID_COMANDA)
VALUES(68.4,'23-FEB-2023','EFFECTUATA',1,5);

INSERTINTO PLATA (VALOARE,DATA_PLATA,STARE,ID_CARD,ID_COMANDA)
VALUES(117.6,'09-MAY-2024','IN CURS DE PROCESARE',4,7);

INSERTINTO PLATA (VALOARE,DATA_PLATA,STARE,ID_CARD,ID_COMANDA)
VALUES(66.98,'15-MAY-2024','IN CURS DE PROCESARE',5,8);

INSERTINTO PLATA (VALOARE,DATA_PLATA,STARE,ID_CARD,ID_COMANDA)
VALUES(62.5,'20-FEB-2024','EFFECTUATA',2,9);

INSERTINTO PLATA (VALOARE,DATA_PLATA,STARE,ID_CARD,ID_COMANDA)
VALUES(216.98,'02-DEC-2023','EFFECTUATA',7,10);

INSERTINTO PLATA (VALOARE,DATA_PLATA,STARE,ID_CARD,ID_COMANDA)
VALUES(116.98,'20-MAY-2024','RESPINSA',6,11);

INSERTINTO PLATA (VALOARE,DATA_PLATA,STARE,ID_CARD,ID_COMANDA)
VALUES(250.68,'17-MAY-2024','IN CURS DE PROCESARE',9,12);

INSERTINTO PLATA (VALOARE,DATA_PLATA,STARE,ID_CARD,ID_COMANDA)
VALUES(346.48,'16-APR-2024','EFFECTUATA',8,13);

select * from plata;

Oracle SQL Developer : tutorial

File Edit View Navigate Run Source Team Tools Window Help

Connections Welcome Page tutorial

Workshop Query Builder

```
CREATE TABLE CARD (
    ID_CARD INT,
    ID_COMANDA INT,
    CONSTRAINT CHEIE_PRIMARA_PLATA PRIMARY KEY (ID_PLATA),
    CONSTRAINT VALORE_CHEIE_CHECK (VALOREARE >= 0)
);
CREATE TABLE PLATA (
    ID_PLATA INT,
    ID_CARD INT,
    ID_COMANDA INT,
    STARE VARCHAR(10),
    DATA_PLATA DATE,
    CONSTRAINT PLATA_CARD_FK FOREIGN KEY (ID_CARD) REFERENCES CARD (ID_CARD),
    CONSTRAINT PLATA_COMANDA_FK FOREIGN KEY (ID_COMANDA) REFERENCES COMANDA (ID_COMANDA),
    CONSTRAINT STARE_PLATA_CHECK CHECK (STARE IN ('IN CURS DE PROCESARE', 'EFFECTUATA', 'RESPISNA'))
);

INSERT INTO PLATA (VALORE, DATA_PLATA, STARE, ID_CARD, ID_COMANDA) VALUES(1100, '23-FEB-2023', 'RESPISNA', 1,1);
INSERT INTO PLATA (VALORE, DATA_PLATA, STARE, ID_CARD, ID_COMANDA) VALUES(1100, '05-MAY-2024', 'EFFECTUATA', 1,4);
INSERT INTO PLATA (VALORE, DATA_PLATA, STARE, ID_CARD, ID_COMANDA) VALUES(1100, '05-MAY-2024', 'EFFECTUATA', 1,4);
INSERT INTO PLATA (VALORE, DATA_PLATA, STARE, ID_CARD, ID_COMANDA) VALUES(116, 5, '10-MAY-2024', 'RESPISNA', 1,6);
INSERT INTO PLATA (VALORE, DATA_PLATA, STARE, ID_CARD, ID_COMANDA) VALUES(116, 5, '10-MAY-2024', 'EFFECTUATA', 1,6);
INSERT INTO PLATA (VALORE, DATA_PLATA, STARE, ID_CARD, ID_COMANDA) VALUES(127, 6, '09-MAY-2024', 'EFFECTUATA', 1,3);
INSERT INTO PLATA (VALORE, DATA_PLATA, STARE, ID_CARD, ID_COMANDA) VALUES(68, 4, '23-FEB-2023', 'EFFECTUATA', 1,5);
INSERT INTO PLATA (VALORE, DATA_PLATA, STARE, ID_CARD, ID_COMANDA) VALUES(117, 6, '09-MAY-2024', 'IN CURS DE PROCESARE', 4,7);
INSERT INTO PLATA (VALORE, DATA_PLATA, STARE, ID_CARD, ID_COMANDA) VALUES(117, 6, '09-MAY-2024', 'IN CURS DE PROCESARE', 4,8);
INSERT INTO PLATA (VALORE, DATA_PLATA, STARE, ID_CARD, ID_COMANDA) VALUES(162, 5, '20-TEZ-2024', 'EFFECTUATA', 8,19);
INSERT INTO PLATA (VALORE, DATA_PLATA, STARE, ID_CARD, ID_COMANDA) VALUES(216, 99, '02-DEC-2023', 'EFFECTUATA', 7,10);
INSERT INTO PLATA (VALORE, DATA_PLATA, STARE, ID_CARD, ID_COMANDA) VALUES(116, 99, '20-MAY-2024', 'RESPISNA', 6,11);
INSERT INTO PLATA (VALORE, DATA_PLATA, STARE, ID_CARD, ID_COMANDA) VALUES(250, 69, '17-MAY-2024', 'IN CURS DE PROCESARE', 9,12);
INSERT INTO PLATA (VALORE, DATA_PLATA, STARE, ID_CARD, ID_COMANDA) VALUES(346, 49, '16-APR-2024', 'EFFECTUATA', 8,13);

select * from plata;
```

Script Output x | Query Result x

Task completed in 0.383 seconds

```
1 row inserted.

1 row inserted.
```

73°F Sunny

1 Line 244 Column 20 | Insert | Modified | Windows |

Script Output | Query Result | SQL | All Rows Fetched: 14 in 0.068 seconds

ID_PLATA	VALOARE	DATA_PLATA	STARE	ID_CARD	ID_COMANDA
1	1	100 23-FEB-23	RESPINSA	1	1
2	2	129.5 15-MAY-24	EFECTUATA	2	4
3	3	106.9 10-MAY-24	EFECTUATA	4	2
4	4	126.9 10-MAY-24	RESPINSA	3	6
5	5	126.9 10-MAY-24	EFECTUATA	12	6
6	6	237.6 09-MAY-24	EFECTUATA	6	3
7	7	68.4 23-FEB-23	EFECTUATA	1	5
8	8	117.6 09-MAY-24	IN CURS DE PROCESARE	4	7
9	9	66.98 15-MAY-24	IN CURS DE PROCESARE	5	8
10	10	62.5 20-FEB-24	EFECTUATA	2	9
11	11	216.98 02-DEC-23	EFECTUATA	7	10
12	12	116.98 20-MAY-24	RESPINSA	6	11
13	13	250.68 17-MAY-24	IN CURS DE PROCESARE	9	12
14	14	346.48 16-APR-24	EFECTUATA	8	13

Cos_Cumparaturi

```

CREATE TABLE COS_CUMPARATURI (
    ID_COS INT DEFAULT COS_CUMPARATURI_SEQ.NEXTVAL,
    ID_UTILIZATOR INT UNIQUE,
    TOTAL_COS NUMBER(10,2),
    CONSTRAINT CHEIE_PRIMARA_COS_CUMPARATURI PRIMARY KEY (ID_COS),
    CONSTRAINT FK_COS_UTILIZATOR FOREIGN KEY(ID_UTILIZATOR)
    REFERENCES UTILIZATOR(ID_UTILIZATOR),
    CONSTRAINT TOTAL_COS_CHECK CHECK (TOTAL_COS >= 0)
);

```

```
INSERT INTO COS_CUMPARATURI (ID_UTILIZATOR, TOTAL_COS) VALUES (1,55.50);
```

```
INSERT INTO COS_CUMPARATURI (ID_UTILIZATOR, TOTAL_COS) VALUES (2,150);
```

```
INSERT INTO COS_CUMPARATURI (ID_UTILIZATOR, TOTAL_COS) VALUES (3,220);
```

```
INSERT INTO COS_CUMPARATURI (ID_UTILIZATOR, TOTAL_COS) VALUES (4,199.99);
```

```
INSERT INTO COS_CUMPARATURI (ID_UTILIZATOR, TOTAL_COS) VALUES  
(5,202.48);
```

```
INSERT INTO COS_CUMPARATURI (ID_UTILIZATOR, TOTAL_COS) VALUES (6,0);
```

```
INSERT INTO COS_CUMPARATURI (ID_UTILIZATOR, TOTAL_COS) VALUES  
(7,129.5);
```

```
INSERT INTO COS_CUMPARATURI (ID_UTILIZATOR, TOTAL_COS) VALUES (8,0);
```

```
INSERT INTO COS_CUMPARATURI (ID_UTILIZATOR, TOTAL_COS) VALUES (9,0);
```

```
INSERT INTO COS_CUMPARATURI (ID_UTILIZATOR, TOTAL_COS) VALUES  
(10,200.69);
```

```
INSERT INTO COS_CUMPARATURI (ID_UTILIZATOR, TOTAL_COS) VALUES (11,0);
```

```
SELECT * FROM COS_CUMPARATURI;
```

The screenshot shows the Oracle SQL Developer interface. The top menu bar includes File, Edit, View, Navigate, Run, Source, Tools, Window, and Help. The title bar says "Welcome Page" and "tutorial". The main area is a Worksheet titled "Query Builder". The code in the worksheet is:

```
--TABELA COS_CUMPARATURI;  
CREATE TABLE COS_CUMPARATURI (  
    ID_COS INT DEFAULT COS_CUMPARATORI_SEQ.NEXTVAL,  
    ID_UTILIZATOR INT,  
    TOTAL_COS NUMBER(10,2),  
    CONSTRAINT CKEY_ID_COS PRIMARY KEY (ID_COS),  
    CONSTRAINT FK_COS_UTILIZATOR FOREIGN KEY (ID_UTILIZATOR) REFERENCES UTILIZATOR(ID_UTILIZATOR),  
    CONSTRAINT TOTAL_COS_CHECK CHECK (TOTAL_COS >= 0)  
);  
INSERT INTO COS_CUMPARATURI (ID_UTILIZATOR, TOTAL_COS) VALUES (1,55.50);  
INSERT INTO COS_CUMPARATURI (ID_UTILIZATOR, TOTAL_COS) VALUES (1,150);  
INSERT INTO COS_CUMPARATURI (ID_UTILIZATOR, TOTAL_COS) VALUES (3,220);  
INSERT INTO COS_CUMPARATURI (ID_UTILIZATOR, TOTAL_COS) VALUES (4,199.99);  
INSERT INTO COS_CUMPARATURI (ID_UTILIZATOR, TOTAL_COS) VALUES (5,202.48);  
INSERT INTO COS_CUMPARATURI (ID_UTILIZATOR, TOTAL_COS) VALUES (6,0);  
INSERT INTO COS_CUMPARATURI (ID_UTILIZATOR, TOTAL_COS) VALUES (7,129.5);  
INSERT INTO COS_CUMPARATURI (ID_UTILIZATOR, TOTAL_COS) VALUES (8,0);  
INSERT INTO COS_CUMPARATURI (ID_UTILIZATOR, TOTAL_COS) VALUES (9,0);  
INSERT INTO COS_CUMPARATURI (ID_UTILIZATOR, TOTAL_COS) VALUES (10,200.69);  
SELECT * FROM COS_CUMPARATURI;
```

The bottom status bar shows "Table COS_CUMPARATURI created.", "1 row inserted.", and "1 row inserted." repeated six times. The system tray indicates it's 10:32 AM on 6/7/2024.

ID_COS	ID_UTILIZATOR	TOTAL_COS
1	1	55.5
2	2	150
3	3	220
4	4	199.99
5	5	202.48
6	6	0
7	7	129.5
8	8	0
9	9	0
10	10	200.69
11	11	0

Lista_Dorinte

```

CREATE TABLE LISTA_DORINTE (
    ID_LISTA INT DEFAULT LISTA_DORINTE_SEQ.NEXTVAL,
    ID_UTILIZATOR INT,
    DENUMIRE_LISTA VARCHAR2(255) NOT NULL,
    CONSTRAINT CHEIE_PRIMARA_LISTA PRIMARY KEY (ID_LISTA),
    CONSTRAINT FK_LISTA_UTILIZATOR FOREIGN KEY(ID_UTILIZATOR)
    REFERENCES UTILIZATOR(ID_UTILIZATOR)
);
INSERT INTO LISTA_DORINTE (ID_UTILIZATOR, DENUMIRE_LISTA) VALUES
(1,'PANTOFI');
INSERT INTO LISTA_DORINTE (ID_UTILIZATOR, DENUMIRE_LISTA) VALUES (1,
'CADOURI');
INSERT INTO LISTA_DORINTE (ID_UTILIZATOR, DENUMIRE_LISTA) VALUES (2,
'HAINE SPORT');
INSERT INTO LISTA_DORINTE (ID_UTILIZATOR, DENUMIRE_LISTA) VALUES (2,
'CASUAL');
INSERT INTO LISTA_DORINTE (ID_UTILIZATOR, DENUMIRE_LISTA) VALUES (3,
'OFFICE');
```

```
INSERT INTO LISTA_DORINTE (ID_UTILIZATOR, DENUMIRE_LISTA) VALUES (4, 'HAINE VACANTA');
```

```
INSERT INTO LISTA_DORINTE (ID_UTILIZATOR, DENUMIRE_LISTA) VALUES (5, 'HAINE TOAMNA');
```

```
INSERT INTO LISTA_DORINTE (ID_UTILIZATOR, DENUMIRE_LISTA) VALUES (5, 'HAINE VARA');
```

```
INSERT INTO LISTA_DORINTE (ID_UTILIZATOR, DENUMIRE_LISTA) VALUES (6, 'HAINE IARNA');
```

```
INSERT INTO LISTA_DORINTE (ID_UTILIZATOR, DENUMIRE_LISTA) VALUES (7, 'HAINE PRIMAVARA');
```

```
INSERT INTO LISTA_DORINTE (ID_UTILIZATOR, DENUMIRE_LISTA) VALUES (8, 'HAINE VARA');
```

```
INSERT INTO LISTA_DORINTE (ID_UTILIZATOR, DENUMIRE_LISTA) VALUES (9, 'DIVERSE');
```

```
INSERT INTO LISTA_DORINTE (ID_UTILIZATOR, DENUMIRE_LISTA) VALUES (10, 'FAMILIE');
```

```
INSERT INTO LISTA_DORINTE (ID_UTILIZATOR, DENUMIRE_LISTA) VALUES (10, 'FITNESS');
```

```
SELECT * FROM LISTA_DORINTE;
```

The screenshot shows the Oracle SQL Developer interface. The top window is titled 'Query Builder' and contains the following SQL code:

```
CREATE TABLE LISTA_DORINTE (
    ID_LISTA INT DEFAULT LISTA_DORINTE_SEQ.NEXTVAL,
    ID_UTILIZATOR INT,
    DENUMIRE_LISTA VARCHAR2(150) NOT NULL,
    CONSTRAINT CKF_PRIMARA_LISTA PRIMARY KEY (ID_LISTA),
    CONSTRAINT FK_LISTA_UTILIZATOR FOREIGN KEY (ID_UTILIZATOR) REFERENCES UTILIZATOR(ID_UTILIZATOR)
);
INSERT INTO LISTA_DORINTE (ID_UTILIZATOR, DENUMIRE_LISTA) VALUES (1, 'FATOFI');
INSERT INTO LISTA_DORINTE (ID_UTILIZATOR, DENUMIRE_LISTA) VALUES (2, 'CLOUTI');
INSERT INTO LISTA_DORINTE (ID_UTILIZATOR, DENUMIRE_LISTA) VALUES (2, 'HAINE SPORT');
INSERT INTO LISTA_DORINTE (ID_UTILIZATOR, DENUMIRE_LISTA) VALUES (2, 'CASOULE');
INSERT INTO LISTA_DORINTE (ID_UTILIZATOR, DENUMIRE_LISTA) VALUES (3, 'HAINE VESTIMENTA');
INSERT INTO LISTA_DORINTE (ID_UTILIZATOR, DENUMIRE_LISTA) VALUES (4, 'HAINE VACANTA');
INSERT INTO LISTA_DORINTE (ID_UTILIZATOR, DENUMIRE_LISTA) VALUES (5, 'HAINE TOAMNA');
INSERT INTO LISTA_DORINTE (ID_UTILIZATOR, DENUMIRE_LISTA) VALUES (5, 'HAINE VARA');
INSERT INTO LISTA_DORINTE (ID_UTILIZATOR, DENUMIRE_LISTA) VALUES (6, 'HAINE IARNA');
INSERT INTO LISTA_DORINTE (ID_UTILIZATOR, DENUMIRE_LISTA) VALUES (7, 'HAINE PRIMAVARA');
INSERT INTO LISTA_DORINTE (ID_UTILIZATOR, DENUMIRE_LISTA) VALUES (8, 'HAINE VARA');
INSERT INTO LISTA_DORINTE (ID_UTILIZATOR, DENUMIRE_LISTA) VALUES (9, 'DIVERSE');
INSERT INTO LISTA_DORINTE (ID_UTILIZATOR, DENUMIRE_LISTA) VALUES (10, 'FAMILIE');
INSERT INTO LISTA_DORINTE (ID_UTILIZATOR, DENUMIRE_LISTA) VALUES (10, 'FITNESS');
SELECT * FROM LISTA_DORINTE;
```

The bottom window is titled 'Script Output' and shows the results of the execution:

```
Table LISTA_DORINTE created.

1 row inserted.

1 row inserted.
```

The status bar at the bottom right indicates the task completed in 0.35 seconds.

The screenshot shows a SQL developer interface with two tabs: 'Script Output' and 'Query Result'. The 'Query Result' tab is active, displaying a table with 14 rows. The table has three columns: 'ID_LISTA', 'ID_UTILIZATOR', and 'DENUMIRE_LISTA'. The data is as follows:

ID_LISTA	ID_UTILIZATOR	DENUMIRE_LISTA
1	1	1 PANTOFI
2	2	1 CADOURI
3	3	2 HAINE SPORT
4	4	2 CASUAL
5	5	3 OFFICE
6	6	4 HAINE VACANTA
7	7	5 HAINE TOAMNA
8	8	5 HAINE VARA
9	9	6 HAINE IARNA
10	10	7 HAINE PRIMAVERA
11	11	8 HAINE VARA
12	12	9 DIVERSE
13	13	10 FAMILIE
14	14	10 FITNESS

Categorie

```
CREATE TABLE CATEGORIE (
    ID_CATEGORIE INT DEFAULT CATEGORIE_SEQ.NEXTVAL,
    DENUMIRE_CATEGORIE VARCHAR2(255) NOT NULL,
    CONSTRAINT CHEIE_PRIMARA_CATEGORIE PRIMARY KEY (ID_CATEGORIE)
);
```

```
INSERT INTO CATEGORIE (DENUMIRE_CATEGORIE) VALUES ('TRICOU');
INSERT INTO CATEGORIE (DENUMIRE_CATEGORIE) VALUES ('PANTALONI');
INSERT INTO CATEGORIE (DENUMIRE_CATEGORIE) VALUES ('ROCHIE');
INSERT INTO CATEGORIE (DENUMIRE_CATEGORIE) VALUES ('GEACA');
INSERT INTO CATEGORIE (DENUMIRE_CATEGORIE) VALUES ('VESTA');
INSERT INTO CATEGORIE (DENUMIRE_CATEGORIE) VALUES ('CAMASA');
INSERT INTO CATEGORIE (DENUMIRE_CATEGORIE) VALUES ('GEANTA');
INSERT INTO CATEGORIE (DENUMIRE_CATEGORIE) VALUES ('OCHELARI DE SOARE');
```

```

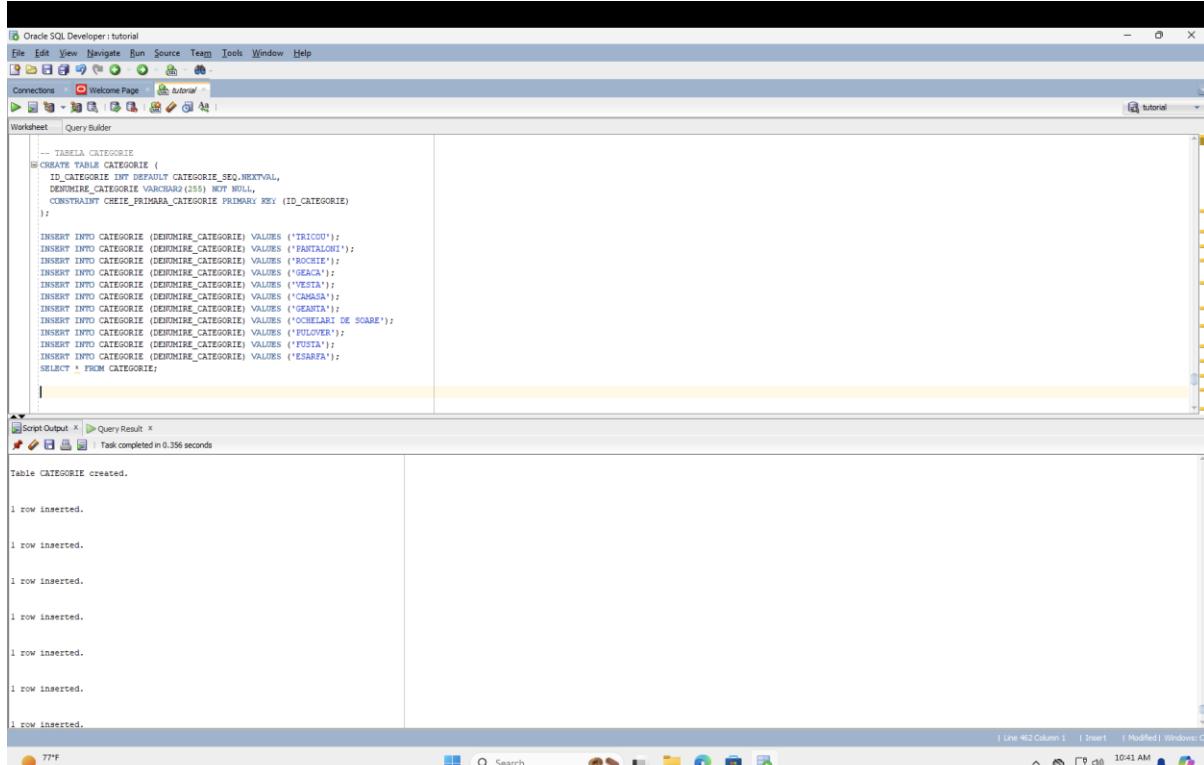
INSERT INTO CATEGORIE (DENUMIRE_CATEGORIE) VALUES ('PULOVER');

INSERT INTO CATEGORIE (DENUMIRE_CATEGORIE) VALUES ('FUSTA');

INSERT INTO CATEGORIE (DENUMIRE_CATEGORIE) VALUES ('ESARFA');

SELECT * FROM CATEGORIE;

```



The screenshot shows the Oracle SQL Developer interface with the following details:

- Worksheet Tab:** Contains the SQL script used to create the table and insert data.
- Script Output Tab:** Shows the execution results, indicating 11 rows inserted successfully.
- Query Result Tab:** Displays the contents of the CATEGORIE table.
- Table Data View:** Shows the 11 rows of data inserted into the CATEGORIE table.

```

-- TABLA CATEGORIE
CREATE TABLE CATEGORIE (
    ID_CATEGORIE INT DEFAULT CATEGORIE_SEQ.NEXTVAL,
    DENUMIRE_CATEGORIE VARCHAR2(155) NOT NULL,
    CONSTRAINT CATEGORIE_PRIMARY KEY (ID_CATEGORIE)
);

INSERT INTO CATEGORIE (DENUMIRE_CATEGORIE) VALUES ('TRICOU');
INSERT INTO CATEGORIE (DENUMIRE_CATEGORIE) VALUES ('PANTALONI');
INSERT INTO CATEGORIE (DENUMIRE_CATEGORIE) VALUES ('ROCHIE');
INSERT INTO CATEGORIE (DENUMIRE_CATEGORIE) VALUES ('GEACA');
INSERT INTO CATEGORIE (DENUMIRE_CATEGORIE) VALUES ('VESTA');
INSERT INTO CATEGORIE (DENUMIRE_CATEGORIE) VALUES ('CAMASA');
INSERT INTO CATEGORIE (DENUMIRE_CATEGORIE) VALUES ('GEANTA');
INSERT INTO CATEGORIE (DENUMIRE_CATEGORIE) VALUES ('OCHELARI DE SOARE');
INSERT INTO CATEGORIE (DENUMIRE_CATEGORIE) VALUES ('PULOVER');
INSERT INTO CATEGORIE (DENUMIRE_CATEGORIE) VALUES ('FUSTA');
INSERT INTO CATEGORIE (DENUMIRE_CATEGORIE) VALUES ('ESARFA');

SELECT * FROM CATEGORIE;

```

Table CATEGORIE created.

1 row inserted.
1 row inserted.

ID_CATEGORIE	DENUMIRE_CATEGORIE
1	1 TRICOU
2	2 PANTALONI
3	3 ROCHIE
4	4 GEACA
5	5 VESTA
6	6 CAMASA
7	7 GEANTA
8	8 OCHELARI DE SOARE
9	9 PULOVER
10	10 FUSTA
11	11 ESARFA

Produs

```

CREATE TABLE PRODUS (
    ID_PRODUS INT DEFAULT PRODUS_SEQ.NEXTVAL,

```

```

DENUMIRE_PRODUS VARCHAR2(255),
PRET NUMBER(10,2) NOT NULL,
STOC NUMBER(10),
ID_CATEGORIE INT,
CONSTRAINT CHEIE_PRIMARA_PRODUS PRIMARY KEY (ID_PRODUS),
CONSTRAINT FK_PRODUS_CATEGORIE FOREIGN KEY (ID_CATEGORIE)
REFERENCES CATEGORIE(ID_CATEGORIE)

);

INSERT INTO PRODUS (DENUMIRE_PRODUS, PRET, STOC, ID_CATEGORIE)
VALUES ('TRICOU ALBASTRU',55.50,10,1);

INSERT INTO PRODUS (DENUMIRE_PRODUS, PRET, STOC, ID_CATEGORIE)
VALUES ('PANTALONI FORMALI',120.00,6,2);

INSERT INTO PRODUS (DENUMIRE_PRODUS, PRET, STOC, ID_CATEGORIE)
VALUES ('CAMASA DE IN',110.70,20,6);

INSERT INTO PRODUS (DENUMIRE_PRODUS, PRET, STOC, ID_CATEGORIE)
VALUES ('TRICOU GRI',49.99,25,1);

INSERT INTO PRODUS (DENUMIRE_PRODUS, PRET, STOC, ID_CATEGORIE)
VALUES ('ROCHIE DE PLAJA',99.99,10,3);

INSERT INTO PRODUS (DENUMIRE_PRODUS, PRET, STOC, ID_CATEGORIE)
VALUES ('OCHELARI DE SOARE COPII',52.50,5,8);

INSERT INTO PRODUS (DENUMIRE_PRODUS, PRET, STOC, ID_CATEGORIE)
VALUES ('GEACA DE IARNA',199.99,35,4);

INSERT INTO PRODUS (DENUMIRE_PRODUS, PRET, STOC, ID_CATEGORIE)
VALUES ('CAMASA VERDE',89.99,17,6);

INSERT INTO PRODUS (DENUMIRE_PRODUS, PRET, STOC, ID_CATEGORIE)
VALUES ('GEANTA DE UMAR',129.50,20,7);

INSERT INTO PRODUS (DENUMIRE_PRODUS, PRET, STOC, ID_CATEGORIE)
VALUES ('TRICOU IN DUNGI',49.99,15,1);

INSERT INTO PRODUS (DENUMIRE_PRODUS, PRET, STOC, ID_CATEGORIE)
VALUES ('VESTA ALBASTRA', 100 ,20,5);

INSERT INTO PRODUS (DENUMIRE_PRODUS, PRET, STOC, ID_CATEGORIE)
VALUES ('FUSTA MIDI', 75.00, 12, 10);

```

```
INSERT INTO PRODUS (DENUMIRE_PRODUS, PRET, STOC, ID_CATEGORIE)  
VALUES ('PULOVER DE LANA', 150.00, 8, 9);
```

```
INSERT INTO PRODUS (DENUMIRE_PRODUS, PRET, STOC, ID_CATEGORIE)  
VALUES ('ESARFA COLORATA', 30.00, 25, 11);
```

```
SELECT * FROM PRODUS;
```

The screenshot shows the Oracle SQL Developer interface with the following details:

- Toolbar:** File, Edit, View, Navigate, Run, Source, Team, Tools, Window, Help.
- Connections:** Welcome Page, tutorial.
- Worksheet:** Query Builder
- Script Content:**

```
-- TABELA PRODUS
CREATE TABLE PRODUS (
    ID_PRODUS INT DEFAULT PRODUS_SEQ.NEXTVAL,
    DENUMIRE_PRODUS VARCHAR2(255),
    PRET NUMBER(10,2) NOT NULL,
    STOC NUMBER(10),
    ID_CATEGORIE INT,
    CONSTRAINT CHEIE_PRIMARA_PRODUS PRIMARY KEY (ID_PRODUS),
    CONSTRAINT FK_PRODUS_CATEGORIE FOREIGN KEY (ID_CATEGORIE)
        REFERENCES CATEGORIE(ID_CATEGORIE)
);

INSERT INTO PRODUS (DENUMIRE_PRODUS, PRET, STOC, ID_CATEGORIE) VALUES ('TRICOU ALBASTRU',55.50,10,1);
INSERT INTO PRODUS (DENUMIRE_PRODUS, PRET, STOC, ID_CATEGORIE) VALUES ('PANTALONI FORMALI',120.00,6,2);
INSERT INTO PRODUS (DENUMIRE_PRODUS, PRET, STOC, ID_CATEGORIE) VALUES ('CAMASA DE IN',110.70,20,6);
INSERT INTO PRODUS (DENUMIRE_PRODUS, PRET, STOC, ID_CATEGORIE) VALUES ('TRICOU GRU',49.99,25,1);
INSERT INTO PRODUS (DENUMIRE_PRODUS, PRET, STOC, ID_CATEGORIE) VALUES ('ROCHIE DE PLAJA',99.99,10,3);
INSERT INTO PRODUS (DENUMIRE_PRODUS, PRET, STOC, ID_CATEGORIE) VALUES ('BLUZĂ DE IARNĂ',52.50,5,8);
INSERT INTO PRODUS (DENUMIRE_PRODUS, PRET, STOC, ID_CATEGORIE) VALUES ('GEACA DE IARNĂ',199.99,25,4);
INSERT INTO PRODUS (DENUMIRE_PRODUS, PRET, STOC, ID_CATEGORIE) VALUES ('CAMASA VERDE',89.99,17,6);
INSERT INTO PRODUS (DENUMIRE_PRODUS, PRET, STOC, ID_CATEGORIE) VALUES ('GEANTA DE IARNA',125.00,20,7);
```
- Output:** Script Output (Task completed in 0.414 seconds), Query Result.
- Log:** Table PRODUS created.
1 row inserted.
1 row inserted.
- System:** 77°F, 10:42 AM, 6/7/2024.

The screenshot shows the Oracle SQL Developer interface with two tabs open: 'Script Output' and 'Query Result'.

Script Output:

```

INSERT INTO PRODUS (DENUMIRE_PRODUS, PRET, STOC, ID_CATEGORIE) VALUES ('ALEXANDRU', 55.5, 10, 1);
INSERT INTO PRODUS (DENUMIRE_PRODUS, PRET, STOC, ID_CATEGORIE) VALUES ('PANTALONI FORMALI', 120, 6, 2);
INSERT INTO PRODUS (DENUMIRE_PRODUS, PRET, STOC, ID_CATEGORIE) VALUES ('CAMASA DE IN', 110.7, 20, 6);
INSERT INTO PRODUS (DENUMIRE_PRODUS, PRET, STOC, ID_CATEGORIE) VALUES ('TRICOU GRI', 49.99, 25, 1);
INSERT INTO PRODUS (DENUMIRE_PRODUS, PRET, STOC, ID_CATEGORIE) VALUES ('ROCHIE DE PLAJA', 99.99, 10, 3);
INSERT INTO PRODUS (DENUMIRE_PRODUS, PRET, STOC, ID_CATEGORIE) VALUES ('OCHELARI DE SOARE COPIII', 52.5, 5, 8);
INSERT INTO PRODUS (DENUMIRE_PRODUS, PRET, STOC, ID_CATEGORIE) VALUES ('GEACA DE IARNA', 199.99, 35, 4);
INSERT INTO PRODUS (DENUMIRE_PRODUS, PRET, STOC, ID_CATEGORIE) VALUES ('CAMASA VERDE', 89.99, 17, 6);
INSERT INTO PRODUS (DENUMIRE_PRODUS, PRET, STOC, ID_CATEGORIE) VALUES ('GEANTA UMAR', 129.5, 20, 7);
INSERT INTO PRODUS (DENUMIRE_PRODUS, PRET, STOC, ID_CATEGORIE) VALUES ('TRICOU IN DUNGI', 49.99, 15, 1);
INSERT INTO PRODUS (DENUMIRE_PRODUS, PRET, STOC, ID_CATEGORIE) VALUES ('VESTA ALBASTRA', 100, 20, 5);
INSERT INTO PRODUS (DENUMIRE_PRODUS, PRET, STOC, ID_CATEGORIE) VALUES ('FUSTA MIDI', 75, 12, 10);
INSERT INTO PRODUS (DENUMIRE_PRODUS, PRET, STOC, ID_CATEGORIE) VALUES ('PULOVER DE LANA', 150, 8, 9);
INSERT INTO PRODUS (DENUMIRE_PRODUS, PRET, STOC, ID_CATEGORIE) VALUES ('ESARFA COLORATA', 30, 25, 11);
SELECT * FROM PRODUS;

```

Query Result:

ID_PRODUS	DENUMIRE_PRODUS	PRET	STOC	ID_CATEGORIE
1	1 TRICOU ALBASTRU	55.5	10	1
2	2 PANTALONI FORMALI	120	6	2
3	3 CAMASA DE IN	110.7	20	6
4	4 TRICOU GRI	49.99	25	1
5	5 ROCHIE DE PLAJA	99.99	10	3
6	6 OCHELARI DE SOARE COPIII	52.5	5	8
7	7 GEACA DE IARNA	199.99	35	4
8	8 CAMASA VERDE	89.99	17	6
9	9 GEANTA DE UMAR	129.5	20	7
10	10 TRICOU IN DUNGI	49.99	15	1
11	11 VESTA ALBASTRA	100	20	5
12	12 FUSTA MIDI	75	12	10
13	13 PULOVER DE LANA	150	8	9
14	14 ESARFA COLORATA	30	25	11

Produs_Comanda

CREATE TABLE PRODUS_COMANDA(

 ID_PRODUS INT,

 ID_COMANDA INT,

```
CONSTRAINT PK_PRODUS_COMANDA PRIMARY KEY (ID_PRODUS,  
ID_COMANDA),  
CONSTRAINT FK_PRODUS_COMANDA_PRODUS FOREIGN KEY (ID_PRODUS)  
REFERENCES PRODUS(ID_PRODUS),  
CONSTRAINT FK_PRODUS_COMANDA_COMANDA FOREIGN KEY  
(ID_COMANDA) REFERENCES COMANDA(ID_COMANDA)  
);
```

```
INSERT INTO PRODUS_COMANDA (ID_PRODUS, ID_COMANDA) VALUES (2, 1);  
INSERT INTO PRODUS_COMANDA (ID_PRODUS, ID_COMANDA) VALUES (11, 2);  
INSERT INTO PRODUS_COMANDA (ID_PRODUS, ID_COMANDA) VALUES (2, 3);  
INSERT INTO PRODUS_COMANDA (ID_PRODUS, ID_COMANDA) VALUES (3, 3);  
INSERT INTO PRODUS_COMANDA (ID_PRODUS, ID_COMANDA) VALUES (9, 4);  
INSERT INTO PRODUS_COMANDA (ID_PRODUS, ID_COMANDA) VALUES (1, 5);  
INSERT INTO PRODUS_COMANDA (ID_PRODUS, ID_COMANDA) VALUES (2, 6);  
INSERT INTO PRODUS_COMANDA (ID_PRODUS, ID_COMANDA) VALUES (3, 7);  
INSERT INTO PRODUS_COMANDA (ID_PRODUS, ID_COMANDA) VALUES (4, 8);  
INSERT INTO PRODUS_COMANDA (ID_PRODUS, ID_COMANDA) VALUES (6, 9);  
INSERT INTO PRODUS_COMANDA (ID_PRODUS, ID_COMANDA) VALUES (7, 10);  
INSERT INTO PRODUS_COMANDA (ID_PRODUS, ID_COMANDA) VALUES (5, 11);  
INSERT INTO PRODUS_COMANDA (ID_PRODUS, ID_COMANDA) VALUES (8, 12);  
INSERT INTO PRODUS_COMANDA (ID_PRODUS, ID_COMANDA) VALUES (3, 12);  
INSERT INTO PRODUS_COMANDA (ID_PRODUS, ID_COMANDA) VALUES (10, 12);  
INSERT INTO PRODUS_COMANDA (ID_PRODUS, ID_COMANDA) VALUES (9, 13);  
INSERT INTO PRODUS_COMANDA (ID_PRODUS, ID_COMANDA) VALUES (7, 13);  
SELECT * FROM PRODUS_COMANDA;
```

The screenshot shows the Oracle SQL Developer interface. The top menu bar includes File, Edit, View, Navigate, Run, Source, Team, Tools, Window, Help. The toolbar has icons for New, Open, Save, Run, Stop, Refresh, Undo, Redo, Copy, Paste, Find, Replace, and Insert. The Connections panel shows a connection named 'tutorial'. The main workspace is titled 'Query Builder' and contains the following code:

```
-- TABELA PRODUS_COMANDA
CREATE TABLE PRODUS_COMANDA(
    ID_PRODUS INT,
    ID_COMANDA INT,
    CONSTRAINT PK_PRODUS_COMANDA PRIMARY KEY (ID_PRODUS, ID_COMANDA),
    CONSTRAINT FK_PRODUS_COMANDA_PRODUS FOREIGN KEY (ID_PRODUS) REFERENCES PRODUS(ID_PRODUS),
    CONSTRAINT FK_PRODUS_COMANDA_COMANDA FOREIGN KEY (ID_COMANDA) REFERENCES COMANDA(ID_COMANDA)
);

INSERT INTO PRODUS_COMANDA (ID_PRODUS, ID_COMANDA) VALUES (2, 1);
INSERT INTO PRODUS_COMANDA (ID_PRODUS, ID_COMANDA) VALUES (11, 21);
INSERT INTO PRODUS_COMANDA (ID_PRODUS, ID_COMANDA) VALUES (2, 3);
INSERT INTO PRODUS_COMANDA (ID_PRODUS, ID_COMANDA) VALUES (3, 3);
INSERT INTO PRODUS_COMANDA (ID_PRODUS, ID_COMANDA) VALUES (9, 4);
INSERT INTO PRODUS_COMANDA (ID_PRODUS, ID_COMANDA) VALUES (1, 5);
INSERT INTO PRODUS_COMANDA (ID_PRODUS, ID_COMANDA) VALUES (1, 6);
INSERT INTO PRODUS_COMANDA (ID_PRODUS, ID_COMANDA) VALUES (7, 7);
INSERT INTO PRODUS_COMANDA (ID_PRODUS, ID_COMANDA) VALUES (3, 7);
INSERT INTO PRODUS_COMANDA (ID_PRODUS, ID_COMANDA) VALUES (4, 8);
INSERT INTO PRODUS_COMANDA (ID_PRODUS, ID_COMANDA) VALUES (6, 9);
INSERT INTO PRODUS_COMANDA (ID_PRODUS, ID_COMANDA) VALUES (7, 10);
INSERT INTO PRODUS_COMANDA (ID_PRODUS, ID_COMANDA) VALUES (8, 11);
INSERT INTO PRODUS_COMANDA (ID_PRODUS, ID_COMANDA) VALUES (8, 12);
INSERT INTO PRODUS_COMANDA (ID_PRODUS, ID_COMANDA) VALUES (10, 12);
INSERT INTO PRODUS_COMANDA (ID_PRODUS, ID_COMANDA) VALUES (10, 13);
INSERT INTO PRODUS_COMANDA (ID_PRODUS, ID_COMANDA) VALUES (9, 13);
SELECT * FROM PRODUS_COMANDA;
```

The 'Script Output' tab shows the results of the execution:

```
Table PRODUS_COMANDA created.

1 row inserted.

1 row inserted.

1 row inserted.

1 row inserted.

1 row inserted.
```

The status bar at the bottom right indicates 'Line 526 Column 1 | Insert | Modified | Windows | 10:45 AM 6/7/2024'.

Script Output | Query Result | All Rows Fetched: 17 in 0.045 seconds

ID_PRODUS	ID_COMANDA	
1	1	5
2	2	1
3	2	3
4	2	6
5	3	3
6	3	7
7	3	12
8	4	8
9	5	11
10	6	9
11	7	10
12	7	13
13	8	12
14	9	4
15	9	13
16	10	12
17	11	2

Produs_Cos

```
CREATE TABLE PRODUS_COS(
    ID_PRODUS INT,
    ID_COS INT,
    CONSTRAINT PK_PRODUS_COS PRIMARY KEY (ID_PRODUS, ID_COS),
    CONSTRAINT FK_PRODUS_COS_PRODUS FOREIGN KEY (ID_PRODUS)
        REFERENCES PRODUS(ID_PRODUS),
    CONSTRAINT FK_PRODUS_COS_COS FOREIGN KEY (ID_COS) REFERENCES
        COS_CUMPARATURI(ID_COS)
);
```

```
INSERT INTO PRODUS_COS VALUES(1,1);
INSERT INTO PRODUS_COS (ID_PRODUS, ID_COS) VALUES(13,2);
INSERT INTO PRODUS_COS (ID_PRODUS, ID_COS) VALUES(2,3);
INSERT INTO PRODUS_COS (ID_PRODUS, ID_COS) VALUES(11,3);
INSERT INTO PRODUS_COS (ID_PRODUS, ID_COS) VALUES(7,4);
INSERT INTO PRODUS_COS (ID_PRODUS, ID_COS) VALUES(4,5);
INSERT INTO PRODUS_COS (ID_PRODUS, ID_COS) VALUES(5,5);
INSERT INTO PRODUS_COS (ID_PRODUS, ID_COS) VALUES(6,5);
INSERT INTO PRODUS_COS (ID_PRODUS, ID_COS) VALUES(9,7);
INSERT INTO PRODUS_COS (ID_PRODUS, ID_COS) VALUES(3,10);
INSERT INTO PRODUS_COS (ID_PRODUS, ID_COS) VALUES(8,10);
SELECT * FROM PRODUS_COS;
```

Produs_Lista

CREATE TABLE PRODUS_LISTA(

```

ID_PRODUS INT,
ID_LISTA INT,
CONSTRAINT PK_PRODUS_LISTA PRIMARY KEY (ID_PRODUS, ID_LISTA),
CONSTRAINT FK_PRODUS_LISTA_PRODUS FOREIGN KEY (ID_PRODUS)
REFERENCES PRODUS(ID_PRODUS),
CONSTRAINT FK_PRODUS_LISTA_LISTA FOREIGN KEY (ID_LISTA)
REFERENCES LISTA_DORINTE(ID_LISTA)
);

```

```

INSERT INTO PRODUS_LISTA (ID_PRODUS, ID_LISTA) VALUES (1, 1);
INSERT INTO PRODUS_LISTA (ID_PRODUS, ID_LISTA) VALUES (2, 1);
INSERT INTO PRODUS_LISTA (ID_PRODUS, ID_LISTA) VALUES (3, 2);
INSERT INTO PRODUS_LISTA (ID_PRODUS, ID_LISTA) VALUES (4, 3);
INSERT INTO PRODUS_LISTA (ID_PRODUS, ID_LISTA) VALUES (5, 4);
INSERT INTO PRODUS_LISTA (ID_PRODUS, ID_LISTA) VALUES (6, 4);
INSERT INTO PRODUS_LISTA (ID_PRODUS, ID_LISTA) VALUES (7, 5);
INSERT INTO PRODUS_LISTA (ID_PRODUS, ID_LISTA) VALUES (8, 6);
INSERT INTO PRODUS_LISTA (ID_PRODUS, ID_LISTA) VALUES (9, 6);
INSERT INTO PRODUS_LISTA (ID_PRODUS, ID_LISTA) VALUES (10, 7);
INSERT INTO PRODUS_LISTA (ID_PRODUS, ID_LISTA) VALUES (11, 7);
INSERT INTO PRODUS_LISTA (ID_PRODUS, ID_LISTA) VALUES (12, 9);
INSERT INTO PRODUS_LISTA (ID_PRODUS, ID_LISTA) VALUES (13, 10);
INSERT INTO PRODUS_LISTA (ID_PRODUS, ID_LISTA) VALUES (1, 11);
INSERT INTO PRODUS_LISTA (ID_PRODUS, ID_LISTA) VALUES (2, 12);
INSERT INTO PRODUS_LISTA (ID_PRODUS, ID_LISTA) VALUES (3, 13);
INSERT INTO PRODUS_LISTA (ID_PRODUS, ID_LISTA) VALUES (4, 14);
INSERT INTO PRODUS_LISTA (ID_PRODUS, ID_LISTA) VALUES (5, 14);
SELECT * FROM PRODUS_LISTA;

```


Script Output | Query Result | SQL | All Rows Fetched: 18 in 0.053 seconds

	ID_PRODUS	ID_LISTA
1	1	1
2	1	11
3	2	1
4	2	12
5	3	2
6	3	13
7	4	3
8	4	14
9	5	4
10	5	14
11	6	4
12	7	5
13	8	6
14	9	6
15	10	7
16	11	7
17	12	9
18	13	10

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

- a) subcereri sincronizate în care intervin cel puțin 3 tabele
- b) subcereri nesincronizate în clauza FROM
- c) grupări de date, funcții grup, filtrare la nivel de grupuri cu subcereri nesincronizate (în clauza de HAVING) în care intervin cel puțin 3 tabele (în cadrul aceleiași cereri)
- d) ordonări și utilizarea funcțiilor NVL și DECODE (în cadrul aceleiași cereri)
- e) utilizarea a cel puțin 2 funcții pe siruri de caractere, 2 funcții pe date calendaristice, a cel puțin unei expresii CASE
- f) utilizarea a cel puțin 1 bloc de cerere (clauza WITH)

Observație: Într-o cerere se vor regăsi mai multe elemente dintre cele enumerate mai sus, astfel încât cele 5 cereri să le cuprindă pe toate.

Cerere 1

Afisati numele si prenumele utilizatorilor care au plasat cel putin 2 comenzi și au au numărul de comenzi cu livrările efectuate sau în curs de livrare mai mare decât media numărului de livrări plasate de ceilalți clienți. În plus, afișați totalul plășilor, câte carduri sunt înregistrate sub numele clientului, daca a avut o plată respinsă. Rezultatele sunt ordonate descrescător în funcție de numărul de livrări.

```
SELECT CONCAT(CONCAT(U.NUME, ' '), U.PRENUME) AS NUME,
       COUNT(DISTINCT C.ID_COMANDA) AS NUMAR_COMENZI,
       NVL(SUM(C.TOTAL_PLATA), 0) AS TOTAL_COMENZI,
       (CASE WHEN (SELECT COUNT(P1.ID_PLATA)
                  FROM PLATA P1
                 JOIN COMANDA C1 ON C1.ID_COMANDA = P1.ID_COMANDA
                WHERE C1.ID_UTILIZATOR = U.ID_UTILIZATOR AND UPPER(P1.STARE) = 'RESPINSA') > 0
            THEN 'DA'
            ELSE 'NU' END) AS PLATI_RESPINSE,
       COUNT(CAR.ID_CARD) AS NUMAR_CARDURI
  FROM UTILIZATOR U
  JOIN COMANDA C ON C.ID_UTILIZATOR = U.ID_UTILIZATOR
  JOIN CARD CAR ON CAR.ID_UTILIZATOR = U.ID_UTILIZATOR
 GROUP BY U.ID_UTILIZATOR, U.NUME, U.PRENUME
 HAVING COUNT(DISTINCT C.ID_COMANDA) >= 2 and (
  SELECT COUNT(C2.ID_COMANDA)
  FROM COMANDA C2
  JOIN PLATA P2 ON P2.ID_COMANDA = C2.ID_COMANDA
  WHERE (UPPER(P2.STARE) = 'EFECTUATA' OR UPPER(P2.STARE) = 'IN CURS DE LIVRARE') AND C2.ID_UTILIZATOR = U.ID_UTILIZATOR) >=
  (SELECT AVG(NUMAR_LIVRARI)
   FROM (SELECT COUNT(C3.ID_COMANDA) AS NUMAR_LIVRARI
         FROM UTILIZATOR U3
        JOIN COMANDA C3 ON U3.ID_UTILIZATOR = C3.ID_UTILIZATOR)
```

```

        GROUP BY U3.ID_UTILIZATOR)))
ORDER BY NUMAR_COMENZI DESC;

--subcereri nesincronizate în clauza FROM
--grupări de date, funcții grup, filtrare la nivel de grupuri cu subcereri nesincronizate
--(în clauza de HAVING) în care intervin cel puțin 3 tabele (în cadrul aceleiași cereri)
--ordonări și utilizarea funcțiilor NVL și DECODE (în cadrul aceleiași cereri)
--concat, upper
--expresie case

```

The screenshot shows the Oracle SQL Developer interface. The top part displays the SQL query in the 'Worksheet' tab:

```

-- Cerere 1
-- Afisati numele si prenumele utilizatorilor care au plasat cel putin 2 comenzi
-- si au au numarul de comenzi cu livrari efectuate sau in curs de livrare mai
-- mare decat media numarului de livrari plasate de ceilalti clienti.
-- In plus, afisati totalul platilor, cate carduri sunt inregistrate sub numele
-- clientului, daca a avut o plata respinsa.
-- Rezultatele sunt ordonate descrescator in functie de numarul de livrari.

SELECT CONCAT(U.NUME, ' ', U.PRENUME) AS NUME,
       COUNT(DISTINCT C.ID_COMANDA) AS NUMAR_COMENZI,
       NVL(SUM(C.TOTAL_PLATA), 0) AS TOTAL_COMENZI,
       (CASE WHEN (SELECT COUNT(P1.ID_PLATA)
                  FROM PLATA P1
                 JOIN COMANDA C1 ON C1.ID_COMANDA = P1.ID_COMANDA
                WHERE C1.ID_UTILIZATOR = U.ID_UTILIZATOR AND UPPER(P1.STARE) = 'RESPINSA') > 0
            THEN 'DA'
            ELSE 'NU' END) AS PLATI_RESPINSE,
       COUNT(CAR.ID_CARD) AS NUMAR_CARDURI
  FROM UTILIZATOR U
 JOIN COMANDA C ON C.ID_UTILIZATOR = U.ID_UTILIZATOR
 JOIN CARD CAR ON CAR.ID_UTILIZATOR = U.ID_UTILIZATOR
 GROUP BY U.ID_UTILIZATOR, U.NUME, U.PRENUME
 HAVING COUNT(DISTINCT C.ID_COMANDA)>=2 and (
      SELECT COUNT(C2.ID_COMANDA)
         FROM COMANDA C2
        JOIN PLATA P2 ON P2.ID_COMANDA = C2.ID_COMANDA
       WHERE (UPPER(P2.STARE) = 'EFECTUATA' OR UPPER(P2.STARE) = 'IN CURS DE LIVRARE') AND C2.ID_UTILIZATOR = U.ID_UTILIZATOR)>=
      (SELECT AVG(NUMAR_LIVRARI)
         FROM (SELECT COUNT(C3.ID_COMANDA) AS NUMAR_LIVRARI
                  FROM UTILIZATOR U3
                 JOIN COMANDA C3 ON U3.ID_UTILIZATOR = C3.ID_UTILIZATOR
                GROUP BY U3.ID_UTILIZATOR))
      ORDER BY NUMAR_COMENZI DESC;

--subcereri nesincronizate in clauza FROM
--grupări de date, funcții grup, filtrare la nivel de grupuri cu subcereri nesincronizate
--(în clauza de HAVING) în care intervin cel puțin 3 tabele (în cadrul aceleiasi cereri)
--ordonări și utilizarea funcțiilor NVL și DECODE (în cadrul aceleiasi cereri)
--concat, upper
--expresie case

```

The bottom part shows the 'Script Output' tab with the results of the query:

NUME	NUMAR_COMENZI	TOTAL_COMENZI	PLATI_RESPINSE	NUMAR_CARDURI
1 Marin Iuliana	2	192 NU		2

Cerere 2

Pentru toți utilizatorii cu cel puțin un produs în coșul de cumpărături să se afișeze cel mai scump produs achiziționat în ultimele 12 luni (numele și prețul), dacă nu au cumpărat nimic se va afișa "fără achizitii" pentru denumirea produsului și 0 la preț.

```
WITH UTILIZATORI_COS AS (
    SELECT U.ID_UTILIZATOR, U.NUME, U.PRENUME
    FROM UTILIZATOR U
    JOIN COS_CUMPARATURI CC ON CC.ID_UTILIZATOR = U.ID_UTILIZATOR
    JOIN PRODUS_COS PC ON PC.ID_COS = CC.ID_COS
    GROUP BY U.ID_UTILIZATOR, U.NUME, U.PRENUME
    HAVING COUNT(PC.ID_PRODUS) > 0),
    PRODUS_MAX AS (SELECT C1.ID_UTILIZATOR, P1.DENUMIRE_PRODUS, P1.PRET
    FROM COMANDA C1
    JOIN PRODUS_COMANDA PC1 ON C1.ID_COMANDA = PC1.ID_COMANDA
    JOIN PRODUS P1 ON PC1.ID_PRODUS = P1.ID_PRODUS
    WHERE C1.DATA_COMANDA >= ADD_MONTHS(SYSDATE, -12)
    AND P1.PRET IN (SELECT MAX(P2.PRET)
    FROM COMANDA C2
    JOIN PRODUS_COMANDA PC2 ON C2.ID_COMANDA = PC2.ID_COMANDA
    JOIN PRODUS P2 ON PC2.ID_PRODUS = P2.ID_PRODUS
    WHERE C2.DATA_COMANDA >= ADD_MONTHS(SYSDATE, -12)
    AND C2.ID_UTILIZATOR = C1.ID_UTILIZATOR)
)
SELECT UC.ID_UTILIZATOR,
    CONCAT(CONCAT(UC.NUME, ' '), UC.PRENUME) AS NUME,
    NVL(PM.DENUMIRE_PRODUS, 'fără achizitii') AS DENUMIRE_PRODUS,
```

```

    NVL(PM.PRET, 0) AS PRET
FROM UTILIZATORI_COS UC
LEFT JOIN PRODUS_MAX PM ON PM.ID_UTILIZATOR = UC.ID_UTILIZATOR
ORDER BY ID_UTILIZATOR;

```

-- 2 blocuri de cerere with

-- sysdate, add_months

-- subcereri sincronizate

```

-- Oracle SQL Developer : tutorial
File Edit View Navigate Run Source Team Window Help
Connections tutorial
Worksheet Query Builder
-- Cerere 2
--Pentru toți utilizatorii cu cel puțin un produs în coșul de cumpărături
--sa se afișeze cel mai scump produs achiziționat în ultimele 12 luni (numele și prețul),
--daca nu au cumpărat nimic se va afișa "fără achiziții" pentru denumirea produsului și 0 la preț.

WITH UTILIZATORI_COS AS (
    SELECT U.ID_UTILIZATOR, U.NUME, U.PRENUME
    FROM UTILIZATOR U
    JOIN COS_CUMPARATURI CC ON CC.ID_UTILIZATOR = U.ID_UTILIZATOR
    JOIN PRODUS_COS PC ON PC.ID_COS = CC.ID_COS
    GROUP BY U.ID_UTILIZATOR, U.NUME, U.PRENUME
    HAVING COUNT(PC.ID_PRODUS) > 0),
PRODUS_MAX AS ( SELECT C1.ID_UTILIZATOR, P1.DENUMIRE_PRODUS, P1.PRET
    FROM COMANDA C1
    JOIN PRODUS_COMANDA PC1 ON C1.ID_COMANDA = PC1.ID_COMANDA
    JOIN PRODUS P1 ON PC1.ID_PRODUS = P1.ID_PRODUS
    WHERE C1.DATA_COMANDA >= ADD_MONTHS(SYSDATE, -12)
    AND P1.PRET IN (SELECT MAX(P2.PRET)
        FROM COMANDA C2
        JOIN PRODUS_COMANDA PC2 ON C2.ID_COMANDA = PC2.ID_COMANDA
        JOIN PRODUS P2 ON PC2.ID_PRODUS = P2.ID_PRODUS
        WHERE C2.DATA_COMANDA >= ADD_MONTHS(SYSDATE, -12)
        AND C2.ID_UTILIZATOR = C1.ID_UTILIZATOR)
)
SELECT UC.ID_UTILIZATOR,
    CONCAT(UC.NUME, ' ', UC.PRENUME) AS NUME,
    NVL(PM.DENUMIRE_PRODUS, 'fără achiziții') AS DENUMIRE_PRODUS,
    NVL(PM.PRET, 0) AS PRET
FROM UTILIZATORI_COS UC
LEFT JOIN PRODUS_MAX PM ON PM.ID_UTILIZATOR = UC.ID_UTILIZATOR
ORDER BY ID_UTILIZATOR;

-- 2 blocuri de cerere with
-- sysdate, add_months
-- subcereri sincronizate

```

ID_UTILIZATOR	NUME	DENUMIRE_PRODUS	PRET
1	Maciuca Emma	fără achizitii	0
2	Popescu Andreea-Ioana	PANTALONI FORMALI	120
3	Ion Andrei	CAMASA DE IN	110.7
4	Andreescu Maria	TRICOU GRI	49.99
5	Maciuca Sara	PANTALONI FORMALI	120
6	Pop Daria	GEACA DE IARNA	199.99
7	Balan Ioana	fără achizitii	0

Cerere 3

Pentru fiecare comanda a carui curier a efectuat cel putin o livrare in Bucuresti si are salariul mai mare decat salariul mediu al tuturor curierilor, selectati numele curierului cu care s-a efectuat comanda, masina, orasul si mesajul "comanda recenta" daca au trecut mai putin de 3 luni, respectiv "comanda trecuta" altfel.

```

SELECT CO.ID_COMANDA, --subcerere sincronizata
       (SELECT CONCAT(CONCAT(C.NUME, ' '), C.PRENUME)
        FROM CURIER C
        WHERE C.ID_CURIER = L.ID_CURIER) AS CURIER,
       (SELECT M.MODEL_MASINA
        FROM MASINA_LIVRARE M
        WHERE M.ID_MASINA = L.ID_MASINA) AS MASINA,
       (SELECT A.ORAS
        FROM ADRESA A
        WHERE A.ID_ADRESA = L.ID_ADRESA) AS ORAS,
       DECODE(SIGN(NVL(MONTHS_BETWEEN(SYSDATE, L.DATA_LIVRARE), 0)), -1, 'LIVRARE VIITOARE', 'LIVRARE EFECTUATA') AS LIVRARE --decode si nvl, sysdate si monthsbetween
      FROM COMANDA CO
      JOIN LIVRARE L ON CO.ID_LIVRARE = L.ID_LIVRARE
      where L.ID_CURIER IN (SELECT L2.ID_CURIER
                            FROM LIVRARE L2

```

```

JOIN ADRESA A ON L2.ID_ADRESA = A.ID_ADRESA
JOIN CURIER C2 ON C2.ID_CURIER = L2.ID_CURIER
WHERE UPPER(A.ORAS) = 'BUCURESTI' AND C2.SALARIU >=
(SELECT AVG(C3.SALARIU)
FROM CURIER C3))
ORDER BY CO.ID_COMANDA;
--ordonări și utilizarea funcțiilor NVL și DECODE (în cadrul aceleiași cereri)
--subcereri sincronizate în care intervin cel puțin 3 tabele
--SYSDATE, MONTHS_BETWEEN

```

The screenshot shows the Oracle SQL Developer interface. The top menu bar includes File, Edit, View, Navigate, Run, Source, Team, Tools, Window, and Help. Below the menu is a toolbar with various icons. The main area has tabs for Connections (set to tutorial), Worksheet, and Query Builder. The Worksheet tab contains the SQL query code. The Query Result tab shows the results of the executed query.

```

-- Cerere 3
-- Pentru fiecare comanda a carul curier a efectuat cel putin o livrare in Bucuresti si are salariul mai mare decat salariul mediu al tuturor curierilor,
-- selectati numele curierului cu care s-a efectuat comanda, masina, orasul si mesajul "comanda recenta" daca au trecut mai putin de 3 luni, respectiv "comanda trecuta" altfel
--SELECT CO.ID_COMANDA, --subcerere sincronizata
--       (SELECT CONCAT(CONCAT(C.NUME, ' '), C.PRENUME)
--        FROM CURIER C
--        WHERE C.ID_CURIER = L.ID_CURIER) AS CURIER,
--       (SELECT M.MODEL_MASINA
--        FROM MASINA_LIVRARE M
--        WHERE M.ID_MASINA = L.ID_MASINA) AS MASINA,
--       (SELECT A.ORAS
--        FROM ADRESA A
--        WHERE A.ID_ADRESA = L.ID_ADRESA) AS ORAS,
--       DECODE(SIGN(NVL(MONTHS_BETWEEN(SYSDATE, L.DATA_LIVRARE), 0)), -1, 'LIVRARE VIITOARE', 'LIVRARE EFECTUATA') AS LIVRARE --decode si nvl, sysdate si monthsbetween
--FROM COMANDA CO
--JOIN LIVRARE L ON CO.ID_LIVRARE = L.ID_LIVRARE
--@where L.ID_CURIER IN (SELECT L2.ID_CURIER
--                        FROM LIVRARE L2
--                        JOIN ADRESA A ON L2.ID_ADRESA = A.ID_ADRESA
--                        JOIN CURIER C2 ON C2.ID_CURIER = L2.ID_CURIER
--                        WHERE UPPER(A.ORAS) = 'BUCURESTI' AND C2.SALARIU >= (SELECT AVG(C3.SALARIU)
--                                                               FROM CURIER C3))
--ORDER BY CO.ID_COMANDA;
--ordonări și utilizarea funcțiilor NVL și DECODE (în cadrul aceleiași cereri)
--subcereri sincronizate in care intervin cel putin 3 tabele
--SYSDATE, MONTHS_BETWEEN

```

The screenshot shows the results of the executed query in the Oracle SQL Developer interface. The results are displayed in a grid format with columns: ID_COMANDA, CURIER, MASINA, ORAS, and LIVRARE. There are 6 rows of data.

ID_COMANDA	CURIER	MASINA	ORAS	LIVRARE
1	4 STEFANESCU VLAD	Citroën Jumper	BUCURESTI	LIVRARE EFECTUATA
2	5 CONSTANTINESCU MARIUS	Opel Movano	BUCURESTI	LIVRARE EFECTUATA
3	6 MARINESCU ANDREI	Volkswagen Crafter	CONSTANTA	LIVRARE EFECTUATA
4	9 ALEXANDRU RADU	Hyundai H350	BUCURESTI	LIVRARE EFECTUATA
5	10 RADU FLORIN	Ford Transit	BUCURESTI	LIVRARE EFECTUATA
6	13 MARINESCU ANDREI	Fiat Ducato	BUCURESTI	LIVRARE EFECTUATA

Cerere 4

Selectați toti utilizatorii născuți înainte de anul 2000, care conțin cel puțin două litere de 'A' în email, au plasat cel puțin o comandă în anul 2024, iar media comenzielor

personale este mai mare sau egal decât media tuturor comenzi din magazin din anul respectiv. Se va afișa numele complet al utilizatorului, cât și numărul de comenzi și prețul mediu al comenzi. Rezultatele vor fi afișate în ordine descrescătoare în funcție de prețul mediu al comenzi.

```
WITH A_UTILIZATORI AS( --BLOC DE CERERE CU WITH
SELECT DISTINCT U.ID_UTILIZATOR,
               CONCAT(CONCAT(U.NUME, ' '), U.PRENUME) AS NUME_INTREG,
               AVG(P.VALOARE) AS COMANDA_MEDIE,
               COUNT(C.ID_COMANDA) AS NUMAR_COMENZI
FROM COMANDA C
JOIN UTILIZATOR U ON C.ID_UTILIZATOR = U.ID_UTILIZATOR
JOIN PLATA P ON P.ID_COMANDA = C.ID_COMANDA
WHERE UPPER(U.EMAIL) LIKE '%A%A%' AND
      UPPER(P.STARE) = 'EFECTUATA' AND
      UPPER(C.STARE_COMANDA) = 'EFECTUATA' AND
      TO_NUMBER(TO_CHAR(U.DATA_NASTERE,'YYYY')) < 2000 AND
      TO_NUMBER(TO_CHAR(C.DATA_COMANDA,'YYYY')) = 2024
GROUP BY U.ID_UTILIZATOR, U.NUME, U.PRENUME)
SELECT *
FROM A_UTILIZATORI A
WHERE A.COMANDA_MEDIE >= (SELECT AVG(NVL(C1.TOTAL_PLATA,0))
                           FROM COMANDA C1
                           WHERE TO_NUMBER(TO_CHAR(C1.DATA_COMANDA,'YYYY')) =
2024)
      AND A.NUMAR_COMENZI >= 1
ORDER BY A.COMANDA_MEDIE;
```

--1 bloc de cerere cu clauza with

--2 funcții pe șiruri de caractere (concat, to_upper) și 1 pe data calendaristică (to_char)

--Grupari de date in care intervin 3 tabele

--Subcerere nesincronizata

The screenshot shows the Oracle SQL Developer interface with the 'tutorial' connection selected. The 'Worksheet' tab is active, displaying the following SQL query:

```
-- Cerere 4
--Selectati toti utilizatorii nascuti inainte de anul 2000, care contin cel putin doua litere de 'A' in email,
--au plasat cel putin o comanda in anul 2024, iar media comenzi personale este mai mare sau egal decat media tuturor comenzi din magazin din anul respectiv.
--Se va afisa numele complet al utilizatorului, cat si numarul de comenzi si pretul mediu al comenzi.
--Rezultatele vor fi afisate in ordine descrescatoare in functie de pretul mediu al comenzi.

WITH A_UTILIZATORI AS(
    SELECT DISTINCT U.ID_UTILIZATOR,
        CONCAT(U.NUME, ' ', U.PRENUME) AS NUME_INTREG,
        AVG(P.VALORARE) AS COMANDA_MEDIE,
        COUNT(C.ID_COMANDA) AS NUMAR_COMENZI
    FROM COMANDA C
    JOIN UTILIZATOR U ON C.ID_UTILIZATOR = U.ID_UTILIZATOR
    JOIN PLATA P ON P.ID_COMANDA = C.ID_COMANDA
    WHERE UPPER(U.EMAIL) LIKE 'AAAA'
        AND UPPER(P.STARE) = 'EFECTUATA' AND
        UPPER(C.STARE_COMANDA) = 'EFECTUATA' AND
        TO_NUMBER(TO_CHAR(U.DATA_NASTERE, 'YYYY')) < 2000 AND
        TO_NUMBER(TO_CHAR(C.DATA_COMANDA, 'YYYY')) = 2024
    GROUP BY U.ID_UTILIZATOR, U.NUME, U.PRENUME)
SELECT *
FROM A_UTILIZATORI A
WHERE A.COMANDA_MEDIE >= (SELECT AVG(NVL(C1.TOTAL_PLATA, 0))
    FROM COMANDA C1
    WHERE TO_NUMBER(TO_CHAR(C1.DATA_COMANDA, 'YYYY')) = 2024)
        AND A.NUMAR_COMENZI = 1
ORDER BY A.COMANDA_MEDIE;

--1 bloc de cerere cu clauza with
--2 functi pe siruri de caractere (concat, to_upper) si 1 pe data calendaristica (to_char)
--Grupe de date in care intervin 3 tabele
--Subcerere nesincronizata
```

The screenshot shows the 'Query Result' tab in Oracle SQL Developer displaying the results of the executed query. The results are as follows:

ID_UTILIZATOR	NUME_INTREG	COMANDA_MEDIE	NUMAR_COMENZI
1	5 Maciuca Sara	237.6	1

Cerere 5

Selectati primele 3 produse comandate de cele mai multe ori si afisati denumirea, pretul, o descriere a stocului si categoria din care fac parte.

SELECT *

```

FROM (SELECT
    P.DENUMIRE_PRODUS,
    P.PRET,
    (CASE WHEN P.STOC < 5 THEN 'STOC REDUS' --case
        WHEN P.STOC > 5 THEN 'STOC RIDICATEND) AS STOC,
    NR_COMENZI,
    cat.denumire_categorie
FROM PRODUS P
JOIN (SELECT ID_PRODUS, COUNT(ID_COMANDA) AS NR_COMENZI --
SUBCERERE IN FROM
    FROM PRODUS_COMANDA
    GROUP BY ID_PRODUS) PC ON PC.ID_PRODUS = P.ID_PRODUS
JOIN CATEGORIE CAT ON CAT.ID_CATEGORIE = P.ID_CATEGORIE
ORDER BY NR_COMENZI DESC
) PRODUSE_POPULARE
WHERE ROWNUM <= 3;

-- subcerere nesincronizata in clauza from
-- expresia case

```

The screenshot shows the Oracle SQL Developer interface. In the Worksheet tab, there is a SQL query:

```

--Cerere 5
-- selectati primele 3 produse comandate de cele mai multe ori si afisati denumirea,
-- pretul, o descriere a stocului si categoria din care fac parte
SELECT *
FROM (SELECT
    P.DENUMIRE_PRODUS,
    P.PRET,
    (CASE WHEN P.STOC < 5 THEN 'STOC REDUS' --case
          WHEN P.STOC > 5 THEN 'STOC RIDICAT') AS STOC,
    NR_COMENZI,
    cat.denumire_categoria
  FROM PRODUS P
  JOIN (SELECT ID_PRODUS, COUNT(ID_COMANDA) AS NR_COMENZI --SUBCERERE IN FROM
        FROM PRODUS_COMANDA
        GROUP BY ID_PRODUS) PC ON PC.ID_PRODUS = P.ID_PRODUS
  JOIN CATEGORIE CAT ON CAT.ID_CATEGORIE = P.ID_CATEGORIE
  ORDER BY NR_COMENZI DESC
) PRODUSE_POPULARE
WHERE ROWNUM <= 3;

-- subcerere nesincronizata in clauza from
-- expresia case

```

In the Query Result tab, the output is:

DENUMIRE_PRODUS	PRET	STOC	NR_COMENZI	DENUMIRE_CATEGORIE
1 PANTALONI FORMALI	120	STOC RIDICAT	3	PANTALONI
2 CAMASA DE IN	110.7	STOC RIDICAT	3	CAMASA
3 GEACA DE IARNA	199.99	STOC RIDICAT	2	GEACA

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

Operatie 1

Setarea stării comenziilor în efectuată unde starea livrării este efectuată.

UPDATE COMANDA

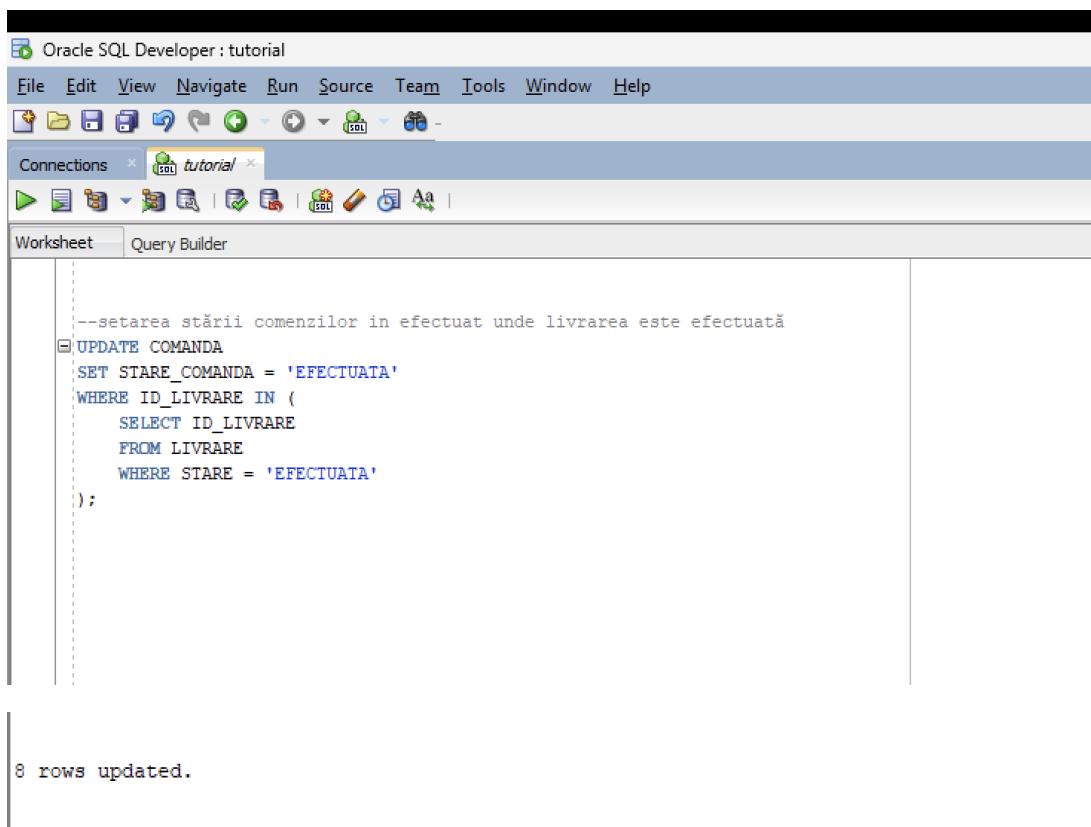
```
SET STARE_COMANDA = 'EFECTUATA'
```

```
WHERE ID_LIVRARE IN (
```

```
    SELECT ID_LIVRARE
```

```
    FROM LIVRARE
```

```
WHERE STARE = 'EFECTUATA'  
);
```



The screenshot shows the Oracle SQL Developer interface. The title bar reads "Oracle SQL Developer : tutorial". The menu bar includes File, Edit, View, Navigate, Run, Source, Team, Tools, Window, and Help. The toolbar has various icons for connection management and code editing. The Connections tab shows a single connection named "tutorial". The Worksheet tab is active, displaying the following SQL code:

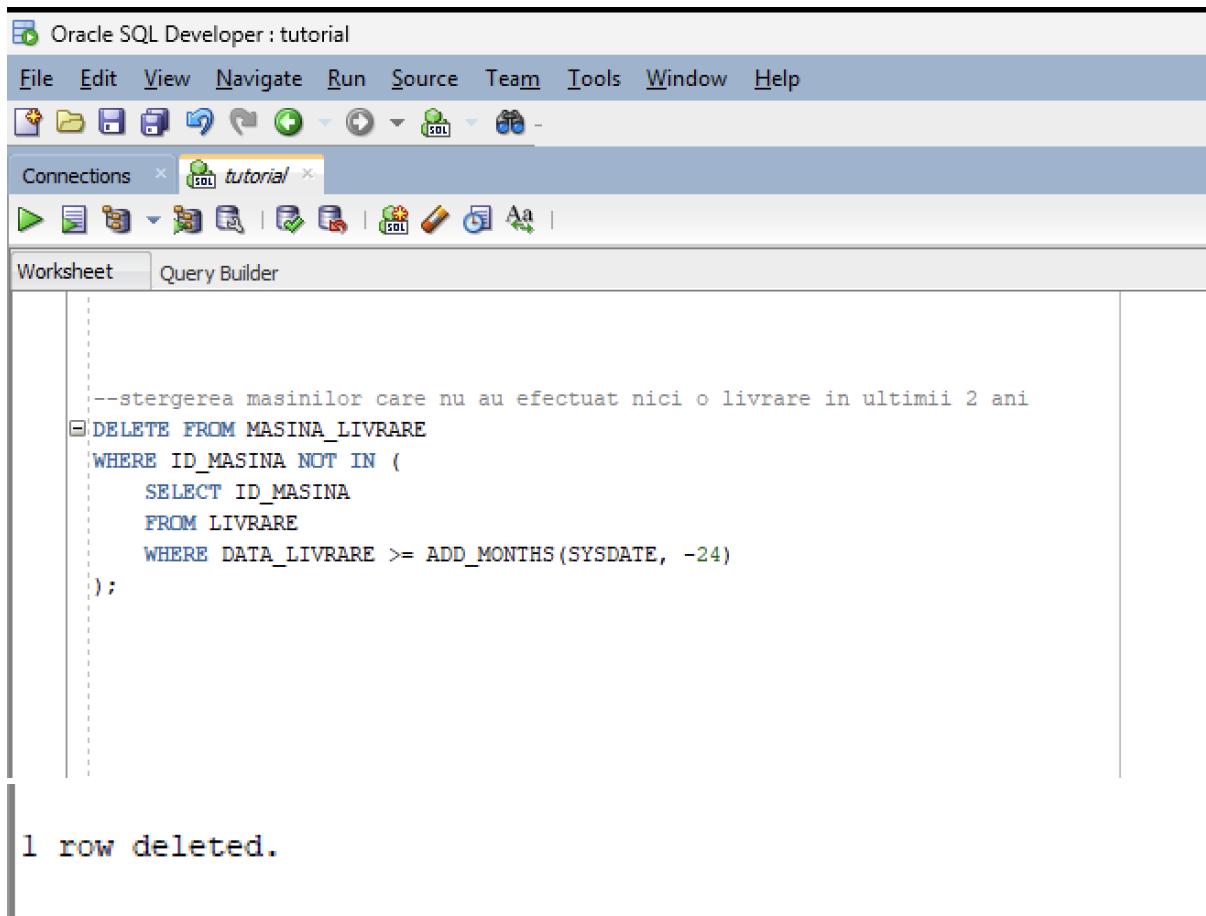
```
--setarea stării comenzilor în efectuat unde livrarea este efectuată  
UPDATE COMANDA  
SET STARE_COMANDA = 'EFECTUATA'  
WHERE ID_LIVRARE IN (  
    SELECT ID_LIVRARE  
    FROM LIVRARE  
    WHERE STARE = 'EFECTUATA'  
);
```

Below the code, the message "8 rows updated." is displayed.

Operatie 2

Stergerea masinilor care nu au efectuat nici o livrare in ultimii 2 ani.

```
DELETE FROM MASINA_LIVRARE  
WHERE ID_MASINA NOT IN (  
    SELECT ID_MASINA  
    FROM LIVRARE  
    WHERE DATA_LIVRARE >= ADD_MONTHS(SYSDATE, -24)  
);
```



The screenshot shows the Oracle SQL Developer interface. The title bar reads "Oracle SQL Developer : tutorial". The menu bar includes File, Edit, View, Navigate, Run, Source, Team, Tools, Window, and Help. The toolbar has various icons for connection management and SQL operations. A Connections panel shows a connection named "tutorial". The main area has two tabs: "Worksheet" (selected) and "Query Builder". The Worksheet tab contains the following SQL code:

```
--stergerea masinilor care nu au efectuat nici o livrare in ultimii 2 ani
DELETE FROM MASINA_LIVRARE
WHERE ID_MASINA NOT IN (
    SELECT ID_MASINA
    FROM LIVRARE
    WHERE DATA_LIVRARE >= ADD_MONTHS(SYSDATE, -24)
);

```

Below the code, the output "1 row deleted." is displayed.

Operatie 3

Aplicarea unei reduceri de 10% la cosul de cumparaturi pentru utilizatorii care au plasat cel putin 2 comenzi.

```
UPDATE COS_CUMPARATURI
SET TOTAL_COS = TOTAL_COS * 0.90
WHERE ID_UTILIZATOR IN (
    SELECT C.ID_UTILIZATOR
    FROM COMANDA C
    GROUP BY ID_UTILIZATOR
    HAVING COUNT(ID_COMANDA) >= 2
);
```

The screenshot shows the Oracle SQL Developer interface. The title bar reads "Oracle SQL Developer : tutorial". The menu bar includes File, Edit, View, Navigate, Run, Source, Team, Tools, Window, and Help. Below the menu is a toolbar with various icons. The Connections tab is selected, showing a connection named "tutorial". The main area is the Worksheet tab, which contains the following SQL code:

```
-- aplicarea unei reduceri de 10% la cosul de cumparaturi
-- pentru utilizatorii care au plasat cel putin 2 comenzi
UPDATE COS_CUMPARATURI
SET TOTAL_COS = TOTAL_COS * 0.90
WHERE ID_UTILIZATOR IN (
    SELECT C.ID_UTILIZATOR
    FROM COMANDA C
    GROUP BY ID_UTILIZATOR
    HAVING COUNT(ID_COMANDA) >= 2
);
```

Below the code, the message "4 rows updated." is displayed.

14. Crearea unei vizualizări complexe. Dați un exemplu de operație LMD permisă pe vizualizarea respectivă și un exemplu de operație LMD nepermisă.

Să se creeze vizualizarea VIZ_PRODUS_CATEGORIE care conține denumirea unui produs, pretul acestuia, categoria din care face parte și de cate ori a fost comandat.

```
CREATE OR REPLACE VIEW VIZ_PRODUS_CATEGORIE AS
SELECT P.DENUMIRE_PRODUS, P.PRET, C.DENUMIRE_CATEGORIE,
```

```

(SELECT COUNT(PC.ID_COMANDA)
  FROM PRODUS_COMANDA PC
 WHERE PC.ID_PRODUS = P.ID_PRODUS
) NR_COMENZI
FROM PRODUS P
JOIN CATEGORIE C ON P.ID_CATEGORIE = C.ID_CATEGORIE;

SELECT * FROM VIZ_PRODUS_CATEGORIE;

```

The screenshot shows the Oracle SQL Developer interface with the following details:

- Toolbar:** File, Edit, View, Navigate, Run, Source, Team, Tools, Window, Help.
- Connections:** A connection named "tutorial" is selected.
- Worksheet:** The main area contains the SQL code for creating the view.
- Code Content:**

```

-- EXERCITIU 14

CREATE OR REPLACE VIEW VIZ_PRODUS_CATEGORIE AS
SELECT P.DENUMIRE_PRODUS, P.PRET, C.DENUMIRE_CATEGORIE,
(SELECT COUNT(PC.ID_COMANDA)
  FROM PRODUS_COMANDA PC
 WHERE PC.ID_PRODUS = P.ID_PRODUS
) NR_COMENZI
FROM PRODUS P
JOIN CATEGORIE C ON P.ID_CATEGORIE = C.ID_CATEGORIE;

SELECT * FROM VIZ_PRODUS_CATEGORIE;

-- OPERATIE PERMISA
-- COLOANELE DENUMIRE_PRODUS SI PRET SUNT SINGURELE COLOANE ACTUALIZABILE DEOARECE APARTIN TABELULUI PRODUS, CARE ESTE KEY-PRESERVED
-- IN PLUS, FUNCTIA ESTE IN SUBCERERE, DECI DENUMIRE_PRODUS SI PRET SUNT ACTUALIZABILE
INSERT INTO VIZ_PRODUS_CATEGORIE (DENUMIRE_PRODUS, PRET) VALUES ('PANTALONI SCURII','89.99');

-- OPERATIE NEPERMISA
-- NU SE POT MODIFICA DATELE DIN MAI MULTE TABLELE DE BAZA
INSERT INTO VIZ_PRODUS_CATEGORIE (DENUMIRE_PRODUS, PRET, DENUMIRE_CATEGORIE) VALUES ('SANDALE DE PIELE','189.99', 'PANTOFI');

-- OPERATIE NEPERMISA
-- TABELUL CATEGORIE NU ESTE KEY-PRESERVED
INSERT INTO VIZ_PRODUS_CATEGORIE (DENUMIRE_CATEGORIE) VALUES ('PIJAMALE');

```

View VIZ_PRODUS_CATEGORIE created.

Script Output | Query Result | SQL | All Rows Fetched: 14 in 0.092 seconds

DENUMIRE_PRODUS	PRET	DENUMIRE_CATEGORIE	NR_COMENZI
1 TRICOU ALBASTRU	55.5	TRICOU	1
2 PANTALONI FORMALI	120	PANTALONI	3
3 CAMASA DE IN	110.7	CAMASA	3
4 TRICOU GRI	49.99	TRICOU	1
5 ROCHIE DE PLAJA	99.99	ROCHIE	1
6 OCHELARI DE SOARE COPII	52.5	OCHELARI DE SOARE	1
7 GEACA DE IARNA	199.99	GEACA	2
8 CAMASA VERDE	89.99	CAMASA	1
9 GEANTA DE UMAR	129.5	GEANTA	2
10 TRICOU IN DUNGI	49.99	TRICOU	1
11 VESTA ALBASTRA	100	VESTA	1
12 FUSTA MIDI	75	FUSTA	0
13 PULOVER DE LANA	150	PULOVER	0
14 ESARFA COLORATA	30	ESARFA	0

O operatie LMD permisa este:

```
INSERT INTO VIZ_PRODUS_CATEGORIE (DENUMIRE_PRODUS, PRET) VALUES ('PANTALONI SCURTI','89.99');
```

1 row inserted.

Operatia este permisa deoarece coloanele denumire_produs si pret sunt singurele coloane actualizabile, ele aparținând tabelului Produs care este key-preserved, iar funcția COUNT este în subcerere.

Operatii LMD nepermise:

```
INSERT INTO VIZ_PRODUS_CATEGORIE (DENUMIRE_PRODUS, PRET, DENUMIRE_CATEGORIE) VALUES ('SANDALE DE PIELE','189.99', 'PANTOFI');
```

Operatia este nepermisa deoarece nu se pot modifica datele din mai multe tabele de baza.

```
Error starting at line : 23 in command -
INSERT INTO VIZ_PRODUS_CATEGORIE (DENUMIRE_PRODUS, PRET, DENUMIRE_CATEGORIE) VALUES ('SANDALE DE PIELE','189.99', 'PANTOFI')
Error at Command Line : 23 Column : 58
Error report -
SQL Error: ORA-01776: cannot modify more than one base table through a join view
01776. 00000 -  "cannot modify more than one base table through a join view"
*Cause:  Columns belonging to more than one underlying table were either
         inserted into or updated.
*Action:  Phrase the statement as two or more separate statements.
```

O alta operatie nepermisa este:

```
INSERT INTO VIZ_PRODUS_CATEGORIE (DENUMIRE_CATEGORIE) VALUES ('PIJAMALE');
```

Tabelul Categorie nu este key-preserved, deci coloana denumire_categorie nu este actualizabila

```
Error starting at line : 27 in command -
INSERT INTO VIZ_PRODUS_CATEGORIE (DENUMIRE_CATEGORIE) VALUES ('PIJAMALE')
Error at Command Line : 27 Column : 35
Error report -
SQL Error: ORA-01779: cannot modify a column which maps to a non key-preserved table
01779. 00000 -  "cannot modify a column which maps to a non key-preserved table"
*Cause: An attempt was made to insert or update columns of a join view which
        map to a non-key-preserved table.
*Action: Modify the underlying base tables directly.
```

15. Formulați în limbaj natural și implementați în SQL: o cerere ce utilizează operația outer-join pe minimum 4 tabele, o cerere ce utilizează operația division și o cerere care implementează analiza top-n. Observație: Cele 3 cereri sunt diferite de cererile de la exercițiul 12.

Outer-join pe minimum 4 tabele

Pentru fiecare produs sa se afiseze denumirea, de cate ori a fost comandat, in cate liste de dorinte se afla, in cate cosuri de cumparaturi se afla si numarul total de aparitii (comenzi+liste+cosuri)

```
SELECT
P.DENUMIRE_PRODUS,
COUNT(DISTINCT PC.ID_COMANDA) AS NUMAR_COMENZI,
COUNT(DISTINCT LP.ID_LISTA) AS NUMAR_LISTE,
COUNT(DISTINCT PCO.ID_COS) AS NUMAR_COSURI,
COUNT(DISTINCT PC.ID_COMANDA) + COUNT(DISTINCT LP.ID_LISTA)+COUNT(DISTINCT PCO.ID_COS) AS APARITII_TOTALE
FROM PRODUS P
LEFT JOIN PRODUS_COMANDA PC ON P.ID_PRODUS = PC.ID_PRODUS
LEFT JOIN PRODUS_LISTA LP ON P.ID_PRODUS = LP.ID_PRODUS
LEFT JOIN PRODUS_COS PCO ON P.ID_PRODUS = PCO.ID_PRODUS
```

```

GROUP BY P.DENUMIRE_PRODUS
ORDER BY APARITII_TOTALE DESC;

```

Am folosit left join deoarece nu toate produsele au fost comandate sau se afla in liste de dorinte/cosuri de cumparaturi, dar dorim sa vizualizam toate produsele, inclusiv cele fara aparitii

The screenshot shows the Oracle SQL Developer interface. The top window is titled "Oracle SQL Developer : tutorial". The "Worksheet" tab is selected, displaying the following SQL code:

```

--Exercitiu 15

--OUTER-JOIN PE MINIMUM 4 TEBELE
--PENTRU FIECARE PRODUS SA SE AFISEZE DENUMIREA, DE CATE ORI A FOST COMANDAT, IN CATE LISTE DE DORINTE SE AFLA,
--IN CATE COSURI DE CUMPARATURI SE AFLA SI NUMARUL TOTAL DE APARITII (COMENZI+LISTE+COSURI)

SELECT
    P.DENUMIRE_PRODUS,
    COUNT(DISTINCT PC.ID_COMANDA) AS NUMAR_COMENZI,
    COUNT(DISTINCT LP.ID_LISTA) AS NUMAR_LISTE,
    COUNT(DISTINCT PCO.ID_COS) AS NUMAR_COSURI,
    COUNT(DISTINCT PC.ID_COMANDA) + COUNT(DISTINCT LP.ID_LISTA)+COUNT(DISTINCT PCO.ID_COS) AS APARITII_TOTALE
FROM PRODUS P
LEFT JOIN PRODUS_COMANDA PC ON P.ID_PRODUS = PC.ID_PRODUS
LEFT JOIN PRODUS_LISTA LP ON P.ID_PRODUS = LP.ID_PRODUS
LEFT JOIN PRODUS_COS PCO ON P.ID_PRODUS = PCO.ID_PRODUS
GROUP BY P.DENUMIRE_PRODUS
ORDER BY APARITII_TOTALE DESC;

-- AM FOLOSIT LEFT JOIN DEOARECE NU TOATE PRODUSELE AU FOST COMANDATE SAU SE AFLA IN LISTE DE DORINTE/COSURI DE CUMPARATURI,
-- DAR DORIM SA VIZUALIZAM TOATE PRODUSELE, INCLUSIV CELE FARĂ APARITII

```

The bottom window is titled "Query Result" and shows the results of the query:

DENUMIRE_PRODUS	NUMAR_COMENZI	NUMAR_LISTE	NUMAR_COSURI	APARITII_TOTALE
1 PANTALONI FORMALI	3	2	1	6
2 CAMASA DE IN	3	2	1	6
3 GEANTA DE UMAR	2	1	1	4
4 TRICOU GRI	1	2	1	4
5 ROCHIE DE PLAJA	1	2	1	4
6 TRICOU ALBASTRU	1	2	1	4
7 GEACA DE IARNA	2	1	1	4
8 OCHELARI DE SOARE COPII	1	1	1	3
9 CAMASA VERDE	1	1	1	3
10 VESTA ALBASTRA	1	1	1	3
11 PULOVER DE LANA	0	1	1	2
12 TRICOU IN DUNGI	1	1	0	2
13 FUSTA MIDI	0	1	0	1
14 ESARFA COLORATA	0	0	0	0
15 PANTALONI SCURTI	0	0	0	0

Division

Să se afiseze informații despre produsele care au fost regasite în toate comenziile plasate în primele 3 luni ale anului.

```
SELECT P.id_producător, P.denumire_producător, P.pret, P.stoc, P.id_categorie
FROM PRODUS P
WHERE NOT EXISTS (
    ( --toate comenziile plasate în primele 3 luni din 2024
        SELECT C.id_comandă
        FROM COMANDA C
        WHERE TO_CHAR(C.data_comandă, 'YYYY-MM') BETWEEN '2024-01' AND '2024-03'
    )
    MINUS
    ( --toate comenziile din primele 3 luni din 2024 în care a fost comandat produsul
        SELECT PC.id_comandă
        FROM PRODUS_COMANDA PC
        JOIN COMANDA C ON PC.id_comandă = C.id_comandă
        WHERE TO_CHAR(C.data_comandă, 'YYYY-MM') BETWEEN '2024-01' AND '2024-03'
        AND PC.id_producător = P.id_producător
    )
);
```

Pentru fiecare produs, am scăzut din multimea comenziilor plasate în primele 3 luni ale anului multimea comenziilor (plasate în primele 3 luni) în care apare respectivul produs. Dacă rezultatul este multumea vida atunci produsul îndeplinește cerința.

Oracle SQL Developer : tutorial

File Edit View Navigate Run Source Team Tools Window Help

tutorial Connections

Worksheet Query Builder

```
-- DIVISION

--Să se afiseze informații despre produsele care au fost regasite în toate comenziile plasate în
--primele 3 luni ale anului

SELECT P.id_produs, P.denumire_produs, P.pret, P.stoc, P.id_categorie
FROM PRODUS P
WHERE NOT EXISTS (
    --toate comenziile plasate în primele 3 luni din 2024
    SELECT C.id_comanda
    FROM COMANDA C
    WHERE TO_CHAR(C.data_comanda, 'YYYY-MM') BETWEEN '2024-01' AND '2024-03'
)
MINUS
--toate comenziile din primele 3 luni din 2024 în care a fost comandat produsul
SELECT PC.id_comanda
FROM PRODUS_COMANDA PC
JOIN COMANDA C ON PC.id_comanda = C.id_comanda
WHERE TO_CHAR(C.data_comanda, 'YYYY-MM') BETWEEN '2024-01' AND '2024-03'
    AND PC.id_produs = P.id_produs
);
;
```

Script Output | Query Result

All Rows Fetched: 1 in 0.258 seconds

ID_PRODUS	DENUMIRE_PRODUS	PRET	STOC	ID_CATEGORIE
1	6 OCHELARI DE SOARE COPII	52.5	5	8

Analiza top-n

Să se afiseze numele, prenumele, numarul de livrari și data ultimei livrari pentru primii 3 curieri cu cele mai multe livrari efectuate în anul 2024.

```
SELECT *
```

```
FROM (SELECT CONCAT(CONCAT(C.NUME, ','), C.PRENUME) AS NUME,
COUNT(CO.ID_LIVRARE) AS NUMAR_COMENZI,
MAX(CO.DATA_LIVRARE) AS ULTIMA_COMANDA
FROM CURIER C
JOIN LIVRARE CO ON CO.ID_CURIER = C.ID_CURIER
WHERE TO_NUMBER(TO_CHAR(CO.DATA_LIVRARE, 'YYYY')) = 2024)
```

```

        GROUP BY C.NUME, C.PRENUME
        ORDER BY NUMAR_COMENZI DESC
    ) CURIER_ACTIVI
    WHERE ROWNUM<=3;

```

The screenshot shows the Oracle SQL Developer interface. The top part is the 'Worksheet' tab where the SQL query is typed. The bottom part is the 'Query Result' tab where the executed query's output is displayed as a table.

```

-- ANALIZA TOP N
-- SA SE AFISEZE NUMELE, PRENUMELE, NUMARUL DE LIVRARI SI DATA ULTIMEI LIVRARI EFECTUATE
-- PENTRU PRIMII 3 CURIERI CU CELE MAI MULTE LIVRARI EFECTUATE IN ANUL 2024
SELECT *
FROM (SELECT CONCAT(CONCAT(C.NUME, ' '), C.PRENUME) AS NUME,
             COUNT(CO.ID_LIVRARE) AS NUMAR_COMENZI,
             MAX(CO.DATA_LIVRARE) AS ULTIMA_COMANDA
        FROM CURIER C
       JOIN LIVRARE CO ON CO.ID_CURIER = C.ID_CURIER
      WHERE TO_NUMBER(TO_CHAR(CO.DATA_LIVRARE, 'YYYY')) = 2024
     GROUP BY C.NUME, C.PRENUME
    ORDER BY NUMAR_COMENZI DESC
) CURIER_ACTIVI
WHERE ROWNUM<=3;

```

	NUME	NUMAR_COMENZI	ULTIMA_COMANDA
1	ION ALEXANDRU	2	20-MAY-24
2	IONESCU GEORGE	2	15-MAY-24
3	POPESCU MARIA	1	12-MAY-24

16. Optimizarea unei cereri, aplicând regulile de optimizare ce derivă din proprietățile operatorilor algebrei relaționale. Cererea va fi exprimată prin expresie algebraică, arbore algebraic și limbaj (SQL), atât anterior cât și ulterior optimizării.

17. a. Realizarea normalizării BCNF, FN4, FN5.
 b. Aplicarea denormalizării, justificând necesitatea acesteia.

a. BCNF

Un tabel se află în BCNF dacă nu există dependențe parțiale (toate atributele trebuie să depindă de întreagă cheie primă), nu există dependențe tranzitive (toate atributele sunt determinante direct de către cheia primă) și nu există dependențe circulare (nu există atribut în cheie care să depindă de atribut non-cheie).

Putem lua ca exemplu tabela ADRESA, cu atributele, id_adresa, strada, numar_strada, oraș, tara, cod_zip

Non-BCNF

id_adresa	strada	oras	tara	numar_strada	cod_zip
1	Bd. Tomis 24	Constanta	Romania	24	900407
2	Bd. Aurel Vlaicu 120	Constanta	Romania	120	900235
3	Str. Dezrobirii 12	Brasov	Romania	12	800321
4	Calea Victoriei 56	Bucuresti	Romania	56	700543

Considerăm următoarele chei candidat: {id_adresa}, {strada, cod_zip}, {strada, oraș}

Perechile {strada, cod_zip}, {strada, oraș} nu îndeplinesc condiția de dependență circulară, având:

(strada, cod_zip) \rightarrow număr_strada

număr_strada \rightarrow strada

Analog pentru (strada, oraș).

Asadar tabela nu este în BCNF. Pentru a aduce tabela în forma BCNF putem schimba atributul strada în denumire_strada, unde vom avea doar numele străzii. O alta alternativă ar fi renunțarea la coloana număr_strada.

Transformare în BCNF

id_adresa	denumire_strada	oras	tara	numar_strada	cod_zip
1	Bd. Tomis	Constanta	Romania	24	900407

2	Bd. Aurel Vlaicu	Constanta	Romania	120	900235
3	Str. Dezrobirii	Brasov	Romania	12	800321
4	Calea Victoriei	Bucuresti	Romania	56	700543

FN4

Un tabel se află în FN4 dacă se află deja în forma BCNF și elimină redundanțele datorate multidependenței.

Să considerăm tabelul UTILIZATOR_CARD_ADRESA (id_utilizator, nume_utilizator, id_card, numar_card, id_adresa, strada, oras). Un utilizator poate avea mai multe carduri, iar fiecare card poate fi asociat cu mai multe adrese de facturare. Acest tabel conține o multidependență nepermisă în forma FN4, id_utilizator determină atât id_card, cât și id_adresa.

Pentru a rezolva asta putem sparge tabelul în două tabele, fiecare conținând o singură multidependență.

Așfel, UTILIZATOR_CARD_ADRESA devine UTILIZATOR_CARD (id_utilizator, nume_utilizator, id_card, numar_card) și UTILIZATOR_ADRESA (id_utilizator, nume_utilizator, id_adresa, strada, oras).

FN5

Un tabel se află în FN5 dacă se află deja în FN4 și elimină redundanțele date de dependență ciclică. Așfel, o tabelă nu este în FN5 dacă există dependențe join nepermise.

Considerăm tabela PRODUS_COMANDA_LIVRARE (id_produs, id_comanda, id_livrare). Acest tabel poate suferi de o problemă de dependență join deoarece există o relație între atributele tabelei.

Pentru a transforma această tabelă în FN5, trebuie să eliminăm dependențele join, descompunând tabelul în tabele mai mici care păstrează informațiile fără a introduce redundanțe.

Asadar, tabelul inițial se poate sparge în PRODUS_COMANDA, PRODUS_LIVRARE și COMANDA_LIVRARE. Prin separarea tabelei PRODUS_COMANDA_LIVRARE eliminăm aceste dependențe join și facem ca fiecare tabel să conțină o singură relație clară.

b. Denormalizarea unei baze de date presupune adăugarea unor redundante pentru a îmbunătăți viteza interogărilor. Aceasta se justifica în cazurile în care interogările de citire sunt foarte frecvente sau dorim să simplificăm interogările, deoarece reduce numărul de join-uri necesare.

Puteam lua în considerare denormalizarea tabelei Comanda pentru a include informații despre utilizatorul care a plasat comanda și detalii despre livrare, pentru a evita operațiile de join.

Tabela COMANDA (id_comanda, total_plata, stare_comanda, data_comanda, id_utilizator, id_livrare) poate fi transformată în COMANDA (id_comanda, total_plata, stare_comanda, data_comanda, id_utilizator, nume_utilizator, prenume_utilizator, email_utilizator, id_livrare, metoda_livrare, pret_livrare, data_livrare)

Avantajul denormalizării constă în imbunătățirea performanței și simplificarea interogărilor prin eliminarea operațiilor de join complexe. Principalele dezavantaje sunt redundanța datelor și costul de actualizare, deoarece informațiile sunt stocate în mai multe locuri și pot apărea inconștiente dacă datele nu sunt actualizate corect.