

## **Studiu de caz**

Exemplele din acest capitol se referă la proiectarea unui model de date ce furnizează informații despre saloanele cosmetice din România, un loc frecventat mai mult sau mai puțin de orice cetățean.

Voi prezenta modelul de date, restricțiile pe care trebuie să le respecte și voi construi diagrama E/R corespunzătoare, dar și diagrama conceptuală.

Modelul de date va gestiona informații legate de organizarea și funcționarea saloanelor cosmetice din România. Există furnizori pentru aceste saloane – ei se afla în colaborare cu salonul pentru a le aproviziona cu produse cosmetice (vopsea, foarfece, ceara, etc.).

Fiecare salon se afla într-o anumită locație, este curățat de îngrijitori și are mai mulți angajați. Îngrijitorul (îngrijitorii) vine la finalul zilei pentru a pregăti salonul de o nouă zi, ceea ce le permite să fie arondați la mai multe saloane.

Angajații au fiecare câte un job (manichiurist, frizer, etc.) și trebuie să își satisfacă toți clienții. Clienții își fac o programare la un anumit serviciu. Având în vedere că fiecare angajat are specializarea sa, clienții trebuie să-și facă câte o programare pentru fiecare serviciu dorit (de exemplu, dacă un client dorește să se tundă și să-și facă manichiura, el trebuie să-și facă două programări separate).

Modelul de date respectă anumite restricții de funcționare:

- Angajații se ocupă de curățenia din timpul programului (nu este luat în considerare acest aspect), iar îngrijitorii se ocupă de curățenia de la final de zi, astfel încât angajații nu stau peste program.
- Furnizorii aprovizionează salonul cu produse cosmetice și diferite materiale, dar în baza de date apare doar suma totală de bani pe care o valorează aceste produse.
- Orice salon are o locație unică.

## **Entități**

Pentru modelul de date referitor la saloanele cosmetice din România, structurile SALON, LOCAȚIE, ÎNGRIJITOR, ANGAJAT, CLIENT, PROGRAMARE, SERVICIU, JOBS, FURNIZOR reprezintă entități.

Voi prezenta entitățile modelului de date, dând o descriere completă a fiecăreia. De asemenea, pentru fiecare entitate se va preciza cheia primară.

Toate entitățile sunt independente.

*SALON* = loc, firmă care se ocupă cu păstrarea unei aparențe fizice deosebite, dar și cu relaxarea clienților. Cheia primară a entității este *cod\_salon*.

*FURNIZOR* = firmă (persoană juridică) specializată care încheie un contract cu saloanele cosmetice pentru a le aproviziona cu produse cosmetice și materiale necesare. Cheia primară a entității este *cod\_furnizor*.

*LOCAȚIE* = entitate care identifică locația unui salon. Cheia primară a entității este *cod\_locatie*.

*ÎNGRIJITOR* = persoană fizică care se ocupă cu întreținerea curățeniei la final de zi în saloane. Cheia primară a entității este *cod\_ingrijitor*.

*ANGAJAT* = persoană fizică, angajată de un salon pentru satisfacerea cerințelor clienților în funcție de job-ul pe care îl are. Cheia primară a entității este *cod\_angajat*.

*CLIENT* = persoană fizică, își face o programare la salon pentru un anumit serviciu și plătește în schimbul acestuia. Cheia primară a entității este *cod\_client*.

*PROGRAMARE* = identifică datele unei programări a unui client. Cheia primară a entității este *cod\_programare*.

*SERVICIU* = entitate care identifică procedurile pe care le oferă orice salon. Cheia primară a entității este *cod\_serviciu*.

*JOBS* = specializarea unui angajat. Cheia primară a entității este *cod\_job*.

## **Relații**

Voi prezenta relațiile modelului de date, dând o descriere completă a fiecăreia. De fapt, denumirile acestor legături sunt sugestive, reflectând conținutul acestora și entitățile pe care le leagă. Pentru fiecare relație se va preciza cardinalitatea minimă și maximă.

*SALON\_se\_afla\_LOCATIE* = relație care leagă entitățile SALON și LOCATIE, reflectând legătura dintre acestea (în ce locație se află un anumit salon). Ea are cardinalitatea minimă și cea maximă 1:1 (un salon se află în exact o locație).

*SALON\_curatat\_de\_INGRIJITOR* = relație care leagă entitățile SALON și ÎNGRIJITOR, reflectând legătura dintre acestea (îngrijitorul face curat în salon la finalul zilei). Relația are cardinalitatea minimă 1:1 (un îngrijitor trebuie să curețe cel puțin un salon și un salon trebuie să aibă cel puțin un îngrijitor) și cardinalitatea maximă m:n (un salon poate avea mai mulți îngrijitori și un îngrijitor poate lucra la mai multe saloane).

*SALON\_aprovizionat\_de\_FURNIZOR* = relație care leagă entitățile SALON și FURNIZOR, reflectând legătura dintre acestea (furnizorii aprovizionează saloanele cu materialele necesare). Relația are cardinalitatea minimă 1:1 (orice salon are nevoie de minim un furnizor și orice furnizor

trebuie să aprovizioneze măcar un salon) și cardinalitatea maximă m:n (un salon poate avea mai mulți furnizori și un furnizor poate încheia contracte cu mai multe saloane).

*SALON\_are\_ANGAJAT* = relație care leagă entitățile SALON și ANGAJAT, reflectând legătura dintre acestea (angajatul lucrează la salon). Relația are cardinalitatea minimă 1:1 (un angajat trebuie să lucreze la un salon și un salon trebuie să aibă cel puțin un angajat) și cardinalitatea maximă 1:n (un salon poate avea mai mulți angajați, dar un angajat nu poate lucra la mai multe saloane deodată).

*ANGAJAT\_are\_JOBS* = relație care leagă entitățile ANGAJAT și JOB, reflectând legătura dintre acestea (un angajat are o anumită specializare). Relația are cardinalitatea minimă 1:1 și cardinalitatea maximă n:1.

*PROGRAMARE\_contine\_SERVICIU* = relație care leagă entitățile PROGRAMARE și SERVICIU, reflectând legătura dintre acestea (o programare se face la un anumit serviciu). Relația are cardinalitatea 1:1.

*CLIENT\_merge\_la\_SALON\_la\_ANGAJAT* = relație de tip 3 ce leagă entitățile CLIENT, SALON și ANGAJAT, reflectând cine este clientul care merge la un anumit angajat de la un anumit salon. Denumirea acestei relații va fi *merge\_la*.

### **Atribute**

Entitatea independentă SALON are ca atribute:

*cod\_salon* = variabilă de tip întreg, de lungime maximă 6, care reprezintă codul unui salon.

*nume\_salon* = variabilă de tip caracter, de lungime maximă 25, care reprezintă numele salonului.

*cod\_locatie* = variabilă de tip întreg, de lungime maximă 6, care reprezintă codul locației salonului. Atributul trebuie să corespundă la o valoare a cheii primare din tabelul LOCATIE.

Entitatea independentă LOCATIE are ca atribute:

*cod\_locatie* = variabilă de tip întreg, de lungime maximă 6, care reprezintă codul unei locatii.

*judet* = variabilă de tip caracter, de lungime maximă 20, care reprezintă numele județului.

*oras* = variabilă de tip caracter, de lungime maximă 20, care reprezintă numele orașului.

*strada* = variabilă de tip caracter, de lungime maximă 30, care reprezintă numele străzii.

*numar* = variabilă de tip întreg, de lungime maximă 3, care reprezintă numărul străzii.

Entitatea independentă INGRIJITOR are ca attribute:

*cod\_ingrijitor* = variabilă de tip întreg, de lungime maximă 6, care reprezintă codul unui ingrijitor.

*nume\_ingrijitor* = variabilă de tip caracter, de lungime maximă 20, care reprezintă numele îngrijitorului.

*prenume\_ingrijitor* = variabilă de tip caracter, de lungime maximă 20, care reprezintă prenumele îngrijitorului.

*salariu\_ingrijitor* = variabilă de tip întreg, de lungime maximă 6, care reprezintă salariul îngrijitorului.

Entitatea independentă ANGAJAT are ca attribute:

*cod\_angajat* = variabilă de tip întreg, de lungime maximă 6, care reprezintă codul unui angajat.

*nume\_angajat* = variabilă de tip caracter, de lungime maximă 20, care reprezintă numele angajatului.

*prenume\_angajat* = variabilă de tip caracter, de lungime maximă 20, care reprezintă prenumele angajatului.

*salariu\_angajat* = variabilă de tip întreg, de lungime maximă 6, care reprezintă salariul angajatului.

*data\_angajarii* = variabilă de tip dată calendaristică, care reprezintă data angajării angajatului respectiv.

*cod\_salon* = variabilă de tip întreg, de lungime maximă 6, care reprezintă codul salonului la care lucrează angajatul. Atributul trebuie să corespundă la o valoare a cheii primare din tabelul SALON.

*cod\_job* = variabilă de tip întreg, de lungime maximă 6, care reprezintă codul job-ului angajatului. Atributul trebuie să corespundă la o valoare a cheii primare din tabelul JOB.

Entitatea independentă CLIENT are ca attribute:

*cod\_client* = variabilă de tip întreg, de lungime maximă 6, care reprezintă codul unui client.

*nume\_client* = variabilă de tip caracter, de lungime maximă 20, care reprezintă numele clientului.

*prenume\_client* = variabilă de tip caracter, de lungime maximă 20, care reprezintă prenumele clientului.

*numar\_telefon* = variabilă de tip caracter, de lungime maximă 13, care reprezintă numărul de telefon al clientului.

*adresa\_email* = variabilă de tip caracter, de lungime maximă 30, care reprezintă adresa de email a clientului.

Entitatea independentă PROGRAMARE are ca atribute:

*cod\_programare* = variabilă de tip întreg, de lungime maximă 6, care reprezintă codul unui programare.

*data\_programare* = variabilă de tip dată calendaristică, care reprezintă data programării.

*cod\_client* = variabilă de tip întreg, de lungime maximă 6, care reprezintă codul clientului care își face programarea. Atributul trebuie să corespundă la o valoare a cheii primare din tabelul CLIENT.

Entitatea independentă SERVICIU are ca atribute:

*cod\_serviciu* = variabilă de tip întreg, de lungime maximă 6, care reprezintă codul unui serviciu.

*denumire* = variabilă de tip caracter, de lungime maximă 20, care reprezintă denumirea serviciului.

*durata\_serviciu* = variabilă de tip întreg, de lungime maximă 1 cu doua zecimale, care reprezintă durata unui serviciu (1 = o oră, 0.5 = 30 minute).

*pret* = variabilă de tip întreg, de lungime maximă 3, care reprezintă prețul unui serviciu.

Entitatea independentă JOBS are ca atribute:

*cod\_job* = variabilă de tip întreg, de lungime maximă 6, care reprezintă codul unui job.

*nume\_job* = variabilă de tip caracter, de lungime maximă 20, care reprezintă numele job-ului.

*salariu\_min* = variabilă de tip întreg, de lungime maximă 5, care reprezintă salariul minim al unui job.

*salariu\_max* = variabilă de tip întreg, de lungime maximă 5, care reprezintă salariul maxim al unui job.

Entitatea independentă FURNIZOR are ca atribute:

*cod\_furnizor* = variabilă de tip întreg, de lungime maximă 6, care reprezintă codul unui furnizor.

*suma* = variabilă de tip întreg, de lungime maximă 10, care reprezintă suma investită în salon.

*data\_contract* = variabilă de tip dată calendaristică, care reprezintă data începerii contractului.

*durata* = variabilă de tip întreg, de lungime maximă 3 cu doua zecimale, care reprezintă durata contractului (1 = 1 an, 1.5 = 1 an și 6 luni). Valoare implicită NULL – perioadă nedeterminată.

### Diagrama entitate-relatie

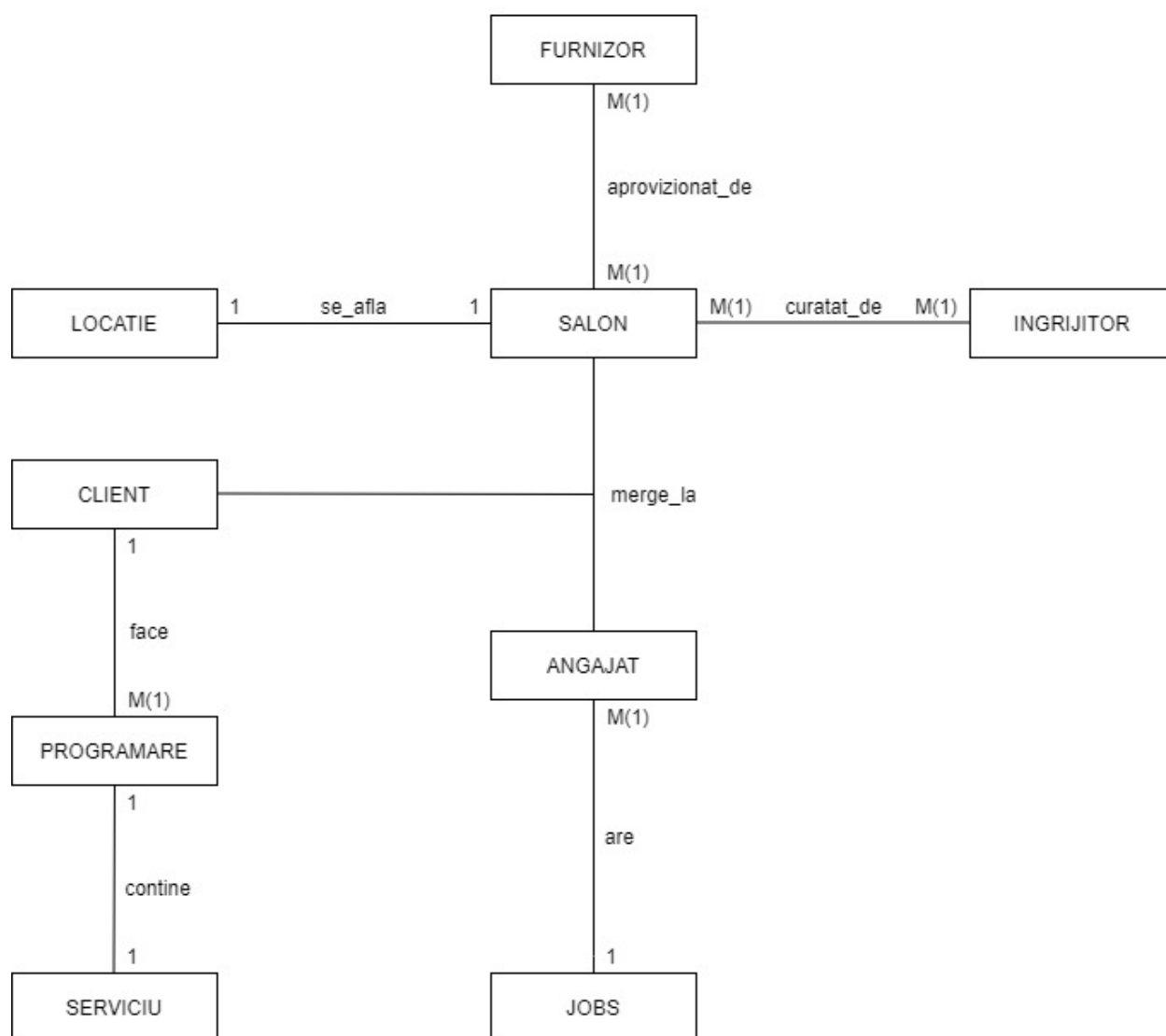
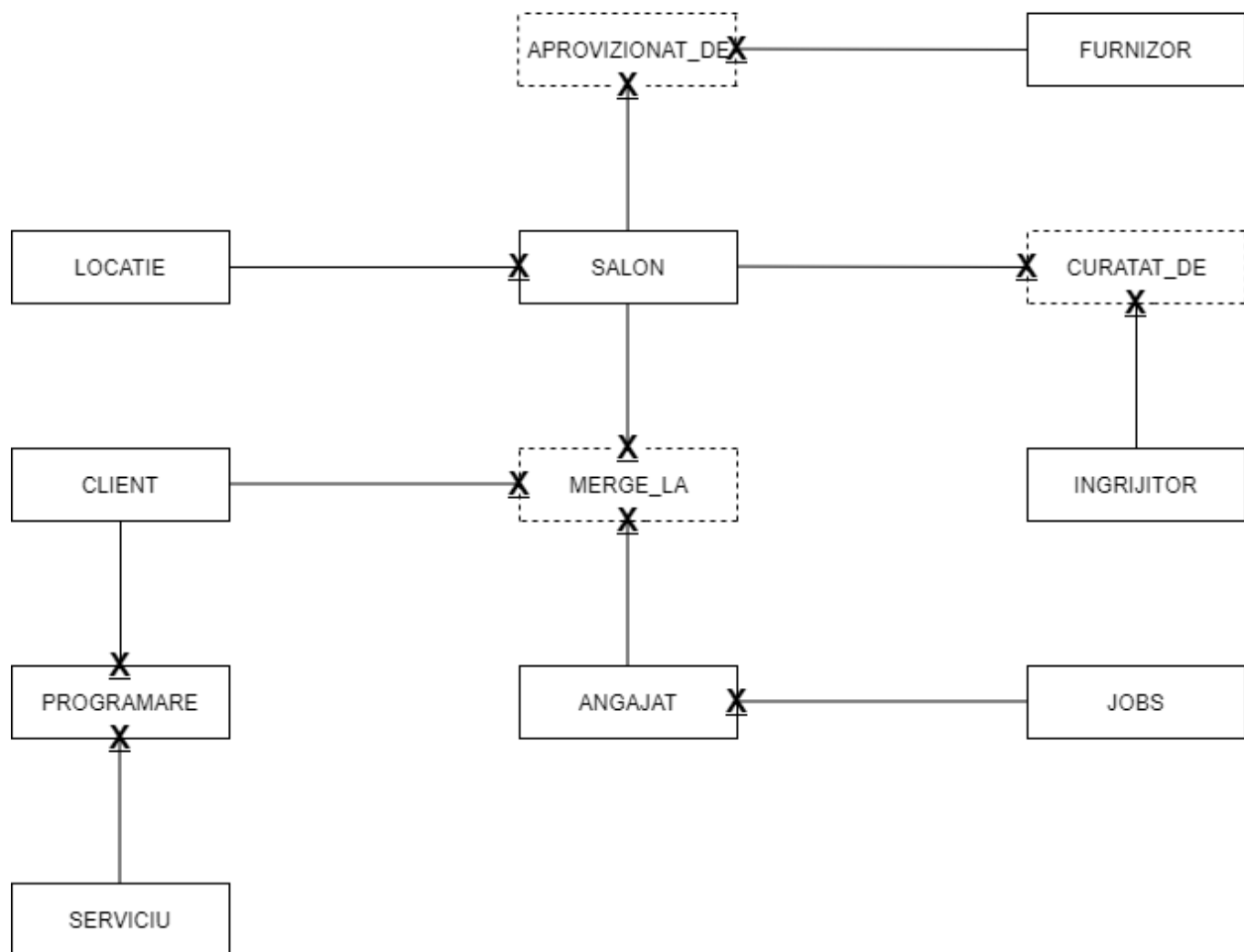


Diagrama conceptuala



### Schemele relaționale

- ✚ SALON (cod\_salon#, nume\_salon, cod\_locatie#);
- ✚ LOCATIE (cod\_locatie#, judet, oras, strada, numar);
- ✚ FURNIZOR (cod\_furnizor#, suma, data\_contract, durata\_contract);
- ✚ INGRIJITOR (cod\_ingrijitor#, nume\_ingrijitor, prenume\_ingrijitor, salariu\_ingrijitor)
- ✚ ANGAJAT (cod\_angajat#, nume\_angajat, prenume\_angajat, data\_angajarii, cod\_salon#, cod\_job#);
- ✚ CLIENT (cod\_client#, nume\_client, prenume\_client, numar\_telefon, adresa\_email);
- ✚ PROGRAMARE (cod\_programare#, data\_programare, cod\_client#, cod\_serviciu#);
- ✚ SERVICIU (cod\_serviciu#, denumire, durata\_serviciu, pret);
- ✚ JOBS (cod\_job#, nume\_job, salariu\_min, salariu\_max);
- ✚ APROVIZIONAT\_DE (cod\_furnizor#, cod\_salon#);
- ✚ CURATAT\_DE (cod\_salon#, cod\_ingrijitor#);
- ✚ MERGE\_LA (cod\_angajat#, cod\_client#, cod\_salon#).

### FORMELE NORMALE (FN1, FN2, FN3)

#### ➤ Forma normală 1 (FN1)

PROGRAMARE (Non-FN1 – exemplu fictiv)

cod_client#	data_programare
200	05-JUL-2021, 07-JUL-2021
900	02-JUL-2021, 05-JUL-2021

Acest exemplu nu este în prima formă normală deoarece valorile din coloana data\_programare sunt divizibile.

#### Transformarea în FN1:

PROGRAMARE (FN1)

cod_client#	cod_programare
200	05-JUL-2021
200	07-JUL-2021
900	02-JUL-2021
900	05-JUL-2021



➤ **Forma normală 2 (FN2)**

CURATAT\_DE (Non FN2 – exemplu fictiv)

cod_ingrijitor#	cod_salon#	ore	salariu
1	2	15	500
1	3	15	600
2	1	9	700
2	3	9	600
3	4	8	400

Acest exemplu fictiv este în FN1 deoarece toate valorile atributelor sunt indivizibile, dar nu este în FN2 deoarece atributul ore depinde doar de cod\_ingrijitor, și nu de întreaga cheie primară.

Astfel avem:

- {cod\_ingrijitor#} => {ore} - cod\_ingrijitor determină funcțional ore
- {cod\_ingrijitor#, cod\_salon#} => {salariu}.

**Transformarea în FN2:**

CURATAT\_2a

cod_ingrijitor	cod_salon	salariu
1	2	500
1	3	600
2	1	700
2	3	600
3	4	400

CURATAT\_2b

cod_ingrijitor	ore
1	15
2	9
3	8

➤ **Forma normală 3 (FN3)**

CURATAT\_DE (Non-FN3 – exemplu fictiv)

cod_ingrijitor#	cod_salon#	ore	salariu
1	1	10	500
1	2	5	250
2	3	10	500
2	4	8	400
3	5	5	250

Relația aceasta este în FN2 deoarece toate atributele depind de întreaga cheie primară, dar nu este în FN3 deoarece nu toate atributele depind direct de cheia primară.

- {cod\_ingrijitor#, cod\_salon#} => {ore} – cod\_ingrijitor și cod\_salon determină functional ore
- {cod\_ingrijitor#, cod\_salon#} => {ore} => {salariu}.

Pentru a aduce relația în FN3 se aplică regula Casey-Delobel. Relația se descompune, prin eliminarea dependențelor funcționale tranzitive, în proiecțiile:

CURATAT\_3a (cod\_ingrijitor#, cod\_salon#, ore)

CURATAT\_3b (ore, salariu).

**Transformarea în FN3:**

CURATAT\_3a

cod_ingrijitor#	cod_salon#	ore
1	1	10
1	2	5
2	3	10
2	4	8
3	5	5

CURATAT\_3b

ore	salariu
5	250
8	400
10	500

## **Creare-inserare**

Aici am pentru fiecare tabel:

- creare tabel,
- creare secventa (pentru primele 3 tabele),
- inserare în tabel,
- printscreen pentru cererea `SELECT * FROM nume_tabel;`

### **1. TABEL LOCATIE**

`CREATE TABLE LOCATIE`

```
(cod_locatie number(6),  
judet varchar2(20) default 'Bucuresti',  
oras varchar2(20) default 'Bucuresti',  
strada varchar2(30) constraint strada_loc not null,  
numar number(3) constraint numar_loc not null,  
constraint pk_locatie primary key(cod_locatie),  
check(numar > 0)  
);
```

`CREATE SEQUENCE SEQ_LOCATIE`

`INCREMENT by 10`

`START WITH 10`

`MINVALUE 0`

`MAXVALUE 50`

`NOCYCLE;`

```
INSERT INTO LOCATIE VALUES (SEQ_LOCATIE.NEXTVAL, 'Cluj', 'Cluj-Napoca', 'Bogdan  
Petriceicu Hasdeu', 82);
```

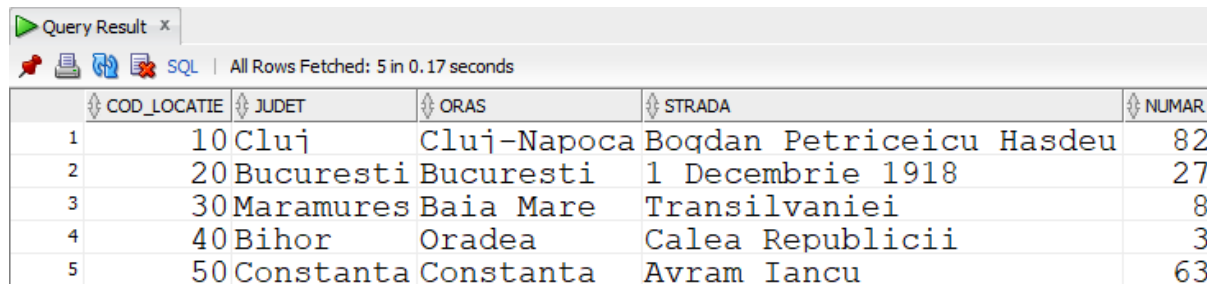
```
INSERT INTO LOCATIE VALUES (SEQ_LOCATIE.NEXTVAL, 'Bucuresti', 'Bucuresti', '1  
Decembrie 1918', 27);
```

```
INSERT INTO LOCATIE VALUES (SEQ_LOCATIE.NEXTVAL, 'Maramures', 'Baia Mare',  
'Transilvaniei', 8);
```

```
INSERT INTO LOCATIE VALUES (SEQ_LOCATIE.NEXTVAL, 'Bihor', 'Oradea', 'Calea  
Republicii', 3);
```

```
INSERT INTO LOCATIE VALUES (SEQ_LOCATIE.NEXTVAL, 'Constanta', 'Constanta',  
'Avram Iancu', 63);
```

```
COMMIT;
```



The screenshot shows a 'Query Result' window with a toolbar and a table of 5 rows. The table has 6 columns: an index, COD\_LOCATIE, JUDET, ORAS, STRADA, and NUMAR. The data is as follows:

	COD_LOCATIE	JUDET	ORAS	STRADA	NUMAR
1	10	Cluj	Cluj-Napoca	Bogdan Petriceicu Hasdeu	82
2	20	Bucuresti	Bucuresti	1 Decembrie 1918	27
3	30	Maramures	Baia Mare	Transilvaniei	8
4	40	Bihor	Oradea	Calea Republicii	3
5	50	Constanta	Constanta	Avram Iancu	63

## 2. TABELUL SALON

```
CREATE TABLE SALON
```

```
(cod_salon number(6),
```

```
nume_salon varchar2(25) constraint nume_sal not null,
```

```
cod_locatie number(6),
```

```
constraint pk_salon primary key(cod_salon),
```

```
constraint fk_salon_locatie foreign key(cod_locatie) references locatie(cod_locatie)
```

```
);
```

```
CREATE SEQUENCE SEQ_SALON  
INCREMENT by 1  
START WITH 0  
MINVALUE 0  
MAXVALUE 5  
NOCYCLE;
```

```
INSERT INTO SALON VALUES (1, 'Rstyle', 30);  
INSERT INTO SALON VALUES (SEQ_SALON.NEXTVAL, 'Black Beauty', 50);  
INSERT INTO SALON VALUES (SEQ_SALON.NEXTVAL, 'Why Not', 20);  
INSERT INTO SALON VALUES (SEQ_SALON.NEXTVAL, 'Bella', 40);  
INSERT INTO SALON VALUES (SEQ_SALON.NEXTVAL, 'Alisa', 10);  
COMMIT;
```



The screenshot shows a 'Query Result' window with a toolbar and a table of 5 rows. The table has three columns: COD\_SALON, NUME\_SALON, and COD\_LOCATIE. The data is as follows:

	COD_SALON	NUME_SALON	COD_LOCATIE
1	2	Black Beauty	50
2	3	Why Not	20
3	4	Bella	40
4	5	Alisa	10
5	1	Rstyle	30

### **3. TABELUL INGRIJITOR**

```
CREATE TABLE INGRIJITOR
```

```
(cod_ingrijitor number(6),  
nume_ingrijitor varchar2(20) constraint nume_in not null,  
prenume_ingrijitor varchar2(20) constraint prenume_in not null,  
salariu_ingrijitor number(6),  
constraint pk_ingrijitor primary key(cod_ingrijitor));
```

```
CREATE SEQUENCE SEQ_INGRIJITOR
```

```
INCREMENT by 1
```

```
START WITH 0
```

```
MINVALUE 0
```

```
MAXVALUE 5
```

```
NOCYCLE;
```

```
INSERT INTO INGRIJITOR VALUES(SEQ_INGRIJITOR.NEXTVAL, 'Constantinescu',  
'Adrian', 2200);
```

```
INSERT INTO INGRIJITOR VALUES(SEQ_INGRIJITOR.NEXTVAL, 'Popescu', 'Radu',  
2340);
```

```
INSERT INTO INGRIJITOR VALUES(SEQ_INGRIJITOR.NEXTVAL, 'Ionescu', 'Paul', 2500);
```

```
INSERT INTO INGRIJITOR VALUES(SEQ_INGRIJITOR.NEXTVAL, 'Paunescu', 'George',  
2105);
```

```
INSERT INTO INGRIJITOR VALUES(SEQ_INGRIJITOR.NEXTVAL, 'Antonescu', 'Ana',  
2550);
```

```
COMMIT;
```

Query Result x				
All Rows Fetched: 5 in 0.014 seconds				
	COD_INGRIJITOR	NUME_INGRIJITOR	PRENUME_INGRIJITOR	SALARIU_INGRIJITOR
1	1	Constantinescu	Adrian	2200
2	2	Popescu	Radu	2340
3	3	Ionescu	Paul	2500
4	4	Paunescu	George	2105
5	5	Antonescu	Ana	2550

#### **4. TABELUL CLIENT**

CREATE TABLE CLIENT

```
(cod_client number(6),  
nume_client varchar(20) constraint nume_cli not null,  
prenume_client varchar(20) constraint prenume_cli not null,  
numar_telefon varchar(13),  
adresa_email varchar2(30),  
constraint pk_client primary key(cod_client)  
);
```

```
INSERT INTO CLIENT VALUES (100, 'Voicu', 'Stefania', '0768486544',  
'stefania.voicu@hotmail.com');
```

```
INSERT INTO CLIENT VALUES (200, 'Sandu', 'Andreea', '0765465145',  
'sandu_andreea@gmail.com');
```

```
INSERT INTO CLIENT VALUES (300, 'Aioanei', 'Lia', '0765321231', 'liavoicu@yahoo.com');
```

```
INSERT INTO CLIENT VALUES (400, 'Trandafir', 'Anton', '0764351235', 'trandafir-  
anton@gmail.com');
```

```
INSERT INTO CLIENT VALUES (500, 'Radulescu', 'Alexandru', '0745216644',  
'alexandru.radulescu@yahoo.com');
```

```
INSERT INTO CLIENT VALUES (600, 'Mihai', 'Ioana', '0745657516',  
'mihai.ioana@hotmail.com');
```

```
INSERT INTO CLIENT VALUES (700, 'Radu', 'Cristina', '0748165297',  
'cristinaradu@gmail.com');
```


```
INSERT INTO CLIENT VALUES (800, 'Panait', 'Ovidiu', '0735482156',  
'panait_ovidiu@yahoo.com');
```

```
INSERT INTO CLIENT VALUES (900, 'Preda', 'Nicu', '0724568154', 'nicu-preda@gmail.com');
```

```
INSERT INTO CLIENT VALUES (1000, 'Oprea', 'Petru', '0756842154',  
'petru.oprea@yahoo.com');
```

```
COMMIT;
```

Query Result x

 All Rows Fetched: 10 in 0.042 seconds

	COD_CLIENT	NUME_CLIENT	PRENUME_CLIENT	NUMAR_TELEFON	ADRESA_EMAIL
1	100	Voicu	Stefania	0768486544	stefania.voicu@hotmail.com
2	200	Sandu	Andreea	0765465145	sandu_andreea@qmail.com
3	300	Aioanei	Lia	0765321231	liavoicu@yahoo.com
4	400	Trandafir	Anton	0764351235	trandafir-anton@qmail.com
5	500	Radulescu	Alexandru	0745216644	alexandru.radulescu@yahoo.com
6	600	Mihai	Ioana	0745657516	mihai.ioana@hotmail.com
7	700	Radu	Cristina	0748165297	cristinaradu@qmail.com
8	800	Panait	Ovidiu	0735482156	panait_ovidiu@yahoo.com
9	900	Preda	Nicu	0724568154	nicu-preda@qmail.com
10	1000	Oprea	Petru	0756842154	petru.oprea@yahoo.com

## 5. TABELUL PROGRAMARE

CREATE TABLE PROGRAMARE

```
(cod_programare number(6),
data_programare date default sysdate,
cod_client number(6),
constraint pk_programare primary key(cod_programare),
constraint fk_programare_client foreign key(cod_client) references client(cod_client)
);
```

ALTER TABLE PROGRAMARE

ADD cod\_serviciu number(6);      --aici am dat alter table add column pentru că am uitat să o pun când am creat tabelul.

ALTER TABLE PROGRAMARE

ADD CONSTRAINT fk\_programare\_serviciu foreign key(cod\_serviciu) references serviciu(cod\_serviciu);      --aici am adăugat constraint pentru coloana adăugată cu alter-ul anterior.

INSERT INTO PROGRAMARE VALUES (1, TO\_DATE('01-07-2021 13:30', 'dd-mm-yyyy hh24:mi'), 300, 6);

INSERT INTO PROGRAMARE VALUES (2, TO\_DATE('10-07-2021 08:30', 'dd-mm-yyyy hh24:mi'), 100, 3);



```
INSERT INTO PROGRAMARE VALUES (3, TO_DATE('05-07-2021 10:30', 'dd-mm-yyyy  
hh24:mi'), 200, 2);
```

```
INSERT INTO PROGRAMARE VALUES (4, TO_DATE('02-07-2021 18:30', 'dd-mm-yyyy  
hh24:mi'), 500, 5);
```

```
INSERT INTO PROGRAMARE VALUES (5, TO_DATE('07-07-2021 12:30', 'dd-mm-yyyy  
hh24:mi'), 400, 1);
```

```
INSERT INTO PROGRAMARE VALUES (6, TO_DATE('01-07-2021 13:30', 'dd-mm-yyyy  
hh24:mi'), 800, 2);
```

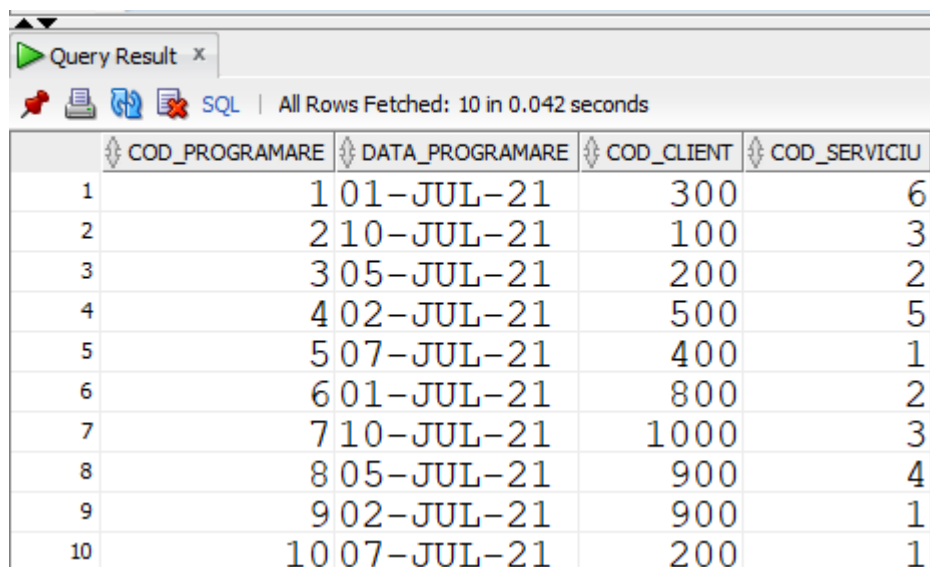
```
INSERT INTO PROGRAMARE VALUES (7, TO_DATE('10-07-2021 08:30', 'dd-mm-yyyy  
hh24:mi'), 1000, 3);
```

```
INSERT INTO PROGRAMARE VALUES (8, TO_DATE('05-07-2021 10:30', 'dd-mm-yyyy  
hh24:mi'), 900, 4);
```

```
INSERT INTO PROGRAMARE VALUES (9, TO_DATE('02-07-2021 18:30', 'dd-mm-yyyy  
hh24:mi'), 900, 1);
```

```
INSERT INTO PROGRAMARE VALUES (10, TO_DATE('07-07-2021 12:30', 'dd-mm-yyyy  
hh24:mi'), 200, 1);
```

```
COMMIT;
```



The screenshot shows a 'Query Result' window with a toolbar and a table of 10 rows. The toolbar includes icons for saving, printing, and refreshing, along with a status bar indicating 'All Rows Fetched: 10 in 0.042 seconds'. The table has four columns: COD\_PROGRAMARE, DATA\_PROGRAMARE, COD\_CLIENT, and COD\_SERVICIU.

	⚙ COD_PROGRAMARE	⚙ DATA_PROGRAMARE	⚙ COD_CLIENT	⚙ COD_SERVICIU
1	1	01-JUL-21	300	6
2	2	10-JUL-21	100	3
3	3	05-JUL-21	200	2
4	4	02-JUL-21	500	5
5	5	07-JUL-21	400	1
6	6	01-JUL-21	800	2
7	7	10-JUL-21	1000	3
8	8	05-JUL-21	900	4
9	9	02-JUL-21	900	1
10	10	07-JUL-21	200	1

**6. TABELUL SERVICIU**

```
CREATE TABLE SERVICIU
```

```
(cod_serviciu number(6),
denumire varchar2(20) constraint denumire_serv not null,
durata_serviciu number(1, 2) default 0,
pret number(3) constraint pret_serv not null,
constraint pk_serviciu primary key(cod_serviciu)
);
```

```
ALTER TABLE SERVICIU
```

```
MODIFY durata_serviciu number (3,2); --modific dimensiunea pentru coloana durata_serviciu,
am introdus-o greșit și am observat după ce am creat tabelul.
```

```
INSERT INTO SERVICIU VALUES (1, 'Tuns', 0.5, 30);
```

```
INSERT INTO SERVICIU VALUES (2, 'Vopsit', 2, 100);
```

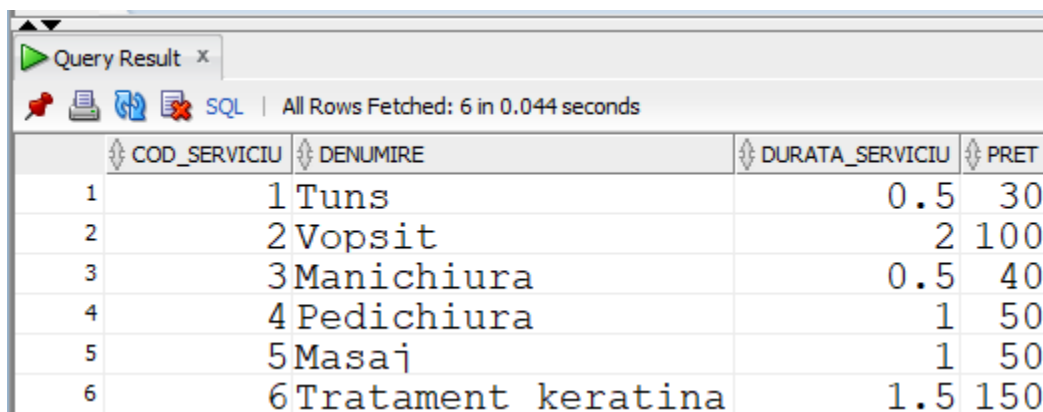
```
INSERT INTO SERVICIU VALUES (3, 'Manichiura', 0.5, 40);
```

```
INSERT INTO SERVICIU VALUES (4, 'Pedichiura', 1, 50);
```

```
INSERT INTO SERVICIU VALUES (5, 'Masaj', 1, 50);
```

```
INSERT INTO SERVICIU VALUES (6, 'Tratament keratina', 1.5, 150);
```

```
COMMIT;
```



Query Result x

SQL | All Rows Fetched: 6 in 0.044 seconds

	COD_SERVICIU	DENUMIRE	DURATA_SERVICIU	PRET
1	1	Tuns	0.5	30
2	2	Vopsit	2	100
3	3	Manichiura	0.5	40
4	4	Pedichiura	1	50
5	5	Masaj	1	50
6	6	Tratament keratina	1.5	150

## 7. TABELUL JOBS

CREATE TABLE JOBS

```
(cod_job number(6),
nume_job varchar2(20) constraint nume_job not null,
salariu_min number(5),
salariu_max number(5),
constraint pk_jobs primary key(cod_job)
);
```

INSERT INTO JOBS VALUES(1, 'Frizer', 2000, 3000);

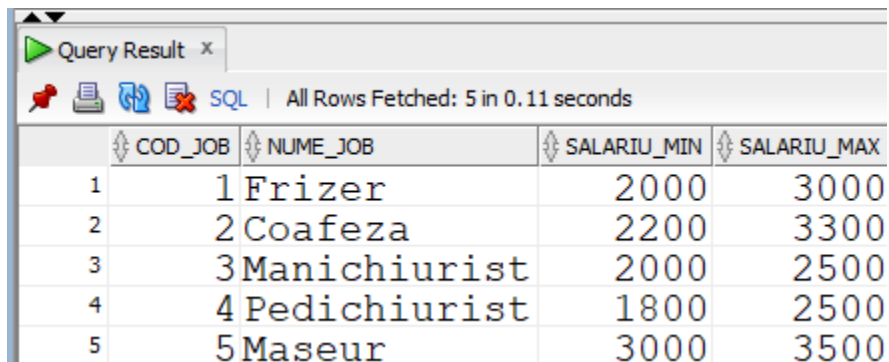
INSERT INTO JOBS VALUES(2, 'Coafeza', 2200, 3300);

INSERT INTO JOBS VALUES(3, 'Manichiurist', 2000, 2500);

INSERT INTO JOBS VALUES(4, 'Pedichiurist', 1800, 2500);

INSERT INTO JOBS VALUES(5, 'Maseur', 3000, 3500);

COMMIT;



The screenshot shows a 'Query Result' window with the following data:

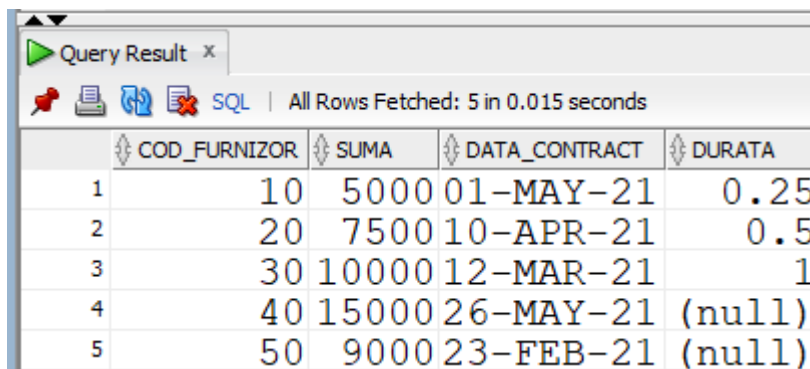
	COD_JOB	NUME_JOB	SALARIU_MIN	SALARIU_MAX
1	1	Frizer	2000	3000
2	2	Coafeza	2200	3300
3	3	Manichiurist	2000	2500
4	4	Pedichiurist	1800	2500
5	5	Maseur	3000	3500

## 8. TABELUL FURNIZOR

CREATE TABLE FURNIZOR

```
(cod_furnizor number(6),
suma number(10) constraint suma_furn not null,
data_contract date default sysdate,
durata number(3,2) default null,
constraint pk_furnizor primary key(cod_furnizor));
```

```
INSERT INTO FURNIZOR VALUES(10, 5000, TO_DATE('01-05-2021', 'dd-mm-yyyy'), 0.25);
INSERT INTO FURNIZOR VALUES(20, 7500, TO_DATE('10-04-2021', 'dd-mm-yyyy'), 0.5);
INSERT INTO FURNIZOR VALUES(30, 10000, TO_DATE('12-03-2021', 'dd-mm-yyyy'), 1);
INSERT INTO FURNIZOR VALUES(40, 15000, SYSDATE, NULL);
INSERT INTO FURNIZOR VALUES(50, 9000, TO_DATE('23-02-2021', 'dd-mm-yyyy'),
NULL);
COMMIT;
```



The screenshot shows a 'Query Result' window with the following data:

	COD_FURNIZOR	SUMA	DATA_CONTRACT	DURATA
1	10	5000	01-MAY-21	0.25
2	20	7500	10-APR-21	0.5
3	30	10000	12-MAR-21	1
4	40	15000	26-MAY-21	(null)
5	50	9000	23-FEB-21	(null)

## 9. TABELUL ANGAJAT

CREATE TABLE ANGAJAT

```
(cod_angajat number(6),
nume_angajat varchar2(20) constraint nume_ang not null,
prenume_angajat varchar2(20) constraint prenume_ang not null,
salariu_angajat number(6),
data_angajarii date default sysdate,
cod_salon number(6),
cod_job number(6),
constraint pk_angajat primary key(cod_angajat),
constraint fk_angajat_salon foreign key(cod_salon) references salon(cod_salon),
constraint fk_angajat_job foreign key(cod_job) references jobs(cod_job)
);
```

INSERT INTO ANGAJAT VALUES (100, 'Georgescu', 'Anastasia', 2115, TO\_DATE('01-05-2018', 'dd-mm-yyyy'), 1, 1);

INSERT INTO ANGAJAT VALUES (101, 'Calinescu', 'Andreea', 2786, TO\_DATE('01-07-2017', 'dd-mm-yyyy'), 1, 2);

INSERT INTO ANGAJAT VALUES (102, 'Voineasa', 'Bianca', 2450, TO\_DATE('03-08-2019', 'dd-mm-yyyy'), 1, 3);

INSERT INTO ANGAJAT VALUES (103, 'Alexandrescu', 'Diana', 1978, TO\_DATE('13-12-2018', 'dd-mm-yyyy'), 1, 4);

INSERT INTO ANGAJAT VALUES (104, 'Anghel', 'Florentina', 3200, TO\_DATE('14-10-2017', 'dd-mm-yyyy'), 1, 5);

INSERT INTO ANGAJAT VALUES (105, 'Andreescu', 'Ana', 2455, TO\_DATE('10-06-2017', 'dd-mm-yyyy'), 2, 1);

INSERT INTO ANGAJAT VALUES (106, 'Pop', 'Dorina', 3126, TO\_DATE('01-07-2016', 'dd-mm-yyyy'), 2, 2);

INSERT INTO ANGAJAT VALUES (107, 'Ion', 'Raluca', 2350, TO\_DATE('13-11-2019', 'dd-mm-yyyy'), 2, 3);

INSERT INTO ANGAJAT VALUES (108, 'Mircescu', 'Maria', 2078, TO\_DATE('12-01-2018', 'dd-mm-yyyy'), 2, 4);

INSERT INTO ANGAJAT VALUES (109, 'Petrescu', 'Horia', 3300, TO\_DATE('13-10-2017', 'dd-mm-yyyy'), 2, 5);

INSERT INTO ANGAJAT VALUES (110, 'Paun', 'Madalina', 2615, TO\_DATE('01-12-2018', 'dd-mm-yyyy'), 3, 1);

INSERT INTO ANGAJAT VALUES (111, 'Dumitru', 'Daria', 2886, TO\_DATE('23-07-2017', 'dd-mm-yyyy'), 3, 2);

INSERT INTO ANGAJAT VALUES (112, 'Dumitriu', 'Antonia', 2150, TO\_DATE('13-10-2019', 'dd-mm-yyyy'), 3, 3);

INSERT INTO ANGAJAT VALUES (113, 'Balan', 'Adriana', 2378, TO\_DATE('12-11-2018', 'dd-mm-yyyy'), 3, 4);

INSERT INTO ANGAJAT VALUES (114, 'Trandafir', 'Toma', 3400, TO\_DATE('12-04-2017', 'dd-mm-yyyy'), 3, 5);

INSERT INTO ANGAJAT VALUES (115, 'Calin', 'Beatrice', 2345, TO\_DATE('01-03-2018', 'dd-mm-yyyy'), 4, 1);

INSERT INTO ANGAJAT VALUES (116, 'Geamanu', 'Laura', 2986, TO\_DATE('14-07-2017', 'dd-mm-yyyy'), 4, 2);

INSERT INTO ANGAJAT VALUES (117, 'Diacon', 'Larisa', 1950, TO\_DATE('03-10-2019', 'dd-mm-yyyy'), 4, 3);

INSERT INTO ANGAJAT VALUES (118, 'Breazu', 'Diana', 1878, TO\_DATE('15-12-2018', 'dd-mm-yyyy'), 4, 4);

INSERT INTO ANGAJAT VALUES (119, 'Savu', 'Andrei', 2800, TO\_DATE('14-12-2017', 'dd-mm-yyyy'), 4, 5);

INSERT INTO ANGAJAT VALUES (120, 'Alecu', 'Alexandra', 2905, TO\_DATE('11-11-2018', 'dd-mm-yyyy'), 5, 1);

INSERT INTO ANGAJAT VALUES (121, 'Florea', 'Delia', 3186, TO\_DATE('18-02-2017', 'dd-mm-yyyy'), 5, 2);

INSERT INTO ANGAJAT VALUES (122, 'Popescu', 'Cornelia', 2220, TO\_DATE('12-06-2019', 'dd-mm-yyyy'), 5, 3);

INSERT INTO ANGAJAT VALUES (123, 'Nae', 'Victoria', 2378, TO\_DATE('04-12-2018', 'dd-mm-yyyy'), 5, 4);

INSERT INTO ANGAJAT VALUES (124, 'Florescu', 'Catalin', 3000, TO\_DATE('25-10-2017', 'dd-mm-yyyy'), 5, 5);

COMMIT;

Query Result x							
All Rows Fetched: 25 in 0.577 seconds							
	COD_ANGAJAT	NUME_ANGAJAT	PRENUME_ANGAJAT	SALARIU_ANGAJAT	DATA_ANGAJ...	COD_SALON	COD_JOB
1	100	Georgescu	Anastasia	2115	01-MAY-18	1	1
2	101	Calinescu	Andreea	2786	01-JUL-17	1	2
3	102	Voineasa	Bianca	2450	03-AUG-19	1	3
4	103	Alexandrescu	Diana	1978	13-DEC-18	1	4
5	104	Anghel	Florentina	3200	14-OCT-17	1	5
6	105	Andreescu	Ana	2455	10-JUN-17	2	1
7	106	Pop	Dorina	3126	01-JUL-16	2	2
8	107	Ion	Raluca	2350	13-NOV-19	2	3
9	108	Mircescu	Maria	2078	12-JAN-18	2	4
10	109	Petrescu	Horia	3300	13-OCT-17	2	5
11	110	Paun	Madalina	2615	01-DEC-18	3	1
12	111	Dumitru	Daria	2886	23-JUL-17	3	2
13	112	Dumitriu	Antonia	2150	13-OCT-19	3	3
14	113	Balan	Adriana	2378	12-NOV-18	3	4
15	114	Trandafir	Toma	3400	12-APR-17	3	5
16	115	Calin	Beatrice	2345	01-MAR-18	4	1
17	116	Geamanu	Laura	2986	14-JUL-17	4	2
18	117	Diacon	Larisa	1950	03-OCT-19	4	3
19	118	Breazu	Diana	1878	15-DEC-18	4	4
20	119	Savu	Andrei	2800	14-DEC-17	4	5
21	120	Alecu	Alexandra	2905	11-NOV-18	5	1
22	121	Florea	Delia	3186	18-FEB-17	5	2
23	122	Popescu	Cornelia	2220	12-JUN-19	5	3
24	123	Nae	Victoria	2378	04-DEC-18	5	4
25	124	Florescu	Catalin	3000	25-OCT-17	5	5

## 10. TABELUL APROVIZIONAT\_DE

CREATE TABLE APROVIZIONAT\_DE

(cod\_furnizor number(6),

cod\_salon number(6),

constraint pk\_aprovizionat\_de primary key(cod\_salon, cod\_furnizor),

constraint fk\_aprovizionat\_de\_salon foreign key (cod\_salon) references salon(cod\_salon),

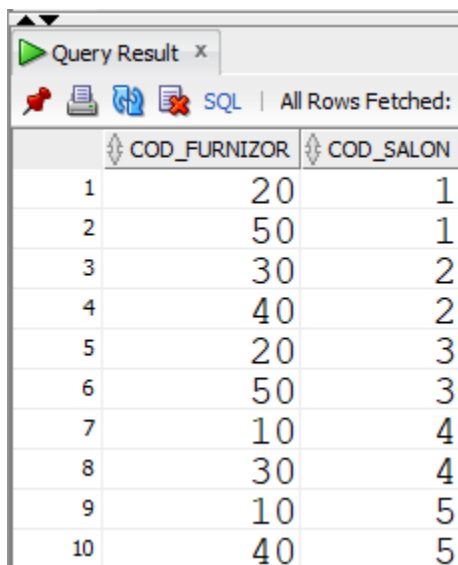
constraint fk\_aprovizionat\_de\_furnizor foreign key (cod\_furnizor) references furnizor(cod\_furnizor)

);

INSERT INTO APROVIZIONAT\_DE VALUES(50,3);

INSERT INTO APROVIZIONAT\_DE VALUES(30,4);

```
INSERT INTO APROVIZIONAT_DE VALUES(20,1);
INSERT INTO APROVIZIONAT_DE VALUES(40,2);
INSERT INTO APROVIZIONAT_DE VALUES(10,5);
INSERT INTO APROVIZIONAT_DE VALUES(20,3);
INSERT INTO APROVIZIONAT_DE VALUES(10,4);
INSERT INTO APROVIZIONAT_DE VALUES(50,1);
INSERT INTO APROVIZIONAT_DE VALUES(30,2);
INSERT INTO APROVIZIONAT_DE VALUES(40,5);
COMMIT;
```



The screenshot shows a 'Query Result' window with a table containing 10 rows. The columns are labeled 'COD\_FURNIZOR' and 'COD\_SALON'. The data is as follows:

	COD_FURNIZOR	COD_SALON
1	20	1
2	50	1
3	30	2
4	40	2
5	20	3
6	50	3
7	10	4
8	30	4
9	10	5
10	40	5

## **11. TABELUL CURATAT\_DE**

```
CREATE TABLE CURATAT_DE
```

```
(cod_ingrijitor number(6),
```

```
cod_salon number(6),
```

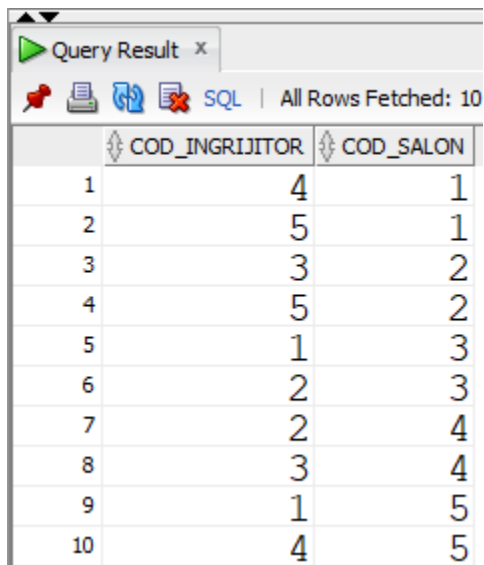
```
constraint pk_curatat_de primary key(cod_salon, cod_ingrijitor),
```

```
constraint fk_curatat_de_salon foreign key (cod_salon) references salon(cod_salon),
```

```
constraint fk_curatat_de_ingrijitor foreign key (cod_ingrijitor) references  
ingrijitor(cod_ingrijitor) );
```



```
INSERT INTO CURATAT_DE VALUES (1, 3);
INSERT INTO CURATAT_DE VALUES (2, 4);
INSERT INTO CURATAT_DE VALUES (3, 2);
INSERT INTO CURATAT_DE VALUES (4, 5);
INSERT INTO CURATAT_DE VALUES (5, 1);
INSERT INTO CURATAT_DE VALUES (1, 5);
INSERT INTO CURATAT_DE VALUES (2, 3);
INSERT INTO CURATAT_DE VALUES (3, 4);
INSERT INTO CURATAT_DE VALUES (4, 1);
INSERT INTO CURATAT_DE VALUES (5, 2);
COMMIT;
```



The screenshot shows a 'Query Result' window with a toolbar and a table of 10 rows. The table has two columns: 'COD\_INGRIJITOR' and 'COD\_SALON'. The data is as follows:

	COD_INGRIJITOR	COD_SALON
1	4	1
2	5	1
3	3	2
4	5	2
5	1	3
6	2	3
7	2	4
8	3	4
9	1	5
10	4	5

## **12. TABELUL MERGE\_LA**

CREATE TABLE MERGE\_LA

(cod\_angajat number(6),

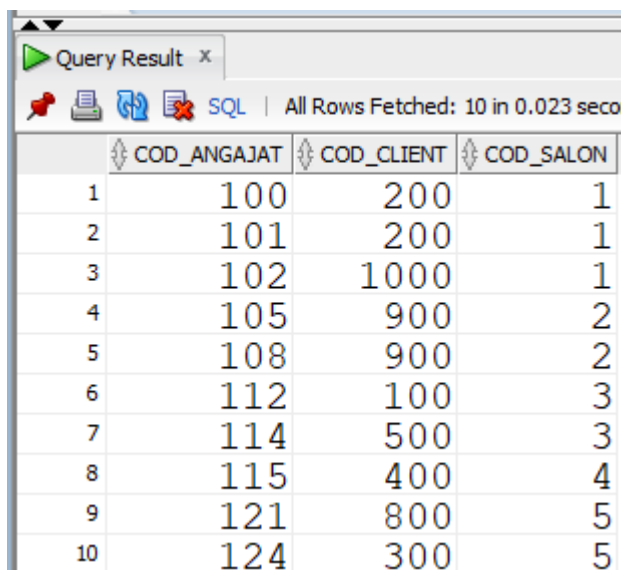
cod\_client number(6),

cod\_salon number(6),

constraint pk\_merge\_la primary key(cod\_angajat, cod\_client, cod\_salon),

```
constraint fk_merge_la_angajat foreign key (cod_angajat) references angajat(cod_angajat),  
constraint fk_merge_la_client foreign key (cod_client) references client(cod_client),  
constraint fk_merge_la_salon foreign key (cod_salon) references salon(cod_salon)  
);
```

```
INSERT INTO MERGE_LA VALUES (124, 300, 5);  
INSERT INTO MERGE_LA VALUES (112, 100, 3);  
INSERT INTO MERGE_LA VALUES (101, 200, 1);  
INSERT INTO MERGE_LA VALUES (114, 500, 3);  
INSERT INTO MERGE_LA VALUES (115 , 400, 4);  
INSERT INTO MERGE_LA VALUES (121, 800, 5);  
INSERT INTO MERGE_LA VALUES (102, 1000, 1);  
INSERT INTO MERGE_LA VALUES (108, 900, 2);  
INSERT INTO MERGE_LA VALUES (105, 900, 2);  
INSERT INTO MERGE_LA VALUES (100, 200, 1);  
COMMIT;
```



The screenshot shows a 'Query Result' window with a toolbar and a table of results. The toolbar includes icons for saving, refreshing, and other database actions, along with the text 'All Rows Fetched: 10 in 0.023 seco'. The table has three columns: COD\_ANGAJAT, COD\_CLIENT, and COD\_SALON. It contains 10 rows of data, numbered 1 to 10 in the first column.

	COD_ANGAJAT	COD_CLIENT	COD_SALON
1	100	200	1
2	101	200	1
3	102	1000	1
4	105	900	2
5	108	900	2
6	112	100	3
7	114	500	3
8	115	400	4
9	121	800	5
10	124	300	5

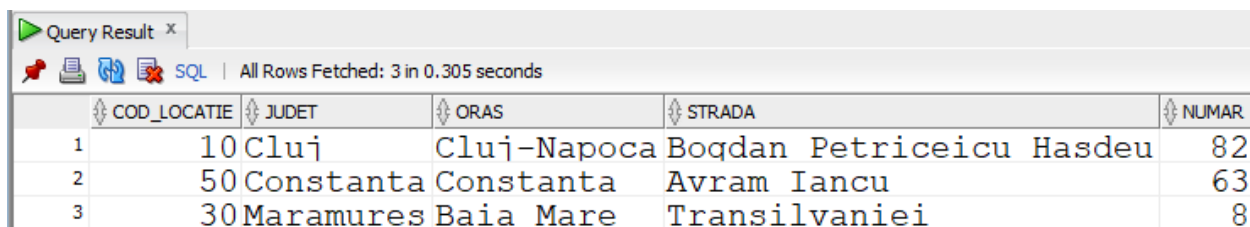
### Cereri SQL – cerința 11

1. Sa se afiseze toate detaliile locațiilor saloanelor din afara Bucureștiului pentru care furnizorii au întocmit contracte pe perioadă nedeterminată, ordonați după județ (durata = null).

Pentru această cerere am utilizat:

- Join pe 4 tabele
- Funcția pe șiruri de caractere upper()
- Filtrare la nivel de linii – clauza WHERE
- Ordonare – clauza ORDER BY.

```
SELECT l.cod_locatie, l.judet, l.oras, l.strada, l.numar
FROM furnizor f JOIN aprovizionat_de a ON (f.cod_furnizor = a.cod_furnizor)
      JOIN salon s ON (s.cod_salon = a.cod_salon)
      JOIN locatie l ON (l.cod_locatie = s.cod_locatie)
WHERE f.durata is null and upper(l.judet)!='BUCURESTI'
ORDER BY l.judet;
```



Query Result x

SQL | All Rows Fetched: 3 in 0.305 seconds

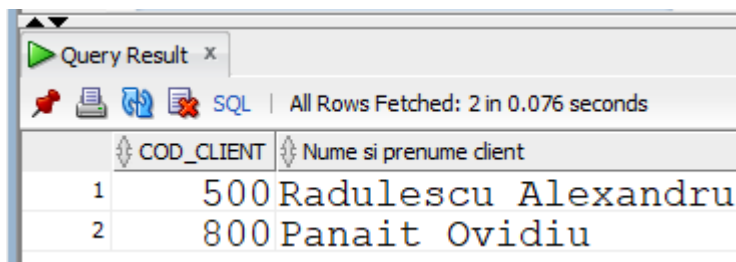
	COD_LOCATIE	JUDET	ORAS	STRADA	NUMAR
1	10	Cluj	Cluj-Napoca	Boqdan Petriceicu Hasdeu	82
2	50	Constanta	Constanta	Avram Iancu	63
3	30	Maramures	Baia Mare	Transilvaniei	8

2. Sa se afiseze clientii care merg la cei mai bine platiti angajati de pe job-ul lor.

Pentru această cerere am utilizat:

- Subcereri nesincronizate pe 3 tabele
- Funcția pe șiruri de caractere concat()
- Group by, having – grupări de date și funcția max() pe grupări de date, dar și filtrare la nivel de grupare de date cu ajutorul clauzei HAVING.

```
SELECT cod_client, concat(ume_client, ' ') || prenume_client "Nume si prenume client"
FROM client c
WHERE c.cod_client IN
    (SELECT m.cod_client
    FROM merge_la m
    WHERE m.cod_angajat IN
        (SELECT a.cod_angajat
        FROM angajat a
        WHERE a.salariu_angajat IN
            (SELECT max(b.salariu_angajat)
            FROM angajat b
            GROUP BY b.cod_job
            HAVING max(b.salariu_angajat) > 3000 )));
```



The screenshot shows a 'Query Result' window with two columns: 'COD\_CLIENT' and 'Nume si prenume client'. It displays two rows of data.

	COD_CLIENT	Nume si prenume client
1	500	Radulescu Alexandru
2	800	Panait Ovidiu

3. Să se afișeze prețul total plătit de fiecare client care are programări în prima săptămână din iulie (01-JUL-2021 - 07-JUL-2021).

Pentru această cerere am utilizat:

- Clauza WITH
- Funcțiile pe date calendaristice add\_months() și next\_day()
- Group by

WITH pret\_total AS

(SELECT nume\_client, prenume\_client, sum(pret) as total\_client

FROM client c JOIN programare p ON (c.cod\_client = p.cod\_client)

JOIN serviciu s ON (s.cod\_serviciu = p.cod\_serviciu)

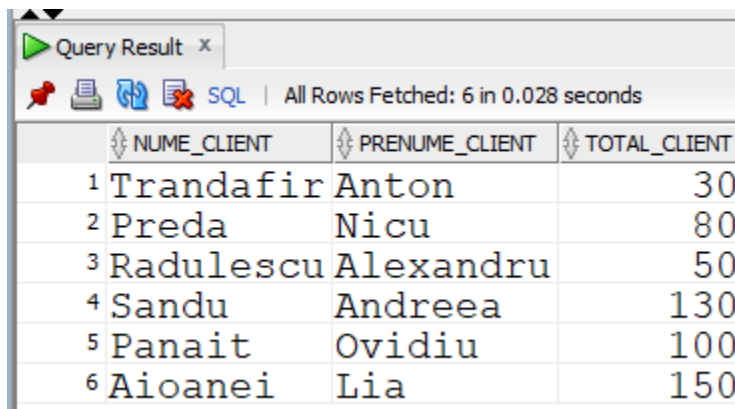
WHERE data\_programare between add\_months('01-JUN-2021', 1) and

next\_day(add\_months('01-JUN-2021', 1), to\_char(add\_months('01-JUN-2021', 1), 'day'))

GROUP BY nume\_client, prenume\_client)

SELECT \*

FROM pret\_total;



Query Result x

All Rows Fetched: 6 in 0.028 seconds

	NUME_CLIENT	PRENUME_CLIENT	TOTAL_CLIENT
1	Trandafir	Anton	30
2	Preda	Nicu	80
3	Radulescu	Alexandru	50
4	Sandu	Andreea	130
5	Panait	Ovidiu	100
6	Aioanei	Lia	150

4. Sa se afiseze cel mai mare salariu din fiecare salon.

Pentru această cerere am utilizat:

- Subcereri sincronizate in clauza SELECT pe 3 tabele
- Ordonare – ORDER BY

SELECT cod\_salon, nume\_salon,

(SELECT MAX(salariu\_angajat)

FROM angajat

WHERE s.cod\_salon = cod\_salon) "Cel mai mare salariu",

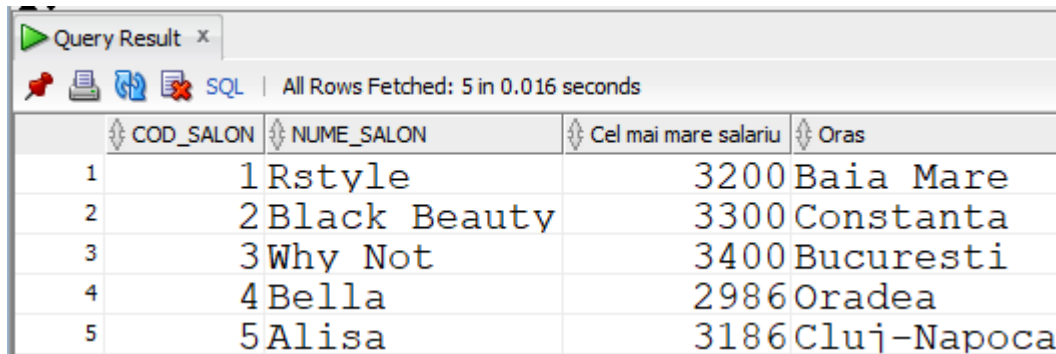
(SELECT oras

FROM locatie

WHERE cod\_locatie = s.cod\_locatie) "Oras"

FROM salon s

ORDER BY cod\_salon;



	COD_SALON	NUME_SALON	Cel mai mare salariu	Oras
1		Rstyle	3200	Baia Mare
2		Black Beauty	3300	Constanta
3		Why Not	3400	Bucuresti
4		Bella	2986	Oradea
5		Alisa	3186	Cluj-Napoca

5. Sa se afiseze numele, prenumele si data angajarii fiecarui angajat, precum si salariul actual, salariul marit daca lucreaza din 2016 sau din 2017 (20%, respectiv 10%), salariul micșorat daca lucreaza din 2018 sau din 2019 (5%, respectiv 10%).

Pentru această cerere am utilizat:

- DECODE și CASE
- Funcția NVL

SELECT nume\_angajat, prenume\_angajat, data\_angajarii, salariu\_angajat,

NVL((DECODE (TO\_CHAR(data\_angajarii, 'yyyy'), '2016', salariu\_angajat \* 1.2, '2017', salariu\_angajat\* 1.1)), salariu\_angajat) "Salariu marit",

CASE TO\_CHAR(data\_angajarii, 'yyyy')

WHEN '2018' THEN salariu\_angajat \* 0.95





WHEN '2019' THEN salariu\_angajat \* 0.9

ELSE salariu\_angajat

END "Salariu micșorat"

FROM angajat;

Query Result x

    SQL | All Rows Fetched: 25 in 0.149 seconds

	NUME_ANGAJAT	PRENUME_ANGAJAT	DATA_ANGAJARII	SALARIU_ANGAJAT	Salariu marit	Salariu microrat
1	Georgescu	Anastasia	01-MAY-18	2115	2115	2009.25
2	Calinescu	Andreea	01-JUL-17	2786	3064.6	2786
3	Voineasa	Bianca	03-AUG-19	2450	2450	2205
4	Alexandrescu	Diana	13-DEC-18	1978	1978	1879.1
5	Anghel	Florentina	14-OCT-17	3200	3520	3200
6	Andreescu	Ana	10-JUN-17	2455	2700.5	2455
7	Pop	Dorina	01-JUL-16	3126	3751.2	3126
8	Ion	Raluca	13-NOV-19	2350	2350	2115
9	Mircescu	Maria	12-JAN-18	2078	2078	1974.1
10	Petrescu	Horia	13-OCT-17	3300	3630	3300
11	Paun	Madalina	01-DEC-18	2615	2615	2484.25
12	Dumitru	Daria	23-JUL-17	2886	3174.6	2886
13	Dumitriu	Antonia	13-OCT-19	2150	2150	1935
14	Balan	Adriana	12-NOV-18	2378	2378	2259.1
15	Trandafir	Toma	12-APR-17	3400	3740	3400
16	Calin	Beatrice	01-MAR-18	2345	2345	2227.75
17	Geamanu	Laura	14-JUL-17	2986	3284.6	2986
18	Diacon	Larisa	03-OCT-19	1950	1950	1755
19	Breazu	Diana	15-DEC-18	1878	1878	1784.1
20	Savu	Andrei	14-DEC-17	2800	3080	2800
21	Alecu	Alexandra	11-NOV-18	2905	2905	2759.75
22	Florea	Delia	18-FEB-17	3186	3504.6	3186
23	Popescu	Cornelia	12-JUN-19	2220	2220	1998
24	Nae	Victoria	04-DEC-18	2378	2378	2259.1
25	Florescu	Catalin	25-OCT-17	3000	3300	3000

### Operații de actualizare/suprimare a datelor

**Am dat rollback după fiecare update/delete.**

1. Actualizarea salariilor angajatilor cu salariul mediu daca salariul lor este mai mic de salariul mediu.

UPDATE ANGAJAT

SET (salariu\_angajat) = (SELECT AVG(salariu\_angajat)

FROM angajat)

WHERE salariu\_angajat < (SELECT AVG(salariu\_angajat)

FROM angajat);

ROLLBACK;

	COD_ANGAJAT	NUME_ANGAJAT	PRENUME_ANGAJAT	SALARIU_ANGA...	DATA_ANGAJARII	COD_SALON	COD_JOB
1	100	Georgescu	Anastasia	2597	01-MAY-18	1	1
2	112	Dumitriu	Antonia	2597	13-OCT-19	3	3
3	123	Nae	Victoria	2597	04-DEC-18	5	4
4	122	Popescu	Cornelia	2597	12-JUN-19	5	3
5	118	Breazu	Diana	2597	15-DEC-18	4	4
6	117	Diacon	Larisa	2597	03-OCT-19	4	3
7	115	Calin	Beatrice	2597	01-MAR-18	4	1
8	113	Balan	Adriana	2597	12-NOV-18	3	4
9	108	Mircescu	Maria	2597	12-JAN-18	2	4
10	107	Ion	Raluca	2597	13-NOV-19	2	3
11	105	Andreescu	Ana	2597	10-JUN-17	2	1
12	103	Alexandrescu	Diana	2597	13-DEC-18	1	4
13	102	Voineasa	Bianca	2597	03-AUG-19	1	3
14	110	Paun	Madalina	2615	01-DEC-18	3	1
15	101	Calinescu	Andreea	2786	01-JUL-17	1	2
16	119	Savu	Andrei	2800	14-DEC-17	4	5
17	111	Dumitru	Daria	2886	23-JUL-17	3	2
18	120	Alecu	Alexandra	2905	11-NOV-18	5	1
19	116	Geamanu	Laura	2986	14-JUL-17	4	2
20	124	Florescu	Catalin	3000	25-OCT-17	5	5
21	106	Pop	Dorina	3126	01-JUL-16	2	2
22	121	Florea	Delia	3186	18-FEB-17	5	2
23	104	Anghel	Florentina	3200	14-OCT-17	1	5
24	109	Petrescu	Horia	3300	13-OCT-17	2	5
25	114	Trandafir	Toma	3400	12-APR-17	3	5



2. Sa se stearga intrarile din aprovizionat\_de unde furnizorul are contract cu salonul pe perioada nedeterminata - durata = null.

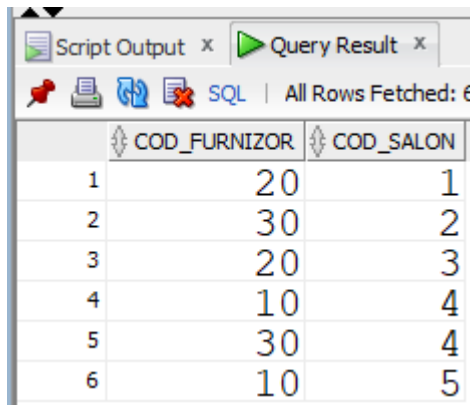
DELETE FROM APROVIZIONAT\_DE

WHERE cod\_furnizor IN (SELECT f.cod\_furnizor

FROM furnizor f

WHERE cod\_furnizor = f.cod\_furnizor and f.durata is null);

ROLLBACK;



	COD_FURNIZOR	COD_SALON
1	20	1
2	30	2
3	20	3
4	10	4
5	30	4
6	10	5

--la crearea tabelului am inserat 10 intrări, după ștergere au rămas 6.

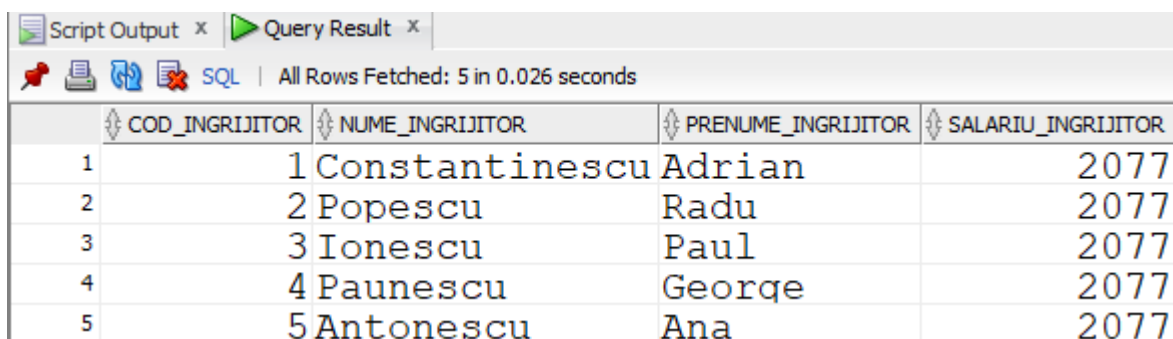
3. Sa se actualizeze tabelul ingrijitor astfel incat fiecare ingrijitor are 80% din media salariilor angajatilor.

UPDATE INGRIJITOR

SET salariu\_ingrijitor =(SELECT 0.8 \* AVG(salariu\_angajat)

FROM angajat);

ROLLBACK;



	COD_INGRIJITOR	NUME_INGRIJITOR	PRENUME_INGRIJITOR	SALARIU_INGRIJITOR
1	1	Constantinescu	Adrian	2077
2	2	Popescu	Radu	2077
3	3	Ionescu	Paul	2077
4	4	Paunescu	George	2077
5	5	Antonescu	Ana	2077

### **Algebră relațională**

#### **Cerință:**

Sa se afișeze toate detaliile locațiilor saloanelor din afara Bucureștiului pentru care furnizorii au întocmit contracte pe perioadă nedeterminată (durata = null).

#### **Cererea SQL:**

```
SELECT l.cod_locatie, l.judet, l.oras, l.strada, l.numar
FROM furnizor f JOIN aprovizionat_de a ON (f.cod_furnizor = a.cod_furnizor)
      JOIN salon s ON (s.cod_salon = a.cod_salon)
      JOIN locatie l ON (l.cod_locatie = s.cod_locatie)
WHERE f.durata is null and upper(l.judet)!='BUCURESTI';
```

#### **Expresia algebrică:**

```
R1 = SELECT(FURNIZOR, durata is null);
R2 = PROJECT(R1, cod_furnizor);
R3 = SEMIJOIN(R2, APROVIZIONAT_DE, cod_furnizor);
R4 = PROJECT(SALON, cod_salon);
R5 = SEMIJOIN(R3, R4, cod_salon);
R6 = SELECT(LOCATIE, upper(judet)!='BUCURESTI');
R7 = SEMIJOIN(R5, R6, cod_locatie);
REZULTAT = R8 = PROJECT(R7, cod_locatie, judet, oras, strada, numar).
```

Cererea este deja optimă:

- SELECT-urile se fac cât mai devreme pentru a înlătura intrările care nu respectă condițiile.
- PROJECT-urile se fac cât mai devreme pentru înlăturarea coloanelor nefolositoare.
- SEMIJOIN-urile se fac pe cât mai puține date posibil.

Arbore algebric:

