

## **PROIECT BAZE DE DATE**

Boldesco Roland  
Grupa 143  
Informatica, Anul I, Sem II

## Cuprins

|  |          |
|--|----------|
| <b>Sistem de gestiune a spitalelor.....</b>  | <b>3</b> |
| 1. Descrierea modelului real, a utilității acestuia și a regulilor de funcționare.....   | 3        |
| 2. Prezentarea constrângerilor (restrictii, reguli) impuse asupra modelului.....   | 4        |
| 3. Descrierea entităților, incluzând precizarea cheii primare.....   | 5        |
| 4. Descrierea relațiilor, incluzând precizarea cardinalității acestora.....  | 5        |
| 5. Descrierea atributelor, incluzând tipul de date și eventualele constrângeri, valori implicate, valori posibile ale atributelor.....   | 6        |
| 6. Realizarea diagramei entitate-relație corespunzătoare descrierii de la punctele 3-5.....  | 9        |
| 7. Realizarea diagramei conceptuale corespunzătoare diagramei entitate-relație proiectate la punctul 6.....  | 9        |
| 8. Enumerarea schemelor relaționale corespunzătoare diagramei conceptuale proiectate la punctul 7.....   | 9        |
| 9. Realizarea normalizării pană la forma normală 3 (FN1 - FN3).....  | 10       |
| 10. Crearea unei secvențe ce va fi utilizată în inserarea înregistrărilor în tabele (punctul 11).....  | 10       |
| 11. Crearea tabelelor în SQL și inserarea de date coerente în fiecare dintre acestea.....  | 11       |
| 12. Formulați în limbaj natural și implementați 5 cereri SQL complexe.....   | 23       |
| 13. Implementarea a 3 operații de actualizare și de suprimare a datelor utilizând subcereri.....   | 28       |
| 14. Crearea unei vizualizări complexe. Dați un exemplu de operație LMD permisă pe vizualizarea respectivă și un exemplu de operație LMD nepermisă.....   | 30       |
| 15. Formulați în limbaj natural și implementați în SQL: o cerere ce utilizează operația outer-join pe minimum 4 tabele, o cerere ce utilizează operația division și o cerere care implementează analiza top-n..... | 32       |
| 16. La alegere.....  | 35       |
| 17. a. Realizarea normalizării BCNF, FN4, FN5.....   | 39       |
| b. Aplicarea denormalizării, justificând necesitatea acesteia.....   | 41       |
| 18. Exemplificarea isolation levels.....   | 43       |
| 19. Justificarea necesității/utilității migrării la o bază de date de tip NoSql.....   | 52       |

## Sistem de gestiune a spitalelor

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

Sistemul de gestiune a spitalelor este conceput pentru a administra informațiile referitoare la pacienți, doctori, programări, tratamente și resurse medicale. Acest sistem optimizează fluxul de lucru prin digitalizarea proceselor de programare, diagnosticare, internare, administrare a tratamentelor și facturare. Prin intermediul acestui sistem, spitalele își pot îmbunătăți eficiența operațională, asigurând o gestionare mai bună a resurselor și oferind pacienților servicii medicale rapide și de calitate.

Spitalul este organizat în mai multe departamente, fiecare având specializări diferite, cum ar fi cardiologie, ortopedie, neurologie, pediatrie, ginecologie, oncologie, radiologie și terapie intensivă. Fiecare departament este dotat cu echipamente specifice și are doctori și asistenți medicali special antrenați pentru a răspunde nevoilor particulare ale pacienților. De asemenea, există unități de imagistică medicală, analize de laborator și farmacii interne pentru a sprijini diagnosticarea și tratamentele aplicate.

Pacienții pot ajunge la spital în două moduri principale:

1. Programare online sau telefonică – Pacienții pot face programări pentru consultații de rutină, analize medicale, investigații imagistice (CT, RMN, ecografii) sau pentru controale periodice post-tratament. Sistemul gestionează programările și distribuie pacienții în funcție de disponibilitatea medicilor și a resurselor spitalului.
2. Cazuri de urgență – În situațiile critice, pacienții sunt aduși direct la Camera de Urgențe fie pe cont propriu, fie cu ambulante echipate corespunzător și însoțiti de paramedici. La sosire, pacienții sunt evaluați de medici pentru a determina gravitatea situației și direcționați fie către internare, fie către tratament ambulatoriu.

În funcție de diagnostic și gravitatea afecțiunii, pacienții pot fi admisi în spital și internați pentru tratamente mai complexe sau pot primi îngrijiri medicale și tratamente în regim ambulatoriu. Pacienții internați beneficiază de servicii complete, inclusiv investigații medicale amănunte, administrare de tratamente medicamentoase, intervenții chirurgicale, servicii de fizioterapie sau consiliere psihologică.

Fiecare pacient internat este repartizat într-o anumită secție a spitalului în funcție de afecțiunea sa și de necesitățile de tratament. Pacienții pot ocupa paturi în saloane comune sau pot opta pentru camere private cu facilități suplimentare, cum ar fi servicii de catering personalizat, asistență medicală dedicată sau acces la tehnologii avansate de diagnostic.

Planul de tratament al fiecărui pacient este elaborat de medicul curant, iar administrarea acestuia este supravegheată de echipa de asistenți medicali. În unele cazuri, pacienții pot necesita intervenții chirurgicale sau alte proceduri medicale avansate, cum ar fi tratament cu radioterapie, chimioterapie, dializă sau fizioterapie.

Pe lângă tratamente, pacienții pot beneficia de servicii auxiliare, cum ar fi:

- Nutriție personalizată – Diete special concepute în funcție de nevoile fiecărui pacient.
- Terapie intensivă și postoperatorie – Pentru pacienții în stare critică sau recuperare post-operatorie.
- Servicii de reabilitare – Recuperare fizică și psihologică după traume sau intervenții chirurgicale.

- Suport psihologic – Consiliere psihologică pentru pacienții care se confruntă cu boli cronice sau traume severe.

Pacienții pot fi externați atunci când medicul curant consideră că starea lor de sănătate le permite plecarea și nu mai este necesară spitalizarea. Înainte de externare, se face o facturare detaliată a costurilor aferente tratamentului primit. Aceste costuri pot include:

- Durata internării și tipul de salon ales (comun sau privat);
- Tratamentele și intervențiile medicale efectuate;
- Medicamentele administrate;
- Analizele de laborator și investigațiile imagistice efectuate;
- Serviciile de ambulanță sau transport medical.

Plata serviciilor se poate face prin asigurare medicală privată sau publică, în funcție de poliță deținută de pacient, sau direct de către pacient, dacă nu beneficiază de asigurare. De asemenea, pacienții pot primi rețete medicale pentru tratamente post-spitalizare și recomandări pentru controale ulterioare.

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

Un doctor este asociat unui singur departament.

Un doctor poate interacționa cu unul sau mai mulți pacienți.

Un doctor poate administra zero sau mai multe tratamente, diferite sau nu, pentru fiecare pacient în grija sa.

Un departament poate avea zero sau mai mulți pacienți internați la un moment dat.

Intr-un departament lucrează unul sau mai mulți doctori.

Un departament conține mai multe saloane.

Un departament conține mai multe echipamente medicale.

Un pacient poate avea o singură programare la un moment dat.

Un pacient poate fi admis la unul sau mai multe departamente ale spitalului.

Un pacient primește cete de factură pentru fiecare internare.

Un pacient poate primi unul sau mai multe tratamente.

Un pacient poate fi asociat cu unul sau mai mulți doctori.

Un pacient este cazat într-un salon, privat sau comun.

Un tratament poate fi administrat pentru zero sau mai mulți pacienți.

Un tratament poate fi administrat de unul sau mai mulți doctori.

Un tratament conține unul sau mai multe medicamente.

Un tratament conține zero sau mai multe echipamente medicale.

O programare este unică, referită prinț-ului id, asociată unui singur pacient.

O programare face referire unui salon.

O factură este unică, referită prinț-ului id, atribuită unui singur pacient.

Un medicament poate fi continut de zero sau mai multe tratamente.

Un salon este atribuit unui singur departament.

Intr-un salon pot fi cazati zero sau mai mulți pacienți.

Un salon poate fi atribuit mai multor programări.

Un echipament medical este atribuit unui sau mai multor departamente.

Un echipament medical poate fi folosit în diferite tratamente.

3. Descrierea entităților, incluzând precizarea cheii primare.

| ENTITATE           | CHEIE PRIMARA   | OBSERVATII  |
|--------------------|-----------------|---|
| Doctor             | ID_Doctor       | Date de angajare, departament, pacienti, tratamente, calificare                               |
| Departament        | ID_Departament  | Contine doctorii, pacientii   |
| Pacient            | CNP, ID_Pacient | Date personale si istoricul medical al pacientului  |
| Programare         | ID_Programare   | Data si timpul programarii, specificarea pacientului  |
| Tratament          | ID_Tratament    | Medicamente si cantitati  |
| Medicament         | ID_Medicament   | Ingrediente   |
| Factura            | ID_Factura      | Personalizata pe cheltuiala pacientului   |
| Salon              | ID_Salon        | Cazarea pacientilor, in functie de situatie si preferinte                                     |
| Echipament Medical | ID_Echipament_m | Echipament special creat pentru facilitarea diferitelor nevoi si sporirea lucrului doctorilor |

4. Descrierea relațiilor, incluzând precizarea cardinalității acestora.

| RELATIE         | CARDINALITATE                                      | OBSERVATII  |
|-----------------|--|---|
| apartine        | departament - doctor<br>one - to - many            | Un doctor apartine unui singur departament.<br>Un departament are mai multi doctori.  |
|                 | departament - salon<br>one - to - many             | Un salon apartine unui singur departament.<br>Un departament are mai multe saloane.   |
| interactioneaza | doctor - pacient<br>many - to - many               | Un doctor interactioneaza cu mai multi pacienti.<br>Un pacient poate interaciona cu mai multi doctori.                        |
| administreaza   | doctor - tratament<br>many - to - many             | Un doctor administreaza mai multe tratamente, mai multor pacienti.<br>Un tratament poate fi administrat de mai multi doctori. |
|                 | tratament - medicament<br>many - to - many         | Un tratament poate contine mai multe medicamente.<br>Un medicament poate fi continut de mai multe tratamente.                 |
|                 | tratament - echipament medical<br>many - to - many | Un tratament poate contine mai multe echipamente medicale.<br>Un echipament medical este regasit in mai multe tratamente.     |
| contine         | salon - programare<br>one - to - many              | O programare contine un salon.<br>Un salon poate fi continut in mai multe programari.   |
|                 | face   | Un pacient face programare, semnalizata cu un id unic.  |

|          |   |  |
|----------|---|--|
|          | one - to - many   | O programare este unica, asociata unui singur pacient.   |
| primește | pacient - factura<br>one - to - many                    | Un pacient primește o factură, semnalizată cu un id unic.<br>O factură revine unui singur pacient.                           |
|          | pacient - tratament<br>many - to - many                 | Un pacient poate primi mai multe tratamente simultan.<br>Un tratament poate fi primit de mai mulți pacienți diferiți.        |
| atribuit | echipament medical -<br>departament<br>many - to - many | Un echipament medical este atribuit mai multor departamente.<br>Un departament poate conține mai multe echipamente medicale. |

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

#### ENTITATE - DOCTOR

| Atribut        | Tip     | Dimensiune / precizie | Val. posibile / default | Observatii  |
|----------------|---------|-----------------------|-------------------------|---|
| ID_Doctor      | integer |                       | [1,3000]                | PK  |
| ID_Departament | integer |                       | [1,20]                  | FK  |
| Calificare     | string  | 20                    |                         | Poate fi sef de secție, doctor asistent, rezident, etc. |
| Nume           | string  | 20                    |                         |   |
| Prenume        | string  | 20                    |                         |   |
| Telefon        | string  | 11                    |                         |   |
| Mail           | string  | 30                    |                         |   |

#### ENTITATE - PACIENT

| Atribut    | Tip     | Dimensiune / precizie | Val. posibile / default | Observatii |
|------------|---------|-----------------------|-------------------------|------------|
| ID_Pacient | integer |                       | [1,1000000]             | PK         |
| CNP        | string  | 16                    |                         | unique     |
| Nume       | string  | 20                    |                         |            |
| Prenume    | string  | 20                    |                         |            |
| Telefon    | string  | 20                    |                         |            |

|                 |        |       |  |   |
|-----------------|--------|-------|--|---|
| Adresa          | string | 50    |  | NOT NULL  |
| Data_nasterii   | date   |       |  |   |
| Istoric_medical | string | 10000 |  | Important pentru a cunoaste problemele medicale din trecut ale pacientului. Contine toate detaliile medicale. |

#### ENTITATE - DEPARTAMENT

| Atribut        | Tip     | Dimensiune / precizie | Val. posibile / default | Observatii |
|----------------|---------|-----------------------|-------------------------|------------|
| ID_Departament | integer |                       | [1,20]                  | PK         |
| Nume           | string  | 20                    |                         |            |
| Etaj           | integer |                       | [1,10]                  |            |

#### ENTITATE - PROGRAMARE

| Atribut        | Tip     | Dimensiune / precizie | Val. posibile / default | Observatii |
|----------------|---------|-----------------------|-------------------------|------------|
| ID_Programare  | integer |                       | [1,1000000]             | PK         |
| ID_Pacient     | integer |                       | [1,1000000]             | FK         |
| ID_Salon       | integer |                       | [1,1000]                | FK         |
| Data_internare | date    |                       |                         |            |
| Data_externare | date    |                       |                         |            |

#### ENTITATE - TRATAMENT

| Atribut             | Tip     | Dimensiune / precizie | Val. posibile / default | Observatii |
|---------------------|---------|-----------------------|-------------------------|------------|
| ID_Tratament        | integer |                       | [1,100000]              | PK         |
| ID_Doctor           | integer | 1000                  | [1,3000]                |            |
| ID_Pacient          | integer | 1000                  | [1,1000000]             |            |
| Observatii_medicale | string  | 1000                  |                         |            |

#### ENTITATE - MEDICAMENT

| Atribut | Tip | Dimensiune / precizie | Val. posibile / default | Observatii |
|---------|-----|-----------------------|-------------------------|------------|
|         |     |                       |                         |            |

|               |         |  |            |  |
|---------------|---------|--|------------|--|
| ID_Medicament | integer |  | [1,100000] | PK   |
| Administrare  | string  |  |            | Informatii pentru administrare corecta a tratamentului |

#### ENTITATE - FACTURA

| Atribut    | Tip     | Dimensiune / precizie | Val. posibile / default | Observatii                                      |
|------------|---------|-----------------------|-------------------------|---|
| ID_Factura | integer |                       | [1,100000]              | PK  |
| ID_Pacient | integer |                       | [1,1000000]             | FK  |
| Istoric    | string  | 10000                 |                         | Istoricul facturilor din trecut ale pacientului |

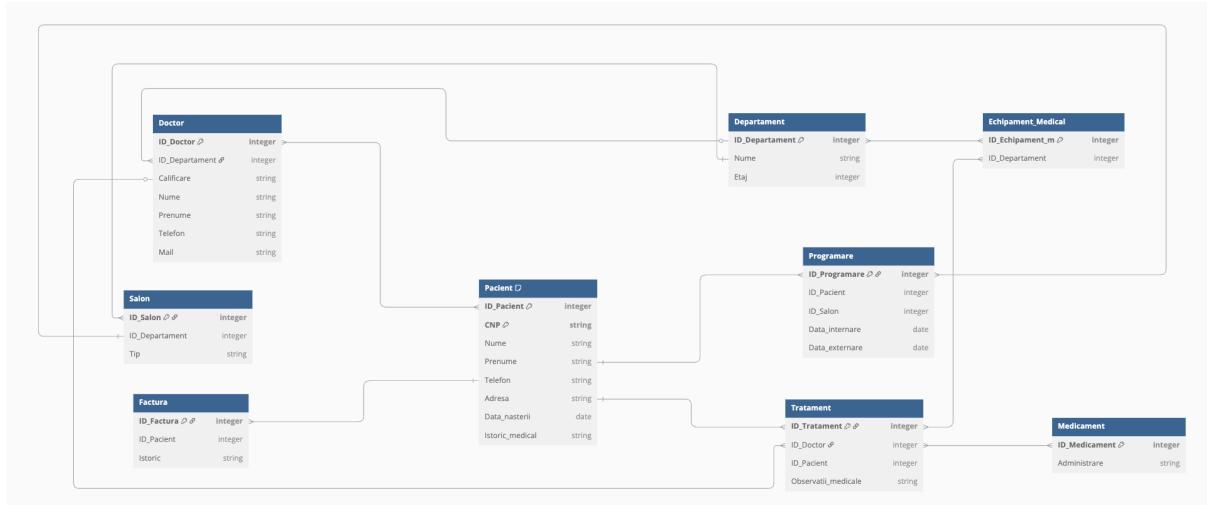
#### ENTITATE - SALON

| Atribut        | Tip     | Dimensiune / precizie | Val. posibile / default | Observatii       |
|----------------|---------|-----------------------|-------------------------|------------------|
| ID_Salon       | integer | [1,1000]              |                         | PK               |
| ID_Departament | integer | [1,20]                |                         | FK               |
| Tip            | string  |                       |                         | Comun sau privat |

#### ENTITATE - ECHIPAMENT\_MEDICAL

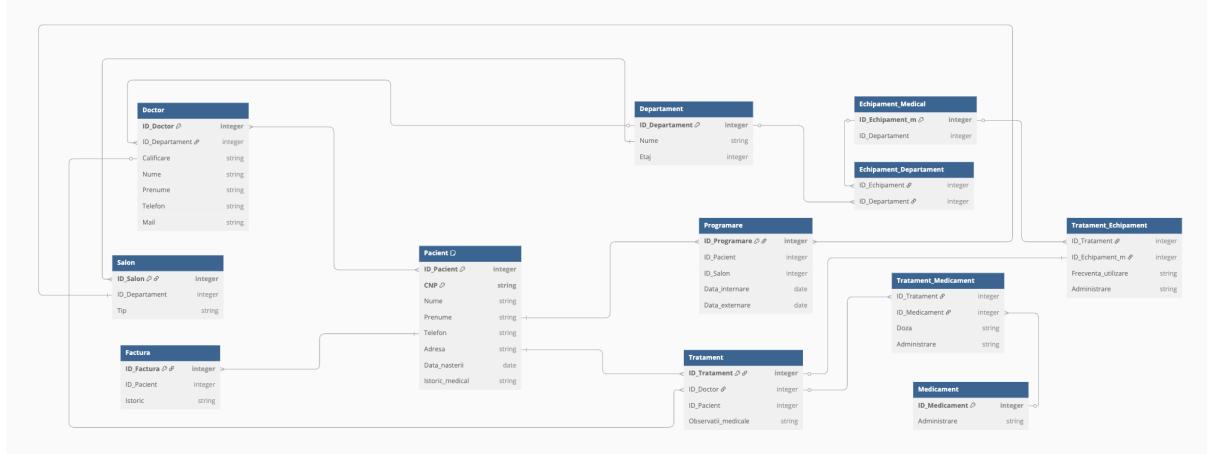
| Atribut         | Tip     | Dimensiune / precizie | Val. posibile / default | Observatii |
|-----------------|---------|-----------------------|-------------------------|------------|
| ID_Echipament_m | integer | [1,1000]              |                         | PK         |
| ID_Departament  | integer | [1,20]                |                         | FK         |

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



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

Diagrama conceptuală obținută trebuie să conțină minimum 7 tabele (fără considerarea subentităților), dintre care cel puțin un tabel asociativ.



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

- DOCTOR (ID\_Doctor#, ID\_Departament, Calificare, Nume, Prenume, Telefon, Mail)
- PACIENT (ID\_Pacient#, CNP#, Nume, Prenume, Telefon, Adresa, Data\_nasterii, Istoric\_medical)
- DEPARTAMENT (ID\_Departament#, Etaj, Nume)
- PROGRAMARE (ID\_Programare#, ID\_Pacient, ID\_Salon, Data\_internare, Data\_externare)
- TRATAMENT (ID\_Tratament#, ID\_Doctor, ID\_Pacient, Observatii\_medicatie)
- MEDICAMENT (ID\_Medicament#, Administrare)
- FACTURA (ID\_Factura#, ID\_Pacient, Istorice)
- SALON (ID\_Salon#, ID\_Departament, Tip)
- ECHIPAMENT\_MEDICAL (ID\_Echipament\_m#, ID\_Departament)
- TRATAMENT\_MEDICAMENT (ID\_Tratament, ID\_Medicament, Doza, Administrare)
- TRATAMENT\_ECHIPAMENT (ID\_Tratament, ID\_Echipament\_m, Frecventa\_utilizare, Administrare)

- ECHIPAMENT\_DEPARTAMENT (ID\_Echipament\_m, ID\_Departament)
9. Realizarea normalizării până la forma normală 3 (FN1 - FN3).
- FN1 - Dacă toate atributele conțin valori atomice și nu există grupuri repetitive de coloane.

Exemplu ipotetic:

Tabelul **Pacient**:

ID\_Pacient | Nume | Prenume | Telefon1 | Telefon2 | Telefon3 | ... (restul coloanelor)

Nu este în FN1 deoarece Telefon1, Telefon2, Telefon3 ... reprezintă grupuri repetitive.

- FN2 - Dacă este în FN1 și toate atributele non-cheie depind complet de cheia primă.

Exemplu ipotetic:

Tabelul **Tratament**:

(ID\_Tratament, ID\_Doctor, ID\_Pacient) | Observatii\_medicale | Nume\_Doctor  
unde (ID\_Tratament, ID\_Doctor, ID\_Pacient) este cheia primă compusă. Presupunem că tabelul **Doctor**, nu are coloanele “Nume” și “Prenume”.

Nu este în FN2 deoarece “Nume\_Doctor” depinde doar de “ID\_Doctor”, nu de toată cheia primă compusă. Pentru a rezolva aceasta problema vom adăuga în tabelul **Doctor** coloanele respective pentru nume și vom sterge “Nume\_Doctor” din tabelul **Tratament**.

- FN3 - Dacă este în FN2 și nu există dependențe tranzitive între atributele non-cheie.

Exemplu ipotetic:

Tabelul **Doctor**:

ID\_Doctor | ID\_Departament | Nume\_Departament | Calificare | ... (restul coloanelor)  
unde am adăugat “Nume\_Departament”

Nu este în FN3 deoarece “Nume\_Departament” depinde de “ID\_Departament” care depinde direct de “ID\_Doctor”, deci există o dependență tranzitivă.

$ID_{Doctor} \rightarrow ID_{Departament} \rightarrow Nume_{Departament}$

“Un doctor este asociat unui singur departament.

Intr-un departament lucrează unul sau mai mulți doctori.”

Pentru a rezolva vom avea în tabelul **Departament** coloana Nume\_Departament și o vom sterge din **Doctor**.

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

```
CREATE SEQUENCE seq_doctor_id START WITH 1 INCREMENT BY 1;
CREATE SEQUENCE seq_pacient_id START WITH 1 INCREMENT BY 1;
```

```

CREATE SEQUENCE seq_programare_id START WITH 1 INCREMENT BY 1;
CREATE SEQUENCE seq_departament_id START WITH 1 INCREMENT BY 1;
CREATE SEQUENCE seq_tratament_id START WITH 1 INCREMENT BY 1;
CREATE SEQUENCE seq_medicament_id START WITH 1 INCREMENT BY 1;
CREATE SEQUENCE seq_factura_id START WITH 1 INCREMENT BY 1;
CREATE SEQUENCE seq_salon_id START WITH 1 INCREMENT BY 1;
CREATE SEQUENCE seq_echipament_id START WITH 1 INCREMENT BY 1;

```

```

-- 10. Sechente
CREATE SEQUENCE seq_doctor_id START WITH 1 INCREMENT BY 1;
CREATE SEQUENCE seq_pacient_id START WITH 1 INCREMENT BY 1;
CREATE SEQUENCE seq_programare_id START WITH 1 INCREMENT BY 1;
CREATE SEQUENCE seq_departament_id START WITH 1 INCREMENT BY 1;
CREATE SEQUENCE seq_tratament_id START WITH 1 INCREMENT BY 1;
CREATE SEQUENCE seq_medicament_id START WITH 1 INCREMENT BY 1;
CREATE SEQUENCE seq_factura_id START WITH 1 INCREMENT BY 1;
CREATE SEQUENCE seq_salon_id START WITH 1 INCREMENT BY 1;
CREATE SEQUENCE seq_echipament_id START WITH 1 INCREMENT BY 1;

```

The screenshot shows the Oracle SQL Developer interface. On the left, there's a sidebar with sections for CONNECTIONS, SQL SNIPPETS, and REPORTS. Under CONNECTIONS, 'Grupa 143' is selected. In the main area, the 'SCRIPT OUTPUT' tab is active, displaying the following log:

```

Sequence SEQ_DOCTOR_ID created.

Sequence SEQ_PACIENT_ID created.

Sequence SEQ_PROGRAMARE_ID created.

Sequence SEQ_DEPARTAMENT_ID created.

Sequence SEQ_TRATAMENT_ID created.

Sequence SEQ_MEDICAMENT_ID created.

Sequence SEQ_FACTURA_ID created.

Sequence SEQ_SALON_ID created.

Sequence SEQ_ECHIPAMENT_ID created.

```

11. Crearea tabelelor in SQL si inserarea de date coerente in fiecare dintre acestea

(minimum 5 inregistrari in fiecare tabel neasociativ; minimum 10 inregistrari in tabele asociative; maxim 30 de inregistrari in fiecare tabel).

```

CREATE TABLE DEPARTAMENT (
    ID_Departament INTEGER PRIMARY KEY,
    Nume VARCHAR(20),
    Etaj INTEGER
)

```

);

```
CREATE TABLE DOCTOR (
    ID_Doctor INTEGER PRIMARY KEY,
    ID_Departament INTEGER,
    Calificare VARCHAR(20),
    Nume VARCHAR(20),
    Prenume VARCHAR(20),
    Telefon VARCHAR(11),
    Mail VARCHAR(30),
    FOREIGN KEY (ID_Departament) REFERENCES DEPARTAMENT(ID_Departament)
);
```

```
CREATE TABLE PACIENT (
    ID_Pacient INTEGER PRIMARY KEY,
    CNP VARCHAR(16) UNIQUE,
    Nume VARCHAR(20),
    Prenume VARCHAR(20),
    Telefon VARCHAR(20),
    Adresa VARCHAR(50),
    Data_nasterii DATE,
    Istorice_medical CLOB
);
```

```
CREATE TABLE SALON (
    ID_Salon INTEGER PRIMARY KEY,
    ID_Departament INTEGER,
    Tip VARCHAR(10),
    FOREIGN KEY (ID_Departament) REFERENCES DEPARTAMENT(ID_Departament)
);
```

```
CREATE TABLE PROGRAMARE (
    ID_Programare INTEGER PRIMARY KEY,
    ID_Pacient INTEGER,
    ID_Salon INTEGER,
    Data_internare DATE,
    Data_externare DATE,
    FOREIGN KEY (ID_Pacient) REFERENCES PACIENT(ID_Pacient),
    FOREIGN KEY (ID_Salon) REFERENCES SALON(ID_Salon)
);
```

```
CREATE TABLE TRATAMENT (
    ID_Tratament INTEGER PRIMARY KEY,
    ID_Doctor INTEGER,
    ID_Pacient INTEGER,
    Observatii_medicale CLOB
    FOREIGN KEY (ID_Doctor) REFERENCES Doctor(ID_Doctor),
    FOREIGN KEY (ID_Pacient) REFERENCES Pacient(ID_Pacient)
```

);

```
CREATE TABLE FACTURA (
    ID_Factura INTEGER PRIMARY KEY,
    ID_Pacient INTEGER,
    Istoric CLOB,
    FOREIGN KEY (ID_Pacient) REFERENCES PACIENT(ID_Pacient)
);
```

```
CREATE TABLE ECHIPAMENT_MEDICAL (
    ID_Echipament_m INTEGER PRIMARY KEY,
    ID_Departament INTEGER
);
```

```
CREATE TABLE MEDICAMENT (
    ID_Medicament INTEGER PRIMARY KEY,
    Administrare CLOB
);
```

```
CREATE TABLE TRATAMENT_MEDICAMENT (
    ID_Tratament INTEGER,
    ID_Medicament INTEGER,
    Doza VARCHAR(10),
    Administrare VARCHAR(50),
    PRIMARY KEY (ID_Tratament, ID_Medicament)
    FOREIGN KEY (ID_Tratament) REFERENCES Tratament(ID_Tratament),
    FOREIGN KEY (ID_Medicament) REFERENCES Medicament(ID_Medicament)
    FOREIGN KEY (ID_Tratament) REFERENCES Tratament(ID_Tratament),
    FOREIGN KEY (ID_Echipament_m) REFERENCES Echipament_Medical(ID_Echipament_m)
);
```

```
CREATE TABLE TRATAMENT_ECHIPAMENT (
    ID_Tratament INTEGER,
    ID_Echipament_m INTEGER,
    Frecventa_utilizare VARCHAR(10),
    Administrare VARCHAR(50),
    PRIMARY KEY (ID_Tratament, ID_Echipament_m)
);
```

```
CREATE TABLE ECHIPAMENT_DEPARTAMENT (
    ID_Echipament_m INTEGER,
    ID_Departament INTEGER,
    PRIMARY KEY (ID_Echipament_m, ID_Departament)
);
```

```

INSERT INTO DEPARTAMENT (ID_Departament, Nume, Etaj) VALUES
(seq_departament_id.NEXTVAL, 'Cardiologie', 2);
INSERT INTO DEPARTAMENT (ID_Departament, Nume, Etaj) VALUES
(seq_departament_id.NEXTVAL, 'Neurologie', 3);
INSERT INTO DEPARTAMENT (ID_Departament, Nume, Etaj) VALUES
(seq_departament_id.NEXTVAL, 'Pediatrie', 1);
INSERT INTO DEPARTAMENT (ID_Departament, Nume, Etaj) VALUES
(seq_departament_id.NEXTVAL, 'Oncologie', 4);
INSERT INTO DEPARTAMENT (ID_Departament, Nume, Etaj) VALUES
(seq_departament_id.NEXTVAL, 'Radiologie', 0);

INSERT INTO DOCTOR (ID_Doctor, ID_Departament, Calificare, Nume, Prenume, Telefon, Mail)
VALUES (seq_doctor_id.NEXTVAL, 1, 'rezident', 'Popescu', 'Ion', '0700000001',
'ion.popescu@email.com');

INSERT INTO DOCTOR (ID_Doctor, ID_Departament, Calificare, Nume, Prenume, Telefon, Mail)
VALUES (seq_doctor_id.NEXTVAL, 2, 'sef sectie', 'Ionescu', 'Maria', '0700000002',
'maria.ionescu@email.com');

INSERT INTO DOCTOR (ID_Doctor, ID_Departament, Calificare, Nume, Prenume, Telefon, Mail)
VALUES (seq_doctor_id.NEXTVAL, 3, 'asistent', 'Georgescu', 'Andrei', '0700000003',
'andrei.g@email.com');

INSERT INTO DOCTOR (ID_Doctor, ID_Departament, Calificare, Nume, Prenume, Telefon, Mail)
VALUES (seq_doctor_id.NEXTVAL, 4, 'rezident', 'Ciobanu', 'Ana', '0700000004',
'ana.c@email.com');

INSERT INTO DOCTOR (ID_Doctor, ID_Departament, Calificare, Nume, Prenume, Telefon, Mail)
VALUES (seq_doctor_id.NEXTVAL, 5, 'sef sectie', 'Dumitru', 'Paul', '0700000005',
'paul.d@email.com');

INSERT INTO PACIENT (ID_Pacient, CNP, Nume, Prenume, Telefon, Adresa, Data_nasterii,
Istoric_medical) VALUES (seq_pacient_id.NEXTVAL, '1960101123456', 'Popa', 'Alex', '0700001111',
'Str. Libertatii 10', TO_DATE('1960-01-01', 'YYYY-MM-DD'), TO_CLOB('hipertensiune cronica'));

INSERT INTO PACIENT (ID_Pacient, CNP, Nume, Prenume, Telefon, Adresa, Data_nasterii,
Istoric_medical) VALUES (seq_pacient_id.NEXTVAL, '2870202123456', 'Marin', 'Elena',
'0700002222', 'Str. Primaverii 45', TO_DATE('1987-02-02', 'YYYY-MM-DD'),
TO_CLOB('migrene'));

INSERT INTO PACIENT (ID_Pacient, CNP, Nume, Prenume, Telefon, Adresa, Data_nasterii,
Istoric_medical) VALUES (seq_pacient_id.NEXTVAL, '5010303123456', 'Iacob', 'Dan',
'0700003333', 'Str. Universitatii 3', TO_DATE('2001-03-03', 'YYYY-MM-DD'), TO_CLOB('alergii'));

INSERT INTO PACIENT (ID_Pacient, CNP, Nume, Prenume, Telefon, Adresa, Data_nasterii,
Istoric_medical) VALUES (seq_pacient_id.NEXTVAL, '6840404123456', 'Grigore', 'Ioana',
'0700004444', 'Str. Stadionului 12', TO_DATE('1984-04-04', 'YYYY-MM-DD'), TO_CLOB('diabet tip
2'));

INSERT INTO PACIENT (ID_Pacient, CNP, Nume, Prenume, Telefon, Adresa, Data_nasterii,
Istoric_medical) VALUES (seq_pacient_id.NEXTVAL, '2770505123456', 'Radu', 'Andreea',
'0700005555', 'Str. Mihai Viteazul 20', TO_DATE('1977-05-05', 'YYYY-MM-DD'),
TO_CLOB('astm'));

INSERT INTO SALON (ID_Salon, ID_Departament, Tip) VALUES (seq_salon_id.NEXTVAL, 1,
'comun');

```

```

INSERT INTO SALON (ID_Salon, ID_Departament, Tip) VALUES (seq_salon_id.NEXTVAL, 2, 'privat');
INSERT INTO SALON (ID_Salon, ID_Departament, Tip) VALUES (seq_salon_id.NEXTVAL, 3, 'comun');
INSERT INTO SALON (ID_Salon, ID_Departament, Tip) VALUES (seq_salon_id.NEXTVAL, 4, 'privat');
INSERT INTO SALON (ID_Salon, ID_Departament, Tip) VALUES (seq_salon_id.NEXTVAL, 5, 'comun');

INSERT INTO PROGRAMARE (ID_Programare, ID_Pacient, ID_Salon, Data_internare, Data_externare) VALUES (seq_programare_id.NEXTVAL, 1, 1, TO_DATE('2024-01-01', 'YYYY-MM-DD'), TO_DATE('2024-01-03', 'YYYY-MM-DD'));
INSERT INTO PROGRAMARE (ID_Programare, ID_Pacient, ID_Salon, Data_internare, Data_externare) VALUES (seq_programare_id.NEXTVAL, 2, 1, TO_DATE('2025-01-04', 'YYYY-MM-DD'), TO_DATE('2025-01-06', 'YYYY-MM-DD'));
INSERT INTO PROGRAMARE (ID_Programare, ID_Pacient, ID_Salon, Data_internare, Data_externare) VALUES (seq_programare_id.NEXTVAL, 3, 2, TO_DATE('2024-02-01', 'YYYY-MM-DD'), TO_DATE('2024-02-02', 'YYYY-MM-DD'));
INSERT INTO PROGRAMARE (ID_Programare, ID_Pacient, ID_Salon, Data_internare, Data_externare) VALUES (seq_programare_id.NEXTVAL, 4, 3, TO_DATE('2025-03-01', 'YYYY-MM-DD'), TO_DATE('2025-03-10', 'YYYY-MM-DD'));
INSERT INTO PROGRAMARE (ID_Programare, ID_Pacient, ID_Salon, Data_internare, Data_externare) VALUES (seq_programare_id.NEXTVAL, 5, 4, TO_DATE('2024-04-01', 'YYYY-MM-DD'), TO_DATE('2024-04-04', 'YYYY-MM-DD'));

INSERT INTO TRATAMENT (ID_Tratament, ID_Doctor, ID_Pacient, Observatii_medicale) VALUES (seq_tratament_id.NEXTVAL, 1, 1, 'Hipertensiune stabilizată');
INSERT INTO TRATAMENT (ID_Tratament, ID_Doctor, ID_Pacient, Observatii_medicale) VALUES (seq_tratament_id.NEXTVAL, 2, 2, 'Tratament migrene');
INSERT INTO TRATAMENT (ID_Tratament, ID_Doctor, ID_Pacient, Observatii_medicale) VALUES (seq_tratament_id.NEXTVAL, 3, 3, 'Alergii sezoniere');
INSERT INTO TRATAMENT (ID_Tratament, ID_Doctor, ID_Pacient, Observatii_medicale) VALUES (seq_tratament_id.NEXTVAL, 4, 4, 'Insulină zilnică');
INSERT INTO TRATAMENT (ID_Tratament, ID_Doctor, ID_Pacient, Observatii_medicale) VALUES (seq_tratament_id.NEXTVAL, 5, 5, 'Terapie astm ușor');

INSERT INTO FACTURA (ID_Factura, ID_Pacient, Istoric) VALUES (seq_factura_id.NEXTVAL, 1, 'Internare cardiologie, 2 zile');
INSERT INTO FACTURA (ID_Factura, ID_Pacient, Istoric) VALUES (seq_factura_id.NEXTVAL, 2, 'Consult neurologie, 1 zi');
INSERT INTO FACTURA (ID_Factura, ID_Pacient, Istoric) VALUES (seq_factura_id.NEXTVAL, 3, 'Alergii pediatrice, tratament ambulatoriu');
INSERT INTO FACTURA (ID_Factura, ID_Pacient, Istoric) VALUES (seq_factura_id.NEXTVAL, 4, 'Internare oncologie, tratament chimioterapie');
INSERT INTO FACTURA (ID_Factura, ID_Pacient, Istoric) VALUES (seq_factura_id.NEXTVAL, 5, 'Tratament fizioterapie pulmonară');

```

```

INSERT INTO ECHIPAMENT_MEDICAL (ID_Echipament_m, ID_Departament) VALUES
(seq_echipament_id.NEXTVAL, 1);
INSERT INTO ECHIPAMENT_MEDICAL (ID_Echipament_m, ID_Departament) VALUES
(seq_echipament_id.NEXTVAL, 2);
INSERT INTO ECHIPAMENT_MEDICAL (ID_Echipament_m, ID_Departament) VALUES
(seq_echipament_id.NEXTVAL, 3);
INSERT INTO ECHIPAMENT_MEDICAL (ID_Echipament_m, ID_Departament) VALUES
(seq_echipament_id.NEXTVAL, 4);
INSERT INTO ECHIPAMENT_MEDICAL (ID_Echipament_m, ID_Departament) VALUES
(seq_echipament_id.NEXTVAL, 5);

INSERT INTO MEDICAMENT (ID_Medicament, Administrare) VALUES
(seq_medicament_id.NEXTVAL, 'De 3 ori pe zi');
INSERT INTO MEDICAMENT (ID_Medicament, Administrare) VALUES
(seq_medicament_id.NEXTVAL, 'De 2 ori pe zi');
INSERT INTO MEDICAMENT (ID_Medicament, Administrare) VALUES
(seq_medicament_id.NEXTVAL, 'Atentie! Contine alergeni');
INSERT INTO MEDICAMENT (ID_Medicament, Administrare) VALUES
(seq_medicament_id.NEXTVAL, 'Variat in functie de nevoia pacientului');
INSERT INTO MEDICAMENT (ID_Medicament, Administrare) VALUES
(seq_medicament_id.NEXTVAL, 'De 4 ori pe zi');

INSERT INTO TRATAMENT_MEDICAMENT VALUES
(1, 1, '10mg', '1/zi dimineata'),
(1, 2, '5mg', '2/zi'),
(2, 3, '50mg', 'la nevoie'),
(2, 4, '25mg', '1/zi seara'),
(3, 5, '15mg', '1/zi'),
(3, 1, '5mg', 'dimineata'),
(4, 2, '10mg', 'injectabil'),
(4, 3, '10mg', 'oral'),
(5, 5, '20mg', '1/zi'),
(5, 4, '30mg', '2/zi');

INSERT INTO TRATAMENT_ECHIPAMENT VALUES
(1,1,'3/zi','monitorizare tensiune'),
(1,2,'2/luna','EKG'),
(2,2,'1/luna','scanare cerebrală'),
(3,3,'1/sapt','test alergii'),
(4,4,'3/zi','chimioterapie'),
(4,5,'2/luna','radioterapie'),
(5,5,'1/zi','hebulizare'),
(5,1,'4/zi','supraveghere respiratorie'),
(3,1,'1/sapt','pulsoximetru'),
(2,1,'1/luna','EEG');

INSERT INTO ECHIPAMENT_DEPARTAMENT VALUES
(1,1), (1,2), (2,2), (2,3), (3,3), (3,4), (4,4), (4,5), (5,5), (5,1);

```

SQL DEVELOPER ... 10-11.sql

Users > roland > Documents > Facultate > Baze de date > 10-11.sql > {} CREATE TABLE MEDICAMENT > Administrare CLOB

```

35 -- 11. Crearea tabelelor si inserarea datelor
36 CREATE TABLE DEPARTAMENT (
37     ID_Departament INTEGER PRIMARY KEY,
38     Nume VARCHAR(20),
39     Etaj INTEGER
40 );
41
42 CREATE TABLE DOCTOR (
43     ID_Docor INTEGER PRIMARY KEY,
44     ID_Departament INTEGER,
45     Calificare VARCHAR(20),
46     Nume VARCHAR(20),
47     Prenume VARCHAR(20),
48     Telefon VARCHAR(11),
49     Mail VARCHAR(30),
50     FOREIGN KEY (ID_Departament) REFERENCES DEPARTAMENT(ID_Departament)
51 );
52
53

```

SQL DEVELOPER ... 10-11.sql

Users > roland > Documents > Facultate > Baze de date > 10-11.sql > {} CREATE TABLE MEDICAMENT > Administrare CLOB

```

53
54 CREATE TABLE PACIENT (
55     ID_Pacient INTEGER PRIMARY KEY,
56     CNP VARCHAR(16) UNIQUE,
57     Nume VARCHAR(20),
58     Prenume VARCHAR(20),
59     Telefon VARCHAR(20),
60     Adresa VARCHAR(50),
61     Data_nasterii DATE,
62     Istoric_medical CLOB
63 );
64
65 CREATE TABLE SALON (
66     ID_Salon INTEGER PRIMARY KEY,
67     ID_Departament INTEGER,
68     Tip VARCHAR(10),
69     FOREIGN KEY (ID_Departament) REFERENCES DEPARTAMENT(ID_Departament)
70 );
71

```

10-11.sql

```

71
72 CREATE TABLE PROGRAMARE (
73     ID_Programare INTEGER PRIMARY KEY,
74     ID_Pacient INTEGER,
75     ID_Salon INTEGER,
76     Data_internare DATE,
77     Data_externare DATE,
78     FOREIGN KEY (ID_Pacient) REFERENCES PACIENT(ID_Pacient),
79     FOREIGN KEY (ID_Salon) REFERENCES SALON(ID_Salon)
80 );
81
82 CREATE TABLE TRATAMENT (
83     ID_Tratament INTEGER PRIMARY KEY,
84     ID_Doctor INTEGER,
85     ID_Pacient INTEGER,
86     Observatii_medicale CLOB,
87     FOREIGN KEY (ID_Doctor) REFERENCES Doctor(ID_Doctor),
88     FOREIGN KEY (ID_Pacient) REFERENCES Pacient(ID_Pacient)
89 );
90
91 CREATE TABLE FACTURA (
92     ID_Factura INTEGER PRIMARY KEY,
93     ID_Pacient INTEGER,
94     Istoric CLOB,
95     FOREIGN KEY (ID_Pacient) REFERENCES PACIENT(ID_Pacient)
96 );
97

```

```

12-17.txt
18.sql
18.txt
19.js
19.txt
143_Boldesco_Roland.txt
143_Boldeso_Roland.sql
bd6.txt
bd7.txt
Proiect BD (1).pdf
Proiect BD.docx
recapitulare.sql
Screenshot 2025-05-25 at 09.36.49....
Tema.sql
test.txt

97   CREATE TABLE ECHIPAMENT_MEDICAL (
98     ID_Echipament_m INTEGER PRIMARY KEY,
99     ID_Departament INTEGER
100    );
101  );
102
103  CREATE TABLE MEDICAMENT (
104    ID_Medicament INTEGER PRIMARY KEY,
105    Administrare CLOB
106  );
107
108  CREATE TABLE TRATAMENT_MEDICAMENT (
109    ID_Tratament INTEGER,
110    ID_Medicament INTEGER,
111    Doza VARCHAR(10),
112    Administrare VARCHAR(50),
113    PRIMARY KEY (ID_Tratament, ID_Medicament),
114    FOREIGN KEY (ID_Tratament) REFERENCES Tratament(ID_Tratament),
115    FOREIGN KEY (ID_Medicament) REFERENCES Medicament(ID_Medicament)
116  );
117
118  CREATE TABLE TRATAMENT_ECHIPAMENT (
119    ID_Tratament INTEGER,
120    ID_Echipament_m INTEGER,
121    Frecventa_utilizare VARCHAR(10),
122    Administrare VARCHAR(50),
123    PRIMARY KEY (ID_Tratament, ID_Echipament_m),
124    FOREIGN KEY (ID_Tratament) REFERENCES Tratament(ID_Tratament),
125    FOREIGN KEY (ID_Echipament_m) REFERENCES Echipament_Medical(ID_Echipament_m)
126  );
127

```

SQL DEVELOPER ...

CONNECTIONS

- Grupa 143
- Tables
- Views
- Duality Views
- Indexes
- Packages
- Procedures
- Functions
- MLE Environments
- JavaScript Modules

SQL SNIPPETS

- Aggregate Functions
- Analytic Functions
- Character Functions
- Conversion Functions
- Date Formats
- Date/Time Functions
- Number Formats
- Numeric Functions
- Optimizer Hints

```

10-11.sql x
Users > roland > Documents > Facultate > Baze de date > 10-11.sql > {} CREATE TABLE MEDICAMENT > Administrare CLOB
122
123  CREATE TABLE ECHIPAMENT_DEPARTAMENT (
124    ID_Echipament_m INTEGER,
125    ID_Departament INTEGER,
126    PRIMARY KEY (ID_Echipament_m, ID_Departament)
127  );
128
129  INSERT INTO DEPARTAMENT (ID_Departament, Nume, Etaj) VALUES (seq_departament_id.NEXTVAL, 'Cardiologie', 2);
130  INSERT INTO DEPARTAMENT (ID_Departament, Nume, Etaj) VALUES (seq_departament_id.NEXTVAL, 'Neurologie', 3);
131  INSERT INTO DEPARTAMENT (ID_Departament, Nume, Etaj) VALUES (seq_departament_id.NEXTVAL, 'Pediatrie', 1);
132  INSERT INTO DEPARTAMENT (ID_Departament, Nume, Etaj) VALUES (seq_departament_id.NEXTVAL, 'Oncologie', 4);
133  INSERT INTO DEPARTAMENT (ID_Departament, Nume, Etaj) VALUES (seq_departament_id.NEXTVAL, 'Radiologie', 0);
134
135  INSERT INTO DOCTOR (ID_Doctor, ID_Departament, Calificare, Nume, Prenume, Telefon, Mail) VALUES (seq_doctor_id.NEXTVAL, 1, 'rezident', 'Dr. Ion Popescu');
136  INSERT INTO DOCTOR (ID_Doctor, ID_Departament, Calificare, Nume, Prenume, Telefon, Mail) VALUES (seq_doctor_id.NEXTVAL, 2, 'sef sectie', 'Dr. Maria Gheorghiu');
137  INSERT INTO DOCTOR (ID_Doctor, ID_Departament, Calificare, Nume, Prenume, Telefon, Mail) VALUES (seq_doctor_id.NEXTVAL, 3, 'asistent', 'Dr. Stefan Marinescu');
138  INSERT INTO DOCTOR (ID_Doctor, ID_Departament, Calificare, Nume, Prenume, Telefon, Mail) VALUES (seq_doctor_id.NEXTVAL, 4, 'rezident', 'Dr. Daniela Popovici');
139  INSERT INTO DOCTOR (ID_Doctor, ID_Departament, Calificare, Nume, Prenume, Telefon, Mail) VALUES (seq_doctor_id.NEXTVAL, 5, 'sef sectie', 'Dr. Bogdan Marinescu');
140
141  INSERT INTO PACIENT (ID_Pacient, CNP, Nume, Prenume, Telefon, Adresa, Data_nasterii, Istoric_medical) VALUES (seq_pacient_id.NEXTVAL, '123456789012345678', 'Popescu', 'Ion', 'Ionel', '0755555555', 'Str. Unirii 12, Sector 1', '1990-01-01', 'Boli cronice');
142  INSERT INTO PACIENT (ID_Pacient, CNP, Nume, Prenume, Telefon, Adresa, Data_nasterii, Istoric_medical) VALUES (seq_pacient_id.NEXTVAL, '123456789012345679', 'Gheorghiu', 'Maria', 'Maria', '0755555556', 'Str. Unirii 12, Sector 1', '1990-01-01', 'Boli cronice');
143  INSERT INTO PACIENT (ID_Pacient, CNP, Nume, Prenume, Telefon, Adresa, Data_nasterii, Istoric_medical) VALUES (seq_pacient_id.NEXTVAL, '123456789012345680', 'Marinescu', 'Stefan', 'Stefan', '0755555557', 'Str. Unirii 12, Sector 1', '1990-01-01', 'Boli cronice');
144  INSERT INTO PACIENT (ID_Pacient, CNP, Nume, Prenume, Telefon, Adresa, Data_nasterii, Istoric_medical) VALUES (seq_pacient_id.NEXTVAL, '123456789012345681', 'Popovici', 'Daniela', 'Daniela', '0755555558', 'Str. Unirii 12, Sector 1', '1990-01-01', 'Boli cronice');
145  INSERT INTO PACIENT (ID_Pacient, CNP, Nume, Prenume, Telefon, Adresa, Data_nasterii, Istoric_medical) VALUES (seq_pacient_id.NEXTVAL, '123456789012345682', 'Marinescu', 'Bogdan', 'Bogdan', '0755555559', 'Str. Unirii 12, Sector 1', '1990-01-01', 'Boli cronice');


```

SQL DEVELOPER ...

CONNECTIONS

- Grupa 143
- Tables
- Views
- Duality Views
- Indexes
- Packages
- Procedures
- Functions
- MLE Environments
- JavaScript Modules

SQL SNIPPETS

- Aggregate Functions
- Analytic Functions
- Character Functions
- Conversion Functions
- Date Formats
- Date/Time Functions
- Number Formats
- Numeric Functions
- Optimizer Hints

```

10-11.sql x
Users > roland > Documents > Facultate > Baze de date > 10-11.sql > {} CREATE TABLE MEDICAMENT > Administrare CLOB
146
147  INSERT INTO SALON (ID_Salon, ID_Departament, Tip) VALUES (seq_salon_id.NEXTVAL, 1, 'comun');
148  INSERT INTO SALON (ID_Salon, ID_Departament, Tip) VALUES (seq_salon_id.NEXTVAL, 2, 'privat');
149  INSERT INTO SALON (ID_Salon, ID_Departament, Tip) VALUES (seq_salon_id.NEXTVAL, 3, 'comun');
150  INSERT INTO SALON (ID_Salon, ID_Departament, Tip) VALUES (seq_salon_id.NEXTVAL, 4, 'privat');
151  INSERT INTO SALON (ID_Salon, ID_Departament, Tip) VALUES (seq_salon_id.NEXTVAL, 5, 'comun');
152
153  INSERT INTO PROGRAMARE (ID_Programare, ID_Pacient, ID_Salon, Data_internare, Data_externare) VALUES (seq_programare_id.NEXTVAL, 1, 1, TO_DATE('2025-05-25', 'YYYY-MM-DD'), TO_DATE('2025-05-25', 'YYYY-MM-DD'));
154  INSERT INTO PROGRAMARE (ID_Programare, ID_Pacient, ID_Salon, Data_internare, Data_externare) VALUES (seq_programare_id.NEXTVAL, 2, 1, TO_DATE('2025-05-25', 'YYYY-MM-DD'), TO_DATE('2025-05-25', 'YYYY-MM-DD'));
155  INSERT INTO PROGRAMARE (ID_Programare, ID_Pacient, ID_Salon, Data_internare, Data_externare) VALUES (seq_programare_id.NEXTVAL, 3, 2, TO_DATE('2025-05-25', 'YYYY-MM-DD'), TO_DATE('2025-05-25', 'YYYY-MM-DD'));
156  INSERT INTO PROGRAMARE (ID_Programare, ID_Pacient, ID_Salon, Data_internare, Data_externare) VALUES (seq_programare_id.NEXTVAL, 4, 3, TO_DATE('2025-05-25', 'YYYY-MM-DD'), TO_DATE('2025-05-25', 'YYYY-MM-DD'));
157  INSERT INTO PROGRAMARE (ID_Programare, ID_Pacient, ID_Salon, Data_internare, Data_externare) VALUES (seq_programare_id.NEXTVAL, 5, 4, TO_DATE('2025-05-25', 'YYYY-MM-DD'), TO_DATE('2025-05-25', 'YYYY-MM-DD'));

158
159  INSERT INTO TRATAMENT (ID_Tratament, ID_Doctor, ID_Pacient, Observatii_medicele) VALUES (seq_tratament_id.NEXTVAL, 1, 1, 'Hipertensiune arteriala');
160  INSERT INTO TRATAMENT (ID_Tratament, ID_Doctor, ID_Pacient, Observatii_medicele) VALUES (seq_tratament_id.NEXTVAL, 2, 2, 'Tratament migra');
161  INSERT INTO TRATAMENT (ID_Tratament, ID_Doctor, ID_Pacient, Observatii_medicele) VALUES (seq_tratament_id.NEXTVAL, 3, 3, 'Alergii sezonii');
162  INSERT INTO TRATAMENT (ID_Tratament, ID_Doctor, ID_Pacient, Observatii_medicele) VALUES (seq_tratament_id.NEXTVAL, 4, 4, 'Insulină zilnică');
163  INSERT INTO TRATAMENT (ID_Tratament, ID_Doctor, ID_Pacient, Observatii_medicele) VALUES (seq_tratament_id.NEXTVAL, 5, 5, 'Terapie astm');
164
165  INSERT INTO FACTURA (ID_Factura, ID_Pacient, Istoric) VALUES (seq_factura_id.NEXTVAL, 1, 'Internare cardiologie, 2 zile');
166  INSERT INTO FACTURA (ID_Factura, ID_Pacient, Istoric) VALUES (seq_factura_id.NEXTVAL, 2, 'Consult neurologie, 1 zi');
167  INSERT INTO FACTURA (ID_Factura, ID_Pacient, Istoric) VALUES (seq_factura_id.NEXTVAL, 3, 'Alergii pediatrice, tratament ambulatoriu');
168  INSERT INTO FACTURA (ID_Factura, ID_Pacient, Istoric) VALUES (seq_factura_id.NEXTVAL, 4, 'Internare oncologie, tratament chimioterapie');
169  INSERT INTO FACTURA (ID_Factura, ID_Pacient, Istoric) VALUES (seq_factura_id.NEXTVAL, 5, 'Tratament fizioterapie pulmonară');


```

**SQL DEVELOPER** ... **10-11.sql** ...

**CONNECTIONS**

- Grupa 143
- > Tables
- > Views
- > Duality Views
- > Indexes
- > Packages
- > Procedures
- > Functions
- > MLE Environments
- > JavaScript Modules

**SQL SNIPPETS**

- > Aggregate Functions
- > Analytic Functions
- > Character Functions
- > Conversion Functions
- > Date Formats
- > Date/Time Functions
- > Number Formats
- > Numeric Functions
- > Optimizer Hints

```

170  INSERT INTO ECHIPAMENT_MEDICAL (ID_Echipament_m, ID_Departament) VALUES (seq_echipament_id.NEXTVAL, 1);
171  INSERT INTO ECHIPAMENT_MEDICAL (ID_Echipament_m, ID_Departament) VALUES (seq_echipament_id.NEXTVAL, 2);
172  INSERT INTO ECHIPAMENT_MEDICAL (ID_Echipament_m, ID_Departament) VALUES (seq_echipament_id.NEXTVAL, 3);
173  INSERT INTO ECHIPAMENT_MEDICAL (ID_Echipament_m, ID_Departament) VALUES (seq_echipament_id.NEXTVAL, 4);
174  INSERT INTO ECHIPAMENT_MEDICAL (ID_Echipament_m, ID_Departament) VALUES (seq_echipament_id.NEXTVAL, 5);
175  INSERT INTO ECHIPAMENT_MEDICAL (ID_Echipament_m, ID_Departament) VALUES (seq_echipament_id.NEXTVAL, 6);
176  INSERT INTO MEDICAMENT (ID_Medicament, Administrare) VALUES (seq_medicament_id.NEXTVAL, 'De 3 ori pe zi');
177  INSERT INTO MEDICAMENT (ID_Medicament, Administrare) VALUES (seq_medicament_id.NEXTVAL, 'De 2 ori pe zi');
178  INSERT INTO MEDICAMENT (ID_Medicament, Administrare) VALUES (seq_medicament_id.NEXTVAL, 'Atentie! Contine alergeni');
179  INSERT INTO MEDICAMENT (ID_Medicament, Administrare) VALUES (seq_medicament_id.NEXTVAL, 'Variat in functie de nevoia pacientului');
180  INSERT INTO MEDICAMENT (ID_Medicament, Administrare) VALUES (seq_medicament_id.NEXTVAL, 'De 4 ori pe zi');

182  INSERT INTO TRATAMENT_MEDICAMENT VALUES
183  (1, 1, '10mg', '1/zi dimineata'),
184  (1, 2, '5mg', '2/zi'),
185  (2, 3, '50mg', 'la nevoie'),
186  (2, 4, '25mg', '1/zi seara'),
187  (3, 5, '15mg', '1/zi'),
188  (3, 1, '5mg', 'dimineata'),
189  (4, 2, '10mg', 'injetabil'),
190  (4, 3, '10mg', 'oral'),
191  (5, 5, '20mg', '1/zi'),
192  (5, 4, '30mg', '2/zi');

194  INSERT INTO TRATAMENT_ECHIPAMENT VALUES
195  (1,1,'3/zi','monitorizare tensiune'),
196  (1,2,'2/luna','EKG'),
197  (2,2,'1/luna','scanare cerebrală'),
198  (3,3,'1/sapt','test alergii'),
199  (4,4,'3/zi','chimioterapie'),
200  (4,5,'2/luna','radioterapie'),
201  (5,5,'1/zi','nebulizare'),
202  (5,1,'4/zi','supraveghere respiratorie'),
203  (3,1,'1/sapt','pulsoximetru'),
204  (2,1,'1/luna','EEG');

206  INSERT INTO ECHIPAMENT_DEPARTAMENT VALUES
207  (1,1), (1,2), (2,2), (2,3), (3,3), (3,4), (4,4), (4,5), (5,5), (5,1);

210  COMMIT;

```

**SQL DEVELOPER** ... **10-11.sql** ...

**CONNECTIONS**

- Grupa 143
- > Tables
- > Views
- > Duality Views
- > Indexes
- > Packages
- > Procedures
- > Functions
- > MLE Environments
- > JavaScript Modules

**SQL SNIPPETS**

- > Aggregate Functions
- > Analytic Functions
- > Character Functions
- > Conversion Functions
- > Date Formats

```

191  (1, 1, '10mg', '1/zi'),
192  (5, 5, '20mg', '1/zi'),
193  (5, 4, '30mg', '2/zi');

194  INSERT INTO TRATAMENT_ECHIPAMENT VALUES
195  (1,1,'3/zi','monitorizare tensiune'),
196  (1,2,'2/luna','EKG'),
197  (2,2,'1/luna','scanare cerebrală'),
198  (3,3,'1/sapt','test alergii'),
199  (4,4,'3/zi','chimioterapie'),
200  (4,5,'2/luna','radioterapie'),
201  (5,5,'1/zi','nebulizare'),
202  (5,1,'4/zi','supraveghere respiratorie'),
203  (3,1,'1/sapt','pulsoximetru'),
204  (2,1,'1/luna','EEG');

206  INSERT INTO ECHIPAMENT_DEPARTAMENT VALUES
207  (1,1), (1,2), (2,2), (2,3), (3,3), (3,4), (4,4), (4,5), (5,5), (5,1);

210  COMMIT;

```

**SQL DEVELOPER** ... **10-11.sql** ...

**CONNECTIONS**

- Grupa 143
- > Tables
- > Views
- > Duality Views
- > Indexes
- > Packages
- > Procedures
- > Functions
- > MLE Environments
- > JavaScript Modules

**SQL SNIPPETS**

- > Aggregate Functions
- > Analytic Functions
- > Character Functions
- > Conversion Functions
- > Date Formats
- > Date/Time Functions
- > Number Formats
- > Numeric Functions
- > Optimizer Hints
- > PL/SQL Programming Techniques

**REPORTS**

- > Database Administration
- > About Your Database
- > APEX
- > ASH and AWR
- > Data Dictionary
- > PLSQL
- > Security
- > Streams
- > Table
- > XML

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS QUERY RESULT **SCRIPT OUTPUT** SQL HISTORY TASK MONITOR

```

Table DEPARTAMENT created.

Table DOCTOR created.

Table PACIENT created.

Table SALON created.

Table PROGRAMARE created.

Table TRATAMENT created.

Table FACTURA created.

Table ECHIPAMENT_MEDICAL created.

Table MEDICAMENT created.

Table TRATAMENT_MEDICAMENT created.

Table TRATAMENT_ECHIPAMENT created.

Table ECHIPAMENT_DEPARTAMENT created.

```

SQL DEVELOPER ... PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS QUERY RESULT SCRIPT OUTPUT SQL HISTORY TASK MONITOR

**CONNECTIONS**

- > Grupa 143
- > Tables
- > Views
- > Duality Views
- > Indexes
- > Packages
- > Procedures
- > Functions
- > MLE Environments
- > JavaScript Modules

1 row inserted.

10 rows inserted.

10 rows inserted.

10 rows inserted.

Commit complete.

## DEPARTAMENT

|   | COLUMN_NAME    | DATA_TYPE         | NULLABLE | DATA_DEFAULT | COLUMN_ID | COMMENTS |
|---|----------------|-------------------|----------|--------------|-----------|----------|
| 1 | ID_DEPARTAMENT | NUMBER(38,0)      | No       | (null)       | 1         | (null)   |
| 2 | NUME           | VARCHAR2(20 BYTE) | Yes      | (null)       | 2         | (null)   |
| 3 | ETAJ           | NUMBER(38,0)      | Yes      | (null)       | 3         | (null)   |

|   | ID_DEPARTAMENT | NUME        | ETAJ |
|---|----------------|-------------|------|
| 1 | 1              | Cardiologie | 2    |
| 2 | 2              | Neurologie  | 3    |
| 3 | 3              | Pediatrie   | 1    |
| 4 | 4              | Oncologie   | 4    |
| 5 | 5              | Radiologie  | 0    |

## DOCTOR

|   | COLUMN_NAME    | DATA_TYPE         | NULLABLE | DATA_DEFAULT | COLUMN_ID | COMMENTS |
|---|----------------|-------------------|----------|--------------|-----------|----------|
| 1 | ID_DOCTOR      | NUMBER(38,0)      | No       | (null)       | 1         | (null)   |
| 2 | ID_DEPARTAMENT | NUMBER(38,0)      | Yes      | (null)       | 2         | (null)   |
| 3 | CALIFICARE     | VARCHAR2(20 BYTE) | Yes      | (null)       | 3         | (null)   |
| 4 | NUME           | VARCHAR2(20 BYTE) | Yes      | (null)       | 4         | (null)   |
| 5 | PRENUME        | VARCHAR2(20 BYTE) | Yes      | (null)       | 5         | (null)   |
| 6 | TELEFON        | VARCHAR2(11 BYTE) | Yes      | (null)       | 6         | (null)   |
| 7 | MAIL           | VARCHAR2(30 BYTE) | Yes      | (null)       | 7         | (null)   |

|   | ID_DOCTOR | ID_DEPARTAMENT | CALIFICARE | NUME      | PRENUME | TELEFON    | MAIL                    |
|---|-----------|----------------|------------|-----------|---------|------------|-------------------------|
| 1 | 1         | 1              | rezident   | Popescu   | Ion     | 0700000001 | ion.popescu@email.com   |
| 2 | 2         | 2              | sef sectie | Ionescu   | Maria   | 0700000002 | maria.ionescu@email.com |
| 3 | 3         | 3              | asistent   | Georgescu | Andrei  | 0700000003 | andrei.g@email.com      |
| 4 | 4         | 4              | rezident   | Ciobanu   | Ana     | 0700000004 | ana.c@email.com         |
| 5 | 5         | 5              | sef sectie | Dumitru   | Paul    | 0700000005 | paul.d@email.com        |

## ECHIPAMENT\_MEDICAL

|   | COLUMN_NAME     | DATA_TYPE    | NULLABLE       | DATA_DEFAULT | COLUMN_ID | COMMENTS |
|---|-----------------|--------------|----------------|--------------|-----------|----------|
| 1 | ID_ECHIPAMENT_M | NUMBER(38,0) | No             | (null)       | 1         | (null)   |
| 2 | ID_DEPARTAMENT  | NUMBER(38,0) | Yes            | (null)       | 2         | (null)   |
|   | ID_ECHIPAMENT_M |              | ID_DEPARTAMENT |              |           |          |
| 1 | 1               |              |                | 1            |           |          |
| 2 | 2               |              |                | 2            |           |          |
| 3 | 3               |              |                | 3            |           |          |
| 4 | 4               |              |                | 4            |           |          |
| 5 | 5               |              |                | 5            |           |          |

## MEDICAMENT

|   | COLUMN_NAME   | DATA_TYPE    | NULLABLE     | DATA_DEFAULT                            | COLUMN_ID | COMMENTS |
|---|---------------|--------------|--------------|---|-----------|----------|
| 1 | ID_MEDICAMENT | NUMBER(38,0) | No           | (null)                                  | 1         | (null)   |
| 2 | ADMINISTRARE  | CLOB         | Yes          | (null)                                  | 2         | (null)   |
|   | ID_MEDICAMENT |              | ADMINISTRARE |   |           |          |
| 1 | 1             |              |              | De 3 ori pe zi                          |           |          |
| 2 | 2             |              |              | De 2 ori pe zi                          |           |          |
| 3 | 3             |              |              | Atentie! Contine alergeni               |           |          |
| 4 | 4             |              |              | Variat in functie de nevoia pacientului |           |          |
| 5 | 5             |              |              | De 4 ori pe zi                          |           |          |

## PACIENT

|   | COLUMN_NAME     | DATA_TYPE         | NULLABLE | DATA_DEFAULT | COLUMN_ID  | COMMENTS               |               |                 |
|---|-----------------|-------------------|----------|--------------|------------|------------------------|---------------|-----------------|
| 1 | ID_PACIENT      | NUMBER(38,0)      | No       | (null)       | 1          | (null)                 |               |                 |
| 2 | CNP             | VARCHAR2(16 BYTE) | Yes      | (null)       | 2          | (null)                 |               |                 |
| 3 | NUME            | VARCHAR2(20 BYTE) | Yes      | (null)       | 3          | (null)                 |               |                 |
| 4 | PRENUME         | VARCHAR2(20 BYTE) | Yes      | (null)       | 4          | (null)                 |               |                 |
| 5 | TELEFON         | VARCHAR2(20 BYTE) | Yes      | (null)       | 5          | (null)                 |               |                 |
| 6 | ADRESA          | VARCHAR2(50 BYTE) | Yes      | (null)       | 6          | (null)                 |               |                 |
| 7 | DATA_NASTERII   | DATE              | Yes      | (null)       | 7          | (null)                 |               |                 |
| 8 | ISTORIC_MEDICAL | CLOB              | Yes      | (null)       | 8          | (null)                 |               |                 |
|   | ID_PACIENT      | CNP               | NUME     | PRENUME      | TELEFON    | ADRESA                 | DATA_NASTERII | ISTORIC_MEDICAL |
| 1 | 1               | 1960101123456     | Popa     | Alex         | 0700001111 | Str. Libertatii 10     | 01/01/60      | hipertensiune   |
| 2 | 2               | 2870202123456     | Marin    | Elena        | 0700002222 | Str. Primaverii 45     | 02/02/87      | migrene         |
| 3 | 3               | 5010303123456     | Iacob    | Dan          | 0700003333 | Str. Universitatii 3   | 03/03/01      | alergii         |
| 4 | 4               | 6840404123456     | Grigore  | Ioana        | 0700004444 | Str. Stadionului 12    | 04/04/84      | diabet tip 2    |
| 5 | 5               | 2770505123456     | Radu     | Andreea      | 0700005555 | Str. Mihai Viteazul 20 | 05/05/77      | astm            |

## SALON

|   | COLUMN_NAME    | DATA_TYPE         | NULLABLE       | DATA_DEFAULT | COLUMN_ID | COMMENTS |
|---|----------------|-------------------|----------------|--------------|-----------|----------|
| 1 | ID_SALON       | NUMBER(38,0)      | No             | (null)       | 1         | (null)   |
| 2 | ID_DEPARTAMENT | NUMBER(38,0)      | Yes            | (null)       | 2         | (null)   |
| 3 | TIP            | VARCHAR2(10 BYTE) | Yes            | (null)       | 3         | (null)   |
|   | ID_SALON       |                   | ID_DEPARTAMENT |              | TIP       |          |
| 1 | 1              |                   | 1              |              | comun     |          |
| 2 | 2              |                   | 2              |              | privat    |          |
| 3 | 3              |                   | 3              |              | comun     |          |
| 4 | 4              |                   | 4              |              | privat    |          |
| 5 | 5              |                   | 5              |              | comun     |          |

## TRATAMENT

|  | COLUMN_NAME         | DATA_TYPE    | NULLABLE | DATA_DEFAULT | COLUMN_ID | COMMENTS |
|--|---------------------|--------------|----------|--------------|-----------|----------|
|  | ID_TRAITEMENT       | NUMBER(38,0) | No       | (null)       | 1         | (null)   |
|  | ID_DOCTOR           | NUMBER(38,0) | Yes      | (null)       | 2         | (null)   |
|  | ID_PACIENT          | NUMBER(38,0) | Yes      | (null)       | 3         | (null)   |
|  | OBSERVATII_MEDICALE | CLOB         | Yes      | (null)       | 4         | (null)   |

|   | ID_TRAITEMENT | ID_DOCTOR | ID_PACIENT | OBSERVATII_MEDICALE       |
|---|---------------|-----------|------------|---------------------------|
| 1 | 1             | 1         | 1          | Hipertensiune stabilizată |
| 2 | 2             | 2         | 2          | Tratament migrene         |
| 3 | 3             | 3         | 3          | Alergii sezoniere         |
| 4 | 4             | 4         | 4          | Insulină zilnică          |
| 5 | 5             | 5         | 5          | Terapie astm ușor         |

## FACTURA

|   | COLUMN_NAME | DATA_TYPE    | NULLABLE | DATA_DEFAULT | COLUMN_ID | COMMENTS |
|---|-------------|--------------|----------|--------------|-----------|----------|
| 1 | ID_FACTURA  | NUMBER(38,0) | No       | (null)       | 1         | (null)   |
| 2 | ID_PACIENT  | NUMBER(38,0) | Yes      | (null)       | 2         | (null)   |
| 3 | ISTORIC     | CLOB         | Yes      | (null)       | 3         | (null)   |

|   | ID_FACTURA | ID_PACIENT | ISTORIC                                      |
|---|------------|------------|--|
| 1 | 1          | 1          | Internare cardiologie, 2 zile                |
| 2 | 2          | 2          | Consult neurologie, 1 zi                     |
| 3 | 3          | 3          | Alergii pediatrice, tratament ambulatoriu    |
| 4 | 4          | 4          | Internare oncologie, tratament chimioterapie |
| 5 | 5          | 5          | Tratament fizioterapie pulmonară             |

## PROGRAMARE

|   | COLUMN_NAME      | DATA_TYPE    | NULLABLE | DATA_DEFAULT | COLUMN_ID | COMMENTS |
|---|------------------|--------------|----------|--------------|-----------|----------|
| 1 | ID_PROGRAMARE    | NUMBER(38,0) | No       | (null)       | 1         | (null)   |
| 2 | ID_PACIENT       | NUMBER(38,0) | Yes      | (null)       | 2         | (null)   |
| 3 | ID_SALON         | NUMBER(38,0) | Yes      | (null)       | 3         | (null)   |
| 4 | DATA_INTERNALARE | DATE         | Yes      | (null)       | 4         | (null)   |
| 5 | DATA_EXTERNALARE | DATE         | Yes      | (null)       | 5         | (null)   |

|   | ID_PROGRAMARE | ID_PACIENT | ID_SALON | DATA_INTERNALARE | DATA_EXTERNALARE |
|---|---------------|------------|----------|------------------|------------------|
| 1 | 1             | 1          | 1        | 01/01/24         | 03/01/24         |
| 2 | 2             | 2          | 1        | 04/01/25         | 06/01/25         |
| 3 | 3             | 3          | 2        | 01/02/24         | 02/02/24         |
| 4 | 4             | 4          | 3        | 01/03/25         | 10/03/25         |
| 5 | 5             | 5          | 4        | 01/04/24         | 04/04/24         |

## ECHIPAMENT\_DEPARTAMENT

|   | COLUMN_NAME     | DATA_TYPE    | NULLABLE | DATA_DEFAULT | COLUMN_ID | COMMENTS |
|---|-----------------|--------------|----------|--------------|-----------|----------|
| 1 | ID_ECHIPAMENT_M | NUMBER(38,0) | No       | (null)       | 1         | (null)   |
| 2 | ID_DEPARTAMENT  | NUMBER(38,0) | No       | (null)       | 2         | (null)   |

|    | ID_ECHIPAMENT_M | ID_DEPARTAMENT |
|----|-----------------|----------------|
| 1  | 1               | 1              |
| 2  | 1               | 2              |
| 3  | 2               | 2              |
| 4  | 2               | 3              |
| 5  | 3               | 3              |
| 6  | 3               | 4              |
| 7  | 4               | 4              |
| 8  | 4               | 5              |
| 9  | 5               | 1              |
| 10 | 5               | 5              |

## TRATAMENT\_ECHIPAMENT

|   | COLUMN_NAME        | DATA_TYPE         | NULLABLE | DATA_DEFAULT | COLUMN_ID | COMMENTS |
|---|--------------------|-------------------|----------|--------------|-----------|----------|
| 1 | ID_TRAITEMENT      | NUMBER(38,0)      | No       | (null)       | 1         | (null)   |
| 2 | ID_ECHIPAMENT_M    | NUMBER(38,0)      | No       | (null)       | 2         | (null)   |
| 3 | FREVENTA_UTILIZARE | VARCHAR2(10 BYTE) | Yes      | (null)       | 3         | (null)   |
| 4 | ADMINISTRARE       | VARCHAR2(50 BYTE) | Yes      | (null)       | 4         | (null)   |

|    | ID_TRAITEMENT | ID_ECHIPAMENT_M | FREVENTA_UTILIZARE | ADMINISTRARE              |
|----|---------------|-----------------|--------------------|---------------------------|
| 1  | 1             | 1               | 3/zi               | monitorizare tensiune     |
| 2  | 1             | 2               | 2/luna             | EKG                       |
| 3  | 2             | 2               | 1/luna             | scanare cerebrală         |
| 4  | 3             | 3               | 1/sapt             | test alergii              |
| 5  | 4             | 4               | 3/zi               | chimioterapie             |
| 6  | 4             | 5               | 2/luna             | radioterapie              |
| 7  | 5             | 5               | 1/zi               | nebulizare                |
| 8  | 5             | 1               | 4/zi               | supraveghere respiratorie |
| 9  | 3             | 1               | 1/sapt             | pulsoximetru              |
| 10 | 2             | 1               | 1/luna             | EEG                       |

## TRATAMENT\_MEDICAMENT

|   | COLUMN_NAME   | DATA_TYPE         | NULLABLE | DATA_DEFAULT | COLUMN_ID | COMMENTS |
|---|---------------|-------------------|----------|--------------|-----------|----------|
| 1 | ID_TRAITEMENT | NUMBER(38,0)      | No       | (null)       | 1         | (null)   |
| 2 | ID_MEDICAMENT | NUMBER(38,0)      | No       | (null)       | 2         | (null)   |
| 3 | DOZA          | VARCHAR2(10 BYTE) | Yes      | (null)       | 3         | (null)   |
| 4 | ADMINISTRARE  | VARCHAR2(50 BYTE) | Yes      | (null)       | 4         | (null)   |

|    | ID_TRAITEMENT | ID_MEDICAMENT | DOZA | ADMINISTRARE   |
|----|---------------|---------------|------|----------------|
| 1  | 1             | 1             | 10mg | 1/zi dimineata |
| 2  | 1             | 2             | 5mg  | 2/zi           |
| 3  | 2             | 3             | 50mg | la nevoie      |
| 4  | 2             | 4             | 25mg | 1/zi seara     |
| 5  | 3             | 5             | 15mg | 1/zi           |
| 6  | 3             | 1             | 5mg  | dimineata      |
| 7  | 4             | 2             | 10mg | injectabil     |
| 8  | 4             | 3             | 10mg | oral           |
| 9  | 5             | 5             | 20mg | 1/zi           |
| 10 | 5             | 4             | 30mg | 2/zi           |

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

ce vor utiliza, în ansamblul lor, următoarele elemente:

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

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

Cererea 1: Afisati numele si calificarea doctorilor si nr pacientilor tratati de acestia in 2024. Pentru fiecare doctor afisati si media lungimii observatiilor medicale. Ordonati dupa nr de pacienti descrescator, iar in caz de egalitate, dupa numele doctorului.

```

select d.nume as departament,
(
    select count(distinct t.id_pacient)
    from tratament t
    join pacient pac on t.id_pacient = pac.id_pacient
    join doctor doc on t.id_doctor = doc.id_doctor
    join programare p on t.id_pacient = p.id_pacient
    where doc.id_departament = d.id_departament
        and extract(year from p.data_internare) = 2024
) as nr_pacienti_2024,
(
    select round(avg(length(t.observatii_medicale)), 2)
    from tratament t
    join pacient pac on t.id_pacient = pac.id_pacient
    join programare p on pac.id_pacient = p.id_pacient
    join doctor doc on t.id_doctor = doc.id_doctor
    where doc.id_departament = d.id_departament
        and extract(year from p.data_internare) = 2024
) as medie_lungime_observatii
from departament d
order by departament;

```

Am folosit:

- subcereri sincronizate (join 3+ table)
- grupare de date
- count, avg
- length, to\_char
- ordonare

```

SQL DEVELOPER ... 10-11.sql 12-16.sql
CONNECTIONS
  Grupa 143 Tables
  Views
  Duality Views
  Indexes
  Packages
  Procedures
  Functions
  MLE Environments
  JavaScript Modules
SQL SNIPPETS
  Aggregate Functions
  Analytic Functions
  Character Functions
  Conversion Functions
  Date Formats
  Date/Time Functions
  Number Formats
  Numeric Functions
  Optimizer Hints
  PL/SQL Programming Techni...
REPORTS
  Database Administration
  About Your Database

```

```

2 --Afișați, pentru fiecare departament, numărul de pacienți internați în 2024 și media lungimii observațiilor medicale asociate tratamentei
3 --- ****
4 --- subcereri sincronizate (join 3+ table)
5 --- grupare de date
6 --- count, avg
7 --- length, to_char
8 --- ordonare
9
10 select d.nume as departament,
11 (
12     select count(distinct t.id_pacient)
13     from tratament t
14     join pacient pac on t.id_pacient = pac.id_pacient
15     join doctor doc on t.id_doctor = doc.id_doctor
16     join programare p on t.id_pacient = p.id_pacient
17     where doc.id_departament = d.id_departament
18         and extract(year from p.data_internare) = 2024
19 ) as nr_pacienti_2024,
20 (
21     select round(avg(length(t.observatii_medicale)), 2)
22     from tratament t
23     join pacient pac on t.id_pacient = pac.id_pacient
24     join programare p on pac.id_pacient = p.id_pacient
25     join doctor doc on t.id_doctor = doc.id_doctor
26     where doc.id_departament = d.id_departament
27         and extract(year from p.data_internare) = 2024
28 ) as medie_lungime_observatii
29 from departament d
30 order by departament;

```

All rows fetched: 5 in 0.189 seconds

|   | DEPARTAMENT | NR_PACIENTI_2024 | MEDIE_LUNGTIME_OBSERVATII |
|---|-------------|------------------|---------------------------|
| 1 | Cardiologie | 1                | 25                        |
| 2 | Neurologie  | 0                | (null)                    |
| 3 | Oncologie   | 0                | (null)                    |
| 4 | Pediatrie   | 1                | 17                        |
| 5 | Radiologie  | 1                | 17                        |

Cererea 2: Pentru pacientii care au fost internati in saloane comune si au stat spitalizati minimum 2 zile, afisati: numele complet al pacientului, ziua saptamanii si luna internarii, numarul total de zile spitalizate, un indicator textual (weekend/weekday) in functie de ziua internarii. Se va ordona rezultatul dupa durata internarii descrescator.

```

select p.Nume || ' ' || p.Prenume as Pacient,
       to_char(sub.Data_internare, 'Day') as Zi_Intrare,
       to_char(sub.Data_internare, 'Month') as Luna_Internalare,
       (sub.Data_externare - sub.Data_internare) as Nr_Zile,
       case
           when to_char(sub.Data_internare, 'D') in ('1', '7') then 'Weekend'
           else 'Weekday'
       end as Tip_Zi
from (
    select * from programare
    where ID_Salon in (select ID_Salon from salon where Tip = 'comun')
) sub
join pacient p on p.ID_Pacient = sub.ID_Pacient
where (sub.Data_externare - sub.Data_internare) >= 2
order by Nr_Zile desc;

```

Am folosit:

- subcerere nesincronizată în from
- to\_char
- case
- ordonare

The screenshot shows the Oracle SQL Developer interface with the SQL Snippets section open. The code is identical to the one provided above, starting with a multi-line comment explaining the purpose of the query. The code then defines a subquery to filter programs for common salons, joins it with the patient table, and finally orders the results by the duration of hospitalization in descending order.

```

SQL DEVELOPER ... 10-11.sql 12-16.sql X
Users > roland > Documents > Facultate > Baze de date > 12-16.sql > Pacient_Proiectati > Programari_Filtrate_Proiectate > PROGRAMARE
-- Pentru pacienții care au fost internați în saloane comune și au stat spitalizați minimum 2 zile, afișați:
-- numele complet al pacientului,
-- ziua săptămânnii și luna internării,
-- numărul total de zile spitalizate,
-- un indicator textual (weekend/weekday) în funcție de ziua internării.
-- Se va ordona rezultatul după durata internării descrescător.
-- *****
-- subcerere nesincronizată în from
-- to_char
-- case
-- ordonare
select p.Nume || ' ' || p.Prenume as Pacient,
       to_char(sub.Data_internare, 'Day') as Zi_Intrare,
       to_char(sub.Data_internare, 'Month') as Luna_Internalare,
       (sub.Data_externare - sub.Data_internare) as Nr_Zile,
       case
           when to_char(sub.Data_internare, 'D') in ('1', '7') then 'Weekend'
           else 'Weekday'
       end as Tip_Zi
from (
    select * from programare
    where ID_Salon in (select ID_Salon from salon where Tip = 'comun')
) sub
join pacient p on p.ID_Pacient = sub.ID_Pacient
where (sub.Data_externare - sub.Data_internare) >= 2
order by Nr_Zile desc;

```

All rows fetched: 3 in 0.204 seconds

|   | PACIENT       | ZI_INTRARE | LUNA_INTERNARE | NR_ZILE | TIP_ZI  |
|---|---------------|------------|----------------|---------|---------|
| 1 | Grigore Ioana | Saturday   | March          | 9       | Weekend |
| 2 | Popa Alex     | Monday     | January        | 2       | Weekday |
| 3 | Marin Elena   | Saturday   | January        | 2       | Weekend |

Cererea 3: Afisati saloanele in care media duratei de spitalizarea pacientilor este mai mica decat media duratei de spitalizare din toate saloanele. Pentru fiecare salon afisati: ID-ul salonului, tipul salonului, media zilelor de spitalizare pentru pacienti. Rezultatele sa fie ordonate descrescator dupa media zilelor de spitalizare.

```
select s.ID_Salon, s.Tip,
       round(avg(p.Data_externare - p.Data_internare), 2) as Media_Salon
  from salon s
 join programare p on s.ID_Salon = p.ID_Salon
 group by s.ID_Salon, s.Tip
 having avg(p.Data_externare - p.Data_internare) < (
      select avg(Data_externare - Data_internare) from programare
)
 order by Media_Salon desc;
```

Am folosit:

- grupare de date
- avg
- subcerere nesincronizată în having
- ordonare

All rows fetched: 3 in 0.287 seconds

| ID_SALON | TIP    | MEDIA_SALON |
|----------|--------|-------------|
| 1        | privat | 3           |
| 2        | comun  | 2           |
| 3        | privat | 1           |

Cererea 4: Afisati numele pacientilor, istoricul medical (sau "Fara istoric" daca e NULL), varsta si categoriile de varsta (minor/adult/senior). Ordinati dupa Categoria de varsta, si in caz de egalitate, dupa numele intreg al pacientului.

```

select p.Nume || ' ' || p.Prenume as Pacient,
       nvl(p.Istoric_medical, 'Fara istoric') as Istoric,
       trunc(months_between(sysdate, p.Data_nasterii)/12) as Varsta,
       decode(
           case
               when months_between(sysdate, p.Data_nasterii)/12 < 18 then 'Minor'
               when months_between(sysdate, p.Data_nasterii)/12 between 18 and 65 then 'Adult'
               else 'Senior'
           end,
           'Minor', 'Minor - Sub 18 ani',
           'Adult', 'Adult - 18-65 ani',
           'Senior', 'Senior - Peste 65 ani'
       ) as Categorie
  from pacient p
 order by Categorie, Pacient;

```

Am folosit:

- nvl
- decode
- months\_between, sysdate, trunc
- case
- ordonare

```

SQL DEVELOPER ... 10-11.sql 12-16.sql X
CONNECTIONS Grupa 143 Tables
> Views
> Duality Views
> Indexes
> Packages
> Procedures
> Functions
> MLE Environments
> JavaScript Modules
SQL SNIPPETS Aggregate Functions
Analytic Functions
Character Functions
Conversion Functions
Date Formats
Date/Time Functions
Number Formats
Numeric Functions
Optimizer Hints
PL/SQL Programming Techni...

```

```

Users > roland > Documents > Facultate > Baze de date > 12-16.sql > Pacienti_Proiectati > Programari_Filtrate_Proiectate > PROGRAMARE
order by Media_Salon desc;
-- Afisati numele pacientilor, istoricul medical (sau "Fara istoric" daca e NULL), varsta si categoria de varsta (minor/adult/senior).
-- Ordonati dupa Categoria de varsta, si in caz de egalitate, dupa numele intreg al pacientului.
-- *****
-- nvl
-- decode
-- months_between, sysdate, trunc
-- case
-- ordonare
select p.Nume || ' ' || p.Prenume as Pacient,
       nvl(p.Istoric_medical, 'Fara istoric') as Istoric,
       trunc(months_between(sysdate, p.Data_nasterii)/12) as Varsta,
       decode(
           case
               when months_between(sysdate, p.Data_nasterii)/12 < 18 then 'Minor'
               when months_between(sysdate, p.Data_nasterii)/12 between 18 and 65 then 'Adult'
               else 'Senior'
           end,
           'Minor', 'Minor - Sub 18 ani',
           'Adult', 'Adult - 18-65 ani',
           'Senior', 'Senior - Peste 65 ani'
       ) as Categorie
  from pacient p
 order by Categorie, Pacient;

```

All rows fetched: 5 in 0.437 seconds

|   | PACIENT       | ISTORIC               | VARSTA | CATEGORIE             |
|---|---------------|-----------------------|--------|-----------------------|
| 1 | Grigore Ioana | diabet tip 2          | 41     | Adult - 18-65 ani     |
| 2 | Iacob Dan     | alergii               | 24     | Adult - 18-65 ani     |
| 3 | Marin Elena   | migrene               | 38     | Adult - 18-65 ani     |
| 4 | Radu Andreea  | astm                  | 48     | Adult - 18-65 ani     |
| 5 | Popa Alex     | hipertensiune cronica | 65     | Senior - Peste 65 ani |

Cererea 5: Afisati primii 3 doctori in functie de numarul total de echipamente medicale folosite in tratamentele lor sau, in caz de egalitate, dupa numele intreg al doctorului.

with DoctorEchipamente as (

```

select d.ID_Doctor, d.Nume || ' ' || d.Prenume as Doctor,
       count(te.ID_Echipament_m) as Nr_Echipamente

```

```

from doctor d
join tratament t on d.ID_Doctor = t.ID_Doctor
join tratament_echipament te on t.ID_Tratament = te.ID_Tratament
group by d.ID_Doctor, d.Nume, d.Prenume
)
select Doctor, Nr_Echipamente
from DoctorEchipamente
order by Nr_Echipamente desc, Doctor
fetch first 3 rows only;

```

Am folosit:

- bloc cerere with
- grupare de date
- count
- concatenare
- ordonare

```

SQL DEVELOPER ...
CONNECTIONS
  Grupa 143 ...
    Tables
    Views
    Duality Views
    Indexes
    Packages
    Procedures
    Functions
    MLE Environments
    JavaScript Modules
SQL SNIPPETS
  Aggregate Functions
  Analytic Functions
  Character Functions
  Conversion Functions
  Date Formats
  Date/Time Functions

104  -- Afisati primii 3 doctori in functie de numarul total de echipamente medicale utilizate in tratamentele lor sau, in caz de egalitate
105  --- *****
106  --- bloc cerere with
107  --- grupare de date
108  --- count
109  --- concatenare
110  --- ordonare
111  with DoctorEchipamente as (
112    select d.ID_Doctor, d.Nume || ' ' || d.Prenume as Doctor,
113      count(te.ID_Echipament_m) as Nr_Echipamente
114    from doctor d
115    join tratament t on d.ID_Doctor = t.ID_Doctor
116    join tratament_echipament te on t.ID_Tratament = te.ID_Tratament
117    group by d.ID_Doctor, d.Nume, d.Prenume
118  )
119  select Doctor, Nr_Echipamente
120  from DoctorEchipamente
121  order by Nr_Echipamente desc, Doctor
122  fetch first 3 rows only;
123

All rows fetched: 3 in 0.116 seconds

```

|   | DOCTOR           | NR_ECHIPAMENTE |
|---|------------------|----------------|
| 1 | Ciobanu Ana      | 2              |
| 2 | Dumitru Paul     | 2              |
| 3 | Georgescu Andrei | 2              |

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

```

-- a. actualizare - Modifica doza medicamentului pentru tratamentele pacienților care au fost internați
în "Cardiologie" și "Oncologie"
update tratament_medicament
set doza = '20mg'
where id_tratament in (
  select t.id_tratament
  from tratament t
  join pacient p on t.id_pacient = p.id_pacient
  join programare pr on p.id_pacient = pr.id_pacient
  join salon s on pr.id_salon = s.id_salon
  join departament d on s.id_departament = d.id_departament
  where d.nume in ('Cardiologie', 'Oncologie')
);

```

```
-- b. stergere - Elimina echipamentele care nu sunt folosite in niciun tratament
delete from echipament_medical
where id_echipament_m not in (
    select distinct id_echipament_m from tratament_echipament
);
- nu este corect deoarece nu se pot sterge echipamentele daca au asociate departamente. Se poate adauga si sa nu existe linii in echipament_departament.
```

#### ALTERNATIVA CORECTA

```
DELETE FROM echipament_medical
WHERE id_echipament_m IN (
    SELECT id_echipament_m
    FROM tratament_echipament
)
AND id_echipament_m NOT IN (
    SELECT id_echipament_m
    FROM echipament_departament
);
```

- dar rezultatul va fi “0 rows deleted”

```
-- c. actualizare - Actualizează calificarea doctorilor care tratează pacienți cu “migrene” în istoric
update doctor
set calificare = 'specialist'
where id_doctor in (
    select distinct t.id_doctor
    from tratament t
    join pacient p on t.id_pacient = p.id_pacient
    where lower(p.istoric_medical) like '%migrene%'
);
```

```

SQL DEVELOPER ... 10-11.sql 12-16.sql x
Users > roland > Documents > Facultate > Baze de date > 12-16.sql > doctor > calificare
126 -- a. actualizare - Modifica doza medicamentului pentru tratamentele pacientilor care au fost internati in "Cardiologie" si "Oncologie"
127 update tratament_medicament
128 set doza = '20mg'
129 where id_tratament in (
130     select t.id_tratament
131         from tratament t
132             join pacient p on t.id_pacient = p.id_pacient
133                 join programare pr on p.id_pacient = pr.id_pacient
134                     join salon s on pr.id_salon = s.id_salon
135                         join departament d on s.id_departament = d.id_departament
136                             where d.nume in ('Cardiologie', 'Oncologie')
137 );
138
139 -- b. stergere - Elimina echipamentele care sunt folosite (exemplu ipotetic)
140 delete from echipament_medical
141 where id_echipament_m in (
142     select distinct id_echipament_m from tratament_echipament
143 );
144
145 -- c. actualizare - Actualizeaza calificarea doctorilor care trateaza pacienti cu "migrene" in istoric
146 update doctor
147 set calificare = 'specialist'
148 where id_doctor in (
149     select distinct t.id_doctor
150         from tratament t
151             join pacient p on t.id_pacient = p.id_pacient
152                 where lower(p.istoric_medical) like '%migrene%'
153 );

```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS SCRIPT OUTPUT SQL HISTORY TASK MONITOR

6 rows updated.

5 rows deleted.

1 row updated.

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

```

create view V_Internalari_Detalii_Pacient_Departament as
select
    P.ID_Pacient,
    P.Nume as Nume_Pacient,
    P.Prenume as Prenume_Pacient,
    P.CNP,
    PR.ID_Programare,
    PR.Data_internare,
    PR.Data_externare,
    (PR.Data_externare - PR.Data_internare) as Durata_Zile_Internare,
    S.ID_Salon,
    S.Tip as Tip_Salon,
    D.ID_Departament,
    D.Nume as Nume_Departament,
    D.Etaj as Etaj_Departament
from pacient P
join programare PR on P.ID_Pacient = PR.ID_Pacient
join salon S on PR.ID_Salon = S.ID_Salon
join departament D on S.ID_Departament = D.ID_Departament;

```

SQL DEVELOPER ...

CONNECTIONS

- > Grupa 143
- < Oracle23... ▾

Tables Views Duality Views Indexes Procedures Functions MLE Environments

SQL SNIPPETS

- > Aggregate Functions
- > Analytic Functions
- > Character Functions
- > Conversion Functions
- > Date Formats
- > Date/Time Functions
- > Number Formats
- > Numeric Functions
- > Optimizer Hints
- > PL/SQL Programming Techniques

REPORTS

- > Database Administration
- > About Your Database
- > APEX
- > ASH and AWR
- > Data Dictionary
- > PLSQL
- > Security
- > Streams
- > Table

10-11.sql 12-16.sql

Users > roland > Documents > Facultate > Baze de date > 12-16.sql > ...

```

155     commit;
156
157 -- 14. Crearea unei vizualizari complexe si exemple de comenzi LMD
158 create view V_Internalari_Detalii_Pacient_Departament as
159 select
160     P.ID_Pacient,
161     P.Nume as Nume_Pacient,
162     P.Prenume as Prenume_Pacient,
163     P.CNP,
164     PR.ID_Programare,
165     PR.Data_internare,
166     PR.Data_externare,
167     (PR.Data_externare - PR.Data_internare) as Durata_Zile_Internare,
168     S.ID_Salon,
169     S.Tip as Tip_Salon,
170     D.ID_Departament,
171     D.Nume as Nume_Departament,
172     D.Etaj as Etaj_Departament
173 from pacient P
174 join programare PR on P.ID_Pacient = PR.ID_Pacient
175 join salon S on PR.ID_Salon = S.ID_Salon
176 join departament D on S.ID_Departament = D.ID_Departament;
177
178 -- Afisare vizualizare
179 select * from V_INTERNALARI_DETALII_PACIENT_DEPARTAMENT;
180
181 -- Stergere vizualizare, daca este nevoie
182 drop view V_Internalari_Detalii_Pacient_Departament;
183
184 -- Operatie LMD permisa: Actualizarea datei de externare pentru o anumita programare.
185 update V_Internalari_Detalii_Pacient_Departament
186 set Data_externare = to_date('2024-01-06', 'YYYY-MM-DD') -- Noua data de externare
187 where ID_Programare = 1;
188
189

```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS SCRIPT OUTPUT SQL HISTORY TASK MONITOR

View V\_INTERNALARI\_DETALII\_PACIENT\_DEPARTAMENT created.

All rows fetched: 5 in 0.131 seconds

|   | ID_PACIENT | NUME_PACIENT | PRENUME_PACIENT | CNP           | ID_PROGRAMARE | DATA_INTERNARE | DATA_EXTERNARE |
|---|------------|--------------|-----------------|---------------|---------------|----------------|----------------|
| 1 | 1          | Popa         | Alex            | 1960101123456 | 1             | 01/01/24       | 03/01/24       |
| 2 | 2          | Marin        | Elena           | 2870202123456 | 2             | 04/01/25       | 06/01/25       |
| 3 | 3          | Iacob        | Dan             | 5010303123456 | 3             | 01/02/24       | 02/02/24       |
| 4 | 4          | Grigore      | Ioana           | 6840404123456 | 4             | 01/03/25       | 10/03/25       |
| 5 | 5          | Radu         | Andreea         | 2770505123456 | 5             | 01/04/24       | 04/04/24       |

ds

| DURATA_ZILE_INTERNARE | ID_SALON | TIP_SALON | ID_DEPARTAMENT | NUME_DEPARTAMENT | ETAJ_DEPARTAMENT |
|-----------------------|----------|-----------|----------------|------------------|------------------|
| 2                     | 1        | comun     | 1              | Cardiologie      | 2                |
| 2                     | 1        | comun     | 1              | Cardiologie      | 2                |
| 1                     | 2        | privat    | 2              | Neurologie       | 3                |
| 9                     | 3        | comun     | 3              | Pediatrie        | 1                |
| 3                     | 4        | privat    | 4              | Oncologie        | 4                |

- Operație LMD permisă: Actualizarea datei de externare pentru o anumită programare.

```

update V_Internalari_Detalii_Pacient_Departament
set Data_externare = to_date('2024-01-06', 'YYYY-MM-DD') -- Noua data de externare
where ID_Programare = 1;

```

CONNECTIONS

- > Grupa 143
- < Oracle23... ▾

Tables Views Duality Views Indexes Procedures Functions MLE Environments

183 drop view V\_Internalari\_Detalii\_Pacient\_Departament;
184
185 -- Operație LMD permisă: Actualizarea datei de externare pentru o anumită programare.
186 update V\_Internalari\_Detalii\_Pacient\_Departament
187 set Data\_externare = to\_date('2024-01-06', 'YYYY-MM-DD') -- Noua data de externare
188 where ID\_Programare = 1;
189

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS QUERY RESULT SCRIPT OUTPUT SQL HIST

1 row updated.

Explicație:

Coloana Data\_externare aparține direct tăbelei de bază PROGRAMARE.

Claузă where ID\_Programare = 1 identifică unic un rând în tăbela PROGRAMARE (deoarece ID\_Programare este cheia primară).

Actualizarea nu creează ambiguități cu privire la tăbelele de bază afectate.

- Operație LMD nepermisă: Încercarea de a actualiza direct coloana calculată 'Durata\_Zile\_Internare'.

Această coloană este derivată din (PR.Data\_externare - PR.Data\_internare) și nu există fizic ca o coloană stocată care poate fi modificată direct.

Această comandă va genera o eroare (<https://docs.oracle.com/error-help/db/ora-01733/01733.00000> - "virtual column not allowed here" \*Cause: An attempt was made to use an INSERT, UPDATE, or DELETE statement on an expression in a view.)

```
update V_Internari_Detalii_Pacient_Departament
set Durata_Zile_Internare = 10
where ID_Programare = 1;
```

The screenshot shows the Oracle SQL Developer interface. On the left, the 'SQL DEVELOPER' sidebar is visible with sections for CONNECTIONS, SQL SNIPPETS, and REPORTS. The main area displays a script named '12-16.sql' with the following content:

```
198 -- Operație LMD nepermisă: Încercarea de a actualiza direct coloana calculată 'Durata_Zile_Internare'.
199 -- Această coloană este derivată din (PR.Data_externare - PR.Data_internare) și nu există fizic ca o coloană stocată care
200 update V_Internari_Detalii_Pacient_Departament
201 set Durata_Zile_Internare = 10
202 where ID_Programare = 1;
203 -- Această comandă va genera o eroare (https://docs.oracle.com/error-help/db/ora-01733/01733.00000 - "virtual column not allowed here")
204
```

Below the code, the 'SCRIPT OUTPUT' tab is selected, showing the error message:

```
Error starting at line : 1 in command -
update V_Internari_Detalii_Pacient_Departament
set Durata_Zile_Internare = 10
where ID_Programare = 1
Error at Command Line : 2 Column : 5
Error report -
SQL Error: ORA-01733: virtual column not allowed here

https://docs.oracle.com/error-help/db/ora-01733/01733.00000 - "virtual column not allowed here"
*Cause: An attempt was made to use an INSERT, UPDATE, or DELETE
          statement on an expression in a view.
*Action: INSERT, UPDATE, or DELETE data in the base tables,
          instead of the view.

More Details :
https://docs.oracle.com/error-help/db/ora-01733/
```

15. Formulați în limbaj natural și implementați în SQL: o cerere ce utilizează operația outer-join pe minimum 4 tăble, o cerere ce utilizează operația division și o cerere care implementează analiza top-n.

**Observație:** Cele 3 cereri sunt diferite de cererile de la exercițiul 12.

-- a. outer-join --

Cerință în limbaj natural:

"Afipați ID-ul, numele și prenumele fiecărui pacient, împreună cu ID-ul programării, data internării și data externării (dacă pacientul are programări), tipul salonului asociat programării (dacă există salon valid pentru programare) și numele departamentului în care se află salonul (dacă există departament

valid pentru salon). Lista trebuie să includă toți pacienții, indiferent dacă au sau nu programări înregistrate sau dacă detaliile despre salon/departament sunt complete pentru programările existente. Rezultatele vor fi ordonate după numele și prenumele pacientului, apoi după data internării."

```

select
    p.ID_Pacient,
    p.Nume as Nume_Pacient,
    p.Prenume as Prenume_Pacient,
    pr.ID_Programare,
    pr.Data_internare,
    pr.Data_externare,
    s.Tip as Tip_Salon,
    d.Nume as Nume_Departament
from PACIENT p
left outer join PROGRAMARE pr on p.ID_Pacient = pr.ID_Pacient
left outer join SALON s on pr.ID_Salon = s.ID_Salon
left outer join DEPARTAMENT d on s.ID_Departament = d.ID_Departament
order by p.Nume, p.Prenume, pr.Data_internare;

```

```

SQL DEVELOPER ... 10-11.sql 12-16.sql X
Users > roland > Documents > Facultate > Baze de date > 12-16.sql > PACIENT > p
206
207
208 -- 15. outer-join, division, top-n
209 -- a. outer-join --
210 -- "Afipați ID-ul, numele și prenumele fiecărui pacient, impreună cu ID-ul programării, data internării și data externării (dacă pacientul este înregistrat) și tipul salونului asociat programării (dacă există salon valid pentru programare) și numele departamentului în care se află salónul (dacă există)."
211 -- Lista trebuie să includă toți pacienții, indiferent dacă au sau nu programări înregistrate sau dacă detaliile despre salon/departament sunt complete pentru programările existente.
212 -- Rezultatele vor fi ordonate după numele și prenumele pacientului, apoi după data internării."
213
214 select
215     p.ID_Pacient,
216     p.Nume as Nume_Pacient,
217     p.Prenume as Prenume_Pacient,
218     pr.ID_Programare,
219     pr.Data_internare,
220     pr.Data_externare,
221     s.Tip as Tip_Salon,
222     d.Nume as Nume_Departament
223 from PACIENT p
224 left outer join PROGRAMARE pr on p.ID_Pacient = pr.ID_Pacient
225 left outer join SALON s on pr.ID_Salon = s.ID_Salon
226 left outer join DEPARTAMENT d on s.ID_Departament = d.ID_Departament
227 order by p.Nume, p.Prenume, pr.Data_internare;
228
229 -- b. division --

```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS QUERY RESULT SCRIPT OUTPUT SQL HISTORY TASK MONITOR

All rows fetched: 5 in 0.178 seconds

| ID_PACIENT | NUME_PACIENT | PRENUME_PACIENT | ID_PROGRAMARE | DATA_INTERNARE | DATA_EXTERNARE | TIP_SALON | NUME_DEPARTAMENT |             |
|------------|--------------|-----------------|---------------|----------------|----------------|-----------|------------------|-------------|
| 1          | 4            | Grigore         | Ioana         |                | 4 01/03/25     | 10/03/25  | comun            | Pediatrie   |
| 2          | 3            | Iacob           | Dan           |                | 3 01/02/24     | 02/02/24  | privat           | Neurologie  |
| 3          | 2            | Marin           | Elena         |                | 2 04/01/25     | 06/01/25  | comun            | Cardiologie |
| 4          | 1            | Popa            | Alex          |                | 1 01/01/24     | 03/01/24  | comun            | Cardiologie |
| 5          | 5            | Radu            | Andreea       |                | 5 01/04/24     | 04/04/24  | privat           | Oncologie   |

-- b. division --

Cerință în limbaj natural:

"Afipați pacienții (ID, Nume, Prenume) care au primit, în cadrul tratamentelor lor, toate medicamentele distincte care au fost utilizate cel puțin o dată în tratamentele efectuate de doctorii din departamentul 'Cardiologie'."

```

with MedicamenteSpecificeDepartament as (
    select distinct tm.ID_Medicament
    from TRATAMENT t
    join DOCTOR doc on t.ID_Doctor = doc.ID_Doctor
    join DEPARTAMENT dep on doc.ID_Departament = dep.ID_Departament
)

```

```

join TRATAMENT_MEDICAMENT tm on t.ID_Tratament = tm.ID_Tratament
where dep.nume = 'Cardiologie'
),
NumarTotalMedicamenteSpecifice as (
    select count(*) as TotalMedicamente from MedicamenteSpecificeDepartament
)
select
    p.ID_Pacient,
    p.Nume as Nume_Pacient,
    p.Prenume as Prenume_Pacient
from PACIENT p
join TRATAMENT t on p.ID_Pacient = t.ID_Pacient
join TRATAMENT_MEDICAMENT tm on t.ID_Tratament = tm.ID_Tratament
where tm.ID_Medicament in (select ID_Medicament from MedicamenteSpecificeDepartament)
group by p.ID_Pacient, p.Nume, p.Prenume
having
    count(distinct tm.ID_Medicament) = (select TotalMedicamente from
NumarTotalMedicamenteSpecifice)
    and (select TotalMedicamente from NumarTotalMedicamenteSpecifice) > 0;

```

```

SQL DEVELOPER ... 10-11.sql 12-16.sql X
Users > roland > Documents > Facultate > Baze de date > 12-16.sql > MedicamenteSpecificeDepartament > TRATAMENT_MEDICAMENT > tm
order by p.Nume, p.Prenume, p.data_internare;

-- b. division --
-- Afisati pacientii (ID, Nume, Prenume) care au primit, in cadrul tratamentelor lor, toate medicamentele distincte care au fost utilizate
with MedicamenteSpecificeDepartament as (
    select distinct tm.ID_Medicament
    from TRATAMENT t
    join DOCTOR doc on t.ID_Doctor = doc.ID_Doctor
    join DEPARTAMENT dep on doc.ID_Departament = dep.ID_Departament
    join TRATAMENT_MEDICAMENT tm on t.ID_Tratament = tm.ID_Tratament
    where dep.nume = 'Cardiologie'
),
NumarTotalMedicamenteSpecifice as (
    select count(*) as TotalMedicamente from MedicamenteSpecificeDepartament
)
select
    p.ID_Pacient,
    p.Nume as Nume_Pacient,
    p.Prenume as Prenume_Pacient
from PACIENT p
join TRATAMENT t on p.ID_Pacient = t.ID_Pacient
join TRATAMENT_MEDICAMENT tm on t.ID_Tratament = tm.ID_Tratament
where tm.ID_Medicament in (select ID_Medicament from MedicamenteSpecificeDepartament)
group by p.ID_Pacient, p.Nume, p.Prenume
having
    count(distinct tm.ID_Medicament) = (select TotalMedicamente from NumarTotalMedicamenteSpecifice)
    and (select TotalMedicamente from NumarTotalMedicamenteSpecifice) > 0;

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS QUERY RESULT SCRIPT OUTPUT SQL HISTORY TASK MONITOR
All rows fetched: 1 in 0.138 seconds

```

|   | ID_PACIENT | NUME_PACIENT | PRENUME_PACIENT |
|---|------------|--------------|-----------------|
| 1 | 1          | Popa         | Alex            |

-- c. top-n --

Cerință în limbaj natural:

"Afisați numele și durata medie de spitalizare (în zile, rotunjită la două zecimale) pentru primele 3 departamente care au cea mai mare durată medie de spitalizare a pacienților în saloanele respectiveelor departamente. Se vor lua în considerare doar programările finalize (cu dată de internare și externare valide). Ordonarea se va face descrescător după durata medie."

with DurataMedieSpitalizarePerDepartament as (

select

d.ID\_Departament,

```

d.Nume as Nume_Departament,
avg(pr.Data_externare - pr.Data_internare) as Medie_Zile_Spitalizare
from PROGRAMARE pr
join SALON s on pr.ID_Salon = s.ID_Salon
join DEPARTAMENT d on s.ID_Departament = d.ID_Departament
where
    pr.Data_externare is not null
    and pr.Data_internare is not null
    and pr.Data_externare >= pr.Data_internare
group by d.ID_Departament, d.Nume
)
select
    Nume_Departament,
    round(Medie_Zile_Spitalizare, 2) as Medie_Zile_Spitalizare_Top
from DurataMedieSpitalizarePerDepartament
order by Medie_Zile_Spitalizare desc
fetch first 3 rows only;

```

The screenshot shows the Oracle SQL Developer interface. In the left sidebar, there are sections for CONNECTIONS, SQL SNIPPETS, and REPORTS. The main area displays two SQL files: 10-11.sql and 12-16.sql. The 12-16.sql file contains the query provided above. Below the code, the QUERY RESULT tab is selected, showing the output:

| NUME_DEPARTAMENT | MEDIE_ZILE_SPITALIZARE_TOP |
|------------------|----------------------------|
| Pediatrie        | 9                          |
| Oncologie        | 3                          |
| Cardiologie      | 2                          |

## 16. La alegere

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

sau

b) Prezentarea planului de execuție a unei cereri complexe, optimizare/compare plan alternativ folosind hint-uri și obiecte specifice optimizării cererilor (spre exemplu indexi).

Rezolvare a)

(Precizare: arborii vor fi introdusi cu dimensiune mai mica in document pentru a putea fi vizualizati corect, inclusi de asemenea in fisierul text 12-17.txt)

Cerinta in limbaj natural:

"Afisati numele si prenumele pacientilor, impreuna cu numele departamentului in care au fost internati, pentru toti pacientii care au fost internati in anul 2024 in saloane de tip 'privat'."

- Cererea initiala - neoptimizata

```
select P.Nume, P.Prenume, D.Nume as Nume_Departament
from PACIENT P
join PROGRAMARE PR on P.ID_Pacient = PR.ID_Pacient
join SALON S on PR.ID_Salon = S.ID_Salon
join DEPARTAMENT D on S.ID_Departament = D.ID_Departament
where extract(year from PR.Data_internare) = 2024 and S.Tip = 'privat';
```

The screenshot shows the Oracle SQL Developer interface. On the left, there's a tree view of connections and snippets. The main area has tabs for PROBLEMS, OUTPUT, DEBUG CONSOLE, TERMINAL, PORTS, QUERY RESULT (which is selected), SCRIPT OUTPUT, SQL HISTORY, and TASK MONITOR. The QUERY RESULT tab displays the SQL code and the results of the query. The results table has columns: NUME, PRENUME, and NUME\_Departament. Two rows are shown:

|   | NUME  | PRENUME | NUME_Departament |
|---|-------|---------|------------------|
| 1 | Iacob | Dan     | Neurologie       |
| 2 | Radu  | Andreea | Oncologie        |

Tabele implicate si atribute relevante:

PACIENT (ID\_Pacient, Nume, Prenume)

PROGRAMARE (ID\_Programare, ID\_Pacient, ID\_Salon, Data\_internare)

SALON (ID\_Salon, ID\_Departament, Tip)

DEPARTAMENT (ID\_Departament, Nume AS Nume\_Departament)

Expresie algebraica initiala - neoptimizata:

**Folosim  $\pi$  pentru proiecție,  $\sigma$  pentru selecție,  $\bowtie$  pentru join**

$$\pi_{P.Nume, P.Prenume, D.Nume\_Departament} (\sigma_{extract(year from PR.Data_internare) = 2024 \text{ and } S.Tip = 'privat'} ((PACIENT P \bowtie_{P.ID_Pacient = PR.ID_Pacient} PROGRAMARE PR) \bowtie_{PR.ID_Salon = S.ID_Salon} SALON S) \bowtie_{S.ID_Departament = D.ID_Departament} DEPARTAMENT D))$$

Arbore algebraic - neoptimizat:

Joinurile combină toate atributele tabelelor implicate.

Selecția filtrează rândurile.

Proiecția finală alege coloanele P.Nume, P.Prenume (din PACIENT) și D.Nume (din DEPARTAMENT, redenumit Nume\_Departament).

```

 $\pi_{P.Nume, P.Prenume, D.Nume\_Departament}$ 
|
|<- Atribute disponibile: toate din P, PR, S, D care satisfac condițiile de join și selecție

```

```

|
σ <sub>extract(year from PR.Data_internare) = 2024 and S.Tip = 'privat'</sub>
|
| <- Atribute disponibile: toate din P, PR, S, D care satisfac condițiile de join
|
&<sub>S.ID_Departament = D.ID_Departament</sub>
/
/
/
&<sub>PR.ID_Salon = S.ID_Salon</sub> DEPARTAMENT D (ID_Departament, Nume)
/
/
/
&<sub>P.ID_Pacient = PR.ID_Pacient</sub> SALON S (ID_Salon, ID_Departament, Tip)
/
/
/
PACIENT P (ID_Pacient, Nume, Prenume) PROGRAMARE PR (ID_Programare, ID_Pacient, ID_Salon, Data_internare)

```

- Cererea optimizată

```

with Pacienti_Proiectati as (
    select ID_Pacient, Nume, Prenume
    from PACIENT
),
Programari_Filtrate_Proiectate as (
    select ID_Pacient, ID_Salon
    from PROGRAMARE
    where extract(year from Data_internare) = 2024
),
Saloane_Filtrate_Proiectate as (
    select ID_Salon, ID_Departament
    from SALON
    where Tip = 'privat'
),
Departamente_Proiectate as (
    select ID_Departament, Nume as Nume_Departament
    from DEPARTAMENT
)
select
    PP.Nume,
    PP.Prenume,
    DP.Nume_Departament
from Pacienti_Proiectati PP
join Programari_Filtrate_Proiectate PFP ON PP.ID_Pacient = PFP.ID_Pacient
join Saloane_Filtrate_Proiectate SFP ON PFP.ID_Salon = SFP.ID_Salon
join Departamente_Proiectate DP ON SFP.ID_Departament = DP.ID_Departament;

```

The screenshot shows the Oracle SQL Developer interface. On the left, the 'CONNECTIONS' tree shows 'Grupa 143' and 'Oracle 23cFree'. Below it, the 'SQL SNIPPETS' tree includes categories like Aggregate Functions, Analytic Functions, Character Functions, etc. The 'REPORTS' tree includes Database Administration, APEX, ASH and AWR, Data Dictionary, PLSQL, Security, and Streams. The main area displays a query editor with the following SQL code:

```

332 -- Cererea optimizata
333 with Pacienti_Proiectati as (
334     select ID_Pacient, Nume, Prenume
335     from PACIENT
336 ),
337 Programari_Filtrate_Proiectate as (
338     select ID_Pacient, ID_Salon
339     from PROGRAMARE
340     where extract(year from Data_internare) = 2024
341 ),
342 Saloane_Filtrate_Proiectate as (
343     select ID_Salon, ID_Departament
344     from SALON
345     where Tip = 'privat'
346 ),
347 Departamente_Proiectate as (
348     select ID_Departament, Nume as Nume_Departament
349     from DEPARTAMENT
350 )
351 select
352     PP.Nume,
353     PP.Prenume,
354     DP.Nume_Departament
355     from Pacienti_Proiectati PP
356     join Programari_Filtrate_Proiectate PFP ON PP.ID_Pacient = PFP.ID_Pacient
357     join Saloane_Filtrate_Proiectate SFP ON PFP.ID_Salon = SFP.ID_Salon
358     join Departamente_Proiectate DP ON SFP.ID_Departament = DP.ID_Departament;
359

```

The results of the query are shown in a table:

|   | NUME  | PRENUME | NUME_DEPARTAMENT |
|---|-------|---------|------------------|
| 1 | Iacob | Dan     | Neurologie       |
| 2 | Radu  | Andreea | Oncologie        |

Below the table, the message 'All rows fetched: 2 in 0.042 seconds' is displayed.

Expresie algebrică optimizată:

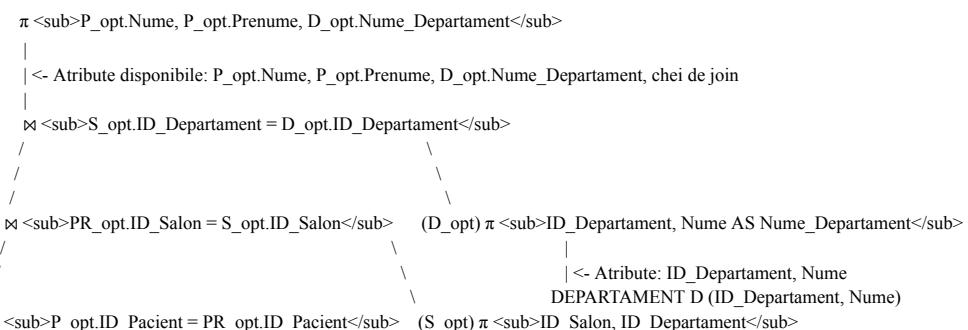
Fie:

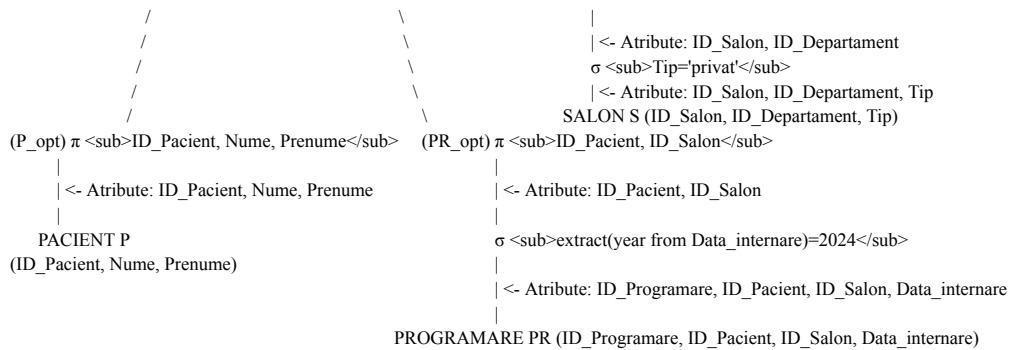
- $\text{PACIENT\_opt} = \pi_{\text{ID\_Pacient}, \text{Nume}, \text{Prenume}}(\text{PACIENT P})$
- $\text{PROGRAMARE\_opt} = \pi_{\text{ID\_Pacient}, \text{ID\_Salon}}(\sigma_{\text{EXTRACT(YEAR FROM Data\_internare)} = 2024}(\text{PROGRAMARE PR}))$
- $\text{SALON\_opt} = \pi_{\text{ID\_Salon}, \text{ID\_Departament}}(\sigma_{\text{Tip} = 'privat'}(\text{SALON S}))$
- $\text{DEPARTAMENT\_opt} = \pi_{\text{ID\_Departament}, \text{Nume AS Nume\_Departament}}(\text{DEPARTAMENT D})$

**Expresia devine:**

$$\begin{aligned}
 & \pi_{\text{Nume}, \text{Prenume}}(\text{PACIENT\_opt}), \\
 & \text{DEPARTAMENT\_opt} \text{.Nume\_Departament} (( (\text{PACIENT\_opt} \\
 & \bowtie_{\text{ID\_Pacient}} \text{PROGRAMARE\_opt}.ID\_Pacient \text{PROGRAMARE\_opt} \\
 & \bowtie_{\text{ID\_Salon}} \text{SALON\_opt}.ID\_Salon \text{SALON\_opt}) \\
 & \bowtie_{\text{ID\_Departament}} \text{DEPARTAMENT\_opt}.ID\_Departament \text{DEPARTAMENT\_opt})
 \end{aligned}$$

Arbore algebric optimizat:





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

- BCNF (Forma Normală Boyce-Codd): Pentru fiecare dependență funcțională (DF) non-trivială  $X \rightarrow Y$ , X trebuie să fie o supercheie.
- FN4: Trebuie să fie în BCNF și să nu aibă dependențe multivaloare (DMV) non-triviale. O DMV  $X \rightarrow\rightarrow Y$  înseamnă că pentru un X dat, există un set de valori Y, iar acest set este independent de alte atrbute din tabel.
- FN5: Trebuie să fie în FN4 și fiecare dependență de join (DJ) trebuie să fie implicată de cheile candidat. Aceasta înseamnă că tabelul nu poate fi descompus fără pierderi în tabele mai mici decât dacă acele descompuneri se bazează pe chei candidat.

#### **Exemplu de Încălcare BCNF (și cum este deja rezolvată în proiect):**

Presupunem că, în loc de a avea tabele separate SALON și DEPARTAMENT, am avea un singur tabel SALON\_DETALIAT structurat astfel:

SALON\_DETALIAT (ID\_Salon, Tip\_Salon, ID\_Departament, Nume\_Departament, Etaj\_Departament), unde cheia primară ar fi ID\_Salon.

Există următoarele dependențe funcționale:

ID\_Salon  $\rightarrow$  Tip\_Salon, ID\_Departament, Nume\_Departament, Etaj\_Departament (deoarece

ID\_Salon este cheia)

ID\_Departament  $\rightarrow$  Nume\_Departament, Etaj\_Departament (un ID de departament determină numele și etajul său)

Problema este că ID\_Departament nu este o supercheie a tabelului SALON\_DETALIAT. Aceasta este o încălcare a BCNF.

Anomalii potențiale:

Anomalie de actualizare: Dacă numele unui departament se schimbă (de ex., 'Cardiologie' devine 'Cardiologie Generală'), trebuie să actualizați fiecare înregistrare din SALON\_DETALIAT pentru acel departament.

Anomalie de inserare: Nu puteți adăuga un departament nou decât dacă are deja un salon asociat.

Anomalie de ștergere: Dacă ștergeți un salon dintr-un departament, pierdeți și informațiile despre departament (nume, etaj).

Rezolvare (aşa cum este în proiect): Se descompune SALON\_DETALIAT în două tabele:

SALON (ID\_Salon, ID\_Departament, Tip)

DEPARTAMENT (ID\_Departament, Nume, Etaj)

Această descompunere respectă BCNF.

#### **Exemplu de Încălcare a FN4:**

Presupunem că dorim să stocăm ce medicamente poate administra un doctor și ce echipamente medicale știe să folosească un doctor, și aceste două attribute sunt independente. S-ar putea crea un tabel:

DOCTOR\_ABILITATI (ID\_Doctor, ID\_Medicament\_Administrabil, ID\_Echipament\_Utilizabil), unde cheia primară compusă este (ID\_Doctor, ID\_Medicament\_Administrabil, ID\_Echipament\_Utilizabil).

Să zicem că Doctorul D1 poate administra Medicamentul M1 și M2, și știe să folosească Echipamentul E1 și E2. Dacă M1 și E1 nu au nicio legătură specifică între ele pentru doctorul D1, alta decât faptul că D1 le cunoaște pe amândouă, atunci tabelul ar arăta aşa pentru a reprezenta toate combinațiile: (D1, M1, E1) (D1, M1, E2) (D1, M2, E1) (D1, M2, E2)

Avem dependențe multivaloare (DMV):

ID\_Doctor →→ ID\_Medicament\_Administrabil (pentru un doctor, există un set de medicamente pe care le poate administra, independent de echipamente)

ID\_Doctor →→ ID\_Echipament\_Utilizabil (pentru un doctor, există un set de echipamente pe care le poate utiliza, independent de medicamente)

Niciuna dintre aceste DMV-uri nu este trivială și nici ID\_Doctor nu este o supercheie pentru întregul set de attribute din tabel (deoarece cheia este (ID\_Doctor, ID\_Medicament\_Administrabil, ID\_Echipament\_Utilizabil)). Aceasta este o încălcare a FN4.

Anomalii potențiale:

Redundanță și dificultăți la actualizare. Pentru a adăuga un nou echipament pe care D1 îl poate folosi (E3), trebuie să adăugați multiple rânduri: (D1, M1, E3), (D1, M2, E3).

Rezolvare: Se descompune DOCTOR\_ABILITATI în două tabele:

DOCTOR\_MEDICAMENTE (ID\_Doctor, ID\_Medicament\_Administrabil) (Cheie: (ID\_Doctor, ID\_Medicament\_Administrabil))

DOCTOR\_ECHIPAMENTE (ID\_Doctor, ID\_Echipament\_Utilizabil) (Cheie: (ID\_Doctor, ID\_Echipament\_Utilizabil))

Acum, pentru a adăuga E3 pentru D1, adăugați un singur rând în DOCTOR\_ECHIPAMENTE: (D1, E3).

#### **Exemplu de Încălcare a FN5:**

FN5 se ocupă de dependențele de join care nu sunt implicate de cheile candidat. O încălcare înseamnă că un tabel poate fi descompus fără pierderi în trei sau mai multe tabele, iar această descompunere este necesară pentru a evita redundanță și anomaliiile, dar nu poate fi descompus fără pierderi în doar două tabele.

Considerăm o situație în care un anumit TRATAMENT necesită o combinație specifică de MEDICAMENT și ECHIPAMENT\_MEDICAL, și aceste asocieri sunt interdependente într-un mod ciclic.

Presupunem că există tabelul:

PRESCRIPTIE\_COMPLEXA (ID\_Tratament, ID\_Medicament, ID\_Echipament\_m), unde o înregistrare (T1, M1, E1) înseamnă că tratamentul T1 implică utilizarea medicamentului M1 administrat alături de echipamentului E1.

Presupunem următoarele reguli independente care, luate împreună, formează constrângerea pentru PRESCRIPTIE\_COMPLEXA:

Anumite tratamente necesită anumite medicamente:

TRATAMENT\_FOLOSESTE\_MEDICAMENT (ID\_Tratament, ID\_Medicament) (similar cu tabelul existent TRATAMENT\_MEDICAMENT)

Anumite tratamente necesită utilizarea anumitor echipamente:

TRATAMENT\_FOLOSESTE\_ECHIPAMENT (ID\_Tratament, ID\_Echipament\_m) (similar cu tabelul existent TRATAMENT\_ECHIPAMENT)

Anumite medicamente sunt compatibile sau administrate optim alături de anumite echipamente:

MEDICAMENT\_COMPATIBIL\_ECHIPAMENT (ID\_Medicament, ID\_Echipament\_m)

Dacă tabelul PRESCRIPTIE\_COMPLEXA este populat astfel încât o înregistrare (t, m, e) există dacă și numai dacă (t,m) există în TRATAMENT\_FOLOSESTE\_MEDICAMENT, (t,e) există în TRATAMENT\_FOLOSESTE\_ECHIPAMENT, și (m,e) există în MEDICAMENT\_COMPATIBIL\_ECHIPAMENT. Dacă tabelul PRESCRIPTIE\_COMPLEXA poate fi reconstituit fără pierderi prin join-ul celor trei tabele de mai sus, dar nu prin join-ul a oricărora două dintre ele, atunci PRESCRIPTIE\_COMPLEXA încalcă FN5. Dependența de join este ((ID\_Tratament, ID\_Medicament), (ID\_Tratament, ID\_Echipament\_m), (ID\_Medicament, ID\_Echipament\_m)).

Anomalii potențiale:

Dacă nu descompunem, am putea avea dificultăți în a reprezenta faptul că un medicament este compatibil cu un echipament independent de orice tratament specific, sau că un tratament necesită un medicament chiar dacă echipamentul specific pentru acea combinație nu este încă decis/disponibil.

Rezolvare: Se descompune PRESCRIPTIE\_COMPLEXA în cele trei tabele menționate:

TRATAMENT\_FOLOSESTE\_MEDICAMENT (ID\_Tratament, ID\_Medicament)

TRATAMENT\_FOLOSESTE\_ECHIPAMENT (ID\_Tratament, ID\_Echipament\_m)

MEDICAMENT\_COMPATIBIL\_ECHIPAMENT (ID\_Medicament, ID\_Echipament\_m)

În proiect există deja primele două. Dacă a treia relație (compatibilitatea medicament-echipament independentă de tratament) ar fi o constrângere care guvernează combinațiile posibile în TRATAMENT\_MEDICAMENT și TRATAMENT\_ECHIPAMENT într-un mod ciclic, atunci ar trebui să existe și al treilea tabel pentru a fi în FN5. Cu toate acestea, tabelele TRATAMENT\_MEDICAMENT și TRATAMENT\_ECHIPAMENT sunt independente și nu formează o astfel de dependență de join ciclică care ar încalcă FN5. Ele reprezintă pur și simplu ce medicamente și ce echipamente sunt folosite pentru un tratament dat, fără o constrângere suplimentară de compatibilitate medicament-echipament care să lege aceste două tabele printr-un al treilea factor.

b. Aplicarea denormalizării, justificând necesitatea acesteia.

Considerăm o cerință frecventă: afișarea unei liste de programări ale pacienților împreună cu numele complet al pacientului și numele departamentului unde se află salonul acestuia.

În schema normalizată curentă, această interogare ar necesita join-uri între PROGRAMARE, PACIENT, SALON și DEPARTAMENT.

```

select
    PR.ID_Programare,
    PA.Nume || ' ' || PA.Prenume AS Nume_Complet_Pacient,
    D.Nume AS Nume_Departament_Salon,
    PR.Data_internare,
    PR.Data_externare
from PROGRAMARE PR
join PACIENT PA on PR.ID_Pacient = PA.ID_Pacient
join SALON S on PR.ID_Salon = S.ID_Salon
join DEPARTAMENT D on S.ID_Departament = D.ID_Departament;

```

The screenshot shows a SQL query being run in a terminal window. The code is identical to the one above. The results section shows the output of the query:

```

All rows fetched: 5 in 0.054 seconds

```

|   | ID_PROGRAMARE | NUME_COMPLET_PACIENT | NUME_DEPARTAMENT_SALON | DATA_INTERNARE | DATA_EXTERNARE |
|---|---------------|----------------------|------------------------|----------------|----------------|
| 1 | 1             | Popa Alex            | Cardiologie            | 01/01/24       | 03/01/24       |
| 2 | 2             | Marin Elena          | Cardiologie            | 04/01/25       | 06/01/25       |
| 3 | 3             | Iacob Dan            | Neurologie             | 01/02/24       | 02/02/24       |
| 4 | 4             | Grigore Ioana        | Pediatrie              | 01/03/25       | 10/03/25       |
| 5 | 5             | Radu Andreea         | Oncologie              | 01/04/24       | 04/04/24       |

**Necesitatea Denormalizării:** Dacă acest tip de interogare este executat foarte frecvent (de ex., pe un tablou de bord principal sau în rapoarte critice) și performanța sa devine un blocaj din cauza multiplelor join-uri, denormalizarea poate fi justificată. Prin adăugarea numelui pacientului și a numelui departamentului direct în tabelul PROGRAMARE, putem recupera aceste informații cu un simplu select dintr-un singur tabel, accelerând semnificativ operațiunile de citire.

#### Aplicarea Denormalizării:

Vom denormaliza tabelul PROGRAMARE adăugând două coloane noi: Nume\_Complet\_Pacient și Nume\_Departament\_Salon.

#### Consecințele acestei Denormalizări:

**Performanță de citire îmbunătățită:** Interogările care preiau liste de programări cu numele pacienților și ale departamentelor sunt mai rapide.

**Redundanță Datelor:** Numele pacienților și numele departamentelor sunt acum stocate în PROGRAMARE, precum și în PACIENT și DEPARTAMENT.

**Complexitatea Actualizării:** Dacă numele unui pacient sau numele unui departament se schimbă, actualizarea trebuie să aibă loc în tabelul original (PACIENT sau DEPARTAMENT) și în toate intrările relevante din tabelul PROGRAMARE. Acest lucru necesită o gestionare atentă pentru a menține consistența datelor.

**Stocare Crescută:** Tabelul PROGRAMARE va consuma mai mult spațiu de stocare.

## 18. Exemplificarea isolation levels

prin exemple de tranzacții care se execută în paralel în condiții de concurență, evidențiind efectele diferitelor niveluri de izolare asupra concurenței și integrității datelor.

Fenomene de Concurență:

- Dirty Read (Citire Necuratată): O tranzacție citește date care au fost modificate de o altă tranzacție, dar care încă nu au fost comise (committed). Dacă a doua tranzacție face rollback, prima tranzacție a citit date "murdare" care nu au existat niciodată oficial în baza de date.
- Non-Repeatable Read (Citire Irepetabilă): O tranzacție citește același rând de mai multe ori și obține valori diferite de fiecare dată, deoarece o altă tranzacție a modificat și a comis acel rând între citirile primei tranzacții.
- Phantom Read (Citire Fantomă): O tranzacție re-execută o interogare care returnează un set de rânduri ce satisfac o anumită condiție de căutare și descoperă că o altă tranzacție a inserat (și a comis) rânduri noi care satisfac condiția respectivă.

### 1. READ UNCOMMITTED (Citire Necomisă)

- Ce previne: Nimic.
- Ce permite: Dirty Reads, Non-Repeatable Reads, Phantom Reads.
- Implementare Oracle: Oracle nu suportă nivelul READ UNCOMMITTED. Orice încercare de a seta acest nivel va rezulta în utilizarea nivelului READ COMMITTED.

**Exemplu Ipotic** (Dirty Read - dacă ar fi suportat): Folosim tabelul DOCTOR pentru exemplu.

Sesiunea 1:

-- Începe tranzacția

```
UPDATE DOCTOR SET Calificare = 'stagiar' WHERE ID_Doctor = 1;
```

-- Nu facem COMMIT încă

```
1  -- Sesiunea 1:  
2  -- Începe tranzacția  
3  UPDATE DOCTOR SET Calificare = 'stagiar' WHERE ID_Doctor = 1;  
4  -- Nu facem COMMIT încă  
5
```

PROBLEMS    OUTPUT    DEBUG CONSOLE    TERMINAL    PORTS    QUERY RESULT

1 row updated.

Sesiunea 2 (setată ipotetic la READ UNCOMMITTED):

-- SET TRANSACTION ISOLATION LEVEL READ UNCOMMITTED; -- Ipotic

```
SELECT Calificare FROM DOCTOR WHERE ID_Doctor = 1;
```

-- Ar returna 'stagiar' (citire necomisă)

```

1  -- Sesiunea 2 (setată ipotetic la READ UNCOMMITTED):
2  -- SET TRANSACTION ISOLATION LEVEL READ UNCOMMITTED; -- Ipotetic
3  SELECT Calificare FROM DOCTOR WHERE ID_Doctor = 1;
4  -- Ar returna 'stagiar' (citire necomisă)

```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS **QUERY RESULT** SCRIP

All rows fetched: 1 in 0.097 seconds

|   | CALIFICARE |
|---|------------|
| 1 | rezident   |

! Rezultatul cererii este incorect deoarece este un exemplu ipotetic, în realitate nu este suportat în Oracle !

Sesiunea 1 continuă:

ROLLBACK; -- Anulează modificarea

```

1  -- Sesiunea 1:
2  -- Începe tranzacția
3  UPDATE DOCTOR SET Calificare = 'stagiar' WHERE ID_Doctor = 1;
4  -- Nu facem COMMIT încă
5
6  -- Sesiunea 1 continuă:
7  ROLLBACK; -- Anulează modificarea

```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS **QUERY RESULT**

1 row updated.

Rollback complete.

**Consecință:** T2 a citit valoarea 'stagiar', care nu a fost niciodată confirmată în baza de date. Dacă T2 ar lua decizii bazate pe această valoare, ar fi incorect.

## 2. READ COMMITTED (Citire Comisă)

- Ce previne: Dirty Reads.
- Ce permite: Non-Repeatable Reads, Phantom Reads.
- Implementare Oracle: Acesta este nivelul implicit în Oracle. Fiecare instrucțiune vede doar datele care au fost comise înainte ca instrucțiunea să înceapă.

- **Exemplu** (Non-Repeatable Read - aşa cum s-a demonstrat anterior):

Sesiunea 1:

```
SET TRANSACTION ISOLATION LEVEL READ COMMITTED;
```

```
-- Citirea inițială
```

```
SELECT Telefon FROM PACIENT WHERE ID_Pacient = 1;
```

```
-- Se returnează '0700001111'
```

```
1  -- Sesiunea 1:
2  SET TRANSACTION ISOLATION LEVEL READ COMMITTED;
3  -- Citirea inițială
4  SELECT Telefon FROM PACIENT WHERE ID_Pacient = 1;
5  -- Se returnează '0700001111'
6
```

PROBLEMS    OUTPUT    DEBUG CONSOLE    TERMINAL    PORTS    QU

Transaction ISOLATION succeeded.

All rows fetched: 1 in 0.026 seconds

| TELEFON |            |
|---------|------------|
| 1       | 0700001111 |

Sesiunea 2:

```
SET TRANSACTION ISOLATION LEVEL READ COMMITTED;
```

```
UPDATE PACIENT SET Telefon = '0700009999' WHERE ID_Pacient = 1;
```

```
COMMIT;
```

```
1 -- Sesiunea 2:  
2 SET TRANSACTION ISOLATION LEVEL READ COMMITTED;  
3 UPDATE PACIENT SET Telefon = '0700009999' WHERE ID_Pacient = 1;  
4 COMMIT;
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS QUERY RESULT

Transaction ISOLATION succeeded.

1 row updated.

Commit complete.

#### Sesiunea 1:

```
-- A doua citire  
SELECT Telefon FROM PACIENT WHERE ID_Pacient = 1;  
COMMIT;
```

```
6  
7 -- Sesiunea 1:  
8 -- A doua citire  
9 SELECT Telefon FROM PACIENT WHERE ID_Pacient = 1;  
10 COMMIT;
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS QU

All rows fetched: 1 in 0.025 seconds

|   | TELEFON    |
|---|------------|
| 1 | 0700009999 |

- **Exemplu** (Phantom Read - aşa cum s-a demonstrat anterior):

#### Sesiunea 1:

```
SET TRANSACTION ISOLATION LEVEL READ COMMITTED;  
-- Numără pacienții cu numele 'Popa'
```

```
SELECT COUNT(*) FROM PACIENT WHERE Nume = 'Popa';
-- Returnează 1
```

```
1  -- Sesiunea 1:
2  SET TRANSACTION ISOLATION LEVEL READ COMMITTED;
3  -- Numără pacienții cu numele 'Popa'
4  SELECT COUNT(*) FROM PACIENT WHERE Nume = 'Popa';
5  -- Returnează 1
6
```

PROBLEMS    OUTPUT    DEBUG CONSOLE    TERMINAL    PORTS    Q

Transaction ISOLATION succeeded.

All rows fetched: 1 in 0.051 seconds

|   | COUNT(*) |   |
|---|----------|---|
| 1 | 1        | 1 |

### Sesiunea 2:

```
SET TRANSACTION ISOLATION LEVEL READ COMMITTED;
INSERT INTO PACIENT (ID_Pacient, CNP, Nume, Prenume, Telefon, Adresa, Data_nasterii,
Istoric_medical)
VALUES (seq_pacient_id.NEXTVAL, '1900101123000', 'Popa', 'Mihai', '0700008888', 'Str. Noua 1',
TO_DATE('1990-01-01', 'YYYY-MM-DD'), TO_CLOB('sanatos'));
COMMIT;
```

```
1  -- Sesiunea 2:
2  SET TRANSACTION ISOLATION LEVEL READ COMMITTED;
3  INSERT INTO PACIENT (ID_Pacient, CNP, Nume, Prenume, Telefon, Adresa, Data_nasterii, Istoric_medical)
4  VALUES (seq_pacient_id.NEXTVAL, '1900101123000', 'Popa', 'Mihai', '0700008888', 'Str. Noua 1', TO_DATE('1990-01-01', 'YYYY-MM-DD'),
5  COMMIT;
```

PROBLEMS    OUTPUT    DEBUG CONSOLE    TERMINAL    PORTS    QUERY RESULT    SCRIPT OUTPUT    SQL HISTORY    TASK MONITOR

Transaction ISOLATION succeeded.

1 row inserted.

Commit complete.

### Sesiunea 1:

```
-- Numără din nou pacienții cu numele 'Popa'
SELECT COUNT(*) FROM PACIENT WHERE Nume = 'Popa';
-- Va returna 2 (o fantomă a apărut).
COMMIT;
```

```

9  -- Sesiunea 1:
10 -- Numără din nou pacienții cu numele 'Popa'
11 SELECT COUNT(*) FROM PACIENT WHERE Nume = 'Popa';
12 -- Va returna 2 (o fantomă a apărut).
13 COMMIT; Debug Console (♂⌘Y)

```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS QUERY RESULT SCRIPT OUTPUT SQL HISTORY TASK MONITOR

All rows fetched: 1 in 0.026 seconds

|   | COUNT(*) |
|---|----------|
| 1 | 2        |

### 3. REPEATABLE READ (Citire Repetabilă)

- Ce previne: Dirty Reads, Non-Repeatable Reads.
- Ce permite: Phantom Reads.
- Implementare Oracle: Oracle nu suportă direct nivelul REPEATABLE READ aşa cum este definit în standardul SQL.
  - SET TRANSACTION READ ONLY; în Oracle oferă o garanție similară pentru operațiile de citire: toate interogările dintr-o tranzacție READ ONLY văd datele aşa cum erau la momentul începerii tranzacției (prevenind atât citirile nerepetabile, cât și fantomele pentru acea tranzacție). Totuși, o tranzacție READ ONLY nu poate efectua modificări (INSERT, UPDATE, DELETE).
  - Nivelul SERIALIZABLE din Oracle este mai strict decât REPEATABLE READ standard, deoarece previne și Phantom Reads.
- **Exemplu Ipotecic** (REPEATABLE READ standard - dacă ar fi suportat cu posibilitatea de scriere):

Sesiunea 1 - setată la REPEATABLE READ (ipotecic):

-- SET TRANSACTION ISOLATION LEVEL REPEATABLE READ; -- Ipotecic

-- Citirea 1

SELECT Calificare FROM DOCTOR WHERE ID\_Doctor = 2;

-- Se returnează 'sef sectie'

-- Citirea 2 (număr de doctori în departamentul 2)

SELECT COUNT(\*) FROM DOCTOR WHERE ID\_Departament = 2;

-- Se returnează 1

```

1  -- Sesiunea 1 – setată la REPEATABLE READ (ipotecic):
2  -- SET TRANSACTION ISOLATION LEVEL REPEATABLE READ; -- Ipotecic
3  -- Citirea 1
4  SELECT Calificare FROM DOCTOR WHERE ID_Doctor = 2;
5  -- Se returnează 'sef sectie'
6  -- Citirea 2 (număr de doctori în departamentul 2)
7  SELECT COUNT(*) FROM DOCTOR WHERE ID_Departament = 2;
8  -- Se returnează 1

```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS QUERY RESULT

All rows fetched: 1 in 0.034 seconds

|   | CALIFICARE |
|---|------------|
| 1 | sef sectie |

|                                      |
|--------------------------------------|
| All rows fetched: 1 in 0.020 seconds |
| COUNT(*)                             |

### Sesiunea 2 - READ COMMITTED:

```
UPDATE DOCTOR SET Calificare = 'profesor' WHERE ID_Doctor = 2;
INSERT INTO DOCTOR (ID_Doctor, ID_Departament, Calificare, Nume, Prenume, Telefon, Mail)
VALUES (seq_doctor_id.NEXTVAL, 2, 'asistent univ.', 'Marinescu', 'Vlad', '0700000008',
'vlad.m@email.com');
COMMIT;
```

```
1  -- Sesiunea 2 - READ COMMITTED:
2  UPDATE DOCTOR SET Calificare = 'profesor' WHERE ID_Doctor = 2;
3  INSERT INTO DOCTOR (ID_Doctor, ID_Departament, Calificare, Nume, Prenume, Telefon, Mail)
4  VALUES (seq_doctor_id.NEXTVAL, 2, 'asistent univ.', 'Marinescu', 'Vlad', '0700000008', 'vlad.m@email.com');
5  COMMIT;
```

PROBLEMS    OUTPUT    DEBUG CONSOLE    TERMINAL    PORTS    QUERY RESULT    SCRIPT OUTPUT    SQL HISTORY    TASK MONITOR

1 row updated.

1 row inserted.

Commit complete.

### Sesiunea 1:

```
-- Citirea 3
SELECT Calificare FROM DOCTOR WHERE ID_Doctor = 2;
-- Returnează 'sef sectie' (Non-Repeatable Read prevenit)
-- Citirea 4
SELECT COUNT(*) FROM DOCTOR WHERE ID_Departament = 2;
-- Returnează 2 (Phantom Read permis, deoarece Sesiunea 2 a inserat un nou doctor în departamental 2)
-- COMMIT;
```

|          |
|----------|
| COUNT(*) |
| 1        |
| 2        |

```
10  -- Sesiunea 1:
11  -- Citirea 3
12  SELECT Calificare FROM DOCTOR WHERE ID_Doctor = 2;
13  -- Returnează 'sef sectie' (Non-Repeatable Read prevenit)
14  -- Citirea 4
15  SELECT COUNT(*) FROM DOCTOR WHERE ID_Departament = 2;
16  -- Returnează 2 (Phantom Read permis, deoarece Sesiunea 2 a inserat un nou doctor în departamental 2)
17  -- COMMIT;
```

PROBLEMS    OUTPUT    DEBUG CONSOLE    TERMINAL    PORTS    QUERY RESULT    SCRIPT OUTPUT    SQL HISTORY    TASK MON

All rows fetched: 1 in 0.018 seconds

| CALIFICARE |
|------------|
| profesor   |

! Rezultatul cererii este incorrect deoarece este un exemplu ipotetic, în realitate nu este suportat în Oracle !

**Consecință** (ipotecă): T1 vede aceeași valoare pentru Calificare la doctorul 2, dar vede un număr crescut de doctori în departamentul 2.

#### 4. SERIALIZABLE (Serializabil)

- Ce previne: Dirty Reads, Non-Repeatable Reads, Phantom Reads.
- Ce permite: Nimic din cele de mai sus. Oferă cel mai înalt nivel de izolare. Tranzacțiile par să se execute una după alta (serial), nu în paralel.
- Implementare Oracle: Suportat. Oracle realizează acest lucru asigurându-se că o tranzacție serializabilă vede un snapshot consistent al bazei de date de la începutul său. Dacă tranzacția încearcă să modifice date care au fost modificate și comise de o altă tranzacție (după ce tranzacția serializabilă a început), și această modificare ar fi afectat rezultatele citirilor anterioare ale tranzacției serializabile, atunci tranzacția serializabilă va eșua cu eroarea ORA-08177: can't serialize access for this transaction. Aplicația trebuie să fie pregătită să trateze această eroare, de obicei printr-un rollback și reîncercarea tranzacției.

- **Exemplu** (Prevenirea Phantom Read și conflict la scriere - aşa cum s-a demonstrat anterior):

##### Sesiunea 1:

```
SET TRANSACTION ISOLATION LEVEL SERIALIZABLE;
-- Numără doctorii din departamentul 1
SELECT COUNT(*) FROM DOCTOR WHERE ID_Departament = 1;
-- Returnează 1
```

```
1  -- Sesiunea 1:
2  SET TRANSACTION ISOLATION LEVEL SERIALIZABLE;
3  -- Numără doctorii din departamentul 1
4  SELECT COUNT(*) FROM DOCTOR WHERE ID_Departament = 1;
5  -- Returnează 1
```

PROBLEMS    OUTPUT    DEBUG CONSOLE    TERMINAL    PORTS    QUER

Transaction ISOLATION succeeded.

| All rows fetched: 1 in 0.050 seconds |          |
|--------------------------------------|----------|
|                                      | COUNT(*) |
| 1                                    | 1        |

##### Sesiunea 2:

```
SET TRANSACTION ISOLATION LEVEL READ COMMITTED; -- Sau SERIALIZABLE
-- Modifică doctorul pe care Sesiunea 1 l-a citit/l-ar putea modifica
UPDATE DOCTOR SET Calificare = 'medic primar' WHERE ID_Doctor = 1;
-- Se inserează un nou doctor în departamentul 1
```

```

INSERT INTO DOCTOR (ID_Doctor, ID_Departament, Calificare, Nume, Prenume, Telefon, Mail)
VALUES (seq_doctor_id.NEXTVAL, 1, 'specialist', 'Georgescu', 'Mihai', '0700000009',
'mihai.g@email.com');

COMMIT;

```

```

1  -- Sesiunea 2:
2  SET TRANSACTION ISOLATION LEVEL READ COMMITTED; -- Sau SERIALIZABLE
3  -- Modifică doctorul pe care Sesiunea 1 l-a citit/l-ar putea modifica
4  UPDATE DOCTOR SET Calificare = 'medic primar' WHERE ID_Doctor = 1;
5  -- Se inserează un nou doctor în departamentul 1
6  INSERT INTO DOCTOR (ID_Doctor, ID_Departament, Calificare, Nume, Prenume, Telefon, Mail)
7  VALUES (seq_doctor_id.NEXTVAL, 1, 'specialist', 'Georgescu', 'Mihai', '0700000009', 'mihai.g@email.com');
8  COMMIT;

PROBLEMS    OUTPUT    DEBUG CONSOLE    TERMINAL    PORTS    QUERY RESULT    SCRIPT OUTPUT    SQL HISTORY    TASK MONITOR

Transaction ISOLATION succeeded.

1 row updated.

1 row inserted.

Commit complete.

```

### Sesiunea 1:

-- Se numără din nou doctorii din departamentul 1  
`SELECT COUNT(*) FROM DOCTOR WHERE ID_Departament = 1;`  
-- Va returna tot 1 (Phantom Read prevenit, Sesiunea 1 vede snapshot-ul său)

-- Încercăm să actualizăm un doctor pe care Sesiunea 1-l-ar fi putut modifica.  
`UPDATE DOCTOR SET Telefon = '0711111111' WHERE ID_Doctor = 1;`  
-- Dacă Sesiunea 1 a modificat și comis rândul pentru ID\_Doctor = 1 între timp, acest UPDATE al Sesiunii 1 va eșua cu ORA-08177.

COMMIT;

```

9  -- Sesiunea 1:
10 -- Se numără din nou doctorii din departamentul 1
11 SELECT COUNT(*) FROM DOCTOR WHERE ID_Departament = 1;
12 -- Va returna tot 1 (Phantom Read preventit, Sesiunea 1 vede snapshot-ul său)
13
14 -- Încercăm să actualizăm un doctor pe care Sesiunea 1-l-ar fi putut modifica.
15 UPDATE DOCTOR SET Telefon = '0711111111' WHERE ID_Doctor = 1;
16 -- Dacă Sesiunea 1 a modificat și comis rândul pentru ID_Doctor = 1 între timp, acest UPDATE al Sesiunii 1 va eșua cu ORA-08177.
17 COMMIT;

PROBLEMS    OUTPUT    DEBUG CONSOLE    TERMINAL    PORTS    QUERY RESULT    SCRIPT OUTPUT    SQL HISTORY    TASK MONITOR

Transaction ISOLATION succeeded.

Error starting at line : 1 in command -
UPDATE DOCTOR SET Telefon = '0711111111' WHERE ID_Doctor = 1
Error at Command Line : 1 Column : 8
Error report -
SQL Error: ORA-08177: can't serialize access for this transaction

https://docs.oracle.com/error-help/db/ora-08177/08177. 00000 - "can't serialize access for this transaction"
*Cause: Encountered data changed by an operation that occurred after
the start of this serializable transaction.
*Action: In read/write transactions, retry the intended operation or
transaction.

More Details :
https://docs.oracle.com/error-help/db/ora-08177/

Commit complete.

```

```

9  -- Sesiunea 1:
10 -- Se numără din nou doctorii din departamentul 1
11 SELECT COUNT(*) FROM DOCTOR WHERE ID_Departament = 1;
12 -- Va returna tot 1 (Phantom Read prevenit, Sesiunea 1 vede snapshot-ul său)
13
14 -- Încercăm să actualizăm un doctor pe care Sesiunea 1-ar fi putut modifica.
15 UPDATE DOCTOR SET Telefon = '0711111111' WHERE ID_Doctor = 1;
16 -- Dacă Sesiunea 1 a modificat și comis rândul pentru ID_Doctor = 1 între timp, acest UPDATE al Sesiunii 1 va eşua cu ORA-08177.
17 COMMIT;

```

PROBLEMS    OUTPUT    DEBUG CONSOLE    TERMINAL    PORTS    QUERY RESULT    SCRIPT OUTPUT    SQL HISTORY    TASK MONITOR

All rows fetched: 1 in 0.023 seconds

| COUNT(*) |   |
|----------|---|
| 1        | 1 |

**Observație:** Sesiunea 1 nu va vedea noul doctor inserat de Sesiunea 2 în SELECT COUNT(\*). Dacă Sesiunea 1 încearcă să facă UPDATE la ID\_Doctor = 1 (pe care Sesiunea 2 l-a modificat și comis), Sesiunea 1 va primi ORA-08177, deoarece modificarea Sesiunii 2 a invalidat premisa pe care se baza tranzacția serializabilă a Sesiunii 1.

19. Justificarea necesității/utilității migrării la o bază de date de tip NoSql.

Identificarea scenariilor în care utilizarea unei baze de date NoSQL este mai avantajoasă decât a unei baze de date relaționale.

- a) Prezentarea structurii baze de date de tip NoSql.
- b) Prezentarea comenzilor pentru crearea bazei de date (spre exemplu a colecțiilor într-o bază de date de tip document)
- c) Prezentarea comenzilor pentru inserarea, modificarea și ștergerea documentelor sau înregistrărilor într-o bază de date NoSQL.
- d) Exemplificarea comenzilor pentru interogarea datelor, incluzând operațiuni de filtrare și sortare..

- Flexibilitatea schemei:

În schema SQL, câmpuri precum PACIENT.Istoric\_medical, TRATAMENT.Observatii\_medicale și FACTURA.Istoric sunt de tip CLOB. Acestea sugerează stocarea unor volume mari de text. Într-o bază de date NoSQL de tip document (ex: MongoDB), aceste informații pot fi stocate într-un format mai natural și flexibil (ex: JSON/BSON), permitând variații în structura datelor fără a modifica schema la nivel de bază de date. De exemplu, istoricul medical al unui pacient ar putea conține diverse attribute sau chiar documente imbricate, diferite de la un pacient la altul.

- Scalabilitate orizontală:

Pe măsură ce volumul de date crește (număr mare de pacienți, programări, tratamente detaliate), bazele de date NoSQL sunt adesea proiectate pentru a scala orizontal (prin adăugarea mai multor servere în cluster - sharding) mai eficient și cu costuri potențial mai mici decât scalarea verticală (upgrade hardware server) tipică bazelor de date relaționale.

- Performanță pentru anumite interogări:

Prin "embedding" (încorporarea) datelor relaționate direct în documentul principal, se pot elimina operațiunile JOIN costisitoare. De exemplu, toate programările, tratamentele și facturile unui pacient ar putea fi stocate în cadrul documentului pacientului. Astfel, recuperarea profilului complet al unui pacient devine o singură operație de citire, ceea ce poate fi mult mai rapid.

- Modele de date diverse:

NoSQL oferă diverse modele de date (document, cheie-valoare, coloană-largă, graf) care pot fi mai potrivite pentru anumite tipuri de date sau cazuri de utilizare specifice decât modelul relațional rigid.

### **Scenarii Avantajoase pentru NoSQL (în contextul proiectului actual):**

- Managementul Datelor Medicale Complexe și Variabile: Dacă Istoric\_medical sau Observatii\_medicale trebuie să stocheze informații complexe cu structuri diferite (ex: rezultate analize în formate diferite, note medicale cu câmpuri optionale multiple, fișiere atașate), un model document NoSQL oferă flexibilitatea necesară.
- Aplicație focusată pe profilul pacientului: Dacă interfața principală a aplicației afișează frecvent o vedere de 360 de grade asupra pacientului (date personale, toate programările, istoricul tratamentelor, facturi), încorporarea acestor date într-un singur document Pacient în NoSQL ar duce la interogări mai rapide.
- Sistem cu volum mare de date și creștere rapidă: Dacă se anticipăază o creștere semnificativă a numărului de pacienți și a datelor asociate, scalabilitatea orizontală a NoSQL este un avantaj.
- Analiza datelor nestructurate/semi-structurate: Pentru analize pe textul liber din Observatii\_medicale sau Istoric\_medical, unele baze de date NoSQL oferă capabilități de indexare full-text și interogare mai avansate.

### **Rezolvare a) și b)**

Operația "insertMany" creează automat colecția(tabelul), nu este nevoie de creare explicită a acesteia. Totuși, putem folosi "db.createCollection("collectionName")", dar nu este strict necesar. Astfel, putem combina "CREATE TABLE" și "INSERT" într-o singura comandă în MongoDB.

Exemple pentru tabelele din proiect: DEPARTAMENT, MEDICAMENT, ECHIPAMENT\_MEDICAL, DOCTOR, PACIENT.

```
// DEPARTAMENTE Collection
// Structure: { id_departament, nume, etaj }
print("Inserting into 'departamente' collection...");
db.departamente.insertMany([
    { id_departament: 1, nume: "Cardiologie", etaj: 2 },
    { id_departament: 2, nume: "Neurologie", etaj: 3 },
    { id_departament: 3, nume: "Pediatrie", etaj: 1 },
    { id_departament: 4, nume: "Oncologie", etaj: 4 },
    { id_departament: 5, nume: "Radiologie", etaj: 0 }
]);
print(`Inserted ${db.departamente.countDocuments()} documents into departamente.`);

// -- MEDICAMENTE Collection --
// Structure: { id_medicament, nume_medicament, administrare_generala }
print("Inserting into 'medicamente' collection...");
db.medicamente.insertMany([
    { id_medicament: 1, nume_medicament: "Lisinopril", administrare_generala: "10mg, 1/zi" },
    { id_medicament: 2, nume_medicament: "Amlodipina", administrare_generala: "5mg, 1/zi" },
    { id_medicament: 3, nume_medicament: "Sumatriptan", administrare_generala: "50mg, la nevoie" },
    { id_medicament: 4, nume_medicament: "Metformin", administrare_generala: "500mg, 2/zi" },
    { id_medicament: 5, nume_medicament: "Salbutamol", administrare_generala: "Inhalator, la nevoie" }
]);
print(`Inserted ${db.medicamente.countDocuments()} documents into medicamente.`);
```

```

// -- ECHIPAMENTE_MEDICALE Collection --
// Structure: { id_echipament, nume_echipament, departament_principal_id }
print("Inserting into 'echipamente_medicale' collection...");
db.echipamente_medicale.insertMany([
  { id_echipament: 1, nume_echipament: "Monitor Tensiune Arteriala", departament_principal_id: 1 },
  { id_echipament: 2, nume_echipament: "Aparat EKG", departament_principal_id: 1 },
  { id_echipament: 3, nume_echipament: "Scanner Cerebral", departament_principal_id: 2 },
  { id_echipament: 4, nume_echipament: "Kit Test Alergii", departament_principal_id: 3 },
  { id_echipament: 5, nume_echipament: "Aparat Chimioterapie", departament_principal_id: 4 }
]);
print('Inserted ${db.echipamente_medicale.countDocuments()} documents into echipamente_medicale.');

// -- DOCTORI Collection --
// Structure: { id_doctor, departament_id, calificare, nume, prenume, telefon, mail }
print("Inserting into 'doctori' collection...");
db.doctori.insertMany([
  { id_doctor: 1, departament_id: 1, calificare: "rezident", nume: "Popescu", prenume: "Ion", telefon: "0700000001", mail: "ion.popescu@email.com" },
  { id_doctor: 2, departament_id: 2, calificare: "sef sectie", nume: "Ionescu", prenume: "Maria", telefon: "0700000002", mail: "maria.ionescu@email.com" },
  { id_doctor: 3, departament_id: 3, calificare: "asistent", nume: "Georgescu", prenume: "Andrei", telefon: "0700000003", mail: "andrei.g@email.com" },
  { id_doctor: 4, departament_id: 4, calificare: "rezident", nume: "Ciobanu", prenume: "Ana", telefon: "0700000004", mail: "ana.c@email.com" },
  { id_doctor: 5, departament_id: 5, calificare: "sef sectie", nume: "Dumitru", prenume: "Paul", telefon: "0700000005", mail: "paul.d@email.com" }
]);
print('Inserted ${db.doctori.countDocuments()} documents into doctori.');

// -- PACIENTI Collection --
// Structure: Incapsulare complexa programari, tratamente, facturi.
print("Inserting into 'pacienti' collection...");
db.pacienti.insertMany([
  {
    id_pacient: 1,
    cnp: "1960101123456",
    nume: "Popa",
    prenume: "Alex",
    telefon: "0700001111",
    adresa: "Str. Libertatii 10",
    data_nasterii: new Date("1960-01-01T00:00:00Z"),
    istoric_general: "hipertensiune cronica",
    programari: [
      { id_programare: 1, salon_id: 1, salon_tip: "comun", departament_salon_id: 1, data_internare: new Date("2024-01-01T00:00:00Z"), data_externare: new Date("2024-01-03T00:00:00Z") }
    ],
    tratamente: [

```

```

{
  id_tratament: 1,
  doctor_id: 1,
  observatii_medicele: "Hipertensiune stabilizată",
  medicamente_prescrise: [
    { medicament_id: 1, doza: "10mg", administrare_specifică: "1/zi dimineata" }
  ],
  echipamente_utilizate: [
    { echipament_id: 1, frecventa_utilizare: "3/zi", scop_utilizare: "monitorizare tensiune" }
  ]
},
],
facturi: [
  { id_factura_sql: 1, istoric_detaliat_factura: "Internare cardiologie, 2 zile", data_emitere: new Date("2024-01-03T00:00:00Z") }
]
},
{
  id_pacient: 2,
  cnp: "2870202123456",
  nume: "Marin",
  prenume: "Elena",
  telefon: "0700002222",
  adresa: "Str. Primaverii 45",
  data_nasterii: new Date("1987-02-02T00:00:00Z"),
  istoric_medical_general: "migrene",
  programari: [
    { id_programare: 2, salon_id: 1, salon_tip: "comun", departament_salon_id: 1, data_internare: new Date("2025-01-04T00:00:00Z"), data_externare: new Date("2025-01-06T00:00:00Z") }
  ],
  tratamente: [
    {
      id_tratament: 2,
      doctor_id: 2,
      observatii_medicele: "Tratament migrene",
      medicamente_prescrise: [
        { medicament_id: 3, doza: "50mg", administrare_specifică: "la nevoie" }
      ]
    }
  ],
  facturi: [
    { id_factura: 2, istoric_detaliat_factura: "Consult neurologie, 1 zi", data_emitere: new Date("2025-01-06T00:00:00Z") }
  ]
},
// Patient for update/delete examples
{
  id_pacient: 100,

```

```

cnp: "1990101010101",
nume: "Test",
prenume: "PacientDel",
telefon: "0712345678",
adresa: "Str. Test 123",
data_nasterii: new Date("1999-12-31T00:00:00Z"),
istoric_medical_general: "pentru stergere",
programari: [], tratamente: [], facturi: []
}
);

print(`Inserted ${db.pacienti.countDocuments()} documents into pacienti.`);

```

```

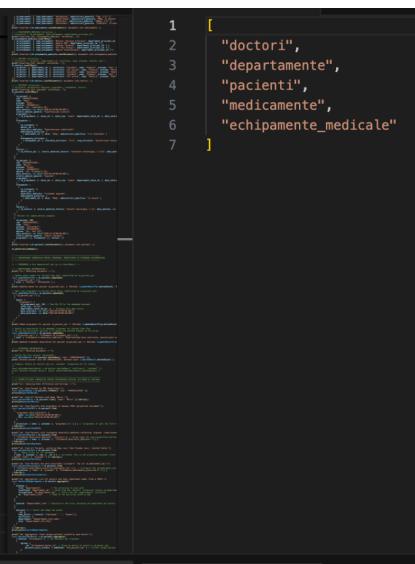
9
10 // DEPARTAMENTE Collection
11 // Structure: { id_departament, nume, etaj }
12 print("Inserting into 'departamente' collection...");
13 db.departamente.insertMany([
14   { id_departament: 1, nume: "Cardiologie", etaj: 2 },
15   { id_departament: 2, nume: "Neurologie", etaj: 3 },
16   { id_departament: 3, nume: "Pediatrie", etaj: 1 },
17   { id_departament: 4, nume: "Oncologie", etaj: 4 },
18   { id_departament: 5, nume: "Radiologie", etaj: 0 }
19 ]);
20 print(`Inserted ${db.departamente.countDocuments()} documents into departamente.`);
21
22 // -- MEDICAMENTE Collection --
23 // Structure: { id_medicament, nume_medicament, administrare_generala }
24 print("Inserting into 'medicamente' collection...");
25 db.medicamente.insertMany([
26   { id_medicament: 1, nume_medicament: "Lisinopril", administrare_generala: "10mg, 1/zi" },
27   { id_medicament: 2, nume_medicament: "Amlodipina", administrare_generala: "5mg, 1/zi" },
28   { id_medicament: 3, nume_medicament: "Sumatriptan", administrare_generala: "50mg, la nevoie" },
29   { id_medicament: 4, nume_medicament: "Metformin", administrare_generala: "500mg, 2/zi" },
30   { id_medicament: 5, nume_medicament: "Salbutamol", administrare_generala: "Inhalator, la nevoie" }
31 ]);
32 print(`Inserted ${db.medicamente.countDocuments()} documents into medicamente.`);
33
34 // -- ECHIPAMENTE_MEDICALE Collection --
35 // Structure: { id_echipament, nume_echipament, departament_principal_id }
36 print("Inserting into 'echipamente_medicale' collection...");
37 db.echipamente_medicale.insertMany([
38   { id_echipament: 1, nume_echipament: "Monitor Tensiune Arteriala", departament_principal_id: 1 },
39   { id_echipament: 2, nume_echipament: "Aparat EKG", departament_principal_id: 1 },
40   { id_echipament: 3, nume_echipament: "Scanner Cerebral", departament_principal_id: 2 },
41   { id_echipament: 4, nume_echipament: "Kit Test Alergii", departament_principal_id: 3 },
42   { id_echipament: 5, nume_echipament: "Aparat Chimioterapie", departament_principal_id: 4 }
43 ]);
44 print(`Inserted ${db.echipamente_medicale.countDocuments()} documents into echipamente_medicale.`);
45
46 // -- DOCTORI Collection --
47 // Structure: { id_doctor, departament_id, calificare, nume, prenume, telefon, mail }
48 print("Inserting into 'doctori' collection...");
49 db.doctori.insertMany([
50   { id_doctor: 1, departament_id: 1, calificare: "rezident", nume: "Popescu", prenume: "Ion", telefon: "0700000001", mail: "ion.popescu@medic.ro" },
51   { id_doctor: 2, departament_id: 2, calificare: "sef sectie", nume: "Ionescu", prenume: "Maria", telefon: "0700000002", mail: "maria.ionescu@medic.ro" },
52   { id_doctor: 3, departament_id: 3, calificare: "asistent", nume: "Georgescu", prenume: "Andrei", telefon: "0700000003", mail: "andrei.georgescu@medic.ro" },
53   { id_doctor: 4, departament_id: 4, calificare: "rezident", nume: "Ciobanu", prenume: "Ana", telefon: "0700000004", mail: "ana.ciobanu@medic.ro" },
54   { id_doctor: 5, departament_id: 5, calificare: "sef sectie", nume: "Dumitru", prenume: "Paul", telefon: "0700000005", mail: "paul.dumitru@medic.ro" }
55 ]);
56 print(`Inserted ${db.doctori.countDocuments()} documents into doctori.`);
57

```

```

58 // -- PACIENTI Collection --
59 // Structure: Incapsulare complexa: programari, tratamente, facturi.
60 print("Inserting into 'pacienti' collection...");
61 db.pacienti.insertMany([
62   {
63     id_pacient: 1,
64     cnp: "19600101123456",
65     nume: "Popa",
66     prenume: "Alex",
67     telefon: "0700001111",
68     adresa: "Str. Libertatii 10",
69     data_nasterii: new Date("1960-01-01T00:00:00Z"),
70     istoric_medical_general: "hipertensiune cronica",
71     programari: [
72       { id_programare: 1, salon_id: 1, salon_tip: "comun", departament_salon_id: 1, data_internare: new Date("2024-01-01T00:00:00Z") },
73     ],
74     tratamente: [
75       {
76         id_tratament: 1,
77         doctor_id: 1,
78         observatii_medicale: "Hipertensiune stabilizată",
79         medicamente_prescrise: [
80           { medicament_id: 1, doza: "10mg", administrare_specifică: "1/zi dimineata" }
81         ],
82         echipamente_utilizate: [
83           { echipament_id: 1, frecventa_utilizare: "3/zi", scop_utilizare: "monitorizare tensiune" }
84         ]
85       }
86     ],
87     facturi: [
88       { id_factura_sql: 1, istoric_detaliat_factura: "Internare cardiochirurgie, 2 zile", data_emitere: new Date("2024-01-03T00:00:00Z") }
89     ]
90   },
91   {
92     id_pacient: 2,
93     cnp: "2870202123456",
94     nume: "Marin",
95     prenume: "Elena",
96     telefon: "0700002222",
97     adresa: "Str. Primaverii 45",
98     data_nasterii: new Date("1987-02-02T00:00:00Z"),
99     istoric_medical_general: "migrene",
100    programari: [
101      { id_programare: 2, salon_id: 1, salon_tip: "comun", departament_salon_id: 1, data_internare: new Date("2025-01-04T00:00:00Z") },
102    ],
103    tratamente: [
104      {
105        id_tratament: 2,
106        doctor_id: 2,
107        observatii_medicale: "Tratament migrene",
108        medicamente_prescrise: [
109          { medicament_id: 3, doza: "50mg", administrare_specifică: "la nevoie" }
110        ]
111      }
112    ],
113    facturi: [
114      { id_factura: 2, istoric_detaliat_factura: "Consult neurologie, 1 zi", data_emitere: new Date("2025-01-06T00:00:00Z") }
115    ]
116  },

```



```

103     tratamente: [
108       medicamente_prescrise: [
109         { medicament_id: 3, doza: "50mg", administrare_specifica: "la nevoie" }
110       ],
111     ],
112   ],
113   facturi: [
114     { id_factura: 2, istoric_detaliat_factura: "Consult neurologie, 1 zi", data_emitere: "" }
115   ],
116 },
117 // Patient for update/delete examples
118 {
119   id_pacient: 100,
120   cnp: "1990101010101",
121   nume: "Test",
122   prenume: "PacientDel",
123   telefon: "0712345678",
124   adresa: "Str. Test 123",
125   data_nasterii: new Date("1999-12-31T00:00:00Z"),
126   istoric_medical_general: "pentru stergere",
127   programari: [], tratamente: [], facturi: []
128 }
129 );
130 print(`Inserted ${db.pacienti.countDocuments()} documents into pacienti.`);
131
132 db.getCollectionNames();
133

```

PROBLEMS    OUTPUT    DEBUG CONSOLE    TERMINAL    ...    Filter    Playground output

```

Inserting into 'departamente' collection...
Inserted 5 documents into departamente.
Inserting into 'medicamente' collection...
Inserted 5 documents into medicamente.
Inserting into 'echipamente_medcale' collection...
Inserted 5 documents into echipamente_medcale.
Inserting into 'doctori' collection...
Inserted 5 documents into doctori.
Inserting into 'pacienti' collection...
Inserted 3 documents into pacienti.

```



cluster0.78erbiz.mongodb.net c...

- > admin
- > config
- > local
- > mongodbVSCodePlaygroundDB
- > sample\_mflix
- < spitalMongoDB
  - < departamente
  - < doctori
  - < Documents 5

"683efee1ff3fdb98bc3ee952"

1 [ {

2 "\_id": {

3 "\$oid": "683efee1ff3fdb98bc3ee952"

4 },

5 "id\_departament": 1,

6 "nume": "Cardiologie",

7 "etaj": 2

8 } ]



cluster0.78erbiz.mongodb.net c...

- > admin
- > config
- > local
- > mongodbVSCodePlaygroundDB
- > sample\_mflix
- < spitalMongoDB
  - < departamente
  - < doctori
  - < Documents 5

"683efee2ff3fdb98bc3ee961"

1 [ {

2 "\_id": {

3 "\$oid": "683efee2ff3fdb98bc3ee961"

4 },

5 "id\_doctor": 1,

6 "departament\_id": 1,

7 "calificare": "rezident",

8 "nume": "Popescu",

9 "prenume": "Ion",

10 "telefon": "0700000001",

11 "mail": "ion.popescu@email.com"

12 } ]

**CONNECTIONS**

- cluster0.78erbiz.mongodb.net c...
  - admin
  - config
  - local
  - mongodbVSCodePlaygroundDB
  - sample\_mflix
  - spitalMongoDB
    - departamente
    - doctori
    - echipamente\_medicale
      - Documents 5

```

1   {
2     "_id": {
3       "$oid": "683efee2ff3fdb98bc3ee95c"
4     },
5     "id_echipament": 1,
6     "nume_echipament": "Monitor Tensiune Arteriala",
7     "departament_principal_id": 1
8   }

```

**CONNECTIONS**

- cluster0.78erbiz.mongodb.net c...
  - admin
  - config
  - local
  - mongodbVSCodePlaygroundDB
  - sample\_mflix
  - spitalMongoDB
    - departamente
    - doctori
    - echipamente\_medicale
    - medicamente
      - Documents 5

```

1   {
2     "_id": {
3       "$oid": "683efee2ff3fdb98bc3ee957"
4     },
5     "id_medicament": 1,
6     "nume_medicament": "Lisinopril",
7     "administrare_generala": "10mg, 1/zi"
8   }

```

**CONNECTIONS**

- cluster0.78erbiz.mongodb.net c...
  - admin
  - config
  - local
  - mongodbVSCodePlaygroundDB
  - sample\_mflix
  - spitalMongoDB
    - departamente
    - doctori
    - echipamente\_medicale
    - medicamente
    - pacienti
      - Documents 3

```

1   [
2     {
3       "_id": {
4         "$oid": "683efee2ff3fdb98bc3ee966"
5       },
6       "id_patient": 1,
7       "cnp": "1960101123456",
8       "nume": "Popa",
9       "prenume": "Alex",
10      "telefon": "0700001111",
11      "adresa": "Str. Libertatii 10",
12      "data_nasterii": {
13        "$date": {
14          "$numberLong": "-315619200000"
15        }
16      },
17      "istoric_medical_general": "hipertensiune cronica",
18      "programari": [
19        {
20          "id_programare": 1,
21          "salon_id": 1,
22          "salon_tip": "comun",
23          "departament_salon_id": 1,
24          "data_internare": {
25            "$date": "2024-01-01T00:00:00Z"
26          },
27          "data_externare": {
28            "$date": "2024-01-03T00:00:00Z"
29          }
30        },
31        {
32          "id_tratament": 1,
33          "doctor_id": 1,
34          "observatii_medicale": "Hipertensiune stabilizata",
35          "medicamente_prescrise": [
36            {
37

```

```

    "medicamente_prescrise": [
      {
        "medicament_id": 1,
        "doza": "10mg",
        "administrare_specifică": "1/zi dimineata"
      }
    ],
    "echipamente_utilizate": [
      {
        "echipament_id": 1,
        "frecvența_utilizare": "3/zi",
        "scop_utilizare": "monitorizare tensiune"
      }
    ]
  ],
  "facturi": [
    {
      "id_factura_sql": 1,
      "istoric_detaliat_factura": "Internare cardioologie, 2 zile",
      "data_emitere": {
        "$date": "2024-01-03T00:00:00Z"
      }
    }
  ]
}

```

### Rezolvare c)

```

// c) modificare și ștergere documente
// inserarea documentelor a fost realizată mai sus - insertMany
// stergerea documentelor: db.collectionName.drop();

print("\n--- Modifying Documents ---");
// Update nr de telefon pentru pacientul Popa Alex
const updateResultTel = db.pacienti.updateOne(
  { "id_pacient": 1 },
  { $set: { "telefon": "0711223344" } }
);
print(`Updated phone for patient id_pacient: 1. Matched: ${updateResultTel.matchedCount},
Modified: ${updateResultTel.modifiedCount}`);

// Adauga programare pacientului Marin Elena
const updateResultProg = db.pacienti.updateOne(

```

```

{ "id_pacient": 2 },
{
$push: {
  "programari": {
    id_programare: 103,
    salon_tip: "privat",
    departament_salon_id: 2,
    data_internare: new Date("2025-09-01T00:00:00Z"),
    data_externare: new Date("2025-09-03T00:00:00Z")
  }
}
);
print(`Added programare for pacient id_pacient: 2. Matched: ${updateResultProg.matchedCount},
Modified: ${updateResultProg.modifiedCount}`);

// Modifica o observatie intr-un tratament incapsulat pentru Pacient Popa Alex
const updateResultTrat = db.pacienti.updateOne(
  { "id_pacient": 1, "tratamente.id_tratament": 1 },
  { $set: { "tratamente.$.observatii_medicale": "Hipertensiune bine controlata, monitorizare continua." }
}
);
print(`Updated treatment observation for pacient id_pacient: 1. Matched:
${updateResultTrat.matchedCount}, Modified: ${updateResultTrat.modifiedCount}`);

print("\n--- Deleting Documents ---");

// Sterge pacientul test
const deleteResult = db.pacienti.deleteOne({ "cnp": "1990101010101" });
print(`Deleted patient with CNP 1990101010101. Deleted count: ${deleteResult.deletedCount}`);

// Sterge toti doctorii rezidenti
const deleteRezidentiResult = db.doctori.deleteMany({ "calificare": "rezident" });
print(`Deleted resident doctors. Count: ${deleteRezidentiResult.deletedCount}`);

```

```

136 // c) modificare și stergere documente
137 // inserarea documentelor a fost realizată mai sus - insertMany
138 // stergerea documentelor: db.collectionName.drop();
139
140 print("\n--- Modifying Documents ---");
141 // Update nr de telefon pentru pacientul Popa Alex
142 const updateResultTel = db.pacienti.updateOne(
143   { "id_pacient": 1 },
144   { $set: { "telefon": "0711223344" } }
145 );
146 print(`Updated phone for pacient id_pacient: 1. Matched: ${updateResultTel.matchedCount}, Modified: ${updateResultTel.modifiedCount}`);
147

```

```

148 // Adauga programare pacientului Marin Elena
149 const updateResultProg = db.pacienti.updateOne(
150   { "id_pacient": 2 },
151   {
152     $push: {
153       "programari": {
154         id_programare: 103,
155         salon_tip: "privat",
156         departament_salon_id: 2,
157         data_internare: new Date("2025-09-01T00:00:00Z"),
158         data_externare: new Date("2025-09-03T00:00:00Z")
159       }
160     }
161   }
162 );
163 print(`Added programare for pacient id_pacient: 2. Matched: ${updateResultProg.matchedCount}, Modified: ${updateResultProg.modifiedCount}`);
164
165 // Modifica o observatie intr-un tratament encapsulat pentru Pacient Popa Alex
166 const updateResultTrat = db.pacienti.updateOne(
167   { "id_pacient": 1, "tratamente.id_tratament": 1 },
168   { $set: { "tratamente.$observatii_medicale": "Hipertensiune bine controlata, monitorizare continua." } }
169 );
170 print(`Updated treatment observation for pacient id_pacient: 1. Matched: ${updateResultTrat.matchedCount}, Modified: ${updateResultTrat.modifiedCount}`);
171
172 print("\n--- Deleting Documents ---");
173
174 // Sterge pacientul test
175 const deleteResult = db.pacienti.deleteOne({ "cnp": "1990101010101" });
176 print(`Deleted patient with CNP 1990101010101. Deleted count: ${deleteResult.deletedCount}`);
177
178 // Sterge toti doctorii rezidenti
179 const deleteRezidentiResult = db.doctori.deleteMany({ "calificare": "rezident" });
180 print(`Deleted resident doctors. Count: ${deleteRezidentiResult.deletedCount}`);
181

```

PROBLEMS    OUTPUT    DEBUG CONSOLE    TERMINAL    PORTS    SQL HISTORY    TASK MONITOR    Filter

```

--- Modifying Documents ---
Updated phone for pacient id_pacient: 1. Matched: 1, Modified: 1
Added programare for pacient id_pacient: 2. Matched: 1, Modified: 1
Updated treatment observation for pacient id_pacient: 1. Matched: 1, Modified: 1

--- Deleting Documents ---
Deleted patient with CNP 1990101010101. Deleted count: 1
Deleted resident doctors. Count: 2

```

## Rezolvare d)

```

// d) exemple comenzi pentru interogarea datelor - filtrare si sortare
print("\n1. Find Pacient by CNP (Popa Alex) - Example of basic filtering:");
const pacientByCnp = db.pacienti.findOne({ "cnp": "1960101123456" });
printjson(pacientByCnp);

print("\n2. Find Pacienti with programari in January 2024 - Example of filtering on embedded array & using operators:");
const pacientiIan2024 = db.pacienti.find(
{
  "programari.data_internare": {
    $gte: new Date("2024-01-01T00:00:00Z"),
    $lt: new Date("2024-02-01T00:00:00Z")
  }
},
{ nume: 1, prenume: 1, "programari.$": 1, _id: 0 }
).toArray();
printjson(pacientiIan2024);

```

```

print("\n3. Find all Pacienti, sorted by Nume (asc) then Prenume (asc) - Example of sorting:");
const pacientiSortati = db.pacienti.find(
  {},
  { nome: 1, prenume: 1, cnp: 1, _id: 0 }
).sort({ "nume": 1, "prenume": 1 }).toArray();
printjson(pacientiSortati);

```

```

184 // d) exemple comenzi pentru interogarea datelor - filtrare si sortare
185 print("\n1. Find Pacient by CNP (Popa Alex) - Example of basic filtering:");
186 const pacientByCnp = db.pacienti.findOne({ "cnp": "1960101123456" });
187 printjson(pacientByCnp);
188
189 print("\n2. Find Pacienti with programari in January 2024 - Example of filtering on embedded array & using operators:");
190 const pacientiIan2024 = db.pacienti.find(
  {
    "programari.data_internare": {
      $gte: new Date("2024-01-01T00:00:00Z"),
      $lt: new Date("2024-02-01T00:00:00Z")
    },
    { nome: 1, prenume: 1, "programari.$": 1, _id: 0 }
  }.toArray();
191 printjson(pacientiIan2024);
192
193 print("\n3. Find all Pacienti, sorted by Nume (asc) then Prenume (asc) - Example of sorting:");
194 const pacientiSortati = db.pacienti.find(
  {},
  { nome: 1, prenume: 1, cnp: 1, _id: 0 }
).sort({ "nume": 1, "prenume": 1 }).toArray();
195 printjson(pacientiSortati);

```

PROBLEMS    **OUTPUT**    DEBUG CONSOLE    TERMINAL    ...    Filter    Playground output

```

1. Find Pacient by CNP (Popa Alex) - Example of basic filtering:
{
  _id: ObjectId('683f0091ae628a681a8f1d30'),
  id_pacient: 1,
  cnp: '1960101123456',
  nume: 'Popa',
  prenume: 'Alex',
  telefon: '0711223344',
  adresa: 'Str. Libertatii 10',
  data_nasterii: 1960-01-01T00:00:00.000Z,
  istoric_medical_general: 'hipertensiune cronica',
  programari: [
    {
      istoric_medical_general: 'hipertensiune cronica',
      programari: [
        {
          id_programare: 1,
          salon_id: 1,
          salon_tip: 'comun',
          departament_salon_id: 1,
          data_internare: 2024-01-01T00:00:00.000Z,
          data_externare: 2024-01-03T00:00:00.000Z
        }
      ],
      tratamente: [
        {
          id_tratament: 1,
          doctor_id: 1,
          observatii_medicale: 'Hipertensiune bine controlata, monitorizare continua.',
          medicamente_prescrise: [Array],
          echipamente_utilizate: [Array]
        }
      ],
      facturi: [
        {
          id_factura_sql: 1,
          istoric_detaliat_factura: 'Internare cardioologie, 2 zile',
          data_emitere: 2024-01-03T00:00:00.000Z
        }
      ]
    }
  ]
}

2. Find Pacienti with programari in January 2024 - Example of filtering on embedded array & using operators:
[ { nome: 'Popa', prenume: 'Alex', programari: [ [Object] ] } ]

3. Find all Pacienti, sorted by Nume (asc) then Prenume (asc) - Example of sorting:
[
  { cnp: '2870202123456', nume: 'Marin', prenume: 'Elena' },
  { cnp: '1960101123456', nume: 'Popa', prenume: 'Alex' }
]
```