FACULTATEA DE MATEMATICĂ ȘI INFORMATICĂ

SEMESTRUL I, ANUL UNIVERSITAR 2022-2023

PROIECT – SISTEME DE BAZE DE DATE

GESTIUNEA UNUI SPITAL

CARAGEA ANDA-MARIA

GRUPA 405, MASTER BAZE DE DATE ȘI TEHNOLOGII SOFTWARE

1. Prezentarea concisă a bazei de date (utilizarea ei).

Baza de date se ocupă cu gestiunea unui spital, mai exact, reprezintă un sistem medical, în care sunt stocate informații despre pacienți, personalul medical, asigurările, programările și condițiile medicale ale pacienților, cât și medicamentele administrate acestora. În cadrul bazei de date se regăsesc următoarele tabele independente:

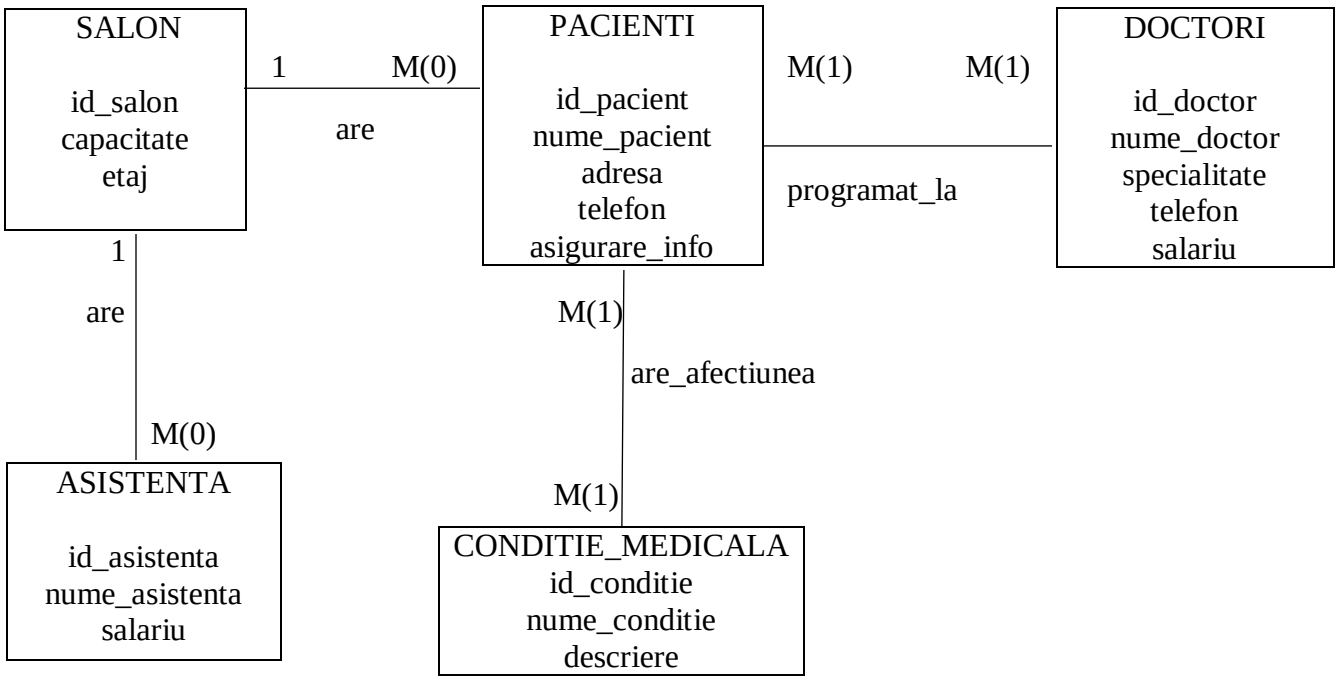
- SALON (conține informații despre salon, cum ar fi id-ul, capacitatea și etajul)
- ASISTENTA (conține informații despre asistente, cum ar fi id-ul, numele, salariul și salonul repartizat fiecăreia)
- DOCTORI (conține informații despre medici, precum id-ul, numele, specialitatea, telefon, salariul, data angajării și adresa)
- ASIGURARE (informații despre asigurări medicale: id, descriere, tip)
- PACIENTI (informații despre pacient: id, nume, adresă, telefon, asigurarea și salonul asociate fiecăruia)
- CONDITIE_MEDICALA (conține informații despre starea medicală: id, nume, descriere
- COND_MED_PREC (conține informații despre afecțiuni medicale anterioare ale unui pacient: id, descriere)
- MEDICAMENTE (informații despre medicamente, precum: id, nume, cantitate, unitate de măsură și efectele secundare)
- PLATA (informații despre plățile efectuate de către pacienți pentru consultație și tot ce presupune aceasta: id, metoda de plată, suma)

De asemenea, baza de date conține și următoarele tabele asociative:

- MEDICAMENTE_PROGRAMARE (asocierea dintre programare și medicamentul acordat în cadrul acesteia)
- CONDITIE_PACIENTI (asocierea dintre pacienți și afecțiunile sale medicale)
- PROGRAMARI (permite ca mai mulți pacienți să fie programați cu mai mulți medici și invers. De asemenea, conține informații despre programări, precum: id, data)

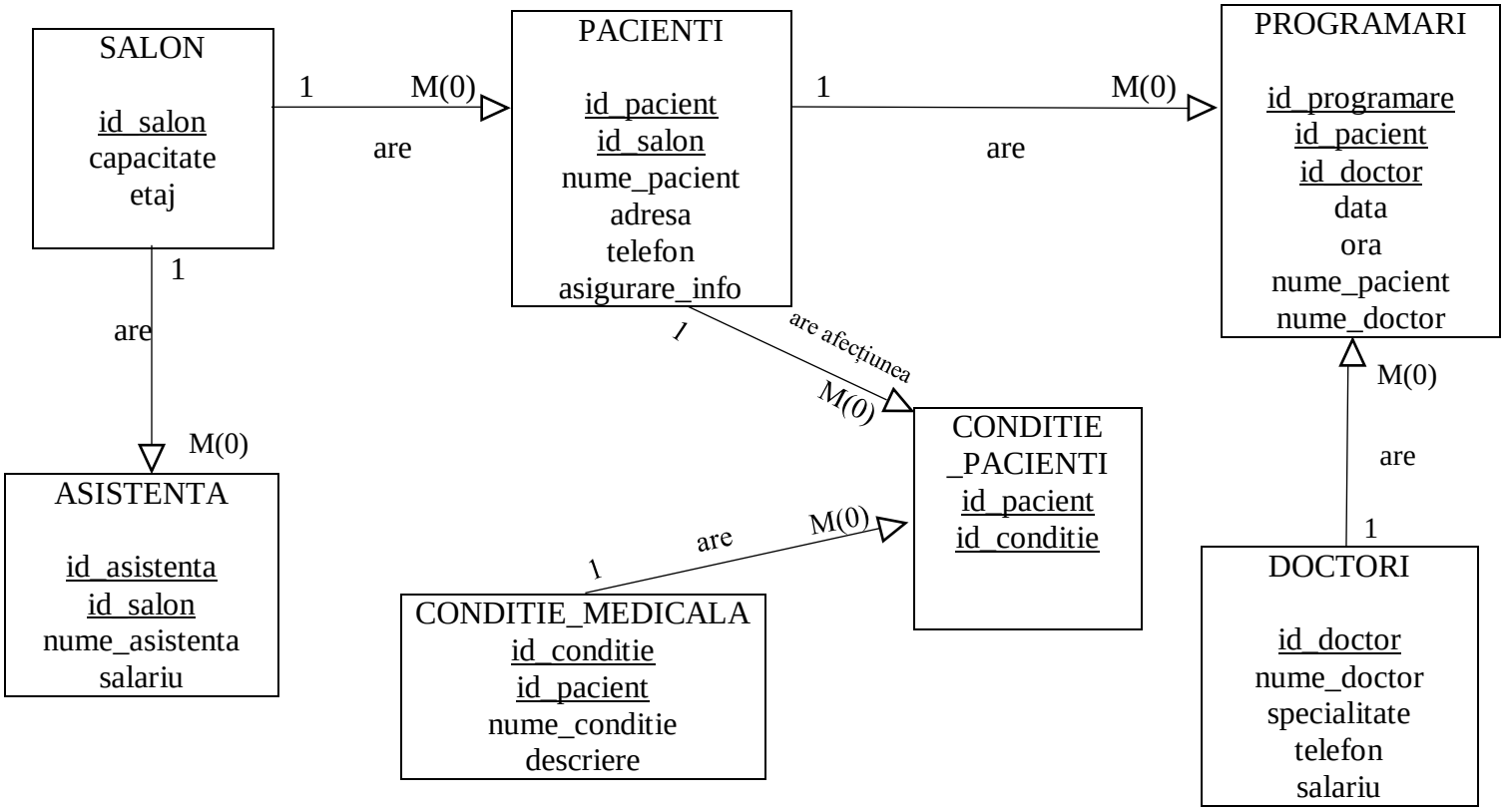
Baza de date permite gestionarea eficientă a informațiilor despre pacienți și medici, planificarea programărilor și urmărirea medicamentelor și a afecțiunilor medicale. Prin utilizarea acestei baze de date, sistemul medical poate ține evidența informațiilor despre pacienți, a programărilor efectuate, a plăților consultațiilor, a personalului medical și pot fi urmărite medicamentele și condițiile medicale, atât cele precedente cât și cele actuale, ceea ce poate ajuta la îmbunătățirea calității îngrijirii pacientului.

2. Realizarea diagramei entitate-relație (ERD):



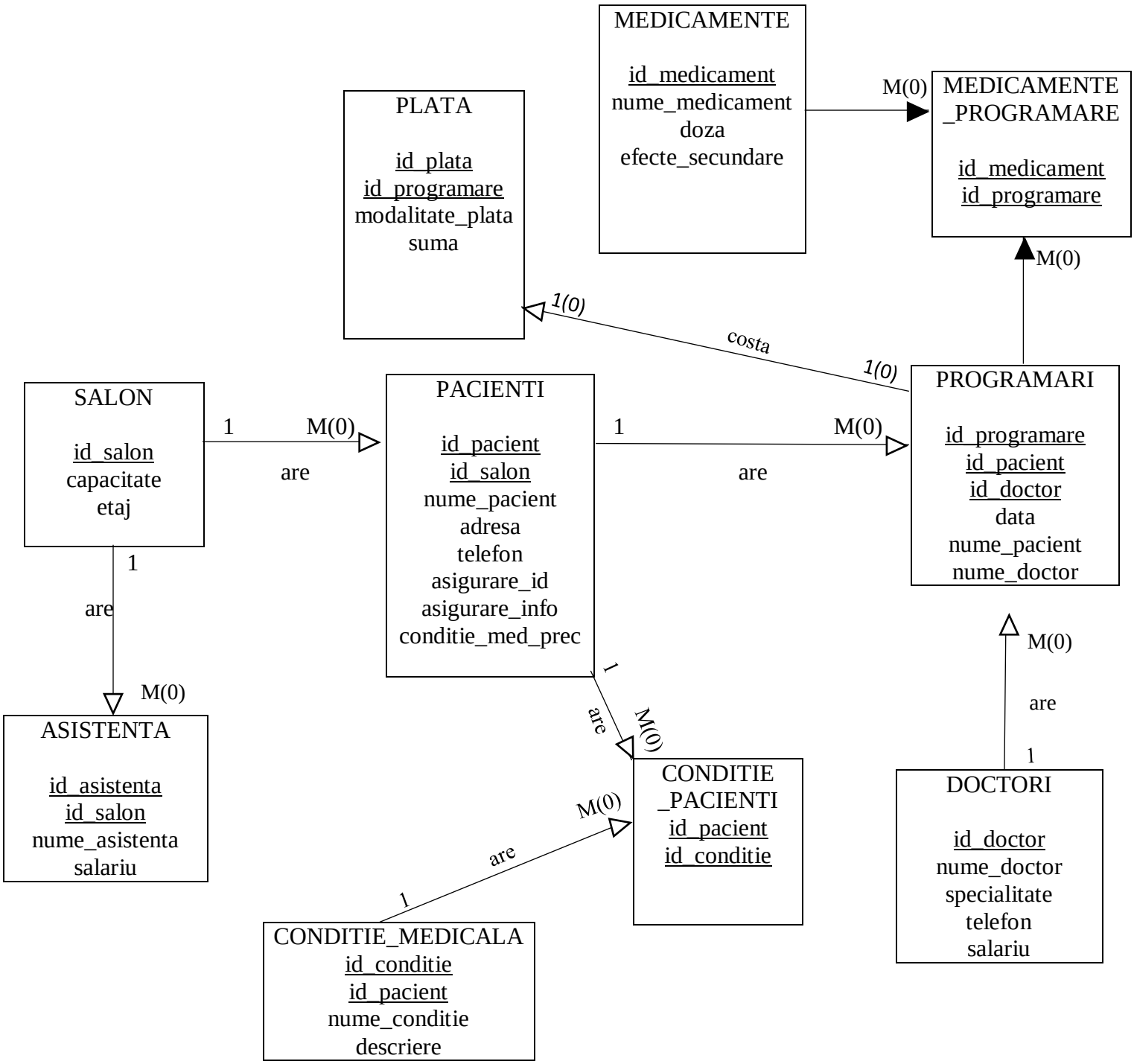
3. Realizarea diagramei conceptuale pornind de la diagrama entitate-relație:

S-a adăugat tabelul asociativ PROGRAMARI și s-au marcat cheile externe.



4. Transformarea sistemului conceptual într-un design logic, subliniind relațiile dintre tabele, cheile primare și străine (externe):

Tabelul asociativ Programari dă naștere tabelelor Plati si Medicamente. Tabelul Medicamente ar fi conectat la tabelul Programari cu o relație de many-to-many, deoarece un medicament poate fi asociat cu mai multe programări și o programare poate avea mai multe medicamente. Pentru a implementa această relație, am creat un al treilea tabel, care reprezintă un join table.



5. **Transformarea design-ului logic într-un design fizic astfel încât sistemul rezultat la punctul 4 să fie în FN3.**

- ***Dați un exemplu de atribut repetitiv (multivaloare) al unei entități din diagramă.***

Un atribut repetitiv este atributul conditie_med_prec, care semnifică condițiile medicale precedente ale pacientului.

PACIENTI

id_pacient	nume_pacient	conditie_med_prec
101	Vasilescu Ionut	ASTM Diabet
101	Vasilescu Ionut	
102	Ionescu Roberto	Hipertensiune

Așadar, se creează tabelul COND_MED_PREC, având cheia primară id_cond_prec, cheia străină id_pacient, care face legătura dintre tabelul PACIENTI și COND_MED_PREC și câmpul cond_med_den care conține denumirea condiției medicale precedente.

COND_MED_PREC

id_cond_prec
id_pacient
cond_med_den

- ***Dați un exemplu de tabel relațional din diagramă care este în FN1, dar nu în FN2. Să se aducă tabelul în FN2.***

Tabelul PROGRAMARI este în FN1 pentru că are cheia primară id_programare și toate celelalte coloane sunt dependente de cheia primară. Cu toate acestea, nu este în FN2 deoarece coloanele nume_pacient și nume_doctor sunt dependente doar de id_pacient și id_doctor. Pentru a aduce tabelul în FN2, aceste coloane ar trebui mutate în tabele separate și legate prin chei străine. Astfel, tabelul PROGRAMARI nu va mai conține câmpurile nume_pacient și nume_doctor. Acestea se vor găsi în cadrul tabelelor PACIENTI, respectiv DOCTORI.

PROGRAMARI

id_programare
id_pacient
id_doctor
data
nume_pacient
nume_doctor

- ***Dați un exemplu de tabel relațional din diagramă care este în FN2, dar nu în FN3. Să se aducă tabelul în FN3.***

PACIENTI
<u>id_pacient</u>
<u>id_salon</u>
nume_pacient
adresa
telefon
asigurare_id
asigurare_info

Tabelul PACIENTI este în FN2 deoarece are o cheie primară (id_pacient) și toate celelalte coloane depind de cheia primară. Cu toate acestea, nu este în FN3, deoarece coloana asigurare_info depinde de coloana asigurare_id, ceea ce creează o dependență parțială. Pentru a o aduce în FN3, această coloană poate fi mutată în tabele separate și legătura dintre tabele se realizează printr-o cheie străină. Așadar, vor arăta astfel:

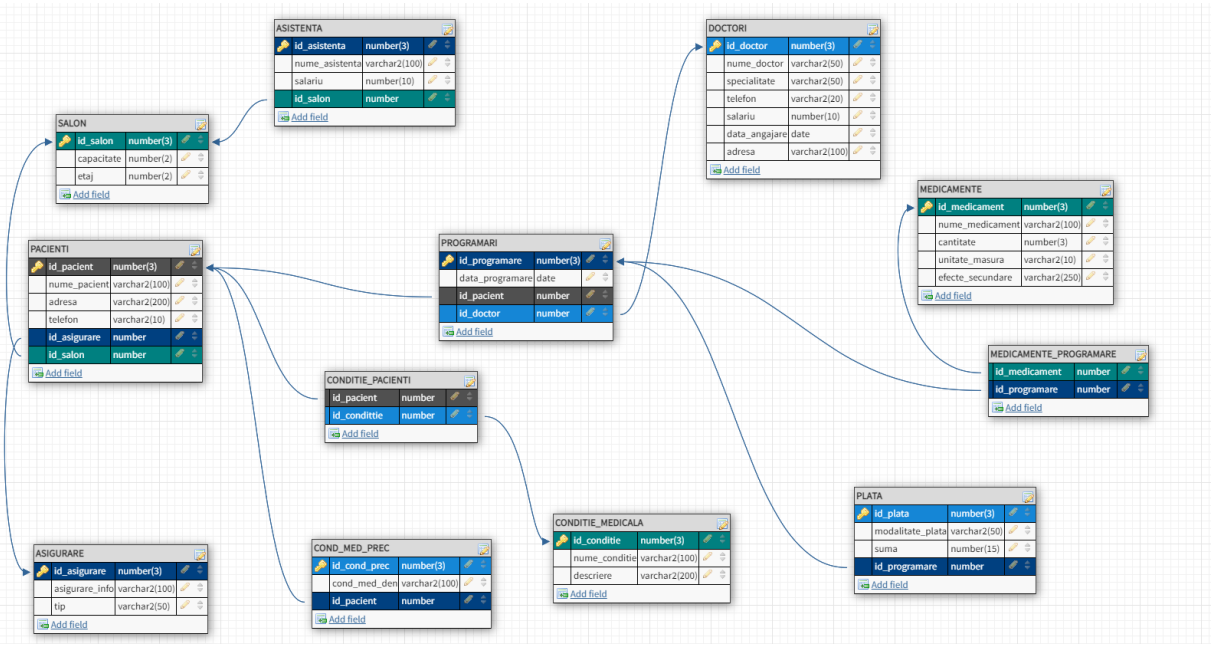
ASIGURARE
id_asigurare
asigurare_info

PACIENTI
<u>id_pacient</u>
<u>id_salon</u>
<u>id_asigurare</u>
nume_pacient
adresa
telefon

Prin mutarea coloanei asigurare_info într-un tabel separat și referirea la acesta printr-o cheie străină, aducem tabelul PACIENTI în a treia formă normală (FN3).

6. **Implementarea tabelelor în Oracle, folosind chei primare, constrângeri de referință și domeniu. Adăugarea de informații coerente(minim 5 înregistrări pentru fiecare entitate independentă; minim 10 înregistrări pentru tabela asociativă).**

SCHEMA BAZEI DE DATE



```
CREATE TABLE SALON
(
  id_salon NUMBER(3) PRIMARY KEY,
  capacitate NUMBER(2),
  etaj NUMBER(2)
);

BEGIN
INSERT INTO SALON (id_salon, capacitate, etaj) VALUES (1, 10, 1);
INSERT INTO SALON (id_salon, capacitate, etaj) VALUES (2, 15, 2);
INSERT INTO SALON (id_salon, capacitate, etaj) VALUES (3, 10, 3);
INSERT INTO SALON (id_salon, capacitate, etaj) VALUES (4, 15, 1);
INSERT INTO SALON (id_salon, capacitate, etaj) VALUES (5, 10, 2);
INSERT INTO SALON (id_salon, capacitate, etaj) VALUES (6, 15, 3);
INSERT INTO SALON (id_salon, capacitate, etaj) VALUES (7, 10, 1);
INSERT INTO SALON (id_salon, capacitate, etaj) VALUES (8, 15, 2);
INSERT INTO SALON (id_salon, capacitate, etaj) VALUES (9, 10, 3);
INSERT INTO SALON (id_salon, capacitate, etaj) VALUES (10, 15, 1);
END;
/
```

```
Table SALON created.


PL/SQL procedure successfully completed.
```

	ID_SALON	CAPACITATE	ETAJ
1	1	10	1
2	2	15	2
3	3	10	3
4	4	15	1
5	5	10	2
6	6	15	3
7	7	10	1
8	8	15	2
9	9	10	3
10	10	15	1

```
CREATE TABLE ASISTENTA
(
  id_asistenta NUMBER(3) PRIMARY KEY,
  nume_asistenta VARCHAR(100) NOT NULL,
  salariu NUMBER(10),
  id_salon REFERENCES SALON(id_salon)
);



BEGIN
INSERT INTO ASISTENTA (id_asistenta, nume_asistenta, salariu, id_salon) VALUES (1, 'Anna Doe', 2000, 1);
INSERT INTO ASISTENTA (id_asistenta, nume_asistenta, salariu, id_salon) VALUES (2, 'Jane Smith', 2500, 2);
INSERT INTO ASISTENTA (id_asistenta, nume_asistenta, salariu, id_salon) VALUES (3, 'Olivia Johnson', 3000, 3);
INSERT INTO ASISTENTA (id_asistenta, nume_asistenta, salariu, id_salon) VALUES (4, 'Emma Williams', 3500, 1);
INSERT INTO ASISTENTA (id_asistenta, nume_asistenta, salariu, id_salon) VALUES (5, 'Michaela Brown', 4000, 2);
INSERT INTO ASISTENTA (id_asistenta, nume_asistenta, salariu, id_salon) VALUES (6, 'Emily Davis', 4500, 3);
INSERT INTO ASISTENTA (id_asistenta, nume_asistenta, salariu, id_salon) VALUES (7, 'Maria Garcia', 5000, 1);
INSERT INTO ASISTENTA (id_asistenta, nume_asistenta, salariu, id_salon) VALUES (8, 'Madison Rodriguez', 5500, 2);
INSERT INTO ASISTENTA (id_asistenta, nume_asistenta, salariu, id_salon) VALUES (9, 'Andra Hernandez', 6000, 3);
INSERT INTO ASISTENTA (id_asistenta, nume_asistenta, salariu, id_salon) VALUES (10, 'Andreea Moore', 6500, 1);
END;
```

/

```
Table ASISTENTA created.


PL/SQL procedure successfully completed.
```

	ID_ASISTENTA	NUME_ASISTENTA	SALARIU	ID_SALON
1	1	Anna Doe	2000	1
2	2	Jane Smith	2500	2
3	3	Olivia Johnson	3000	3
4	4	Emma Williams	3500	1
5	5	Michaela Brown	4000	2
6	6	Emily Davis	4500	3
7	7	Maria Garcia	5000	1
8	8	Madison Rodriguez	5500	2
9	9	Andra Hernandez	6000	3
10	10	Andreea Moore	6500	1

```
CREATE TABLE DOCTORI
(
  id_doctor NUMBER(3) PRIMARY KEY,
  nume_doctor VARCHAR(50) NOT NULL,
  specialitate VARCHAR(50) NOT NULL,
  telefon VARCHAR(20) NOT NULL,
  salariu NUMBER(10),
  data_angajare DATE,
  adresa VARCHAR(100) NOT NULL

);

BEGIN
INSERT INTO DOCTORI (id_doctor, nume_doctor, specialitate, telefon, salariu, data_angajare, adresa) VALUES (1, 'John Doe', 'Chirurgie Plastica', '1234567890', 10000, to_date('2022-05-01', 'yyyy-mm-dd'), 'Bucuresti');
INSERT INTO DOCTORI (id_doctor, nume_doctor, specialitate, telefon, salariu, data_angajare, adresa) VALUES (2, 'Richard David', 'Pediatrie', '2345678901', 12000, to_date('2020-08-10', 'yyyy-mm-dd'), 'Bucuresti');
INSERT INTO DOCTORI (id_doctor, nume_doctor, specialitate, telefon, salariu, data_angajare, adresa) VALUES (3, 'Bob Johnson', 'Cardiologie', '3456789012', 15000, to_date('2020-10-20', 'yyyy-mm-dd'), 'Bucuresti');
```

INSERT INTO DOCTORI (id_doctor, nume_doctor, specialitate, telefon, salariu, data_angajare, adresa) VALUES (4, 'Alisia Brown', 'Oncologie', '4567890123', 20000, to_date('2018-03-15', 'yyyy-mm-dd'), 'Bucuresti');

INSERT INTO DOCTORI (id_doctor, nume_doctor, specialitate, telefon, salariu, data_angajare, adresa) VALUES (5, 'Michael Brown', 'Ortopedie', '5678901234', 25000, to_date('2019-04-12', 'yyyy-mm-dd'), 'Bucuresti');

INSERT INTO DOCTORI (id_doctor, nume_doctor, specialitate, telefon, salariu, data_angajare, adresa) VALUES (6, 'Patrick Johnas', 'Anesteziologie', '6789012345', 30000, to_date('2020-09-11', 'yyyy-mm-dd'), 'Timisoara');

INSERT INTO DOCTORI (id_doctor, nume_doctor, specialitate, telefon, salariu, data_angajare, adresa) VALUES (7, 'Joshua Garcia', 'Neurologie', '7890123456', 35000, to_date('2021-01-21', 'yyyy-mm-dd'), 'Timisoara');

INSERT INTO DOCTORI (id_doctor, nume_doctor, specialitate, telefon, salariu, data_angajare, adresa) VALUES (8, 'Maddie Bryan', 'Gastroenterologie', '8901234567', 40000, to_date('2020-01-20', 'yyyy-mm-dd'), 'Bucuresti');

INSERT INTO DOCTORI (id_doctor, nume_doctor, specialitate, telefon, salariu, data_angajare, adresa) VALUES (9, 'Matthew Pop', 'Dermatologie', '9012345678', 45000, to_date('2018-12-01', 'yyyy-mm-dd'), 'Craiova');

INSERT INTO DOCTORI (id_doctor, nume_doctor, specialitate, telefon, salariu, data_angajare, adresa) VALUES (10, 'Andrew Robinson', 'Radiologie', '0123456789', 50000, to_date('2018-12-22', 'yyyy-mm-dd'), 'Bucuresti');

END;

```
Table DOCTORI created.


PL/SQL procedure successfully completed.
/
```

	ID_DOCTOR	NUME_DOCTOR	SPECIALITATE	TELEFON	SALARIU	DATA_ANGAJARE	ADRESA
1	1	John Doe	Chirurgie Plastica	1234567890	10000	01-MAY-22	Bucuresti
2	2	Richard David	Pediatrie	2345678901	12000	10-AUG-20	Bucuresti
3	3	Bob Johnson	Cardiologie	3456789012	15000	20-OCT-20	Bucuresti
4	4	Alisia Brown	Oncologie	4567890123	20000	15-MAR-18	Bucuresti
5	5	Michael Brown	Ortopedie	5678901234	25000	12-APR-19	Bucuresti
6	6	Patrick Johnas	Anesteziologie	6789012345	30000	11-SEP-20	Timisoara
7	7	Joshua Garcia	Neurologie	7890123456	35000	21-JAN-21	Timisoara
8	8	Maddie Bryan	Gastroenterologie	8901234567	40000	20-JAN-20	Bucuresti
9	9	Matthew Pop	Dermatologie	9012345678	45000	01-DEC-18	Craiova
10	10	Andrew Robinson	Radiologie	0123456789	50000	22-DEC-18	Bucuresti

```
CREATE TABLE ASIGURARE
(
  id_asigurare NUMBER(3) PRIMARY KEY,
  asigurare_info VARCHAR(100) NOT NULL,
  tip VARCHAR(50) NOT NULL
);


BEGIN
INSERT INTO ASIGURARE (id_asigurare, asigurare_info, tip) VALUES (1, 'Blue Cross Blue Shield', 'publica');
INSERT INTO ASIGURARE (id_asigurare, asigurare_info, tip) VALUES (2, 'Aetna', 'privata');
INSERT INTO ASIGURARE (id_asigurare, asigurare_info, tip) VALUES (3, 'United Healthcare', 'publica');
INSERT INTO ASIGURARE (id_asigurare, asigurare_info, tip) VALUES (4, 'Humana', 'privata');
INSERT INTO ASIGURARE (id_asigurare, asigurare_info, tip) VALUES (5, 'Cigna','publica');
INSERT INTO ASIGURARE (id_asigurare, asigurare_info, tip) VALUES (6, 'Kaiser Permanente', 'privata');
INSERT INTO ASIGURARE (id_asigurare, asigurare_info, tip) VALUES (7, 'Health Net', 'privata');
INSERT INTO ASIGURARE (id_asigurare, asigurare_info, tip) VALUES (8, 'Medicare', 'privata');
INSERT INTO ASIGURARE (id_asigurare, asigurare_info, tip) VALUES (9, 'Medicaid', 'privata');
INSERT INTO ASIGURARE (id_asigurare, asigurare_info, tip) VALUES (10, 'Amerigroup', 'publica');
END;
```

/

	ID_ASIGURARE	ASIGURARE_INFO	TIP
1	1	Blue Cross Blue Shield	publica
2	2	Aetna	privata
3	3	United Healthcare	publica
4	4	Humana	privata
5	5	Cigna	publica
6	6	Kaiser Permanente	privata
7	7	Health Net	privata
8	8	Medicare	privata
9	9	Medicaid	privata
10	10	Amerigroup	publica

```
Table ASIGURARE created.


PL/SQL procedure successfully completed.
```

CREATE TABLE PACIENTI

(

id_pacient NUMBER(3) PRIMARY KEY,

nume_pacient VARCHAR(100) NOT NULL,

adresa VARCHAR(200) NOT NULL,

telefon VARCHAR(10) NOT NULL,

id_asigurare REFERENCES ASIGURARE(id_asigurare),

id_salon REFERENCES SALON(id_salon)

);

BEGIN

INSERT INTO PACIENTI (id_pacient, nume_pacient, adresa, telefon, id_asigurare, id_salon) VALUES (1, 'James Doe', 'Bucuresti', '1234567890', 1, 1);

INSERT INTO PACIENTI (id_pacient, nume_pacient, adresa, telefon, id_asigurare, id_salon) VALUES (2, 'Patricia Smith', 'Timisoara', '2345678901', 2, 2);

INSERT INTO PACIENTI (id_pacient, nume_pacient, adresa, telefon, id_asigurare, id_salon) VALUES (3, 'Jennifer Johnson', 'Craiova', '3456789012', 3, 3);

INSERT INTO PACIENTI (id_pacient, nume_pacient, adresa, telefon, id_asigurare, id_salon) VALUES (4, 'Linda Williams', 'Cluj', '4567890123', 4, 1);

INSERT INTO PACIENTI (id_pacient, nume_pacient, adresa, telefon, id_asigurare, id_salon) VALUES (5, 'Elizabeth Brown', 'Timisoara', '5678901234', 5, 2);

INSERT INTO PACIENTI (id_pacient, nume_pacient, adresa, telefon, id_asigurare, id_salon) VALUES (6, 'Barbara Smith', 'Bucuresti', '1234567890', 6, 1);

INSERT INTO PACIENTI (id_pacient, nume_pacient, adresa, telefon, id_asigurare, id_salon) VALUES (7, 'Joseph Brown', 'Brasov', '2345678901', 7, 2);

INSERT INTO PACIENTI (id_pacient, nume_pacient, adresa, telefon, id_asigurare, id_salon) VALUES (8, 'Charles Johnson', 'Brasov', '3456789012', 9, 3);

INSERT INTO PACIENTI (id_pacient, nume_pacient, adresa, telefon, id_asigurare, id_salon) VALUES (9, 'Daniel Williams', 'Bucuresti', '4567890123', 8, 1);

INSERT INTO PACIENTI (id_pacient, nume_pacient, adresa, telefon, id_asigurare, id_salon) VALUES (10, 'Nancy Brown', 'Timisoara', '5678901234', 10, 2);

INSERT INTO PACIENTI (id_pacient, nume_pacient, adresa, telefon, id_asigurare, id_salon) VALUES (11, 'Kimberly Davis', 'Bucuresti', '6789012345', 1, 3);

INSERT INTO PACIENTI (id_pacient, nume_pacient, adresa, telefon, id_asigurare, id_salon) VALUES (12, 'Karen Garcia', 'Bucuresti', '7890123456', 2, 1);

INSERT INTO PACIENTI (id_pacient, nume_pacient, adresa, telefon, id_asigurare, id_salon) VALUES (13, 'Sarah Taylor', 'Craiova', '8901234567', 3, 2);

INSERT INTO PACIENTI (id_pacient, nume_pacient, adresa, telefon, id_asigurare, id_salon) VALUES (14, 'Sophia Perez', 'Bucuresti', '9012345678', 4, 3);

INSERT INTO PACIENTI (id_pacient, nume_pacient, adresa, telefon, id_asigurare, id_salon) VALUES (15, 'Paul Garcia', 'Bucuresti', '0123456789', 5, 1);

INSERT INTO PACIENTI (id_pacient, nume_pacient, adresa, telefon, id_asigurare, id_salon) VALUES (16, 'Donna Martinez', 'Bucuresti', '1234567890', 1, 2);

INSERT INTO PACIENTI (id_pacient, nume_pacient, adresa, telefon, id_asigurare, id_salon) VALUES (17, 'Jessica Lopez', 'Timisoara', '2345678901', 2, 3);

INSERT INTO PACIENTI (id_pacient, nume_pacient, adresa, telefon, id_asigurare, id_salon) VALUES (18, 'George Rodriguez', 'Cluj', '2345678901', 3, 3);

INSERT INTO PACIENTI (id_pacient, nume_pacient, adresa, telefon, id_asigurare, id_salon) VALUES (19, 'Timothy Brian', 'Bucuresti', '2345678901', 4, 3);

INSERT INTO PACIENTI (id_pacient, nume_pacient, adresa, telefon, id_asigurare, id_salon) VALUES (20, 'Cynthia Pop', 'Timisoara', '2345678901', 5, 3);

INSERT INTO PACIENTI (id_pacient, nume_pacient, adresa, telefon, id_asigurare, id_salon) VALUES (21, 'Mara Popescu', '624 Elm St', '2236478901', 5, 3);

INSERT INTO PACIENTI (id_pacient, nume_pacient, adresa, telefon, id_asigurare, id_salon) VALUES (22, 'Sassy Green', 'Timisoara', '5678557634', 10, 2);

END;

/

```
Table PACIENTI created.


PL/SQL procedure successfully completed.
```

	ID_PACIENT	NUME_PACIENT	ADRESA	TELEFON	ID_ASIGURARE	ID_SALON
1	1	James Doe	Bucuresti	1234567890	1	1
2	2	Patricia Smith	Timisoara	2345678901	2	2
3	3	Jennifer Johnson	Craiova	3456789012	3	3
4	4	Linda Williams	Cluj	4567890123	4	1
5	5	Elizabeth Brown	Timisoara	5678901234	5	2
6	6	Barbara Smith	Bucuresti	1234567890	6	1
7	7	Joseph Brown	Brasov	2345678901	7	2
8	8	Charles Johnson	Brasov	3456789012	9	3
9	9	Daniel Williams	Bucuresti	4567890123	8	1
10	10	Nancy Brown	Timisoara	5678901234	10	2
11	11	Kimberly Davis	Bucuresti	6789012345	1	3
12	12	Karen Garcia	Bucuresti	7890123456	2	1
13	13	Sarah Taylor	Craiova	8901234567	3	2
14	14	Sophia Perez	Bucuresti	9012345678	4	3
15	15	Paul Garcia	Bucuresti	0123456789	5	1
16	16	Donna Martinez	Bucuresti	1234567890	1	2
17	17	Jessica Lopez	Timisoara	2345678901	2	3
18	18	George Rodriguez	Cluj	2345678901	3	3
19	19	Timothy Brian	Bucuresti	2345678901	4	3
20	20	Cynthia Pop	Timisoara	2345678901	5	3
21	21	Mara Popescu	624 Elm St	2236478901	5	3
22	22	Sassy Green	Timisoara	5678557634	10	2

CREATE TABLE COND_MED_PREC

(

id_cond_prec NUMBER(3) PRIMARY KEY,

cond_med_den VARCHAR(100) NOT NULL,

id_pacient REFERENCES PACIENTI(id_pacient)

);

BEGIN

INSERT INTO COND_MED_PREC (id_cond_prec, cond_med_den, id_pacient) VALUES (1, 'Diabet', 1);

INSERT INTO COND_MED_PREC (id_cond_prec, cond_med_den, id_pacient) VALUES (2, 'Hipertensiune', 2);

INSERT INTO COND_MED_PREC (id_cond_prec, cond_med_den, id_pacient) VALUES (3, 'Astm', 3);

INSERT INTO COND_MED_PREC (id_cond_prec, cond_med_den, id_pacient) VALUES (4, 'Artrita', 1);

INSERT INTO COND_MED_PREC (id_cond_prec, cond_med_den, id_pacient) VALUES (5, 'Depresie', 1);

INSERT INTO COND_MED_PREC (id_cond_prec, cond_med_den, id_pacient) VALUES (6, 'Anxietate', 2);

INSERT INTO COND_MED_PREC (id_cond_prec, cond_med_den, id_pacient) VALUES (7, 'Cancer', 4);

INSERT INTO COND_MED_PREC (id_cond_prec, cond_med_den, id_pacient) VALUES (8, 'Boala cardiovasculara', 5);

INSERT INTO COND_MED_PREC (id_cond_prec, cond_med_den, id_pacient) VALUES (9, 'Obezitate', 6);

INSERT INTO COND_MED_PREC (id_cond_prec, cond_med_den, id_pacient) VALUES (10, 'Accident vascular cerebral', 7);

INSERT INTO COND_MED_PREC (id_cond_prec, cond_med_den, id_pacient) VALUES (11, 'Boala la rinichi', 8);

INSERT INTO COND_MED_PREC (id_cond_prec, cond_med_den, id_pacient) VALUES (12, 'Boala la ficat', 9);

INSERT INTO COND_MED_PREC (id_cond_prec, cond_med_den, id_pacient) VALUES (13, 'Boala autoimuna', 10);

INSERT INTO COND_MED_PREC (id_cond_prec, cond_med_den, id_pacient) VALUES (14, 'Boala tiroidiana', 11);

INSERT INTO COND_MED_PREC (id_cond_prec, cond_med_den, id_pacient) VALUES (15, 'Boala la plamani', 12);

```
INSERT INTO COND_MED_PREC (id_cond_prec, cond_med_den, id_pacient) VALUES (16, 'Boli gastrointestinale', 13);

INSERT INTO COND_MED_PREC (id_cond_prec, cond_med_den, id_pacient) VALUES (17, 'Tulburări neurologice', 13);

INSERT INTO COND_MED_PREC (id_cond_prec, cond_med_den, id_pacient) VALUES (18, 'Obezitate', 14);

INSERT INTO COND_MED_PREC (id_cond_prec, cond_med_den, id_pacient) VALUES (19, 'Boala infectioasa', 15);

INSERT INTO COND_MED_PREC (id_cond_prec, cond_med_den, id_pacient) VALUES (20, 'Boala de sange', 16);

END;

/
```

```
Table COND_MED_PREC created.


PL/SQL procedure successfully completed.
```

	ID_COND_PREC	COND_MED_DEN	ID_PACIENT
1	1	Diabet	1
2	2	Hipertensiune	2
3	3	Astm	3
4	4	Artrita	1
5	5	Depresie	1
6	6	Anxietate	2
7	7	Cancer	4
8	8	Boala cardiovasculara	5
9	9	Obezitate	6
10	10	Accident vascular cerebral	7
11	11	Boala la rinichi	8
12	12	Boala la ficat	9
13	13	Boala autoimuna	10
14	14	Boala tiroidiana	11
15	15	Boala la plamani	12
16	16	Boli gastrointestinale	13
17	17	Tulburări neurologice	13
18	18	Obezitate	14
19	19	Boala infectioasa	15
20	20	Boala de sange	16

```
CREATE TABLE CONDITIE_MEDICALA
(
  id_conditie NUMBER(3) PRIMARY KEY,
  nume_conditie VARCHAR(100) NOT NULL,
  descriere VARCHAR(200) NOT NULL
);
```

```
BEGIN
INSERT INTO CONDITIE_MEDICALA (id_conditie, nume_conditie, descriere) VALUES (1, 'Cancer', 'O boala caracterizata prin cresterea si raspandirea necontrolata a celulelor anormale');
INSERT INTO CONDITIE_MEDICALA (id_conditie, nume_conditie, descriere) VALUES (2, 'Diabet', 'O afectiune cronica caracterizata prin niveluri ridicate de zahar in ssnge');
INSERT INTO CONDITIE_MEDICALA (id_conditie, nume_conditie, descriere) VALUES (3, 'Astm', 'O boală pulmonara cronica care inflameaza si ingusteaza caile respiratorii');
INSERT INTO CONDITIE_MEDICALA (id_conditie, nume_conditie, descriere) VALUES (4, 'Artrita', 'Un grup de afectiuni caracterizate prin inflamatie la nivelul articulatiilor');
INSERT INTO CONDITIE_MEDICALA (id_conditie, nume_conditie, descriere) VALUES (5, 'Hipertensiune', 'O afectiune cronica caracterizata prin hipertensiune arteriala');
INSERT INTO CONDITIE_MEDICALA (id_conditie, nume_conditie, descriere) VALUES (6, 'Boala cardiovasculara', 'Un termen general pentru o varietate de afectiuni care afecteaza inima');
INSERT INTO CONDITIE_MEDICALA (id_conditie, nume_conditie, descriere) VALUES (7, 'Depresie', 'O tulburare mentala caracterizata prin sentimente persistente de tristete, lipsa de speranta si pierderea interesului');
INSERT INTO CONDITIE_MEDICALA (id_conditie, nume_conditie, descriere) VALUES (8, 'Anemie', 'O afectiune caracterizata prin lipsa globulelor rosii sau a hemoglobinei in sange');
INSERT INTO CONDITIE_MEDICALA (id_conditie, nume_conditie, descriere) VALUES (9, 'Alergii', 'O afectiune caracterizata printr-un raspuns imunitar hiperactiv la o substanta');
INSERT INTO CONDITIE_MEDICALA (id_conditie, nume_conditie, descriere) VALUES (10, 'Gripa tip A', 'O infectie virala care afecteaza sistemul respirator');
INSERT INTO CONDITIE_MEDICALA (id_conditie, nume_conditie, descriere) VALUES (11, 'Dureri de spate', 'Durere resimtita in partea inferioara a spatelui');
```

```
INSERT INTO CONDITIE_MEDICALA (id_conditie, nume_conditie, descriere) VALUES (12, 'Migrene', 'Un tip de durere de cap caracterizata prin durere severa si alte simptome');

INSERT INTO CONDITIE_MEDICALA (id_conditie, nume_conditie, descriere) VALUES (13, 'Ulcer', 'Ulcerul gastric este dat de prezenta unei ulceratii dureroase sau rani ale mucoasei gastrice');

INSERT INTO CONDITIE_MEDICALA (id_conditie, nume_conditie, descriere) VALUES (14, 'Bronsita', 'Inflamatie a tuburilor bronsice');

INSERT INTO CONDITIE_MEDICALA (id_conditie, nume_conditie, descriere) VALUES (15, 'Pneumonie', 'O infectie a plamanilor care provoaca inflamatie si acumulare de lichid');

INSERT INTO CONDITIE_MEDICALA (id_conditie, nume_conditie, descriere) VALUES (16, 'Tuberculoza', 'O infectie bacteriana care afecteaza in primul rand plamanii');

INSERT INTO CONDITIE_MEDICALA (id_conditie, nume_conditie, descriere) VALUES (17, 'Gripa tip B', 'O infectie virala care afecteaza sistemul respirator');

INSERT INTO CONDITIE_MEDICALA (id_conditie, nume_conditie, descriere) VALUES (18, 'Pojar', 'O infectie virala foarte contagioasa');

INSERT INTO CONDITIE_MEDICALA (id_conditie, nume_conditie, descriere) VALUES (19, 'Varicelă', 'O infectie virala foarte contagioasa');

INSERT INTO CONDITIE_MEDICALA (id_conditie, nume_conditie, descriere) VALUES (20, 'Eczema', 'Un grup de afectiuni care fac ca pielea sa devina rosie, inflamata si provoaca mancarime');

END;

/
```

```
Table CONDITIE_MEDICALA created.


PL/SQL procedure successfully completed.
```

	ID_CONDITIE	NUME_CONDITIE	DESCRIERE
1	1	Cancer	O boala caracterizata prin cresterea si raspandirea necontrolata a celulelor anormale
2	2	Diabet	O afectiune cronica caracterizata prin niveluri ridicate de zahar in ssnge
3	3	Astm	O boală pulmonara cronica care inflameaza si ingusteaza caile respiratorii
4	4	Artrita	Un grup de afectiuni caracterizate prin inflamatie la nivelul articulatiilor
5	5	Hipertensiune	O afectiune cronica caracterizata prin hipertensiune arteriala
6	6	Boala cardiovasculara	Un termen general pentru o varietate de afectiuni care afecteaza inima
7	7	Depresie	O tulburare mentala caracterizata prin sentimente persistente de tristete, lipsa de speranta si pierderea interesului
8	8	Anemie	O afectiune caracterizata prin lipsa globulelor rosii sau a hemoglobinei in sange
9	9	Alergii	O afectiune caracterizata printr-un raspuns imunitar hiperactiv la o substanta
10	10	Gripa tip A	O infectie virala care afecteaza sistemul respirator
11	11	Dureri de spate	Durere resimtita in partea inferioara a spatelui
12	12	Migrene	Un tip de durere de cap caracterizata prin durere severa si alte simptome
13	13	Ulcer	Ulcerul gastric este dat de prezenta unei ulceratii dureroase sau rani ale mucoasei gastrice
14	14	Bronsita	Inflamatie a tuburilor bronsice
15	15	Pneumonie	O infectie a plamanilor care provoaca inflamatie si acumulare de lichid
16	16	Tuberculoza	O infectie bacteriana care afecteaza in primul rand plamanii
17	17	Gripa tip B	O infectie virala care afecteaza sistemul respirator
18	18	Pojar	O infectie virala foarte contagioasa
19	19	Varicelă	O infectie virala foarte contagioasa
20	20	Eczema	Un grup de afectiuni care fac ca pielea sa devina rosie, inflamata si provoaca mancarime

```
CREATE TABLE CONDITIE_PACIENTI
(
  id_pacient REFERENCES PACIENTI(id_pacient),
  id_conditie REFERENCES CONDITIE_MEDICALA(id_conditie)
);

BEGIN
INSERT INTO CONDITIE_PACIENTI (id_pacient, id_conditie) VALUES (1, 10);
INSERT INTO CONDITIE_PACIENTI (id_pacient, id_conditie) VALUES (2, 17);
INSERT INTO CONDITIE_PACIENTI (id_pacient, id_conditie) VALUES (3, 8);
INSERT INTO CONDITIE_PACIENTI (id_pacient, id_conditie) VALUES (4, 9);
INSERT INTO CONDITIE_PACIENTI (id_pacient, id_conditie) VALUES (5, 11);
INSERT INTO CONDITIE_PACIENTI (id_pacient, id_conditie) VALUES (6, 12);
INSERT INTO CONDITIE_PACIENTI (id_pacient, id_conditie) VALUES (7, 13);
INSERT INTO CONDITIE_PACIENTI (id_pacient, id_conditie) VALUES (8, 14);
INSERT INTO CONDITIE_PACIENTI (id_pacient, id_conditie) VALUES (9, 15);
```

```
INSERT INTO CONDITIE_PACIENTI (id_pacient, id_conditie) VALUES (10, 16);
INSERT INTO CONDITIE_PACIENTI (id_pacient, id_conditie) VALUES (11, 17);
INSERT INTO CONDITIE_PACIENTI (id_pacient, id_conditie) VALUES (12, 18);
INSERT INTO CONDITIE_PACIENTI (id_pacient, id_conditie) VALUES (13, 19);
INSERT INTO CONDITIE_PACIENTI (id_pacient, id_conditie) VALUES (14, 20);
INSERT INTO CONDITIE_PACIENTI (id_pacient, id_conditie) VALUES (15, 7);
INSERT INTO CONDITIE_PACIENTI (id_pacient, id_conditie) VALUES (16, 6);
INSERT INTO CONDITIE_PACIENTI (id_pacient, id_conditie) VALUES (17, 5);
INSERT INTO CONDITIE_PACIENTI (id_pacient, id_conditie) VALUES (18, 4);
INSERT INTO CONDITIE_PACIENTI (id_pacient, id_conditie) VALUES (19, 3);
INSERT INTO CONDITIE_PACIENTI (id_pacient, id_conditie) VALUES (20, 2);
END;
/
```

```
Table CONDITIE_PACIENTI created.


PL/SQL procedure successfully completed.
```

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

```
CREATE TABLE PROGRAMARI
(
  id_programare NUMBER(3) PRIMARY KEY,
  data_programare DATE NOT NULL,
  id_pacient REFERENCES PACIENTI(id_pacient),
  id_doctor REFERENCES DOCTORI(id_doctor)

);

BEGIN
INSERT INTO PROGRAMARI (id_programare, data_programare, id_pacient, id_doctor)
VALUES(1,  to_date('2023-02-01', 'yyyy-mm-dd'), 1, 1);

INSERT INTO PROGRAMARI (id_programare, data_programare, id_pacient, id_doctor)
VALUES(2,  to_date('2023-02-02', 'yyyy-mm-dd'), 2, 2);

INSERT INTO PROGRAMARI (id_programare, data_programare, id_pacient, id_doctor)
VALUES(3,  to_date('2023-02-03', 'yyyy-mm-dd'), 3, 3);

INSERT INTO PROGRAMARI (id_programare, data_programare, id_pacient, id_doctor)
VALUES(4,  to_date('2023-02-04', 'yyyy-mm-dd'), 4, 4);

INSERT INTO PROGRAMARI (id_programare, data_programare, id_pacient, id_doctor)
VALUES(5,  to_date('2023-02-05', 'yyyy-mm-dd'), 5, 5);

INSERT INTO PROGRAMARI (id_programare, data_programare, id_pacient, id_doctor)
VALUES(6,  to_date('2023-02-06', 'yyyy-mm-dd'), 6, 6);

INSERT INTO PROGRAMARI (id_programare, data_programare, id_pacient, id_doctor)
VALUES(7,  to_date('2023-02-07', 'yyyy-mm-dd'), 7, 7);

INSERT INTO PROGRAMARI (id_programare, data_programare, id_pacient, id_doctor)
VALUES(8,  to_date('2023-02-08', 'yyyy-mm-dd'), 8, 8);

INSERT INTO PROGRAMARI (id_programare, data_programare, id_pacient, id_doctor)
VALUES(9,  to_date('2023-02-09', 'yyyy-mm-dd'), 9, 9);

INSERT INTO PROGRAMARI (id_programare, data_programare, id_pacient, id_doctor)
VALUES(10,  to_date('2023-02-10', 'yyyy-mm-dd'), 10, 10);

INSERT INTO PROGRAMARI (id_programare, data_programare, id_pacient, id_doctor)
VALUES(11,  to_date('2023-02-11', 'yyyy-mm-dd'), 11, 1);
```

```
INSERT INTO PROGRAMARI (id_programare, data_programare, id_pacient, id_doctor)
VALUES(12,  to_date('2023-02-12', 'yyyy-mm-dd'), 12, 2);

INSERT INTO PROGRAMARI (id_programare, data_programare, id_pacient, id_doctor)
VALUES(13,  to_date('2023-02-13', 'yyyy-mm-dd'), 13, 3);

INSERT INTO PROGRAMARI (id_programare, data_programare, id_pacient, id_doctor)
VALUES(14,  to_date('2023-02-14', 'yyyy-mm-dd'), 14, 4);

INSERT INTO PROGRAMARI (id_programare, data_programare, id_pacient, id_doctor)
VALUES(15,  to_date('2023-02-15', 'yyyy-mm-dd'), 15, 5);

INSERT INTO PROGRAMARI (id_programare, data_programare, id_pacient, id_doctor)
VALUES(16,  to_date('2023-02-16', 'yyyy-mm-dd'), 16, 6);

INSERT INTO PROGRAMARI (id_programare, data_programare, id_pacient, id_doctor)
VALUES(17,  to_date('2023-02-17', 'yyyy-mm-dd'), 17, 7);

INSERT INTO PROGRAMARI (id_programare, data_programare, id_pacient, id_doctor)
VALUES(18,  to_date('2023-02-18', 'yyyy-mm-dd'), 18, 8);

INSERT INTO PROGRAMARI (id_programare, data_programare, id_pacient, id_doctor)
VALUES(19,  to_date('2023-02-19', 'yyyy-mm-dd'), 19, 9);

INSERT INTO PROGRAMARI (id_programare, data_programare, id_pacient, id_doctor)
VALUES(20,  to_date('2023-02-20', 'yyyy-mm-dd'), 20, 1);

INSERT INTO PROGRAMARI
(id_programare, data_programare, id_pacient,
id_doctor) VALUES(21,  to_date('2023-02-20',
'yyyy-mm-dd'), 20, 1);

INSERT INTO PROGRAMARI
(id_programare, data_programare, id_pacient,
id_doctor) VALUES(22,  to_date('2023-05-10',
'yyyy-mm-dd'), 22, 2);

INSERT INTO PROGRAMARI
(id_programare, data_programare, id_pacient,
id_doctor) VALUES(23,  to_date('2023-03-21',
'yyyy-mm-dd'), 22, 2);

END;

/
```

```
Table PROGRAMARI created.


PL/SQL procedure successfully completed.
```

	ID_PROGRAMARE	DATA_PROGRAMARE	ID_PACIENT	ID_DOCTOR
1	1	01-FEB-23	1	1
2	2	02-FEB-23	2	2
3	3	03-FEB-23	3	3
4	4	04-FEB-23	4	4
5	5	05-FEB-23	5	5
6	6	06-FEB-23	6	6
7	7	07-FEB-23	7	7
8	8	08-FEB-23	8	8
9	9	09-FEB-23	9	9
10	10	10-FEB-23	10	10
11	11	11-FEB-23	11	1
12	12	12-FEB-23	12	2
13	13	13-FEB-23	13	3
14	14	14-FEB-23	14	4
15	15	15-FEB-23	15	5
16	16	16-FEB-23	16	6
17	17	17-FEB-23	17	7
18	18	18-FEB-23	18	8
19	19	19-FEB-23	19	9
20	20	20-FEB-23	20	1
21	21	20-FEB-23	20	1
22	22	10-MAY-23	22	2
23	23	21-MAR-23	22	2

```
CREATE TABLE MEDICAMENTE
(
  id_medicament NUMBER(3) PRIMARY KEY,
  nume_medicament VARCHAR(100) NOT NULL,
  cantitate NUMBER(6),
  unitate_masura VARCHAR(10) NOT NULL,
  efecte_secundare VARCHAR(250) NOT NULL
);
```

```
BEGIN
INSERT INTO MEDICAMENTE (id_medicament, nume_medicament, cantitate, unitate_masura, efecte_secundare) VALUES (1, 'Aspirin', 200, 'mg', 'Iritatie la stomac, risc crescut de sangerare');
INSERT INTO MEDICAMENTE (id_medicament, nume_medicament, cantitate, unitate_masura, efecte_secundare) VALUES (2, 'Ibuprofen', 200, 'mg', 'Iritatie la stomac, risc crescut de sangerare');
INSERT INTO MEDICAMENTE (id_medicament, nume_medicament, cantitate, unitate_masura, efecte_secundare) VALUES (3, 'Paracetamol', 500, 'mg', 'Leziuni hepatice la doze mari');
INSERT INTO MEDICAMENTE (id_medicament, nume_medicament, cantitate, unitate_masura, efecte_secundare) VALUES (4, 'Penicilina', 50, 'mg', 'Reactii alergice, diaree');
INSERT INTO MEDICAMENTE (id_medicament, nume_medicament, cantitate, unitate_masura, efecte_secundare) VALUES (5, 'Ampicilina', 20, 'mg', 'Reactii alergice, diaree');
INSERT INTO MEDICAMENTE (id_medicament, nume_medicament, cantitate, unitate_masura, efecte_secundare) VALUES (6, 'Eritromicina', 100, 'mg', 'Diaree, dureri abdominale');
INSERT INTO MEDICAMENTE (id_medicament, nume_medicament, cantitate, unitate_masura, efecte_secundare) VALUES (7, 'Ciprofloxacina', 300, 'mg', 'Greata, diaree');
INSERT INTO MEDICAMENTE (id_medicament, nume_medicament, cantitate, unitate_masura, efecte_secundare) VALUES (8, 'Metronidazol', 150, 'mg', 'Greata, diaree, gust metalic');
```

```
INSERT INTO MEDICAMENTE (id_medicament, nume_medicament, cantitate, unitate_masura, efecte_secundare) VALUES (9, 'Doxiciclina', 200, 'mg', 'Greata, diaree, sensibilitate la lumina soarelui');

INSERT INTO MEDICAMENTE (id_medicament, nume_medicament, cantitate, unitate_masura, efecte_secundare) VALUES (10, 'Hidroxiclorochina', 300, 'mg', 'Diaree, dureri de stomac, dureri de cap');

INSERT INTO MEDICAMENTE (id_medicament, nume_medicament, cantitate, unitate_masura, efecte_secundare) VALUES (11, 'Azitromicina', 100, 'mg', 'Diaree, greata, dureri de stomac');

INSERT INTO MEDICAMENTE (id_medicament, nume_medicament, cantitate, unitate_masura, efecte_secundare) VALUES (12, 'Levofloxacina', 50, 'mg', 'Diaree, greata, dureri de stomac');

INSERT INTO MEDICAMENTE (id_medicament, nume_medicament, cantitate, unitate_masura, efecte_secundare) VALUES (13, 'Amoxicilina', 20, 'mg', 'Diaree, reactii alergice');

INSERT INTO MEDICAMENTE (id_medicament, nume_medicament, cantitate, unitate_masura, efecte_secundare) VALUES (14, 'Clindamycin', 10, 'mg', 'Diaree, dureri de stomac');

INSERT INTO MEDICAMENTE (id_medicament, nume_medicament, cantitate, unitate_masura, efecte_secundare) VALUES (15, 'Sulfamethoxazole', 15, 'mg', 'Diaree, greata, reactii alergice');

INSERT INTO MEDICAMENTE (id_medicament, nume_medicament, cantitate, unitate_masura, efecte_secundare) VALUES (16, 'Trimethoprim', 10, 'mg', 'Greata, dureri de stomac');

INSERT INTO MEDICAMENTE (id_medicament, nume_medicament, cantitate, unitate_masura, efecte_secundare) VALUES (17, 'Cefuroxime', 50, 'mg', 'Dureri de stomac, reactii alergice');

INSERT INTO MEDICAMENTE (id_medicament, nume_medicament, cantitate, unitate_masura, efecte_secundare) VALUES (18, 'Cefdinir', 20, 'mg', 'Dureri de stomac, reactii alergice');

INSERT INTO MEDICAMENTE (id_medicament, nume_medicament, cantitate, unitate_masura, efecte_secundare) VALUES (20, 'Cefaclor', 10, 'mg', 'Dureri de stomac, reactii alergice');

END;

/
```

```
Table MEDICAMENTE created.
```

```
PL/SQL procedure successfully completed.
```

	ID_MEDICAMENT	NUME_MEDICAMENT	CANTITATE	UNITATE_MASURA	EFECTE_SECUNDARE
1	1	Aspirin	200	mg	Iritatie la stomac, risc crescut de sangerare
2	2	Ibuprofen	200	mg	Iritatie la stomac, risc crescut de sangerare
3	3	Paracetamol	500	mg	Leziuni hepatice la doze mari
4	4	Penicilina	50	mg	Reactii alergice, diaree
5	5	Ampicilina	20	mg	Reactii alergice, diaree
6	6	Eritromicina	100	mg	Diaree, dureri abdominale
7	7	Ciprofloxacina	300	mg	Greata, diaree
8	8	Metronidazol	150	mg	Greata, diaree, gust metalic
9	9	Doxiciclina	200	mg	Greata, diaree, sensibilitate la lumina soarelui
10	10	Hidroxiclorochina	300	mg	Diaree, dureri de stomac, dureri de cap
11	11	Azitromicina	100	mg	Diaree, greata, dureri de stomac
12	12	Levofloxacina	50	mg	Diaree, greata, dureri de stomac
13	13	Amoxicilina	20	mg	Diaree, reactii alergice
14	14	Clindamycin	10	mg	Diaree, dureri de stomac
15	15	Sulfamethoxazole	15	mg	Diaree, greata, reactii alergice
16	16	Trimethoprim	10	mg	Greata, dureri de stomac
17	17	Cefuroxime	50	mg	Dureri de stomac, reactii alergice
18	18	Cefdinir	20	mg	Dureri de stomac, reactii alergice
19	20	Cefaclor	10	mg	Dureri de stomac, reactii alergice

```
CREATE TABLE MEDICAMENTE_PROGRAMARE
(
  id_medicament REFERENCES MEDICAMENTE(id_medicament),
  id_programare REFERENCES PROGRAMARI(id_programare)
);
```

```
BEGIN
INSERT INTO MEDICAMENTE_PROGRAMARE (id_medicament, id_programare) VALUES (1, 1);
INSERT INTO MEDICAMENTE_PROGRAMARE (id_medicament, id_programare) VALUES (2, 2);
```

INSERT INTO MEDICAMENTE_PROGRAMARE (id_medicament, id_programare) VALUES (3, 3);

INSERT INTO MEDICAMENTE_PROGRAMARE (id_medicament, id_programare) VALUES (4, 4);

INSERT INTO MEDICAMENTE_PROGRAMARE (id_medicament, id_programare) VALUES (5, 5);

INSERT INTO MEDICAMENTE_PROGRAMARE (id_medicament, id_programare) VALUES (6, 6);

INSERT INTO MEDICAMENTE_PROGRAMARE (id_medicament, id_programare) VALUES (7, 7);

INSERT INTO MEDICAMENTE_PROGRAMARE (id_medicament, id_programare) VALUES (8, 8);

INSERT INTO MEDICAMENTE_PROGRAMARE (id_medicament, id_programare) VALUES (9, 9);

INSERT INTO MEDICAMENTE_PROGRAMARE (id_medicament, id_programare) VALUES (10, 10);

INSERT INTO MEDICAMENTE_PROGRAMARE (id_medicament, id_programare) VALUES (11, 11);

INSERT INTO MEDICAMENTE_PROGRAMARE (id_medicament, id_programare) VALUES (12, 12);

INSERT INTO MEDICAMENTE_PROGRAMARE (id_medicament, id_programare) VALUES (13, 13);

INSERT INTO MEDICAMENTE_PROGRAMARE (id_medicament, id_programare) VALUES (14, 14);

INSERT INTO MEDICAMENTE_PROGRAMARE (id_medicament, id_programare) VALUES (15, 15);

INSERT INTO MEDICAMENTE_PROGRAMARE (id_medicament, id_programare) VALUES (16, 16);

INSERT INTO MEDICAMENTE_PROGRAMARE (id_medicament, id_programare) VALUES (17, 17);

INSERT INTO MEDICAMENTE_PROGRAMARE (id_medicament, id_programare) VALUES (18, 18);

INSERT INTO MEDICAMENTE_PROGRAMARE (id_medicament, id_programare) VALUES (20, 19);

```
INSERT INTO MEDICAMENTE_PROGRAMARE (id_medicament, id_programare) VALUES
(20, 20);

INSERT INTO MEDICAMENTE_PROGRAMARE (id_medicament, id_programare) VALUES
(1, 10);

INSERT INTO MEDICAMENTE_PROGRAMARE (id_medicament, id_programare) VALUES
(2, 11);

INSERT INTO MEDICAMENTE_PROGRAMARE (id_medicament, id_programare) VALUES
(3, 12);

INSERT INTO MEDICAMENTE_PROGRAMARE (id_medicament, id_programare) VALUES
(4, 13);

INSERT INTO MEDICAMENTE_PROGRAMARE (id_medicament, id_programare) VALUES
(5, 14);


END;

/
```

```
Table MEDICAMENTE_PROGRAMARE created.


PL/SQL procedure successfully completed.
```

	ID_MEDICAMENT	ID_PROGRAMARE
1	1	1
2	2	2
3	3	3
4	4	4
5	5	5
6	6	6
7	7	7
8	8	8
9	9	9
10	10	10
11	11	11
12	12	12
13	13	13
14	14	14
15	15	15
16	16	16
17	17	17
18	18	18
19	20	19
20	20	20
21	1	10
22	2	11
23	3	12
24	4	13
25	5	14

```
CREATE TABLE PLATA (
    id_plata NUMBER(3) PRIMARY KEY,
    id_programare NUMBER(3)  REFERENCES PROGRAMARI(id_programare),
    modalitate_plata VARCHAR(50),
    suma NUMBER(15),
    FOREIGN KEY (id_programare) REFERENCES PROGRAMARI(id_programare)
);
```

```
BEGIN
INSERT INTO PLATA (id_plata, id_programare, modalitate_plata, suma) VALUES (1, 1, 'Cash', 300);
INSERT INTO PLATA (id_plata, id_programare, modalitate_plata, suma) VALUES (2, 2, 'Card de Credit', 500);
INSERT INTO PLATA (id_plata, id_programare, modalitate_plata, suma) VALUES (3, 3, 'Card de Debit', 250);
INSERT INTO PLATA (id_plata, id_programare, modalitate_plata, suma) VALUES (4, 4, 'Transfer Bancar Online', 450);
INSERT INTO PLATA (id_plata, id_programare, modalitate_plata, suma) VALUES (5, 5, 'Cash', 520);
INSERT INTO PLATA (id_plata, id_programare, modalitate_plata, suma) VALUES (6, 6, 'Card de Credit', 480);
INSERT INTO PLATA (id_plata, id_programare, modalitate_plata, suma) VALUES (7, 7, 'Card de Debit', 230);
INSERT INTO PLATA (id_plata, id_programare, modalitate_plata, suma) VALUES (8, 8, 'Transfer Bancar Online', 520);
INSERT INTO PLATA (id_plata, id_programare, modalitate_plata, suma) VALUES (9, 9, 'Cash', 650);
INSERT INTO PLATA (id_plata, id_programare, modalitate_plata, suma) VALUES (10, 10, 'Card de Credit', 400);
```

```
INSERT INTO PLATA (id_plata, id_programare, modalitate_plata, suma) VALUES (11, 11, 'Card de Debit', 330);

INSERT INTO PLATA (id_plata, id_programare, modalitate_plata, suma) VALUES (12, 12, 'Transfer Bancar Online', 250);

INSERT INTO PLATA (id_plata, id_programare, modalitate_plata, suma) VALUES (13, 13, 'Cash', 400);

INSERT INTO PLATA (id_plata, id_programare, modalitate_plata, suma) VALUES (14, 14, 'Card de Credit', 280);

INSERT INTO PLATA (id_plata, id_programare, modalitate_plata, suma) VALUES (15, 15, 'Card de Debit', 350);

INSERT INTO PLATA (id_plata, id_programare, modalitate_plata, suma) VALUES (16, 16, 'Transfer Bancar Online', 250);

INSERT INTO PLATA (id_plata, id_programare, modalitate_plata, suma) VALUES (17, 17, 'Cash', 420);

INSERT INTO PLATA (id_plata, id_programare, modalitate_plata, suma) VALUES (18, 18, 'Card de Credit', 350);

INSERT INTO PLATA (id_plata, id_programare, modalitate_plata, suma) VALUES (19, 19, 'Card de Debit', 230);

INSERT INTO PLATA (id_plata, id_programare, modalitate_plata, suma) VALUES (20, 20, 'Transfer Bancar Online', 220);

END;

/
```

```
Table PLATA created.


PL/SQL procedure successfully completed.
```

	ID_PLATA	ID_PROGRAMARE	MODALITATE_PLATA	SUMA
1	1	1	Cash	300
2	2	2	Card de Credit	500
3	3	3	Card de Debit	250
4	4	4	Transfer Bancar Online	450
5	5	5	Cash	520
6	6	6	Card de Credit	480
7	7	7	Card de Debit	230
8	8	8	Transfer Bancar Online	520
9	9	9	Cash	650
10	10	10	Card de Credit	400
11	11	11	Card de Debit	330
12	12	12	Transfer Bancar Online	250
13	13	13	Cash	400
14	14	14	Card de Credit	280
15	15	15	Card de Debit	350
16	16	16	Transfer Bancar Online	250
17	17	17	Cash	420
18	18	18	Card de Credit	350
19	19	19	Card de Debit	230
20	20	20	Transfer Bancar Online	220

7. **Scrierea a 15 interogări, cât mai complexe, care să ilustreze toate aspectele învățate din lista (formulați în limbaj natural problemele ce urmează a fi rezolvate):**

1) **Returnează salariul total al tuturor medicilor, grupați pe specialitate, care au lucrat mai mult de un anumit număr de luni și au un salariu mai mare decât o anumită sumă.**

```
SELECT specialitate, SUM(salariu) as SALARIU_TOTAL,
MONTHS_BETWEEN(SYSDATE, data_angajare) AS LUNI_LUCRATE
FROM doctori
GROUP BY specialitate, MONTHS_BETWEEN(SYSDATE, data_angajare)
HAVING MONTHS_BETWEEN(SYSDATE, data_angajare) > 12
AND SUM(salariu) > 20000
```

	SPECIALITATE	SALARIU_TOTAL	LUNI_LUCRATE
1	Radiologie	50000	49.086109244324970131421744324970131421744
2	Ortopedie	25000	45.408689889486260454002389486260454002389
3	Anesteziologie	30000	28.440947954002389486260454002389486260454
4	Neurologie	35000	24.118367308841099163679808841099163679808
5	Gastroenterologie	40000	36.150625373357228195937873357228195937873
6	Dermatologie	45000	49.763528599163679808841099163679808841099

2) **Returnează numele pacientului, numele medicamentului, cantitatea și unitatea de măsură a medicamentului pentru toți pacienții cărora li s-a prescris un anumit medicament, iar numele medicamentului conține un cuvânt specific, ordonat după cantitatea de medicament în ordine descrescătoare.**

```
SELECT nume_pacient, nume_medicament, cantitate, unitate_masura
FROM MEDICAMENTE_PROGRAMARE
JOIN PROGRAMARI ON MEDICAMENTE_PROGRAMARE.id_programare =
PROGRAMARI.id_programare
JOIN PACIENTI ON PROGRAMARI.id_pacient = PACIENTI.id_pacient
JOIN MEDICAMENTE ON MEDICAMENTE_PROGRAMARE.id_medicament =
MEDICAMENTE.id_medicament
WHERE INSTR(nume_medicament, 'cili') > 0
ORDER BY cantitate DESC
```

	NUME_PACIENT	NUME_MEDICAMENT	CANTITATE	UNITATE_MASURA
1	Sarah Taylor	Penicilina	50	mg
2	Linda Williams	Penicilina	50	mg
3	Sophia Perez	Ampicilina	20	mg
4	Elizabeth Brown	Ampicilina	20	mg
5	Sarah Taylor	Amoxicilina	20	mg

3) **Găsirea medicului asociat fiecărui pacient și preluarea informațiilor suplimentare, precum numele pacientului, primele 3 cifre ale numărului de telefon al pacientului și ultimele 4 cifre ale numărului de telefon al medicului. Interogarea ordonează și rezultatul după medicul asociat.**

```
WITH PACIENTI_DOCTORI AS (
  SELECT PACIENTI.id_pacient, CONNECT_BY_ROOT (DOCTORI.nume_doctor) AS
doctor_asociat_pacientului
  FROM PROGRAMARI
  JOIN PACIENTI ON PACIENTI.id_pacient = PROGRAMARI.id_pacient
  JOIN DOCTORI ON DOCTORI.id_doctor = PROGRAMARI.id_doctor
  START WITH PACIENTI.id_pacient = PACIENTI.id_pacient
  CONNECT BY NOCYCLE PACIENTI.id_pacient = PRIOR PACIENTI.id_pacient
)
SELECT nume_pacient, doctor_asociat_pacientului, SUBSTR(PACIENTI.telefon, 1, 3) AS
telefon_pacient, SUBSTR(DOCTORI.telefon, -4) AS telefon_doctor
FROM PACIENTI_DOCTORI
LEFT JOIN PACIENTI
ON PACIENTI_DOCTORI.id_pacient = PACIENTI.id_pacient
LEFT JOIN DOCTORI
ON PACIENTI_DOCTORI.doctor_asociat_pacientului = DOCTORI.nume_doctor
ORDER BY doctor_asociat_pacientului;
```

	NUME_PACIENT	DOCTOR_ASOCIAT_PACIENTULUI	TELEFON_PACIENT	TELEFON_DOCTOR
1	Linda Williams	Alisia Brown	456	0123
2	Sophia Perez	Alisia Brown	901	0123
3	Nancy Brown	Andrew Robinson	567	6789
4	Jennifer Johnson	Bob Johnson	345	9012
5	Sarah Taylor	Bob Johnson	890	9012
6	James Doe	John Doe	123	7890
7	Cynthia Pop	John Doe	234	7890
8	Kimberly Davis	John Doe	678	7890
9	Joseph Brown	Joshua Garcia	234	3456
10	Jessica Lopez	Joshua Garcia	234	3456
11	George Rodriguez	Maddie Bryan	234	4567
12	Charles Johnson	Maddie Bryan	345	4567
13	Daniel Williams	Matthew Pop	456	5678
14	Timothy Brian	Matthew Pop	234	5678
15	Paul Garcia	Michael Brown	012	1234
16	Elizabeth Brown	Michael Brown	567	1234
17	Donna Martinez	Patrick Johnas	123	2345
18	Barbara Smith	Patrick Johnas	123	2345
19	Patricia Smith	Richard David	234	8901
20	Karen Garcia	Richard David	789	8901

4) Preluarea numărului de programări pentru fiecare medic în ultimele 3 luni și gruparea după nume și specialitatea medicului.

```
SELECT nume_doctor, specialitate, COUNT(id_programare) as NUMAR_PROGRAMARI
FROM DOCTORI
JOIN PROGRAMARI
ON DOCTORI.id_doctor = PROGRAMARI.id_doctor
WHERE data_programare >= ADD_MONTHS(TO_DATE(TO_CHAR(SYSDATE,
'YYYY-MM-DD'), 'YYYY-MM-DD'), -3)
GROUP BY nume_doctor, specialitate
```

	NUME_DOCTOR	SPECIALITATE	NUMAR_PROGRAMARI
1	Bob Johnson	Cardiologie	2
2	Alisia Brown	Oncologie	2
3	Richard David	Pediatrie	2
4	Joshua Garcia	Neurologie	2
5	Maddie Bryan	Gastroenterologie	2
6	John Doe	Chirurgie Plastica	3
7	Matthew Pop	Dermatologie	2
8	Patrick Johnas	Anesteziologie	2
9	Andrew Robinson	Radiologie	1
10	Michael Brown	Ortopedie	2

5) **Afișarea numelor pacienților care începe cu litera J și informațiile de asigurare ale acestora.**

```
SELECT nume_pacient, asigurare_info
FROM PACIENTI
JOIN ASIGURARE
ON PACIENTI.id_asigurare = ASIGURARE.id_asigurare
WHERE UPPER(nume_pacient) LIKE 'J%'
```

	NUME_PACIENT	ASIGURARE_INFO
1	James Doe	Blue Cross Blue Shield
2	Jessica Lopez	Aetna
3	Jennifer Johnson	United Healthcare
4	Joseph Brown	Health Net

6) **Afișarea medicamentului, unde efectele secundare conțin cuvântul "dureri", cantitatea totală folosită și numărul de programări în care s-a dat spre folosință medicamentul respectiv.**

```
SELECT nume_medicament, SUM(cantitate), COUNT(id_programare)
FROM MEDICAMENTE
JOIN MEDICAMENTE_PROGRAMARE
ON MEDICAMENTE.id_medicament =
MEDICAMENTE_PROGRAMARE.id_medicament
WHERE LOWER(efecte_secundare) LIKE '%dureri%'
GROUP BY nume_medicament
```

	NUME_MEDICAMENT	SUM(CANTITATE)	COUNT(ID_PROGRAMARE)
1	Hidroxiclorochina	300	1
2	Levofloxacina	50	1
3	Cefuroxime	50	1
4	Cefaclor	10	1
5	Clindamycin	10	1
6	Trimethoprim	10	1
7	Cefdinir	20	1
8	Eritromicina	100	1
9	Azitromicina	100	1

7) **Afișarea numelor pacienților, a numelui medicului, dar și starea medicală actuală a fiecărui pacient. Dacă id_conditie este 10, va afișa “TRATAT”, dacă id_conditie este 2 va afișa “INCURABIL” și dacă nu este nici una dintre acestea va afișa “IN CURS DE VERIFICARE”.**

```
SELECT PACIENTI.nume_pacient, DOCTORI.nume_doctor,
DECODE(CONDITIE_PACIENTI.id_conditie, 10, 'TRATAT', 2, 'INCURABIL', 'IN CURS DE VERIFICARE') as STARE_PACIENTI
FROM PROGRAMARI
JOIN PACIENTI ON PACIENTI.id_pacient = PROGRAMARI.id_pacient
JOIN DOCTORI ON DOCTORI.id_doctor = PROGRAMARI.id_doctor
JOIN CONDITIE_PACIENTI ON PACIENTI.id_pacient = CONDITIE_PACIENTI.id_pacient;
```

	NUME_PACIENT	NUME_DOCTOR	STARE_PACIENTI
1	James Doe	John Doe	TRATAT
2	Patricia Smith	Richard David	IN CURS DE VERIFICARE
3	Jennifer Johnson	Bob Johnson	IN CURS DE VERIFICARE
4	Linda Williams	Alisia Brown	IN CURS DE VERIFICARE
5	Elizabeth Brown	Michael Brown	IN CURS DE VERIFICARE
6	Barbara Smith	Patrick Johnas	IN CURS DE VERIFICARE
7	Joseph Brown	Joshua Garcia	IN CURS DE VERIFICARE
8	Charles Johnson	Maddie Bryan	IN CURS DE VERIFICARE
9	Daniel Williams	Matthew Pop	IN CURS DE VERIFICARE
10	Nancy Brown	Andrew Robinson	IN CURS DE VERIFICARE
11	Kimberly Davis	John Doe	IN CURS DE VERIFICARE
12	Karen Garcia	Richard David	IN CURS DE VERIFICARE
13	Sarah Taylor	Bob Johnson	IN CURS DE VERIFICARE
14	Sophia Perez	Alisia Brown	IN CURS DE VERIFICARE
15	Paul Garcia	Michael Brown	IN CURS DE VERIFICARE
16	Donna Martinez	Patrick Johnas	IN CURS DE VERIFICARE
17	Jessica Lopez	Joshua Garcia	IN CURS DE VERIFICARE
18	George Rodriguez	Maddie Bryan	IN CURS DE VERIFICARE
19	Timothy Brian	Matthew Pop	IN CURS DE VERIFICARE
20	Cynthia Pop	John Doe	INCURABIL

8) Afișarea numelor tuturor pacienților, numele medicului care i-a tratat ultima dată, data ultimei vizite, salariul minim și mediu al tuturor medicilor care au tratat pacientul, indicând salariul minim și mediu ca fiind 0 dacă pacientul nu are antecedente de tratament.

```
SELECT p.nume_pacient,
CASE
WHEN pr.id_programare IS NULL THEN 'Nu are un istoric al tratamentelor'
ELSE d.nume_doctor
END AS "Doctor",
MAX(pr.data_programare) as "Data ultimei vizite",
MIN(NULLIF(d.salariu, 0)) as "SALARIU MINIM",
AVG(NULLIF(d.salariu, 0)) as "SALARIU MEDIU"
FROM PROGRAMARI pr
RIGHT JOIN PACIENTI p ON p.id_pacient = pr.id_pacient
RIGHT JOIN DOCTORI d ON pr.id_doctor = d.id_doctor
GROUP BY p.nume_pacient,
CASE
WHEN pr.id_programare IS NULL THEN 'Nu are un istoric al tratamentelor'
ELSE d.nume_doctor
END;
```

	NUME_PACIENT	Doctor	Data ultimei vizite	SALARIU MINIM	SALARIU MEDIU
1	James Doe	John Doe	01-FEB-23	10000	10000
2	Sophia Perez	Alisia Brown	14-FEB-23	20000	20000
3	Elizabeth Brown	Michael Brown	05-FEB-23	25000	25000
4	Jessica Lopez	Joshua Garcia	17-FEB-23	35000	35000
5	Timothy Brian	Matthew Pop	19-FEB-23	45000	45000
6	Patricia Smith	Richard David	02-FEB-23	12000	12000
7	Nancy Brown	Andrew Robinson	10-FEB-23	50000	50000
8	Karen Garcia	Richard David	12-FEB-23	12000	12000
9	Jennifer Johnson	Bob Johnson	03-FEB-23	15000	15000
10	Barbara Smith	Patrick Johnas	06-FEB-23	30000	30000
11	Charles Johnson	Maddie Bryan	08-FEB-23	40000	40000
12	Daniel Williams	Matthew Pop	09-FEB-23	45000	45000
13	Joseph Brown	Joshua Garcia	07-FEB-23	35000	35000
14	Kimberly Davis	John Doe	11-FEB-23	10000	10000
15	Donna Martinez	Patrick Johnas	16-FEB-23	30000	30000
16	Linda Williams	Alisia Brown	04-FEB-23	20000	20000
17	Sarah Taylor	Bob Johnson	13-FEB-23	15000	15000
18	Paul Garcia	Michael Brown	15-FEB-23	25000	25000
19	George Rodriguez	Maddie Bryan	18-FEB-23	40000	40000
20	Cynthia Pop	John Doe	20-FEB-23	10000	10000

9) **Afișarea numărului de medici și camere diferite vizitate de fiecare pacient, chiar dacă pacientul nu a fost încă tratat. Se afișează, așadar, numele tuturor pacienților, iar pentru fiecare pacient numărul de medici diferiți vizitați, cât și numărul de camere diferite vizitate.**

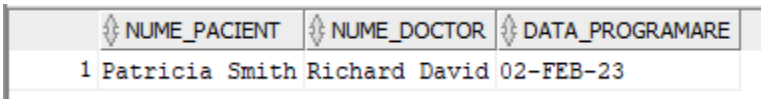
```
SELECT DISTINCT(p.nume_pacient),
    NVL((SELECT COUNT(pr.id_doctor) FROM PROGRAMARI pr WHERE
pr.id_pacient = p.id_pacient),0) as "Doctori diferiti vizitati",
    NVL((SELECT COUNT(p.id_salon) FROM PROGRAMARI pr WHERE pr.id_pacient
= p.id_pacient),0) as "Camere diferite vizitate"
FROM PROGRAMARI pr
JOIN PACIENTI p on p.id_pacient = pr.id_pacient
```

	NUME_PACIENT	Doctori diferiti vizitati	Camere diferite vizitate
1	Joseph Brown	1	1
2	Charles Johnson	1	1
3	Kimberly Davis	1	1
4	Cynthia Pop	2	2
5	James Doe	1	1
6	Patricia Smith	1	1
7	Karen Garcia	1	1
8	Paul Garcia	1	1
9	George Rodriguez	1	1
10	Daniel Williams	1	1
11	Linda Williams	1	1
12	Elizabeth Brown	1	1
13	Timothy Brian	1	1
14	Nancy Brown	1	1
15	Sarah Taylor	1	1
16	Sophia Perez	1	1
17	Jessica Lopez	1	1
18	Jennifer Johnson	1	1
19	Barbara Smith	1	1
20	Donna Martinez	1	1
21	Sassy Green	2	2

10) **Afișarea numelui pacientului, a numelui medicului și următoarea programare a pacientului folosind o subinterogare în clauza SELECT.**

```
SELECT nume_pacient, nume_doctor, data_programare
FROM PACIENTI
```

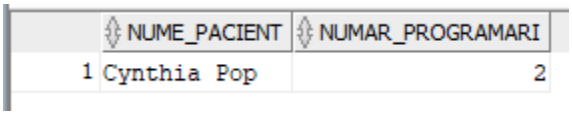
```
JOIN PROGRAMARI
ON PACIENTI.id_pacient = PROGRAMARI.id_pacient
JOIN DOCTORI
ON PROGRAMARI.id_doctor = DOCTORI.id_doctor
WHERE PROGRAMARI.id_pacient = (SELECT id_pacient FROM PACIENTI
WHERE nume_pacient = 'Patricia Smith')
```



	NUME_PACIENT	NUME_DOCTOR	DATA_PROGRAMARE
1	Patricia Smith	Richard David	02-FEB-23

11) Se afișează numai pacienții care au programat mai multe întâlniri decât numărul mediu de întâlniri programate de toți pacienții.

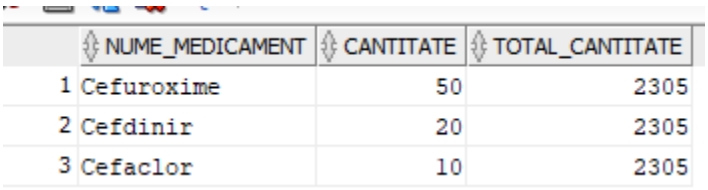
```
SELECT nume_pacient, COUNT(id_programare) as numar_programari
FROM PACIENTI
JOIN PROGRAMARI
ON PACIENTI.id_pacient = PROGRAMARI.id_pacient
GROUP BY nume_pacient
HAVING COUNT(id_programare) > (SELECT AVG(numar_programari) FROM
(SELECT COUNT(id_programare) as numar_programari FROM PROGRAMARI
GROUP BY id_pacient))
```



	NUME_PACIENT	NUMAR_PROGRAMARI
1	Cynthia Pop	2

12) Afișarea numelui, cantității și cantitatea totală a tuturor medicamentelor care au ca efect secundar 'Dureri de stomac, reactii alergice'.

```
SELECT nume_medicament, cantitate, (SELECT SUM(cantitate) FROM MEDICAMENTE) as
total_cantitate
FROM MEDICAMENTE
WHERE efecte_secundare = 'Dureri de stomac, reactii alergice'
```



	NUME_MEDICAMENT	CANTITATE	TOTAL_CANTITATE
1	Cefuroxime	50	2305
2	Cefdinir	20	2305
3	Cefaclor	10	2305

13) Afișarea sumei costului tuturor programărilor este împărțită la numărul de programară pentru a calcula costul mediu pe programare pentru fiecare medic.

```
SELECT nume_doctor, SUM(plata.suma) / COUNT(programari.id_programare) as
cost_mediu_per_programare
FROM DOCTORI
JOIN PROGRAMARI
ON DOCTORI.id_doctor = PROGRAMARI.id_doctor
JOIN PLATA
ON PROGRAMARI.id_programare = PLATA.id_programare
GROUP BY nume_doctor;
```

	NUME_DOCTOR	COST_MEDIU_PER_PROGRAMARE
1	Andrew Robinson	400
2	Alisia Brown	365
3	Richard David	375
4	Maddie Bryan	435
5	Bob Johnson	325
6	Michael Brown	435
7	Patrick Johnas	365
8	Joshua Garcia	325
9	Matthew Pop	440
10	John Doe	283.33333333333333333333333333333333333

14) Afișarea pacienților și a condiției medicale pe care aceștia o au numai dacă descrierea condiției medicale conține cuvântul 'infecție'

```
SELECT PACIENTI.nume_pacient, CONDITIE_MEDICALA.nume_conditie
FROM (
  SELECT id_conditie, nume_conditie FROM CONDITIE_MEDICALA WHERE descriere
LIKE '%infectie%'
) pacient_conditions
JOIN CONDITIE_PACIENTI ON pacient_conditions.id_conditie =
CONDITIE_PACIENTI.id_conditie
JOIN PACIENTI ON CONDITIE_PACIENTI.id_pacient = PACIENTI.id_pacient
JOIN CONDITIE_MEDICALA ON pacient_conditions.id_conditie =
CONDITIE_MEDICALA.id_conditie;
```

	NUME_PACIENT	NUME_CONDITIE
1	James Doe	Gripa tip A
2	Patricia Smith	Gripa tip B
3	Daniel Williams	Pneumonie
4	Nancy Brown	Tuberculoza
5	Kimberly Davis	Gripa tip B
6	Karen Garcia	Pojar
7	Sarah Taylor	Varicelă

15) Se afișează numele pacientului, informațiile de asigurare, numele medicamentului și efectele secundare ale medicamentului.

```
SELECT nume_pacient, asigurare_info, nume_medicament, efecte_secundare
FROM PACIENTI
FULL JOIN ASIGURARE ON PACIENTI.id_asigurare = ASIGURARE.id_asigurare
FULL JOIN PROGRAMARI ON PACIENTI.id_pacient = PROGRAMARI.id_pacient
FULL JOIN MEDICAMENTE_PROGRAMARE ON PROGRAMARI.id_programare =
MEDICAMENTE_PROGRAMARE.id_programare
FULL JOIN MEDICAMENTE ON MEDICAMENTE_PROGRAMARE.id_medicament
= MEDICAMENTE.id_medicament
```

	NUME_PACIENT	ASIGURARE_INFO	NUME_MEDICAMENT	EFECTE_SECUNDARE
1	James Doe	Blue Cross Blue Shield	Aspirin	Iritatie la stomac, risc crescut de sangerare
2	Patricia Smith	Aetna	Ibuprofen	Iritatie la stomac, risc crescut de sangerare
3	Jennifer Johnson	United Healthcare	Paracetamol	Leziuni hepatice la doze mari
4	Linda Williams	Humana	Penicilina	Reactii alergice, diaree
5	Elizabeth Brown	Cigna	Ampicilina	Reactii alergice, diaree
6	Barbara Smith	Kaiser Permanente	Eritromicina	Diaree, dureri abdominale
7	Joseph Brown	Health Net	Ciprofloxacina	Greata, diaree
8	Charles Johnson	Medicaid	Metronidazol	Greata, diaree, gust metalic
9	Daniel Williams	Medicare	Doxiciclina	Greata, diaree, sensibilitate la lumina soarelui
10	Nancy Brown	Amerigroup	Hidroxiclorochina	Diaree, dureri de stomac, dureri de cap
11	Kimberly Davis	Blue Cross Blue Shield	Azitromicina	Diaree, greata, dureri de stomac
12	Karen Garcia	Aetna	Levofloxacina	Diaree, greata, dureri de stomac
13	Sarah Taylor	United Healthcare	Amoxicilina	Diaree, reactii alergice
14	Sophia Perez	Humana	Clindamycin	Diaree, dureri de stomac
15	Paul Garcia	Cigna	Sulfamethoxazole	Diaree, greata, reactii alergice
16	Donna Martinez	Blue Cross Blue Shield	Trimethoprim	Greata, dureri de stomac
17	Jessica Lopez	Aetna	Cefuroxime	Dureri de stomac, reactii alergice
18	George Rodriguez	United Healthcare	Cefdinir	Dureri de stomac, reactii alergice
19	Cynthia Pop	Cigna	Cefaclor	Dureri de stomac, reactii alergice
20	Nancy Brown	Amerigroup	Aspirin	Iritatie la stomac, risc crescut de sangerare
21	Kimberly Davis	Blue Cross Blue Shield	Ibuprofen	Iritatie la stomac, risc crescut de sangerare
22	Karen Garcia	Aetna	Paracetamol	Leziuni hepatice la doze mari
23	Sarah Taylor	United Healthcare	Penicilina	Reactii alergice, diaree
24	Sophia Perez	Humana	Ampicilina	Reactii alergice, diaree
25	Cynthia Pop	Cigna	(null)	(null)
26	Timothy Brian	Humana	(null)	(null)

8. Crearea unui tabel de mesaje

```
CREATE TABLE MESAJE (
message_id NUMBER PRIMARY KEY,
message VARCHAR(255),
message_type VARCHAR2(1) CHECK (message_type IN ('E', 'W', 'I')),
created_by VARCHAR2(40) NOT NULL,
created_at DATE NOT NULL
);
```

```
Table MESAJE created.
```

9. Ilustrarea noțiunilor de PL/SQL

- ***Subprogram stocat independent(inclusiv apelare) care să utilizeze 2 tipuri de colecție învățate***

Prin intermediul acestei proceduri sunt afișate, pentru fiecare asigurare, pacienții care o au și numărul de programări pe care îl are fiecare. Pentru afișarea datelor, a fost necesară folosirea a două tablouri imbricate, în cadrul cărora au fost stocate id-ul și numele asigurării, iar în cadrul tabloului indexat au fost stocate informații despre pacienți, precum: id-ul pacientului, numele și id-ul asigurării.

```
CREATE OR REPLACE PROCEDURE proc2 IS
  TYPE tab_imb_id_asig IS TABLE OF ASIGURARE.id_asigurare%TYPE;
  t_id_asig tab_imb_id_asig := tab_imb_id_asig();

  TYPE tab_imb_det_asiginfo IS TABLE OF ASIGURARE.asigurare_info%TYPE;
  t_det_asiginfo tab_imb_det_asiginfo := tab_imb_det_asiginfo();

  TYPE pacienti_inreg IS RECORD
```

```
    (id_pacient PACIENTI.id_pacient%TYPE,
    nume_pacient PACIENTI.nume_pacient%TYPE,
    id_asigurare ASIGURARE.id_asigurare%TYPE);

  TYPE tab_pacienti IS TABLE OF pacienti_inreg INDEX BY BINARY_INTEGER;
  t_pac tab_pacienti;

  i INTEGER;
  cont INTEGER;
  j INTEGER;
  nr_programari INTEGER;

BEGIN
  SELECT id_asigurare
  BULK COLLECT INTO t_id_asig
  FROM ASIGURARE
  ORDER BY id_asigurare;


  SELECT asigurare_info
  BULK COLLECT INTO t_det_asiginfo
  FROM ASIGURARE
  ORDER BY id_asigurare;

  i := t_id_asig.FIRST;
  j := t_det_asiginfo.FIRST;
```

```
    WHILE i <= t_id_asig.LAST LOOP
        DBMS_OUTPUT.PUT_LINE('Asigurarea cu id-ul '||t_id_asig(i)||'
('||t_det_asiginfo(j)||')');



        SELECT id_pacient, nume_pacient, ASIGURARE.id_asigurare
        BULK COLLECT INTO t_pac
        FROM PACIENTI
        JOIN ASIGURARE ON PACIENTI.id_asigurare = ASIGURARE.id_asigurare
        WHERE ASIGURARE.id_asigurare = t_id_asig(i);


           IF t_pac.COUNT = 0 THEN
        DBMS_OUTPUT.PUT_LINE('    '||'Niciun pacient nu are acest tip de asigurare!');
        DBMS_OUTPUT.NEW_LINE;
        ELSE
        IF t_pac.COUNT >= 1 THEN
            DBMS_OUTPUT.PUT_LINE('    '||'Pacientii care au acest tip de asigurare:');
            DBMS_OUTPUT.NEW_LINE;
        ELSE
            DBMS_OUTPUT.PUT_LINE('    '||'Pacientul care are acest tip de asigurare:');
            DBMS_OUTPUT.NEW_LINE;
        END IF;


        cont := 0;
        FOR k IN t_pac.FIRST..t_pac.LAST LOOP
            cont := cont+1;
            DBMS_OUTPUT.PUT('        '||cont||'. '||t_pac(k).nume_pacient);
```

```
            SELECT COUNT(*)
            INTO nr_programari
            FROM  PROGRAMARI
            WHERE id_pacient = t_pac(k).id_pacient;

            DBMS_OUTPUT.PUT_LINE(' | numar de programari: '||nr_programari);
          END LOOP;
       END IF;

       i:=i+1;
       j:=j+1;
       DBMS_OUTPUT.NEW_LINE;
       DBMS_OUTPUT.NEW_LINE;
    END LOOP;

END proc2;


SET SERVEROUTPUT ON;
BEGIN
    proc2;
END;
```

```
Asigurarea cu id-ul 5 (Cigna)
    Pacientii care au acest tip de asigurare:

        1. Elizabeth Brown | numar de programari: 1
        2. Paul Garcia | numar de programari: 1
        3. Cynthia Pop | numar de programari: 2
        4. Mara Popescu | numar de programari: 0


Asigurarea cu id-ul 6 (Kaiser Permanente)
    Pacientii care au acest tip de asigurare:

        1. Barbara Smith | numar de programari: 1


Asigurarea cu id-ul 7 (Health Net)
    Pacientii care au acest tip de asigurare:

        1. Joseph Brown | numar de programari: 1


Asigurarea cu id-ul 8 (Medicare)
    Pacientii care au acest tip de asigurare:

        1. Daniel Williams | numar de programari: 1


Asigurarea cu id-ul 9 (Medicaid)
    Pacientii care au acest tip de asigurare:

        1. Charles Johnson | numar de programari: 1


Asigurarea cu id-ul 10 (Amerigroup)
    Pacientii care au acest tip de asigurare:

        1. Nancy Brown | numar de programari: 1
        2. Sassy Green | numar de programari: 2
```

- ***Subprogram stocat independent(inclusiv apelare) care să utilizeze 2 tipuri de cursoare învățate, unul dintre acestea fiind cursor parametrizat;***

Această procedură afișează numărul medicilor din București, iar pentru specialitatea "Dermatologie" afișează numele medicului și salariu anual al acestuia.

```
CREATE OR REPLACE PROCEDURE proc1 AS
CURSOR c1 IS
SELECT COUNT(id_doctor) FROM DOCTORI WHERE adresa = 'Bucuresti';

cursor c2(p_specialitate varchar2) IS
SELECT nume_doctor, salariu*12 anual_sal
FROM DOCTORI
WHERE specialitate = p_specialitate;

v1 NUMBER;
v2 c2%ROWTYPE;

BEGIN
OPEN c1;
LOOP
FETCH c1 INTO v1;
EXIT WHEN c1%NOTFOUND;
DBMS_OUTPUT.PUT_LINE('Exista: '|| v1 ||' doctori in Bucuresti');
END LOOP;
CLOSE c1;

OPEN c2('Dermatologie');
FETCH c2 INTO v2;
WHILE (c2%FOUND) LOOP
```

```
DBMS_OUTPUT.PUT_LINE ('Doctorul: ' || v2.nume_doctor ||
'are salariul anual : ' || v2.anual_sal);
FETCH c2 INTO v2;
END LOOP;
CLOSE c2;
END;

EXECUTE proc1;
```

```
Exista: 7 doctori in Bucuresti
Doctorul: Dr. Matthew Popare salariul anual : 540000


PL/SQL procedure successfully completed.
```

- ***Subprogram stocat independent de tip funcție, care să utilizeze într-o singură comandă SQL 3 dintre tabelele definite; tratarea tuturor excepțiilor care pot apărea(definiți minim 2 excepții); apelarea subprogramului astfel încât să fie evidențiate toate cazurile tratate***

Afișează numele pacientului, informații despre asigurarea acestuia și starea medicala pentru un anumit pacient al cărui ID este trecut ca parametru.

```
CREATE OR REPLACE FUNCTION pacienti_info (p_id_pacient NUMBER)
RETURN VARCHAR2
AS
v_nume_pacient VARCHAR2(50);
v_nume_asigurare VARCHAR2(50);
v_nume_conditie VARCHAR2(50);
v_eroare VARCHAR2(255);
BEGIN
BEGIN
```

```
SELECT nume_pacient, asigurare_info
INTO v_nume_pacient, v_nume_asigurare
FROM PACIENTI p
JOIN ASIGURARE a ON p.id_asigurare = a.id_asigurare
WHERE p.id_pacient = p_id_pacient;
EXCEPTION
WHEN NO_DATA_FOUND THEN
v_eroare := 'Pacientul cu ID-ul ' || p_id_pacient || ' nu a fost gasit in baza de date.';
RAISE_APPLICATION_ERROR(-20001, v_eroare);
END;

BEGIN
SELECT nume_conditie
INTO v_nume_conditie
FROM CONDITIE_MEDICALA c
JOIN CONDITIE_PACIENTI cp ON c.id_conditie = cp.id_conditie
WHERE cp.id_pacient = p_id_pacient;
EXCEPTION
WHEN NO_DATA_FOUND THEN
v_eroare := 'Nu exista conditii medicale inregistrate pentru pacientul cu ID-ul ' || p_id_pacient;
RAISE_APPLICATION_ERROR(-20002, v_eroare);
END;

RETURN v_nume_pacient || ' ' || v_nume_asigurare || ' ' || v_nume_conditie;
END;

BEGIN
dbms_output.put_line(retrieve_patient_info(100));
```

END;

/

```
Error starting at line : 666 in command -
BEGIN
dbms_output.put_line(retrieve_patient_info(100));
END;
Error report -
ORA-20001: Pacientul cu ID-ul 100 nu a fost gasit in baza de date.
ORA-06512: at "SYSTEM.RETRIEVE_PATIENT_INFO", line 18
ORA-06512: at line 2
```

BEGIN

dbms_output.put_line(retrieve_patient_info(21));

END;

/

```
Error starting at line : 666 in command -
BEGIN
dbms_output.put_line(retrieve_patient_info(21));
END;
Error report -
ORA-20002: Nu exista conditii medicale inregistrate pentru pacientul cu ID-ul 21
ORA-06512: at "SYSTEM.RETRIEVE_PATIENT_INFO", line 30
ORA-06512: at line 2
```

BEGIN

dbms_output.put_line(retrieve_patient_info(1));

END;

/

```
James Doe Blue Cross Blue Shield Gripa tip A


PL/SQL procedure successfully completed.
```

- ***Trigger de tip LMD la nivel de comandă (inclusiv declanșare)***

Se restricționează inserarea în tabelul PROGRAMARI în zilele cuprinse între 25 și 31 ale lunii.

```
CREATE OR REPLACE TRIGGER restrict_insert_programari
   BEFORE INSERT ON PROGRAMARI
DECLARE
   ziua_curenta NUMBER;
BEGIN

   ziua_curenta := EXTRACT(DAY FROM sysdate);

   IF ziua_curenta BETWEEN 25 AND 31 THEN
      raise_application_error(-20100,'Nu se pot face inserari in tabela intre zilele de 25-31');
   END IF;
END;

INSERT INTO PROGRAMARI (id_programare, data_programare, id_pacient, id_doctor)
VALUES(22,  to_date('2023-03-01', 'yyyy-mm-dd'), 1, 1);
```

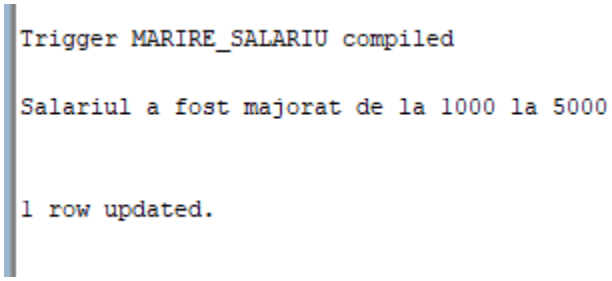
```
INSERT INTO PROGRAMARI (id_programare, data_programare, id_pacient, id_doctor) VALUES(22,  to_date('2023-03-01', 'yyyy-mm-dd'), 1, 1)
Error report -
ORA-20100: Nu se pot face inserari in tabela intre zilele de 25-31
```

- ***Trigger de tip LMD la nivel de linie (inclusiv declanșare)***

Prin intermediul trigger-ului ne asigurăm că salariul asistentei poate fi doar mărit, nu și micșorat. Dacă salariul este micșorat, atunci este afișată o eroare, dacă este mărit, atunci este afișat un mesaj cu salariul vechi și salariul actualizat, nou.

```
CREATE OR REPLACE TRIGGER marire_salariu
BEFORE UPDATE OF salariu ON ASISTENTA
FOR EACH ROW
```

```
BEGIN
  IF :NEW.salariu < :OLD.salariu THEN
    RAISE_APPLICATION_ERROR (-20222, 'salariul asistentei nu poate fi micsorat');
  END IF;
  DBMS_OUTPUT.PUT_LINE('Salariul a fost majorat de la ' || :OLD.salariu || ' la ' ||
:NEW.salariu);
END;

UPDATE ASISTENTA SET salariu = 5000 WHERE id_asistenta = 1;
```



- ***Trigger de tip LDD (inclusiv declanșare)***

La crearea, ștergerea sau modificarea unui tabel din baza de date, va fi afișat numele bazei de date, userul, evenimentul care s-a produs și data curentă.

```
CREATE OR REPLACE TRIGGER trigLDD
 AFTER CREATE OR DROP OR ALTER ON DATABASE
BEGIN
   DBMS_OUTPUT.PUT_LINE('Baza de date: '||SYS.DATABASE_NAME);
   DBMS_OUTPUT.PUT_LINE('User: '||SYS.LOGIN_USER);
   DBMS_OUTPUT.PUT_LINE('Eveniment: '||SYS.SYSEVENT);
   DBMS_OUTPUT.PUT_LINE('Data: '||SYSTIMESTAMP);
END;
/
```

```
CREATE TABLE ttest
(
id_test NUMBER(2) PRIMARY KEY,
nume VARCHAR(20)
);
```

```
Trigger TRIGLDD compiled

Baza de date: XE
User: SYSTEM
Eveniment: CREATE
Tip obiect: INDEX
Nume obiect: SYS_C009632
Data: 26-JAN-23 05.05.30.934000000 PM +02:00
Baza de date: XE
User: SYSTEM
Eveniment: CREATE
Data: 26-JAN-23 05.05.30.934000000 PM +02:00
Baza de date: XE
User: SYSTEM
Eveniment: CREATE
Tip obiect: TABLE
Nume obiect: TTEST
Data: 26-JAN-23 05.05.30.935000000 PM +02:00
Baza de date: XE
User: SYSTEM
Eveniment: CREATE
Data: 26-JAN-23 05.05.30.935000000 PM +02:00


Table TTEST created.
```

- ***Pachet care să conțină toate obiectele definite în cadrul punctului 9 (inclusiv apelarea lor)***

```
CREATE OR REPLACE PACKAGE package_exercise IS
PROCEDURE proc2;
PROCEDURE proc1;
FUNCTION pacienti_info (p_id_pacient NUMBER)
```

```
RETURN VARCHAR2;
END package_exercise;
/
SHOW ERRORS
```

```
Package PACKAGE_EXERCISE compiled

No errors.
```

```
CREATE OR REPLACE PACKAGE BODY package_exercise IS
PROCEDURE proc2 IS
 TYPE tab_imb_id_asig IS TABLE OF ASIGURARE.id_asigurare%TYPE;
   t_id_asig tab_imb_id_asig := tab_imb_id_asig();

   TYPE tab_imb_det_asiginfo IS TABLE OF ASIGURARE.asigurare_info%TYPE;
   t_det_asiginfo tab_imb_det_asiginfo := tab_imb_det_asiginfo();

   TYPE pacienti_inreg IS RECORD
      (id_pacient PACIENTI.id_pacient%TYPE,
      nume_pacient PACIENTI.nume_pacient%TYPE,
      id_asigurare ASIGURARE.id_asigurare%TYPE);

   TYPE tab_pacienti IS TABLE OF pacienti_inreg INDEX BY BINARY_INTEGER;
   t_pac tab_pacienti;

   i INTEGER;
   cont INTEGER;
   j INTEGER;
   nr_programari INTEGER;
```

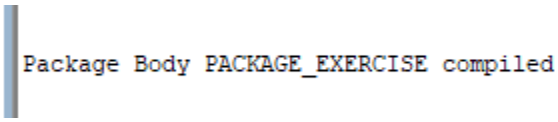
```
BEGIN
    SELECT id_asigurare
    BULK COLLECT INTO t_id_asig
    FROM ASIGURARE
    ORDER BY id_asigurare;


    SELECT asigurare_info
    BULK COLLECT INTO t_det_asiginfo
    FROM ASIGURARE
    ORDER BY id_asigurare;

    i := t_id_asig.FIRST;
    j := t_det_asiginfo.FIRST;


    WHILE i <= t_id_asig.LAST LOOP
        DBMS_OUTPUT.PUT_LINE('Asigurarea cu id-ul '||t_id_asig(i)||' ('||t_det_asiginfo(j)||')');



        SELECT id_pacient, nume_pacient, ASIGURARE.id_asigurare
        BULK COLLECT INTO t_pac
        FROM PACIENTI
        JOIN ASIGURARE ON PACIENTI.id_asigurare = ASIGURARE.id_asigurare
        WHERE ASIGURARE.id_asigurare = t_id_asig(i);

            IF t_pac.COUNT = 0 THEN
```

```
    DBMS_OUTPUT.PUT_LINE('    '||'Niciun pacient nu are acest tip de asigurare!');
    DBMS_OUTPUT.NEW_LINE;
  ELSE
    IF t_pac.COUNT >= 1 THEN
      DBMS_OUTPUT.PUT_LINE('    '||'Pacientii care au acest tip de asigurare:');
      DBMS_OUTPUT.NEW_LINE;
    ELSE
      DBMS_OUTPUT.PUT_LINE('    '||'Pacientul care are acest tip de asigurare:');
      DBMS_OUTPUT.NEW_LINE;
    END IF;

    cont := 0;
    FOR k IN t_pac.FIRST..t_pac.LAST LOOP
      cont := cont+1;
      DBMS_OUTPUT.PUT('        '||cont||'. '||t_pac(k).nume_pacient);

      SELECT COUNT(*)
      INTO nr_programari
      FROM  PROGRAMARI
      WHERE id_pacient = t_pac(k).id_pacient;

      DBMS_OUTPUT.PUT_LINE(' | numar de programari: '||nr_programari);
    END LOOP;
  END IF;

  i:=i+1;
  j:=j+1;
  DBMS_OUTPUT.NEW_LINE;
```

```
      DBMS_OUTPUT.NEW_LINE;
   END LOOP;

END proc2;

PROCEDURE proc1 AS
CURSOR c1 IS
SELECT COUNT(id_doctor) FROM DOCTORI WHERE adresa = 'Bucuresti';

cursor c2(p_specialitate varchar2) IS
SELECT nume_doctor, salariu*12 anual_sal
FROM DOCTORI
WHERE specialitate = p_specialitate;

v1 NUMBER;
v2 c2%ROWTYPE;

BEGIN
OPEN c1;
LOOP
FETCH c1 INTO v1;
EXIT WHEN c1%NOTFOUND;
DBMS_OUTPUT.PUT_LINE('Exista: '|| v1 ||' doctori in Bucuresti');
END LOOP;
CLOSE c1;

OPEN c2('Dermatologie');
FETCH c2 INTO v2;
```

```
WHILE (c2%FOUND) LOOP
DBMS_OUTPUT.PUT_LINE ('Doctorul: ' || v2.nume_doctor ||
'are salariul anual : ' || v2.anual_sal);
FETCH c2 INTO v2;
END LOOP;
CLOSE c2;
END;

FUNCTION pacienti_info (p_id_pacient NUMBER)
RETURN VARCHAR2
AS
v_nume_pacient VARCHAR2(50);
v_nume_asigurare VARCHAR2(50);
v_nume_conditie VARCHAR2(50);
v_eroare VARCHAR2(255);
BEGIN
BEGIN
SELECT nume_pacient, asigurare_info
INTO v_nume_pacient, v_nume_asigurare
FROM PACIENTI p
JOIN ASIGURARE a ON p.id_asigurare = a.id_asigurare
WHERE p.id_pacient = p_id_pacient;
EXCEPTION
WHEN NO_DATA_FOUND THEN
v_eroare := 'Pacientul cu ID-ul ' || p_id_pacient || ' nu a fost gasit in baza de date.';
RAISE_APPLICATION_ERROR(-20001, v_eroare);
END;
```

```
BEGIN
SELECT nume_conditie
INTO v_nume_conditie
FROM CONDITIE_MEDICALA c
JOIN CONDITIE_PACIENTI cp ON c.id_conditie = cp.id_conditie
WHERE cp.id_pacient = p_id_pacient;
EXCEPTION
WHEN NO_DATA_FOUND THEN
v_eroare := 'Nu exista conditii medicale inregistrate pentru pacientul cu ID-ul ' || p_id_pacient;
RAISE_APPLICATION_ERROR(-20002, v_eroare);
END;

RETURN v_nume_pacient || ' ' || v_nume_asigurare || ' ' || v_nume_conditie;
END;
END package_exercise;
```




```
BEGIN
proc2;
proc1;
dbms_output.put_line(retrieve_patient_info(1));
dbms_output.put_line(retrieve_patient_info(21));
dbms_output.put_line(retrieve_patient_info(100));
END;
/
```

10. În cazul obținerii unor excepții, dacă doriți să înregistrați unele avertismente sau alte informații, mesajele corespunzătoare vor fi inserate în tabelul MESAJE (id-ul mesajului se va insera automat folosind o secvență)

```
CREATE SEQUENCE increment_id
INCREMENT BY 1
START WITH 1
NOMAXVALUE
NOCYCLE
CACHE 20;
```

```
Sequence INCREMENT_ID created.
```

```
SET VERIFY OFF
ACCEPT adresa PROMPT 'Introduceti localitatea:'
DECLARE
v_nume DOCTORI.nume_doctor%TYPE;
v_specialitate DOCTORI.specialitate%TYPE;
v_adresa DOCTORI.adresa%TYPE:='&adresa';
BEGIN
SELECT nume_doctor, specialitate
INTO v_nume, v_specialitate
FROM DOCTORI
WHERE adresa = v_adresa;
INSERT INTO MESAJE
VALUES (increment_id.NEXTVAL, v_nume ||' - '|| v_specialitate, 'I', USER, SYSDATE);
EXCEPTION
WHEN TOO_MANY_ROWS THEN
```

```
INSERT INTO MESAJE

VALUES (increment_id.NEXTVAL, 'Exista mai multi doctori din aceasta localitate', 'W',
USER, SYSDATE);

WHEN NO_DATA_FOUND THEN

INSERT INTO MESAJE

VALUES (increment_id.NEXTVAL, 'Nu exista doctori din aceasta localitate', 'E', USER,
SYSDATE);

END;

/

SET VERIFY ON
```

La prima apelare s-a introdus ca localitate „Craiova”, ceea ce a condus la inserarea primului mesaj. La a doua apelare s-a introdus ca localitate „Bucuresti”, ceea ce a condus la inserarea celui de-al doilea mesaj.

La a treia apelare s-a introdus ca localitate „Brasov”, ceea ce a condus la inserarea celui de-al treilea mesaj.

4	4	Matthew Pop - Dermatologie	I	SYSTEM	26-JAN-23
5	5	Exista mai multi doctori din aceasta localitate	W	SYSTEM	26-JAN-23
6	6	Nu exista doctori din aceasta localitate	E	SYSTEM	26-JAN-23