*Lupascu Miruna-Stefania 151*

***PROIECT-SALON INFRUMUSETARE***

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

Patronul unui lant de saloane de frumusete doreste ca activitatea acestuia sa fie îmbunatatita. Localul dispune de personal calificat. Acestia sunt angajati pe baza unui contract care poate fi personalizat.

Salonul de frumusete ofera diverse tipuri de servicii (coafura, manichiura, solar etc.), încercând sa satisfaca cele mai diverse cerinte. La o sedinta, clientul poate beneficia de mai multe servicii. Fiecare serviciu poate fi onorat de angajati diferiti, astfel, la o sedinta, mai multi angajati se pot ocupa de client (un angajat de manichiura, altul de pedichiura, altul de machiaj, etc).

Fiecare angajat poate oferi cupoane de reducere clientilor fideli pe care sa le foloseasca la oricare dintre serviciile oferite de salon (trebuie specificat la ce serviciu doreste sa il foloseasca). Un client poate folosi pentru o sedinta mai multe cupoane(poate beneficia de reduceri atat la manichiura, coafor, cosmetica..)

Salonul ofera si facturi. Factura poate fi platita atat de clientul care a beneficiat de sedinta, cat si de un alt client(in cazul unui cadou), astfel ca trebuie sa se cunoasca id-ul clientului care a platit.

Anual se face un top pe categorii(coafura, cosmetioca, machiaj...) al saloanelor din lant.

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

Constrangerile modelului de date se pot observa din diagrama, urmand sa prezentam doar regulile ce se considera a nu fi evidente.

-un angajta poate incheia un singur contract la un singur salon

-un client are exact o preferinta factura pe care o aplica mereu

3.Descrierea entităților, incluzând precizarea cheii primare

Entităţile modelului de date se pot observa din diagrama, urmand sa prezentam doar entitatile ce se considera a nu fi evidente (se va preciza si cheia primară).

Observatii: entitate ce tine evidenta aspectelor legate de contracte(perioada si concediul - daca este cazul). Cheia primara este observatii\_id.

Factura : entitate ce tine evidenta aspectelor legate de facturi(modul de trimitere a facturii pentru o anumita sedinta). Cheia primara este factura\_id.

4. Descrierea relațiilor, incluzând precizarea cardinalității acestora.

Relatiile se pot observa din diagrama, urmand sa prezentam doar entitatila ce se considera a nu fi evidente.

*Client -preferinta\_factura (one to one)*

Un client are exact un tip de preferinta factura pe care il „aplica” mereu

*Client-sedinta (one to many)*

-un client poate solicita una sau mai multe sedinte. O sedinta apartine unui singur client

*Sedinta-sevicii (many to many)* -🡪 se transforma in **tabelul asociativ *sedinta\_serviciu***

**-** o sedinta contine unu sau mai multe servicii, care sunt trecute in tabelul ascociativ. Un serviciu poate fi trecut pe mai multe sedinte

*Sedinta - sedinta\_serviciu(one to many)*

*Servicii- sedinta\_serviciu (one to many)*

*Angajat-serviciu(one to many)*

-un angajat poate prelua unu sau mai multe servicii.Un serviciu poate fi preluat de un singur angajat.

*Angajat-contracte*

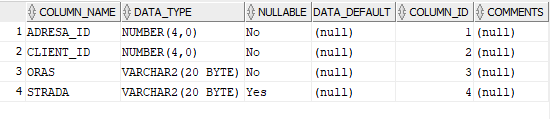
-un angajat poate avea un singur contract. Un contract e incheait cu un singur angajat.

*Salon-contract*

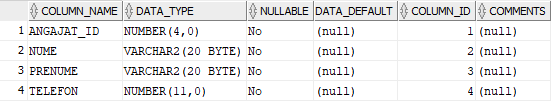
-un contract apartine unui singur salon. Un salon are mai multe contracte incheiate.

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

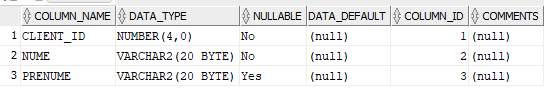
ADRESA:



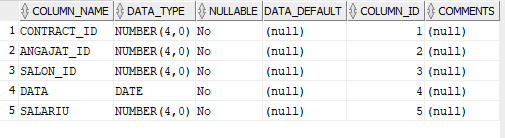
ANGAJAT:



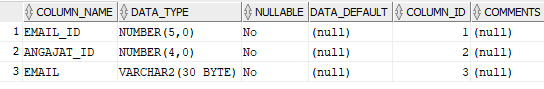
CLIENT:



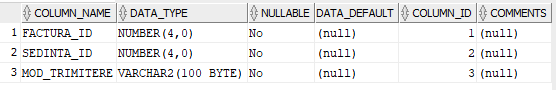
CONTRACT:



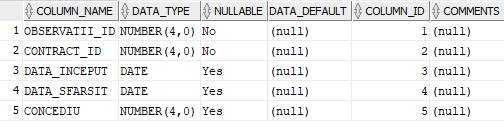
EMAIL:



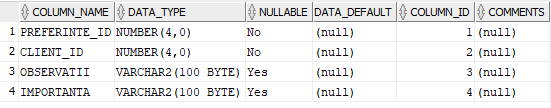
FACTURA:



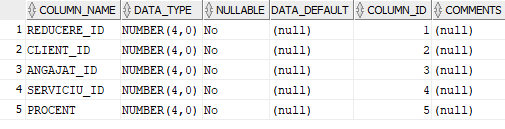
OBSERVATII:



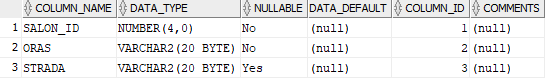
PREFERINTE\_FACTURA:



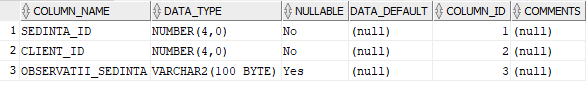
REDUCERE:



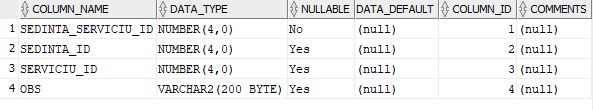
SALON:



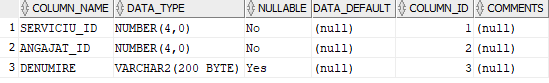
SEDINTA:



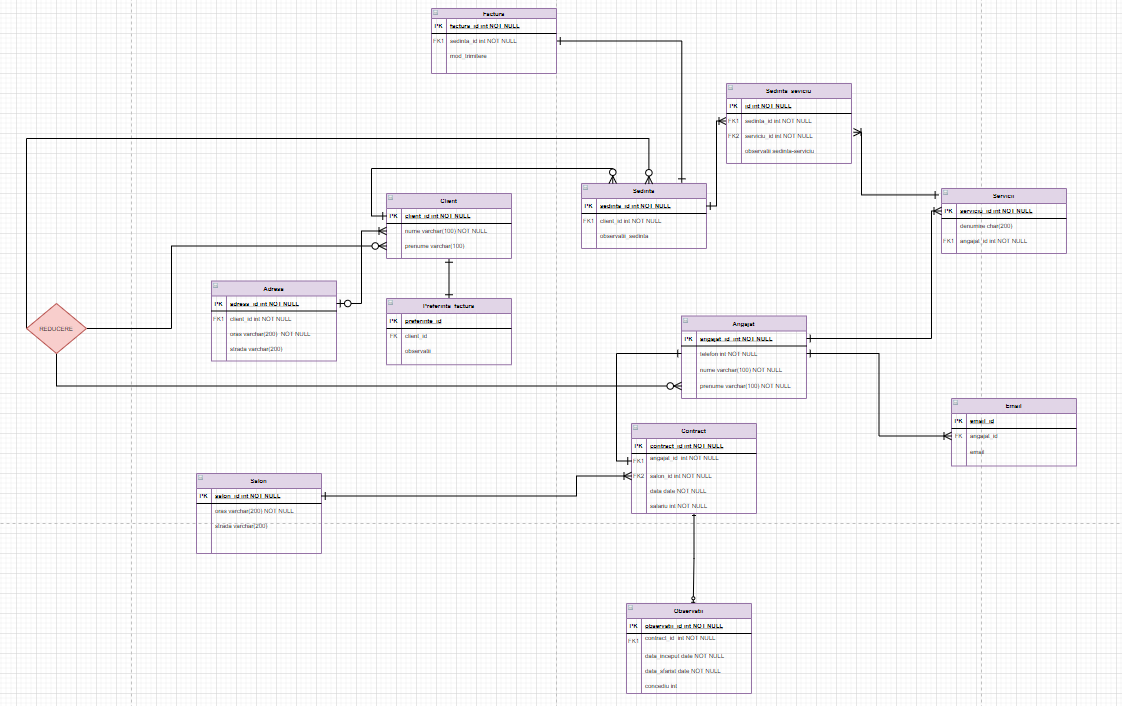
SEDINTA\_SERVICIU:



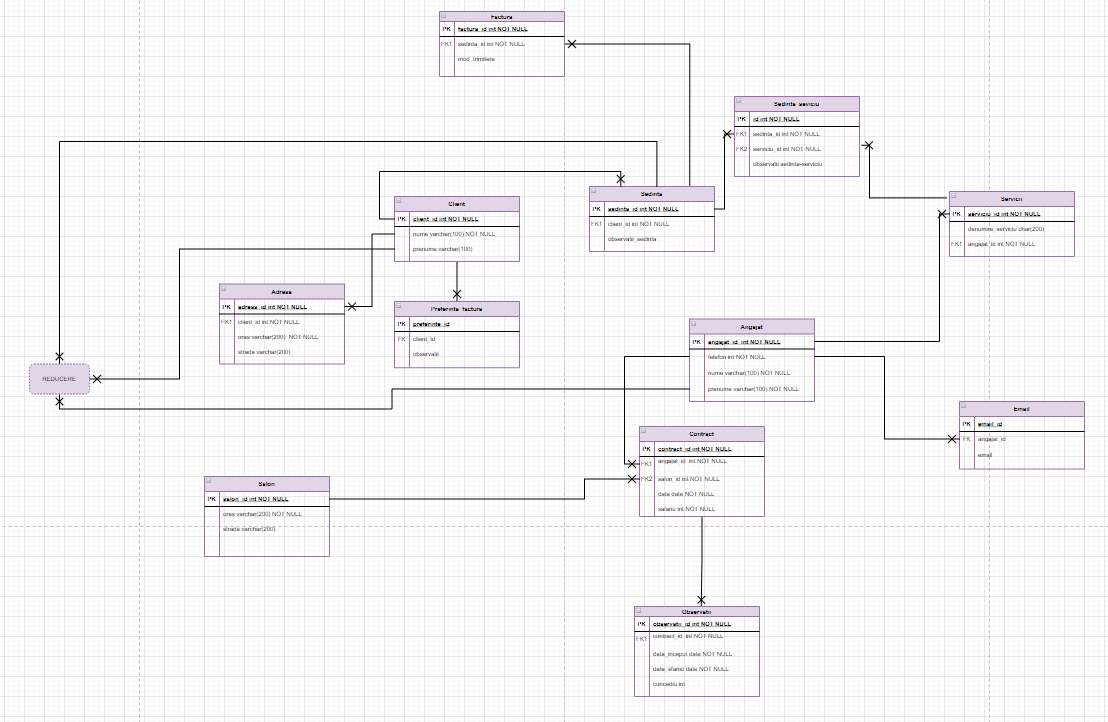
SERVICII:



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 6 tabele (fără considerarea subentităților), dintre care cel puțin un tabel asociativ.



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

Atributele se afla in totalitate in diagrama entitate-relatie.

Exemple de schema relationala:

Reducere(#client\_id, #sedinta\_id, #angajat\_id, suma)

contract(#contract\_id, angajat\_id, salon\_id, data, salariu)

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

**non-FN1**

**Daca era asa: (un angajat are mai multe email-uri)**

angajat(#angajat\_id, telefon, nume, prenume, email)

*email nu este atribut atomic –-> transformam*

email(#email\_id, angajat\_id, email)

**non-FN2**

**Daca era asa:**

(#sedinta\_id, #client\_id, preferinte, mod\_trimitere)

Pentru sedinte se emite o factura(poate fi platita si de un alt client in cazul in care este un cadou) se stie ca fiecare persoana are anumite preferinte legate de factura(daca sa aiba sau nu anumite semnaturi etc). Pentru fiecare factura se poate alege daca sa fie trimisa pe mail sau listata.

*client\_id 🡪preferinte (dependenta partiala)*

*#sedinta\_id, #client\_id 🡪 mod trimitere*

Descompunere:

(#sedinta\_id, # client\_id, mod\_trimitere)

(#client\_id, preferinte)

**non-FN2 &non-FN3**

Anual se face un top pe categorii(coafura, cosmetica, machiaj...) al saloanelor din lant. Castigatorii primesc sposorizari. Premiul in bani difera de la an la an.

**Daca era asa:**

TOP(#an, #categorie, salon\_castigator, manager\_salon, premiu)

*An 🡪 premiu*

*(an, categorie) 🡪 (salon\_castigator, manager\_salon)*

**Transformam in FN2:**

(#an, premiu)

(#an, #categorie, salon\_castigantor, manager\_Salon)

Dar inca nu este in fn3 deoarece *salon\_castigator->manager\_castigator*

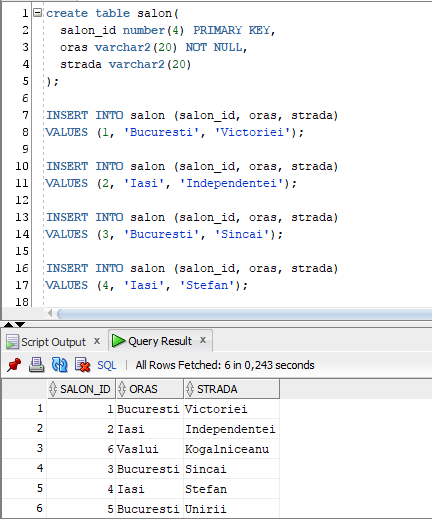
**Transformam in FN3:**

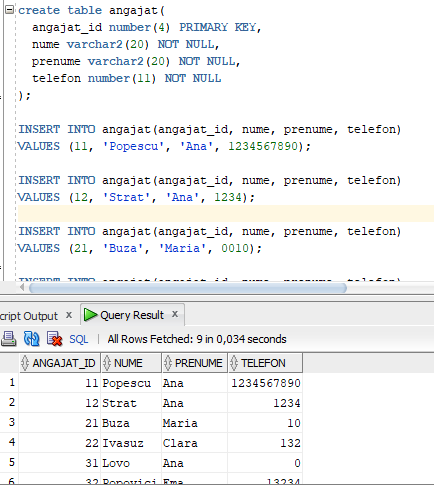
(#an, premiu)

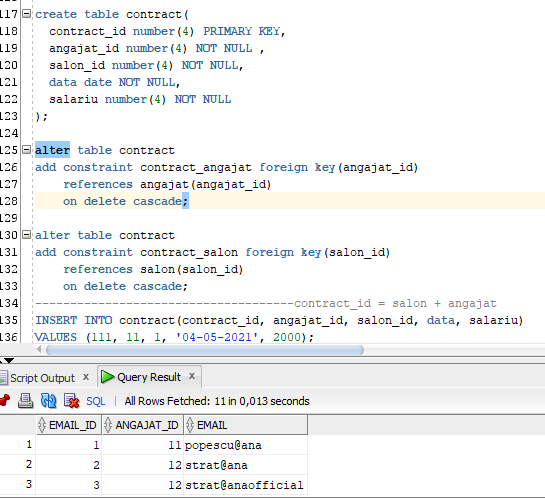
(#an, #categorie, salon\_castigantor)

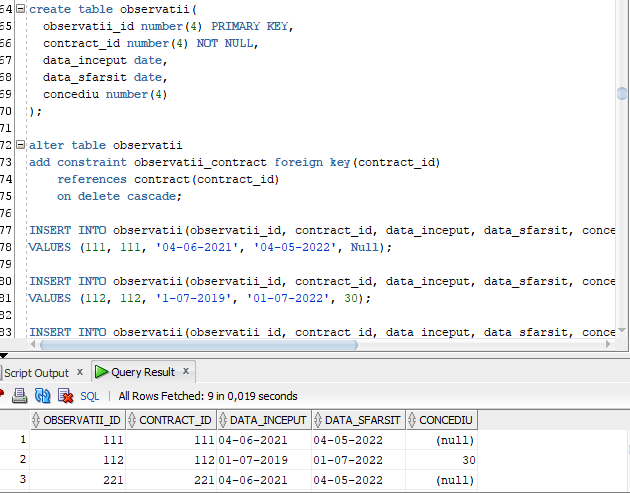
(salon\_castigator, manager\_Salon)

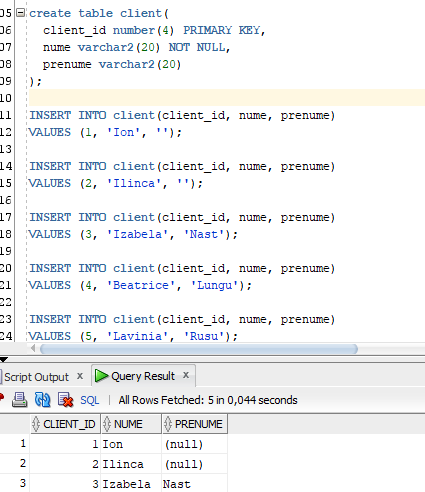
10. Crearea tabelelor în SQL și inserarea de date coerente în fiecare dintre acestea (minimum 5 înregistrări în fiecare tabel neasociativ; minimum 10 înregistrări în tabelele asociative).

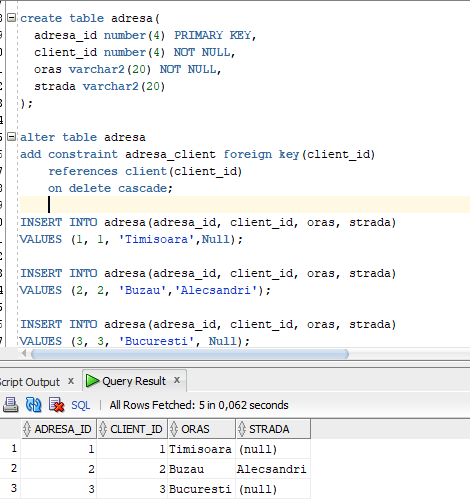


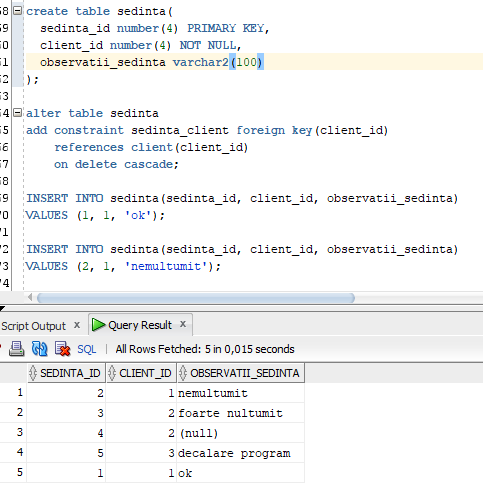


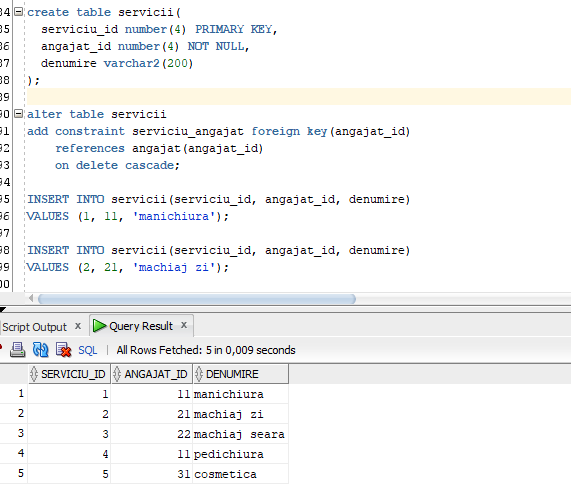


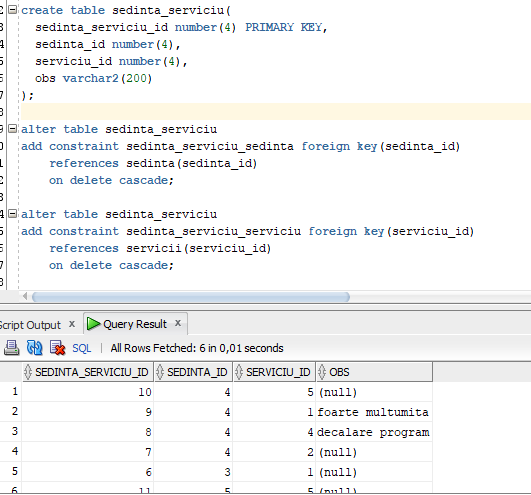


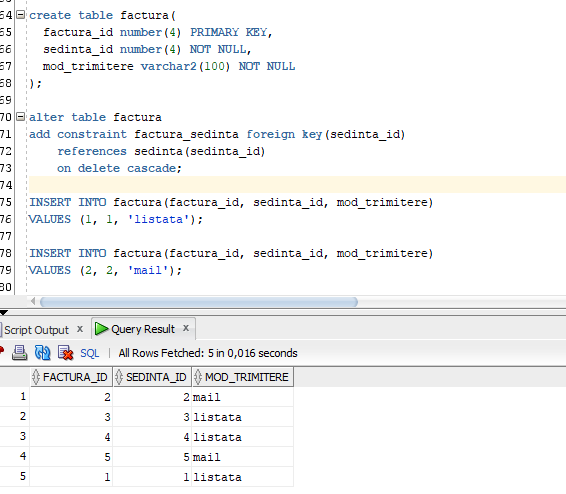


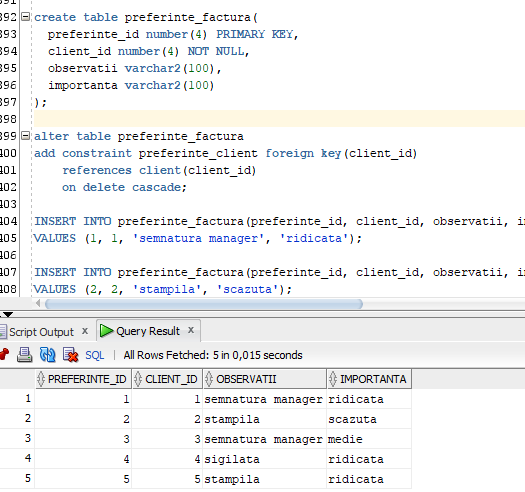


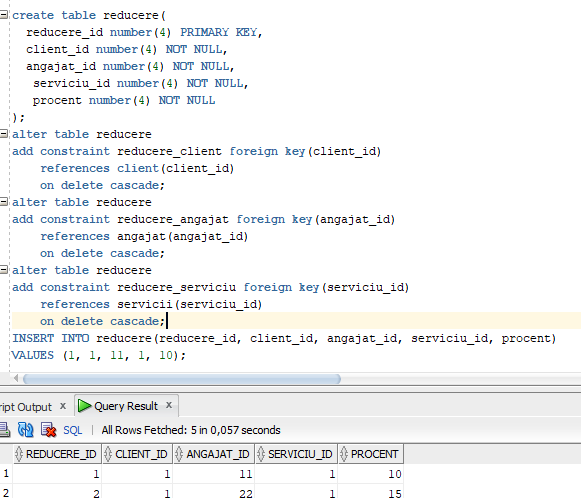




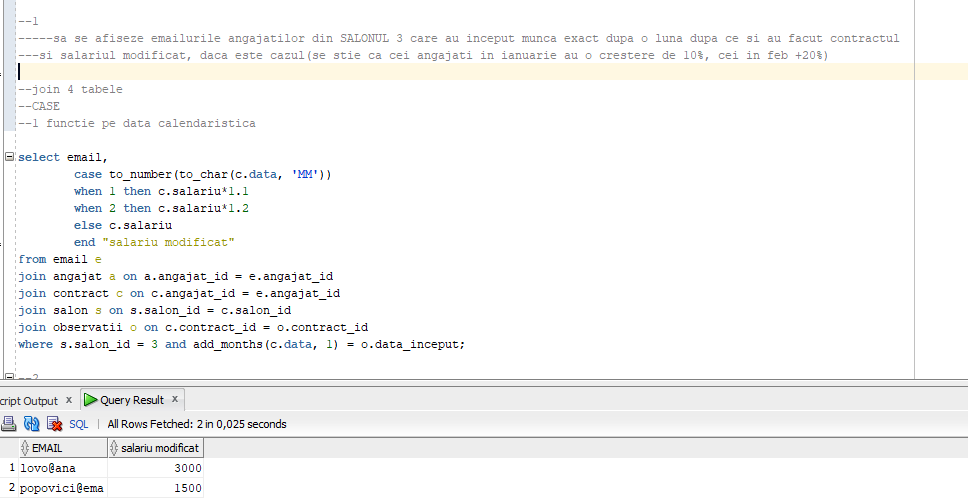


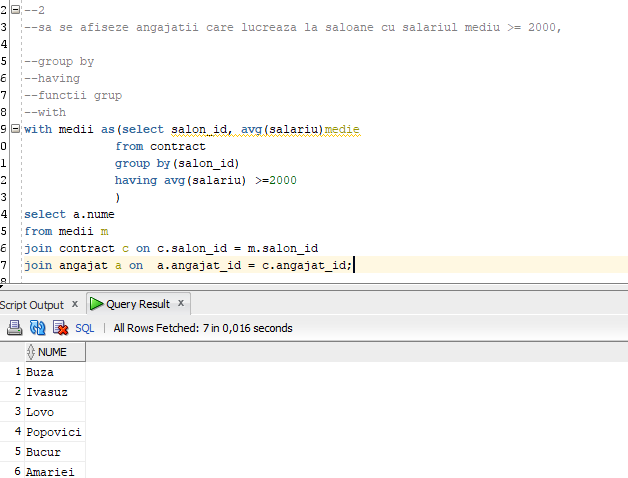


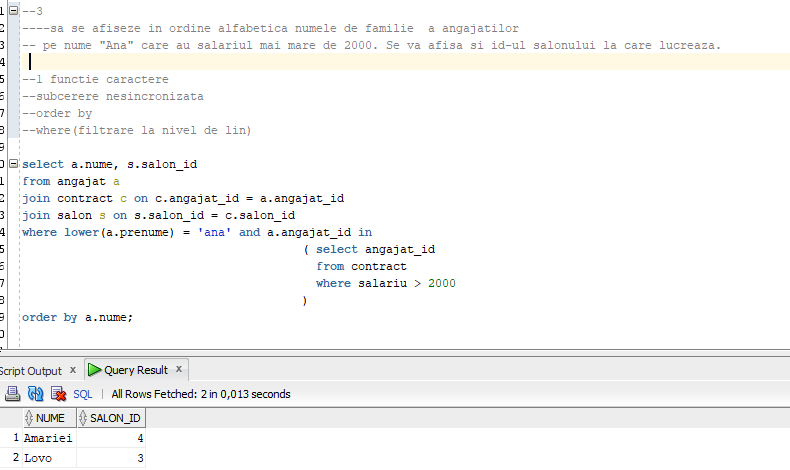


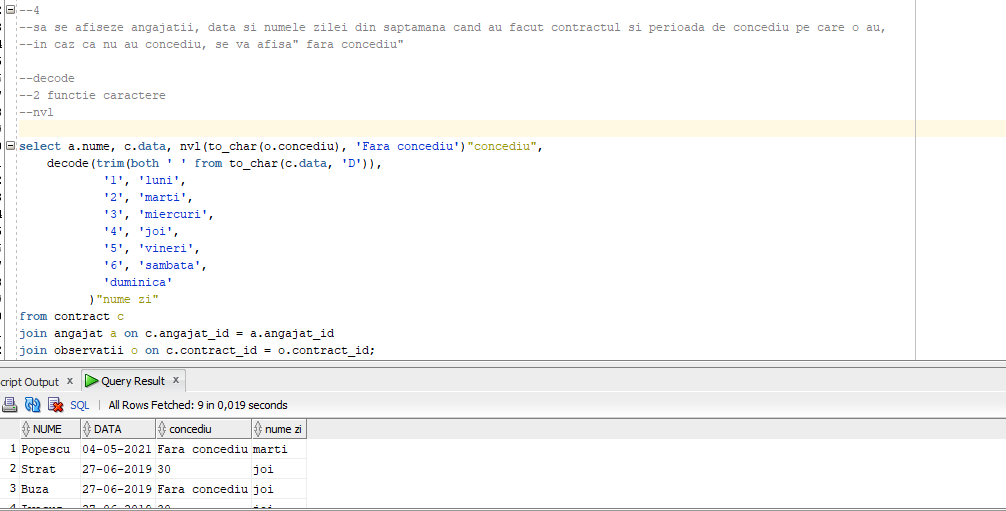


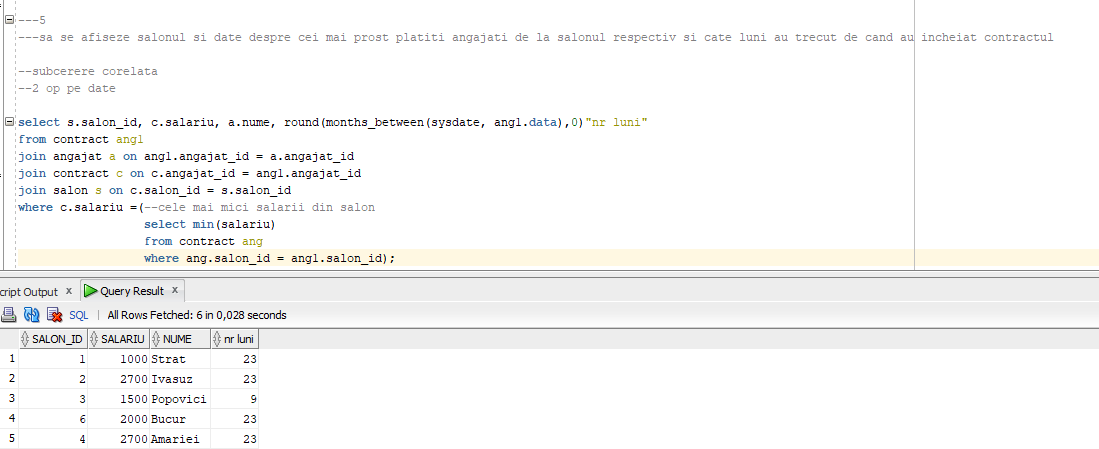
11. Formulați în limbaj natural și implementați 5 cereri SQL complexe.



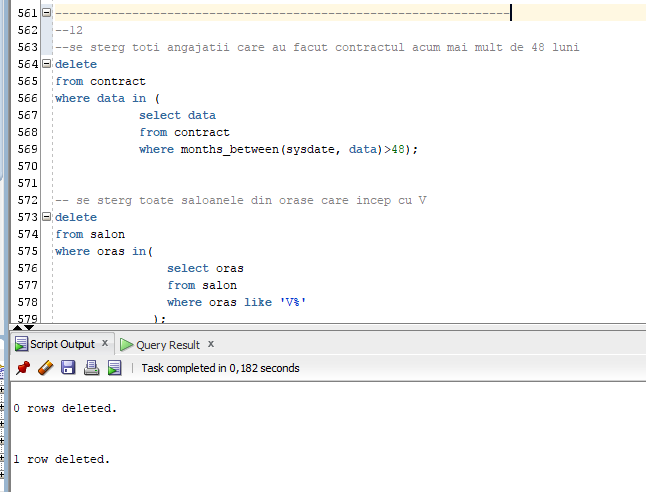


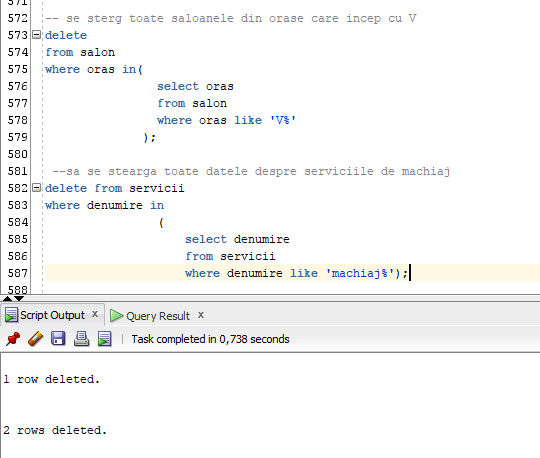




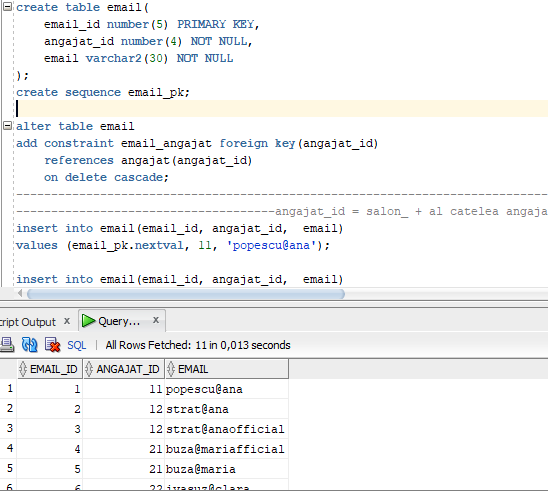


12.Implementarea a 3 operații de actualizare sau suprimare a datelor utilizând subcereri.





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



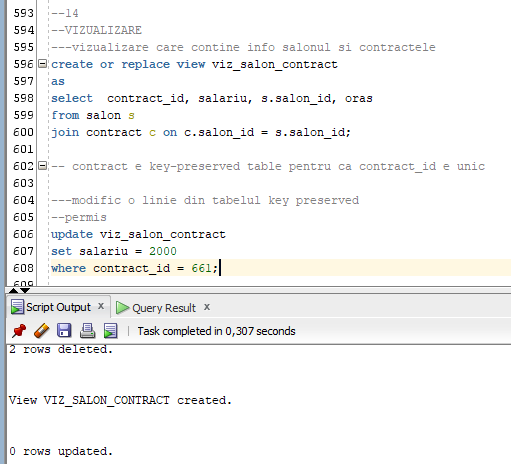
14.Crearea unei vizualizări compuse. Dați un exemplu de operație LMD permisă pe vizualizarea respectivă și un exemplu de operație LMD nepermisă.

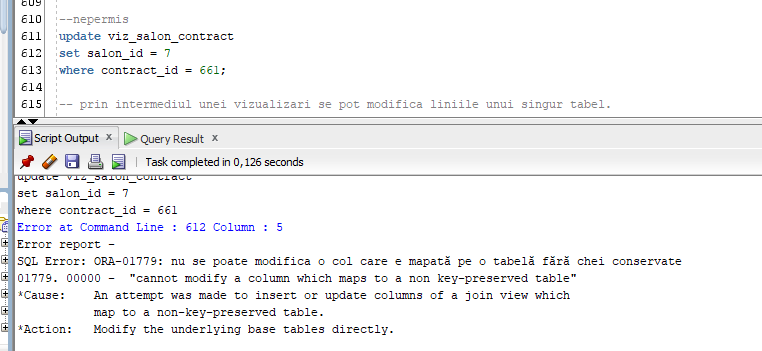
„Pentru vizualizările bazate pe mai multe tabele, orice operaţie INSERT, UPDATE sau DELETE poate modifica datele doar din unul din tabelele de bază. Acest tabel este cel protejat prin cheie (*key preserved*). În cadrul unei astfel de vizualizări, un tabel de bază se numeşte *keypreserved* dacă are proprietatea că fiecare valoare a cheii sale primare sau a unei coloane având constrângerea de unicitate, este unică şi în vizualizare. Prima condiţie ca o vizualizare a cărei cerere conţine un join să fie modificabilă este ca instrucţiunea LMD să afecteze un singur tabel din operaţia de join.”

Vizualizare*: viz\_salon\_contract* in care retinem informatii legate de saloane si contracte.

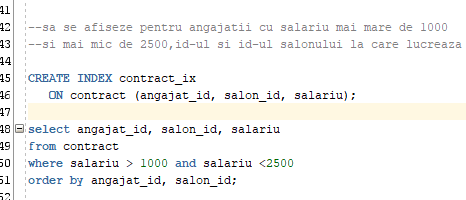
Tabelul „contract” este key-preserved pentru ca atributul „contract\_id” mentine propietatea de unicitate.

Putem sa modificam salariul unui contract deoarece modificam o singura linie a tabelului key preserved.





15.Crearea unui index care să optimizeze o cerere de tip căutare cu 2 criterii. Specificați cererea.



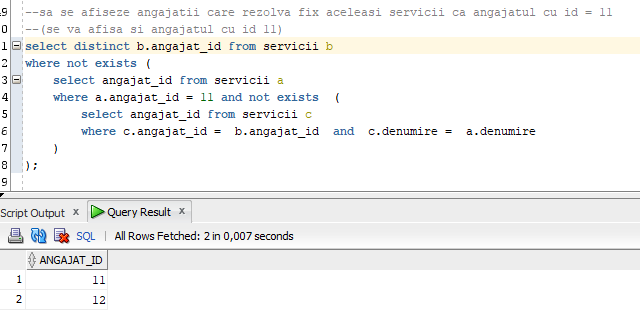
16**.** Formulați în limbaj natural și implementați în SQL: o cerere ce utilizează operația *outer-join* pe minimum 4 tabele și două cereri ce utilizează operația *division*.

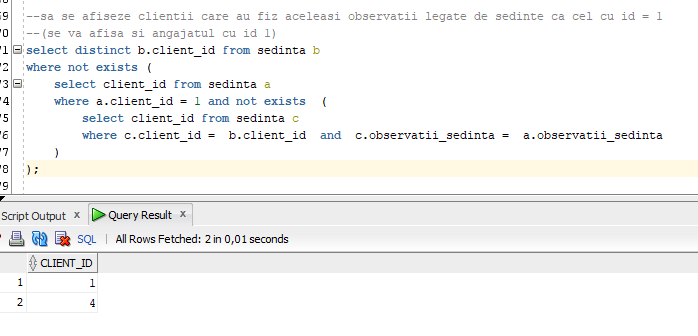
„Diviziunea este o operaţie binară care defineşte o relaţie ce conţine valorile atributelor dintr-o relaţie care apar **în toate** valorile atributelor din cealaltă relaţie.

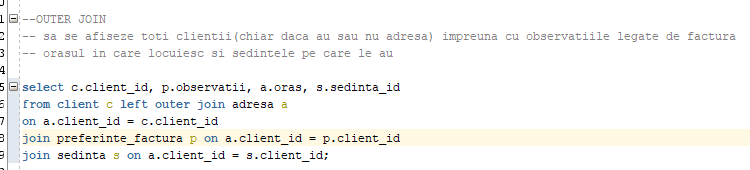
Operatorul *DIVISION* este legat de cuantificatorul universal care nu există în *SQL*. Cuantificatorul universal poate fi însă simulat cu ajutorul cuantificatorului existenţial utilizând relaţia:

(oricare)x P(x) <=> ¬ exista x ¬ exista P(x).

Prin urmare, operatorul *DIVISION* poate fi exprimat în *SQL* prin succesiunea a doi operatori *NOT EXISTS*.”





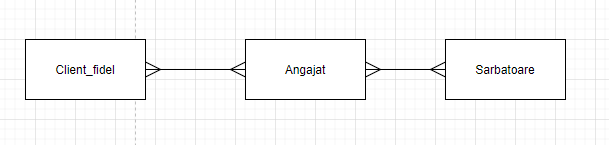


18.a. Realizarea normalizării BCNF, FN4, FN5.

**non-FN4**

Story:

Angajatii au clienti fideli. Un client poate fi fidel (care beneficiaza de anumite privilegii) la mai multi angajati, iar un angajat poate avea mai multi clienti fideli. Pe parcursul anului sunt mai multe sarbatiori in care angajatii sunt liberi, dar acestia fac cu randul (pot fi mai multi angajati liberi in aceeasi zi) pentru a nu da salonul faliment.



Normalizare:

Angajat se descompune astfel:

Angajat \_1(Angajat, sarbatoare)

Angajat \_2(Angajat, client\_fidel)

**non-BCNF**

Story:

Salonul pune la dispozitie mai multe masini pentru ca angajatii sa isi poata procura produsele necesare. O masina apartine unui singur angajat, dar un angajat poate sa aiba mai multe masini. Din cand in cand, angajatii se duc sa alimenteze la benzinaria unde e cel mai ieftin din oras. Stiind ca salonul deconteaza si acest transport, trebuie sa tinem evidenta.

alimentare(#benzinarie, #angajat, #data, nr\_masina, pret)

(benzinarie, angajat, data) 🡪 nr\_masina

(benzinarie, angajat, data) 🡪 pret

Curier -/->nr\_masina nr\_masina 🡪curier

K1 = benzinarie

K2 = angajat

X = nr\_masina

Y = data, pret

(Benzinarie, nr\_masina, data, pret)

(nr\_masina, curier)

**non-FN5**

Descompunem Reducere(#client, #angajat, #serviciu) in 3 tabele:

Reducere\_client\_serviciu(#client, #serviciu)

Reducere\_client\_angajat(#client, #angajat)

Reducere\_angajat\_serviciu(#angajat, #serviciu)

b) Princialul obiectiv al denormalizarii este cresterea performantei programelor de manipulare a datelor. Aceasta denormalizare se foloseste pentru operatii sau calcule efectuate frecvent.

Sa presupunem ca in tabelul anterior „reducere” pentru fiecare angajat vrem sa stim si cator clienti le a oferit cupon de reducere.

Aceasta informatie poate fi extrasa din cele 3 tabele, dar inseamna ca va fi recalculat de fiecare data.

Astefel ca putem sa avem un tabel angajat\_nr\_clienti(#angajat, nr\_clienti). Datele sunt redundante, dar folosirea lor este mai facila si rapida decat recalcularea datelor.

Pe de alta parte insa, folosirea acestui tabel suplimentar are si dezavantaje, putand duce la pierderea integritatii datelor.

Modificare datelor in tabelul principal nu se reflecta automat in angajat\_nr\_clienti