

GESTIUNEA INFORMAȚIILOR UNEI COMPANII DE UTILITĂȚI

1. Proiectul își propune să proiecteze o bază de date care să gestioneze o parte din informațiile și problemele unei companii ce oferă utilități clienților (această bază de date poate fi folosită pentru o aplicație a acestei companii, dar nu numai. Această aplicație poate fi destinată persoanelor obișnuite pentru a se înregistra, devenind astfel clienți ai companiei, pentru a-și vedea facturile, pentru a trimite plângeri în legătură cu eventuale defecțiuni, pentru a-și plăti facturile direct din aplicație).

Regulile de funcționare ale modelului sunt:

- 1) Despre clienți se cunosc informații personale cât și bancare.
- 2) Un client are adrese, iar pentru fiecare dintre adrese se emit facturi.
- 3) Un client poate depune o plângere dacă există o defecțiune ce trebuie reparată.
- 4) Dacă plângerea este considerată justificată va avea loc o intervenție pe baza acelei plângeri (sau mai multor plângeri dacă există mai mulți clienți în aceeași zonă care au depus o plângere similară), iar la intervenții vor participa muncitori (angajați ai companiei).
- 5) De asemenea o singură plângere poate genera mai multe intervenții dacă se consideră că problema nu este una locală ci are loc și în cazul altor clienți.
- 6) Se cunosc informații personale despre un muncitor (angajat al companiei), cât și specializările sale.

2. Restricțiile impuse asupra modelului sunt:

- 1) Despre un client se cunosc informații din maxim un singur obiect ce oferă informații despre clienți (adică din tabelul Informatii_Clienti)
- 2) Într-un obiect ce oferă informații despre clienți se cunosc informații despre cel mult un singur client.
- 3) Despre un client se cunosc informații bancare din maxim un singur obiect ce oferă informații bancare despre clienți.
- 4) Într-un obiect ce oferă informații bancare despre clienți se cunosc informații bancare despre un singur client.
- 5) Informațiile bancare ale unui client pot avea asociate mai multe carduri bancare.
- 6) Un card bancar trebuie să fie asociat informațiilor bancare ale unui singur client.
- 7) Un client poate să nu dețină nicio adresă, dar poate deține și mai multe adrese.
- 8) O adresă este deținută de un singur client.
- 9) La o adresă poate să nu fie emisă nicio factură, dar pot fi emise și mai multe facturi.
- 10) O factură este emisă doar la o adresă.
- 11) Un client poate să nu dispună de nicio plângere, dar poate depune și mai multe plângeri.
- 12) O plângere poate fi depusă de un singur client.
- 13) O plângere poate să nu determine nicio intervenție, dar poate determina și mai multe.
- 14) Un muncitor poate să nu participe la nicio intervenție, dar poate participa și la mai multe intervenții.

- 15) O intervenție este declanșată de cel puțin o plângere, dar poate fi declanșată și de mai multe plângeri.
- 16) La o intervenție trebuie să participe cel puțin un muncitor, dar pot participa și mai mulți muncitori.
- 17) Despre un muncitor se cunosc informații din cel mult un singur obiect ce oferă informații despre muncitori.
- 18) Într-un obiect ce oferă informații despre muncitori (din tabelul Informatii_Muncitori) se cunosc informații despre maxim un singur muncitor.
- 19) Un muncitor trebuie să aibă cel puțin o specialitate, dar poate avea și mai multe.
- 20) Pentru fiecare specialitate există cel puțin un muncitor care o deține, dar pot exista și mai mulți.

3. Entitățile prezente în model sunt următoarele:

- 1) CLIENTI = persoană ce este client al companiei de utilități. **Cheia primară** asociată acestei entități este **cod_client**.
- 2) INFORMATII_CLIENTI = oferă informații personale despre un client al companiei. **Cheia primară** asociată acestei entități este **cod_informatii_client**.
- 3) INFORMATII_BANCARE = oferă informații bancare pentru un client (cum am spus mai sus, această bază de date poate fi folosită pentru o aplicație destinată clienților companiei în care aceștia își pot plăti facturile, printre altele, așa că toate informațiile prezentate în acest tabel se referă strict la informațiile bancare din cadrul aplicației), cum are fi soldul disponibil (soldul din aplicație), banii cheltuiți până acum pe facturi (banii cheltuiți în cadrul aplicației). **Cheia primară** asociată acestei entități este compusă din **cod_informatii_bancare** și **cod_client**.
- 4) CARDURI_BANCARE = card bancar introdus în aplicație de către client, pentru a-și plăti facturile. **Cheia primară** asociată acestei entități este compusă din **cod_card_bancar**, **cod_informatii_bancare** și **cod_client**.
- 5) ADRESE = informații privind o adresă unde se realizează un consum de utilități a unui client. **Cheia primară** asociată acestei entități este compusă din **cod_adresa** și **cod_client**.
- 6) FACTURI = o factură emisă unui client pe baza utilităților consumate. Pentru fiecare adresă a unui client este emisă o factură odată la o perioadă de timp. **Cheia primară** asociată acestei entități este compusă din **cod_factura**, **cod_adresa** și **cod_client**.
- 7) PLANGERI = sesizare depusă de client în legătură cu o problemă ce ține de utilitățile furnizate de companie (ex: nu funcționează gazele, a fost spartă o țevă, nu este curent). **Cheia primară** asociată acestei entități este compusă din **cod_plangere** și **cod_client**.
- 8) MUNCITORI = persoană angajată în cadrul companiei ce se ocupă cu rezolvarea problemelor sesizate de clienți prin intermediul plângerilor. **Cheia primară** asociată acestei entități este **cod_muncitor**.
- 9) INTERVENTII = reprezintă o acțiune de reparare a unei defecțiuni întreprinsă de unul sau mai mulți muncitori. O intervenție survine în urma unei sau mai multor plângeri depuse de clienți. O plângere nu va rezulta totdeauna într-o intervenție, în cazul care plângerea este considerată una nejustificată. De asemenea, pot exista mai multe intervenții ca urmare a unei singure plângeri, în cazul în care se autosesizează (adică fără depunerea unei plângeri de

alți clienți) existența unei probleme similare în mai multe locuri. **Cheia primara** asociată acestei entitati este **cod_interventie**.

- 10) INFORMATII_MUNCITORI = oferă informații personale despre un muncitor (angajat ce se ocupă cu intervențiile) al companiei. **Cheia primară** asociată acestei entități este **cod_informatii_muncitor**.
- 11) SPECIALIZARI = arie de activitate a unui muncitor. Printre specializări putem regăsi sudor, sofer, stivuitor, electrician, instalator etc. **Cheia primară** asociată acestei entități este **cod_specializare**.

4. Relațiile dintre entitățile modelului sunt următoarele:

- 1) Relația CLIENT_se cunosc_INFORMATII_CLIENT desemnează asocierea dintre un client și informațiile personale despre acesta. Ținând cont de regulile 1 și 2 impuse asupra modelului, putem deduce tipul de cardinalitate al acestei relații: **cardinalitate minimă 0:0 și cardinalitate maximă 1:1**.
- 2) Relația CLIENT_se cunosc_INFORMATII_BANCARE desemnează asocierea dintre un client și informațiile bancare ale contului acestuia. Ținând cont de regulile 3 și 4 impuse asupra modelului, putem deduce tipul de cardinalitate al acestei relații: **cardinalitate minimă 1:0 și cardinalitate maximă 1:1**.
- 3) Relația INFORMATII_BANCARE_se asociază_CARD_BANCARE desemnează asocierea dintre informații bancare ale unui client și cardurile bancare ale acestuia. Ținând cont de regulile 5 și 6 impuse asupra modelului, putem deduce tipul de cardinalitate al acestei relații: **cardinalitate minimă 1:0 și cardinalitate maximă 1:m**.
- 4) Relația CLIENT_detine_ADRESA desemnează asocierea dintre un client și adresele deținute de acesta. Ținând cont de regulile 7 și 8 impuse asupra modelului, putem deduce tipul de cardinalitate al acestei relații: **cardinalitate minimă 1:0 și cardinalitate maximă 1:m**.
- 5) Relația ADRESA_emite_FACTURA desemnează asocierea dintre adresa unui client și facturile emise la aceasta adresa. Ținând cont de regulile 9 și 10 impuse asupra modelului, putem deduce tipul de cardinalitate al acestei relații: **cardinalitate minimă 1:0 și cardinalitate maximă 1:m**.
- 6) Relația CLIENT_depune_PLANGERE desemnează asocierea dintre un client plângerile depuse de acesta legate de diverse defecțiuni ce au legatură cu sistemul de utilități. Ținând cont de regulile 11 și 12 impuse asupra modelului, putem deduce tipul de cardinalitate al acestei relații: **cardinalitate minimă 1:0 și cardinalitate maximă 1:m**.
- 7) Relația CLIENT_se cunosc_INFORMATII_CLIENT desemnează asocierea dintre un client și informațiile personale despre acesta. Ținând cont de regulile 1 și 2 impuse asupra modelului, putem deduce tipul de cardinalitate al acestei relații: **cardinalitate minimă 0:0 și cardinalitate maximă 1:1**.
- 8) Relația MUNCITOR_lucrează_la_INTERVENTIE_determinată_de_PLANGERE (relație de tip 3) desemnează asocierea unei intervenții determinată de una sau mai multe plângeri și muncitorul/muncitorii care lucrează la această intervenție. Relația arată ce muncitori lucrează la ce intervenție, determinată de care plângere/plângeri.

- 9) Relatia MUNCITOR_se cunosc_INFORMATII_MUNCITOR desemnează asocierea dintre un muncitor și informațiile personale despre acesta. Ținând cont de regulile 17 și 18 impuse asupra modelului, putem deduce tipul de cardinalitate al acestei relații: **cardinalitate minimă 0:0 și cardinalitate maximă 1:1.**
- 10) Relatia MUNCITOR_se cunosc_SPECIALIZARE desemnează asocierea dintre un muncitor și specializările pe care le are acesta. Ținând cont de regulile 19 și 20 impuse asupra modelului, putem deduce tipul de cardinalitate al acestei relații: **cardinalitate minimă 1:1 și cardinalitate maximă n:m.**

5. Descrierea atributelor asociate entităților:

1) Atributele entității CLIENTI:

- cod_client: variabilă de tip întreg, de lungime maximă 4, ce reprezintă codul clientului.
- cod_informatii_client: variabilă de tip întreg, de lungime maximă 4, ce reprezintă codul obiectului ce ne oferă informații personale despre client; atributul este o cheie externă care referențiază cheia primară cod_informatii_client a entității INFORMATII_CLIENTI; valoarea atributului trebuie să fie UNIQUE.
- nume_utilizator: variabilă de tip caracter, de lungime maximă 50, ce reprezintă numele de utilizator în aplicație al clientului (după cum am menționat mai sus această baza de data poate fi folosită și pentru o aplicație în care clienții își pot crea cont); valoarea atributului trebuie să fie diferită de NULL; valoarea atributului trebuie să fie UNIQUE.
- parola: variabilă de tip caracter, de lungime maximă 50, ce reprezintă parola în aplicație al clientului; valoarea atributului trebuie să fie diferită de NULL.
- email: variabilă de tip caracter, de lungime maximă 50, ce reprezintă email-ul în aplicație al clientului; valoarea atributului trebuie să fie diferită de NULL și trebuie să fie UNIQUE.

2) Atributele entității INFORMATII_CLIENTI:

- cod_informatii_client: variabilă de tip întreg, de lungime maximă 4, ce reprezintă codul obiectului ce ne oferă informații personale despre client.
- nume: variabilă de tip caracter, de lungime maximă 50, ce reprezintă numele clientului; valoarea atributului trebuie să fie diferită de NULL.
- prenume: variabilă de tip caracter, de lungime maximă 50, ce reprezintă prenumele clientului; valoarea atributului trebuie să fie diferită de NULL.
- cnp: variabilă de tip caracter, de lungime maximă 50, ce reprezintă cnp-ul clientului; valoarea atributului trebuie să fie diferită de NULL și UNIQUE.

- numar_telefon: variabilă de tip caracter, de lungime maximă 50, ce reprezintă numărul de telefon al clientului; valoarea atributului trebuie sa fie diferită de NULL și UNIQUE.

3) Atributele entității INFORMATII_BANCARE:

- cod_informatii_bancare: variabilă de tip întreg, de lungime maximă 4, ce reprezintă codul obiectului ce ne oferă informații bancare despre clienți.

- cod_client: variabilă de tip întreg, de lungime maximă 4, ce reprezintă codul clientului căruia îi aparțin informațiile bancare; valoarea atributului trebuie sa fie UNIQUE.

- sold_curent: variabilă de tip întreg, de lungime maximă 6, ce reprezintă soldul disponibil al clientului în cadrul aplicației; valoarea atributului trebuie să fie diferită de NULL, valoarea implicită a atributului este 0.

- suma_cheltuita: variabilă de tip întreg, de lungime maximă 6, ce reprezintă suma de bani cheltuită de clienți pentru plata facturilor în cadrul aplicației; valoarea atributului trebuie să fie diferita de NULL, valoarea implicită a atributului este 0.

4) Atributele entității CARDURI_BANCARE:

- cod_card_bancar: variabilă de tip întreg, de lungime maximă 4, ce reprezintă codul obiectului ce ne oferă informații despre cardul bancar introdus în aplicație (a nu se confunda cu codul cardului, adică cel de 16 cifre inscripționat pe acesta).

- cod_informatii_bancare: variabilă de tip întreg, de lungime maximă 4, care reprezintă codul obiectului ce ne oferă informații bancare despre clientul căruia îi aparține acest card.

- cod_client: variabilă de tip întreg, de lungime maximă 4, ce reprezintă codul clientului căruia îi aparține cardul bancar.

- namar_card: variabilă de tip caracter, de lungime maximă 50, ce reprezintă numărul cardului (numărul de 16 cifre inscripționat pe card); valoarea atributului trebuie să fie diferită de NULL.

- data_expirare_card: variabilă de tip dată calendaristică, ce reprezintă data de expirare a cardului; valoarea atributului trebuie să fie diferită de NULL.

- cod_securitate_card: variabila de tip întreg, de lungime maximă 4, ce reprezintă codul de securitate de pe card (CVV-ul de 3 cifre inscripționat pe card); valoarea atributului trebuie să fie diferită de NULL.

5) Atributele entității ADRESE:

- cod_adresa: variabilă de tip întreg, de lungime maximă 4, ce reprezintă codul adresei.

- cod_client: variabilă de tip întreg, de lungime maximă 4, ce reprezintă codul clientului ce deține această adresă.
- tara: variabilă de tip caracter, de lungime maximă 50, ce reprezintă numele țării în care se află respectiva adresa.
- oras: variabilă de tip caracter, de lungime maximă 50, ce reprezintă numele orașului în care se află respectiva adresa.
- strada: variabilă de tip caracter, de lungime maximă 50, ce reprezintă numele străzii pe care se află respectiva adresă.
- numar: variabilă de tip întreg, de lungime maximă 4, ce reprezintă numărul adresei respective pe stradă adresei (a nu se confunda numărul adresei cu cheia de identificare din cadrul bazei de date).

6) Atributele entității FACTURI:

- cod_factura: variabilă de tip întreg, de lungime maximă 4, ce reprezintă codul facturii.
- cod_adresa: variabilă de tip întreg, de lungime maximă 4, care reprezintă codul adresei pentru care a fost emisă respectiva factură.
- cod_client: variabilă de tip întreg, de lungime maximă 4, care reprezintă codul clientului căruia pentru care a fost emisă această factură.
- total: variabilă de tip întreg, de lungime maximă 4, ce reprezintă suma de bani ce trebuie plătită pentru respectivă factură; valoarea atributului trebuie să fie diferită NULL.
- data_elibărare: variabilă de tip dată calendaristică, ce reprezintă data la care a fost emisă factura; valoarea atributului trebuie să fie diferită de NULL, valoarea implicită a atributului este ziua curentă.
- termen_plata: variabilă de tip dată calendaristică, ce reprezintă data până la care trebuie plătită factură; valoarea atributului trebuie să fie diferită de NULL.
- status: variabilă de tip caracter, de lungime maximă 50, ce reprezintă statusul plății acestei facturi (ex: "Platita", "Neplatita", "Termen depasit"); valoarea atributului trebuie să fie diferită de NULL, valoarea implicită a atributului este "Neplatit".

7) Atributele entității PLANGERI:

- cod_plangere: variabilă de tip întreg, de lungime maximă 4, ce reprezintă codul plangerii.
- cod_client: variabilă de tip întreg, de lungime maximă 4, care reprezintă codul clientului căruia îi aparține această plângere.

- data_plangere: variabilă de tip dată calendaristică, ce reprezintă data la care a fost făcută respectiva plângere; valoarea atributului trebuie să fie diferită de NULL, valoarea implicită a atributului este ziua curentă.

- adresa: variabilă de tip caracter, de lungime maximă 200, ce reprezintă adresa pentru care se face plângerea (ex: "România, București, Strada Preciziei, 34"); valoarea atributului trebuie să fie diferită de NULL.

- mesaj: variabilă de tip caracter, de lungime maximă 2000, ce reprezintă mesajul în care clientul descrie defecțiunea pentru care depune respectiva plângere.

8) Atributele entității MUNCITORI:

- cod_muncitor: variabilă de tip întreg, de lungime maximă 4, ce reprezintă codul muncitorului.

- cod_informatii_muncitor: variabilă de tip întreg, de lungime maximă 4, ce reprezintă codul obiectului ce ne oferă informații personale despre muncitor; atributul este o cheie externă care referențiază cheia primară cod_informatii_muncitor a entității INFORMATII_MUNCITORI; valoarea atributului trebuie să fie UNIQUE.

- salariu: variabilă de tip întreg, de lungime maximă 4, ce reprezintă salariul muncitorului respectiv; valoarea atributului trebuie să fie diferită de NULL.

- data_angajare: variabilă de tip dată calendaristică, ce reprezintă data angajării respectivului angajat; valoarea atributului trebuie să fie diferită de NULL, valoarea implicită a atributului este ziua curentă.

- rating: variabilă de tip întreg, de lungime maximă 4, ce reprezintă ratingul pentru respectivul muncitor (cu cât angajatul prestează o muncă mai calitativă cu atât ratingul este mai mare).

9) Atributele entității INFORMATII_MUNCITORI:

- cod_informatii_muncitor: variabilă de tip întreg, de lungime maximă 4, ce reprezintă codul obiectului ce ne oferă informații personale despre muncitor.

- nume: variabilă de tip caracter, de lungime maximă 50, ce reprezintă numele muncitorului; valoarea atributului trebuie să fie diferită de NULL.

- prenume: variabilă de tip caracter, de lungime maximă 50, ce reprezintă prenumele muncitorului; valoarea atributului trebuie să fie diferită de NULL.

- cnp: variabilă de tip caracter, de lungime maximă 50, ce reprezintă cnp-ul muncitorului; valoarea atributului trebuie să fie diferită de NULL.

- numar_telefon_utilizator: variabilă de tip caracter, de lungime maximă 50, ce reprezintă numărul de telefon al muncitorului; valoarea atributului trebuie să fie diferită de NULL.

10) Atributele entității SPECIALIZARI:

- cod_specializare: variabilă de tip întreg, de lungime maximă 4, ce reprezintă codul specializării.

- nume_specializare: variabilă de tip caracter, de lungime maximă 50, ce reprezintă numele specializării respective (ex: “Sudor”); valoarea atributului trebuie să fie diferită de NULL.

- descriere: variabilă de tip caracter, de lungime maximă 2000, ce reprezintă descrierea specializării respective (ex: pentru sudor: “Persoană care se ocupă cu sudatul țevelor sparte”); valoarea atributului trebuie să fie diferită de NULL.

11) Atributele entității INTERVENTII:

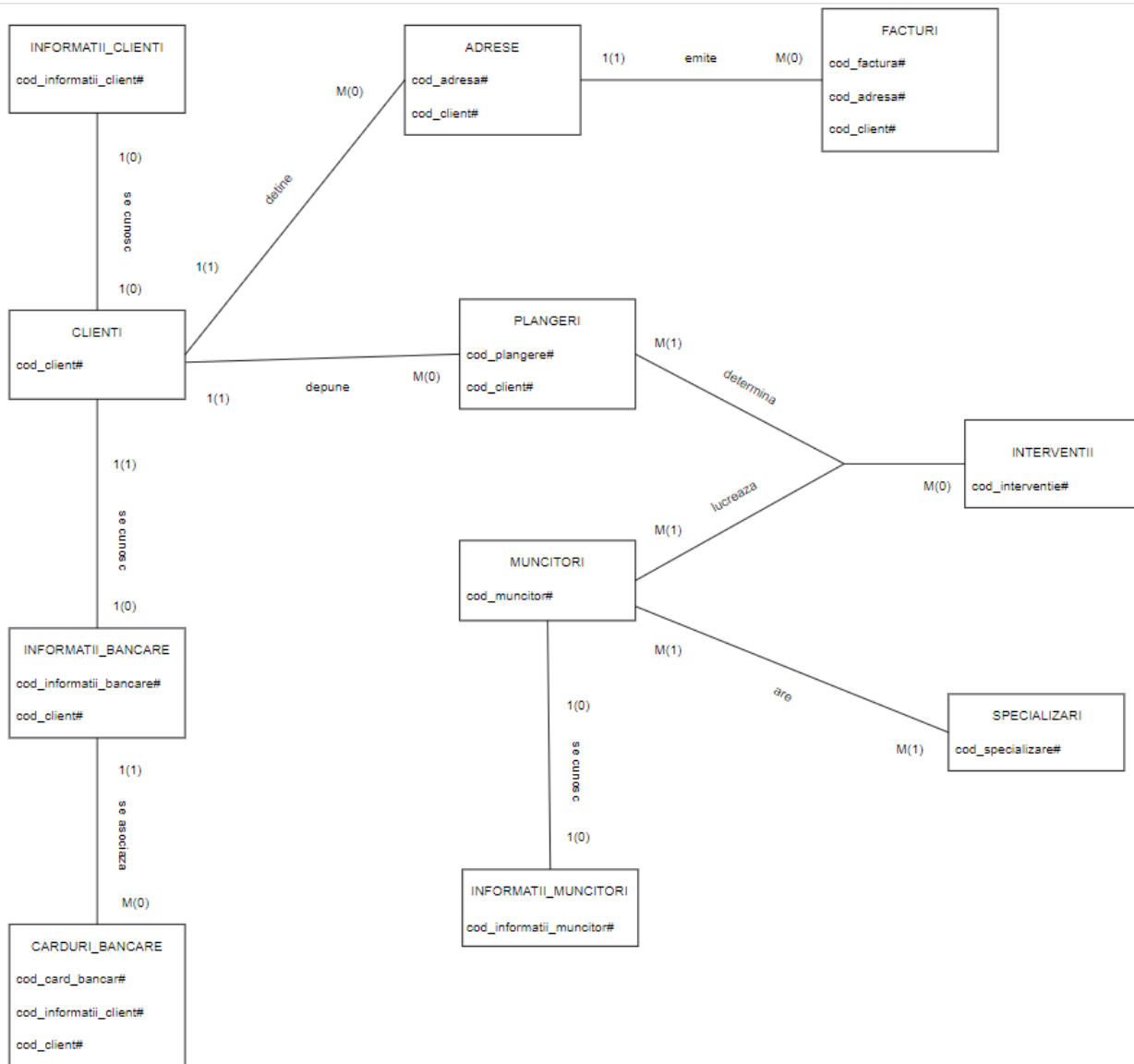
- cod_interventie: variabilă de tip întreg, de lungime maximă 4, ce reprezintă codul intervenției.

- data: variabilă de tip dată calendaristică, ce reprezintă data la care are loc intervenția; valoarea atributului trebuie să fie diferită de NULL.

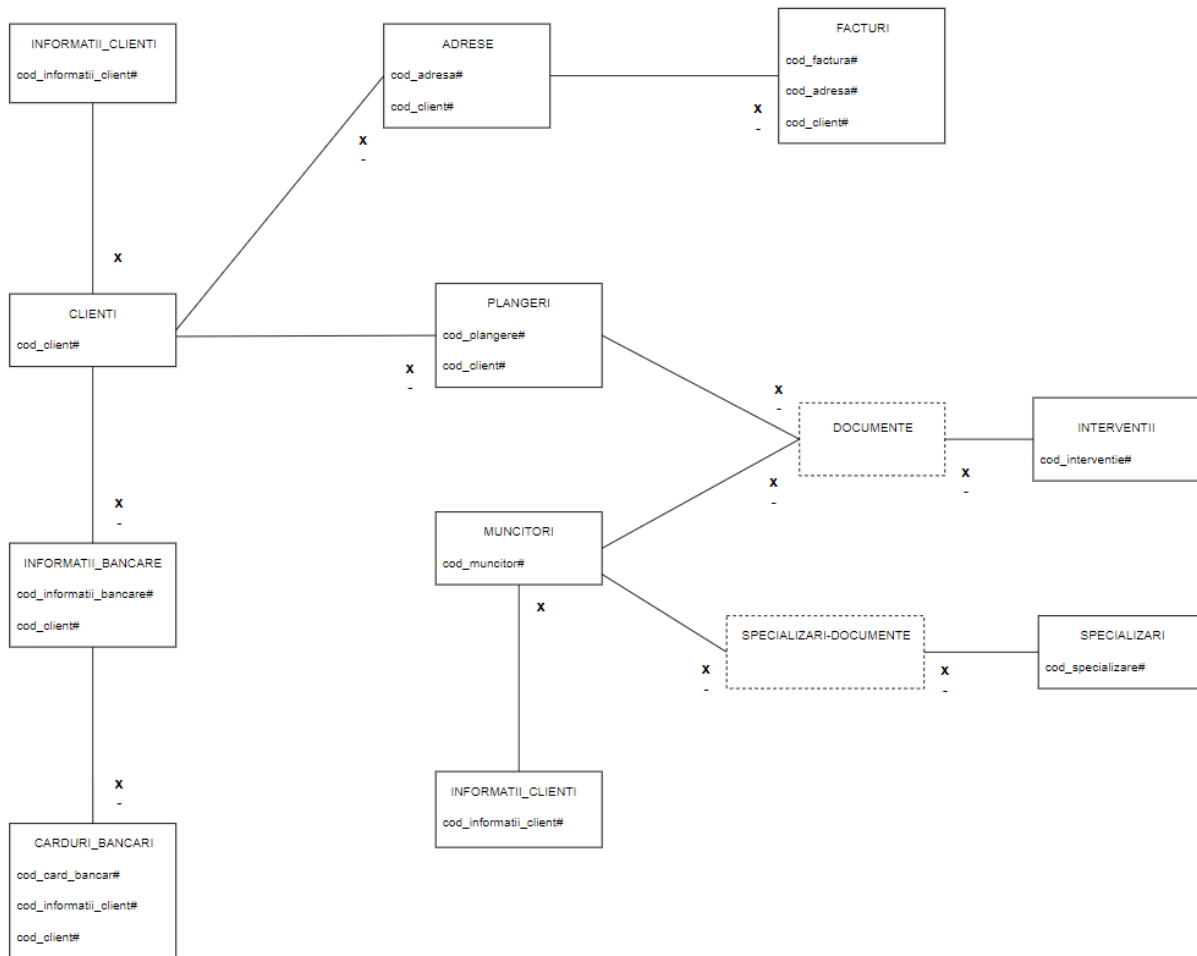
- nume_defectiune: variabilă de tip caracter, de lungime maximă 50, ce reprezintă numele defecțiunii (ex: “Teava sparta”).

- status: variabilă de tip caracter, de lungime maximă 50, ce reprezintă statusul intervenției (ex: “În curs”, “Încheiată”, “Neincepută”); valoarea atributului trebuie să fie diferită de NULL, valoarea implicită a atributului este ‘Neincepută’.

6. Diagrama Entitate/Relatie



7. Diagrama conceptuala



8. Schemele relationale corespunzatoare diagramei conceptuale realizate sunt urmatoarele:

- CLIENTI (cod_client# cod_informatii_client, nume_utilizator, parola, email);
- INFORMATII_CLIENTI (cod_informatii_client#, nume, prenume, cnp, numar_telefon);
- INFORMATII_BANCARE (cod_informatii_bancare#, cod_client#, sold_curent, suma_cheltuita);
- CARDURI_BANCARE (cod_card_bancar#, cod_informatii_bancare#, cod_client#, numar_card, data_expirare_card, cod_securitate_card);
- ADRESE (cod_adresa#, cod_client#, tara, oras, strada, numar);
- FACTURI (cod_factura#, cod_adresa#, cod_client#, total, data_eliberare, termen_plata, status);
- PLANGERI (cod_plangere# , cod_client#, data_plangere, adresa, mesaj);
- MUNCITORI (cod_muncitor#, cod_informatii_muncitor, salariu, data_angajare, rating);

- INFORMATII_MUNCITORI (cod_informatii_muncitor#, nume, prenume, cnp, numar_telefon);
- SPECIALIZARI (cod_specializare#, nume_specializare, descriere);
- INTERVENTII (cod_interventie#, data_incepere, nume_defectiune, status);
- DOCUMENTE (cod_plangere#, cod_muncitor#, cod_interventie#, nume_semnatar, prenume_semnatar);
- SPECIALIZARI_MUNCITORI (cod_muncitor#, cod_specializare#);

9. Realizarea normalizarii:

1) Exemplu de non-FN1 și transformarea acestuia în FN1:

Presupunem că schema relațională a entității MUNCITORI ar fi fost inițial:

MUNCITORI (cod_muncitor#, cod_informatii_muncitor, salariu, data_angajare, rating, specializare)

Conform regulilor 19 și 20 ale modelului, atributul specializare poate avea valori multiple pentru un muncitor, așa că, pentru a aduce modelul la FN1, eliminăm acest atribut și introducem o nouă entitate, SPECIALIZARI. Între entitățile MUNCITORI și SPECIALIZARI apare o relație many-to-many.

2) Exemplu de non-FN2 și transformarea acestuia în FN2:

Presupunem că schema relațională a entității DOCUMENTE ar fi fost inițial:

DOCUMENTE (cod_plangere#, cod_muncitor#, cod_interventie#, nume_semnatar, prenume_semnatar, data_incepere, nume_defectiune, status)

Se observa ca attributele “data_incepere”, “nume_defectiune”, “status” nu depinde de intreaga cheie primara, ci doar de “cod_interventie”. Relatia este in FN1, dar nu este in FN2. Pentru a realiza transformarea în FN2, proiectam în două relații, cu schemele relaționale următoare:

INTERVENTIE (cod_interventie#, data_incepere, nume_defectiune, status)

DOCUMENTE (cod_plangere#, cod_muncitor#, cod_interventie#, nume_semnatar, prenume_semnatar)

3) Exemplu de non-FN3 și transformarea acestuia în FN3:

Presupunem că schema relațională a entității CLIENTI ar fi fost inițial:

CLIENTI (cod_client#, cod_informatii_client, nume_utilizator, parola, email, nume, prenume, cnp, numar_telefon)

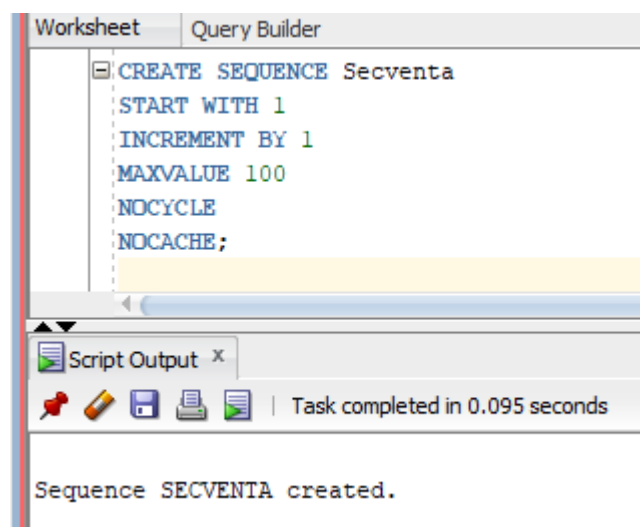
Relație este in FN2 dar nu este in FN3, deoarece attributele “nume”, “prenume”, “cnp”, “numar_telefon”, depind indirect de cheia primară a entitatii INFORMATII_CLIENTI, prin intermediul atributului cod_informatii_client. Pentru a obține o relație în FN3, se proiectează în două relații:

CLIENTI (cod_client#, cod_informatii_client, nume_utilizator, parola, email)

INFORMATII_CLIENTI (cod_informatii_client#, nume, prenume, cnp, numar_telefon)

10. Secvența ce va fi utilizată în inserările în tabele. Această secvență este utilizată pentru inserarea în tabelele INFORMATII_CLIENTI, CLIENTI, INFORMATII_MUNCITORI, SPECIALIZARI și INTERVENTII (Înainte de folosirea în fiecare dintre aceste tabele se dă drop și apoi se creează din nou pentru a reseta valoarea de la 1):

```
CREATE SEQUENCE Secventa  
START WITH 1  
INCREMENT BY 1  
MAXVALUE 100  
NOCYCLE  
NOCACHE;
```

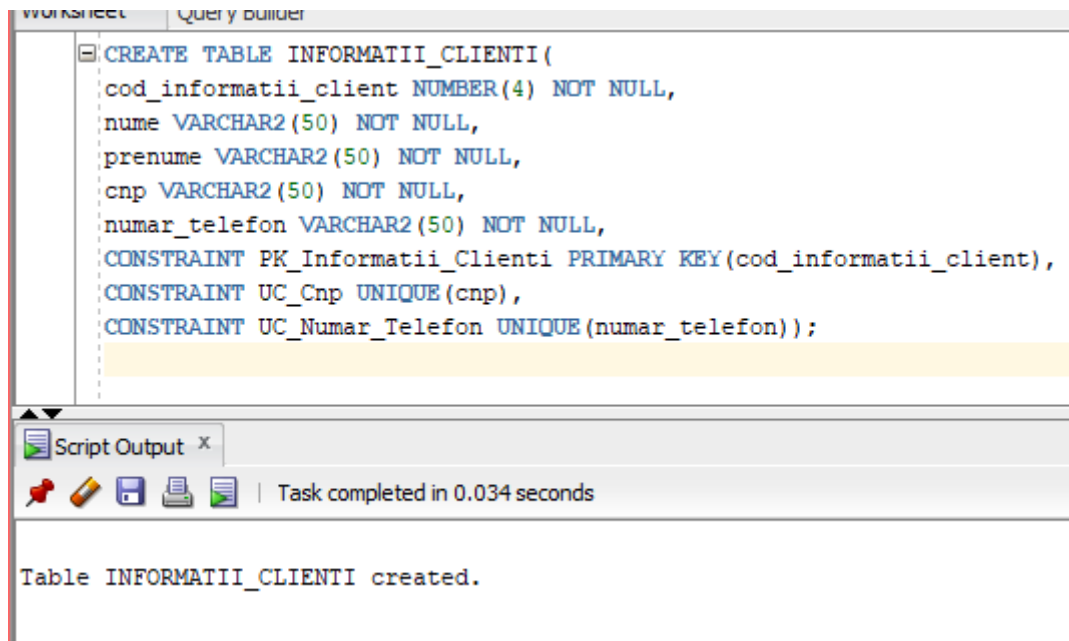


11. Crearea și inserarea în tabele:

- INFORMATII_CLIENTI:

1) Creare:

```
CREATE TABLE INFORMATII_CLIENTI(  
cod_informatii_client NUMBER(4) NOT NULL,  
nume VARCHAR2(50) NOT NULL,  
prenume VARCHAR2(50) NOT NULL,  
cnp VARCHAR2(50) NOT NULL,  
numar_telefon VARCHAR2(50) NOT NULL,  
CONSTRAINT PK_Informatii_Clienti PRIMARY KEY(cod_informatii_client),  
CONSTRAINT UC_Cnp UNIQUE(cnp),  
CONSTRAINT UC_Numar_Telefon UNIQUE(numar_telefon));
```



2) Inserare:

```
INSERT INTO Informatii_Clienti
VALUES (Secventa.nextval, 'Marian', 'Andrei', '5020413150634', '0734135426');
INSERT INTO Informatii_Clienti
VALUES (Secventa.nextval, 'Ion', 'Vasile', '5026262782345', '0774357120');
INSERT INTO Informatii_Clienti
VALUES (Secventa.nextval, 'Ioana', 'Cristina', '5022494157062', '0732584214');
INSERT INTO Informatii_Clienti
VALUES (Secventa.nextval, 'Buzatu', 'Ionut', '5021395602456', '0712690369');
INSERT INTO Informatii_Clienti
VALUES (Secventa.nextval, 'Ana', 'Cosmina', '5021049258568', '0725790632');
INSERT INTO Informatii_Clienti
VALUES (Secventa.nextval, 'Gigel', 'Frone', '5021256073548', '0724679520');
INSERT INTO Informatii_Clienti
VALUES (Secventa.nextval, 'Andreea', 'Maria', '5021249056732', '0724905687');
INSERT INTO Informatii_Clienti
VALUES (Secventa.nextval, 'Cristi', 'Sebastian', '5042458037893', '0738952469');
INSERT INTO Informatii_Clienti
VALUES (Secventa.nextval, 'Dragos', 'Andrei', '5015730458972', '0739503182');
INSERT INTO Informatii_Clienti
VALUES (Secventa.nextval, 'Ionut', 'Dima', '5072490567120', '0719023567');
INSERT INTO Informatii_Clienti
VALUES (Secventa.nextval, 'Ion', 'Matei', '5073905481235', '0709326784');
```

```

INSERT INTO Informatii_Clienti
VALUES (Secventa.nextval, 'Marian', 'Andrei', '5020413150634', '0734135426');
INSERT INTO Informatii_Clienti
VALUES (Secventa.nextval, 'Ion', 'Vasile', '5026262782345', '0774357120');
INSERT INTO Informatii_Clienti
VALUES (Secventa.nextval, 'Ioana', 'Cristina', '5022494157062', '0732584214');
INSERT INTO Informatii_Clienti
VALUES (Secventa.nextval, 'Buzatu', 'Ionut', '5021395602456', '0712690369');
INSERT INTO Informatii_Clienti
VALUES (Secventa.nextval, 'Ana', 'Cosmina', '5021049258568', '0725790632');
INSERT INTO Informatii_Clienti
VALUES (Secventa.nextval, 'Gigel', 'Frone', '5021256073548', '0724679520');
INSERT INTO Informatii_Clienti
VALUES (Secventa.nextval, 'Andreea', 'Maria', '5021249056732', '0724905687');
INSERT INTO Informatii_Clienti
VALUES (Secventa.nextval, 'Cristi', 'Sebastian', '5042458037893', '0738952469');
INSERT INTO Informatii_Clienti
VALUES (Secventa.nextval, 'Dragos', 'Andrei', '5015730458972', '0739503182');
INSERT INTO Informatii_Clienti
VALUES (Secventa.nextval, 'Ionut', 'Dima', '5072490567120', '0719023567');
INSERT INTO Informatii_Clienti
VALUES (Secventa.nextval, 'Ion', 'Matei', '5073905481235', '0709326784');

SELECT *
FROM Informatii_Clienti;

```

Script Output x Query Result x					
SQL All Rows Fetched: 11 in 0.002 seconds					
	COD_INFORMATII_CLIENT	NUME	PRENUME	CNP	NUMAR_TELEFON
1		1 Marian	Andrei	5020413150634	0734135426
2		2 Ion	Vasile	5026262782345	0774357120
3		3 Ioana	Cristina	5022494157062	0732584214
4		4 Buzatu	Ionut	5021395602456	0712690369
5		5 Ana	Cosmina	5021049258568	0725790632
6		6 Gigel	Frone	5021256073548	0724679520
7		7 Andreea	Maria	5021249056732	0724905687
8		8 Cristi	Sebastian	5042458037893	0738952469
9		9 Dragos	Andrei	5015730458972	0739503182
10		10 Ionut	Dima	5072490567120	0719023567
11		11 Ion	Matei	5073905481235	0709326784

- CLIENTI:

1) Creare:

```

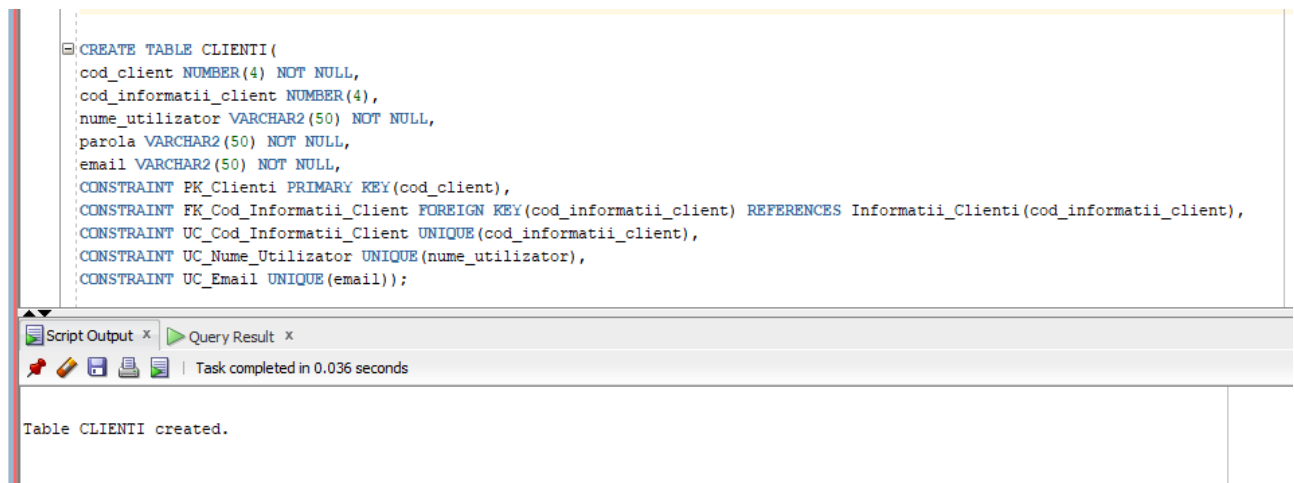
CREATE TABLE CLIENTI(
cod_client NUMBER(4) NOT NULL,
cod_informatii_client NUMBER(4),

```

```

nume_utilizator VARCHAR2(50) NOT NULL,
parola VARCHAR2(50) NOT NULL,
email VARCHAR2(50) NOT NULL,
CONSTRAINT PK_Clienti PRIMARY KEY(cod_client),
CONSTRAINT FK_Cod_Informatii_Client FOREIGN KEY(cod_informatii_client)
REFERENCES Informatii_Clienti(cod_informatii_client),
CONSTRAINT UC_Cod_Informatii_Client UNIQUE(cod_informatii_client),
CONSTRAINT UC_Nume_Utilizator UNIQUE(nume_utilizator),
CONSTRAINT UC_Email UNIQUE(email));

```



The screenshot shows a database IDE with a script editor and a results pane. The script editor contains the following SQL code:

```

CREATE TABLE CLIENTI(
  cod_client NUMBER(4) NOT NULL,
  cod_informatii_client NUMBER(4),
  nume_utilizator VARCHAR2(50) NOT NULL,
  parola VARCHAR2(50) NOT NULL,
  email VARCHAR2(50) NOT NULL,
  CONSTRAINT PK_Clienti PRIMARY KEY(cod_client),
  CONSTRAINT FK_Cod_Informatii_Client FOREIGN KEY(cod_informatii_client) REFERENCES Informatii_Clienti(cod_informatii_client),
  CONSTRAINT UC_Cod_Informatii_Client UNIQUE(cod_informatii_client),
  CONSTRAINT UC_Nume_Utilizator UNIQUE(nume_utilizator),
  CONSTRAINT UC_Email UNIQUE(email));

```

The results pane shows the message: "Table CLIENTI created." and "Task completed in 0.036 seconds".

2) Inserare:

```

INSERT INTO Clienti
VALUES (Secventa.nextval, 1, 'marian08', 'parolamarian', 'marian08@yahoo.com');
INSERT INTO Clienti
VALUES (Secventa.nextval, NULL, 'john_vasy', '12345678', 'vasy_john@outlook.ro');
INSERT INTO Clienti
VALUES (Secventa.nextval, 3, 'Icris', 'crisspass', 'criss@gmail.com');
INSERT INTO Clienti
VALUES (Secventa.nextval, 4, 'buzion', 'parola', 'buzion23@s.unibuc.ro');
INSERT INTO Clienti
VALUES (Secventa.nextval, 5, 'anacos', 'cos04322', 'ana@gmail.com');
INSERT INTO Clienti
VALUES (Secventa.nextval, NULL, 'dimaI', 'A45fFD43', 'ion004@yahoo.com');
INSERT INTO Clienti
VALUES (Secventa.nextval, 9, 'dragos244', 'parolaMea', 'dragos244@s.unibuc.ro');
INSERT INTO Clienti
VALUES (Secventa.nextval, 8, 'seby', 'sebyyy435', 'seb@yahoo.com');
INSERT INTO Clienti
VALUES (Secventa.nextval, 7, 'andreeaMaria', '987654321', 'maria_andreea@gmail.ro');
INSERT INTO Clienti

```

VALUES (Secventa.nextval, NULL, 'kingFrone', 'gigel34', 'frone34@outlook.com');

INSERT INTO Clienti

VALUES (Secventa.nextval, 2, 'mexicanu', 'meml34', 'mexicanu@outlook.com');

INSERT INTO Clienti

VALUES (Secventa.nextval, 11, 'MAtei321', '43mat', 'matei@outlook.com');

```
INSERT INTO Clienti
VALUES (Secventa.nextval, 1, 'marian08', 'parolamarian', 'marian08@yahoo.com');
INSERT INTO Clienti
VALUES (Secventa.nextval, NULL, 'john_vasy', '12345678', 'vasy_john@outlook.ro');
INSERT INTO Clienti
VALUES (Secventa.nextval, 3, 'Icris', 'crisspass', 'criss@gmail.com');
INSERT INTO Clienti
VALUES (Secventa.nextval, 4, 'buzion', 'parola', 'buzion23@s.unibuc.ro');
INSERT INTO Clienti
VALUES (Secventa.nextval, 5, 'anacos', 'cos04322', 'ana@gmail.com');
INSERT INTO Clienti
VALUES (Secventa.nextval, NULL, 'dimaI', 'A45fFD43', 'ion004@yahoo.com');
INSERT INTO Clienti
VALUES (Secventa.nextval, 9, 'dragos244', 'parolaMea', 'dragos244@s.unibuc.ro');
INSERT INTO Clienti
VALUES (Secventa.nextval, 8, 'seby', 'sebyyy435', 'seb@yahoo.com');
INSERT INTO Clienti
VALUES (Secventa.nextval, 7, 'andreeaMaria', '987654321', 'maria_andreea@gmail.ro');
INSERT INTO Clienti
VALUES (Secventa.nextval, NULL, 'kingFrone', 'gigel34', 'frone34@outlook.com');
INSERT INTO Clienti
VALUES (Secventa.nextval, 2, 'mexicanu', 'meml34', 'mexicanu@outlook.com');
INSERT INTO Clienti
VALUES (Secventa.nextval, 11, 'MAtei321', '43mat', 'matei@outlook.com');

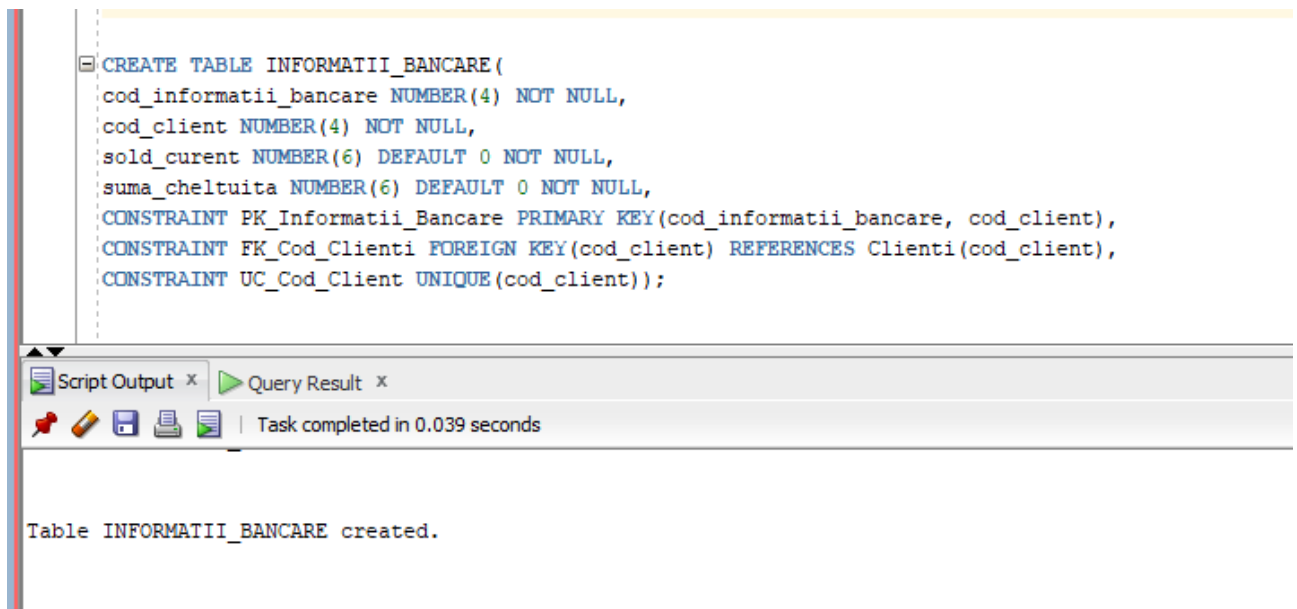
SELECT *
FROM Clienti;
```

Script Output x Query Result x					
SQL All Rows Fetched: 12 in 0.001 seconds					
	COD_CLIENT	COD_INFORMATII_CLIENT	NUME_UTILIZATOR	PAROLA	EMAIL
1	1		marian08	parolamarian	marian08@yahoo.com
2	2	(null)	john_vasy	12345678	vasy_john@outlook.ro
3	3	3	Icris	crisspass	criss@gmail.com
4	4	4	buzion	parola	buzion23@s.unibuc.ro
5	5	5	anacos	cos04322	ana@gmail.com
6	6	(null)	dimaI	A45fFD43	ion004@yahoo.com
7	7	9	dragos244	parolaMea	dragos244@s.unibuc.ro
8	8	8	seby	sebyyy435	seb@yahoo.com
9	9	7	andreeaMaria	987654321	maria_andreea@gmail.ro
10	10	(null)	kingFrone	gigel34	frone34@outlook.com
11	11	2	mexicanu	meml34	mexicanu@outlook.com
12	12	11	MAtei321	43mat	matei@outlook.com

- INFORMATII_BANCARE:

1) Creare:

```
CREATE TABLE INFORMATII_BANCARE(  
cod_informatii_bancare NUMBER(4) NOT NULL,  
cod_client NUMBER(4) NOT NULL,  
sold_curent NUMBER(6) DEFAULT 0 NOT NULL,  
suma_cheltuita NUMBER(6) DEFAULT 0 NOT NULL,  
CONSTRAINT PK_Informatii_Bancare PRIMARY KEY(cod_informatii_bancare, cod_client),  
CONSTRAINT FK_Cod_Clienti FOREIGN KEY(cod_client) REFERENCES Clienti(cod_client),  
CONSTRAINT UC_Cod_Client UNIQUE(cod_client));
```



2) Inserare:

```
INSERT INTO informatii_bancare  
VALUES (1, 1, 200, 150);  
INSERT INTO informatii_bancare  
VALUES (2, 2, 300, 200);  
INSERT INTO informatii_bancare (cod_informatii_bancare, cod_client)  
VALUES (3, 3);  
INSERT INTO informatii_bancare  
VALUES (4, 10, 100, 400);  
INSERT INTO informatii_bancare  
VALUES (5, 9, 500, 270);
```

```
INSERT INTO informatii_bancare (cod_informatii_bancare, cod_client)
VALUES (6, 8);
INSERT INTO informatii_bancare
VALUES (7, 7, 850, 1300);
INSERT INTO informatii_bancare
VALUES (8, 6, 2500, 2000);
INSERT INTO informatii_bancare (cod_informatii_bancare, cod_client)
VALUES (9, 5);
INSERT INTO informatii_bancare
VALUES (10, 4, 2700, 4000);
INSERT INTO informatii_bancare
VALUES (11, 12, 2400, 3500);
```

```

INSERT INTO informatii_bancare
VALUES (1, 1, 200, 150);
INSERT INTO informatii_bancare
VALUES (2, 2, 300, 200);
INSERT INTO informatii_bancare (cod_informatii_bancare, cod_client)
VALUES (3, 3);
INSERT INTO informatii_bancare
VALUES (4, 10, 100, 400);
INSERT INTO informatii_bancare
VALUES (5, 9, 500, 270);
INSERT INTO informatii_bancare (cod_informatii_bancare, cod_client)
VALUES (6, 8);
INSERT INTO informatii_bancare
VALUES (7, 7, 850, 1300);
INSERT INTO informatii_bancare
VALUES (8, 6, 2500, 2000);
INSERT INTO informatii_bancare (cod_informatii_bancare, cod_client)
VALUES (9, 5);
INSERT INTO informatii_bancare
VALUES (10, 4, 2700, 4000);
INSERT INTO informatii_bancare
VALUES (11, 12, 2400, 3500);

SELECT *
FROM Informatii_Bancare;

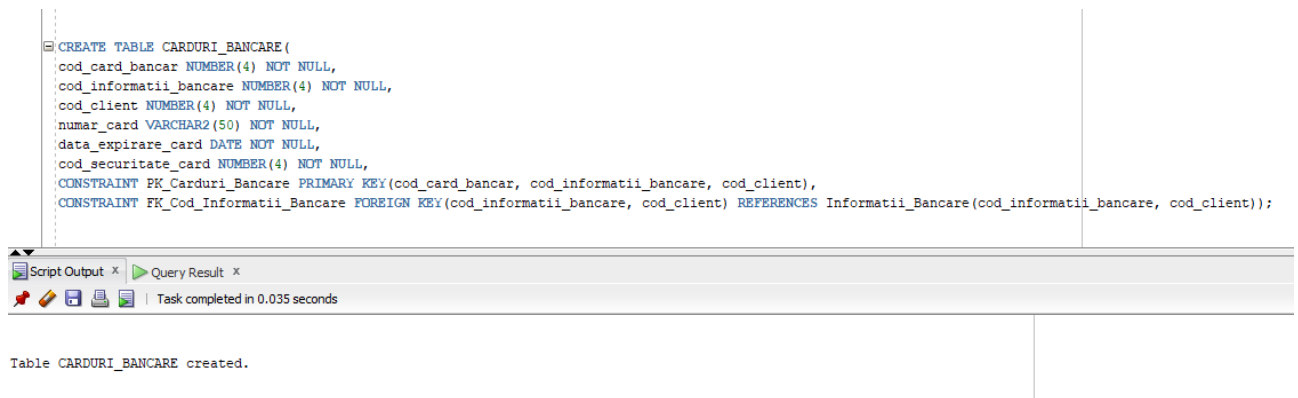
```

Script Output x Query Result x				
SQL All Rows Fetched: 11 in 0.002 seconds				
	COD_INFORMATII_BANCARE	COD_CLIENT	SOLD_CURENT	SUMA_CHELTUITA
1	1	1	200	150
2	2	2	300	200
3	3	3	0	0
4	4	10	100	400
5	5	9	500	270
6	6	8	0	0
7	7	7	850	1300
8	8	6	2500	2000
9	9	5	0	0
10	10	4	2700	4000
11	11	12	2400	3500

- CARDURI_BANCARE:

1) Creare:

```
CREATE TABLE CARDURI_BANCARE(  
cod_card_bancar NUMBER(4) NOT NULL,  
cod_informatii_bancare NUMBER(4) NOT NULL,  
cod_client NUMBER(4) NOT NULL,  
numar_card VARCHAR2(50) NOT NULL,  
data_expirare_card DATE NOT NULL,  
cod_securitate_card NUMBER(4) NOT NULL,  
CONSTRAINT PK_Carduri_Bancare PRIMARY KEY(cod_card_bancar, cod_informatii_bancare,  
cod_client),  
CONSTRAINT FK_Cod_Informatii_Bancare FOREIGN KEY(cod_informatii_bancare, cod_client)  
REFERENCES Informatii_Bancare(cod_informatii_bancare, cod_client));
```



2) Inserare:

```
INSERT INTO Carduri_Bancare  
VALUES (1, 1, 1, '4053123405931285', TO_DATE('23-MAR-2027', 'DD-MON-YYYY'), 643);  
INSERT INTO Carduri_Bancare  
VALUES (2, 2, 2, '5902345967102923', TO_DATE('03-APR-2028', 'DD-MON-YYYY'), 346);  
INSERT INTO Carduri_Bancare  
VALUES (3, 3, 3, '5678765321426541', TO_DATE('14-DEC-2025', 'DD-MON-YYYY'), 964);  
INSERT INTO Carduri_Bancare  
VALUES (4, 4, 10, '876541234765423', TO_DATE('18-FEB-2024', 'DD-MON-YYYY'), 245);  
INSERT INTO Carduri_Bancare  
VALUES (5, 5, 9, '2134357568505663', TO_DATE('22-AUG-2029', 'DD-MON-YYYY'), 325);  
INSERT INTO Carduri_Bancare  
VALUES (6, 6, 8, '7809675421345633', TO_DATE('27-SEP-2025', 'DD-MON-YYYY'), 234);  
INSERT INTO Carduri_Bancare  
VALUES (7, 7, 7, '1242365406784353', TO_DATE('22-DEC-2025', 'DD-MON-YYYY'), 641);  
INSERT INTO Carduri_Bancare  
VALUES (8, 8, 6, '1234326347314746', TO_DATE('06-OCT-2030', 'DD-MON-YYYY'), 123);  
INSERT INTO Carduri_Bancare
```

```
VALUES (9, 9, 5, '7437217457897523', TO_DATE('08-NOV-2023', 'DD-MON-YYYY'), 435);
INSERT INTO Carduri_Bancare
VALUES (10, 10, 4, '3242356343473442', TO_DATE('17-APR-2028', 'DD-MON-YYYY'), 543);
INSERT INTO Carduri_Bancare
VALUES (11, 11, 12, '3202356343473569', TO_DATE('26-DEC-2030', 'DD-MON-YYYY'), 206);
```

```
INSERT INTO Carduri_Bancare
VALUES (1, 1, 1, '4053123405931285', TO_DATE('23-MAR-2027', 'DD-MON-YYYY'), 643);
INSERT INTO Carduri_Bancare
VALUES (2, 2, 2, '5902345967102923', TO_DATE('03-APR-2028', 'DD-MON-YYYY'), 346);
INSERT INTO Carduri_Bancare
VALUES (3, 3, 3, '5678765321426541', TO_DATE('14-DEC-2025', 'DD-MON-YYYY'), 964);
INSERT INTO Carduri_Bancare
VALUES (4, 4, 10, '876541234765423', TO_DATE('18-FEB-2024', 'DD-MON-YYYY'), 245);
INSERT INTO Carduri_Bancare
VALUES (5, 5, 9, '2134357568505663', TO_DATE('22-AUG-2029', 'DD-MON-YYYY'), 325);
INSERT INTO Carduri_Bancare
VALUES (6, 6, 8, '7809675421345633', TO_DATE('27-SEP-2025', 'DD-MON-YYYY'), 234);
INSERT INTO Carduri_Bancare
VALUES (7, 7, 7, '1242365406784353', TO_DATE('22-DEC-2025', 'DD-MON-YYYY'), 641);
INSERT INTO Carduri_Bancare
VALUES (8, 8, 6, '1234326347314746', TO_DATE('06-OCT-2030', 'DD-MON-YYYY'), 123);
INSERT INTO Carduri_Bancare
VALUES (9, 9, 5, '7437217457897523', TO_DATE('08-NOV-2023', 'DD-MON-YYYY'), 435);
INSERT INTO Carduri_Bancare
VALUES (10, 10, 4, '3242356343473442', TO_DATE('17-APR-2028', 'DD-MON-YYYY'), 543);
INSERT INTO Carduri_Bancare
VALUES (11, 11, 12, '3202356343473569', TO_DATE('26-DEC-2030', 'DD-MON-YYYY'), 206);

SELECT *
FROM Carduri_Bancare;
```

	COD_CARD_BANCAR	COD_INFORMATII_BANCARE	COD_CLIENT	NUMAR_CARD	DATA_EXPIRARE_CARD	COD_SECURITATE_CARD
1	1	1	1	4053123405931285	23-MAR-27	643
2	2	2	2	5902345967102923	03-APR-28	346
3	3	3	3	5678765321426541	14-DEC-25	964
4	4	10	4	876541234765423	18-FEB-24	245
5	5	9	5	2134357568505663	22-AUG-29	325
6	6	8	6	7809675421345633	27-SEP-25	234
7	7	7	7	1242365406784353	22-DEC-25	641
8	8	6	8	1234326347314746	06-OCT-30	123
9	9	5	9	7437217457897523	08-NOV-23	435
10	10	4	10	3242356343473442	17-APR-28	543
11	11	12	11	3202356343473569	26-DEC-30	206

- ADRESE:

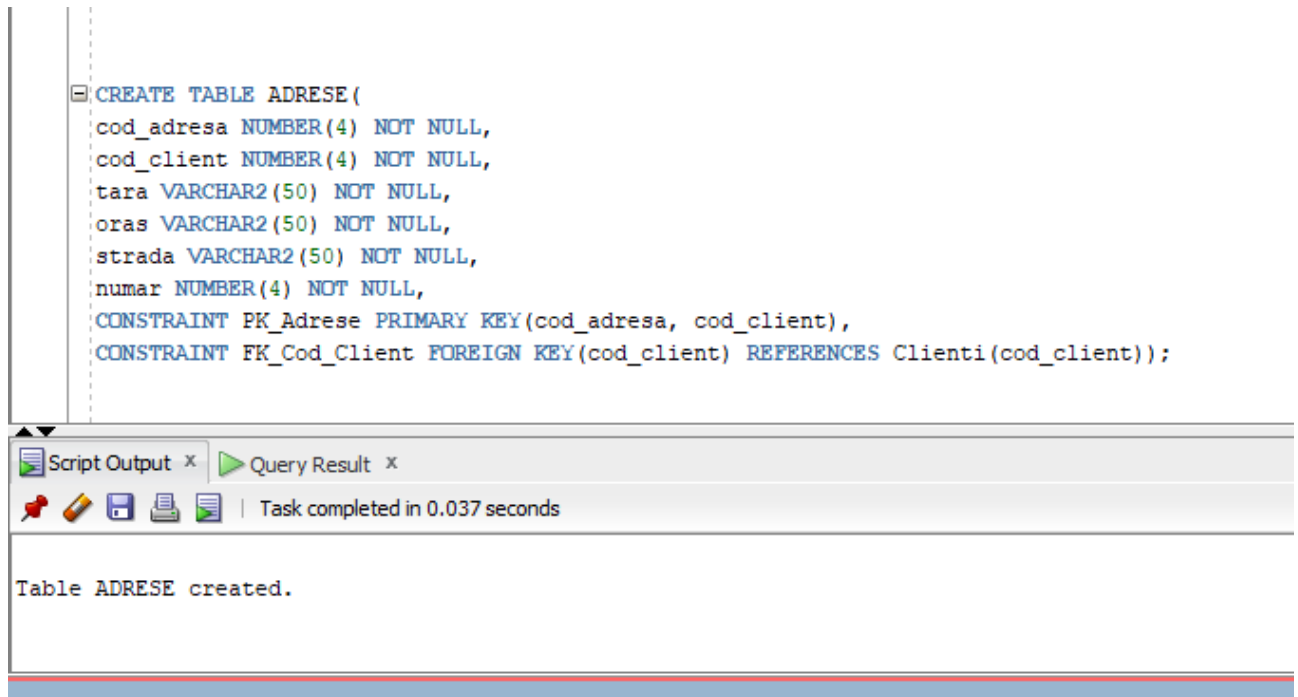
1) Creare:

```
CREATE TABLE ADRESE(
cod_adresa NUMBER(4) NOT NULL,
```

```

cod_client NUMBER(4) NOT NULL,
tara VARCHAR2(50) NOT NULL,
oras VARCHAR2(50) NOT NULL,
strada VARCHAR2(50) NOT NULL,
numar NUMBER(4) NOT NULL,
CONSTRAINT PK_Adrese PRIMARY KEY(cod_adresa, cod_client),
CONSTRAINT FK_Cod_Client FOREIGN KEY(cod_client) REFERENCES Clienti(cod_client));

```



The screenshot shows a database IDE interface. The top pane displays a SQL script for creating a table named ADRESE. The script includes column definitions for cod_adresa, cod_client, tara, oras, strada, and numar, along with a primary key constraint (PK_Adrese) and a foreign key constraint (FK_Cod_Client). The bottom pane shows the 'Script Output' tab with the message 'Table ADRESE created.' and a status bar indicating 'Task completed in 0.037 seconds'.

```

CREATE TABLE ADRESE (
  cod_adresa NUMBER(4) NOT NULL,
  cod_client NUMBER(4) NOT NULL,
  tara VARCHAR2(50) NOT NULL,
  oras VARCHAR2(50) NOT NULL,
  strada VARCHAR2(50) NOT NULL,
  numar NUMBER(4) NOT NULL,
  CONSTRAINT PK_Adrese PRIMARY KEY(cod_adresa, cod_client),
  CONSTRAINT FK_Cod_Client FOREIGN KEY(cod_client) REFERENCES Clienti(cod_client));

```

Script Output x Query Result x

Task completed in 0.037 seconds

Table ADRESE created.

2) Inserare:

```

INSERT INTO Adrese
VALUES (1, 1, 'Romania', 'Craiova', 'Brestei', 56);
INSERT INTO Adrese
VALUES (2, 1, 'Romania', 'Bucuresti', 'Drumul Taberei', 23);
INSERT INTO Adrese
VALUES (3, 1, 'Olanda', 'Amsterdam', 'Amsterdam Street', 4535);
INSERT INTO Adrese
VALUES (1, 4, 'Romania', 'Iasi', 'Strada Iasului', 654);
INSERT INTO Adrese
VALUES (2, 4, 'Romania', 'Timisioara', 'Aleea Actorilor', 547);
INSERT INTO Adrese
VALUES (1, 7, 'Republica Moldova', 'Chisinau', 'Strada Mosilor', 234);
INSERT INTO Adrese
VALUES (1, 8, 'Franta', 'Paris', 'Sans Elise', 1235);
INSERT INTO Adrese

```

```
VALUES (1, 10, 'Anglia', 'Londra', 'Fournier Street', 12);
INSERT INTO Adrese
VALUES (2, 10, 'Romania', 'Constanta', 'Faleza Marii', 634);
INSERT INTO Adrese
VALUES (3, 10, 'Romania', 'Cluj', 'Strada Clujului', 8765);
INSERT INTO Adrese
VALUES (1, 2, 'Romania', 'Arad', 'Strda Aradului', 65);
INSERT INTO Adrese
VALUES (2, 2, 'Romania', 'Bucuresti', 'Crangasi', 654);
INSERT INTO Adrese
VALUES (1, 9, 'Romania', 'Buzau', 'Strda Buzaului', 23);
INSERT INTO Adrese
VALUES (1, 11, 'Romania', 'Botosani', 'Strda Botosani', 653);
INSERT INTO Adrese
VALUES (1, 12, 'Romania', 'Vaslui', 'Strda Vslui', 13);
```

worksneet

Query Builder

```

VALUES (2, 1, 'Romania', 'Bucuresti', 'Strada Iasului', 23);
INSERT INTO Adrese
VALUES (3, 1, 'Olanda', 'Amsterdam', 'Amsterdam Street', 4535);
INSERT INTO Adrese
VALUES (1, 4, 'Romania', 'Iasi', 'Strada Iasului', 654);
INSERT INTO Adrese
VALUES (2, 4, 'Romania', 'Timisioara', 'Aleea Actorilor', 547);
INSERT INTO Adrese
VALUES (1, 7, 'Republica Moldova', 'Chisinau', 'Strada Mosilor', 234);
INSERT INTO Adrese
VALUES (1, 8, 'Franta', 'Paris', 'Sans Elise', 1235);
INSERT INTO Adrese
VALUES (1, 10, 'Anglia', 'Londra', 'Fournier Street', 12);
INSERT INTO Adrese
VALUES (2, 10, 'Romania', 'Constanta', 'Faleza Marii', 634);
INSERT INTO Adrese
VALUES (3, 10, 'Romania', 'Cluj', 'Strada Clujului', 8765);
INSERT INTO Adrese
VALUES (1, 2, 'Romania', 'Arad', 'Strda Aradului', 65);
INSERT INTO Adrese
VALUES (2, 2, 'Romania', 'Bucuresti', 'Crangasi', 654);
INSERT INTO Adrese
VALUES (1, 9, 'Romania', 'Buzau', 'Strda Buzaului', 23);
INSERT INTO Adrese
VALUES (1, 11, 'Romania', 'Botosani', 'Strda Botosani', 653);
INSERT INTO Adrese
VALUES (1, 12, 'Romania', 'Vaslui', 'Strda Vslui', 13);

SELECT *
FROM Adrese;

```

Script Output x

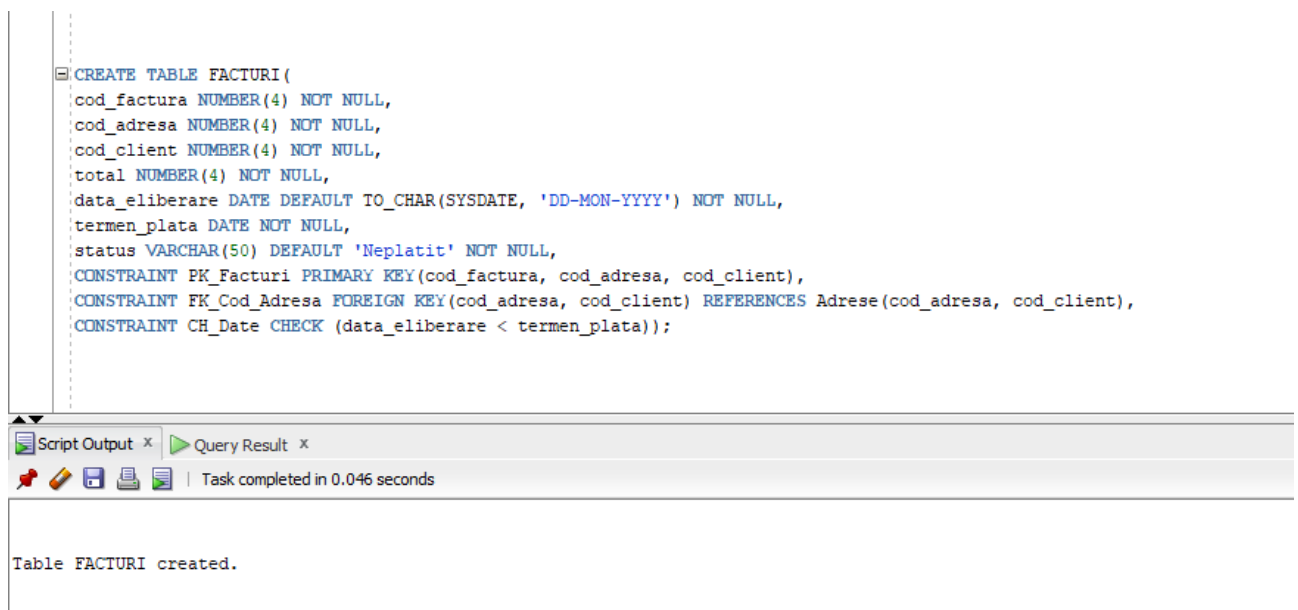
Query Result x

	COD_ADRESA	COD_CLIENT	TARA	ORAS	STRADA	NUMAR
1	1	4	Romania	Iasi	Strada Iasului	654
5	2	4	Romania	Timisioara	Aleea Actorilor	547
6	1	7	Republica Moldova	Chisinau	Strada Mosilor	234
7	1	8	Franta	Paris	Sans Elise	1235
8	1	10	Anglia	Londra	Fournier Street	12
9	2	10	Romania	Constanta	Faleza Marii	634
10	3	10	Romania	Cluj	Strada Clujului	8765
11	1	2	Romania	Arad	Strda Aradului	65
12	2	2	Romania	Bucuresti	Crangasi	654
13	1	9	Romania	Buzau	Strda Buzaului	23
14	1	11	Romania	Botosani	Strda Botosani	653
15	1	12	Romania	Vaslui	Strda Vslui	13

- FACTURI:

1) Creare:

```
CREATE TABLE FACTURI(  
cod_factura NUMBER(4) NOT NULL,  
cod_adresa NUMBER(4) NOT NULL,  
cod_client NUMBER(4) NOT NULL,  
total NUMBER(4) NOT NULL,  
data_eliberare DATE DEFAULT TO_CHAR(SYSDATE, 'DD-MON-YYYY') NOT NULL,  
termen_plata DATE NOT NULL,  
status VARCHAR(50) DEFAULT 'Neplatit' NOT NULL,  
CONSTRAINT PK_Facturi PRIMARY KEY(cod_factura, cod_adresa, cod_client),  
CONSTRAINT FK_Cod_Adresa FOREIGN KEY(cod_adresa, cod_client) REFERENCES  
Adrese(cod_adresa, cod_client),  
CONSTRAINT CH_Date CHECK (data_eliberare < termen_plata));
```



2) Inserare:

```
INSERT INTO Facturi  
VALUES (1, 1, 1, 200, TO_DATE('18-FEB-2024', 'DD-MON-YYYY'), TO_DATE('23-FEB-2025',  
'DD-MON-YYYY'), 'Platit');  
INSERT INTO Facturi  
VALUES (2, 1, 1, 350, TO_DATE('04-MAR-2023', 'DD-MON-YYYY'),  
TO_DATE('15-APR-2024', 'DD-MON-YYYY'), 'In procesare');  
INSERT INTO Facturi  
VALUES (3, 1, 1, 570, TO_DATE('20-SEP-2022', 'DD-MON-YYYY'), TO_DATE('01-APR-2026',  
'DD-MON-YYYY'), 'Platit');
```

```

INSERT INTO Facturi (cod_factura, cod_adresa, cod_client, total, data_eliberare, termen_plata)
VALUES (1, 2, 4, 700, TO_DATE('13-DEC-2022', 'DD-MON-YYYY'), TO_DATE('17-APR-2024',
'DD-MON-YYYY'));
INSERT INTO Facturi (cod_factura, cod_adresa, cod_client, total, data_eliberare, termen_plata)
VALUES (2, 2, 4, 620, TO_DATE('07-AUG-2024', 'DD-MON-YYYY'),
TO_DATE('15-APR-2028', 'DD-MON-YYYY'));
INSERT INTO Facturi
VALUES (1, 1, 7, 200, TO_DATE('23-FEB-2023', 'DD-MON-YYYY'), TO_DATE('27-OCT-2027',
'DD-MON-YYYY'), 'Neplatit');
INSERT INTO Facturi
VALUES (2, 1, 7, 850, TO_DATE('20-AUG-2023', 'DD-MON-YYYY'),
TO_DATE('27-APR-2025', 'DD-MON-YYYY'), 'In procesare');
INSERT INTO Facturi
VALUES (1, 2, 10,340, TO_DATE('14-JAN-2023', 'DD-MON-YYYY'),
TO_DATE('19-NOV-2023', 'DD-MON-YYYY'), 'Platit');
INSERT INTO Facturi (cod_factura, cod_adresa, cod_client, total, data_eliberare, termen_plata)
VALUES (2, 2, 10, 890, TO_DATE('27-DEC-2024', 'DD-MON-YYYY'),
TO_DATE('08-APR-2027', 'DD-MON-YYYY'));
INSERT INTO Facturi (cod_factura, cod_adresa, cod_client, total, data_eliberare, termen_plata)
VALUES (3, 2, 10, 510, TO_DATE('03-APR-2023', 'DD-MON-YYYY'),
TO_DATE('22-APR-2025', 'DD-MON-YYYY'));
INSERT INTO Facturi
VALUES (1, 1, 2, 280, TO_DATE('24-SEP-2020', 'DD-MON-YYYY'), TO_DATE('27-APR-2022',
'DD-MON-YYYY'), 'Neplatit');
INSERT INTO Facturi
VALUES (1, 1, 9, 320, TO_DATE('14-DEC-2022', 'DD-MON-YYYY'),
TO_DATE('25-MAR-2025', 'DD-MON-YYYY'), 'Neplatit');
INSERT INTO Facturi
VALUES (1, 1, 11, 765, TO_DATE('05-FEB-2022', 'DD-MON-YYYY'),
TO_DATE('02-MAY-2024', 'DD-MON-YYYY'), 'Neplatit');
INSERT INTO Facturi
VALUES (1, 1, 12, 180, TO_DATE('19-APR-2021', 'DD-MON-YYYY'),
TO_DATE('02-APR-2022', 'DD-MON-YYYY'), 'Neplatit');

```

Worksheet Query Builder

```
VALUES (2, 1, 1, 350, TO_DATE('04-MAR-2023', 'DD-MON-YYYY'), TO_DATE('15-APR-2024', 'DD-MON-YYYY'), 'In procesare');
INSERT INTO Facturi
VALUES (3, 1, 1, 570, TO_DATE('20-SEP-2022', 'DD-MON-YYYY'), TO_DATE('01-APR-2026', 'DD-MON-YYYY'), 'Platit');
INSERT INTO Facturi (cod_factura, cod_adresa, cod_client, total, data_eliberare, termen_plata)
VALUES (1, 2, 4, 700, TO_DATE('13-DEC-2022', 'DD-MON-YYYY'), TO_DATE('17-APR-2024', 'DD-MON-YYYY'));
INSERT INTO Facturi (cod_factura, cod_adresa, cod_client, total, data_eliberare, termen_plata)
VALUES (2, 2, 4, 620, TO_DATE('07-AUG-2024', 'DD-MON-YYYY'), TO_DATE('15-APR-2028', 'DD-MON-YYYY'));
INSERT INTO Facturi
VALUES (1, 1, 7, 200, TO_DATE('23-FEB-2023', 'DD-MON-YYYY'), TO_DATE('27-OCT-2027', 'DD-MON-YYYY'), 'Neplatit');
INSERT INTO Facturi
VALUES (2, 1, 7, 850, TO_DATE('20-AUG-2023', 'DD-MON-YYYY'), TO_DATE('27-APR-2025', 'DD-MON-YYYY'), 'In procesare');
INSERT INTO Facturi
VALUES (1, 2, 10, 340, TO_DATE('14-JAN-2023', 'DD-MON-YYYY'), TO_DATE('19-NOV-2023', 'DD-MON-YYYY'), 'Platit');
INSERT INTO Facturi (cod_factura, cod_adresa, cod_client, total, data_eliberare, termen_plata)
VALUES (2, 2, 10, 890, TO_DATE('27-DEC-2024', 'DD-MON-YYYY'), TO_DATE('08-APR-2027', 'DD-MON-YYYY'));
INSERT INTO Facturi (cod_factura, cod_adresa, cod_client, total, data_eliberare, termen_plata)
VALUES (3, 2, 10, 510, TO_DATE('03-APR-2023', 'DD-MON-YYYY'), TO_DATE('22-APR-2025', 'DD-MON-YYYY'));
INSERT INTO Facturi
VALUES (1, 1, 2, 280, TO_DATE('24-SEP-2020', 'DD-MON-YYYY'), TO_DATE('27-APR-2022', 'DD-MON-YYYY'), 'Neplatit');
INSERT INTO Facturi
VALUES (1, 1, 9, 320, TO_DATE('14-DEC-2022', 'DD-MON-YYYY'), TO_DATE('25-MAR-2025', 'DD-MON-YYYY'), 'Neplatit');
INSERT INTO Facturi
VALUES (1, 1, 11, 765, TO_DATE('05-FEB-2022', 'DD-MON-YYYY'), TO_DATE('02-MAY-2024', 'DD-MON-YYYY'), 'Neplatit');
INSERT INTO Facturi
VALUES (1, 1, 12, 180, TO_DATE('19-APR-2021', 'DD-MON-YYYY'), TO_DATE('02-APR-2022', 'DD-MON-YYYY'), 'Neplatit');

SELECT *
FROM Facturi;
```

Script Output x Query Result x

SQL | All Rows Fetched: 14 in 0.003 seconds

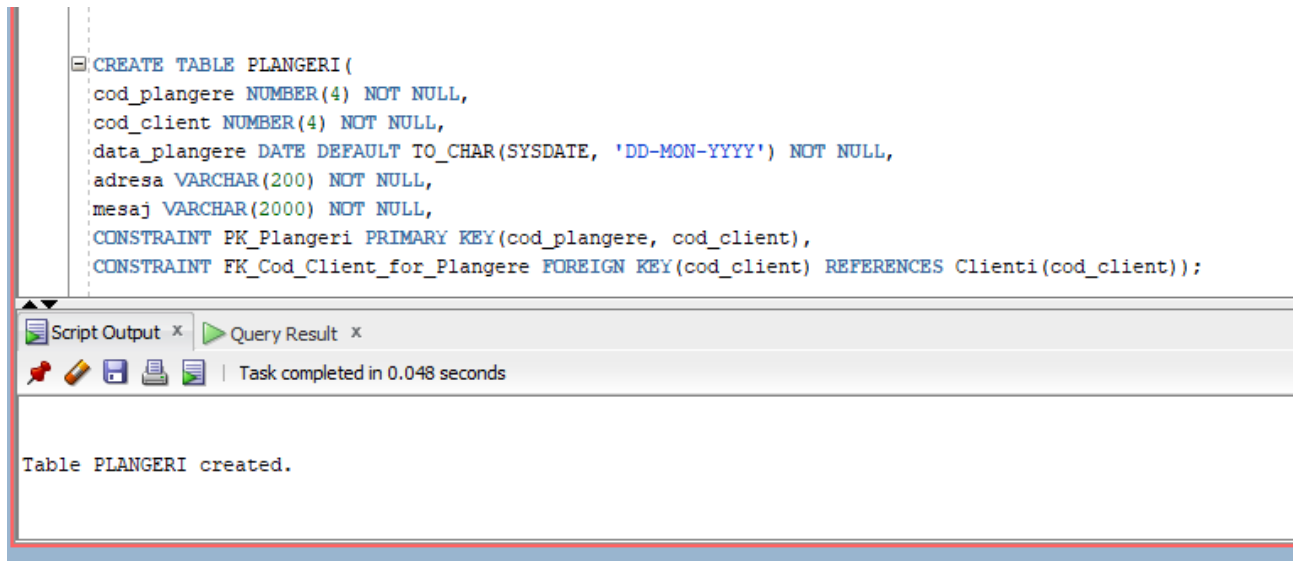
	COD_FACTURA	COD_ADRESA	COD_CLIENT	TOTAL	DATA ELIBERARE	TERMEN_PLATA	STATUS
4	1	2	4	700	13-DEC-22	17-APR-24	Neplatit
5	2	2	4	620	07-AUG-24	15-APR-28	Neplatit
6	1	1	7	200	23-FEB-23	27-OCT-27	Neplatit
7	2	1	7	850	20-AUG-23	27-APR-25	In procesare
8	1	2	10	340	14-JAN-23	19-NOV-23	Platit
9	2	2	10	890	27-DEC-24	08-APR-27	Neplatit
10	3	2	10	510	03-APR-23	22-APR-25	Neplatit
11	1	1	2	280	24-SEP-20	27-APR-22	Neplatit
12	1	1	9	320	14-DEC-22	25-MAR-25	Neplatit
13	1	1	11	765	05-FEB-22	02-MAY-24	Neplatit
14	1	1	12	180	19-APR-21	02-APR-22	Neplatit

- PLANGERI:

1) Creare:

```
CREATE TABLE PLANGERI(
cod_plangere NUMBER(4) NOT NULL,
cod_client NUMBER(4) NOT NULL,
data_plangere DATE DEFAULT TO_CHAR(SYSDATE, 'DD-MON-YYYY') NOT NULL,
adresa VARCHAR(200) NOT NULL,
mesaj VARCHAR(2000) NOT NULL,
```

```
CONSTRAINT PK_Plangeri PRIMARY KEY(cod_plangere, cod_client),  
CONSTRAINT FK_Cod_Client_for_Plangere FOREIGN KEY(cod_client) REFERENCES  
Clienti(cod_client));
```



2) Inserare:

```
INSERT INTO Plangeri  
VALUES (1, 1, TO_DATE('03-APR-2023', 'DD-MON-YYYY'), 'Romania, Bucuresti, Strada  
Giulesti, Nr 20', 'S-a spart o teava de gaz');  
INSERT INTO Plangeri  
VALUES (2, 1, TO_DATE('12-AUG-2023', 'DD-MON-YYYY'), 'Romania, Craiova, Strada  
Bretești, Nr 35', 'Curge apa din perete');  
INSERT INTO Plangeri  
VALUES (1, 3, TO_DATE('22-SEP-2020', 'DD-MON-YYYY'), 'Olanda, Amsterdam, Strada  
Lalelelor, Nr 44', 'S-a luat curentul');  
INSERT INTO Plangeri (cod_plangere, cod_client, adresa, mesaj)  
VALUES (2, 3, 'Romania, Cluj, Strada Florilor, Nr 120', 'A sarit o siguranta');  
INSERT INTO Plangeri  
VALUES (1, 4, TO_DATE('02-JUN-2020', 'DD-MON-YYYY'), 'Romania, Bucuresti, Strada Unirii,  
Nr 67', 'Nu este lumina pe strada');  
INSERT INTO Plangeri  
VALUES (1, 6, TO_DATE('14-FEB-2022', 'DD-MON-YYYY'), 'Romania, Timisoara, Strada  
Palatului, Nr 123', 'Nu functioneaza internetul');  
INSERT INTO Plangeri  
VALUES (2, 6, TO_DATE('17-DEC-2021', 'DD-MON-YYYY'), 'Romania, Oradea, Strada  
Moldovei, Nr 45', 'Nu avem apa calda');  
INSERT INTO Plangeri (cod_plangere, cod_client, adresa, mesaj)  
VALUES (1, 9, 'Romania, Craiova, Strada Craiova, Nr 56', 'S-a ars siguranta pe bloc');
```

```

INSERT INTO Plangeri (cod_plangere, cod_client, adresa, mesaj)
VALUES (2, 9, 'Romania, Oradea, Strada Raului, Nr 12', 'S-a luat apa de 3 zile');
INSERT INTO Plangeri
VALUES (1, 10, TO_DATE('23-NOV-2020', 'DD-MON-YYYY'), 'Romania, Oradea, Strada Marii,
Nr 67', 'Nu avem apa');

```

```

INSERT INTO Plangeri
VALUES (1, 1, TO_DATE('03-APR-2023', 'DD-MON-YYYY'), 'Romania, Bucuresti, Strada Giulesti, Nr 20', 'S-a spart o teava de gaz');
INSERT INTO Plangeri
VALUES (2, 1, TO_DATE('12-AUG-2023', 'DD-MON-YYYY'), 'Romania, Craiova, Strada Brestei, Nr 35', 'Curge apa din perete');
INSERT INTO Plangeri
VALUES (1, 3, TO_DATE('22-SEP-2020', 'DD-MON-YYYY'), 'Olanda, Amsterdam, Strada Lalelelor, Nr 44', 'S-a luat curentul');
INSERT INTO Plangeri (cod_plangere, cod_client, adresa, mesaj)
VALUES (2, 3, 'Romania, Cluj, Strada Florilor, Nr 120', 'A sarit o siguranta');
INSERT INTO Plangeri
VALUES (1, 4, TO_DATE('02-JUN-2020', 'DD-MON-YYYY'), 'Romania, Bucuresti, Strada Unirii, Nr 67', 'Nu este lumina pe strada');
INSERT INTO Plangeri
VALUES (1, 6, TO_DATE('14-FEB-2022', 'DD-MON-YYYY'), 'Romania, Timisoara, Strada Palatului, Nr 123', 'Nu functioneaza internetul');
INSERT INTO Plangeri
VALUES (2, 6, TO_DATE('17-DEC-2021', 'DD-MON-YYYY'), 'Romania, Oradea, Strada Moldovei, Nr 45', 'Nu avem apa calda');
INSERT INTO Plangeri (cod_plangere, cod_client, adresa, mesaj)
VALUES (1, 9, 'Romania, Craiova, Strada Craiova, Nr 56', 'S-a ars siguranta pe bloc');
INSERT INTO Plangeri (cod_plangere, cod_client, adresa, mesaj)
VALUES (2, 9, 'Romania, Oradea, Strada Raului, Nr 12', 'S-a luat apa de 3 zile');
INSERT INTO Plangeri
VALUES (1, 10, TO_DATE('23-NOV-2020', 'DD-MON-YYYY'), 'Romania, Oradea, Strada Marii, Nr 67', 'Nu avem apa');

SELECT *
FROM Plangeri;

```

	COD_PLANGERE	COD_CLIENT	DATA_PLANGERE	ADRESA	MESAJ
1	1	1	03-APR-23	Romania, Bucuresti, Strada Giulesti, Nr 20	S-a spart o teava de gaz
2	2	1	12-AUG-23	Romania, Craiova, Strada Brestei, Nr 35	Curge apa din perete
3	1	3	22-SEP-20	Olanda, Amsterdam, Strada Lalelelor, Nr 44	S-a luat curentul
4	2	3	15-MAY-22	Romania, Cluj, Strada Florilor, Nr 120	A sarit o siguranta
5	1	4	02-JUN-20	Romania, Bucuresti, Strada Unirii, Nr 67	Nu este lumina pe strada
6	1	6	14-FEB-22	Romania, Timisoara, Strada Palatului, Nr 123	Nu functioneaza internetul
7	2	6	17-DEC-21	Romania, Oradea, Strada Moldovei, Nr 45	Nu avem apa calda
8	1	9	15-MAY-22	Romania, Craiova, Strada Craiova, Nr 56	S-a ars siguranta pe bloc
9	2	9	15-MAY-22	Romania, Oradea, Strada Raului, Nr 12	S-a luat apa de 3 zile
10	1	10	23-NOV-20	Romania, Oradea, Strada Marii, Nr 67	Nu avem apa

- INFORMATII_MUNCITORI:

1) Creare:

```

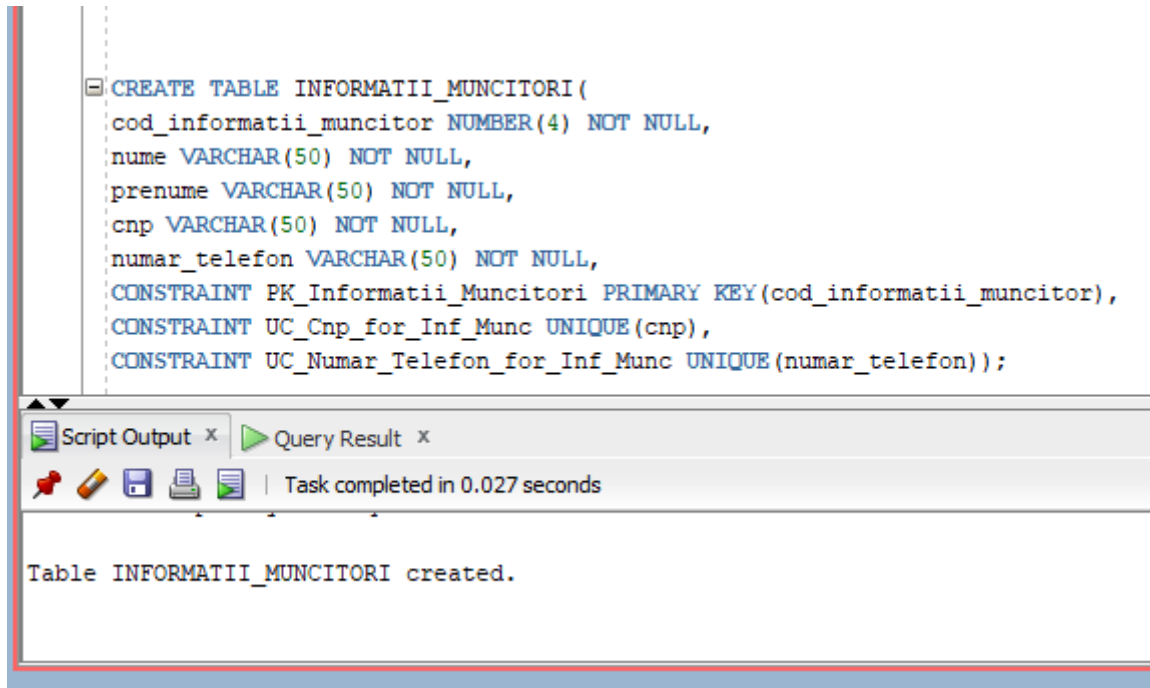
CREATE TABLE INFORMATII_MUNCITORI(
cod_informatii_muncitor NUMBER(4) NOT NULL,
nume VARCHAR(50) NOT NULL,
prenume VARCHAR(50) NOT NULL,
cnp VARCHAR(50) NOT NULL,

```

```

numar_telefon VARCHAR(50) NOT NULL,
CONSTRAINT PK_Informatii_Muncitori PRIMARY KEY(cod_informatii_muncitor),
CONSTRAINT UC_Cnp_for_Inf_Munc UNIQUE(cnp),
CONSTRAINT UC_Numar_Telefon_for_Inf_Munc UNIQUE(numar_telefon));

```



2) Inserare:

```

INSERT INTO Informatii_Muncitori
VALUES (SECVENTA.nextval, 'Ion', 'Andrei', '5021256073548', '0724679520');
INSERT INTO Informatii_Muncitori
VALUES (SECVENTA.nextval, 'Mincu', 'Ionut', '5023490567123', '0730941278');
INSERT INTO Informatii_Muncitori
VALUES (SECVENTA.nextval, 'Ioana', 'Maria', '5034091285674', '0710923856');
INSERT INTO Informatii_Muncitori
VALUES (SECVENTA.nextval, 'Cosmin', 'Marian', '5021092375834', '0730923857');
INSERT INTO Informatii_Muncitori
VALUES (SECVENTA.nextval, 'Loredana', 'Mariana', '5039054673120', '0745093285');
INSERT INTO Informatii_Muncitori
VALUES (SECVENTA.nextval, 'Sebastian', 'Marius', '5021093457835', '0749012356');
INSERT INTO Informatii_Muncitori
VALUES (SECVENTA.nextval, 'Oana', 'Floarea', '5038902135672', '0709431846');
INSERT INTO Informatii_Muncitori
VALUES (SECVENTA.nextval, 'Ion', 'Antonescu', '5024095689123', '0756908323');
INSERT INTO Informatii_Muncitori
VALUES (SECVENTA.nextval, 'Marius', 'Lica', '5021905623895', '0701938567');

```

```
INSERT INTO Informatii_Muncitori
VALUES (SECVENTA.nextval, 'Sorin', 'Mircea', '5023095671834', '0789042175');
INSERT INTO Informatii_Muncitori
VALUES (SECVENTA.nextval, 'Miruna', 'Mircea', '5023095671309', '0789042111');
INSERT INTO Informatii_Muncitori
VALUES (SECVENTA.nextval, 'Ionut', 'Ionel', '5010495671309', '0709342111');
INSERT INTO Informatii_Muncitori
VALUES (SECVENTA.nextval, 'Vasile', 'Vasilievici', '5023095673091', '0789043331');
```

```

VALUES (SECVENTA.nextval, 'Ion', 'Andrei', '5021256073548', '0724679520');
INSERT INTO Informatii_Muncitori
VALUES (SECVENTA.nextval, 'Mincu', 'Ionut', '5023490567123', '0730941278');
INSERT INTO Informatii_Muncitori
VALUES (SECVENTA.nextval, 'Ioana', 'Maria', '5034091285674', '0710923856');
INSERT INTO Informatii_Muncitori
VALUES (SECVENTA.nextval, 'Cosmin', 'Marian', '5021092375834', '0730923857');
INSERT INTO Informatii_Muncitori
VALUES (SECVENTA.nextval, 'Loredana', 'Mariana', '5039054673120', '0745093285');
INSERT INTO Informatii_Muncitori
VALUES (SECVENTA.nextval, 'Sebastian', 'Marius', '5021093457835', '0749012356');
INSERT INTO Informatii_Muncitori
VALUES (SECVENTA.nextval, 'Oana', 'Floarea', '5038902135672', '0709431846');
INSERT INTO Informatii_Muncitori
VALUES (SECVENTA.nextval, 'Ion', 'Antonescu', '5024095689123', '0756908323');
INSERT INTO Informatii_Muncitori
VALUES (SECVENTA.nextval, 'Marius', 'Lica', '5021905623895', '0701938567');
INSERT INTO Informatii_Muncitori
VALUES (SECVENTA.nextval, 'Sorin', 'Mircea', '5023095671834', '0789042175');
INSERT INTO Informatii_Muncitori
VALUES (SECVENTA.nextval, 'Miruna', 'Mircea', '5023095671309', '0789042111');
INSERT INTO Informatii_Muncitori
VALUES (SECVENTA.nextval, 'Ionut', 'Ionel', '5010495671309', '0709342111');
INSERT INTO Informatii_Muncitori
VALUES (SECVENTA.nextval, 'Vasile', 'Vasilievici', '5023095673091', '0789043331');

SELECT *
FROM Informatii_Muncitori;

```

Script Output x Query Result x

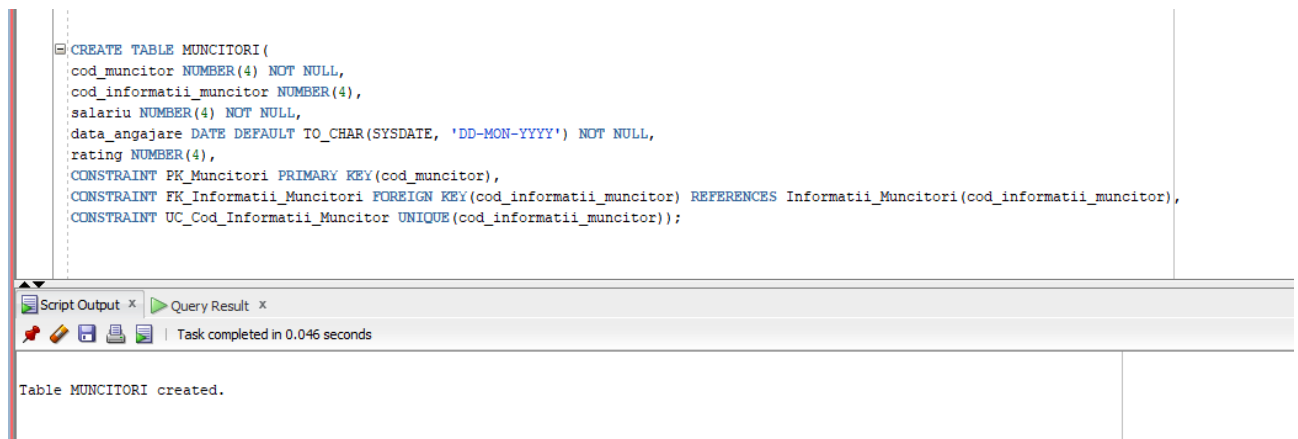
SQL | All Rows Fetched: 13 in 0.002 seconds

	COD_INFORMATII_MUNCITOR	NUME	PRENUME	CNP	NUMAR_TELEFON
2	2	Mincu	Ionut	5023490567123	0730941278
3	3	Ioana	Maria	5034091285674	0710923856
4	4	Cosmin	Marian	5021092375834	0730923857
5	5	Loredana	Mariana	5039054673120	0745093285
6	6	Sebastian	Marius	5021093457835	0749012356
7	7	Oana	Floarea	5038902135672	0709431846
8	8	Ion	Antonescu	5024095689123	0756908323
9	9	Marius	Lica	5021905623895	0701938567
10	10	Sorin	Mircea	5023095671834	0789042175
11	11	Miruna	Mircea	5023095671309	0789042111
12	12	Ionut	Ionel	5010495671309	0709342111
13	13	Vasile	Vasilievici	5023095673091	0789043331

- MUNCITORI:

1) Creare:

```
CREATE TABLE MUNCITORI(  
cod_muncitor NUMBER(4) NOT NULL,  
cod_informatii_muncitor NUMBER(4),  
salariu NUMBER(4) NOT NULL,  
data_angajare DATE DEFAULT TO_CHAR(SYSDATE, 'DD-MON-YYYY') NOT NULL,  
rating NUMBER(4),  
CONSTRAINT PK_Muncitori PRIMARY KEY(cod_muncitor),  
CONSTRAINT FK_Informatii_Muncitori FOREIGN KEY(cod_informatii_muncitor)  
REFERENCES Informatii_Muncitori(cod_informatii_muncitor),  
CONSTRAINT UC_Cod_Informatii_Muncitor UNIQUE(cod_informatii_muncitor));
```



2) Inserare:

```
INSERT INTO Muncitori  
VALUES (1, 1, 2300, TO_DATE('23-NOV-2019', 'DD-MON-YYYY'), 20);  
INSERT INTO Muncitori  
VALUES (2, 2, 1800, TO_DATE('10-SEP-2020', 'DD-MON-YYYY'), 50);  
INSERT INTO Muncitori  
VALUES (3, NULL, 1000, TO_DATE('27-DEC-2021', 'DD-MON-YYYY'), 10);  
INSERT INTO Muncitori  
VALUES (4, 6, 3000, TO_DATE('15-JAN-2018', 'DD-MON-YYYY'), 200);  
INSERT INTO Muncitori  
VALUES (5, 8, 8000, TO_DATE('01-OCT-2012', 'DD-MON-YYYY'), 800);  
INSERT INTO Muncitori  
VALUES (6, 5, 4000, TO_DATE('14-AUG-2017', 'DD-MON-YYYY'), 400);  
INSERT INTO Muncitori  
VALUES (7, NULL, 2300, TO_DATE('20-FEB-2020', 'DD-MON-YYYY'), NULL);  
INSERT INTO Muncitori
```

```
VALUES (8, 7, 3300, TO_DATE('19-JUL-2020', 'DD-MON-YYYY'), 70);
INSERT INTO Muncitori
VALUES (9, 9, 1700, TO_DATE('08-MAR-2021', 'DD-MON-YYYY'), 30);
INSERT INTO Muncitori
VALUES (10, 3, 1600, TO_DATE('25-MAY-2017', 'DD-MON-YYYY'), NULL);
INSERT INTO Muncitori
VALUES (11, 11, 1200, TO_DATE('13-DEC-2020', 'DD-MON-YYYY'), 456);
INSERT INTO Muncitori
VALUES (12, 12, 2000, TO_DATE('24-SEP-2020', 'DD-MON-YYYY'), 555);
INSERT INTO Muncitori
VALUES (13, 13, 1300, TO_DATE('13-DEC-2017', 'DD-MON-YYYY'), 456);
```

```

INSERT INTO Muncitori
VALUES (1, 1, 2300, TO_DATE('23-NOV-2019', 'DD-MON-YYYY'), 20);
INSERT INTO Muncitori
VALUES (2, 2, 1800, TO_DATE('10-SEP-2020', 'DD-MON-YYYY'), 50);
INSERT INTO Muncitori
VALUES (3, NULL, 1000, TO_DATE('27-DEC-2021', 'DD-MON-YYYY'), 10);
INSERT INTO Muncitori
VALUES (4, 6, 3000, TO_DATE('15-JAN-2018', 'DD-MON-YYYY'), 200);
INSERT INTO Muncitori
VALUES (5, 8, 8000, TO_DATE('01-OCT-2012', 'DD-MON-YYYY'), 800);
INSERT INTO Muncitori
VALUES (6, 5, 4000, TO_DATE('14-AUG-2017', 'DD-MON-YYYY'), 400);
INSERT INTO Muncitori
VALUES (7, NULL, 2300, TO_DATE('20-FEB-2020', 'DD-MON-YYYY'), NULL);
INSERT INTO Muncitori
VALUES (8, 7, 3300, TO_DATE('19-JUL-2020', 'DD-MON-YYYY'), 70);
INSERT INTO Muncitori
VALUES (9, 9, 1700, TO_DATE('08-MAR-2021', 'DD-MON-YYYY'), 30);
INSERT INTO Muncitori
VALUES (10, 3, 1600, TO_DATE('25-MAY-2017', 'DD-MON-YYYY'), NULL);
INSERT INTO Muncitori
VALUES (11, 11, 1200, TO_DATE('13-DEC-2020', 'DD-MON-YYYY'), 456);
INSERT INTO Muncitori
VALUES (12, 12, 2000, TO_DATE('24-SEP-2020', 'DD-MON-YYYY'), 555);
INSERT INTO Muncitori
VALUES (13, 13, 1300, TO_DATE('13-DEC-2017', 'DD-MON-YYYY'), 456);

SELECT *
FROM Muncitori;

```

Script Output x Query Result x

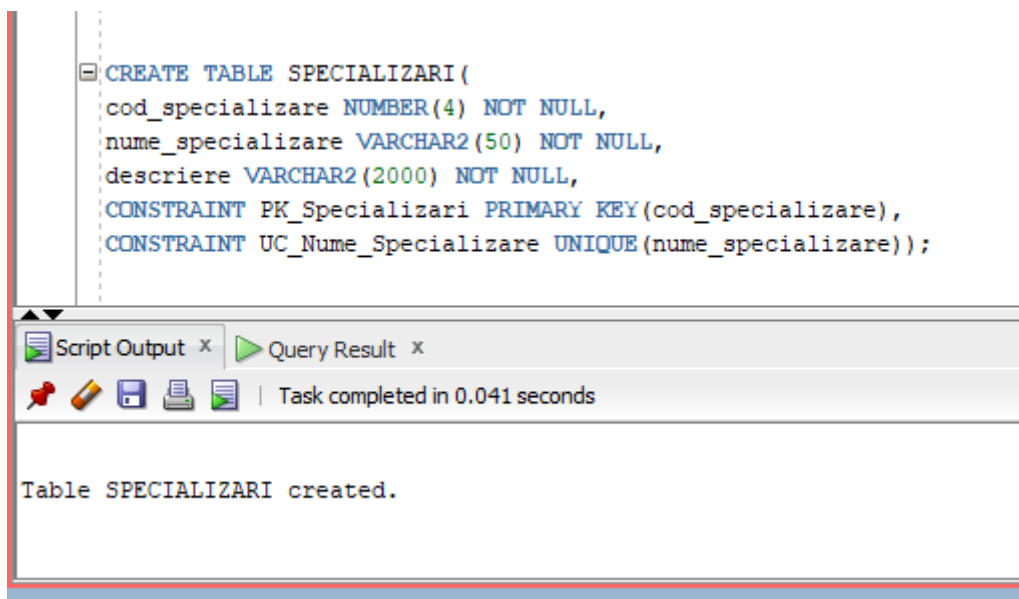
SQL | All Rows Fetched: 13 in 0.002 seconds

	COD_MUNCITOR	COD_INFORMATII_MUNCITOR	SALARIU	DATA_ANGAJARE	RATING
1	1		2300	23-NOV-19	20
2	2		1800	10-SEP-20	50
3	3	(null)	1000	27-DEC-21	10
4	4	6	3000	15-JAN-18	200
5	5	8	8000	01-OCT-12	800
6	6	5	4000	14-AUG-17	400
7	7	(null)	2300	20-FEB-20	(null)
8	8	7	3300	19-JUL-20	70
9	9	9	1700	08-MAR-21	30
10	10	3	1600	25-MAY-17	(null)
11	11	11	1200	13-DEC-20	456
12	12	12	2000	24-SEP-20	555
13	13	13	1300	13-DEC-17	456

- SPECIALIZARI:

1) Creare:

```
CREATE TABLE SPECIALIZARI(  
cod_specializare NUMBER(4) NOT NULL,  
nume_specializare VARCHAR2(50) NOT NULL,  
descriere VARCHAR2(2000) NOT NULL,  
CONSTRAINT PK_Specializari PRIMARY KEY(cod_specializare),  
CONSTRAINT UC_Nume_Specializare UNIQUE(nume_specializare));
```



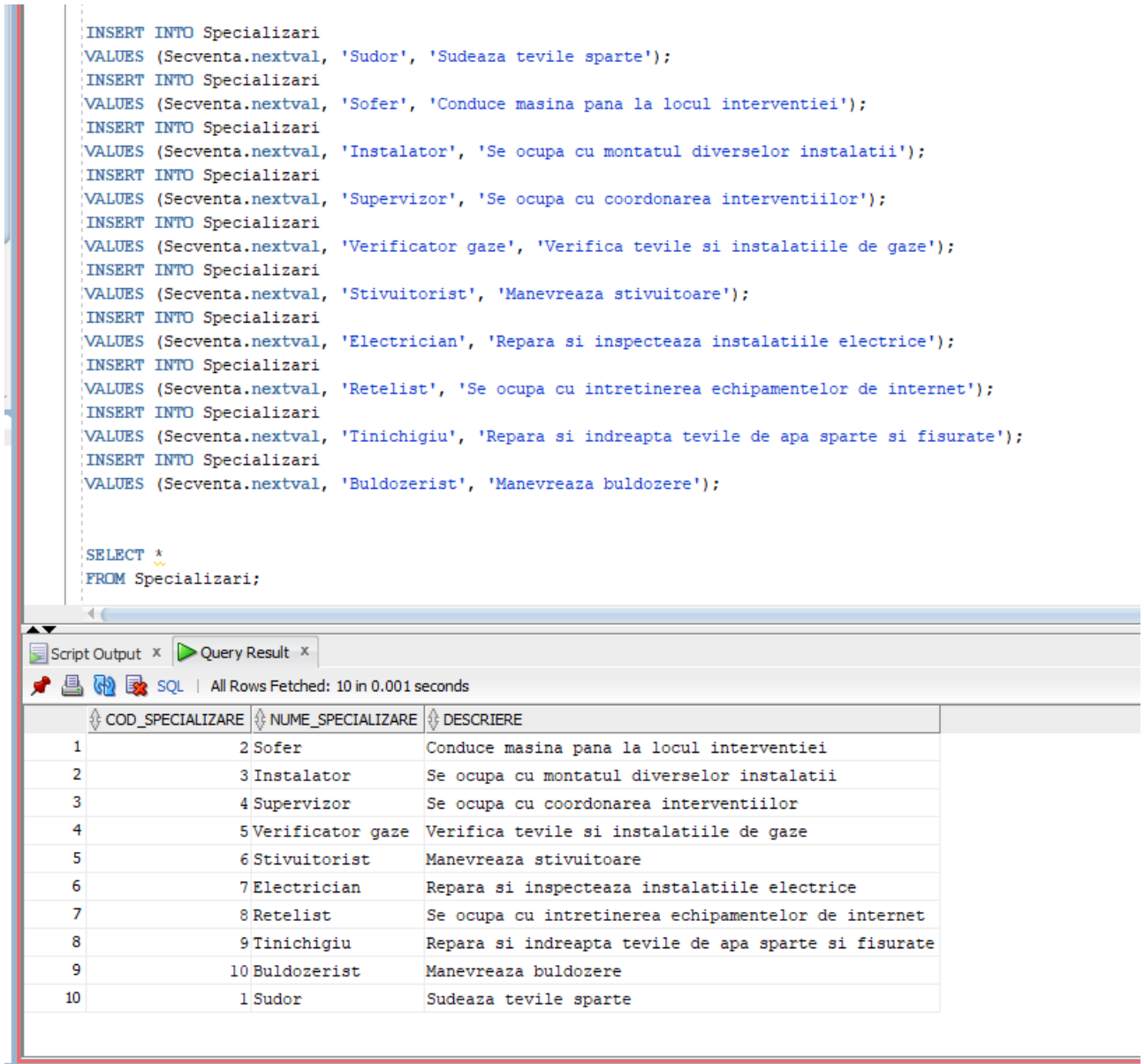
2) Inserare:

```
INSERT INTO Specializari  
VALUES (Secventa.nextval, 'Sudor', 'Sudeaza tevile sparte');  
INSERT INTO Specializari  
VALUES (Secventa.nextval, 'Sofer', 'Conduce masina pana la locul interventiei');  
INSERT INTO Specializari  
VALUES (Secventa.nextval, 'Instalator', 'Se ocupa cu montatul diverselor instalatii');  
INSERT INTO Specializari  
VALUES (Secventa.nextval, 'Supervizor', 'Se ocupa cu coordonarea interventiilor');  
INSERT INTO Specializari  
VALUES (Secventa.nextval, 'Verificator gaze', 'Verifica tevile si instalatiile de gaze');  
INSERT INTO Specializari  
VALUES (Secventa.nextval, 'Stivuatorist', 'Manevreaza stivuitoare');  
INSERT INTO Specializari  
VALUES (Secventa.nextval, 'Electrician', 'Repara si inspecteaza instalatiile electrice');
```

```

INSERT INTO Specializari
VALUES (Secventa.nextval, 'Retelist', 'Se ocupa cu intretinerea echipamentelor de internet');
INSERT INTO Specializari
VALUES (Secventa.nextval, 'Tinichigiu', 'Repara si indreapta tevile de apa sparte si fisurate');
INSERT INTO Specializari
VALUES (Secventa.nextval, 'Buldozerist', 'Manevreaza buldozere');

```



```

INSERT INTO Specializari
VALUES (Secventa.nextval, 'Sudor', 'Sudeaza tevile sparte');
INSERT INTO Specializari
VALUES (Secventa.nextval, 'Sofer', 'Conduce masina pana la locul interventiei');
INSERT INTO Specializari
VALUES (Secventa.nextval, 'Instalator', 'Se ocupa cu montatul diverselor instalatii');
INSERT INTO Specializari
VALUES (Secventa.nextval, 'Supervizor', 'Se ocupa cu coordonarea interventiilor');
INSERT INTO Specializari
VALUES (Secventa.nextval, 'Verificator gaze', 'Verifica tevile si instalatiile de gaze');
INSERT INTO Specializari
VALUES (Secventa.nextval, 'Stivuitorist', 'Manevreaza stivuitoare');
INSERT INTO Specializari
VALUES (Secventa.nextval, 'Electrician', 'Repara si inspecteaza instalatiile electrice');
INSERT INTO Specializari
VALUES (Secventa.nextval, 'Retelist', 'Se ocupa cu intretinerea echipamentelor de internet');
INSERT INTO Specializari
VALUES (Secventa.nextval, 'Tinichigiu', 'Repara si indreapta tevile de apa sparte si fisurate');
INSERT INTO Specializari
VALUES (Secventa.nextval, 'Buldozerist', 'Manevreaza buldozere');

SELECT *
FROM Specializari;

```

Script Output x Query Result x

SQL | All Rows Fetched: 10 in 0.001 seconds

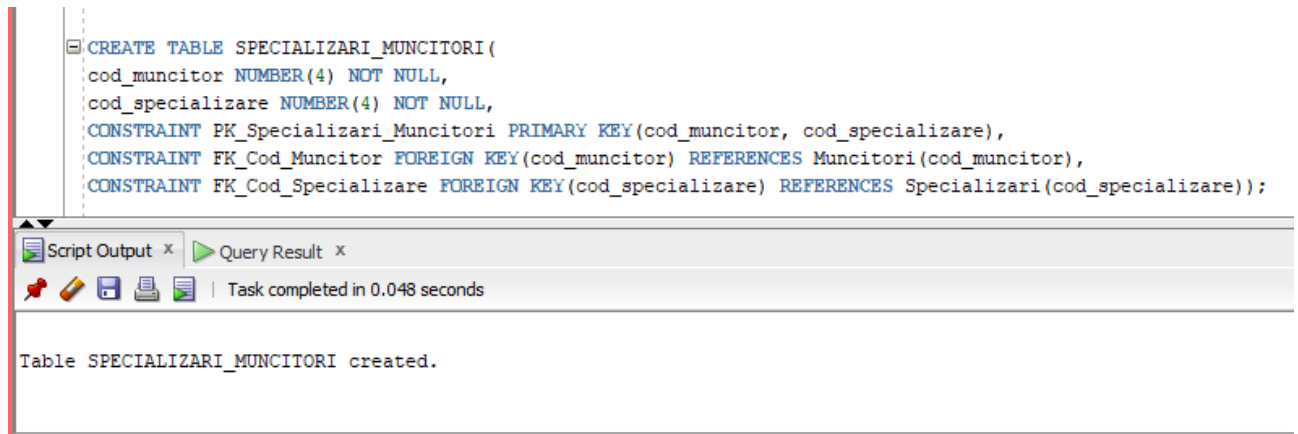
	COD_SPECIALIZARE	NUME_SPECIALIZARE	DESCRIERE
1	2	Sofer	Conduce masina pana la locul interventiei
2	3	Instalator	Se ocupa cu montatul diverselor instalatii
3	4	Supervizor	Se ocupa cu coordonarea interventiilor
4	5	Verificator gaze	Verifica tevile si instalatiile de gaze
5	6	Stivuitorist	Manevreaza stivuitoare
6	7	Electrician	Repara si inspecteaza instalatiile electrice
7	8	Retelist	Se ocupa cu intretinerea echipamentelor de internet
8	9	Tinichigiu	Repara si indreapta tevile de apa sparte si fisurate
9	10	Buldozerist	Manevreaza buldozere
10	1	Sudor	Sudeaza tevile sparte

- SPECIALIZARI_MUNCITORI:

1) Creare:

CREATE TABLE SPECIALIZARI_MUNCITORI(

```
cod_muncitor NUMBER(4) NOT NULL,  
cod_specializare NUMBER(4) NOT NULL,  
CONSTRAINT PK_Specializari_Muncitori PRIMARY KEY(cod_muncitor, cod_specializare),  
CONSTRAINT FK_Cod_Muncitor FOREIGN KEY(cod_muncitor) REFERENCES  
Muncitori(cod_muncitor),  
CONSTRAINT FK_Cod_Specializare FOREIGN KEY(cod_specializare) REFERENCES  
Specializari(cod_specializare));
```



2) Inserare:

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

```
INSERT INTO Specializari_Muncitori
VALUES (8, 2);
INSERT INTO Specializari_Muncitori
VALUES (9, 5);
INSERT INTO Specializari_Muncitori
VALUES (10, 2);
INSERT INTO Specializari_Muncitori
VALUES (10, 8);
INSERT INTO Specializari_Muncitori
VALUES (1, 2);
INSERT INTO Specializari_Muncitori
VALUES (11, 2);
INSERT INTO Specializari_Muncitori
VALUES (12, 2);
INSERT INTO Specializari_Muncitori
VALUES (13, 2);
INSERT INTO Specializari_Muncitori
VALUES (1, 7);
INSERT INTO Specializari_Muncitori
VALUES (1, 8);
INSERT INTO Specializari_Muncitori
VALUES (10, 1);
```

```
VALUES (6, 5);
INSERT INTO Specializari_Muncitori
VALUES (6, 7);
INSERT INTO Specializari_Muncitori
VALUES (8, 2);
INSERT INTO Specializari_Muncitori
VALUES (9, 5);
INSERT INTO Specializari_Muncitori
VALUES (10, 2);
INSERT INTO Specializari_Muncitori
VALUES (10, 8);
INSERT INTO Specializari_Muncitori
VALUES (1, 2);
INSERT INTO Specializari_Muncitori
VALUES (11, 2);
INSERT INTO Specializari_Muncitori
VALUES (12, 2);
INSERT INTO Specializari_Muncitori
VALUES (13, 2);
INSERT INTO Specializari_Muncitori
VALUES (1, 7);
INSERT INTO Specializari_Muncitori
VALUES (1, 8);
INSERT INTO Specializari_Muncitori
VALUES (10, 1);
```

```
SELECT *
FROM Specializari_Muncitori;
```

Script Output x Query Result x

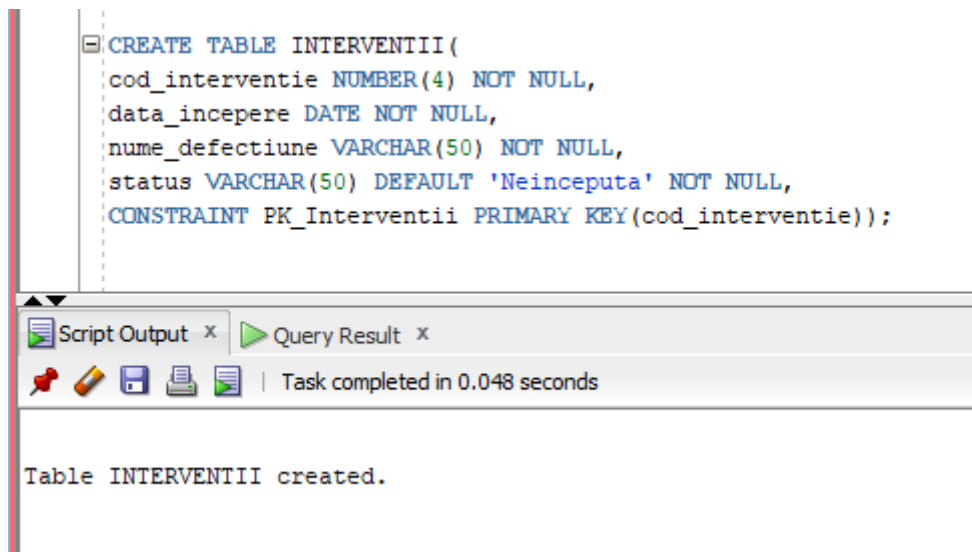
SQL | All Rows Fetched: 21 in 0.001 seconds

	COD_MUNCITOR	COD_SPECIALIZARE
10	5	4
11	6	3
12	6	5
13	6	7
14	8	2
15	9	5
16	10	1
17	10	2
18	10	8
19	11	2
20	12	2
21	13	2

- INTERVENTII:

1) Creare:

```
CREATE TABLE INTERVENTII(  
cod_interventie NUMBER(4) NOT NULL,  
data_incepere DATE NOT NULL,  
nume_defectiune VARCHAR(50) NOT NULL,  
status VARCHAR(50) DEFAULT 'Neinceputa' NOT NULL,  
CONSTRAINT PK_Interventii PRIMARY KEY(cod_interventie));
```



2) Inserare:

```
INSERT INTO Interventii  
VALUES (Secventa.nextval, TO_DATE('01-OCT-2021', 'DD-MON-YYYY'), 'Teava de apa sparta',  
'In lucru');  
INSERT INTO Interventii  
VALUES (Secventa.nextval, TO_DATE('12-DEC-2022', 'DD-MON-YYYY'), 'Scurgere de gaze',  
'Neinceputa');  
INSERT INTO Interventii  
VALUES (Secventa.nextval, TO_DATE('25-NOV-2025', 'DD-MON-YYYY'), 'Teava de gaze  
sparte', 'Finalizata');  
INSERT INTO Interventii (cod_interventie, data_incepere, nume_defectiune)  
VALUES (Secventa.nextval, TO_DATE('15-MAY-2022', 'DD-MON-YYYY'), 'Cablu de  
electricitate intrerupt');  
INSERT INTO Interventii  
VALUES (Secventa.nextval, TO_DATE('20-SEP-2021', 'DD-MON-YYYY'), 'Internet picat',  
'Finalizata');
```

```
INSERT INTO Interventii
VALUES (Secventa.nextval, TO_DATE('05-MAY-2022', 'DD-MON-YYYY'), 'Apa nefunctionala',
'In lucru');
INSERT INTO Interventii
VALUES (Secventa.nextval, TO_DATE('19-APR-2021', 'DD-MON-YYYY'), 'Siguranta sarita', 'In
lucru');
INSERT INTO Interventii (cod_interventie, data_incepere, nume_defectiune)
VALUES (Secventa.nextval, TO_DATE('20-JUN-2025', 'DD-MON-YYYY'), 'Teava de apa
fisurata');
INSERT INTO Interventii
VALUES (Secventa.nextval, TO_DATE('23-APR-2023', 'DD-MON-YYYY'), 'Siguranta sarita',
'Neinceputa');
INSERT INTO Interventii
VALUES (Secventa.nextval, TO_DATE('25-APR-2020', 'DD-MON-YYYY'), 'Scurgere de gaze',
'Finalizata');
```

```

INSERT INTO Interventii
VALUES (Secventa.nextval, TO_DATE('01-OCT-2021', 'DD-MON-YYYY'), 'Teava de apa sparta', 'In lucru');
INSERT INTO Interventii
VALUES (Secventa.nextval, TO_DATE('12-DEC-2022', 'DD-MON-YYYY'), 'Scurgere de gaze', 'Neinceputa');
INSERT INTO Interventii
VALUES (Secventa.nextval, TO_DATE('25-NOV-2025', 'DD-MON-YYYY'), 'Teava de gaze sparte', 'Finalizata');
INSERT INTO Interventii (cod_interventie, data_incepere, nume_defectiune)
VALUES (Secventa.nextval, TO_DATE('15-MAY-2022', 'DD-MON-YYYY'), 'Cablu de electricitate intrerupt');
INSERT INTO Interventii
VALUES (Secventa.nextval, TO_DATE('20-SEP-2021', 'DD-MON-YYYY'), 'Internet picat', 'Finalizata');
INSERT INTO Interventii
VALUES (Secventa.nextval, TO_DATE('05-MAY-2022', 'DD-MON-YYYY'), 'Apa nefunctionala', 'In lucru');
INSERT INTO Interventii
VALUES (Secventa.nextval, TO_DATE('19-APR-2021', 'DD-MON-YYYY'), 'Siguranta sarita', 'In lucru');
INSERT INTO Interventii (cod_interventie, data_incepere, nume_defectiune)
VALUES (Secventa.nextval, TO_DATE('20-JUN-2025', 'DD-MON-YYYY'), 'Teava de apa fisurata');
INSERT INTO Interventii
VALUES (Secventa.nextval, TO_DATE('23-APR-2023', 'DD-MON-YYYY'), 'Siguranta sarita', 'Neinceputa');
INSERT INTO Interventii
VALUES (Secventa.nextval, TO_DATE('25-APR-2020', 'DD-MON-YYYY'), 'Scurgere de gaze', 'Finalizata');

SELECT *
FROM Interventii;

```

Script Output x

Query Result x

SQL

All Rows Fetched: 10 in 0.001 seconds

	COD_INTERVENTIE	DATA_INCEPERE	NUME_DEFECTIUNE	STATUS
1	1	01-OCT-21	Teava de apa sparta	In lucru
2	2	12-DEC-22	Scurgere de gaze	Neinceputa
3	3	25-NOV-25	Teava de gaze sparte	Finalizata
4	4	15-MAY-22	Cablu de electricitate intrerupt	Neinceputa
5	5	20-SEP-21	Internet picat	Finalizata
6	6	05-MAY-22	Apa nefunctionala	In lucru
7	7	19-APR-21	Siguranta sarita	In lucru
8	8	20-JUN-25	Teava de apa fisurata	Neinceputa
9	9	23-APR-23	Siguranta sarita	Neinceputa
10	10	25-APR-20	Scurgere de gaze	Finalizata

- DOCUMENTE:

1) Creare:

```

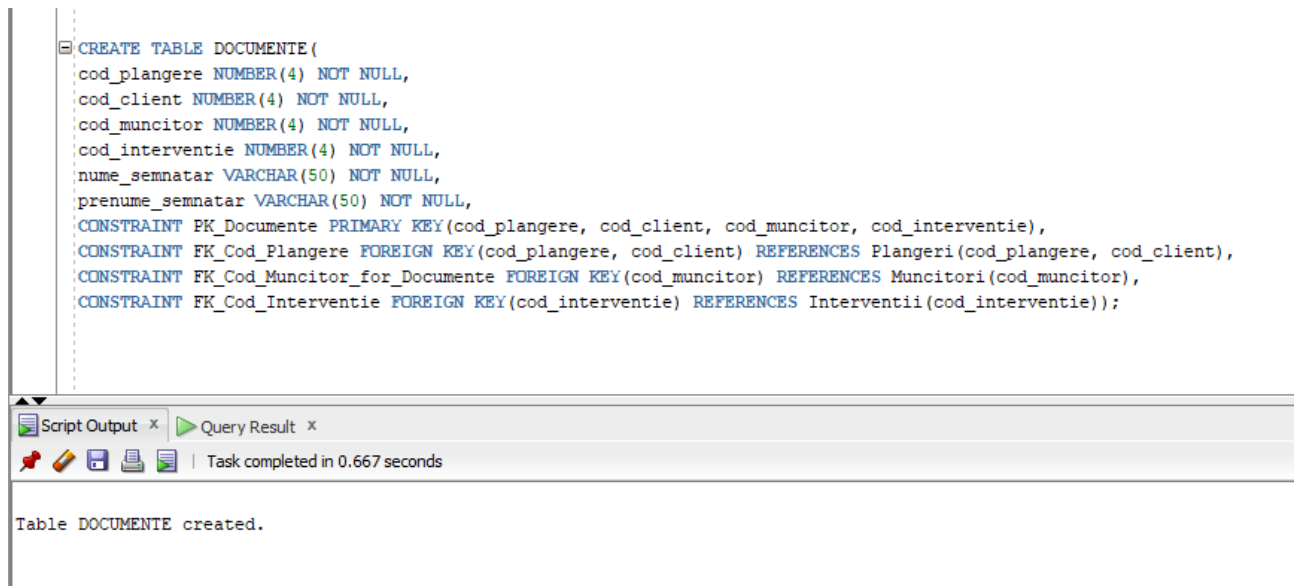
CREATE TABLE DOCUMENTE(
cod_plangere NUMBER(4) NOT NULL,
cod_client NUMBER(4) NOT NULL,
cod_muncitor NUMBER(4) NOT NULL,
cod_interventie NUMBER(4) NOT NULL,
nume_semnatar VARCHAR(50) NOT NULL,
prenume_semnatar VARCHAR(50) NOT NULL,

```

```

CONSTRAINT PK_Documente PRIMARY KEY(cod_plangere, cod_client, cod_muncitor,
cod_interventie),
CONSTRAINT FK_Cod_Plangere FOREIGN KEY(cod_plangere, cod_client) REFERENCES
Plangeri(cod_plangere, cod_client),
CONSTRAINT FK_Cod_Muncitor_for_Documente FOREIGN KEY(cod_muncitor)
REFERENCES Muncitori(cod_muncitor),
CONSTRAINT FK_Cod_Interventie FOREIGN KEY(cod_interventie) REFERENCES
Interventii(cod_interventie));

```



The screenshot shows a SQL IDE window with a script editor and a results pane. The script editor contains the following SQL code:

```

CREATE TABLE DOCUMENTE (
    cod_plangere NUMBER(4) NOT NULL,
    cod_client NUMBER(4) NOT NULL,
    cod_muncitor NUMBER(4) NOT NULL,
    cod_interventie NUMBER(4) NOT NULL,
    nume_semnatar VARCHAR(50) NOT NULL,
    prenume_semnatar VARCHAR(50) NOT NULL,
    CONSTRAINT PK_Documente PRIMARY KEY(cod_plangere, cod_client, cod_muncitor, cod_interventie),
    CONSTRAINT FK_Cod_Plangere FOREIGN KEY(cod_plangere, cod_client) REFERENCES Plangeri(cod_plangere, cod_client),
    CONSTRAINT FK_Cod_Muncitor_for_Documente FOREIGN KEY(cod_muncitor) REFERENCES Muncitori(cod_muncitor),
    CONSTRAINT FK_Cod_Interventie FOREIGN KEY(cod_interventie) REFERENCES Interventii(cod_interventie));

```

The results pane shows the message: "Table DOCUMENTE created." and a status bar indicates "Task completed in 0.667 seconds".

2) Inserare:

```

INSERT INTO Documente
VALUES (1, 3, 6, 4, 'Dumitrescu', 'Marian');
INSERT INTO Documente
VALUES (1, 3, 2, 4, 'Dumitrescu', 'Marian');
INSERT INTO Documente
VALUES (1, 3, 5, 4, 'Dumitrescu', 'Marian');
INSERT INTO Documente
VALUES (1, 9, 6, 9, 'Andrei', 'Constantin');
INSERT INTO Documente
VALUES (1, 10, 1, 1, 'Andrei', 'Constantin');
INSERT INTO Documente
VALUES (1, 10, 6, 6, 'Dumitrescu', 'Marian');
INSERT INTO Documente
VALUES (1, 10, 3, 6, 'Dumitrescu', 'Marian');
INSERT INTO Documente
VALUES (1, 10, 10, 6, 'Dumitrescu', 'Marian');

```

```
INSERT INTO Documente
VALUES (1, 1, 9, 3, 'Andrei', 'Constantin');
INSERT INTO Documente
VALUES (2, 9, 1, 8, 'Andrei', 'Constantin');
INSERT INTO Documente
VALUES (2, 9, 6, 8, 'Andrei', 'Constantin');
INSERT INTO Documente
VALUES (2, 9, 3, 8, 'Andrei', 'Constantin');
INSERT INTO Documente
VALUES (2, 6, 1, 8, 'Andrei', 'Constantin');
INSERT INTO Documente
VALUES (2, 6, 6, 8, 'Andrei', 'Constantin');
INSERT INTO Documente
VALUES (2, 6, 3, 8, 'Andrei', 'Constantin');
```

Worksheet

Query Builder

```

VALUES (1, 3, 2, 4, 'Dumitrescu', 'Marian'),
INSERT INTO Documente
VALUES (1, 3, 5, 4, 'Dumitrescu', 'Marian');
INSERT INTO Documente
VALUES (1, 9, 6, 9, 'Andrei', 'Constantin');
INSERT INTO Documente
VALUES (1, 10, 1, 1, 'Andrei', 'Constantin');
INSERT INTO Documente
VALUES (1, 10, 6, 6, 'Dumitrescu', 'Marian');
INSERT INTO Documente
VALUES (1, 10, 3, 6, 'Dumitrescu', 'Marian');
INSERT INTO Documente
VALUES (1, 10, 10, 6, 'Dumitrescu', 'Marian');
INSERT INTO Documente
VALUES (1, 1, 9, 3, 'Andrei', 'Constantin');
INSERT INTO Documente
VALUES (2, 9, 1, 8, 'Andrei', 'Constantin');
INSERT INTO Documente
VALUES (2, 9, 6, 8, 'Andrei', 'Constantin');
INSERT INTO Documente
VALUES (2, 9, 3, 8, 'Andrei', 'Constantin');
INSERT INTO Documente
VALUES (2, 6, 1, 8, 'Andrei', 'Constantin');
INSERT INTO Documente
VALUES (2, 6, 6, 8, 'Andrei', 'Constantin');
INSERT INTO Documente
VALUES (2, 6, 3, 8, 'Andrei', 'Constantin');

SELECT *
FROM Documente;

```

Script Output x

Query Result x

SQL | All Rows Fetched: 15 in 0.003 seconds

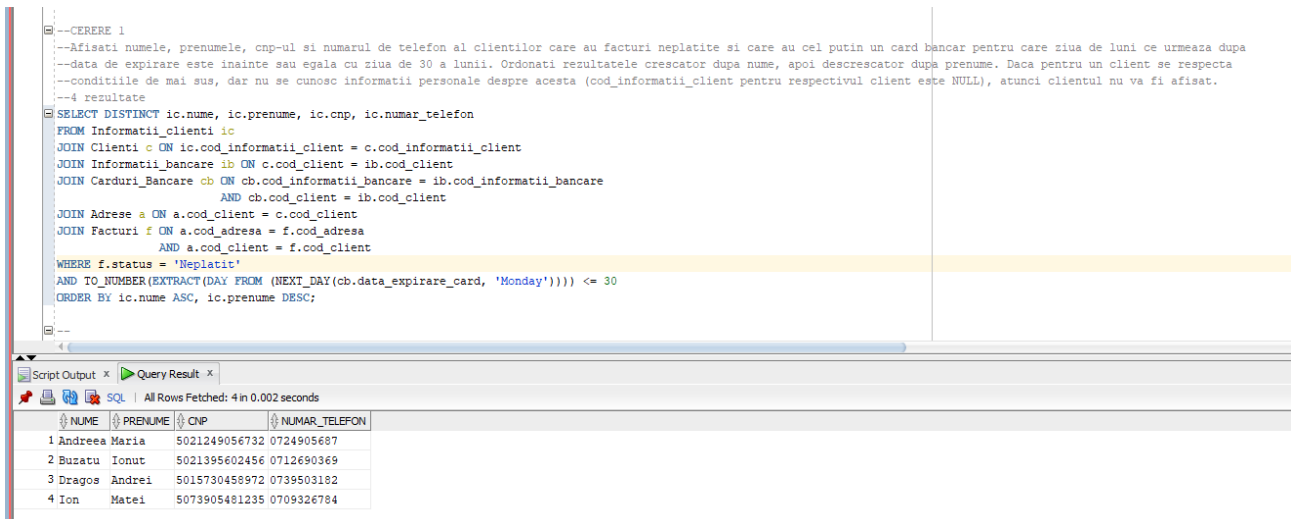
	COD_PLANGERE	COD_CLIENT	COD_MUNCITOR	COD_INTERVENTIE	NUME_SEMNATAR	PRENUME_SEMNATAR
1	1	9	6	9	Andrei	Constantin
5	1	10	1	1	Andrei	Constantin
6	1	10	6	6	Dumitrescu	Marian
7	1	10	3	6	Dumitrescu	Marian
8	1	10	10	6	Dumitrescu	Marian
9	1	1	9	3	Andrei	Constantin
10	2	9	1	8	Andrei	Constantin
11	2	9	6	8	Andrei	Constantin
12	2	9	3	8	Andrei	Constantin
13	2	6	1	8	Andrei	Constantin
14	2	6	6	8	Andrei	Constantin
15	2	6	3	8	Andrei	Constantin

12. Cereri complexe SQL:

1) Cerere 1:

Enunt: Afisati numele, prenumele, cnp-ul si numarul de telefon al clientilor care au facturi neplatite si care au cel putin un card bancar pentru care ziua de luni ce urmeaza dupa data de expirare este inainte sau egala cu ziua de 30 a lunii. Ordonati rezultatele crescator dupa nume, apoi descrescator dupa prenume. Daca pentru un client se respecta conditiile de mai sus, dar nu se cunosc informatii personale despre acesta (cod_informatii_client pentru respectivul client este NULL), atunci clientul nu va fi afisat.

```
SELECT DISTINCT ic.numa, ic.prenume, ic.cnp, ic.numar_telefon
FROM Informatii_clienti ic
JOIN Clienti c ON ic.cod_informatii_client = c.cod_informatii_client
JOIN Informatii_bancare ib ON c.cod_client = ib.cod_client
JOIN Carduri_Bancare cb ON cb.cod_informatii_bancare = ib.cod_informatii_bancare
AND cb.cod_client = ib.cod_client
JOIN Adrese a ON a.cod_client = c.cod_client
JOIN Facturi f ON a.cod_adresa = f.cod_adresa
AND a.cod_client = f.cod_client
WHERE f.status = 'Neplatit'
AND TO_NUMBER(EXTRACT(DAY FROM (NEXT_DAY(cb.data_expirare_card, 'Monday'))))
<= 30
ORDER BY ic.numa ASC, ic.prenume DESC;
```



```
--CERERE 1
--Afisati numele, prenumele, cnp-ul si numarul de telefon al clientilor care au facturi neplatite si care au cel putin un card bancar pentru care ziua de luni ce urmeaza dupa
--data de expirare este inainte sau egala cu ziua de 30 a lunii. Ordonati rezultatele crescator dupa nume, apoi descrescator dupa prenume. Daca pentru un client se respecta
--conditiile de mai sus, dar nu se cunosc informatii personale despre acesta (cod_informatii_client pentru respectivul client este NULL), atunci clientul nu va fi afisat.
--4 rezultate

SELECT DISTINCT ic.numa, ic.prenume, ic.cnp, ic.numar_telefon
FROM Informatii_clienti ic
JOIN Clienti c ON ic.cod_informatii_client = c.cod_informatii_client
JOIN Informatii_bancare ib ON c.cod_client = ib.cod_client
JOIN Carduri_Bancare cb ON cb.cod_informatii_bancare = ib.cod_informatii_bancare
AND cb.cod_client = ib.cod_client
JOIN Adrese a ON a.cod_client = c.cod_client
JOIN Facturi f ON a.cod_adresa = f.cod_adresa
AND a.cod_client = f.cod_client
WHERE f.status = 'Neplatit'
AND TO_NUMBER(EXTRACT(DAY FROM (NEXT_DAY(cb.data_expirare_card, 'Monday')))) <= 30
ORDER BY ic.numa ASC, ic.prenume DESC;
```

	NUME	PRENUME	CNP	NUMAR_TELEFON
1	Andreea	Maria	5021249056732	0724905687
2	Buzatu	Ionut	5021395602456	0712690369
3	Dragos	Andrei	5015730459972	0739503182
4	Ion	Matei	5073905481235	0709326784

Comentarii:

Am facut join pe tabelele Facturi, Adrese, Clienti, Informatii_Clienti, Informatii_Bancare si Carduri_Bancare pentru a putea extrage datele personale si bancare despre clientul caruia ii apartine factura respectivă. Am folosit “EXTRACT(DAY FROM (NEXT_DAY(cb.data_expirare_card, 'Monday')))” pentru a afla data calendaristica ce reprezinta ziua de luni urmatoare expirarii fiecăruia dintre cardurile clientului respectiv, apoi am extras ziua din aceasta data. Dacă un client are mai multe carduri se va verifica fiecare dintre acestea, iar dacă exista cel puțin un card care exista condiția atunci clientul va fi selectat. Dacă acesta are mai multe carduri ce respecta condiția, clientul nu va fi afișat de mai multe ori deoarece am folosit DISTINCT.

S-au folosit in aceasta cerere:

- Operație join pe cel puțin 4 tabele
- Ordonari
- 2 functii pe date calendaristice (NEXT_DAY și EXTRACT DAY FROM)
- filtrare la nivel de linii

2) Cerere 2:

Enunt: Afisati numele, prenumele, salariul si data angajarii muncitorilor care au participat la interventiile survenite in urma plangerilor depuse de clientul al carui nume se termina cu 'a' si prenumele incepe cu 'M'. Daca nu se cunosc informatii despre muncitorul afisat (cod_informatii_muncitor pentru respectivul muncitor este NULL) se vor afisa NULL in locul numelui si prenumelui.

```
WITH
cod_mun_cautati AS (SELECT DISTINCT d.cod_muncitor
                     FROM Documente d, Clienti c, Plangeri p
                     WHERE d.cod_client = p.cod_client
                     AND p.cod_client = c.cod_client
                     AND c.cod_informatii_client = (SELECT cod_informatii_client
                                                    FROM Informatii_clienti
                                                    WHERE UPPER(nume) LIKE('%A')
                                                    AND LOWER(prenume) LIKE('m%'))),

cod_inf_munc_cautati AS (SELECT m.cod_muncitor, im.cod_informatii_muncitor
                        FROM Muncitori m, Informatii_muncitori im
                        WHERE m.cod_muncitor in (SELECT cod_muncitor
                                                FROM cod_mun_cautati)
                        AND m.cod_informatii_muncitor = im.cod_informatii_muncitor)

SELECT DECODE((SELECT cod_informatii_muncitor
               FROM cod_inf_munc_cautati
               WHERE cod_inf_munc_cautati.cod_muncitor = m.cod_muncitor), NULL, NULL,
              (SELECT nume
               FROM Informatii_muncitori im, cod_inf_munc_cautati imc
```


WHERE im.cod_informatii_muncitor = imc.cod_informatii_muncitor
AND imc.cod_muncitor = m.cod_muncitor)) AS "Nume",

DECODE((SELECT cod_informatii_muncitor
FROM cod_inf_munc_cautati
WHERE cod_inf_munc_cautati.cod_muncitor = m.cod_muncitor), NULL, NULL,
(SELECT prenume
FROM Informatii_muncitori im, cod_inf_munc_cautati imc
WHERE im.cod_informatii_muncitor = imc.cod_informatii_muncitor
AND imc.cod_muncitor = m.cod_muncitor)) AS "Prenume",

m.salariu, m.data_angajare

FROM Muncitori m
WHERE m.cod_muncitor in (SELECT *
FROM cod_mun_cautati);

--CERERE 2

--Afisati numele, prenumele, salariul si data angajarii muncitorilor care au participat la interventiile survenite in urma plangerilor
--depuze de clientul al carui nume se termina cu 'a' si prenumele incepe cu 'M'. Daca nu se cunosc informatii despre muncitorul afisat (cod_informatii_muncitor pentru
--respectivul muncitor este NULL) se vor afisa NULL in locul numelui si prenumelui
--3 rezultate
WITH
cod_mun_cautati AS (SELECT DISTINCT d.cod_muncitor
FROM Documente d, Clienti c, Plangeri p
WHERE d.cod_client = p.cod_client
AND p.cod_client = c.cod_client
AND c.cod_informatii_client = (SELECT cod_informatii_client
FROM Informatii_clienti
WHERE UPPER(nume) LIKE ('%A')
AND LOWER(prenume) LIKE ('M%'))),
cod_inf_munc_cautati AS (SELECT m.cod_muncitor, im.cod_informatii_muncitor
FROM Muncitori m, Informatii_muncitori im
WHERE m.cod_muncitor in (SELECT cod_muncitor
FROM cod_mun_cautati)
AND m.cod_informatii_muncitor = im.cod_informatii_muncitor)
SELECT DECODE((SELECT cod_informatii_muncitor
FROM cod_inf_munc_cautati
WHERE cod_inf_munc_cautati.cod_muncitor = m.cod_muncitor), NULL, NULL, (SELECT nume
FROM Informatii_muncitori im, cod_inf_munc_cautati imc
WHERE im.cod_informatii_muncitor = imc.cod_informatii_muncitor
AND imc.cod_muncitor = m.cod_muncitor)) AS "Nume", DECODE((SELECT cod_informatii_muncitor
FROM cod_inf_munc_cautati
WHERE cod_inf_munc_cautati.cod_muncitor = m.cod_muncitor), NULL, NULL, (SELECT prenume
FROM Informatii_muncitori im, cod_inf_munc_cautati imc
WHERE im.cod_informatii_muncitor = imc.cod_informatii_muncitor
AND imc.cod_muncitor = m.cod_muncitor)) AS "Prenume", m.salariu, m.data_angajare
FROM Muncitori m
WHERE m.cod_muncitor in (SELECT *
FROM cod_mun_cautati);

Query Result x
All Rows Fetched: 3 in 0.14 seconds

Nume	Prenume	SALARIU	DATA_ANGAJARE
1 Ion	Andrei	2300	23-NOV-19
2 (null)	(null)	1000	27-DEC-21
3 Loredana Mariana		4000	14-AUG-17

if_munc_cautati imc
imc.cod_informatii_muncitor
2)) AS "Nume", DECODE((SELECT cod_informatii_muncitor
FROM cod_inf_munc_cautati
WHERE cod_inf_munc_cautati.cod_muncitor = m.cod_muncitor), NULL, NULL, (SELECT prenume
FROM Informatii_muncitori im, cod_inf_munc_cautati imc
WHERE im.cod_informatii_muncitor = imc.cod_informatii_muncitor
AND imc.cod_muncitor = m.cod_muncitor)) AS "Prenume", m.salariu, m.data_angajare

Script Output x Query Result x
All Rows Fetched: 3 in 0.003 seconds

Nume	Prenume	SALARIU	DATA_ANGAJARE
1 Loredana Mariana		4000	14-AUG-17
2 Ion	Andrei	2300	23-NOV-19
3 (null)	(null)	1000	27-DEC-21

```

AND p.cod_client = c.cod_client
AND c.cod_informatii_client = (SELECT cod_informatii_client
                                FROM Informatii_clienti
                                WHERE UPPER(name) LIKE ('%A')
                                AND LOWER(prenume) LIKE ('m%')),
cod_inf_munc_cautati AS (SELECT m.cod_muncitor, im.cod_informatii_muncitor
                          FROM Muncitori m, Informatii_muncitori im
                          WHERE m.cod_muncitor in (SELECT cod_muncitor
                                                    FROM cod_mun_cautati)
                          AND m.cod_informatii_muncitor = im.cod_informatii_muncitor)
SELECT DECODE((SELECT cod_informatii_muncitor
               FROM cod_inf_munc_cautati
               WHERE cod_inf_munc_cautati.cod_muncitor = m.cod_muncitor), NULL, NULL, (SELECT nume
                                                                                       FROM Informatii_muncitori
                                                                                       WHERE im.cod_informatii_muncitor = m.cod_informatii_muncitor
                                                                                       AND imc.cod_muncitor = m.cod_muncitor)
FROM Muncitori m
WHERE m.cod_muncitor in (SELECT *
                        FROM cod_mun_cautati);
--CERERE_3

```

Script Output x Query Result x

SQL | All Rows Fetched: 3 in 0.003 seconds

	Nume	Prenume	SALARIU	DATA_ANGAJARE
1	Loredana	Mariana	4000	14-AUG-17
2	Ion	Andrei	2300	23-NOV-19
3	(null)	(null)	1000	27-DEC-21

Comentarii:

Am definit 2 blocuri folosind WITH. Primul bloc “cod_mun_cautati” contine codurile muncitorilor care au participat la intervenții ce au survenit în urma plangerilor depuse de clientul al cărui nume se termina cu ‘a’ și prenumele începe cu ‘M’. Al doilea bloc “cod_inf_munc_cautati” contine pentru fiecare dintre muncitorii selectati mai sus in “cod_mun_cautati” codul acestora si codul informatiilor despre acestia (adica contine aceleasi informaii ca si “cod_mun_cautati” + cod_informatii_muncitor pentru fiecare muncitor. Am ales sa creez 2 blocuri distincte in loc de unul singur pentru a face codul mai lizibil). Apoi folosim DECODE pentru a selecta numele si prenumele muncitorilor. În decode folosim pentru ‘value’ o condiție care selectează cod_informatii_muncitori, daca aceasta e NULL atunci pentru nume/prenume vom afișa NULL, altfel vom selecta numele/prenumele in tabela Informatii_Muncitori. Apoi mai ramana sa selectam salariul și data angajării; toate selectiile de mai sus se vor face pentru muncitorii ale caror coduri se afla in blocul “cod_mun_cautati”.

S-au folosit in aceasta cerere:

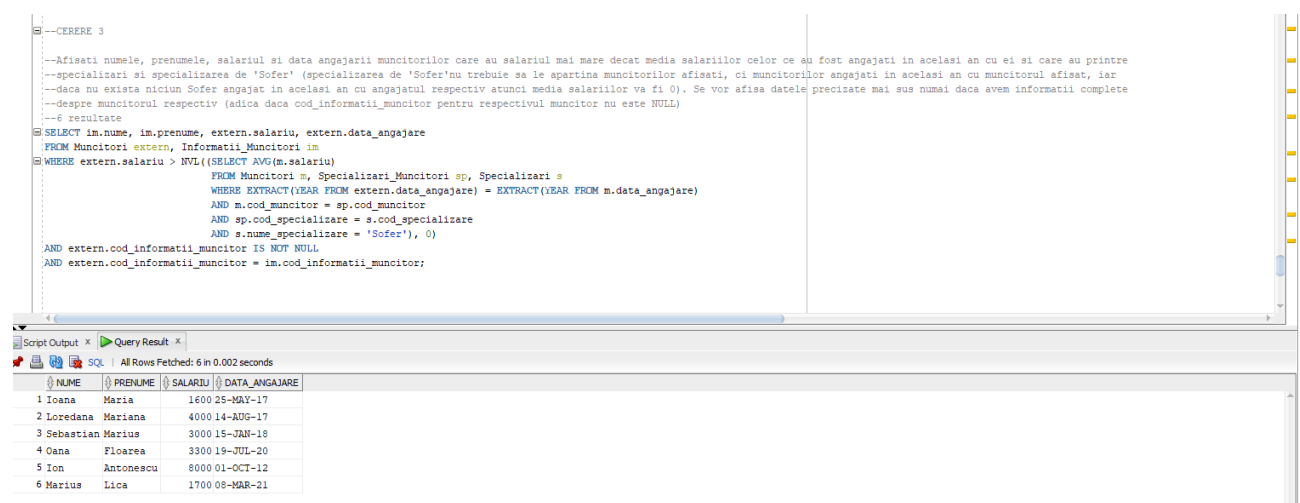
- Subcerere nesincronizata în care intervin 3 tabele
- utilizarea a cel puțin un bloc de cerere (WITH)

- 2 functii pe șiruri de caractere(LOWER si UPPER)
- DECODE

3) Cerere 3:

Enunt: Afisati numele, prenumele, salariul si data angajarii muncitorilor care au salariul mai mare decat media salariilor celor ce au fost angajati in acelasi an cu ei si care au printre specializari si specializarea de 'Sofer' (specializarea de 'Sofer' nu trebuie sa le apartina muncitorilor afisati, ci muncitorilor angajati in acelasi an cu muncitorul afisat, iar daca nu exista niciun Sofer angajat in acelasi an cu angajatul respectiv atunci media salariilor va fi 0). Se vor afisa datele precizate mai sus numai daca avem informatii complete despre muncitorul respectiv (adica daca cod_informatii_muncitor pentru respectivul muncitor nu este NULL)

```
SELECT im.ume, im.prenume, extern.salariu, extern.data_angajare
FROM Muncitori extern, Informatii_Muncitori im
WHERE extern.salariu > NVL((SELECT AVG(m.salariu)
                           FROM Muncitori m, Specializari_Muncitori sp, Specializari s
                           WHERE EXTRACT(YEAR FROM extern.data_angajare) = EXTRACT(YEAR
FROM m.data_angajare)
                           AND m.cod_muncitor = sp.cod_muncitor
                           AND sp.cod_specializare = s.cod_specializare
                           AND s.ume_specializare = 'Sofer'), 0)
AND extern.cod_informatii_muncitor IS NOT NULL
AND extern.cod_informatii_muncitor = im.cod_informatii_muncitor;
```



```
--CERERE 3
--Afisati numele, prenumele, salariul si data angajarii muncitorilor care au salariul mai mare decat media salariilor celor ce au fost angajati in acelasi an cu ei si care au printre
--specializari si specializarea de 'Sofer' (specializarea de 'Sofer' nu trebuie sa le apartina muncitorilor afisati, ci muncitorilor angajati in acelasi an cu muncitorul afisat, iar
--daca nu exista niciun Sofer angajat in acelasi an cu angajatul respectiv atunci media salariilor va fi 0). Se vor afisa datele precizate mai sus numai daca avem informatii complete
--despre muncitorul respectiv (adica daca cod_informatii_muncitor pentru respectivul muncitor nu este NULL)
--6 rezultate
SELECT im.ume, im.prenume, extern.salariu, extern.data_angajare
FROM Muncitori extern, Informatii_Muncitori im
WHERE extern.salariu > NVL((SELECT AVG(m.salariu)
                           FROM Muncitori m, Specializari_Muncitori sp, Specializari s
                           WHERE EXTRACT(YEAR FROM extern.data_angajare) = EXTRACT(YEAR FROM m.data_angajare)
                           AND m.cod_muncitor = sp.cod_muncitor
                           AND sp.cod_specializare = s.cod_specializare
                           AND s.ume_specializare = 'Sofer'), 0)
AND extern.cod_informatii_muncitor IS NOT NULL
AND extern.cod_informatii_muncitor = im.cod_informatii_muncitor;
```

	NUME	PRENUME	SALARIU	DATA_ANGAJARE
1	Ioana	Maria	1600	25-MAY-17
2	Loredana	Mariana	4000	14-AUG-17
3	Sebastian	Marius	3000	15-JAN-18
4	Oana	Floarea	3300	19-JUL-20
5	Ion	Antonescu	8000	01-OCT-12
6	Marius	Lica	1700	08-MAR-21

Comentarii:

Folosim o subcerere cu NVL pentru a extrage salariul mediu al Șoferilor angajați în același an cu angajatul respectiv. In NVL avem ca prim element media salariilor celor care au printre specializari și specializarea "Șofer", iar pentru a verifica dacă aceștia au fost angajați în același an cu angajatul verificat folosim o subcerere sincronizata. Dacă nu exista soferi angajați în același an cu angajatul respectiv (adică subcererea returnează NULL) atunci NVL va returna 0. Facem acest lucru pentru fiecare dintre muncitorii din tabela Muncitori care au atribuit un cod_informatii_muncitor (pentru muncitorii care au cod_informatii_muncitor NULL nu se afiseaza nimic chiar dacă acestea respecta condițiile de mai sus).

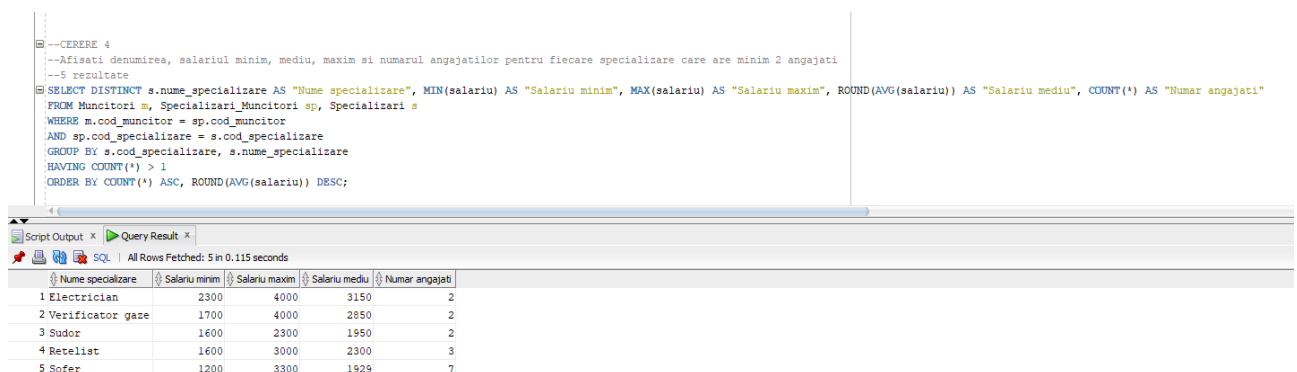
S-au folosit in aceasta cerere:

- Subcerere sincronizata in care intervin 3 tabele
- Funcție pe data calendaristică (EXTRACT YEAR FROM)
- NVL
- Filtrare la nivel de linii

4) Cerere 4:

Enunt: Afisati denumirea, salariul minim, mediu, maxim si numarul angajatilor pentru fiecare specializare care are minim 2 angajati.

```
SELECT DISTINCT s.ume_specializare AS "Nume specializare", MIN(salariu) AS "Salariu minim", MAX(salariu) AS "Salariu maxim", ROUND(AVG(salariu)) AS "Salariu mediu", COUNT(*) AS "Numar angajati"
FROM Muncitori m, Specializari_Muncitori sp, Specializari s
WHERE m.cod_muncitor = sp.cod_muncitor
AND sp.cod_specializare = s.cod_specializare
GROUP BY s.cod_specializare, s.ume_specializare
HAVING COUNT(*) > 1
ORDER BY COUNT(*) ASC, ROUND(AVG(salariu)) DESC;
```



```
--CERERE 4
--Afisati denumirea, salariul minim, mediu, maxim si numarul angajatilor pentru fiecare specializare care are minim 2 angajati
--5 rezultate
SELECT DISTINCT s.ume_specializare AS "Nume specializare", MIN(salariu) AS "Salariu minim", MAX(salariu) AS "Salariu maxim", ROUND(AVG(salariu)) AS "Salariu mediu", COUNT(*) AS "Numar angajati"
FROM Muncitori m, Specializari_Muncitori sp, Specializari s
WHERE m.cod_muncitor = sp.cod_muncitor
AND sp.cod_specializare = s.cod_specializare
GROUP BY s.cod_specializare, s.ume_specializare
HAVING COUNT(*) > 1
ORDER BY COUNT(*) ASC, ROUND(AVG(salariu)) DESC;
```

	Nume specializare	Salariu minim	Salariu maxim	Salariu mediu	Numar angajati
1	Electrician	2300	4000	3150	2
2	Verificator gaze	1700	4000	2850	2
3	Sudor	1600	2300	1950	2
4	Retelist	1600	3000	2300	3
5	Sofer	1200	3300	1929	7

Comentarii:

Grupam specializările după codul și numele acestora. Le selectam și afisam pe cele care au minim 2 angajați. Le ordonam crescator dupa numarul de angajati, iar în caz de egalitate descrescător după salariul mediu.

S-au folosit in aceasta cerere:

- Grupări de date, funcții grup, filtrare la nivel de grupuri
- Ordonari

5) Cerere 5:

Enunt: Sa se afiseaze numele, prenumele, data angajarii, ratingul, salariul actual si salariul dupa marire a primilor 5 angajati cu cel mai mare rating. Salariul urmeaza sa se mareasca numai angajatilor care se afla in top 5 dupa rating astfel: deloc daca este angajat dupa 2021 (inclusiv), cu 10% daca este angajat din 2020, cu 20% daca este angajat din 2019, cu 30% daca este angajat din 2018, cu 40% daca este angajat din 2017, cu 50% daca este angajat inainte de 2017.

```
WITH
aux AS (SELECT cod_muncitor, cod_informatii_muncitor, salariu, data_angajare, rating
        FROM Muncitori
        WHERE rating IS NOT NULL
        ORDER BY rating DESC)

SELECT DECODE(a.cod_informatii_muncitor, NULL, NULL,
              (SELECT im.nume
               FROM Informatii_Muncitori im
               WHERE im.cod_informatii_muncitor = a.cod_informatii_muncitor)) AS "Nume",

DECODE(a.cod_informatii_muncitor, NULL, NULL,
       (SELECT im.prenume
        FROM Informatii_Muncitori im
        WHERE im.cod_informatii_muncitor = a.cod_informatii_muncitor)) AS "Prenume",

data_angajare AS "Data Angajare", rating AS "Rating", salariu AS "Salariu actual",
CASE TO_NUMBER(EXTRACT (YEAR FROM data_angajare))
WHEN 2020
THEN salariu
WHEN 2021
THEN salariu
WHEN 2020
THEN salariu * 1.1
WHEN 2019
THEN salariu * 1.2
```

```

WHEN 2018
THEN salariu * 1.3
WHEN 2017
THEN salariu * 1.4
ELSE salariu * 1.5
END AS "Salariu dupa marire"
FROM aux a
WHERE ROWNUM <= 5;

```

```

--CERERE 5
--Sa se afiseaze numele, prenumele, data angajarii, ratingul, salariul actual si salariul dupa marire a primilor 5 angajati cu cel mai mare rating. Salariul urmeaza sa se
--mareasca numai angajatilor care se afla in top 5 dupa rating astfel: deloc daca este angajat dupa 2021 (inclusiv), cu 10% daca este angajat din 2020, cu 20% daca este
--angajat din 2019, cu 30% daca este angajat din 2018, cu 40% daca este angajat din 2017, cu 50% daca este angajat inainte de 2017
--5 rezultate
WITH
aux AS (SELECT cod_muncitor, cod_informatii_muncitor, salariu, data_angajare, rating
FROM Muncitori
WHERE rating IS NOT NULL
ORDER BY rating DESC)
SELECT DECODE(a.cod_informatii_muncitor, NULL, NULL, (SELECT im.nume
FROM Informatii_Muncitori im
WHERE im.cod_informatii_muncitor = a.cod_informatii_muncitor)) AS "Nume",
DECODE(a.cod_informatii_muncitor, NULL, NULL, (SELECT im.prenume
FROM Informatii_Muncitori im
WHERE im.cod_informatii_muncitor = a.cod_informatii_muncitor)) AS "Prenume",
data_angajare AS "Data Angajare", rating AS "Rating", salariu AS "Salariu actual",
CASE TO_NUMBER(EXTRACT (YEAR FROM data_angajare))
WHEN 2020
THEN salariu
WHEN 2021
THEN salariu
WHEN 2020
THEN salariu * 1.1
WHEN 2019
THEN salariu * 1.2
WHEN 2018
THEN salariu * 1.3
WHEN 2017
THEN salariu * 1.4
ELSE salariu * 1.5
END AS "Salariu dupa marire"
FROM aux a
WHERE ROWNUM <= 5;

```

Nume	Prenume	Data Angajare	Rating	Salariu actual	Salariu dupa marire
1 Ion	Antonescu	01-OCT-12	800	8000	12000
2 Ionut	Ionel	24-SEP-20	555	2000	2000
3 Miruna	Mircea	13-DEC-20	456	1200	1200
4 Vasile	Vasilevici	13-DEC-17	456	1300	1820
5 Loredana	Mariana	14-AUG-17	400	4000	5600

Comentarii:

Creem un bloc în care reținem `cod_muncitor`, `cod_informatii_muncitor`, `salariu`, `data_angajare`, `rating` pentru fiecare muncitor care are rating (adică un rating nu este NULL). Ordonăm acest bloc descrescător după rating. Apoi folosind `DECODE` verificăm dacă pentru respectivul muncitor cunoaștem informații (adică `cod_informatii_muncitor` este diferit de NULL). Dacă cunoaștem informații atunci selectăm numele/prenumele, altfel în locul numelui/prenumelui vom afișa NULL. Folosim `CASE` pentru a calcula salariul după mărire a fiecărui angajat care se afla în top 5. La final folosim `WHERE ROWNUM <= 5` pentru a selecta angajații cu cel mai mare rating.

S-au folosit in aceasta cerere:

- Bloc de cerere (WITH)
- Ordonari
- Top n
- DECODE
- CASE
- Functie pe date calendaristice (EXTRACT YEAR FROM)

13. Cele 3 operații de actualizare și suprimare a datelor:

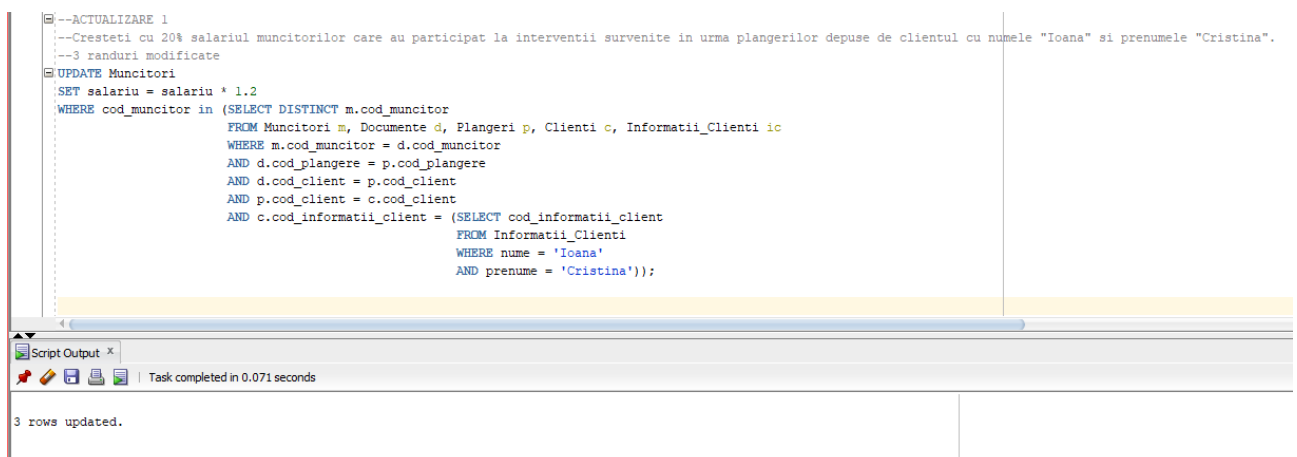
1) Operație 1:

Enunt: Cresteti cu 20% salariul muncitorilor care au participat la interventii survenite in urma plangerilor depuse de clientul cu numele "Ioana" si prenumele "Cristina".

UPDATE Muncitori

SET salariu = salariu * 1.2

WHERE cod_muncitor in (SELECT DISTINCT m.cod_muncitor
FROM Muncitori m, Documente d, Plangeri p, Clienti c, Informatii_Clienti ic
WHERE m.cod_muncitor = d.cod_muncitor
AND d.cod_plangere = p.cod_plangere
AND d.cod_client = p.cod_client
AND p.cod_client = c.cod_client
AND c.cod_informatii_client = (SELECT cod_informatii_client
FROM Informatii_Clienti
WHERE nume = 'Ioana'
AND prenume = 'Cristina'));



```
--ACTUALIZARE 1
--Cresteti cu 20% salariul muncitorilor care au participat la interventii survenite in urma plangerilor depuse de clientul cu numele "Ioana" si prenumele "Cristina".
--3 randuri modificate
UPDATE Muncitori
SET salariu = salariu * 1.2
WHERE cod_muncitor in (SELECT DISTINCT m.cod_muncitor
FROM Muncitori m, Documente d, Plangeri p, Clienti c, Informatii_Clienti ic
WHERE m.cod_muncitor = d.cod_muncitor
AND d.cod_plangere = p.cod_plangere
AND d.cod_client = p.cod_client
AND p.cod_client = c.cod_client
AND c.cod_informatii_client = (SELECT cod_informatii_client
FROM Informatii_Clienti
WHERE nume = 'Ioana'
AND prenume = 'Cristina'));
```





Script Output x

Task completed in 0.071 seconds

3 rows updated.

a) Datele din tabel înainte de modificare:


Script Output x | Query Result x

    SQL | All Rows Fetched: 13 in 0.004 seconds

	↕ COD_MUNCITOR	↕ COD_INFORMATII_MUNCITOR	↕ SALARIU	↕ DATA_ANGAJARE	↕ RATING
1	1	1	2300	23-NOV-19	20
2	2	2	1800	10-SEP-20	50
3	3	(null)	1000	27-DEC-21	10
4	4	6	3000	15-JAN-18	200
5	5	8	8000	01-OCT-12	800
6	6	5	4000	14-AUG-17	400
7	7	(null)	2300	20-FEB-20	(null)
8	8	7	3300	19-JUL-20	70
9	9	9	1700	08-MAR-21	30
10	10	3	1600	25-MAY-17	(null)
11	11	11	1200	13-DEC-20	456
12	12	12	2000	24-SEP-20	555
13	13	13	1300	13-DEC-17	456

b) Datele din tabel după modificare:

Script Output x | Query Result x

 | All Rows Fetched: 13 in 0.017 seconds

	⚡ COD_MUNCITOR	⚡ COD_INFORMATII_MUNCITOR	⚡ SALARIU	⚡ DATA_ANGAJARE	⚡ RATING
1	1	1	2300	23-NOV-19	20
2	2	2	2160	10-SEP-20	50
3	3	(null)	1000	27-DEC-21	10
4	4	6	3000	15-JAN-18	200
5	5	8	9600	01-OCT-12	800
6	6	5	4800	14-AUG-17	400
7	7	(null)	2300	20-FEB-20	(null)
8	8	7	3300	19-JUL-20	70
9	9	9	1700	08-MAR-21	30
10	10	3	1600	25-MAY-17	(null)
11	11	11	1200	13-DEC-20	456
12	12	12	2000	24-SEP-20	555
13	13	13	1300	13-DEC-17	456

Comentarii:

Sunt folosite 2 subcereri. Una care sa returneze “cod_informatii_client” pentru clientul numit “Ioana Cristina”, iar cealalta pentru a selecta codul muncitorilor care au participat la intervenții survenite în urma plangerilor depuse de acest client. Acestor muncitori le este mărit salariul cu 20%.

2) Operație 2:

Enunt: Stergeti facturile cu statusul 'Neplatit' ale clientilor despre care nu se cunosc informatii (au cod_informatii_client NULL) sau despre care se cunosc informatii si al caror numar de telefon incepe cu "073".

DELETE FROM Facturi

WHERE cod_client in (SELECT DISTINCT c.cod_client
FROM Clienti c, Informatii_Clienti ic, Facturi f
WHERE f.cod_client = c.cod_client
AND (c.cod_informatii_client IS NULL
OR (c.cod_informatii_client = ic.cod_informatii_client
AND ic.numar_telefon LIKE('073%'))))
AND status = 'Neplatit';

```

--ACTUALIZARE 2
--Stergeti facturile cu statusul 'Neplatit' ale clientilor despre care nu se cunosc informatii (au cod_informatii_client NULL) sau despre care se cunosc informatii si al
--caror numar de telefon incepe cu "073".
--4 randuri sterse
DELETE FROM Facturi
WHERE cod_client in (SELECT DISTINCT c.cod_client
FROM Clienti c, Informatii_Clienti ic, Facturi f
WHERE f.cod_client = c.cod_client
AND (c.cod_informatii_client IS NULL
OR (c.cod_informatii_client = ic.cod_informatii_client
AND ic.numar_telefon LIKE ('073%'))))
AND status = 'Neplatit';

```

Script Output x

Task completed in 0.066 seconds

4 rows deleted.

a) Datele din tabel inainte de stergere:

	COD_FACTURA	COD_ADRESA	COD_CLIENT	TOTAL	DATA_ELIBERARE	TERMEN_PLATA	STATUS
1	1	1	1	200	18-FEB-24	23-FEB-25	Platit
2	2	1	1	350	04-MAR-23	15-APR-24	In procesare
3	3	1	1	570	20-SEP-22	01-APR-26	Platit
4	1	2	4	700	13-DEC-22	17-APR-24	Neplatit
5	2	2	4	620	07-AUG-24	15-APR-28	Neplatit
6	1	1	7	200	23-FEB-23	27-OCT-27	Neplatit
7	2	1	7	850	20-AUG-23	27-APR-25	In procesare
8	1	2	10	340	14-JAN-23	19-NOV-23	Platit
9	2	2	10	890	27-DEC-24	08-APR-27	Neplatit
10	3	2	10	510	03-APR-23	22-APR-25	Neplatit
11	1	1	2	280	24-SEP-20	27-APR-22	Neplatit
12	1	1	9	320	14-DEC-22	25-MAR-25	Neplatit
13	1	1	11	765	05-FEB-22	02-MAY-24	Neplatit
14	1	1	12	180	19-APR-21	02-APR-22	Neplatit

b) Datele din tabel dupa stergere:

	COD_FACTURA	COD_ADRESA	COD_CLIENT	TOTAL	DATA_ELIBERARE	TERMEN_PLATA	STATUS
1	1	1	1	200	18-FEB-24	23-FEB-25	Platit
2	2	1	1	350	04-MAR-23	15-APR-24	In procesare
3	3	1	1	570	20-SEP-22	01-APR-26	Platit
4	1	2	4	700	13-DEC-22	17-APR-24	Neplatit
5	2	2	4	620	07-AUG-24	15-APR-28	Neplatit
6	2	1	7	850	20-AUG-23	27-APR-25	In procesare
7	1	2	10	340	14-JAN-23	19-NOV-23	Platit
8	1	1	9	320	14-DEC-22	25-MAR-25	Neplatit
9	1	1	11	765	05-FEB-22	02-MAY-24	Neplatit
10	1	1	12	180	19-APR-21	02-APR-22	Neplatit

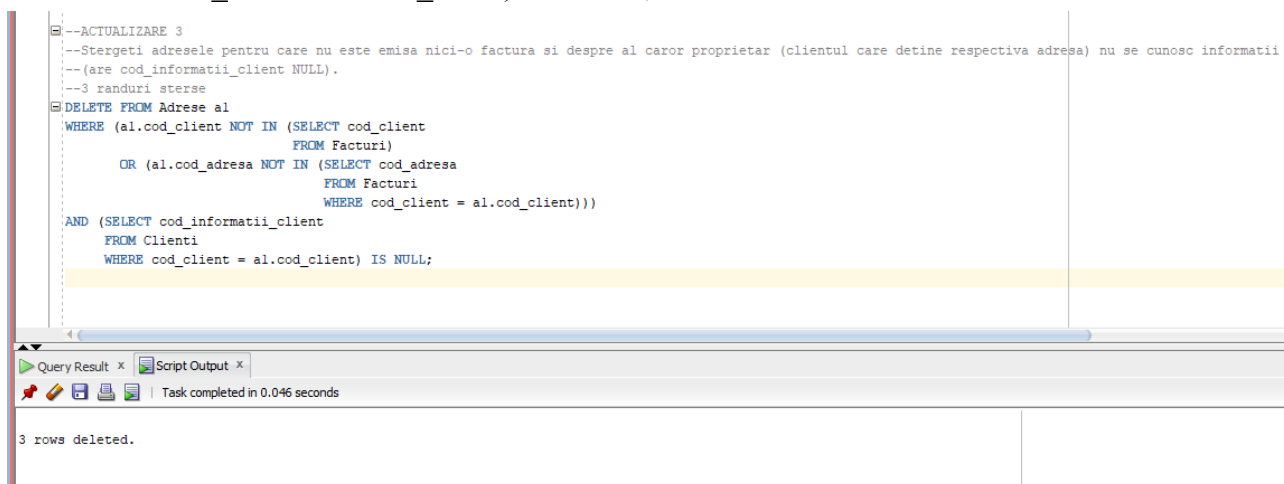
Comentarii:

Folosim o subcerere care returnează codurile clienților care au facturi și despre care nu se cunosc informații sau se cunosc și al căror număr de telefon începe cu “073”. Apoi stingem facturile neplătite ale acestor clienți;

3) Operație 3:

Enunț: Stergeți adresele pentru care nu este emisă nici-o factură și despre al căror proprietar (clientul care deține respectiva adresă) nu se cunosc informații (are cod_informatii_client NULL).

```
DELETE FROM Adrese a1
WHERE (a1.cod_client NOT IN (SELECT cod_client
                             FROM Facturi)
      OR (a1.cod_adresa NOT IN (SELECT cod_adresa
                                FROM Facturi
                                WHERE cod_client = a1.cod_client)))
AND (SELECT cod_informatii_client
     FROM Clienti
     WHERE cod_client = a1.cod_client) IS NULL;
```



The screenshot shows a SQL query execution window with the following content:

```
--ACTUALIZARE 3
--Stergeti adresele pentru care nu este emisă nici-o factură și despre al căror proprietar (clientul care deține respectiva adresă) nu se cunosc informații
--(are cod_informatii_client NULL).
--3 randuri sterse
DELETE FROM Adrese a1
WHERE (a1.cod_client NOT IN (SELECT cod_client
                             FROM Facturi)
      OR (a1.cod_adresa NOT IN (SELECT cod_adresa
                                FROM Facturi
                                WHERE cod_client = a1.cod_client)))
AND (SELECT cod_informatii_client
     FROM Clienti
     WHERE cod_client = a1.cod_client) IS NULL;
```

Below the query, the status bar indicates "Task completed in 0.046 seconds". At the bottom, a message states "3 rows deleted."

a) Datele din tabel înainte de stergere:

SQL All Rows Fetched: 15 in 0.065 seconds						
	COD_ADRESA	COD_CLIENT	TARA	ORAS	STRADA	NUMAR
1	1	1	Romania	Craiova	Brestei	56
2	2	1	Romania	Bucuresti	Drumul Taberei	23
3	3	1	Olanda	Amsterdam	Amsterdam Street	4535
4	1	4	Romania	Iasi	Strada Iasului	654
5	2	4	Romania	Timisioara	Aleea Actorilor	547
6	1	7	Republica Moldova	Chisinau	Strada Mosilor	234
7	1	8	Franta	Paris	Sans Elise	1235
8	1	10	Anglia	Londra	Fournier Street	12
9	2	10	Romania	Constanta	Faleza Marii	634
10	3	10	Romania	Cluj	Strada Clujului	8765
11	1	2	Romania	Arad	Strda Aradului	65
12	2	2	Romania	Bucuresti	Crangasi	654
13	1	9	Romania	Buzau	Strda Buzaului	23
14	1	11	Romania	Botosani	Strda Botosani	653
15	1	12	Romania	Vaslui	Strda Vslui	13

b) Datele din tabel după ștergere:

SQL All Rows Fetched: 12 in 0.003 seconds						
	COD_ADRESA	COD_CLIENT	TARA	ORAS	STRADA	NUMAR
1	1	1	Romania	Craiova	Brestei	56
2	2	1	Romania	Bucuresti	Drumul Taberei	23
3	3	1	Olanda	Amsterdam	Amsterdam Street	4535
4	1	4	Romania	Iasi	Strada Iasului	654
5	2	4	Romania	Timisioara	Aleea Actorilor	547
6	1	7	Republica Moldova	Chisinau	Strada Mosilor	234
7	1	8	Franta	Paris	Sans Elise	1235
8	2	10	Romania	Constanta	Faleza Marii	634
9	1	2	Romania	Arad	Strda Aradului	65
10	1	9	Romania	Buzau	Strda Buzaului	23
11	1	11	Romania	Botosani	Strda Botosani	653
12	1	12	Romania	Vaslui	Strda Vslui	13

Comentari:

Mai întâi este făcută o verificare pentru a găsi adresele fără facturi. Dacă codul clientului adresei respective nu se găsește printre codurile clienților din tabelul Facturi înseamnă ca pentru adresa

respectivă nu au fost emise facturi. Astfel, se verifica dacă codul adresei respective este în tabelul Facturi printre randurile care au ca și “cod_client” codul clientului ce deține adresa respectivă. Apoi se verifica dacă pentru clientul caruia îi aparține adresa există un obiect “cod_informatii_client asociat”. Dacă nu există un astfel de obiect și pentru adresa respectivă nu au fost emise facturi, atunci ștergem adresa.

16. Cerere **outer-join** și cereri ce utilizează **division**:

OBSERVAȚIE: După ce au fost efectuate update-urile și delete-urile de la ex 13 s-a dat ROLLBACK, așa că cererile de mai jos sunt efectuate pe datele inițiale ale bazei de date.

1) Cerere ce utilizează outer-join:

Enunț: Afișați numele, prenumele, cnp-ul și numărul de telefon al clienților care dețin carduri bancare care expiră înainte de 1 decembrie 2028.

```
SELECT DISTINCT c.cod_client, c.cod_informatii_client, ic.nume, ic.prenume, ic.cnp,
ic.numar_telefon
FROM Carduri_bancare cb
LEFT OUTER JOIN Informatii_Bancare ib
ON cb.cod_informatii_bancare = ib.cod_informatii_bancare
AND cb.cod_client = ib.cod_client
LEFT OUTER JOIN Clienti c
ON ib.cod_client = c.cod_client
LEFT OUTER JOIN informatii_clienti ic
ON c.cod_informatii_client = ic.cod_informatii_client
WHERE cb.data_expirare_card < TO_DATE('1-DEC-2028', 'DD-MON-YYYY');
```

```
--Afisati numele, prenumele, cnp-ul si numarul de telefon al clientilor care detin carduri bancare care expira inainte de 1 decembrie 2028.
--8 rezultate
SELECT DISTINCT c.cod_client, c.cod_informatii_client, ic.numa, ic.prenume, ic.cnp, ic.numar_telefon
FROM Carduri_bancare cb
LEFT OUTER JOIN Informatii_Bancare ib
ON cb.cod_informatii_bancare = ib.cod_informatii_bancare
AND cb.cod_client = ib.cod_client
LEFT OUTER JOIN Clienti c
ON ib.cod_client = c.cod_client
LEFT OUTER JOIN informatii_clienti ic
ON c.cod_informatii_client = ic.cod_informatii_client
WHERE cb.data_expirare_card < TO_DATE('1-DEC-2028', 'DD-MON-YYYY');
```

	COD_CLIENT	COD_INFORMATII_CLIENT	NUME	PRENUME	CNP	NUMAR_TELEFON
1	1		Marian	Andrei	5020413150634	0734135426
2	3		Ioana	Cristina	5022494157062	0732584214
3	4		Buzatu	Ionut	5021395602456	0712690369
4	5		Ana	Cosmina	5021049258568	0725790632
5	8		Cristi	Sebastian	5042458037893	0738952469
6	7		Dragos	Andrei	5015730458972	0739503182
7	2		(null)	(null)	(null)	(null)
8	10		(null)	(null)	(null)	(null)

Comentarii:

Am făcut left outer-join pe Carduri_Bancare, Informatii_Bancare, Clienti si Informatii_Clienti. Am selectat cardurile care expiră înainte de 1 decembrie 2028 și apoi am aflat informații despre clienții acestora, dacă avem cod_informatii_client pentru aceștia, altfel NULL.

2) Prima cerere ce implementează division:

Enunt: Selectati codurile muncitorilor atasati tuturor meseriilor a caror meserie are numele mai scurt de 10 litere.

```
SELECT DISTINCT sp.cod_muncitor
FROM Specializari_Muncitori sp
WHERE NOT EXISTS (SELECT 1
FROM Specializari s
WHERE LENGTH(s.numa_specializare) < 10
AND NOT EXISTS (SELECT 1
FROM Specializari_Muncitori sp2
WHERE s.cod_specializare = sp2.cod_specializare
AND sp2.cod_muncitor = sp.cod_muncitor));
```

```
--Selectati codurile muncitorilor atasati tuturor meseriilor a caror meserie are numele mai scurt de 10 litere
--2 rezultate
SELECT DISTINCT sp.cod_muncitor
FROM Specializari_Muncitori sp
WHERE NOT EXISTS (SELECT 1
                  FROM Specializari s
                  WHERE LENGTH(s.num_e_specializare) < 10
                  AND NOT EXISTS (SELECT 1
                                FROM Specializari_Muncitori sp2
                                WHERE s.cod_specializare = sp2.cod_specializare
                                AND sp2.cod_muncitor = sp.cod_muncitor));
```

Script Output x Query Result x

SQL | All Rows Fetched: 2 in 0.002 seconds

	COD_MUNCITOR
1	1
2	10

Comentarii:

În a doua subcerere selectăm 1 pentru specializările care îndeplinesc condiția din prima subcerere și se află printre specializările muncitorului verificat. În prima subcerere, folosind a doua subcerere, selectăm 1 pentru specializările care îndeplinesc condiția de a avea mai puțin de 10 litere în componența denumirii acestora și pentru care muncitorul pe care îl verificăm nu este specializat. În cererea principală selectăm codul muncitorilor pentru care nu există astfel de specializări, adică aceștia dețin toate specializările care îndeplinesc condiția respectivă.

3) A doua cerere ce implementează division:

Enunț: Selectați codurile clientilor care au adrese în România și pentru fiecare dintre aceste adrese să existe cel puțin o factură emisă.

```
SELECT DISTINCT c.cod_client
FROM Clienti c
WHERE NOT EXISTS (SELECT 1
                  FROM Adrese a
                  WHERE a.tara = 'Romania'
                  AND a.cod_client = c.cod_client
                  AND NOT EXISTS (SELECT 1
                                FROM Facturi f
                                WHERE f.cod_adresa = a.cod_adresa
                                AND f.cod_client = a.cod_client))
AND EXISTS (SELECT 1
```

```

FROM Adrese a2
WHERE a2.tara = 'Romania'
AND a2.cod_client = c.cod_client);

```

--Selectati codurile clientilor care au adrese in Romania si pentru fiecare dintre aceste adrese sa exista cel putin o factura emisa.
 --3 rezultate

```

SELECT DISTINCT c.cod_client
FROM Clienti c
WHERE NOT EXISTS (SELECT 1
                  FROM Adrese a
                  WHERE a.tara = 'Romania'
                  AND a.cod_client = c.cod_client
                  AND NOT EXISTS (SELECT 1
                                FROM Facturi f
                                WHERE f.cod_adresa = a.cod_adresa
                                AND f.cod_client = a.cod_client))
AND EXISTS (SELECT 1
            FROM Adrese a2
            WHERE a2.tara = 'Romania'
            AND a2.cod_client = c.cod_client);

```

Script Output x Query Result x

SQL | All Rows Fetched: 3 in 0.001seconds

	COD_CLIENT
1	11
2	12
3	9

Comentarii:

Folosind a doua subcerere din cadrul primei conditii selectam 1 pentru facturile existente la adresa verificata in cadrul primei subcereri. In cadrul primei subcereri selectam 1 pentru adresele ce indeplinesc conditia de se afla in Romania, care NU au facturi emise si sunt ale clientului verificat in cererea principala. In cererea principala selectam clientii care nu se afla in aceasta situatie, adica ii selectam pe cei ce au facturi emise pentru toate adresele pe care le au in Romania. In cea de a doua conditie (cea care incepe cu AND EXISTS) verificam ca clientii verificati sa aiba cel putin o adresa in Romania, pentru a nu afisa clientii fara adrese, sau pe cei care au adresa, dar niciuna nu este situata in Romania.