

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

Modelul de date va gestiona informații despre un magazin care se ocupă cu vânzarea componentelor, accesoriilor, și serviciilor destinate calculatoarelor și laptop-urilor. Serviciile oferite constau în: diagnosticarea, montarea, asamblarea, și testarea produselor. În afara celor precizate anterior, magazinul mai are inclus în oferta sa atât sisteme deja asamblate, precum: calculatoare și laptop-uri.

Există mai multe categorii de produse, fiecare produs aparținându-i unei singure categorii.

Magazinul deține și un website, prin intermediul căruia clienții își pot crea unul sau mai multe conturi, cu scopul de a explora ofertele magazinului, de a plasa comenzi, de a comunica cu angajații în legătură cu anumite nelămuriri sau eventuale sfaturi, cât și de a adăuga recenzii produselor cumpărate de aceștia.

Magazinul are două tipuri de angajați: paznici și tehnicieni. De asemenea, angajații pot asista clienții în legătură cu diferite întrebări ale acestora, atât în incinta magazinului, cât și prin intermediul website-ului.

2. Prezentarea constrângerilor (restricții, reguli) impuse asupra modelului:

- Un client poate șterge doar recenziile scrise de acesta;
- Un client nu poate să adauge mai mult de o recenzie pentru același produs;
- Atât tehnicienii cât și paznicii lucrează într-un singur department, iar paznicii lucrează într-un singur tip de departament, anume cel de securitate;
- Clienții nu își pot șterge conturile;
- Nu există conturi cu același nume de utilizator;
- Un produs trebuie să facă parte dintr-o singură categorie;
- Orice cont trebuie să aibă specificată o adresă de e-mail;
- Orice angajat trebuie să aibă specificată o adresă de e-mail și un număr de telefon;
- Trebuie să se cunoască data sosirii și prețul total al tuturor comenzilor;
- Trebuie să se cunoască data realizării tuturor serviciilor.

3. Descrierea entităților, incluzând precizarea cheii primare:

Entitățile modelului de date prezentat sunt: CATEGORIE, PRODUS (cu subentitățile: COMPONENTA, ACCESORIU, SISTEM), SERVICIU (cu subentitățile: DIAGNOSTICARE, ASAMBLARE, TESTARE), ADRESA_LIVRARE, CONT_SITE, RECENZIE, CLIENT, ANGAJAT (cu subentitățile: TEHNICIAN, PAZNIC), DEPARTAMENT.

CATEGORIE: încadrează produsele magazinului într-o anumită categorie. Cheia primară a entității este `id_categorie#`.

PRODUS: bunul material principal vândut de către magazin, atât prin intermediul site-ului, cât și prin intermediul magazinului real. Acesta poate fi de trei tipuri, anume: componentă, accesoriu, sistem. Cheia primară a entității este `id_produs#`.

COMPONENTA: subentitate a entității PRODUS, care conține informații despre diverse componente pentru calculatoare sau laptop-uri: plăci video, procesoare, plăci de bază, memorii RAM, spații de stocare, etc. Cheia primară a entității este `id_produs#`.

ACCESORIU: subentitate a entității PRODUS, care conține informații despre diverse accesorii pentru calculatoare sau laptop-uri: cabluri, adaptoare, camere web, etc. Cheia primară a entității este `id_produs#`.

SISTEM: subentitate a entității PRODUS, care poate conține informații doar despre două tipuri de produse: PC, laptop. Cheia primară a entității este `id_produs#`.

SERVICIU: activitate prestată de către magazin, asupra propriilor produse cât și asupra produselor care au fost cumpărate din cadrul altor magazine, de către clienți. Această activitate este de patru tipuri: diagnosticare, montare, asamblare, testare. Cheia primară a entității este `id_serviciu#`.

DIAGNOSTICARE: subentitate a entității SERVICIU, care conține informații despre modalitățile de diagnosticare a diferitelor produse cumpărate de către clienți, în cazul defectării acestora. Cheia primară a entității este `id_serviciu#`.

MONTARE: subentitate a entității SERVICIU, care conține informații despre modalitățile de montare a diferitelor componente, cumpărate de către clienți. Cheia primară a entității este `id_serviciu#`.

ASAMBLARE: subentitate a entității SERVICIU, care conține informații despre modalitățile de asamblare a diferitelor sisteme, cumpărate de către clienți. Cheia primară a entității este `id_serviciu#`.

TESTARE: subentitate a entității SERVICIU, care conține informații despre modalitățile de testare a diferitelor componente, sisteme, și accesorii, cumpărate de către clienți. Cheia primară a entității este id_serviciu#.

ADRESA_LIVRARE: adresa la care se livrează produsele, comandate de către clienți. Plasarea comenzii se poate realiza atât la sediul magazinului, cât și pe site-ul acestuia. Cheia primară a entității este id_adresa#.

CONT_SITE: contul unui client de pe site-ul magazinului. Cheia primară a entității este compusă din id_cont# și nume_utilizator#. nume_utilizator# este cheie primară artificială.

RECENZIE: recenzie scrisă de către un client, prin intermediul contului său, pentru un anumit produs. Cheia primară a entității este compusă din id_recenzie# și id_produs#.

CLIENT: persoană fizică care beneficiază de produsele și serviciile, și care poate avea unul sau mai multe conturi pe site-ul acestuia. Cheia primară a entității este id_client#.

ANGAJAT: persoană fizică, angajată a magazinului, care poate ocupa unul din două posturi: tehnician sau paznic. Cheia primară a entității este id_angajat#.

TEHNICIAN: subentitate a entității ANGAJAT, care conține informații despre angajații tehnicieni ai magazinului. Cheia primară a entității este id_angajat#.

PAZNIC: subentitate a entității ANGAJAT, care conține informații despre angajații al căror statut este cel de paznic. Cheia primară a entității este id_angajat#.

DEPARTAMENT: secțiune a magazinului în care pot să lucreze mai mulți angajați sau niciunul. Cheia primară a entității este id_departament#.

4. Descrierea relațiilor, incluzând precizarea cardinalității acestora:

PRODUS_face_parte_CATEGORIE: relație care leagă entitățile PRODUS și CATEGORIE (din ce categorie face parte un produs), având cardinalitatea minimă 0:1 și cardinalitatea maximă n:1.

COMPONENTA_IS A_PRODUS: relație care leagă subentitatea COMPONENTA de superentitatea PRODUS (un produs poate fi o componentă), având cardinalitatea minimă 1:0 și cardinalitatea maximă 1:1.

ACCESORIU_IS A_PRODUS: relație care leagă subentitatea ACCESORIU de superentitatea PRODUS (un produs poate fi un accesoriu), având cardinalitatea minimă 1:0 și cardinalitatea maximă 1:1.

SISTEM_IS A_PRODUS: relație care leagă subentitatea SISTEM de superentitatea PRODUS (un produs poate fi un sistem), având cardinalitatea minimă 1:0 și cardinalitatea maximă 1:1.

PRODUS_este_compatibil_SERVICIU: relație care leagă entitățile PRODUS și SERVICIU (cu ce servicii sunt compatibile produsele), având cardinalitatea minimă 0:0 și cardinalitatea maximă m:n.

DIAGNOSTICARE_IS A_SERVICIU: relație care leagă subentitatea DIAGNOSTICARE de superentitatea SERVICIU (un serviciu poate fi de tipul diagnosticare), având cardinalitatea minimă 1:0 și cardinalitatea maximă 1:1.

ASAMBLARE_IS A_SERVICIU: relație care leagă subentitatea ASAMBLARE de superentitatea SERVICIU (un serviciu poate fi de tipul asamblare), având cardinalitatea minimă 1:0 și cardinalitatea maximă 1:1.

TESTARE_IS A_SERVICIU: relație care leagă subentitatea TESTARE de superentitatea SERVICIU (un serviciu poate fi de tipul testare), având cardinalitatea minimă 1:0 și cardinalitatea maximă 1:1.

CLIENT_cumpara_SERVICIU: relație care leagă entitățile CLIENT și SERVICIU (ce servicii pot cumpăra clienții), având cardinalitatea minimă 0:0 și cardinalitatea maximă m:n.

CLIENT_detine_CONT_SITE: relație care leagă entitățile CLIENT și CONT_SITE (care sunt conturile deținute de un client), având cardinalitatea minimă 1:0 și cardinalitatea maximă 1:n.

RECENZIE_este_scrisa_CONT_SITE: relație care leagă entitățile RECENZIE și CONT_SITE (care sunt recenziile scrise de către un cont), având cardinalitatea minimă 0:1 și cardinalitatea maximă n:1.

PRODUS_are_RECENZIE: relație care leagă entitățile PRODUS și RECENZIE (ce recenzii au produsele), având cardinalitatea minimă 1:0 și cardinalitatea maximă 1:n.

CLIENT_comanda_PRODUS_la_ADRESA_LIVRARE: relație de tip 3 care leagă entitățile CLIENT, PRODUS și ADRESA_LIVRARE, arătând ce produse a cumpărat clientul, alături de data la care au fost livrate produsele respective, și prețul total al comenzii. Denumirea acestei relații va fi "comanda" și va avea următoarele cardinalități: cardinalitate minimă 0:0 și cardinalitate maximă m:n între entitățile PRODUS și CLIENT, cardinalitate minimă 0:0 și cardinalitate maximă m:n între entitățile PRODUS și ADRESA_LIVRARE, cardinalitate minimă 0:1 și cardinalitate maximă m:n între entitățile CLIENT și ADRESA_LIVRARE.

CLIENT_este_asistat_ANGAJAT: relație care leagă entitățile CLIENT și ANGAJAT (ce angajați asistă clienții), având cardinalitatea minimă 0:0 și cardinalitatea maximă m:n.

TEHNICIAN_IS A_ANGAJAT: relație care leagă subentitatea TEHNICIAN de superentitatea ANGAJAT (un angajat poate fi tehnician), având cardinalitatea minimă 1:0 și cardinalitatea maximă 1:1.

PAZNIC_IS A_ANGAJAT: relație care leagă subentitatea PAZNIC de superentitatea ANGAJAT (un angajat poate fi paznic), având cardinalitatea minimă 1:0 și cardinalitatea maximă 1:1.

ANGAJAT_lucreaza_DEPARTAMENT: relație care leagă entitățile ANGAJAT și DEPARTAMENT (în ce departamente lucrează angajații), având cardinalitatea minimă 0:1 și cardinalitatea maximă n:1.

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

Entitatea CATEGORIE are attributele:

- id_categorie: variabilă de tip întreg, de lungime maximă 3, (aparținând intervalului închis [1, 399]) care reprezintă id-ul unei categorii de produse.
- denumire: variabilă de tip caracter, de lungime maximă 40, care reprezintă numele unei categorii de produse.

Entitatea PRODUS are attributele:

- id_produs: variabilă de tip întreg, pozitivă, care reprezintă id-ul unui produs.
- id_categorie: variabilă de tip întreg, pozitivă, de lungime maximă 3, (aparținând intervalului închis [1, 399]) care reprezintă id-ul unei categorii de produse. Atributul trebuie să corespundă unei valori a cheii primare din tabelul CATEGORIE.
- tip: variabilă de tip caracter, de lungime maximă 10, care reprezintă tipul unui produs (componentă, accesoriu, sau sistem).
- denumire: variabilă de tip caracter, de lungime maximă 200, care reprezintă numele unui produs.
- vanzator: variabilă de tip caracter, de lungime maximă 40, care reprezintă numele vânzătorului unui produs.
- recomandat_pentru_gaming: variabilă de tip caracter, de lungime 2, luând valorile "Da" sau "Nu", indicând dacă un produs este recomandat pentru gaming sau nu.
- pret: variabilă de tip numeric, reprezentând prețul unui produs în RON.

- average_rating: variabilă de tip numeric, de lungime 1, reprezentând media aritmetică a rating-urilor produsului date de către clienți (rating-ul fiecărui produs este o valoare reală cuprinsă între 1 și 5).

Subentitatea COMPONENTA a superentității PRODUS are attributele:

- id_produș: variabilă de tip întreg, pozitivă, care reprezintă id-ul unui produs de tip componentă.

Subentitatea ACCESORIU a superentității PRODUS are attributele:

- id_produș: variabilă de tip întreg, pozitivă, care reprezintă id-ul unui produs de tip accesoriu.

Subentitatea SISTEM a superentității PRODUS are attributele:

- id_produș: variabilă de tip întreg, pozitivă, care reprezintă id-ul unui produs de tip sistem.
- tip_sistem: variabilă de tip caracter, de lungime maximă 30, reprezentând tipul unui sistem (PC sau laptop).

Entitatea SERVICIU are attributele:

- id_serviciu: variabilă de tip întreg, pozitivă, care reprezintă id-ul unui serviciu.
- tip: variabilă de tip caracter, de lungime maximă 13, care reprezintă tipul unui serviciu (diagnosticare, montare, asamblare, sau testare).
- denumire: variabilă de tip caracter, de lungime maximă 70, care reprezintă numele unui serviciu.
- descriere: variabilă de tip caracter, de lungime maximă 1000, care reprezintă o descriere a serviciului.
- pret: variabilă de tip numeric, pozitivă, care reprezintă prețul unui serviciu în RON.

Subentitatea DIAGNOSTICARE a superentității SERVICIU are attributele:

- id_serviciu: variabilă de tip întreg, pozitivă, care reprezintă id-ul unui serviciu de tip diagnosticare.

Subentitatea MONTARE a superentității SERVICIU are attributele:

- id_serviciu: variabilă de tip întreg, pozitivă, de lungime maximă 2, care reprezintă id-ul unui serviciu de tip montare.

Subentitatea ASAMBLARE a superentității SERVICIU are attributele:

- id_serviciu: variabilă de tip întreg, pozitivă, de lungime maximă 2, care reprezintă id-ul unui serviciu de tip asamblare.

Subentitatea TESTARE a superentității SERVICIU are attributele:

- id_serviciu: variabilă de tip întreg, pozitivă, de lungime maximă 2, care reprezintă id-ul unui serviciu de tip testare.

Entitatea ADRESA_LIVRARE are attributele:

- id_adresa: variabilă de tip întreg, pozitivă, care reprezintă id-ul unei adrese de livrare.
- localitate: variabilă de tip caracter, de lungime maximă 30, care reprezintă numele localității în care se livrează o comandă.
- strada: variabilă de tip caracter, de lungime maximă 80, care reprezintă numele unei străzi la care se livrează o comandă.
- nr_strada: variabilă de tip întreg, pozitivă, care reprezintă numărul unei străzi la care se livrează o comandă.

Entitatea CLIENT are attributele:

- id_client: variabilă de tip întreg, pozitivă, care reprezintă id-ul unui client.
- nume: variabilă de tip caracter, de lungime maximă 40, care reprezintă numele unui client.
- prenume: variabilă de tip caracter, de lungime maximă 40, care reprezintă prenumele unui client.

Entitatea CONT_SITE are attributele:

- id_cont: variabilă de tip întreg, pozitivă, care reprezintă id-ul unui cont de pe website.
- nume_utilizator: variabilă de tip caracter, reprezentând numele de utilizator al unui cont de pe website.
- id_client: variabilă de tip întreg, pozitivă, care reprezintă id-ul unui client. Atributul trebuie să corespundă unei valori a cheii primare din tabelul CLIENT.
- email: variabilă de tip caracter, care reprezintă adresa de e-mail a unui cont de pe website.

Entitatea RECENZIE are attributele:

- id_recenzie: variabilă de tip întreg, pozitivă, care reprezintă id-ul unei recenzii scrise de către un client, pentru un produs cumpărat de acesta, prin intermediul unui cont de pe website.
- id_produs: variabilă de tip întreg, pozitivă, care reprezintă id-ul unui produs. Atributul trebuie să corespundă unei valori a cheii primare din tabelul PRODUS.
- nume_utilizator: variabilă de tip caracter, reprezentând numele de utilizator al unui cont de pe website. Atributul trebuie să corespundă unei valori a cheii primare din tabelul CONT_SITE.
- data_scriere: variabilă de tip dată calendaristică, reprezentând data la care a fost scrisă o recenzie.
- nr_stele: variabilă de tip numeric, de lungime 1, reprezentând rating-ul dat de către un client, prin intermediul unui cont de pe website, pentru un produs cumpărat de acesta (rating-ul fiecărui client este o valoare reală cuprinsă între 1 și 5).

Entitatea ANGAJAT are attributele:

- id_angajat: variabilă de tip întreg, pozitivă, care reprezintă id-ul unui angajat.
- id_departament: variabilă de tip întreg, pozitivă, care reprezintă id-ul unui departament.
- tip: variabilă de tip caracter, de lungime maximă 9, care reprezintă tipul unui angajat (tehnician, paznic).
- nume: variabilă de tip caracter, de lungime maximă 30, care reprezintă numele unui angajat.
- prenume: variabilă de tip caracter, de lungime maximă 30, care reprezintă prenumele unui angajat.
- salariu: variabilă de tip întreg, pozitivă, care reprezintă salariul unui angajat, în RON.
- email: variabilă de tip caracter, de lungime maximă 40, care reprezintă adresa de e-mail a unui angajat.
- nr_telefon: variabilă de tip caracter, de lungime 10, care reprezintă numărul de telefon al unui angajat.

Subentitatea TEHNICIAN a superentității ANGAJAT are attributele:

- id_angajat: variabilă de tip întreg, pozitivă, care reprezintă id-ul unui angajat de tip tehnician.

Subentitatea PAZNIC a superentității ANGAJAT are attributele:

- id_angajat: variabilă de tip întreg, pozitivă, care reprezintă id-ul unui angajat de tip paznic.

Entitatea DEPARTAMENT are attributele:

- id_departament: variabilă de tip întreg, pozitivă, care reprezintă id-ul unui departament (poate lua valori cuprinse între 1 și 20).
- denumire: variabilă de tip caracter, de lungime maximă 30, care reprezintă numele unui departament.

Relația ESTE_COMPATIBIL are attributele:

- id_serviciu: variabilă de tip întreg, pozitivă, care reprezintă id-ul unui serviciu. Atributul trebuie să corespundă unei valori a cheii primare din tabelul SERVICIU.
- id_produs: variabilă de tip întreg, pozitivă, care reprezintă id-ul unui produs. Atributul trebuie să corespundă unei valori a cheii primare din tabelul PRODUS.

Relația CUMPARA are attributele:

- id_cumpara: variabilă de tip întreg, pozitivă, care reprezintă id-ul unei achiziții a unui serviciu de către un client.
- id_serviciu: variabilă de tip întreg, pozitivă, care reprezintă id-ul unui serviciu. Atributul trebuie să corespundă unei valori a cheii primare din tabelul SERVICIU.
- id_client: variabilă de tip întreg, pozitivă, care reprezintă id-ul unui client. Atributul trebuie să corespundă unei valori a cheii primare din tabelul CLIENT.
- data_realizare: variabilă de tip dată calendaristică, reprezentând data la care s-a realizat un serviciu cumpărat de către un client.

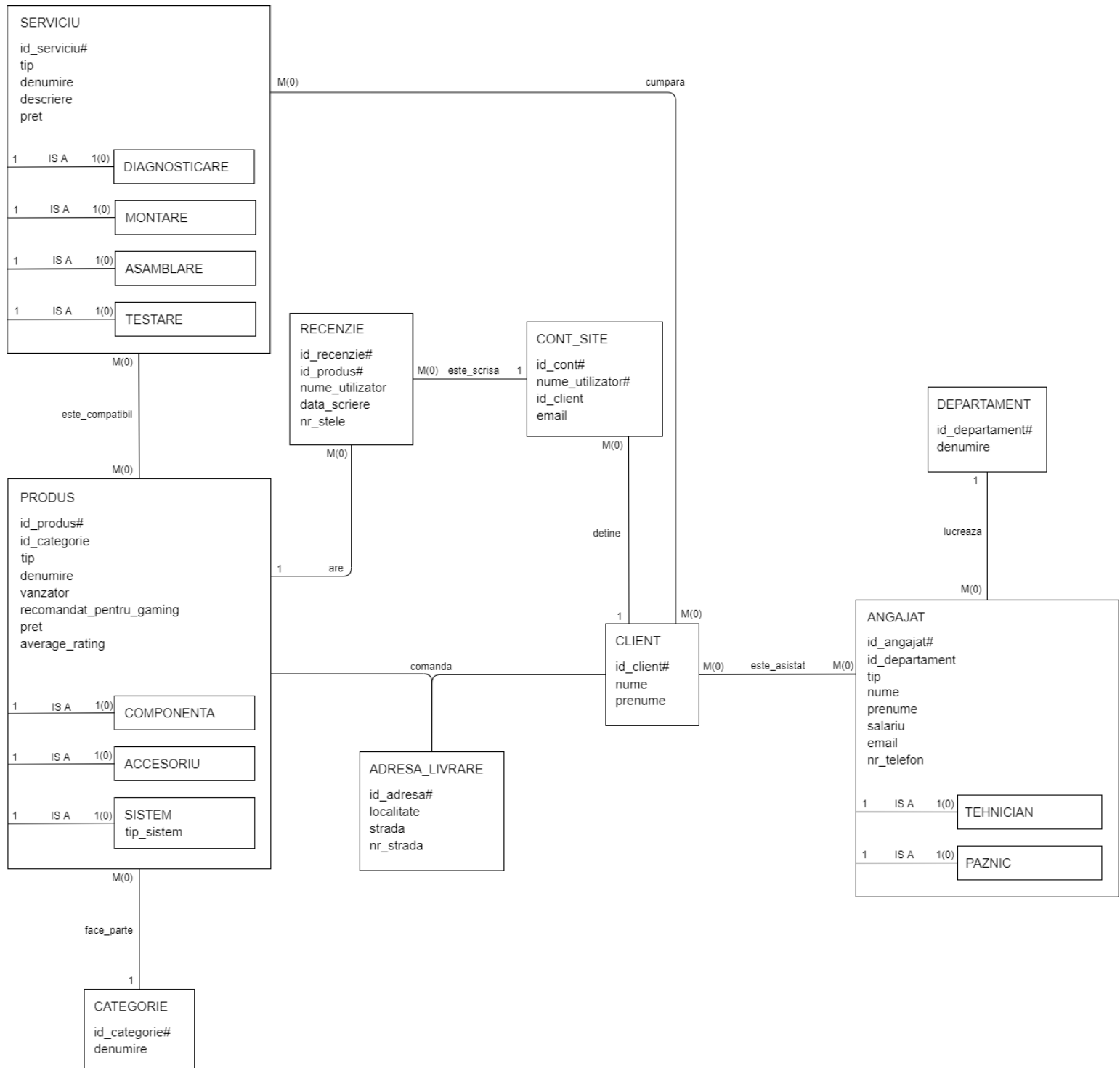
Relația COMANDA are attributele:

- id_comanda: variabilă de tip întreg, pozitivă, care reprezintă id-ul unei comenzi.
- id_produs: variabilă de tip întreg, pozitivă, care reprezintă id-ul unui produs de tip sistem. Atributul trebuie să corespundă unei valori a cheii primare din tabelul PRODUS.
- id_client: variabilă de tip întreg, pozitivă, care reprezintă id-ul unui client. Atributul trebuie să corespundă unei valori a cheii primare din tabelul CLIENT.
- id_adresa: variabilă de tip întreg, pozitivă, care reprezintă id-ul unei adrese de livrare. Atributul trebuie să corespundă unei valori a cheii primare din tabelul ADRESA_LIVRARE.
- data_sosire: variabilă de tip dată calendaristică, care reprezintă data la care o comandă i-a fost livrată unui client.
- pret_total: variabilă de tip numeric, pozitivă, reprezentând prețul unui produs în RON.

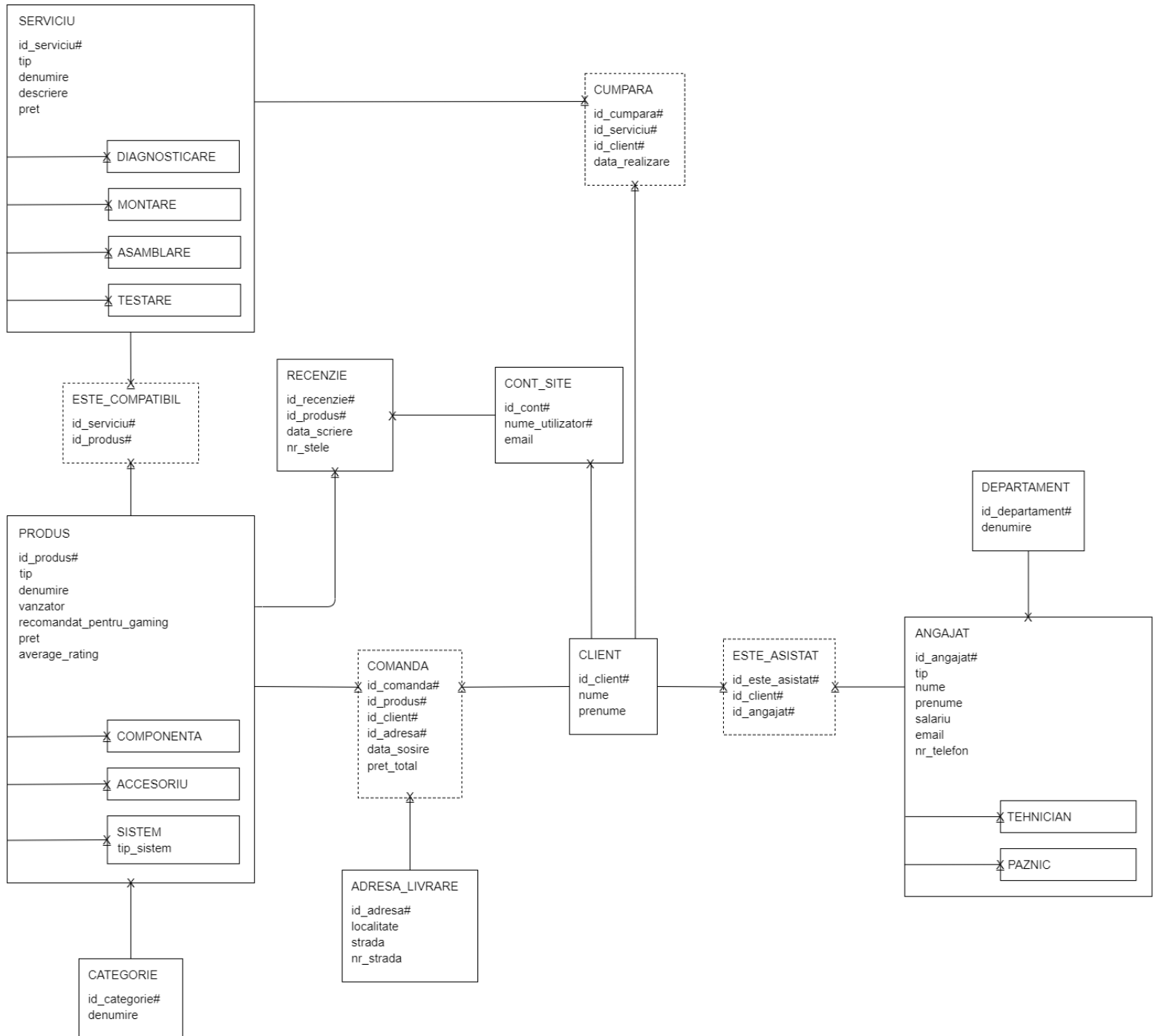
Relația ESTE_ASISTAT are attributele:

- `id_este_asistat`: variabilă de tip întreg, pozitivă, care reprezintă id-ul unei asistări a unui client, realizată de către un angajat.
- `id_client`: variabilă de tip întreg, pozitivă, care reprezintă id-ul unui client. Atributul trebuie să corespundă unei valori a cheii primare din tabelul CLIENT.
- `id_angajat`: variabilă de tip întreg, pozitivă, care reprezintă id-ul unui angajat. Atributul trebuie să corespundă unei valori a cheii primare din tabelul ANGAJAT.

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



7. Realizarea diagramei conceptuale corespunzătoare diagramei entitate-relație proiectată la punctul 6:



8. Enumerarea schemelor relaționale corespunzătoare diagramei conceptuale proiectată la punctul 7.:

- CATEGORIE(id_categorie#, denumire)
- PRODUS(id_produs#, id_categorie, tip, denumire, vanzator, recomandat_pentru_gaming, pret, average_rating)
- COMPONENTA(id_produs#)
- ACCESORIU(id_produs#)
- SISTEM(id_produs#, tip_sistem)
- ESTE_COMPATIBIL(id_serviciu#, id_produs#)
- SERVICIU(id_serviciu#, tip, denumire, descriere, pret)
- DIAGNOSTICARE(id_serviciu#)
- ASAMBLARE(id_serviciu#)
- TESTARE(id_serviciu#)
- CUMPARA(id_cumpara#, id_serviciu#, id_client#, data_realizare)
- ADRESA_LIVRARE(id_adresa#, localitate, strada, nr_strada)
- CLIENT(id_client#, nume , prenume)
- COMANDA(id_comanda#, id_produs#, id_client#, id_adresa#, data_sosire, pret_total)
- RECENZIE(id_recenzie#, id_produs#, nume_utilizator, data_scriere, nr_stele)
- CONT_SITE(id_cont#, nume_utilizator#, id_client, email)
- ESTE_ASISTAT(id_este_asistat#, id_client#, id_angajat#)
- ANGAJAT(id_angajat#, id_department, tip, nume, prenume, salariu, email, nr_telefon)
- TEHNICIAN(id_angajat#)
- PAZNIC(id_angajat#)
- DEPARTAMENT(id_department#, denumire)

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

FN1:

Pentru această formă normală, ne vom lega atât de informații din modelul real, cât și de informații fictive.

Varianta non-FN1:

INFORMATII_CLIENT				
id_client#	nume	prenume	nume_utilizator	email
1	Rares	Mihai	Utilizator Mihai, MihaiR5, Mihai Rares	uzmihai2005@gmail.com
2	Castan	Mirela	Mirela Castan, Mirela_C	mirela_c@gmail.com

Varianta FN1:

CLIENT		
id_client#	nume	prenume
1	Rares	Mihai
2	Castan	Mirela

CONT_SITE			
id_cont#	nume_utilizator#	id_client	email
1	Utilizator Mihai	1	uzmihai2005@gmail.com
2	MihaiR5	1	uzmihai2005@gmail.com
3	Mihai Rares	1	uzmihai2005@gmail.com
4	Mirela Castan	2	mirela_c@gmail.com
5	Mirela_C	2	mirela_c@gmail.com

- Relația inițială (cea de la „Varianta non-FN1”) nu se află în FN1, deoarece atributul „nume_utilizator” poate avea valori multiple.
- Pentru a aduce relația inițială în FN1, am spart tabelul „INFORMATII_CLIENT” în tabelele „CLIENT” și „CONT_SITE”, în care fiecărui atribut îi corespunde o valoare indivizibilă.

FN2:

Pentru această formă normală, ne vom lega atât de informații din modelul real, cât și de informații fictive.

Varianta non-FN2:

INFORMATII_CLIENT					
id_client#	id_cont#	nume_utilizator#	nume	prenume	email
1	1	Utilizator Mihai	Rares	Mihai	uzmihai2005@gmail.com
1	2	xXMihaiXx	Rares	Mihai	uzmihai2005@gmail.com
1	3	MihaiUZ	Rares	Mihai	mihaiemail05@yahoo.com

Varianta FN2:

CLIENT		
id_client#	nume	prenume
1	Rares	Mihai

CONT_SITE			
id_cont#	nume_utilizator#	id_client	email
1	Utilizator Mihai	1	uzmihai2005@gmail.com
2	xXMihaiXx	1	uzmihai2005@gmail.com
3	MihaiUZ	1	mihaiemail05@yahoo.com

- Relația inițială (cea de la „Varianta non-FN2”) se află deja în FN1, deoarece atributele nu pot avea valori multiple, astfel că acestora le corespunde o valoare indivizibilă.
- Relația inițială nu se află în FN2, deoarece atributele „nume” și „prenume” depind direct doar de „id_client#”, nu și de restul cheii primare compuse.
- Pentru a aduce relația inițială în FN2, am spart tabelul „INFORMATII_CLIENT” în tabelele „CLIENT” și „CONT_SITE”, astfel reducând numărul de apariții al numelui și prenumelui clientului cu id-ul 1.
- Aducerea relației inițiale în FN2 s-a realizat prin intermediul regulii Casey-Delobel, cu ajutorul căreia am obținut proiecțiile:
 - CLIENT(id_client#, nume ,prenume) cu dependența {id_client#} -> {nume, prenume}
 - CONT_SITE(id_cont#, nume_utilizator#, id_client, email) cu dependența {id_cont#, nume_utilizator#} -> {email}

FN3:

Pentru această formă normală, ne vom lega atât de informații din modelul real, cât și de informații fictive.

Varianta non-FN3:

CONT_SITE			
id_cont#	nume_utilizator#	email	nume_utilizator_email
1	Utilizator Mihai	uzmihai2005@gmail.com	Rares Mihai
2	RaresMihai05	uzmihai2005@gmail.com	Rares Mihai
3	Darian Stefan	carsteastef@yahoo.com	Carstea Stefan
4	Carstea89	carsteastef@yahoo.com	Carstea Stefan

Varianta FN-3:

CONT_SITE		
id_cont#	nume_utilizator#	email
1	Utilizator Mihai	uzmihai2005@gmail.com
2	RaresMihai05	uzmihai2005@gmail.com
3	Darian Stefan	carsteastef@yahoo.com
4	Carstea89	carsteastef@yahoo.com

CONT_ADRESA_EMAIL	
email	nume_utilizator_email
uzmihai2005@gmail.com	Rares Mihai
carsteastef@yahoo.com	Carstea Stefan

- Relația inițială (cea de la „Varianta non-FN3”) se află deja în FN2, deoarece toate atributele care nu fac parte din cheie, depind de întreaga cheie.
- Relația inițială nu se află în FN3, deoarece atributul „nume_utilizator_email” depinde indirect de cheia primară, prin intermediul atributului „email”, existând dependențele:
 - {id_cont#, nume_utilizator#} -> {email}
 - {id_cont#, nume_utilizator#} -> {email} -> {nume_utilizator_email}
- Pentru a aduce relația inițială în FN3, am spart tabelul „CONT_SITE” într-un nou tabel „CONT_SITE”, și în tabelul „CONT_ADRESA_EMAIL”, după care am aplicat regula Casey-Delobel, și am eliminat dependențele funcționale tranzitive, obținând proiecțiile:
 - CONT_SITE(id_cont#, nume_utilizator#, email)
 - CONT_EMAIL(email, nume_utilizator_email)

10. Crearea tabelelor în SQL și inserarea de date coerente în fiecare dintre acestea:

-- Crearea tabelelor:

```
CREATE TABLE CATEGORIE
```

```
(id_categorie INTEGER,
```



```
denumire VARCHAR2(40) CONSTRAINT denumire_categorie NOT NULL,  
CONSTRAINT pk_categorie PRIMARY KEY(id_categorie),  
CONSTRAINT u_denumire_categorie UNIQUE(denumire)  
);
```

CREATE TABLE PRODUS

```
(id_produs INTEGER,  
id_categorie INTEGER,  
tip VARCHAR(10) CONSTRAINT tip_produs NOT NULL,  
denumire VARCHAR2(200) CONSTRAINT denumire_produs NOT NULL,  
tip_sistem VARCHAR2(30),  
vanzator VARCHAR2(40) CONSTRAINT vanzator_produs NOT NULL,  
recomandat_pentru_gaming VARCHAR2(2),  
pret NUMBER CONSTRAINT pret_produs NOT NULL,  
average_rating NUMBER,  
CONSTRAINT pk_produs PRIMARY KEY(id_produs),  
CONSTRAINT fk_categorie_in_produs FOREIGN KEY(id_categorie) REFERENCES  
CATEGORIE(id_categorie) ON DELETE CASCADE,  
CONSTRAINT u_denumire_produs UNIQUE(denumire)  
);
```

CREATE TABLE SERVICIU

```
(id_serviciu INTEGER,  
tip VARCHAR(13) CONSTRAINT tip_serviciu NOT NULL,  
denumire VARCHAR2(70) CONSTRAINT denumire_serviciu NOT NULL,  
descriere VARCHAR2(1000),  
pret NUMBER CONSTRAINT pret_serviciu NOT NULL,
```

```
CONSTRAINT pk_serviciu PRIMARY KEY(id_serviciu),  
CONSTRAINT u_denumire_serviciu UNIQUE(denumire)  
);
```

```
CREATE TABLE ADRESA_LIVRARE
```

```
(id_adresa INTEGER,  
localitate VARCHAR2(30) CONSTRAINT localitate_adresa_livrare NOT NULL,  
strada VARCHAR2(80) CONSTRAINT strada_adresa_livrare NOT NULL,  
nr_strada INTEGER CONSTRAINT nr_strada_adresa_livrare NOT NULL,  
CONSTRAINT pk_adresa_livrare PRIMARY KEY(id_adresa)  
);
```

```
CREATE TABLE CLIENT
```

```
(id_client INTEGER,  
nume VARCHAR2(40) CONSTRAINT nume_client NOT NULL,  
prenume VARCHAR2(40) CONSTRAINT prenume_client NOT NULL,  
CONSTRAINT pk_client PRIMARY KEY(id_client)  
);
```

```
CREATE TABLE CONT_SITE
```

```
(id_cont INTEGER,  
nume_utilizator VARCHAR2(40),  
id_client INTEGER,  
email VARCHAR2(40) CONSTRAINT email NOT NULL,  
CONSTRAINT pk_compus_cont_site PRIMARY KEY(id_cont, nume_utilizator),  
CONSTRAINT u_nume_utilizator UNIQUE(nume_utilizator),
```

```
CONSTRAINT fk_client_in_cont_site FOREIGN KEY(id_client) REFERENCES CLIENT(id_client)  
ON DELETE CASCADE
```

```
);
```

```
CREATE TABLE RECENZIE
```

```
(id_recenzie INTEGER,
```

```
id_produs INTEGER CONSTRAINT id_produs_recenzie NOT NULL,
```

```
nume_utilizator VARCHAR2(40) CONSTRAINT nume_utilizator_recenzie NOT NULL,
```

```
data_scriere DATE CONSTRAINT data_scriere_recenzie NOT NULL,
```

```
nr_stele NUMBER,
```

```
CONSTRAINT pk_compus_recenzie PRIMARY KEY(id_recenzie, id_produs),
```

```
CONSTRAINT fk_produs_in_recenzie FOREIGN KEY(id_produs) REFERENCES  
PRODUS(id_produs) ON DELETE CASCADE,
```

```
CONSTRAINT fk_cont_site_in_recenzie FOREIGN KEY(nume_utilizator) REFERENCES  
CONT_SITE(nume_utilizator) ON DELETE CASCADE
```

```
);
```

```
CREATE TABLE DEPARTAMENT
```

```
(id_departament INTEGER,
```

```
denumire VARCHAR2(30) CONSTRAINT denumire_departament NOT NULL,
```

```
CONSTRAINT pk_departament PRIMARY KEY(id_departament),
```

```
CONSTRAINT u_denumire_departament UNIQUE(denumire)
```

```
);
```

```
CREATE TABLE ANGAJAT
```

```
(id_angajat INTEGER,
```

```
id_departament INTEGER,
```

```
tip VARCHAR2(9) CONSTRAINT tip_angajat NOT NULL,  
nume VARCHAR2(30) CONSTRAINT nume_angajat NOT NULL,  
prenume VARCHAR2(30) CONSTRAINT prenume_angajat NOT NULL,  
salariu NUMBER CONSTRAINT salariu_angajat NOT NULL,  
email VARCHAR2(40) CONSTRAINT email_angajat NOT NULL,  
nr_telefon VARCHAR2(10) CONSTRAINT nr_telefon_angajat NOT NULL,  
CONSTRAINT pk_angajat PRIMARY KEY(id_angajat),  
CONSTRAINT fk_departament_in_angajat FOREIGN KEY(id_departament) REFERENCES  
DEPARTAMENT(id_departament) ON DELETE CASCADE,  
CONSTRAINT u_email_angajat UNIQUE(email)  
);
```

```
CREATE TABLE ESTE_COMPATIBIL
```

```
(id_serviciu INTEGER,  
id_produs INTEGER,  
CONSTRAINT pk_compus_este_compatibil PRIMARY KEY(id_serviciu, id_produs),  
CONSTRAINT fk_serviciu_in_este_compatibil FOREIGN KEY(id_serviciu) REFERENCES  
SERVICIU(id_serviciu) ON DELETE CASCADE,  
CONSTRAINT fk_produs_in_este_compatibil FOREIGN KEY(id_produs) REFERENCES  
PRODUS(id_produs) ON DELETE CASCADE  
);
```

```
CREATE TABLE COMANDA
```

```
(id_comanda INTEGER,  
id_produs INTEGER,  
id_client INTEGER,  
id_adresa INTEGER,
```

```
data_sosire DATE CONSTRAINT data_sosire_comanda NOT NULL,  
pret_total NUMBER CONSTRAINT pret_total_comanda NOT NULL,  
CONSTRAINT pk_compus_comanda PRIMARY KEY(id_comanda, id_produs, id_client,  
id_adresa),  
CONSTRAINT fk_produs_in_comanda FOREIGN KEY(id_produs) REFERENCES  
PRODUS(id_produs) ON DELETE CASCADE,  
CONSTRAINT fk_client_in_comanda FOREIGN KEY(id_client) REFERENCES CLIENT(id_client)  
ON DELETE CASCADE,  
CONSTRAINT fk_adresa_livrare_in_comanda FOREIGN KEY(id_adresa) REFERENCES  
ADRESA_LIVRARE(id_adresa) ON DELETE CASCADE  
);
```

```
CREATE TABLE CUMPARA
```

```
(id_cumpara INTEGER,  
id_serviciu INTEGER,  
id_client INTEGER,  
data_realizare DATE CONSTRAINT data_realizare_cumpara NOT NULL,  
CONSTRAINT pk_compus_cumpara PRIMARY KEY(id_cumpara, id_serviciu, id_client),  
CONSTRAINT fk_serviciu_in_cumpara FOREIGN KEY(id_serviciu) REFERENCES  
SERVICIU(id_serviciu) ON DELETE CASCADE,  
CONSTRAINT fk_client_in_cumpara FOREIGN KEY(id_client) REFERENCES CLIENT(id_client)  
ON DELETE CASCADE  
);
```

```
CREATE TABLE ESTE_ASISTAT
```

```
(id_este_asistat INTEGER,  
id_client INTEGER,  
id_angajat INTEGER,  
CONSTRAINT pk_compus_este_asistat PRIMARY KEY(id_este_asistat, id_client, id_angajat),
```

```
        CONSTRAINT fk_client_in_este_asistat FOREIGN KEY(id_client) REFERENCES
CLIENT(id_client) ON DELETE CASCADE,

        CONSTRAINT fk_angajat_in_este_asistat FOREIGN KEY(id_angajat) REFERENCES
ANGAJAT(id_angajat) ON DELETE CASCADE

    );
```

-- Inserarea datelor in tabelele create:

-- DROP SEQUENCE SEQ_CATEGORIE;

-- Secventa pentru inserarea inregistrarilor in tabelul "CATEGORIE".

```
CREATE SEQUENCE SEQ_CATEGORIE
```

```
INCREMENT by 1
```

```
START WITH 0
```

```
MINVALUE -1
```

```
MAXVALUE 399
```

```
NOCYCLE;
```

```
INSERT INTO CATEGORIE
```

```
VALUES(SEQ_CATEGORIE.NEXTVAL, 'Placi video');
```

```
INSERT INTO CATEGORIE
```

```
VALUES(SEQ_CATEGORIE.NEXTVAL, 'Procesoare');
```

```
INSERT INTO CATEGORIE
```

```
VALUES(SEQ_CATEGORIE.NEXTVAL, 'Memorii');
```

```
INSERT INTO CATEGORIE
```

```
VALUES(SEQ_CATEGORIE.NEXTVAL, 'HDD');
```

```
INSERT INTO CATEGORIE
```

```
VALUES(SEQ_CATEGORIE.NEXTVAL, 'Adaptoare');
```

```
INSERT INTO CATEGORIE
```

```
VALUES(SEQ_CATEGORIE.NEXTVAL, 'PC-uri');
```

```
INSERT INTO CATEGORIE
```

```
VALUES(SEQ_CATEGORIE.NEXTVAL, 'Laptop-uri');
```

```
-- ROLLBACK;
```

```
-- COMMIT;
```

```
-- DROP SEQUENCE SEQ_PRODUS;
```

```
-- Secventa pentru inserarea inregistrarilor in tabelul "PRODUS".
```

```
CREATE SEQUENCE SEQ_PRODUS
```

```
INCREMENT by 5
```

```
START WITH 0
```

```
MINVALUE -1
```

```
NOCYCLE;
```

```
INSERT INTO PRODUS
```

```
VALUES(SEQ_PRODUS.NEXTVAL, 1, 'Componenta', 'Placa video Palit GeForce RTX 3090  
GamingPro 24GB GDDR6X 384-bit', NULL, 'GIGABYTE', 'Da', 14999.99, 5);
```

```
INSERT INTO PRODUS
```

```
VALUES(SEQ_PRODUS.NEXTVAL, 2, 'Componenta', 'Procesor AMD Ryzen 5 3600 3.6GHz box',  
NULL, 'AMD', 'Da', 999.99, 5);
```

```
INSERT INTO PRODUS
```

```
VALUES(SEQ_PRODUS.NEXTVAL, 5, 'Accesoriu', 'Adaptor Gembird 1x HDMI 1.4 Male - 1x VGA Female', NULL, 'Gembird', 'Nu', 35.99, NULL);
```

```
INSERT INTO PRODUS
```

```
VALUES(SEQ_PRODUS.NEXTVAL, 6, 'Sistem', 'PC Gaming Raptor5, Intel i5-9400F 2.9GHz Coffee Lake, 16GB DDR4, 960GB SSD, RX 5600 XT 6GB GDDR6, Iluminare RGB', 'PC', 'Raptor5', 'Da', 4799.99, 4);
```

```
INSERT INTO PRODUS
```

```
VALUES(SEQ_PRODUS.NEXTVAL, 7, 'Sistem', 'Laptop Lenovo 15.6" ThinkPad E15 Gen 2, FHD, Procesor Intel® Core™ i5-1135G7 (8M Cache, up to 4.20 GHz), 8GB DDR4, 256GB SSD, Intel Iris Xe, No OS, Black', 'Lenovo', 'Laptop', 'Nu', 3398.99, 5);
```

```
INSERT INTO PRODUS
```

```
VALUES(SEQ_PRODUS.NEXTVAL, 1, 'Componenta', 'Placa video PowerColor Radeon RX 6900 XT Red Devil 16GB GDDR6 256-bit', NULL, 'Red Devil', 'Da', 11999.99, NULL);
```

```
INSERT INTO PRODUS
```

```
VALUES(SEQ_PRODUS.NEXTVAL, 3, 'Componenta', 'Memorie Corsair Vengeance LPX Black 16GB DDR4 3200MHz CL16 Dual Channel Kit', NULL, 'Corsair', 'Da', 514.99, NULL);
```

```
INSERT INTO PRODUS
```

```
VALUES(SEQ_PRODUS.NEXTVAL, 4, 'Componenta', 'Hard disk Seagate BarraCuda 2TB SATA-III 7200RPM 256MB', NULL, 'Seagate', 'Da', 262.99, NULL);
```

```
-- ROLLBACK;
```

```
-- COMMIT;
```

```
-- DROP SEQUENCE SEQ_SERVICIU;
```

```
-- Secventa pentru inserarea inregistrarilor in tabelul "SERVICIU".
```

```
CREATE SEQUENCE SEQ_SERVICIU
```

```
INCREMENT by 1
```

```
START WITH 0
```


MINVALUE -1

NOCYCLE;

INSERT INTO SERVICIU

VALUES(SEQ_SERVICIU.NEXTVAL, 'Asamblare', 'Serviciu Asamblare Standard',

'Asamblare sistem de calcul desktop de catre un specialist calificat.

Instalare sistem de operare test. Verificare compatibilitate componente.

Preinstalare sisteme de operare si aplicatii (daca s-au achizitionat)',

159);

INSERT INTO SERVICIU

VALUES(SEQ_SERVICIU.NEXTVAL, 'Testare', 'Serviciu Testare Produs',

'Verificare functionalitate produs.

Aplicabil oricarui produs comercializat.',

135.99);

INSERT INTO SERVICIU

VALUES(SEQ_SERVICIU.NEXTVAL, 'Montare', 'Serviciu Montare Componenta',

'Montarea diverselor componente in sistemul de calcul: placa video,

procesor, RAM, HDD, cooler pentru procesor, etc.',

59.99);

INSERT INTO SERVICIU

VALUES(SEQ_SERVICIU.NEXTVAL, 'Diagnosticare', 'Serviciu Diagnosticare Produs',

'Identificarea problemelor de functionare si furnizarea de recomandari.',

119.99);

INSERT INTO SERVICIU

VALUES(SEQ_SERVICIU.NEXTVAL, 'Asamblare', 'Serviciu Asamblare Premium',

'Asamblare si testare sistem de calcul desktop de catre un specialist calificat.

Cooling management. Wire management. Instalare sistem de operare test.

Verificare compatibilitate componente. Testare sistem (rulare in conditii de stres 24 ore).

Generarea unui raport operatiune asamblare premium. Updateuri la ultimele versiuni stabile (BIOS placa de baza).

Preinstalare sisteme de operare si aplicatii (daca s-au achizitionat).',

249.99);

-- ROLLBACK;

-- COMMIT;

INSERT INTO ADRESA_LIVRARE

VALUES(1, 'Bucuresti', 'Mihai Eminescu', 1);

INSERT INTO ADRESA_LIVRARE

VALUES(2, 'Iasi', 'Morii', 12);

INSERT INTO ADRESA_LIVRARE

VALUES(3, 'Bucuresti', 'Unirii', 34);

INSERT INTO ADRESA_LIVRARE

VALUES(4, 'Constanta', 'Stejarul mic', 2);

INSERT INTO ADRESA_LIVRARE

VALUES(5, 'Bucuresti', 'Eroilor', 10);

INSERT INTO ADRESA_LIVRARE

VALUES(6, 'Bucuresti', 'Mihail Kogalniceanu', 8);

-- ROLLBACK;

-- COMMIT;

INSERT INTO CLIENT

```
VALUES(1, 'Aurel', 'Popescu-Mihai');  
INSERT INTO CLIENT  
VALUES(2, 'Carstea', 'Darian-Stefan');  
INSERT INTO CLIENT  
VALUES(3, 'Castan', 'Mirela');  
INSERT INTO CLIENT  
VALUES(4, 'Popa', 'Marcel-Radu');  
INSERT INTO CLIENT  
VALUES(5, 'Manole', 'Iuliana-Elena');  
INSERT INTO CLIENT  
VALUES(6, 'Florian', 'Andrei-Cosmin');
```

```
-- ROLLBACK;  
-- COMMIT;
```

```
INSERT INTO CONT_SITE  
VALUES(1, 'Utilizator Mihai', NULL, 'uzmihai2005@gmail.com');  
INSERT INTO CONT_SITE  
VALUES(2, 'Darian Stefan', 2, 'carsteastef@yahoo.com');  
INSERT INTO CONT_SITE  
VALUES(3, 'Mirela Castan', 3, 'mirela_c@gmail.com');  
INSERT INTO CONT_SITE  
VALUES(4, 'ANTON PAVEL MIHAI', NULL, 'apav_mihai@gmail.com');  
INSERT INTO CONT_SITE  
VALUES(5, 'Elena 2008', 5, 'elenamanole@yahoo.com');
```

```
-- ROLLBACK;
```

```
-- COMMIT;
```

```
INSERT INTO RECENZIE
```

```
VALUES(1, 5, 'Utilizator Mihai', TO_DATE('15-01-2021','dd-mm-yyyy'), 5);
```

```
INSERT INTO RECENZIE
```

```
VALUES(2, 10, 'Utilizator Mihai', TO_DATE('15-01-2021','dd-mm-yyyy'), 5);
```

```
INSERT INTO RECENZIE
```

```
VALUES(3, 20, 'Darian Stefan', TO_DATE('23-05-2020','dd-mm-yyyy'), 4);
```

```
INSERT INTO RECENZIE
```

```
VALUES(4, 25, 'Mirela Castan', TO_DATE('02-08-2019','dd-mm-yyyy'), 5);
```

```
INSERT INTO RECENZIE
```

```
VALUES(5, 25, 'Elena 2008', TO_DATE('11-09-2020','dd-mm-yyyy'), 5);
```

```
INSERT INTO RECENZIE
```

```
VALUES(6, 20, 'Darian Stefan', TO_DATE('23-05-2020','dd-mm-yyyy'), 4);
```

```
-- ROLLBACK;
```

```
-- COMMIT;
```

```
INSERT INTO DEPARTAMENT
```

```
VALUES(1, 'Relationare clienti');
```

```
INSERT INTO DEPARTAMENT
```

```
VALUES(2, 'Prestare servicii');
```

```
INSERT INTO DEPARTAMENT
```

```
VALUES(3, 'Intretinere website');
```

```
INSERT INTO DEPARTAMENT
```

```
VALUES(4, 'Inventar produse');
```

```
INSERT INTO DEPARTAMENT
```

```
VALUES(5, 'Securitate');
```

```
-- ROLLBACK;
```

```
-- COMMIT;
```

```
INSERT INTO ANGAJAT
```

```
VALUES(1, 2, 'Tehnician', 'Aurel', 'Tudor-Mihai', 7000, 'aurel_mihai@gmail.com', '0793333333');
```

```
INSERT INTO ANGAJAT
```

```
VALUES(2, 1, 'Tehnician', 'Mihail', 'Maria-Andreea', 4500, 'mariaandreea09@yahoo.com',  
'0791111111');
```

```
INSERT INTO ANGAJAT
```

```
VALUES(3, 1, 'Tehnician', 'Vladoi', 'Rares', 4000, 'raresVld@gmail.com', '0749999999');
```

```
INSERT INTO ANGAJAT
```

```
VALUES(4, 4, 'Tehnician', 'Grigorescu', 'Stefan', 4500, 'grgrsc_stf@yahoo.com', '0755555555');
```

```
INSERT INTO ANGAJAT
```

```
VALUES(5, 5, 'Paznic', 'Marian', 'Alexandru', 3300, 'marianalex3@yahoo.com', '0788888885');
```

```
-- ROLLBACK;
```

```
-- COMMIT;
```

```
INSERT INTO ESTE_COMPATIBIL
```

```
VALUES(2, 5);
```

```
INSERT INTO ESTE_COMPATIBIL
```

```
VALUES(2, 10);
```

```
INSERT INTO ESTE_COMPATIBIL
```

```
VALUES(2, 20);  
INSERT INTO ESTE_COMPATIBIL  
VALUES(2, 25);  
INSERT INTO ESTE_COMPATIBIL  
VALUES(2, 30);  
INSERT INTO ESTE_COMPATIBIL  
VALUES(4, 20);  
INSERT INTO ESTE_COMPATIBIL  
VALUES(4, 25);  
INSERT INTO ESTE_COMPATIBIL  
VALUES(3, 5);  
INSERT INTO ESTE_COMPATIBIL  
VALUES(3, 10);  
INSERT INTO ESTE_COMPATIBIL  
VALUES(3, 30);
```

```
-- ROLLBACK;  
-- COMMIT;
```

```
INSERT INTO COMANDA  
VALUES(1, 5, 1, 1, TO_DATE('20-09-2020','dd-mm-yyyy'), 14999.99);  
INSERT INTO COMANDA  
VALUES(2, 10, 1, 1, TO_DATE('20-09-2020','dd-mm-yyyy'), 999.99);  
INSERT INTO COMANDA  
VALUES(3, 20, 2, 3, TO_DATE('07-12-2019','dd-mm-yyyy'), 4799.99);  
INSERT INTO COMANDA
```

```
VALUES(4, 25, 3, 4, TO_DATE('16-02-2019','dd-mm-yyyy'), 3398.99);  
INSERT INTO COMANDA  
VALUES(5, 25, 5, 2, TO_DATE('11-09-2020','dd-mm-yyyy'), 3398.99);  
INSERT INTO COMANDA  
VALUES(6, 30, 4, 5, TO_DATE('07-02-2021','dd-mm-yyyy'), 11999.99);  
INSERT INTO COMANDA  
VALUES(7, 15, 1, 5, TO_DATE('07-02-2021','dd-mm-yyyy'), 71.98);  
INSERT INTO COMANDA  
VALUES(8, 10, 4, 3, TO_DATE('07-02-2021','dd-mm-yyyy'), 999.99);  
INSERT INTO COMANDA  
VALUES(9, 15, 2, 3, TO_DATE('10-12-2019','dd-mm-yyyy'), 35.99);  
INSERT INTO COMANDA  
VALUES(10, 20, 6, 6, TO_DATE('20-05-2021','dd-mm-yyyy'), 4799.99);  
  
-- ROLLBACK;  
  
-- COMMIT;
```

```
INSERT INTO CUMPARA  
VALUES(1, 2, 1, TO_DATE('20-09-2020','dd-mm-yyyy'));  
INSERT INTO CUMPARA  
VALUES(2, 5, 1, TO_DATE('20-09-2020','dd-mm-yyyy'));  
INSERT INTO CUMPARA  
VALUES(3, 2, 6, TO_DATE('14-04-2020','dd-mm-yyyy'));  
INSERT INTO CUMPARA  
VALUES(4, 3, 6, TO_DATE('14-04-2020','dd-mm-yyyy'));  
INSERT INTO CUMPARA
```

```
VALUES(5, 2, 3, TO_DATE('16-02-2019','dd-mm-yyyy'));
INSERT INTO CUMPARA
VALUES(6, 4, 3, TO_DATE('16-02-2019','dd-mm-yyyy'));
INSERT INTO CUMPARA
VALUES(7, 2, 5, TO_DATE('11-09-2020','dd-mm-yyyy'));
INSERT INTO CUMPARA
VALUES(8, 4, 5, TO_DATE('11-09-2020','dd-mm-yyyy'));
INSERT INTO CUMPARA
VALUES(9, 2, 2, TO_DATE('07-12-2019','dd-mm-yyyy'));
INSERT INTO CUMPARA
VALUES(10, 2, 4, TO_DATE('07-02-2021','dd-mm-yyyy'));
INSERT INTO CUMPARA
VALUES(11, 3, 4, TO_DATE('08-02-2021','dd-mm-yyyy'));

-- ROLLBACK;

-- COMMIT;
```

```
INSERT INTO ESTE_ASISTAT
VALUES(1, 1, 1);
INSERT INTO ESTE_ASISTAT
VALUES(2, 1, 2);
INSERT INTO ESTE_ASISTAT
VALUES(3, 2, 1);
INSERT INTO ESTE_ASISTAT
VALUES(4, 2, 2);
INSERT INTO ESTE_ASISTAT
```



```
VALUES(5, 3, 1);
INSERT INTO ESTE_ASISTAT
VALUES(6, 3, 2);
INSERT INTO ESTE_ASISTAT
VALUES(7, 5, 1);
INSERT INTO ESTE_ASISTAT
VALUES(8, 5, 2);
INSERT INTO ESTE_ASISTAT
VALUES(9, 4, 3);
INSERT INTO ESTE_ASISTAT
VALUES(10, 4, 2);

-- ROLLBACK;
COMMIT;
```

11. Formulați în limbaj natural și implementați 5 cereri SQL complexe:

-- Cererea 1:

-- Sa se afiseze numele vanzatorului si pretul tuturor produselor al caror rating
-- are o valoare egala 5, si care au fost cumparate de catre un client al carui nume
-- este Aurel, care detine o adresa de livrare in localitatea Bucuresti. Rezultatele
-- se vor ordona in ordine descrescatoare dupa pretul produselor.

```
SELECT vanzator, pret
FROM PRODUS p JOIN COMANDA co ON (p.id_produs = co.id_produs)
      JOIN CLIENT cl ON (co.id_client = cl.id_client)
```

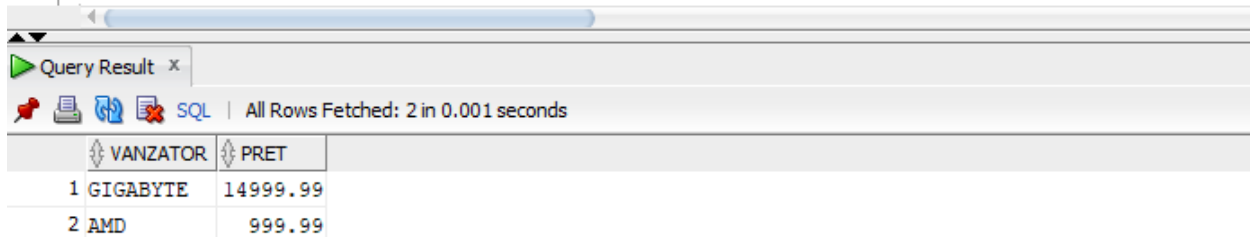
JOIN ADRESA_LIVRARE a_l ON (co.id_adresa = a_l.id_adresa)

WHERE (average_rating = 5) AND (INITCAP(numa) = 'Aurel') AND (a_l.localitate = 'Bucuresti')

ORDER BY pret DESC;

```
428 -- Cererea 1:
429 -- Sa se afiseze numele vanzatorului si pretul tuturor produselor al caror rating
430 -- are o valoare egala 5, si care au fost cumparate de catre un client al carui nume
431 -- este Aurel, care detine o adresa de livrare in localitatea Bucuresti. Rezultatele
432 -- se vor ordona in ordine descrescatoare dupa pretul produselor.
433
```

```
434 SELECT vanzator, pret
435 FROM PRODUS p JOIN COMANDA co ON (p.id_produs = co.id_produs)
436      JOIN CLIENT cl ON (co.id_client = cl.id_client)
437      JOIN ADRESA_LIVRARE a_l ON (co.id_adresa = a_l.id_adresa)
438 WHERE (average_rating = 5) AND (INITCAP(numa) = 'Aurel') AND (a_l.localitate = 'Bucuresti')
439 ORDER BY pret DESC;
440
```



	VANZATOR	PRET
1	GIGABYTE	14999.99
2	AMD	999.99

-- Cererea 2:

-- Sa se afiseze numele intreg al clientilor carora li s-a realizat un serviciu de la

-- inceputul anului si pana in prezent, alaturi de data la care s-au realizat serviciile

-- respective, unde serviciul respectiv este compatibil cu un produs recomandat pentru gaming.

-- Rezultatele se vor ordona in ordine crescatoare dupa numele intreg al clientilor.

SELECT CONCAT(numa, CONCAT(' ', prenuma)) NUME_INTREG_CLIENT, data_realizare

FROM CLIENT cl JOIN CUMPARA cu ON (cl.id_client = cu.id_client)

WHERE cl.id_client IN (SELECT id_client

FROM CUMPARA, DUAL

```

WHERE data_realizare >= TRUNC(SYSDATE, 'YEAR')

AND id_serviciu IN (SELECT id_serviciu

FROM SERVICIU

WHERE id_serviciu IN (SELECT id_serviciu

FROM ESTE_COMPATIBIL

WHERE id_produs IN (SELECT id_produs

FROM PRODUS

WHERE UPPER(recomandat_pentru_gaming) =

'DA'))))

ORDER BY CONCAT(nume, CONCAT(' ', prenume));

```

```

442 -- Cererea 2:
443 -- Sa se afiseze numele intreg al clientilor carora li s-a realizat un serviciu de la
444 -- inceputul anului si pana in prezent, alaturi de data la care s-au realizat serviciile
445 -- respective, unde serviciul respectiv este compatibil cu un produs recomandat pentru gaming.
446 -- Rezultatele se vor ordona in ordine crescatoare dupa numele intreg al clientilor.
447
448 SELECT CONCAT(nume, CONCAT(' ', prenume)) NUME_INTREG_CLIENT, data_realizare
449 FROM CLIENT cl JOIN CUMPARA cu ON(cl.id_client = cu.id_client)
450 WHERE cl.id_client IN (SELECT id_client
451 FROM CUMPARA, DUAL
452 WHERE data_realizare >= TRUNC(SYSDATE, 'YEAR')
453 AND id_serviciu IN (SELECT id_serviciu
454 FROM SERVICIU
455 WHERE id_serviciu IN (SELECT id_serviciu
456 FROM ESTE_COMPATIBIL
457 WHERE id_produs IN (SELECT id_produs
458 FROM PRODUS
459 WHERE UPPER(recomandat_pentru_gaming) = 'DA'))))
460 ORDER BY CONCAT(nume, CONCAT(' ', prenume));

```

Query Result x

All Rows Fetched: 2 in 0.003 seconds

	NUME_INTREG_CLIENT	DATA_REALIZARE
1	Popa Marcel-Radu	08-FEB-21
2	Popa Marcel-Radu	07-FEB-21

-- Cererea 3:

-- Sa se afiseze numele departamentului, suma salariilor tuturor angajatilor,

-- si numarul de angajati, pentru fiecare departament in care lucreaza cel putin

-- un angajat. Se va restrictiona afisarea doar la acele departamente pentru care
-- suma tuturor salariilor este mai mare sau egala cu 4000 RON.

```
WITH tabel_aux AS (SELECT denumire, SUM(salariu) AS suma_salarii, COUNT(id_angajat) AS  
nr_angajati
```

```
FROM DEPARTAMENT d join ANGAJAT a ON (d.id_departament = a.id_departament)
```

```
GROUP BY denumire
```

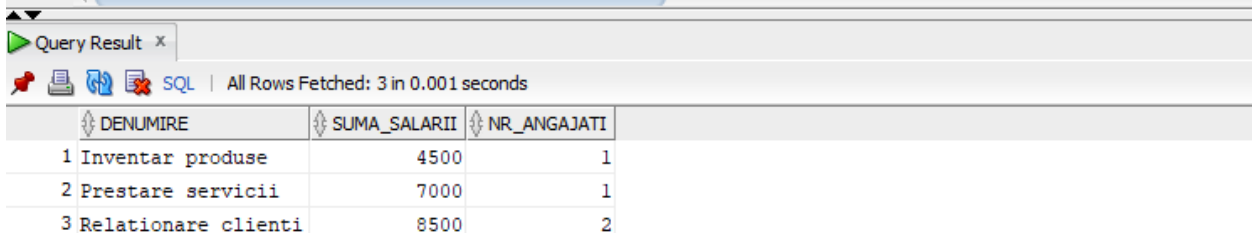
```
HAVING SUM(salariu) >= 4000)
```

```
SELECT denumire, suma_salarii, nr_angajati
```

```
FROM tabel_aux
```

```
ORDER BY denumire;
```

```
463 -- Cererea 3:  
464 -- Sa se afiseze numele departamentului, suma salariilor tuturor angajatilor, si numarul  
465 -- de angajati, pentru fiecare departament in care lucreaza cel putin un angajat. Se va  
466 -- restrictiona afisarea doar la acele departamente pentru care suma tuturor salariilor  
467 -- este mai mare sau egala cu 4000 RON.  
468  
469 WITH tabel_aux AS (SELECT denumire, SUM(salariu) AS suma_salarii, COUNT(id_angajat) AS nr_angajati  
470 FROM DEPARTAMENT d join ANGAJAT a ON (d.id_departament = a.id_departament)  
471 GROUP BY denumire  
472 HAVING SUM(salariu) >= 4000)  
473  
474 SELECT denumire, suma_salarii, nr_angajati  
475 FROM tabel_aux  
476 ORDER BY denumire;
```



	DENUMIRE	SUMA_SALARII	NR_ANGAJATI
1	Inventar produse	4500	1
2	Prestare servicii	7000	1
3	Relationare clienti	8500	2

-- Cererea 4:

-- Sa se afiseze numele de utilizator, alaturi de id-ul de client, pentru conturile
-- care au scris recenzii pentru produse, avand un numar de stele egal cu rating-ul mediu
-- al produselor respective. In cazul in care un anumit cont nu este detinut de catre un
-- client cunoscut, se va afisa, in locul id-ului de client, mesajul "Contul nu apartine
-- unui client cunoscut."

```
SELECT nume_utilizator, DECODE(TO_CHAR(id_client), NULL, 'Contul nu apartine unui client  
cunoscut.', TO_CHAR(id_client)) ID_CLIENT
```

```
FROM CONT_SITE
```

```
WHERE nume_utilizator IN (SELECT nume_utilizator
```

```
FROM RECENZIE
```

```
WHERE id_produs IN (SELECT id_produs
```

```
FROM PRODUS
```

```
WHERE nr_stele = average_rating));
```

```
479 -- Cererea 4:  
480 -- Sa se afiseze numele de utilizator, alaturi de id-ul de client, pentru conturile  
481 -- care au scris recenzii pentru produse, avand un numar de stele egal cu rating-ul mediu  
482 -- al produselor respective. In cazul in care un anumit cont nu este detinut de catre un  
483 -- client cunoscut, se va afisa, in locul id-ului de client, mesajul "Contul nu apartine  
484 -- unui client cunoscut."  
485  
486 SELECT nume_utilizator, DECODE(TO_CHAR(id_client), NULL, 'Contul nu apartine unui client cunoscut.', TO_CHAR(id_client)) ID_CLIENT  
487 FROM CONT_SITE  
488 WHERE nume_utilizator IN (SELECT nume_utilizator  
489 FROM RECENZIE  
490 WHERE id_produs IN (SELECT id_produs  
491 FROM PRODUS  
492 WHERE nr_stele = average_rating));
```

Query Result x

All Rows Fetched: 4 in 0.001 seconds

NUME_UTILIZATOR	ID_CLIENT
1 Utilizator Mihai	Contul nu apartine unui client cunoscut.
2 Darian Stefan	2
3 Mirela Castan	3
4 Elena 2008	5

-- Cererea 5:

-- Sa se afiseze denumirea, alaturi de numele vanzatorului, pentru produsele al caror

-- pret depaseste pretul mediu al produselor din aceeasi categorie cu acestea. De asemenea,

-- pentru fiecare produs, se va afisa si mesajul "Produsul este profitabil.", in cazul in

-- care produsul are un rating mai mare decat 3.5, respectiv mesajul "Nu se poate deduce

statusul

-- produsului.", in caz contrar. Daca pentru un anumit produs nu se cunoaste numele

-- vanzatorului, se va afisa mesajul "Vanzator necunoscut.".

```
SELECT denumire, NVL(vanzator, 'Vanzator necunoscut.') VANZATOR,
       CASE WHEN average_rating >= 3.5 THEN 'Produsul este profitabil.'
       ELSE 'Nu se poate deduce statusul produsului.' END STATUS_PRODUS
FROM PRODUS p
WHERE pret > (SELECT AVG(pret)
              FROM PRODUS
              WHERE p.id_categorie = id_categorie);
```

```
495 -- Cererea 5:
496 -- Sa se afiseze denumirea, alaturi de numele vanzatorului, pentru produsele al caror
497 -- pret depaseste pretul mediu al produselor din aceeasi categorie cu acestea. De asemenea,
498 -- pentru fiecare produs, se va afisa si mesajul "Produsul este profitabil.", in cazul in
499 -- care produsul are un rating mai mare decat 3.5, respectiv mesajul "Nu se poate deduce statusul
500 -- produsului.", in caz contrar. Daca pentru un anumit produs nu se cunoaste numele
501 -- vanzatorului, se va afisa mesajul "Vanzator necunoscut.".
```

```
502
503 SELECT denumire, NVL(vanzator, 'Vanzator necunoscut.') VANZATOR,
504        CASE WHEN average_rating >= 3.5 THEN 'Produsul este profitabil.'
505        ELSE 'Nu se poate deduce statusul produsului.' END STATUS_PRODUS
506 FROM PRODUS p
507 WHERE pret > (SELECT AVG(pret)
508              FROM PRODUS
509              WHERE p.id_categorie = id_categorie);
```

Query Result x		
All Rows Fetched: 1 in 0.002 seconds		
DENUMIRE	VANZATOR	STATUS_PRODUS
1 Placa video Palit GeForce RTX 3090 GamingPro 24GB GDDR6X 384-bit GIGABYTE		Produsul este profitabil.

12. Implementarea a 3 operații de actualizare sau suprimare a datelor utilizând subcereri:

SAVEPOINT sp;

-- Sa se stearga toate produsele care fac parte din categoria cu id-ul 1.

DELETE

FROM PRODUS

WHERE id_categorie IN (SELECT id_categorie

FROM PRODUS

WHERE id_categorie = 1);

```
518 -- Sa se stearga toate produsele care fac parte din categoria cu id-ul 1.
519 DELETE
520 FROM PRODUS
521 WHERE id_categorie IN (SELECT id_categorie
522                        FROM PRODUS
523                        WHERE id_categorie = 1);
524
```

Script Output x

Task completed in 0.027 seconds

2 rows deleted.

524 | SELECT * FROM PRODUS;

Script Output x Query Result x

SQL | All Rows Fetched: 6 in 0.001 seconds

	ID_PRODUS	ID_CATEGORIE	TIP	DENUMIRE
1	10	2	Componenta	Procesor AMD Ry
2	15	5	Accesoriu	Adaptor Gembird
3	20	6	Sistem	PC Gaming Raptc
4	25	7	Sistem	Laptop Lenovo 1
5	35	3	Componenta	Memorie Corsair
6	40	4	Componenta	Hard disk Seaga

-- Sa se stearga toate adresele de livrare la care au fost livrate produse
-- care nu sunt recomandate pentru gaming.

DELETE

FROM ADRESA_LIVRARE

WHERE id_adresa IN (SELECT id_adresa

FROM COMANDA

WHERE id_produc IN (SELECT id_produc

FROM PRODUS

WHERE UPPER(recomandat_pentru_gaming) = 'NU'));

```
525 -- Sa se stearga toate adresele de livrare la care au fost livrate produse
526 -- care nu sunt recomandate pentru gaming.
527 DELETE
528 FROM ADRESA_LIVRARE
529 WHERE id_adresa IN (SELECT id_adresa
530                     FROM COMANDA
531                     WHERE id_produc IN (SELECT id_produc
532                                         FROM PRODUS
533                                         WHERE UPPER(recomandat_pentru_gaming) = 'NU'));
```

Script Output x Query Result x
Task completed in 0.036 seconds

4 rows deleted.

534 SELECT * FROM ADRESA_LIVRARE;

Script Output x Query Result x
All Rows Fetched: 2 in 0.011 seconds

ID_ADRESA	LOCALITATE	STRADA	NR_STRADA
1	1 Bucuresti	Mihai Eminescu	1
2	6 Bucuresti	Mihail Kogalniceanu	8

-- Sa se stearga toate conturile care nu au scris nicio recenzie.

DELETE

FROM CONT_SITE

WHERE nume_utilizator NOT IN (SELECT nume_utilizator
FROM RECENZIE);

ROLLBACK TO SAVEPOINT sp;

```
535 | -- Sa se stearga toate conturile care nu au scris nicio recenzie.  
536 | DELETE  
537 | FROM CONT_SITE  
538 | WHERE nume_utilizator NOT IN (SELECT nume_utilizator  
539 | FROM RECENZIE);
```

Script Output x

Task completed in 0.031 seconds

1 row deleted.

```
540 | SELECT * FROM CONT_SITE;
```

Script Output x Query Result x

SQL | All Rows Fetched: 4 in 0.002 seconds

	ID_CONT	NUME_UTILIZATOR	ID_CLIENT	EMAIL
1	1	Utilizator Mihai	(null)	uzmihai2005@gmail.com
2	2	Darian Stefan	2	carsteastef@yahoo.com
3	3	Mirela Castan	3	mirela_c@gmail.com
4	5	Elena 2008	5	elenamanole@yahoo.com

13. Crearea unei secvențe ce va fi utilizată în inserarea înregistrărilor în tabele:

-- Secventa pentru inserarea inregistrarilor in tabelul "CATEGORIE".

CREATE SEQUENCE SEQ_CATEGORIE

INCREMENT by 1

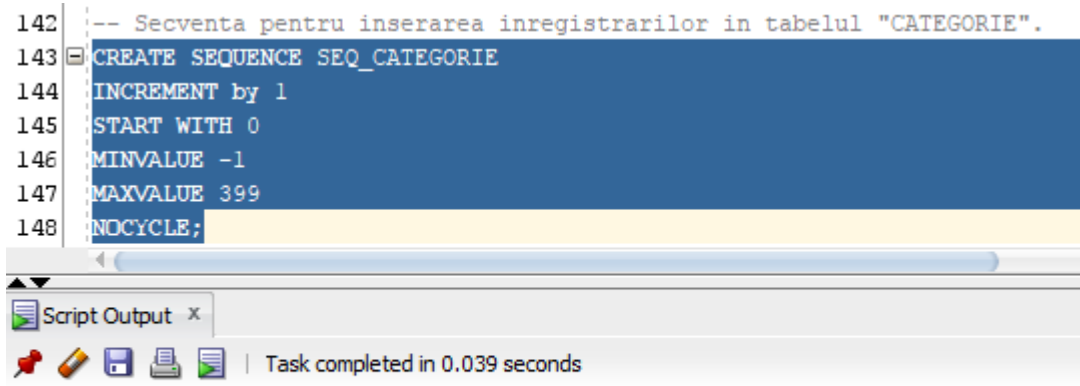
START WITH 0

MINVALUE -1

MAXVALUE 399

NOCYCLE;

```
142 | -- Secventa pentru inserarea inregistrarilor in tabelul "CATEGORIE".
143 | CREATE SEQUENCE SEQ_CATEGORIE
144 | INCREMENT by 1
145 | START WITH 0
146 | MINVALUE -1
147 | MAXVALUE 399
148 | NOCYCLE;
```



Sequence SEQ_CATEGORIE created.

-- Secventa pentru inserarea inregistrarilor in tabelul "PRODUS".

CREATE SEQUENCE SEQ_PRODUS

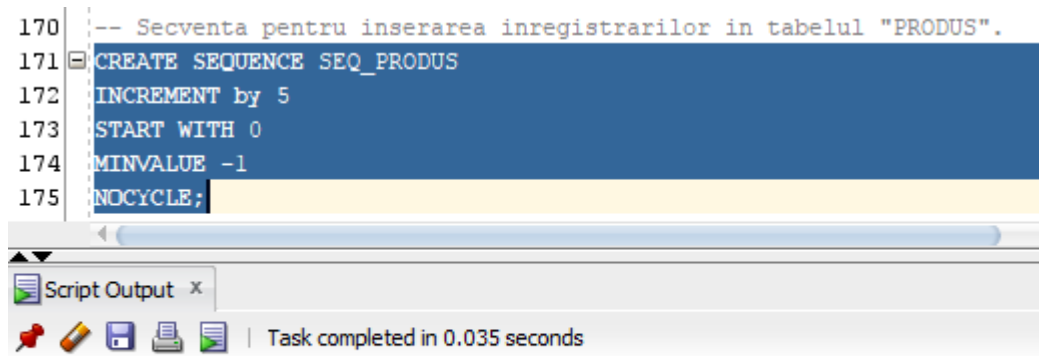
INCREMENT by 5

START WITH 0

MINVALUE -1

NOCYCLE;

```
170 | -- Secventa pentru inserarea inregistrarilor in tabelul "PRODUS".
171 | CREATE SEQUENCE SEQ_PRODUS
172 | INCREMENT by 5
173 | START WITH 0
174 | MINVALUE -1
175 | NOCYCLE;
```



Sequence SEQ_PRODUS created.

-- Secventa pentru inserarea inregistrarilor in tabelul "SERVICIU".

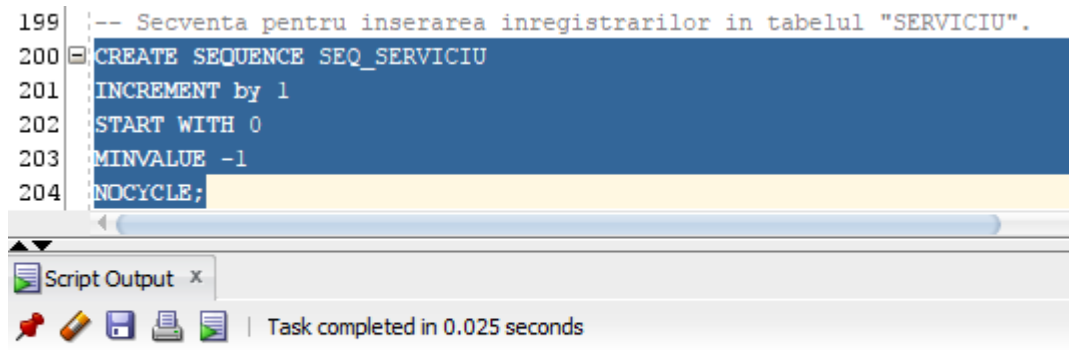
CREATE SEQUENCE SEQ_SERVICIU

INCREMENT by 1

START WITH 0

MINVALUE -1

NOCYCLE;

A screenshot of a SQL script editor window. The script contains the following lines: 199: -- Secventa pentru inserarea inregistrarilor in tabelul "SERVICIU"., 200: CREATE SEQUENCE SEQ_SERVICIU, 201: INCREMENT by 1, 202: START WITH 0, 203: MINVALUE -1, 204: NOCYCLE;. The script is highlighted in blue. Below the script, there is a 'Script Output' window showing the message 'Task completed in 0.025 seconds'.

Sequence SEQ_SERVICIU created.

16. Formulați în limbaj natural și implementați în SQL: o cerere ce utilizează operația outer-join pe minimum 4 tabele și două cereri ce utilizează operația division:

-- Cerere care utilizeaza operatia outer-join pe minimum 4 tabele:

-- Sa se afiseze numele intreg, alaturi de numele de utilizator, al clientilor cunoscuti

-- (adica cei care figureaza in tabelul CLIENT), si numele tuturor produselor carora le-au

-- dat un rating, alaturi de rating-ul respectiv. Daca exista clienti care nu au conuri,

-- se va afisa mesajul "Clientul nu are cont.".

SELECT nume||' '||prenume NUME_INTREG,

CASE WHEN c_s.numa_utilizator IS NOT NULL THEN c_s.numa_utilizator

```

ELSE 'Clientul nu are cont.' END NUME_UTILIZATOR,

denumire, nr_stele

FROM CLIENT cl LEFT OUTER JOIN CONT_SITE c_s ON (cl.id_client = c_s.id_client)

LEFT OUTER JOIN RECENZIE r ON (c_s.ume_utilizator = r.ume_utilizator)

LEFT OUTER JOIN PRODUS p ON (r.id_produs = p.id_produs);

```

```

551 -- Cerere care utilizeaza operatia outer-join pe minimum 4 tabele:
552
553 -- Sa se afiseze numele intreg, alaturi de numele de utilizator, al clientilor cunoscuti
554 -- (adica cei care figureaza in tabelul CLIENT), si numele tuturor produselor carora le-au
555 -- dat un rating, alaturi de rating-ul respectiv. Daca exista clienti care nu au conuri,
556 -- se va afisa mesajul "Clientul nu are cont.".
557
558 SELECT nume||' '||prenume NUME_INTREG,
559        CASE WHEN c_s.ume_utilizator IS NOT NULL THEN c_s.ume_utilizator
560             ELSE 'Clientul nu are cont.' END NUME_UTILIZATOR,
561        denumire, nr_stele
562 FROM CLIENT cl LEFT OUTER JOIN CONT_SITE c_s ON (cl.id_client = c_s.id_client)
563             LEFT OUTER JOIN RECENZIE r ON (c_s.ume_utilizator = r.ume_utilizator)
564             LEFT OUTER JOIN PRODUS p ON (r.id_produs = p.id_produs);

```

	NUME_INTREG	NUME_UTILIZATOR	DENUMIRE
1	Carstea Darian-Stefan	Darian Stefan	PC Gaming Raptor5, Intel i5-9400F 2.9GHz Coffee I
2	Carstea Darian-Stefan	Darian Stefan	PC Gaming Raptor5, Intel i5-9400F 2.9GHz Coffee I
3	Manole Iuliana-Elena	Elena 2008	Laptop Lenovo 15.6' ThinkPad E15 Gen 2, FHD, Proc
4	Castan Mirela	Mirela Castan	Laptop Lenovo 15.6' ThinkPad E15 Gen 2, FHD, Proc
5	Florian Andrei-Cosmin	Clientul nu are cont.	(null)
6	Popa Marcel-Radu	Clientul nu are cont.	(null)
7	Aurel Popescu-Mihai	Clientul nu are cont.	(null)

-- Cele doua cereri care utilizeaza operatia division:

-- Sa se afiseze numele intreg, alaturi de salariul angajatilor care au asistat cel putin

-- aceeasi clienti ca si angajatul cu codul 1.

```

SELECT a.ume||' '||a.preume NUME_ANGAJAT, salariu

```

```





FROM ANGAJAT a JOIN ESTE_ASISTAT e_a ON (a.id_angajat = e_a.id_angajat)
        JOIN CLIENT c ON (e_a.id_client = c.id_client)
WHERE c.id_client IN (SELECT id_client
        FROM ESTE_ASISTAT
        WHERE id_angajat = 1)
AND a.id_angajat != 1
GROUP BY a.num||'|'||a.preume, salariu
HAVING COUNT(*) = (SELECT COUNT(id_client)
        FROM ESTE_ASISTAT
        WHERE id_angajat = 1);

```

```

569 -- Sa se afiseze numele intreg, alaturi de salariul angajatilor care au asistat cel putin
570 -- aceeasi clienti ca si angajatul cu codul 1.
571
572 SELECT a.num||'|'||a.preume NUME_ANGAJAT, salariu
573 FROM ANGAJAT a JOIN ESTE_ASISTAT e_a ON (a.id_angajat = e_a.id_angajat)
574        JOIN CLIENT c ON (e_a.id_client = c.id_client)
575 WHERE c.id_client IN (SELECT id_client
576        FROM ESTE_ASISTAT
577        WHERE id_angajat = 1)
578 AND a.id_angajat != 1
579 GROUP BY a.num||'|'||a.preume, salariu
580 HAVING COUNT(*) = (SELECT COUNT(id_client)
581        FROM ESTE_ASISTAT
582        WHERE id_angajat = 1);

```

Query Result x	
    SQL All Rows Fetched: 1 in 0.002 seconds	
NUME_ANGAJAT	SALARIU
1 Mihail Maria-Andreea	4500

-- Sa se afiseze denumirea si pretul produselor care sunt compatibile cu exact aceleasi
-- servicii ca si produsul cu id-ul 5.

```
SELECT p.denumire, p.pret
FROM PRODUS p JOIN ESTE_COMPATIBIL e_c ON (p.id_produs = e_c.id_produs)
        JOIN SERVICIU s ON (e_c.id_serviciu = s.id_serviciu)
WHERE s.id_serviciu IN (SELECT id_serviciu
        FROM ESTE_COMPATIBIL
        WHERE id_produs = 5)
        AND p.id_produs != 5
GROUP BY p.denumire, p.pret
HAVING COUNT(*) = (SELECT COUNT(id_serviciu)
        FROM ESTE_COMPATIBIL
        WHERE id_produs = 5)
```

MINUS

```
SELECT p.denumire, p.pret
FROM PRODUS p JOIN ESTE_COMPATIBIL e_c ON (p.id_produs = e_c.id_produs)
        JOIN SERVICIU s ON (e_c.id_serviciu = s.id_serviciu)
WHERE s.id_serviciu NOT IN (SELECT id_serviciu
        FROM ESTE_COMPATIBIL
        WHERE id_produs = 5);
```

```

584 -- Sa se afiseze denumirea si pretul produselor care sunt compatibile cu exact aceleasi
585 -- servicii ca si produsul cu id-ul 5.
586
587 SELECT p.denumire, p.pret
588 FROM PRODUS p JOIN ESTE_COMPATIBIL e_c ON (p.id_produs = e_c.id_produs)
589         JOIN SERVICIU s ON (e_c.id_serviciu = s.id_serviciu)
590 WHERE s.id_serviciu IN (SELECT id_serviciu
591                         FROM ESTE_COMPATIBIL
592                         WHERE id_produs = 5)
593        AND p.id_produs != 5
594 GROUP BY p.denumire, p.pret
595 HAVING COUNT(*) = (SELECT COUNT(id_serviciu)
596                  FROM ESTE_COMPATIBIL
597                  WHERE id_produs = 5)
598
599 MINUS
600
601 SELECT p.denumire, p.pret
602 FROM PRODUS p JOIN ESTE_COMPATIBIL e_c ON (p.id_produs = e_c.id_produs)
603         JOIN SERVICIU s ON (e_c.id_serviciu = s.id_serviciu)
604 WHERE s.id_serviciu NOT IN (SELECT id_serviciu
605                            FROM ESTE_COMPATIBIL
606                            WHERE id_produs = 5);

```

Query Result x	
SQL All Rows Fetched: 2 in 0.002 seconds	
DENUMIRE	PRET
1 Placa video PowerColor Radeon RX 6900 XT Red Devil 16GB GDDR6 256-bit	11999.99
2 Procesor AMD Ryzen 5 3600 3.6GHz box	999.99

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

Vom optimiza Cererea 1 de la exercițiul 11.

Cerința:

-- Sa se afiseze numele vanzatorului si pretul tuturor produselor al caror rating
 -- are o valoare egala 5, si care au fost cumparate de catre un client al carui nume
 -- este Aurel, care detine o adresa de livrare in localitatea Bucuresti. Rezultatele
 -- se vor ordona in ordine descrescatoare dupa pretul produselor.

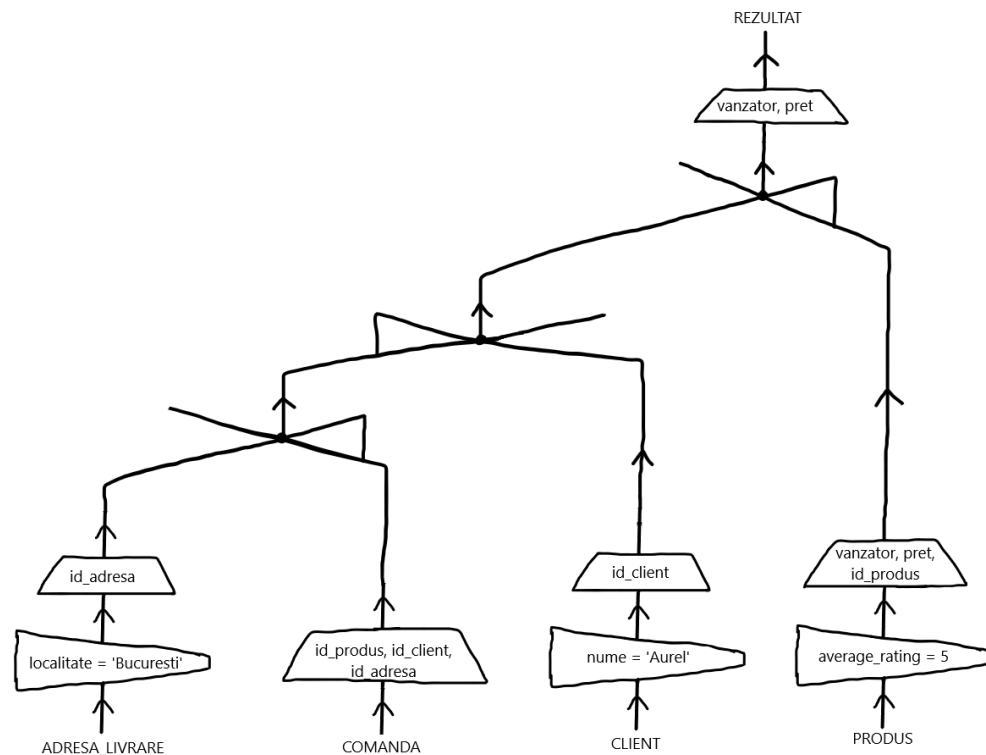
Cererea SQL:

```
SELECT vanzator, pret
FROM PRODUS p JOIN COMANDA co ON (p.id_produs = co.id_produs)
      JOIN CLIENT cl ON (co.id_client = cl.id_client)
      JOIN ADRESA_LIVRARE a_l ON (co.id_adresa = a_l.id_adresa)
WHERE (average_rating = 5) AND (INITCAP(ume) = 'Aurel') AND (a_l.localitate = 'Bucuresti')
ORDER BY pret DESC;
```

Expresia algebrică:

```
R1 = SELECT(ADRESA_LIVRARE, localitate = 'Bucuresti')
R2 = PROJECT(R1, id_adresa)
R3 = PROJECT(COMANDA, id_produs, id_client, id_adresa)
R4 = SEMIJOIN(R3, R2, id_adresa)
R5 = SELECT(CLIENT, nume = 'Aurel')
R6 = PROJECT(R5, id_client)
R7 = SEMIJOIN(R4, R6, id_client)
R8 = SELECT(PRODUS, average_rating = 5)
R9 = PROJECT(R8, vanzator, pret, id_produs)
R10 = SEMIJOIN(R9, R7, id_produs)
REZULTAT = R11 = PROJECT(R10, vanzator, pret)
```


Arborele algebric:



Arborele asociat cererii a fost optimizat la maximum inca de la bun inceput.

În cadrul optimizării acestuia, am aplicat regulile de optimizare 1 și 4, din cadrul cursului, alături de proprietățile 5 și 11, tot din cadrul cursului:

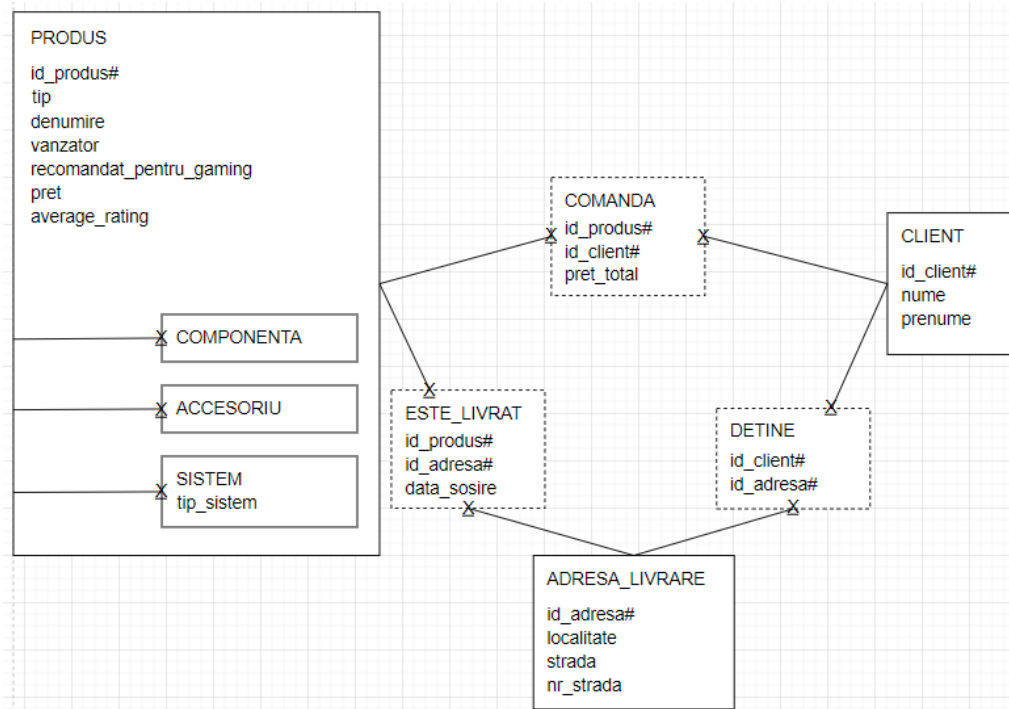
- regula de optimizare 1 a fost utilizată pentru a reduce dimensiunea relațiilor pe baza cărora se realizează SEMIJOIN-uri;
- regula 4 de optimizare a fost utilizată pentru a elimina atributele nefolositoare din relațiile implicate în operațiile de SEMIJOIN;
- proprietatea 5 a fost utilizată pentru a eficientiza arborele algebric (obținem un arbore mai eficient dacă realizăm SELECT-urile înainte a PROJECT-urilor). Astfel, prin intermediul acesteia se realizează, într-un mod optim, legăturile dintre toate cele 4 relații.

18. a. Realizarea normalizării BCNF, FN4, FN5.

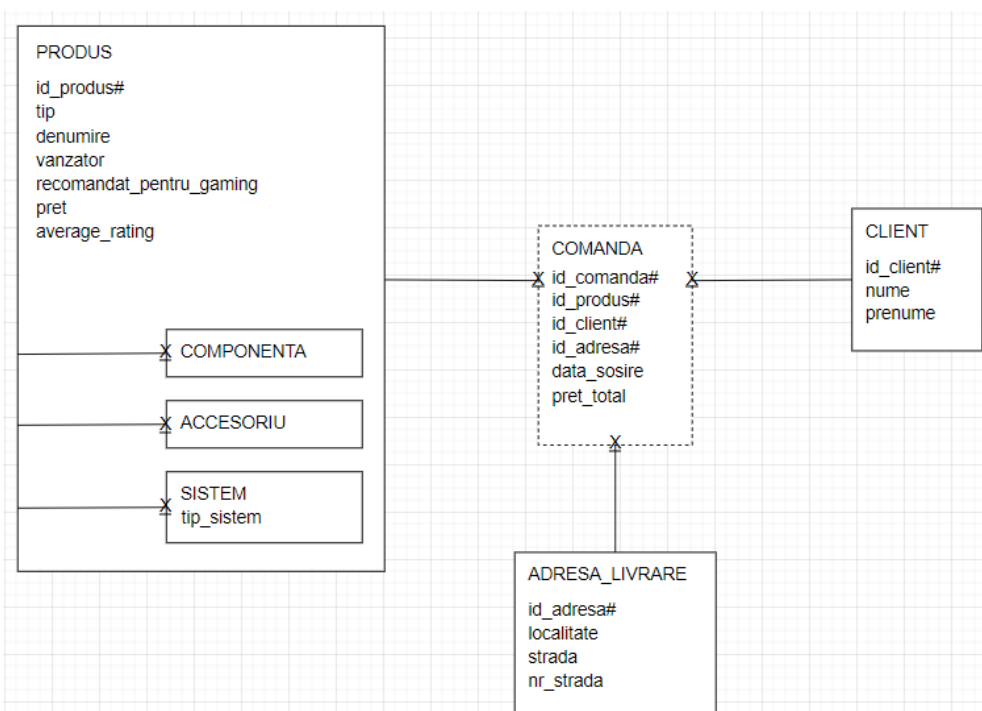
FN5:

Pentru această formă normală, ne vom lega atât de informații din modelul real, cât și de informații fictive.

Varianta non-FN5:



Varianta FN5:



- Relația inițială (cea de la „Varianta non-FN5”) se află deja în FN4, deoarece nu conține dependențe multiple.
- Relația inițială nu se află în FN-5, deoarece este alcătuită din 3 relații de tip 2, care prezintă dependențe ciclice.

Pentru a aduce relația inițială în FN-5, am eliminat dependențele ciclice (implicit și cele trei tabele asociative), și am introdus un singur tabel asociativ, care leagă tabelele „PRODUS”, „ADRESA_LIVRARE”, și „CLIENT”.

18. b. Aplicarea denormalizării, justificând necesitatea acesteia.

Pentru denormalizare, ne vom lega atât de informații din modelul real, cât și de informații fictive.

Varianta fără denormalizare:

PRODUS					
id_produș#	tip	denumire	recomandat_pentru_gaming	pret	average_rating
1	Componenta	Produș 1	Da	1799.29	4.5
2	Componenta	Produș 2	Da	2469.30	5
3	Componenta	Produș 3	Nu	499.99	3.5
4	Componenta	Produș 4	Da	1049.99	5

DETALII_PRODUS	
id_produș#	vanzator
1	Vanzator 1
2	Vanzator 1
3	Vanzator 2
4	Vanzator 1

Varianta cu denormalizare:

PRODUS						
id_produș#	tip	denumire	vanzator	recomandat_pentru_gaming	pret	average_rating
1	Componenta	Produș 1	Vanzator 1	Da	1799.29	4.5
2	Componenta	Produș 2	Vanzator 1	Da	2469.30	5
3	Componenta	Produș 3	Vanzator 2	Nu	499.99	3.5
4	Componenta	Produș 4	Vanzator 1	Da	1049.99	5

- Relațiilor din „Varianta fără denormalizare” li s-a aplicat procesul de denormalizare, deoarece operațiile de join dintre cele două tabele pot fi costisitoare din punct de vedere al timpului de executare. Prin urmare, s-a mutat atributul „vanzator” în tabelul „PRODUS”.