Gestiunea unui spital

Simion Maria Liana Grupa 1056, Seria E

# Cuprins:

Descrierea bazei de date…………………………………………………3

Schema logica………………………………………………………………….4

Capitolul 1: Introducere în PL/SQL …………………………………..5

Capitolul 2: Tipuri de date compuse…………………………………12

Capitolul 3: Structuri fundamentale de programare…………13

Capitolul 4: Cursorul în PL/SQL…………………………………………18

Capitolul 5: Excepţii………………………………………………………….32

Capitolul 6: Proceduri………………………………………………………38

Capitolul 7: Pachete de subprograme ………………………………44

Capitolul 8: Triggeri………………………………………………………….49

Capitolul 9: APEX………………………………………………………………53

**ANEXE:……………………………………………………………………………60**

Construirea bazei de date

Adaugarea înregistrărilor

Interogări

# Descrierea temei:

Proiectul meu are ca temă gestiunea bazei de date a unui spital de stat. Baza de date stochează informații cu privire la doctorii, pacienții și secțiile spitalului. În momentul internării sunt înregistrate datele personale ale pacientului, iar într-o fișă separată se ține evidența diagnosticelor și a tratamentelor. De asemenea, baza de date trebuie să asigure gestionarea corespunzătoare a datelor referitoare la medicamentele stocate în depozitele spitalului.

Având în vedere ca spitalul este împărțit pe mai multe secții, fiecare având la rândul său mai mulți doctori, trebuie reținute informații atât despre medici cât si despre secțiile de care aparțin aceștia. De asemenea, este nevoie să se stocheze informații cu privire la pacienții spitalului, despre fișele medicale ale acestora și trebuie ținută o evidență a investigațiilor realizate pe perioada internării.

Baza de date conține șapte entități. În structură am întâlnit o relație de tip *many to many* intre REȚETE și MEDICAMENTE, dar am rezolvat-o prin entitatea TRATAMENTE. În final au rămas numai relații de tip *one to many*.

Prin modelarea entităților în limbajul SQL acestea au devenit tabele cu anumite restricții de tipul *primary key*(în cazul ID-urilor), *foreign key* sau *NOT NULL.*

# Schema bazei de date:

# 

# Capitolul 1:

Introducere în PL/SQL – Procedural Language extension to SQL

EXERCITIUL 1:

--sa se afiseze numele si prenumele medicului cu codul 9876

set SERVEROUTPUT ON

DECLARE

v\_nume doctori.nume\_d%TYPE;

v\_prenume doctori.prenume\_d%TYPE;

BEGIN

SELECT nume\_d, prenume\_d

INTO v\_nume, v\_prenume

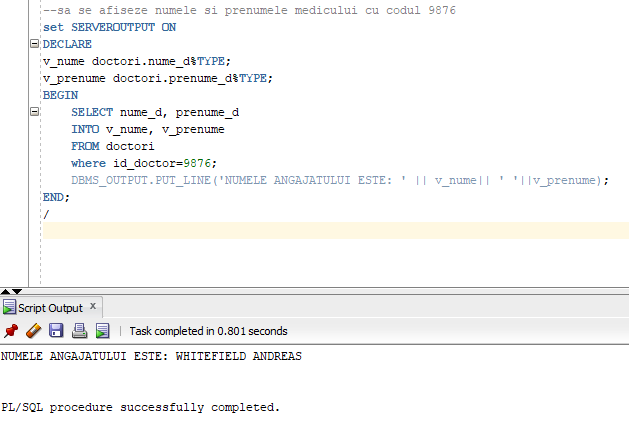
FROM doctori

where id\_doctor=9876;

DBMS\_OUTPUT.PUT\_LINE('NUMELE ANGAJATULUI ESTE: ' || v\_nume|| ' '||v\_prenume);

END;

/



EXERCITIUL 2:

--sa se afiseze numarul de pacienti ai medicului al carui cod este introdus de utilizator prin

--intermediul variabilei de substitutie &id\_medic

DECLARE

v\_nr\_pacienti number(2);

BEGIN

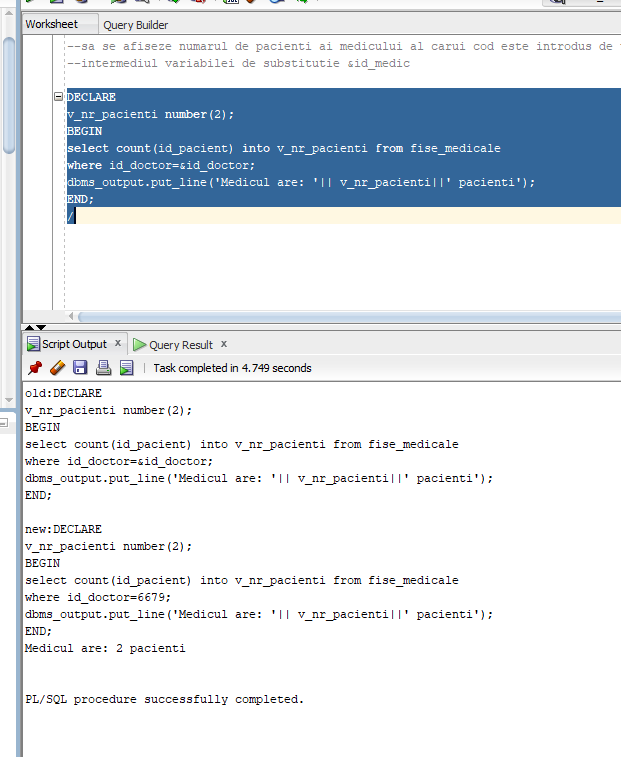
select count(id\_pacient) into v\_nr\_pacienti from fise\_medicale

where id\_doctor=&id\_doctor;

dbms\_output.put\_line('Medicul are: '|| v\_nr\_pacienti||' pacienti');

END;

/



EXERCITIUL 3:

--sa se creeze tabela med

SET SERVEROUTPUT ON

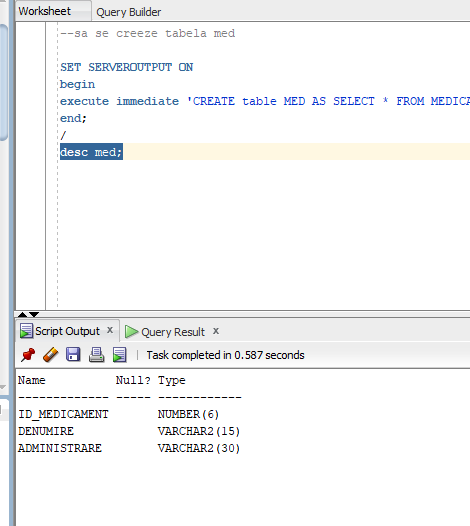
begin

execute immediate 'CREATE table MED AS SELECT \* FROM MEDICAMENTE where 1=2';

end;

/

desc med;



EXERCITIUL 4:

--sa se adauge inregistrari din tabela medicamente in tabela med

declare

v\_idmed medicamente.id\_medicament%type;

v\_denmed medicamente.denumire%type;

v\_admed medicamente.administrare%type;

begin

SELECT id\_medicament, denumire, administrare INTO v\_idmed, v\_denmed, v\_admed

FROM medicamente where id\_medicament=12000;

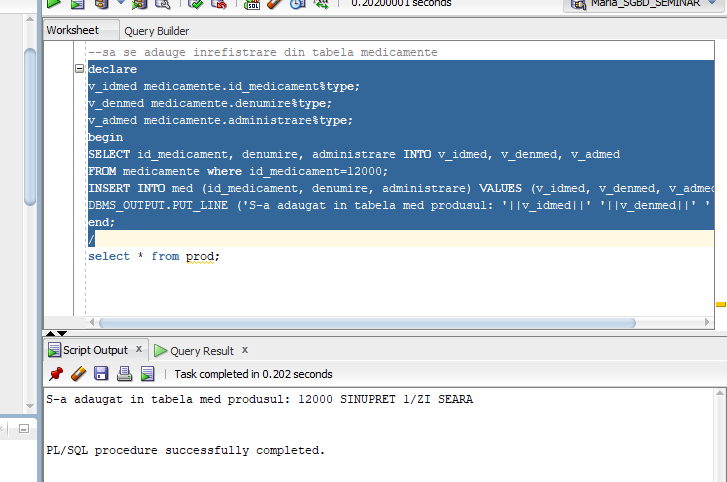
INSERT INTO med (id\_medicament, denumire, administrare) VALUES (v\_idmed, v\_denmed, v\_admed);

DBMS\_OUTPUT.PUT\_LINE ('S-a adaugat in tabela med produsul: '||v\_idmed||' '||v\_denmed||' '||v\_admed);

end;

/

select \* from med;



EXERCITIUL 5:

--sa se adauge o noua coloana stoc in tabela medicamente

DECLARE

V\_SIR VARCHAR2(200);

BEGIN

V\_SIR:='ALTER TABLE MEDICAMENTE ADD (STOC NUMBER (7))';

DBMS\_OUTPUT.PUT\_LINE (V\_SIR);

EXECUTE IMMEDIATE V\_SIR;

END;

/

select \* from MEDICAMENTE;

****

EXERCITIUL 6:

--sa se adauge o noua inregistrare in tabela pacienti

BEGIN

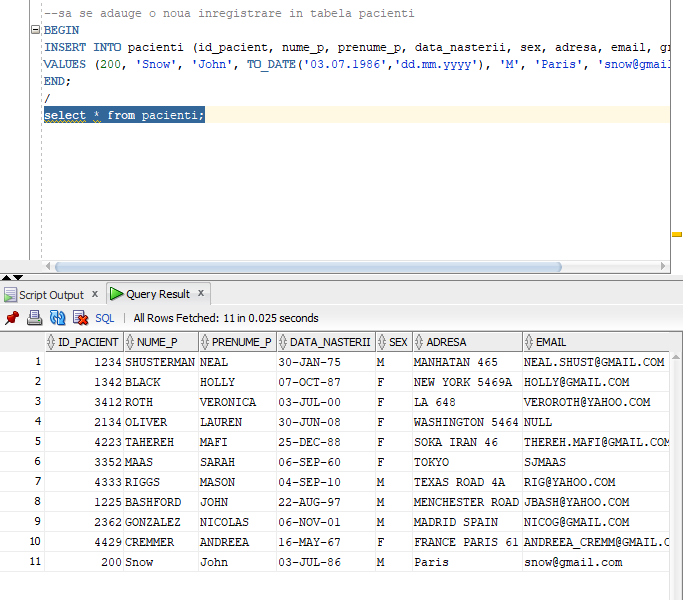
INSERT INTO pacienti (id\_pacient, nume\_p, prenume\_p, data\_nasterii, sex, adresa, email, grupa\_sange)

VALUES (200, 'Snow', 'John', TO\_DATE('03.07.1986','dd.mm.yyyy'), 'M', 'Paris', 'snow@gmail.com', '0 NEG');

END;

/

select \* from pacienti;

****

EXERCITIUL 7:

-- sa se adauge o noua inregistrare in tabela medicamente prin introducerea valorilor cu ajutorul variabileleor de substitutie

BEGIN

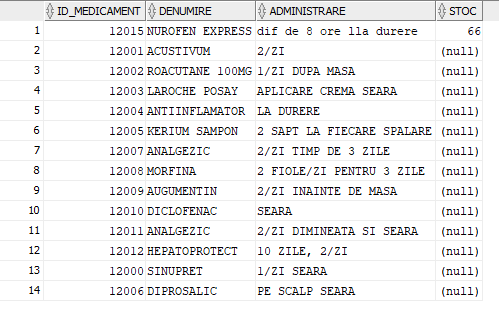
INSERT INTO medicamente (id\_medicament, denumire,administrare , stoc)

VALUES (&id, '&denumire', '&administrare', &stoc);

END;

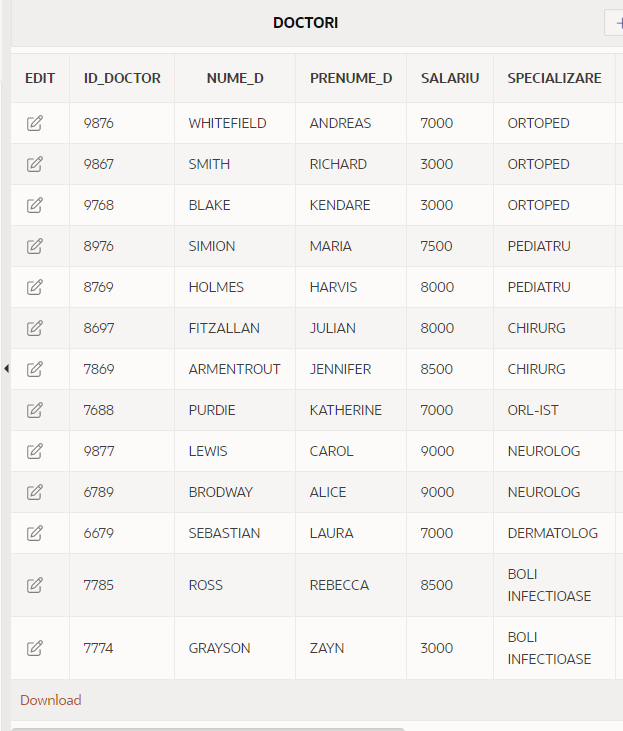
/

select \* from medicamente order by stoc;

****

EXERCITIUL 8:

--sa se mareasca cu un procent salariul angajatilor care au in prezent salariul mai mic decat o anumita valoare(4000)

****

DECLARE

v\_procent number:=0.1;

v\_prag angajati.salariul%type:=4000;

BEGIN

UPDATE doctori

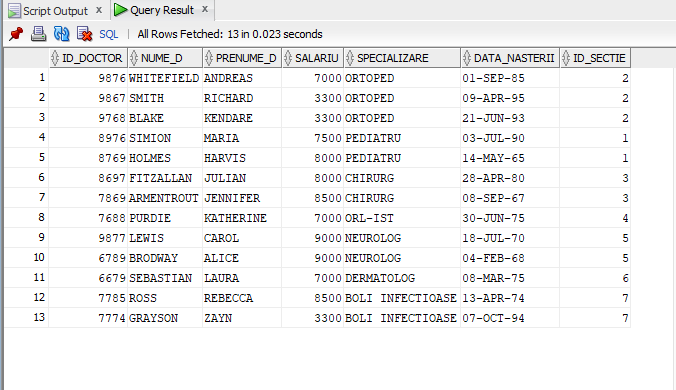
SET salariu=salariu\*(1+v\_procent)

WHERE salariu<v\_prag;

END;

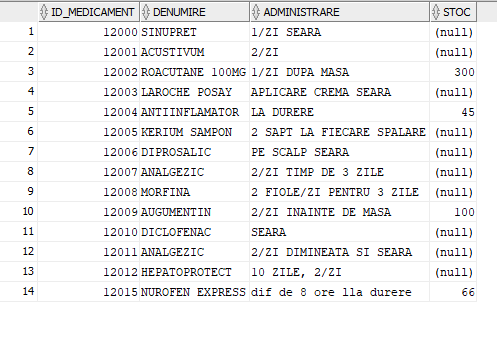
/

select \* from doctori;

****

EXERCITIUL 9:

--sa se realizeze o aprovizionare in cadrul depozitului prin care se maresc stocurile tuturor produselor cu 100 bucati

****

BEGIN

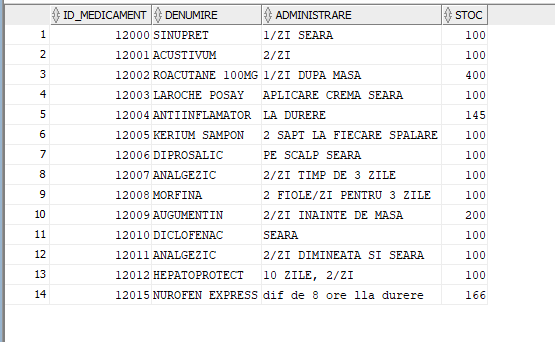
UPDATE medicamente

SET stoc=nvl(stoc,0)+100;

END;

/

select \* from medicamente;

****

EXERCITIUL 10:

-- sa se scada stocul medicamentului Hepatoprotect cu 3, valoarea fiind introdusa de la tastatura

BEGIN

UPDATE medicamente

SET stoc=stoc-&nr\_buc\_vandute

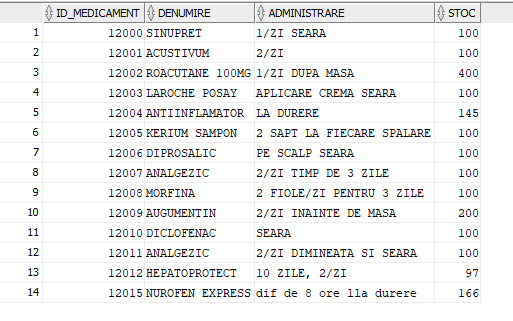
WHERE lower(denumire) like 'hepatoprotect';

COMMIT;

END;

/

select \* from medicamente;

****

EXERCITIUL 11:

--sa se stearga pacientul John Snow din tabela pacienti

DECLARE

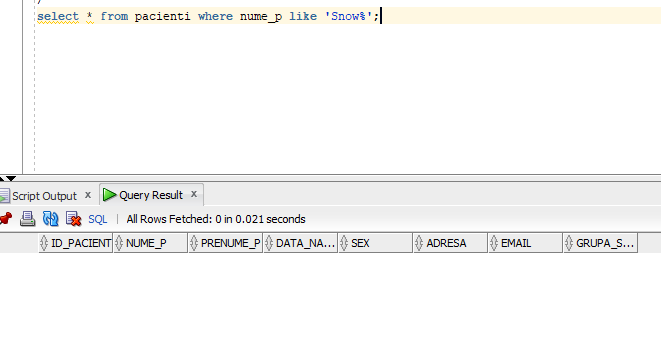
BEGIN

DELETE FROM pacienti WHERE initcap(nume\_p) like 'Snow%';

END;

/

select \* from pacienti where nume\_p like 'Snow%';



# Capitolul 2:

Tipuri de date compuse

EXERCITIUL 1:

--Utilizând un tip de dată înregistrare de același tip cu tabela doctori să se afişeze salariul medicului cu id-ul 8697.

DECLARE

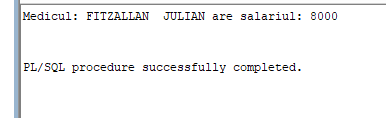
vrec\_doc doctori%rowtype;

BEGIN

SELECT \* into vrec\_doc from doctori where id\_doctor=8697;

dbms\_output.put\_line('Medicul: '|| vrec\_doc.nume\_d||' '|| vrec\_doc.prenume\_d||' are salariul: '||vrec\_doc.salariu);

END;

/

EXERCITIUL 2:

--Utilizând un tip de dată înregistrare de același tip cu un rând din tabela SECTII să se afișeze denumirea fiecărei sectii.

DECLARE

vrec\_sec sectii%rowtype;

i number:=1;

BEGIN

loop

SELECT \* into vrec\_sec from sectii where id\_sectie=i;

dbms\_output.put\_line('Sectia: '|| vrec\_sec.id\_sectie|| ' are denumirea de: '||vrec\_sec.denumire);

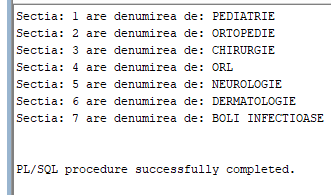
exit when i>=7;

i:=i+1;

end loop;

END;

/



# Capitolul 3:

Structuri fundamentale de programare

EXERCITIUL 1:

--In functie de stocul medicamentului avand codul citit de la tastatura se va afisa modificat pe ecran noua valoare a stocului

SET SERVEROUTPUT ON

DECLARE

v\_stoc medicamente.stoc%type;

BEGIN

SELECT stoc into v\_stoc from medicamente where id\_medicament= &m;

DBMS\_OUTPUT.PUT\_LINE('Stocul initial este: ' || v\_stoc);

IF v\_stoc<130 THEN

v\_stoc:=2\*v\_stoc;

ELSEIF v\_stoc between 130 and 300 THEN

v\_stoc:=1.5\*v\_stoc;

ELSE

v\_stoc:=1.25\*v\_stoc;

END IF;

DBMS\_OUTPUT.PUT\_LINE('Pretul final este: '|| v\_stoc);

end;

/

EXERCITIUL 2:

-- Se afişează în ordine medicii cu codurile în intervalul 93-96 atât timp cât salariul acestora este mai mic decât media:

set serveroutput on

DECLARE

v\_sal doctori.salariu%type;

v\_salMediu v\_sal%type;

i number(6):=93;

BEGIN

SELECT avg(salariu) into v\_salMediu from doctori;

dbms\_output.put\_line('Salariul mediu este: '||v\_salMediu);

loop

select salariu into v\_sal from doctori where id\_doctor=i;

dbms\_output.put\_line('Medicul cu codul '||i||' are salariul: '||v\_sal);

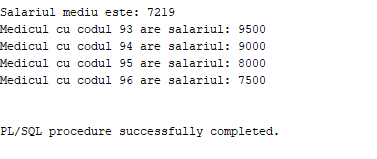
i:=i+1;

exit when v\_sal<v\_salMediu or i>96;

end loop;

end;

/



EXERCITIUL 3:

--aceeasi cerinta, folosind **while…loop**

set serveroutput on

DECLARE

v\_sal doctori.salariu%type;

v\_salMediu v\_sal%type;

i number(6):=93;

BEGIN

SELECT avg(salariu) into v\_salMediu from doctori;

dbms\_output.put\_line('Salariul mediu este: '||v\_salMediu);

while i<=96 loop

select salariu into v\_sal from doctori where id\_doctor=i;

dbms\_output.put\_line('Salariatul cu codul '||i||' are salariul: '||v\_sal);

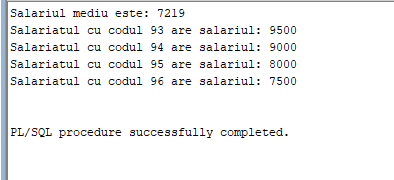
i:=i+1;

exit when v\_sal<v\_salMediu;

end loop;

end;

/



EXERCITIUL 4:

--aceeasi cerinta utilizand **for …loop**

set serveroutput on

DECLARE

v\_sal doctori.salariu%type;

v\_salMediu v\_sal%type;

BEGIN

SELECT avg(salariu) into v\_salMediu from doctori;

dbms\_output.put\_line('Salariul mediu este: '||v\_salMediu);

for i in 93..96 loop

select salariu into v\_sal from doctori where id\_doctor=i;

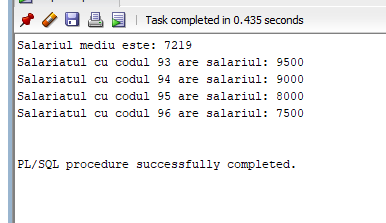
dbms\_output.put\_line('Salariatul cu codul '||i||' are salariul: '||v\_sal);

exit when v\_sal<v\_salMediu;

end loop;

end;

/



EXERCITIUL 5:

--sa se afiseze numarul de pacienti al fiecarui medic al carui id este situat in intervalul 9687 9876 dar sa se intrerupa afisarea in cazul in care se gaseste primul medic din acest interval care nu are niciun pacient

DECLARE

v\_nr number;

v\_nume doctori.nume\_d%type;

v\_id doctori.id\_doctor%type;

BEGIN

for v\_id in 8697..9876 loop

v\_nr:=0;

SELECT count(f.id\_pacient) into v\_nr from fise\_medicale f,doctori d

where f.id\_doctor=d.id\_doctor and d.id\_doctor=v\_id;

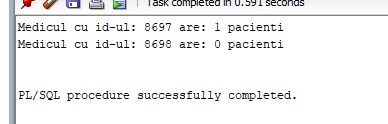
dbms\_output.put\_line('Medicul cu id-ul: '||v\_id||' are: '||v\_nr||' pacienti');

exit when v\_nr=0;

end loop;

end;

/



EXERCITIUL 6:

--Utilizarea unui tablou indexat

DECLARE

type num\_table is table of medicamente.denumire%type index by pls\_integer;

v\_tab num\_table;

i number(7):=12000;

BEGIN

loop

SELECT denumire into v\_tab(i) from medicamente where id\_medicament=i;

i:=i+1;

exit when i>12009;

end loop;

for i in v\_tab.first..v\_tab.last loop

IF v\_tab.EXISTS(i) then

dbms\_output.put\_line('Nume medicament: '|| v\_tab(i));

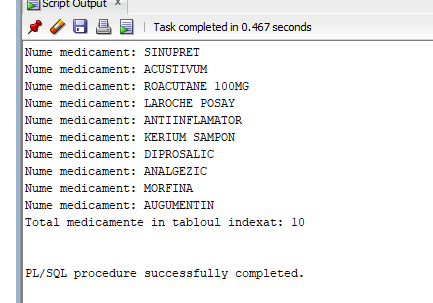
end if;

end loop;

dbms\_output.put\_line('Total medicamente in tabloul indexat: '|| v\_tab.count);

END;

/



EXERCITIUL 7:

--Utilizarea unui tablou indexat de același tip cu un rând din tabela DOCTORI - %ROWTYPE

DECLARE

type doc\_table is table of doctori%rowtype index by pls\_integer;

v\_tab doc\_table;

BEGIN

for i in 90..97 loop

SELECT \* into v\_tab(i) from doctori where id\_doctor=i;

end loop;

for i in v\_tab.first..v\_tab.last loop

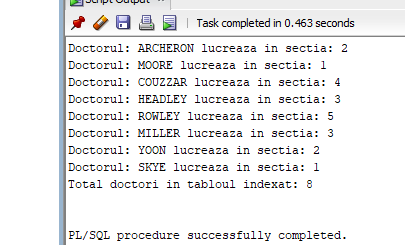
dbms\_output.put\_line('Doctorul: '|| v\_tab(i).nume\_d|| ' lucreaza in sectia: '||v\_tab(i).id\_sectie);

end loop;

dbms\_output.put\_line('Total doctori in tabloul indexat: '|| v\_tab.count);

END;

/



# Capitolul 4:

Cursorul in PL/SQL

EXERCITIUL 1:

--se sterg medicii din sectia pediatrie care nu au pacienti

SET SERVEROUTPUT ON

BEGIN

DELETE FROM doctori d

WHERE denumire= 'PEDIATRIE' and not exists (select 1 from fise\_medicale f where d.id\_doctor=f.id\_doctor);

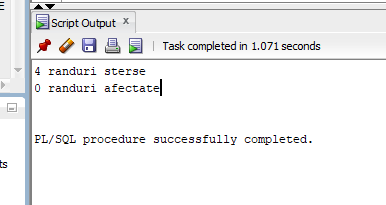
DBMS\_OUTPUT.PUT\_LINE (SQL%ROWCOUNT || ' randuri sterse');

ROLLBACK;

DBMS\_OUTPUT.PUT\_LINE (SQL%ROWCOUNT || ' randuri afectate');

END;

/



EXERCITIUL 2:

--Se încearcă adăugarea unei sectii și apoi modificarea denumirii medicamentului având codul 12012. În cazul în care acest produs nu există (comanda update nu realizează nici o modificare) va fi afişat un mesaj corespunzător.

BEGIN

INSERT INTO sectii VALUES(8,'REUMATOLOGIE');

UPDATE medicamente

SET denumire='HEPATO PLUS'

WHERE id\_medicament=12012;

IF SQL%NOTFOUND THEN

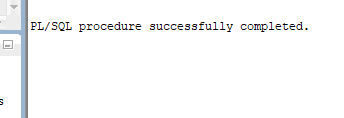
DBMS\_OUTPUT.PUT\_LINE('Nu exista produsul cu acest cod');

END IF;

ROLLBACK;

END;

/



EXERCITIUL 3:

--Se încearcă adăugarea unei sectii și apoi modificarea denumirii medicamentului având codul 12012. În cazul în care acest produs nu există

--(comanda update nu realizează nici o modificare) va fi afişat un mesaj corespunzător.

BEGIN

INSERT INTO sectii VALUES(8,'REUMATOLOGIE');

UPDATE medicamente

SET denumire='HEPATO PLUS'

WHERE id\_medicament=543;

IF SQL%NOTFOUND THEN

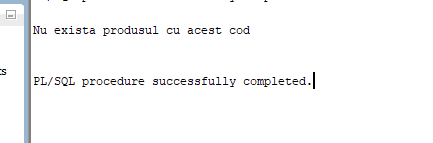
DBMS\_OUTPUT.PUT\_LINE('Nu exista produsul cu acest cod');

END IF;

ROLLBACK;

END;

/



EXERCITIUL 4:

--Se șterge din tabela sectii, sectia a cărei ID este introdus de utilizator prin intermediul variabilei de substituție g\_rid. Mesajul este afișat folosind variabila de mediu nr\_sters.

set server output on

ACCEPT g\_rid PROMPT 'Introduceti id-ul sectiei:'

VARIABLE nr\_sters varchar2(100)

DECLARE

BEGIN

DELETE FROM sectii WHERE id\_sectie=&g\_rid;

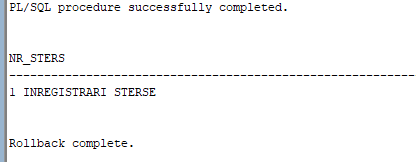
:nr\_sters:=TO\_CHAR(SQL%ROWCOUNT)||' INREGISTRARI STERSE';

END;

/

PRINT nr\_sters

ROLLBACK;



EXERCITIUL 5:

--Să se afişeze lista cu numele şi salariul medicilor sectiei 2 folosind un cursor explicit și trei variabile scalare:

set serveroutput on

DECLARE

cursor doc\_cursor is select id\_doctor, nume\_d, salariu from doctori where id\_sectie=2;

doc\_id doctori.id\_doctor%type;

doc\_nume doctori.nume\_d%type;

doc\_sal doctori.salariu%type;

BEGIN

dbms\_output.put\_line('Lista cu salariariile medicilor sectiei 2');

open doc\_cursor;

loop

fetch doc\_cursor into doc\_id, doc\_nume, doc\_sal;

exit when doc\_cursor%notfound;

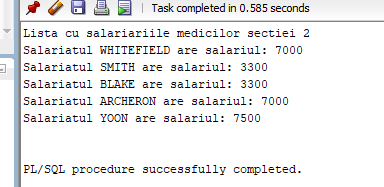
dbms\_output.put\_line('Salariatul '||doc\_nume||' are salariul: '||doc\_sal);

end loop;

close doc\_cursor;

end;

/



EXERCITIUL 6:

--Să se încarce în tabela MESAJE primii 5 pacienti (id şi nume)

CREATE TABLE mesaje

(cod varchar2(7),

nume varchar2(20)

);

DECLARE

v\_id pacienti.id\_pacient%type;

v\_nume pacienti.nume\_p%type;

CURSOR c1 IS SELECT id\_pacient, nume\_p FROM pacienti;

BEGIN

OPEN c1;

FOR i IN 1..5 LOOP

FETCH c1 INTO v\_id, v\_nume;

INSERT INTO mesaje VALUES(v\_id, v\_nume);

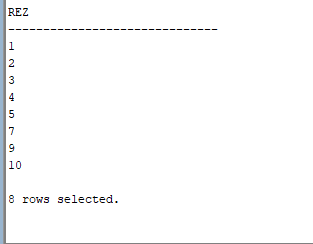
END LOOP;

CLOSE c1;

END;

/

SELECT \* FROM mesaje;



EXERCITIUL 7:

--Să se afişeze primii 3 medici care au cei mai multi pacienti. În acest caz înregistrările vor fi ordonate descrescător în funcţie de numărul pacientilor.

--Am folosit funcția CAST pentru a transforma în interogare coloana DATA din TIMESTAMP în DATE.

SET SERVEROUTPUT ON

DECLARE

CURSOR c\_doc IS

select d.id\_doctor, cast (d.data\_nasterii as date) data, count(f.id\_pacient) numar

from doctori d, fise\_medicale f

where d.id\_doctor=f.id\_doctor

group by d.id\_doctor, d.data\_nasterii

order by count(f.id\_pacient) desc;

rec\_doc c\_doc%rowtype;

BEGIN

DBMS\_OUTPUT.PUT\_LINE('Numarul de pacienti pentru fiecare medic:');

IF NOT c\_doc%ISOPEN THEN

OPEN c\_doc;

END IF;

LOOP

FETCH c\_doc INTO rec\_doc;

EXIT WHEN c\_doc%NOTFOUND OR c\_doc%ROWCOUNT>3;

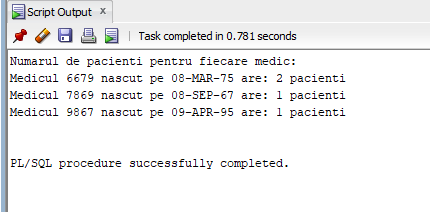
DBMS\_OUTPUT.PUT\_LINE('Medicul '||rec\_doc.id\_doctor||' nascut pe '||rec\_doc.data||' are: '||rec\_doc.numar||' pacienti');

END LOOP;

CLOSE c\_doc;

END;

/



Cursorul explicitl

EXERCITIUL 1:

--Se afişează printr-un ciclu FOR numele şi salariile medicilor de pe sectia de neurologie (id sectie=5):

SET SERVEROUTPUT ON

DECLARE

CURSOR c\_doc IS SELECT id\_doctor, nume\_d, salariu FROM doctori WHERE id\_sectie=5;

BEGIN

DBMS\_OUTPUT.PUT\_LINE('Lista cu salariariile doctorilor de pe sectia de neurologie este');

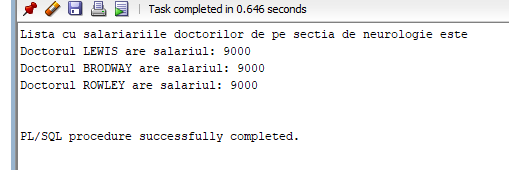
FOR r in c\_doc LOOP

DBMS\_OUTPUT.PUT\_LINE('Doctorul '|| r.nume\_d ||' are salariul: '||r.salariu);

END LOOP;

END;

/



EXERCITIUL 2:

--Să se afişeze suma aferentă salariilor din fiecare sectie:

SET SERVEROUTPUT ON

DECLARE

BEGIN

DBMS\_OUTPUT.PUT\_LINE('Total salarii pe fiecare sectie:');

FOR r IN

(SELECT s. id\_sectie sec, sum(d.salariu) sal

FROM doctori d, sectii s

WHERE d.id\_sectie=s.id\_sectie

GROUP BY s.id\_sectie)

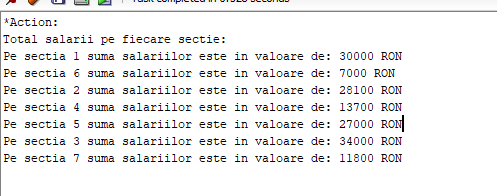
LOOP

DBMS\_OUTPUT.PUT\_LINE('Pe sectia '|| r.sec ||' suma salariilor este in valoare de: '|| r.sal ||' RON');

END LOOP;

END;

/



EXERCITIUL 3:

**Am creat o noua tabela denumita vanzari\_medicamente in care am introdus cateva inregistrari pentru a avea ceva asemanator cu tabela rand comenzi.**

Create table VANZARI\_MEDICAMENTE(

id\_vanzare number(2),

id\_medicament number(6),

pret number(3),

cantitate number(3),

constraint vanzari\_medicamente\_pk primary key(id\_vanzare),

CONSTRAINT FK\_ID\_MEDICAMENT FOREIGN KEY (ID\_MEDICAMENT) REFERENCES MEDICAMENTE(ID\_MEDICAMENT));

insert into vanzari\_medicamente(id\_vanzare, id\_medicament, pret, cantitate) values (10, 12000, 4, 2);

insert into vanzari\_medicamente(id\_vanzare, id\_medicament, pret, cantitate) values (11, 12000, 4, 3);

insert into vanzari\_medicamente(id\_vanzare, id\_medicament, pret, cantitate) values (12, 12001, 50, 1);

insert into vanzari\_medicamente(id\_vanzare, id\_medicament, pret, cantitate) values (13, 12001, 50, 2);

insert into vanzari\_medicamente(id\_vanzare, id\_medicament, pret, cantitate) values (14, 12002, 68, 3);

insert into vanzari\_medicamente(id\_vanzare, id\_medicament, pret, cantitate) values (15, 12002, 68, 1);

insert into vanzari\_medicamente(id\_vanzare, id\_medicament, pret, cantitate) values (16, 12003, 70, 1);

insert into vanzari\_medicamente(id\_vanzare, id\_medicament, pret, cantitate) values (17, 12004, 11, 6);

insert into vanzari\_medicamente(id\_vanzare, id\_medicament, pret, cantitate) values (18, 12005, 60, 1);

insert into vanzari\_medicamente(id\_vanzare, id\_medicament, pret, cantitate) values (19, 12004, 11, 4);

insert into vanzari\_medicamente(id\_vanzare, id\_medicament, pret, cantitate) values (20, 12006, 25, 3);

insert into vanzari\_medicamente(id\_vanzare, id\_medicament, pret, cantitate) values (21, 12007, 9, 7);

insert into vanzari\_medicamente(id\_vanzare, id\_medicament, pret, cantitate) values (22, 12007, 9, 13);

insert into vanzari\_medicamente(id\_vanzare, id\_medicament, pret, cantitate) values (23, 12012, 30, 6);

insert into vanzari\_medicamente(id\_vanzare, id\_medicament, pret, cantitate) values (24, 12009, 20, 7);

insert into vanzari\_medicamente(id\_vanzare, id\_medicament, pret, cantitate) values (25, 12010, 45, 1);

insert into vanzari\_medicamente(id\_vanzare, id\_medicament, pret, cantitate) values (26, 12015, 15, 10);

--Să se afişeze produsele al căror cantitate totală comandată este mai mare

--decât o valoare primită drept parametru.

SET SERVEROUTPUT ON

DECLARE

CURSOR c (p\_val NUMBER) IS

SELECT m.id\_medicament, m.denumire, sum(v.cantitate) total

FROM medicamente m, vanzari\_medicamente v

WHERE m.id\_medicament =v.id\_medicament

GROUP BY m.id\_medicament, m.denumire

HAVING sum(v.cantitate)>p\_val

ORDER BY total desc;

v\_val NUMBER(5);

r c %rowtype;

BEGIN

v\_val:=4;

DBMS\_OUTPUT.PUT\_LINE('Medicamentele a caror cantitate vândută este mai mare decat '|| v\_val);

IF NOT c%ISOPEN THEN

OPEN c (v\_val);

END IF;

LOOP

FETCH c into r;

EXIT WHEN c%NOTFOUND;

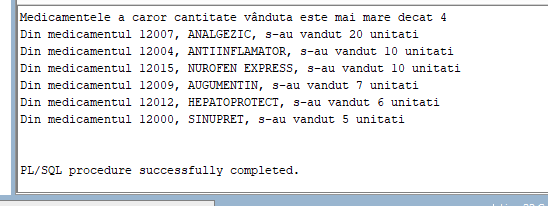
DBMS\_OUTPUT.PUT\_LINE('Din medicamentul '||r.id\_medicament||', '||r.denumire||', s-au vandut ' ||r.total||' unitati');

END LOOP;

CLOSE c;

END;

/



EXERCITIUL 4:

--Aceeaşi cerinţă rezolvata fără cursor, cu parametru.

SET SERVEROUTPUT ON

DECLARE

CURSOR c IS

SELECT m.id\_medicament, m.denumire, sum(v.cantitate) total

FROM medicamente m, vanzari\_medicamente v

WHERE m.id\_medicament =v.id\_medicament

GROUP BY m.id\_medicament, m.denumire

ORDER BY total desc;

r c%rowtype;

BEGIN

DBMS\_OUTPUT.PUT\_LINE('Medicamentele a caror cantitate vândută este mai mare decat '|| 5);

IF NOT c%ISOPEN THEN

OPEN c;

END IF;

LOOP

FETCH c into r;

IF r.total >5 THEN

DBMS\_OUTPUT.PUT\_LINE('Din medicamentul '||r.id\_medicament||', '||r.denumire||', s-au vandut ' ||r.total||' unitati');

END IF;

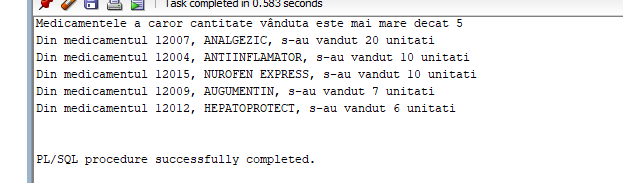
EXIT WHEN c%notfound;

END LOOP;

CLOSE c;

END;

/



EXERCITIUL 5:

--Sa se afiseze medicii de pe sectia de chirurgie (id=3) cu salariul mai mare de 4000 si pe cei de pe sectia de neurologie (id=5) cu salariul mai mare de 5000 cu ajutorul unui cursor care primeşte ca parametru id\_sectie si salariu.

DECLARE

CURSOR c(p\_id NUMBER, p\_id1 NUMBER) IS SELECT id\_sectie, nume\_d, salariu FROM doctori WHERE id\_sectie = p\_id AND salariu >= p\_id1;

BEGIN

FOR r IN c(3, 4000) LOOP

DBMS\_OUTPUT.PUT\_LINE('Pe sectia ' ||r.id\_sectie|| ' medicul '||r.nume\_d||' are salariul ' ||r.salariu);

END LOOP;

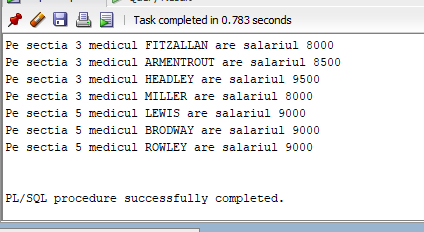
FOR r IN c(5, 5000) LOOP

DBMS\_OUTPUT.PUT\_LINE('Pe sectia '|| r.id\_sectie|| ' medicul ' ||r.nume\_d ||' are salariul ' ||r.salariu);

END LOOP;

END;

/



EXERCITIUL 6:

alter table doctori add ID\_SEF\_SECTIE number(4);

alter table sectii add ID\_SEF\_SECTIE number(4);

alter table doctori add constraint DOC\_SEF\_FK foreign key (ID\_SEF\_SECTIE) references doctori (ID\_DOCTOR);

alter table sectii add constraint SECTII\_SEF\_FK foreign key (ID\_SEF\_SECTIE) references doctori (ID\_DOCTOR);

update doctori set id\_sef\_sectie=8769 where id\_doctor in(91,97,8976);

update doctori set id\_sef\_sectie=96 where id\_doctor in(9876,9867,9768,90);

update doctori set id\_sef\_sectie=7869 where id\_doctor in(93,95,8697);

update doctori set id\_sef\_sectie=7688 where id\_doctor in(92);

update doctori set id\_sef\_sectie=6789 where id\_doctor in(94,9877);

update doctori set id\_sef\_sectie=7785 where id\_doctor in(7774);

-- Să se afişeze medicii de pe o anumita sectie împreuna cu salariile lor şi ale tuturor subordonaţilor ierarhici

DECLARE

CURSOR c (p\_id NUMBER) IS SELECT nume\_d, sum(salariu) sal FROM (SELECT CONNECT\_BY\_ROOT nume\_d nume\_d, salariu FROM doctori CONNECT BY PRIOR id\_doctor = id\_sef\_sectie START WITH id\_sectie = p\_id) GROUP BY nume\_d ORDER BY sal DESC;

BEGIN

FOR r IN c(1) LOOP

DBMS\_OUTPUT.PUT\_LINE(r.nume\_d|| ' ' ||r.sal);

END LOOP;

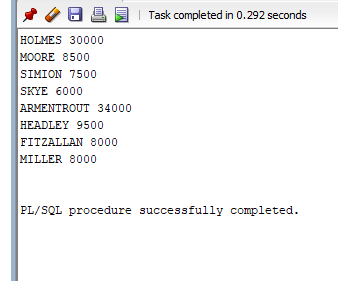
FOR r IN c(3) LOOP

DBMS\_OUTPUT.PUT\_LINE(r.nume\_d|| ' ' ||r.sal);

END LOOP;

END;

/



EXERCITIUL 8:

--Se actualizează salariul dacă acesta este mai mare de 3000 şi au fost nascuti înainte de 1980. Se afişează numărul de înregistrări modificate;

DECLARE

CURSOR c IS SELECT \* FROM doctori FOR UPDATE of salariu NOWAIT;

v\_rez NUMBER:=0;

BEGIN

FOR r IN c LOOP

IF r.salariu > 3000 AND r.data\_nasterii < TO\_DATE('01.01.1980','DD.MM.YYYY') THEN

UPDATE doctori

SET salariu = salariu \* 1.01

WHERE id\_doctor = r.id\_doctor;

v\_rez:=v\_rez+1;

END IF;

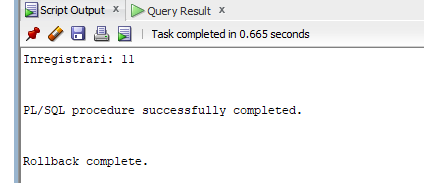
END LOOP;

DBMS\_OUTPUT.PUT\_LINE('Inregistrari: ' || v\_rez);

END;

/

ROLLBACK;



EXERCITIUL 9:

--Afişaţi toţi medicii şi pacientii acestora. Folosiţi un cursor pentru a încărca numele medicilor şi un cursor parametrizat pentru încărcarea pacientilor. Pentru medicii care nu au pacienti se va afisa mesajul corespunzător.

declare

cursor c is select id\_doctor, nume\_d from doctori;

cursor c1 (p\_id\_doctor doctori.id\_doctor%type) is select id\_pacient, cod\_fisa, diagnostic from fise\_medicale where id\_doctor=p\_id\_doctor;

v\_verificare boolean;

begin

for r in c loop

dbms\_output.put\_line('Doctorul '|| r.nume\_d);

v\_verificare:=false;

for r1 in c1 (r.id\_doctor) loop

dbms\_output.put\_line('Pacientul cu id-ul '||r1.id\_pacient|| ' fiind inscris pe fisa medicala cu codul '||r1.cod\_fisa || 'are diagnosticul: ' || r1.diagnostic);

v\_verificare:=true;

end loop;

if v\_verificare = false then

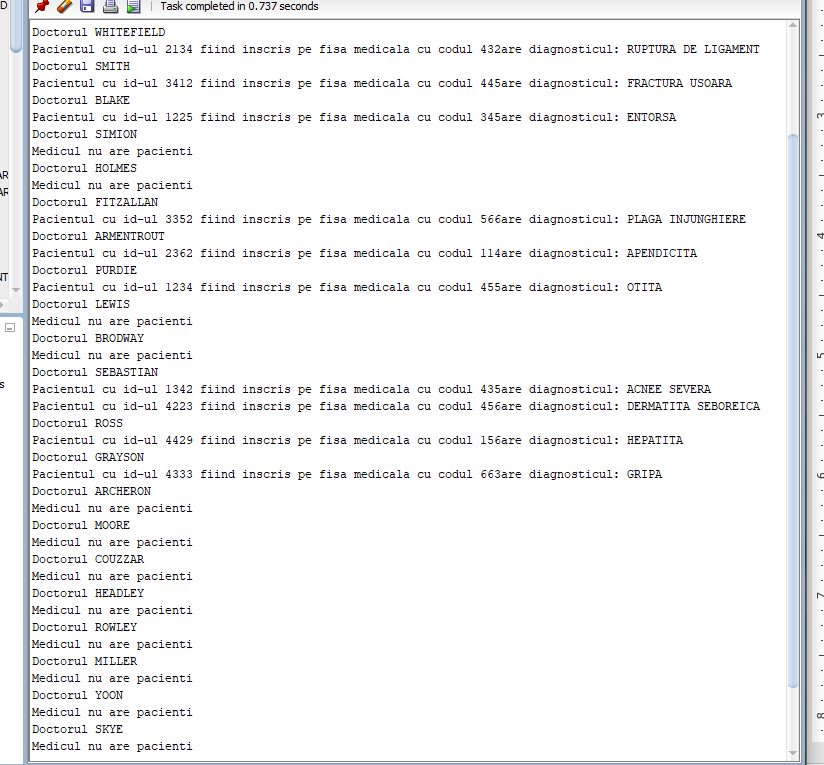
dbms\_output.put\_line('Medicul nu are pacienti');

end if;

end loop;

end;

/



EXERCITIUL 10:

--Afişaţi informaţii despre primele 3 vanzari care au cea mai mare valoare.

set SERVEROUTPUT ON

declare

cursor c is select id\_vanzare, sum(pret\*cantitate) as total from vanzari\_medicamente group by id\_vanzare;

r c%rowtype;

begin

open c;

loop

fetch c into r;

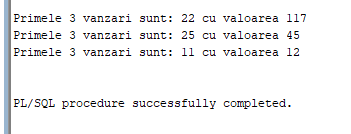
exit when c%notfound or c%rowcount>3;

dbms\_output.put\_line('Primele 3 vanzari sunt: ' || r.id\_vanzare || ' ' || r.total);

end loop;

end;

/



EXERCITIUL 11:

--Afişaţi informatii despre primii 5 cei mai in varsta doctori (filtrarea funcţie de câmpul Data\_nasterii).

declare

cursor c is select id\_doctor, nume\_d, data\_nasterii from doctori order by data\_nasterii;

r c%rowtype;

begin

open c;

loop

fetch c into r;

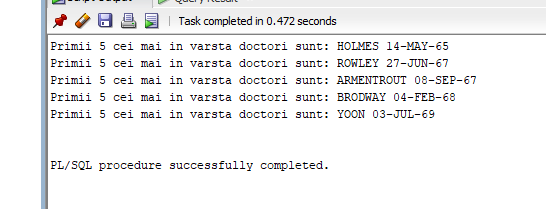
exit when c%notfound or c%rowcount>5;

dbms\_output.put\_line('Primii 5 cei mai in varsta doctori sunt: ' || r.nume\_d || ' ' || r.data\_nasterii);

end loop;

end;

/



# Capitolul 5:

*Excepţii*

EXERCITIUL 1:

--sa se afiseze medicul cu codul 20. sa se trateze eroarea aparuta in cazul in care nu exista niciun medic cu acest cod

set serveroutput on

declare v\_nume varchar2(20);

begin

select nume\_d into v\_nume

from doctori where id\_doctor=20;

dbms\_output.put\_line(v\_nume);

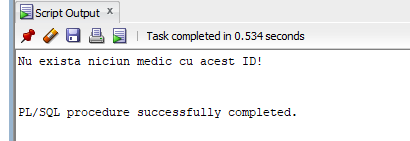
exception

when no\_data\_found then

dbms\_output.put\_line('Nu exista niciun medic cu acest ID!');

end;

/



EXERCITIUL 2:

--sa se afiseze salariul medicului cu prenumele EVELYN. sa se trateze eroarea aparuta in cazul in care exista mai multi medici cu acelasi nume.

set serveroutput on

declare

sal doctori.salariu%type;

begin

select salariu into sal from doctori where prenume\_d='EVELYN';

dbms\_output.put\_line('Evelyn are salariul '|| sal);

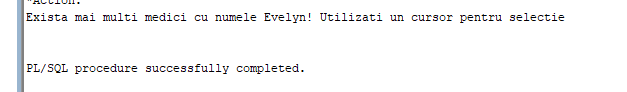
exception

when too\_many\_rows then

dbms\_output.put\_line('Exista mai multi medici cu numele Evelyn! Utilizati un cursor pentru selectie' );

end;

/



declare cursor c is select id\_doctor, nume\_d, prenume\_d, salariu from doctori where prenume\_d='EVELYN';

begin

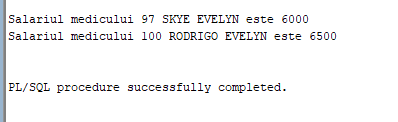
for r in c loop

dbms\_output.put\_line('Salariul medicului '||r.id\_doctor||' '||r.nume\_d||' '||r.prenume\_d||' este '||r.salariu);

end loop;

end;

/



EXERCITIUL 3:

--Să se insereze în tabela sectii o noua sectie cu ID-ul 11,

--fără a preciza denumirea acestuia. În acest caz va apare o eroarea cu codul ORA-02290

--prin care programatorul este avertizat de

--încălcarea unei restricţii de integritate. Sa se trateze aceasta exceptie:

set SERVEROUTPUT ON

declare

insert\_exception exception;

PRAGMA EXCEPTION\_INIT(insert\_except, -02290);

begin

insert into sectii (id\_sectie, denumire) values (11,null);

exception

when insert\_except then

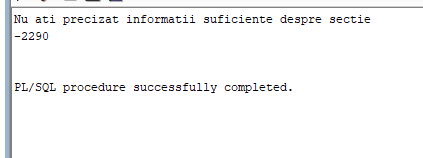
dbms\_output.put\_line('Nu ati precizat informatii suficiente pentru sectie');

DBMS\_OUTPUT.PUT\_LINE(SQLERRM);

dbms\_output.put\_line(sqlcode);

end;

/



EXERCITIUL 4:

--Să se şteargă toate înregistrările din tabela MEDICAMENTE. Acest lucru va duce

--la apariţia erorii cu codul –2292, reprezentând încălcarea restricţiei referenţiale.

--Valorile SQLCODE şi SQLERRM vor fi inserate în tabela ERORI. ATENTIE!

--Aceste variabile nu se pot utiliza direct într-o comandă SQL, drept pentru care vor fi

--încărcate mai întâi in variabile PL/SQL şi apoi utilizate în instrucţiuni SQL.

CREATE TABLE erori

(utilizator VARCHAR2(40),

data DATE,

cod\_eroare NUMBER(10),

mesaj\_eroare VARCHAR2(255)

);

DECLARE

cod NUMBER;

mesaj VARCHAR2(255);

del\_exception EXCEPTION;

PRAGMA EXCEPTION\_INIT(del\_exception, -2292);

BEGIN

DELETE FROM medicamente;

EXCEPTION

WHEN del\_exception THEN

dbms\_output.put\_line('Nu puteti sterge produsul');

dbms\_output.put\_line('Exista retete asignate lui');

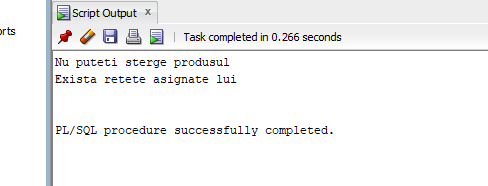
cod:=SQLCODE;

mesaj:=SQLERRM;

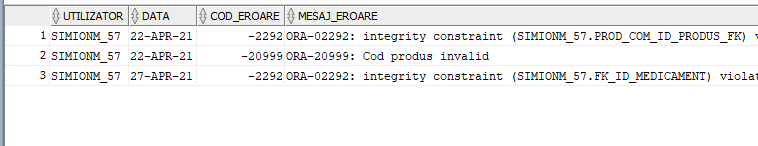
INSERT INTO erori VALUES(USER, SYSDATE, cod, mesaj);

END;

/



SELECT \* FROM erori;



EXERCITIUL 5:

--Să se modifice denumirea medicamentului cu id-ul 123. Dacă nu se produce nici o actualizare

--(valoarea atributului SQL%ROWCOUNT este 0) sau dacă apare o altă eroare (OTHERS) atunci

--să se declanşeze o excepţie prin care să fie avertizat utilizatorul:

set SERVEROUTPUT ON

DECLARE

invalid\_med EXCEPTION;

BEGIN

UPDATE medicamente

SET denumire='LP Crema'

WHERE id\_medicament=123;

IF SQL%NOTFOUND THEN

RAISE invalid\_med;

END IF;

EXCEPTION

WHEN invalid\_med THEN

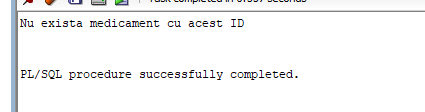
DBMS\_OUTPUT.PUT\_LINE('Nu exista medicament cu acest ID');

WHEN OTHERS THEN

DBMS\_OUTPUT.PUT\_LINE('A aparut o eroare! Nu se poate actualiza denumirea medicamentului!');

END;

/



EXERCITIUL 6:

--Să se atribuie excepţiei din exemplul anterior un cod şi un mesaj de eroare

--şi să se insereze aceste valori în tabela ERORI.

DECLARE

cod NUMBER;

mesaj VARCHAR2(255);

invalid\_med EXCEPTION;

PRAGMA EXCEPTION\_INIT(invalid\_med,-20999);

BEGIN

UPDATE medicamente

SET denumire='LP Crema'

WHERE id\_medicament=123;

IF SQL%NOTFOUND THEN

RAISE\_APPLICATION\_ERROR (-20999,'Cod medicament invalid!');

END IF;

EXCEPTION

WHEN invalid\_med THEN

DBMS\_OUTPUT.PUT\_LINE('Nu exista produsul cu acest ID');

cod:=SQLCODE;

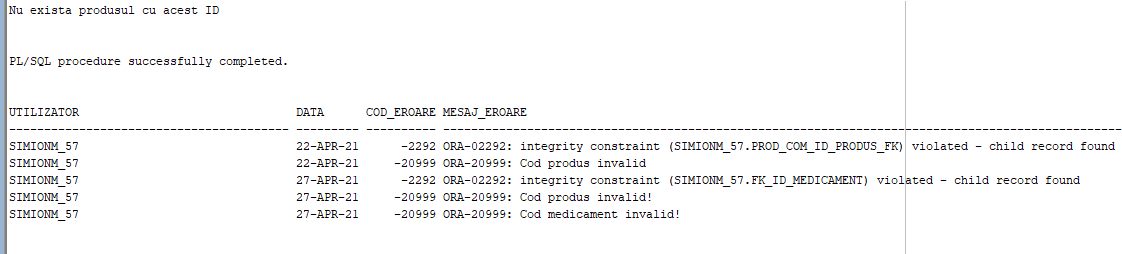
mesaj:=SQLERRM;

INSERT INTO ERORI VALUES(USER, SYSDATE, cod, mesaj);

END;

/

SELECT \* FROM ERORI;



EXERCITIUL 7:

**--1. Creaţi o tabela numita Mesaje, având un câmp unic, de tip Varchar2.**

**--2. Scrieţi un bloc PL/SQL pentru a selecta codul comenzilor încheiate în anul 2000.**

**--a. Dacă interogarea returnează mai mult de o valoare pentru numărul comenzii, trataţi excepţia cu o rutină de tratare corespunzătoare şi inseraţi în tabela MESAJE mesajul “Interogarea returneaza mai multe inregistrari”.**

**--b. Dacă interogarea nu returnează nici o valoare pentru numărul comenzii, trataţi excepţia cu o rutină de tratare corespunzătoare şi inseraţi în tabela Mesaje mesajul “Interogarea nu returneaza inregistrari”.**

**--c. Dacă se returnează o singura linie, introduceţi în tabela Mesaje numărul comenzii.**

**--d. Trataţi orice altă excepţie cu o rutină de tratare corespunzătoare şi inseraţi în tabela MESAJE mesajul “A apărut o eroare!”.**

**--1**

**create table mesaje1 (camp varchar2(50));**

**--2**

**set serveroutput on**

**declare**

**v\_id comenzi.id\_comanda%type;**

**begin**

**select id\_comanda into v\_id from comenzi where extract(year from data) = &an;**

**dbms\_output.put\_line('Comanda este: ' || v\_id);**

**INSERT INTO mesaje1 VALUES(v\_id);**

**exception**

**when no\_data\_found then**

**dbms\_output.put\_line('Interogarea nu returneaza inregistrari');**

**when too\_many\_rows then**

**dbms\_output.put\_line('Interogarea returneaza mai multe inregistrari');**

**when others then**

**dbms\_output.put\_line('A aparut o eroare!');**

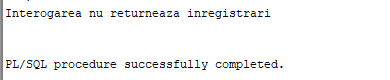
**DBMS\_OUTPUT.PUT\_LINE(SQLCODE);**

**DBMS\_OUTPUT.PUT\_LINE(SQLERRM);**

**end;**

**/**

**select \* from mesaje1;**



# Capitolul 6:

Proceduri

**Parametrii de tip IN**

EXERCITIUL 1:

--Procedura modifica\_salariul primeşte doi parametrii: p\_id\_medic şi procent şi majorează cu procentul specificat salariul medicului cu id\_medic=p\_id\_medic:

set SERVEROUTPUT ON

CREATE OR REPLACE

PROCEDURE modifica\_salariul\_procent

(p\_id\_doctor IN doctori.id\_doctor%type, procent IN number)

IS

v\_salariu doctori.salariu%type;

BEGIN

Select salariu into v\_salariu from doctori where id\_doctor=p\_id\_doctor;

dbms\_output.put\_line('Medicul are salariul de '||v\_salariu);

Update doctori

Set salariu=salariu\*(1+procent/100)

Where id\_doctor=p\_id\_doctor;

Select salariu into v\_salariu from doctori where id\_doctor=p\_id\_doctor;

Dbms\_output.put\_line('Medicul are acum salariul de '||v\_salariu);

END;

/

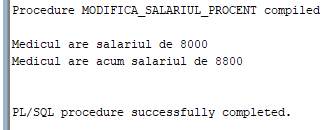
show errors;

begin

modifica\_salariul\_procent(8769, 10);

end;

/



**Parametrii de tip OUT**

EXERCITIUL 2:

--Procedura primeşte ca parametru de tip IN id\_ul unui medic şi returnează prin parametrii de tip OUT numele şi salariul acestuia:

CREATE OR REPLACE PROCEDURE cauta\_medic

(p\_id\_doctor IN doctori.id\_doctor%type,

p\_nume OUT doctori.nume\_d%type,

p\_salariul OUT doctori.salariu%type)

IS

BEGIN

Select nume\_d, salariu into p\_nume, p\_salariul from doctori where id\_doctor=p\_id\_doctor;

DBMS\_OUTPUT.PUT\_LINE(' Medicul '||p\_nume||' are salariul de: '||p\_salariul);

END;

/

SET SERVEROUTPUT ON

DECLARE

v\_nume doctori.nume\_d%type;

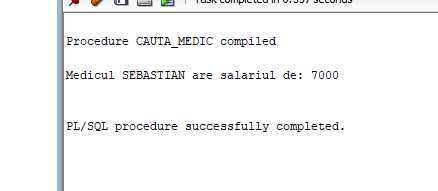
v\_salariul doctori.salariu%type;

BEGIN

Cauta\_medic(6679, v\_nume, v\_salariul);

END;

/



EXERCITIUL 3:

--Procedura calculează salariul mediu şi îl returnează printr-un parametru de tip OUT:

set SERVEROUTPUT ON

CREATE or REPLACE PROCEDURE sal\_mediu

(p\_sal\_mediu OUT number)

IS

BEGIN

Select AVG(salariu) into p\_sal\_mediu from doctori;

END;

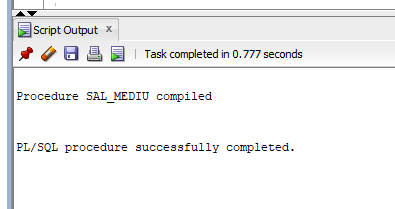
/

show errors;

VARIABLE v\_sal\_mediu NUMBER

EXECUTE sal\_mediu(:v\_sal\_mediu)

Print v\_sal\_mediu



**Parametrii de tip IN OUT**

EXERCITIUL 4:

--Procedura modifică salariul unui medic doar în cazul în care este mai mic decât media

--prin apelarea procedurii create mai sus, MODIFICA\_SALARIUL\_PROCENT. Procedura primeşte id-ul

--medicului ca parametru de intrare şi salariul mediu actual şi prin returnează prin parametrul

--de tip IN OUT salariul mediu modificat prin apelul procedurii SAL\_MEDIU.

CREATE or REPLACE PROCEDURE modifica\_salariul\_med

(p\_id\_doctor IN doctori.id\_doctor%type, p\_sal\_mediu IN OUT number)

IS

nume doctori.nume\_d%type;

sal doctori.salariu%type;

BEGIN

Select nume\_d, salariu into nume, sal from doctori where id\_doctor= p\_id\_doctor;

IF sal<p\_sal\_mediu then

MODIFICA\_SALARIUL\_PROCENT (p\_id\_doctor, 9768);

END IF;

SAL\_MEDIU (p\_sal\_mediu);

End;

/

SET SERVEROUTPUT ON

DECLARE

v\_id\_doctor doctori.id\_doctor%type;

v\_nume doctori.nume\_d%type;

v\_salariul doctori.salariu%type;

v\_sal\_mediu number;

BEGIN

v\_id\_doctor:=9768;

CAUTA\_MEDIC(v\_id\_doctor, v\_nume, v\_salariul);

SAL\_MEDIU (v\_sal\_mediu);

DBMS\_OUTPUT.PUT\_LINE('Salariul mediu este acum: '||round(v\_sal\_mediu,2));

modifica\_salariul\_med (v\_id\_doctor, v\_sal\_mediu);

CAUTA\_MEDIC(v\_id\_doctor, v\_nume, v\_salariul);

v\_id\_doctor:=123;

CAUTA\_MEDIC(v\_id\_doctor, v\_nume, v\_salariul);

modifica\_salariul\_med (v\_id\_doctor, v\_sal\_mediu);

Exception

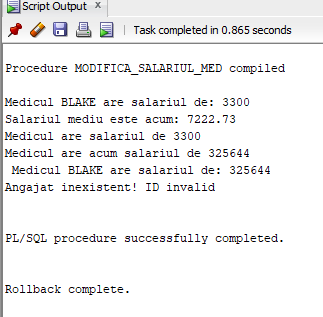
When NO\_DATA\_FOUND then

DBMS\_OUTPUT.PUT\_LINE('Angajat inexistent! ID invalid');

END;

/

rollback;



Functii

EXERCITIUL 5:

--Funcția verifica\_salariul returnează TRUE/FALSE daca salariatul are

--salariul mai mare/mai mic sau egal cu salariul mediu si NULL daca salariatul nu exista

CREATE OR REPLACE FUNCTION verifica\_salariul

(p\_id\_doctor IN doctori.id\_doctor%type, p\_sal\_mediu IN number)

RETURN Boolean

IS

v\_salariul doctori.salariu%type;

BEGIN

SELECT salariu into v\_salariul from doctori where id\_doctor=p\_id\_doctor;

IF v\_salariul > p\_sal\_mediu then

return true;

ELSE

return false;

end if;

EXCEPTION

WHEN no\_data\_found THEN

return NULL;

end;

/

show errors;

describe verifica\_salariul;

SET SERVEROUTPUT ON

DECLARE

v\_sal\_mediu number;

BEGIN

SAL\_MEDIU (v\_sal\_mediu);

IF (verifica\_salariul(11, v\_sal\_mediu) IS NULL) then

dbms\_output.put\_line('Angajat cu ID invalid!');

elsif (verifica\_salariul(11, v\_sal\_mediu)) then

dbms\_output.put\_line('Salariatul are salariul mai mare decat media!');

else

dbms\_output.put\_line(' Salariatul are salariul mai mic decat media!');

end if;

IF (verifica\_salariul(9876, v\_sal\_mediu) IS NULL) then

dbms\_output.put\_line('Angajat cu ID invalid!');

elsif (verifica\_salariul(110, v\_sal\_mediu)) then

dbms\_output.put\_line('Salariatul are salariul mai mare decat media!');

else

dbms\_output.put\_line(' Salariatul are salariul mai mic decat media!');

end if;

IF (verifica\_salariul(7774, v\_sal\_mediu) IS NULL) then

dbms\_output.put\_line('Angajat cu ID invalid!');

elsif (verifica\_salariul(7774, v\_sal\_mediu)) then

dbms\_output.put\_line('Salariatul are salariul mai mare decat media!');

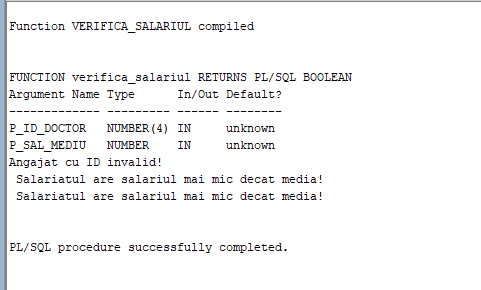
else

dbms\_output.put\_line(' Salariatul are salariul mai mic decat media!');

end if;

END;

/



EXERCITIUL 6:

CREATE OR REPLACE FUNCTION TAXA (value IN NUMBER, proc IN NUMBER)

RETURN NUMBER IS

BEGIN

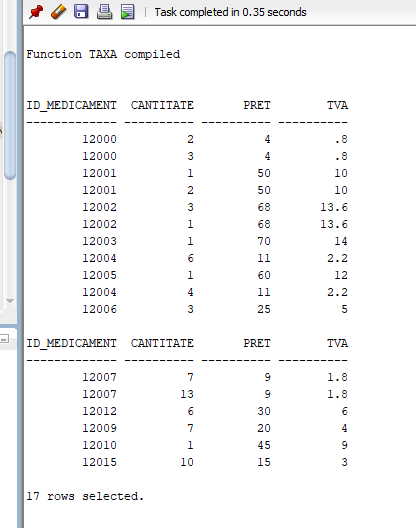
RETURN (value\*proc/100);

END taxa;

/

SELECT id\_medicament, cantitate, pret, taxa (pret, 20) as tva

FROM vanzari\_medicamente;



# Capitolul 7:

Pachete de subprograme

EXERCITIUL 1:

create or replace PACKAGE actualizare\_medicamente IS

procedure adauga\_medicament

(m\_codm medicamente.id\_medicament%type,

m\_denm medicamente.denumire%type,

m\_adm medicamente.administrare%type,

m\_stocm medicamente.stoc%type);

procedure modifica\_medicament

(m\_codm medicamente.id\_medicament%type,

m\_denm medicamente.denumire%type,

m\_adm medicamente.administrare%type,

m\_stocm medicamente.stoc%type);

procedure modifica\_medicament

(m\_codm medicamente.id\_medicament%type,

m\_denm medicamente.denumire%type);

procedure sterge\_medicament

(m\_codm medicamente.id\_medicament%type);

function exista\_cod

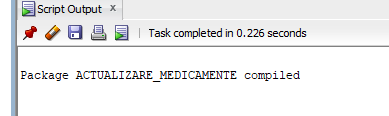
(m\_codm medicamente.id\_medicament%type)

return boolean;

exceptie exception;

END;

/



create or replace PACKAGE BODY actualizare\_medicamente IS

procedure adauga\_medicament

(m\_codm medicamente.id\_medicament%type,

m\_denm medicamente.denumire%type,

m\_adm medicamente.administrare%type,

m\_stocm medicamente.stoc%type)

is

begin

if exista\_cod(m\_codm) then

raise exceptie;

else

insert into medicamente(id\_medicament,denumire,administrare, stoc) values (m\_codm, m\_denm, m\_adm, m\_stocm);

end if;

exception

when exceptie then

dbms\_output.put\_line('Medicament existent!');

end;

procedure modifica\_medicament

(m\_codm medicamente.id\_medicament%type,

m\_denm medicamente.denumire%type,

m\_adm medicamente.administrare%type,

m\_stocm medicamente.stoc%type)

is

begin

if exista\_cod(m\_codm) then

update medicamente

set denumire=m\_denm,administrare=m\_adm, stoc=m\_stocm

where id\_medicament=m\_codm;

else

raise exceptie;

end if;

exception

when exceptie then

dbms\_output.put\_line('Medicamentul cu acest cod nu exista!');

end;

--supraîncarcare a procedurii modifica\_produs

procedure modifica\_medicament

(m\_codm medicamente.id\_medicament%type,

m\_denm medicamente.denumire%type)

is

begin

if exista\_cod(m\_codm) then

update medicamente

set stoc=m\_stocm

where id\_medicament=m\_codm;

else

raise exceptie;

end if;

exception

when exceptie then

dbms\_output.put\_line('Medicamentul cu acest cod nu exista!');

end;

procedure sterge\_medicament

(m\_codm medicamente.id\_medicament%type)

is

begin

if exista\_cod(m\_codm) then

delete from medicamente

where id\_medicament=m\_codm;

dbms\_output.put\_line('Medicamentul cu codul '||m\_codm||' a fost sters!');

else

raise exceptie;

end if;

exception

when exceptie then

dbms\_output.put\_line('Medicamentul cu acest cod nu exista!');

end;

function exista\_cod

(m\_codm medicamente.id\_medicament%type)

return boolean

is

v\_unu number;

begin

select 1 into v\_unu

from medicamente

where id\_medicament=m\_codm;

return true;

exception

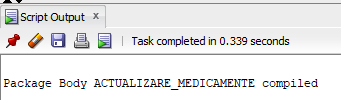
when no\_data\_found then

return false;

end;

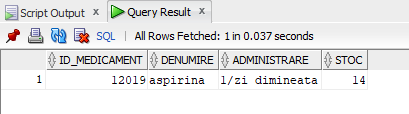
END;

/



execute actualizare\_medicamente.adauga\_medicament(12019,'aspirina', '1/zi dimineata', 14);

select \* from medicamente where id\_medicament=12019;



--Apelarea procedurii supra-încărcate

execute actualizare\_medicamente.modifica\_medicament(12019,15);

select \* from medicamente where id\_medicament=12019;

--Se poate folosi și CALL în loc de execute

call actualizare\_medicamente.modifica\_medicament(12019, null);

select \* from medicamente where id\_medicament=12019;

--Apelarea dintr-un bloc anonim

begin

actualizare\_medicamente.sterge\_medicament(12019);

end;

/

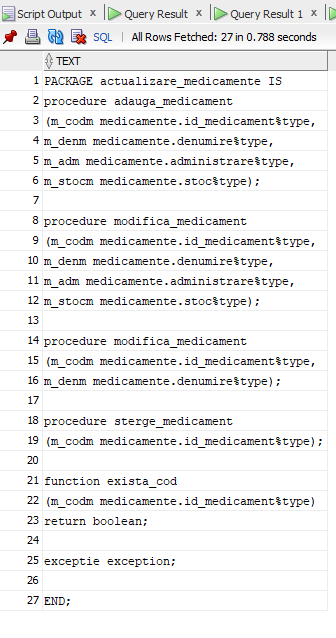
select \* from medicamente where id\_medicament=12019;

--Vizualizarea specificațiilor pachetelor in dicționarul metadatelor

Select text

From user\_source

Where name='ACTUALIZARE\_MEDICAMENTE' and type='PACKAGE'

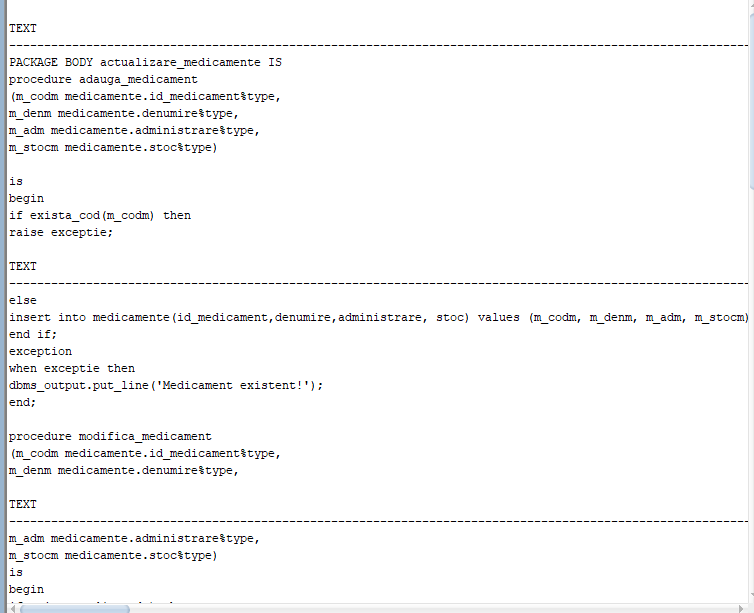


--vizualizarea corpului pachetului

Select text

From user\_source

Where name='ACTUALIZARE\_MEDICAMENTE' and type='PACKAGE BODY';



# Capitolul 8:

Triggeri

EXERCITIUL 1:

--Se creează un trigger care se declanşează înaintea fiecărei operaţii de inserare în tabela medicamente

CREATE OR REPLACE TRIGGER medicamente\_trig

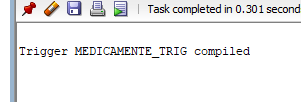
BEFORE INSERT ON medicamente

BEGIN

dbms\_output.put\_line('triggerul s-a executat');

END;

/



EXERCITIUL 2:

--Triggerul se declanşează la operaţiile de INSERT, DELETE sau UPDATE pe tabela Medicamente

--In tabela TEMP\_LOG se introduce tipul operaţiei, utilizatorul care a executat-o, data curentă.

drop table temp\_log;

CREATE TABLE temp\_log

(tip CHAR(10),

utilizator VARCHAR2(50),

data DATE DEFAULT SYSDATE);

CREATE OR REPLACE TRIGGER medicamente\_trig\_log

BEFORE INSERT or UPDATE or DELETE on medicamente

DECLARE

v\_tip temp\_log.tip%TYPE;

BEGIN

case

when INSERTING then v\_tip :='INSERT';

when UPDATING then v\_tip:='UPDATE';

ELSE v\_tip :='DELETE';

END case;

INSERT INTO temp\_log(tip, utilizator, data) VALUES (v\_tip, user, sysdate);

END;

/

--Verificarea execuţiei:

--inserarea in tabela

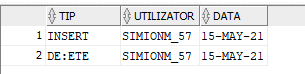
insert into medicamente (id\_medicament, denumire, administrare, stoc) values (12020, 'picaturi nas', 'la nevoie', 233);

--stergere

delete from medicamente where id\_medicament=12020;

commit;

Select \* from temp\_log;



EXERCITIUL 3:

--Se creează un trigger pentru a nu se permite depăşirea unei limite maxime a salariului unui medic

CREATE OR REPLACE TRIGGER restrictie\_salariu

BEFORE INSERT or UPDATE on doctori

FOR EACH ROW

DECLARE

v\_sal\_max number;

BEGIN

select salariu\_max into v\_sal\_max from functii\_medici where id\_functie= :new.id\_functie;

IF :new.salariu>v\_sal\_max then

RAISE\_APPLICATION\_ERROR (-20202, 'Nu se poate depasi salariul maxim pentru functia data');

end if;

END;

/

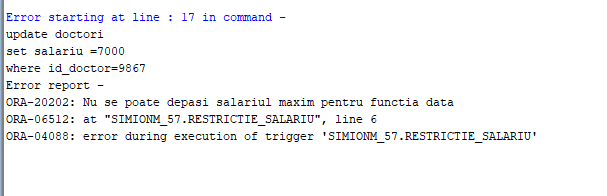


--verificare executie

update doctori

set salariu =7000

where id\_doctor=9867;



EXERCITIUL 4:

--Se creează un trigger care asigură unicitatea codului medicamentului folosind valorile generate de o secvenţă

CREATE SEQUENCE medicamente\_secv

START WITH 1

INCREMENT BY 1

MAXVALUE 100

NOCYCLE;

CREATE OR REPLACE TRIGGER generare\_codmedicament

BEFORE INSERT ON medicamente

FOR EACH ROW

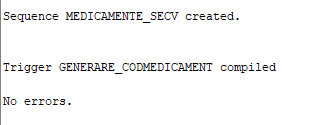
BEGIN

SELECT medicamente\_secv.nextval INTO :new.id\_medicament FROM dual;

END;

/

show errors;



--crearea tabelei virtuale

create or replace view pacienti\_v as

select pa.id\_pacient, pa.prenume\_p, pa.nume\_p, pa.grupa\_sange,fm.cod\_fisa, fm.data\_internarii

from pacienti pa, fise\_medicale fm

where pa.id\_pacient=fm.id\_pacient;

--crearea triggerului

create or replace trigger pac\_trigger

instead of insert or update or delete on pacienti\_v

for each row

begin

if inserting then

insert into pacienti(id\_pacient, prenume\_p, nume\_p, grupa\_sange) values (:new.id\_pacient, :new.prenume\_p, :new.nume\_p, :new.grupa\_sange);

insert into fise\_medicale (cod\_fisa, data\_internarii, id\_pacient) values (:new.cod\_fisa, :new.data\_internarii, :new.id\_pacient);

elsif deleting then

delete from fise\_medicale where cod\_fisa=:old.cod\_fisa;

elsif updating ('nume\_pacient') then

update pacienti

set nume\_p=:new.nume\_p

where id\_pacient=:old.id\_pacient;

elsif updating ('data\_internarii') then

update fise\_medicale

set data\_internarii=:new.data\_internarii

where cod\_fisa=:old.cod\_fisa;

end if;

end;

/

show errors;

--testare trigger

insert into pacienti\_v values (3352,'Sarah','Maas','B3 POZ',566,sysdate);

insert into pacienti\_v values (4429,'Andreea','Crammer','AB4 NEG',156,sysdate);

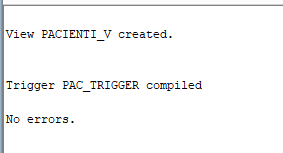
delete from pacienti\_v where nume\_p='Andreea';

update pacienti\_v

set nume\_p='Popescu'

where id\_pacient=4429;

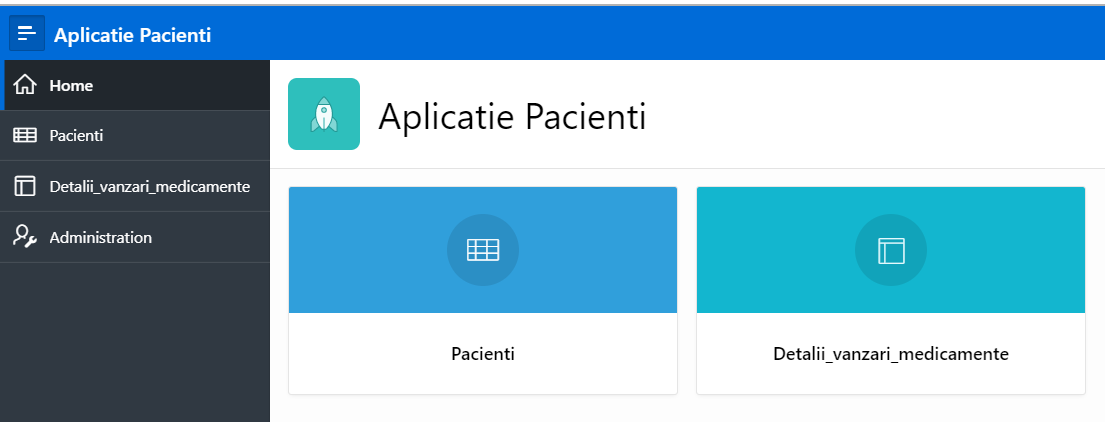
select \* from pacienti where id\_pacient =4429;



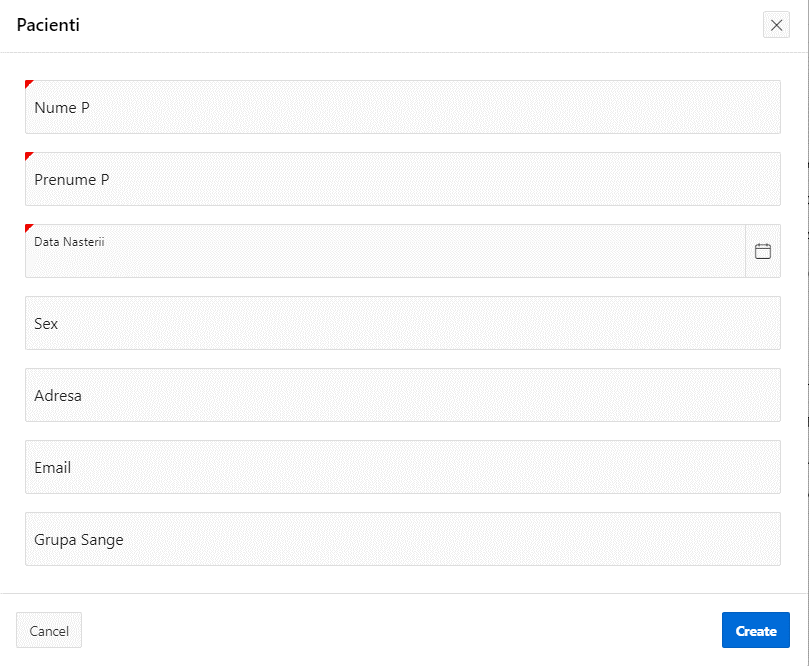
# Capitolul 9:

APEX

Am creat o aplicaţie pentru pacienţi si vânzările de medicamente. Care conţine o pagină de tipul *Interactive report* şi una de tipul *Master Detail.*

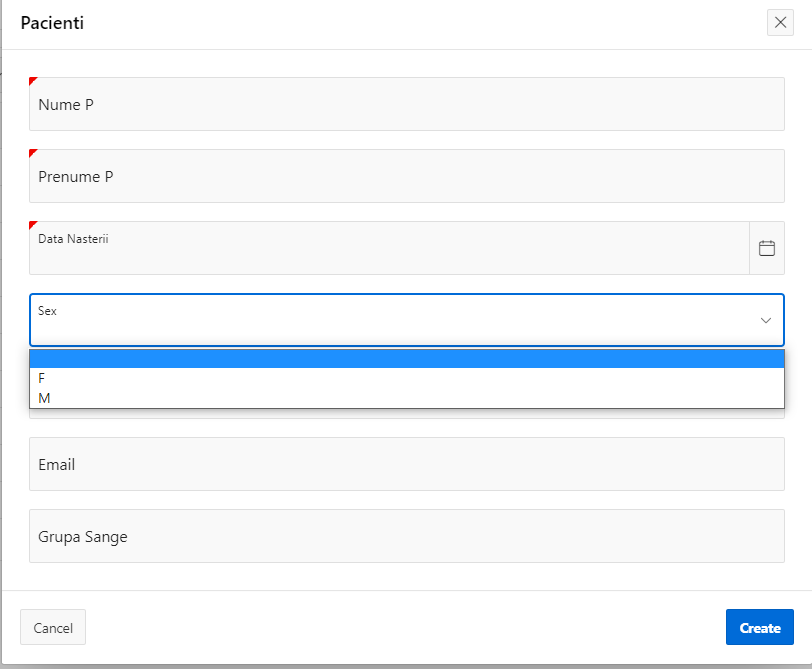


Mai departe am setat ca pentru formularul de adaugare a unui pacient, câmpurile *sex* şi *grupa de sânge* să avem câte o listă de valori din care să se poată alege variantele corespunzătoare în cazul efectuării unor modificări ale înregistrărilor tabelei sau în cazul adăugării unui pacient nou.

 🡪Formularul înainte de modificare

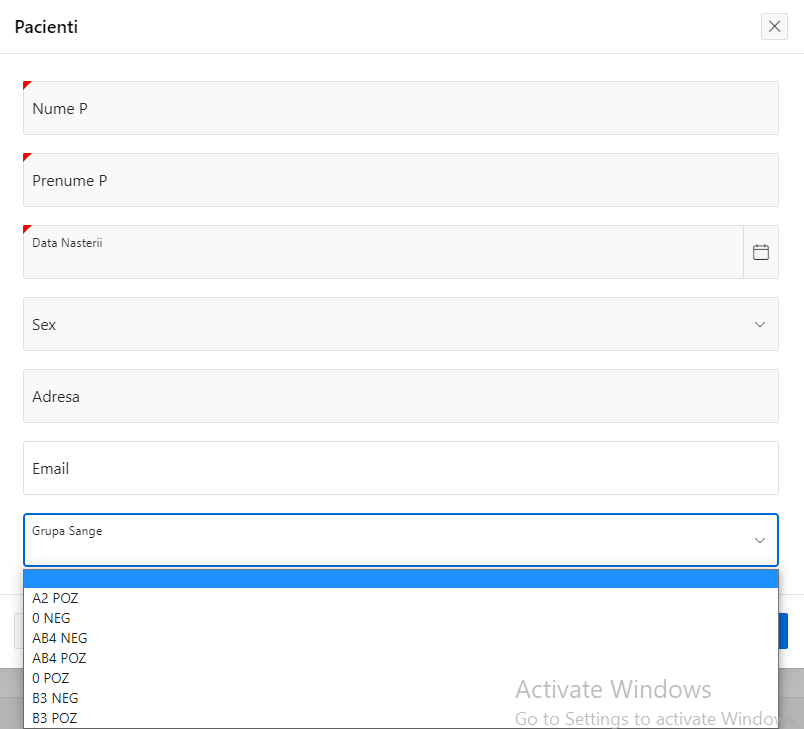
Astfel, pentru atributul **Sex**, se alege pentru proprietatea **Type** valoarea **Select List**. În tabul **List of values** alegem **SQL query** şi deoarece dorim să fie preluate valorile distincte aferente sexului, vom completa corespunzător interogarea:

select distinct sex as display\_value, sex as return\_value from pacienti

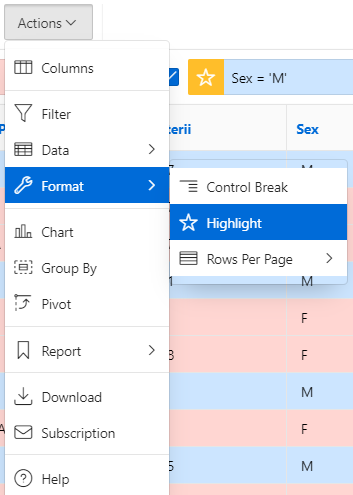
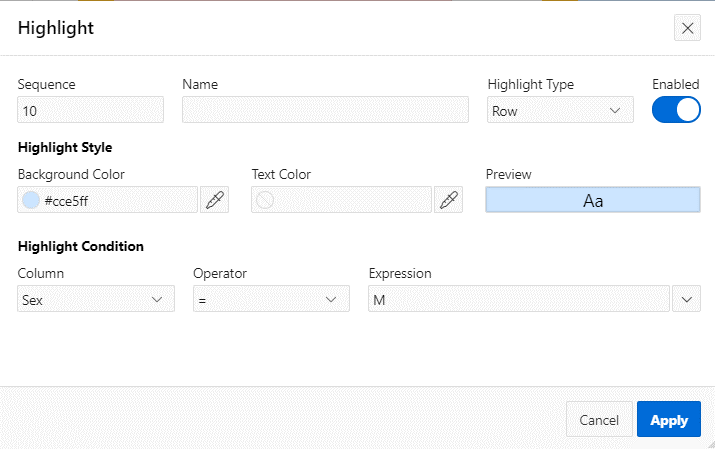
🡪 Formularul după modificare

Pentru atributul **Grupă de sânge** am urmat aceeaşi paşi, completând interogarea astfel:

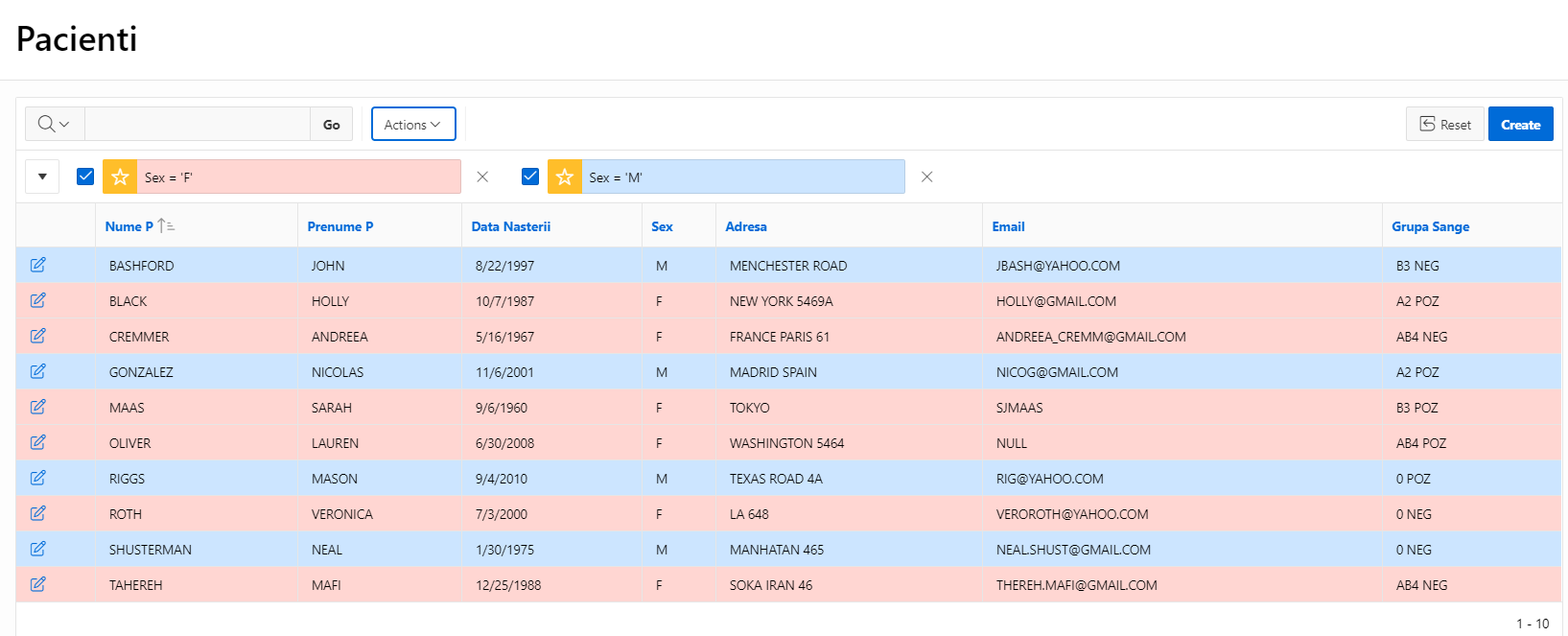
select distinct grupa\_sange as display\_value, grupa\_sange as return\_value from pacienti

🡪Formularul după modificare

Mai departe, pentru a evidenţia sexul pacienţilor m-am folosit de meniul **Actions**, opţiunea **Highlight** de unde am adăugat un fundal colorat, mai exact roz pentru sexul feminin şi albastru pentru masculin.

**DUPA MODIFICARE:**



Pentru pagina **Detalii vânzări medicamente** am vrut ca în momentul selectării unui camp să poată fi vizibilă şi denumirea medicamentului, nu doar identificatorul acestuia. Astfel, adăugăm câmpul **id\_medicament** și **denumire** în lista de medicamente comandate si adăugăm o listă de valori (de tipul **Popup LOV**) şi pentru câmpul **id\_medicament.** La nivelul grupului de proprietăţi **SQL Query**, vom scrie interogarea:

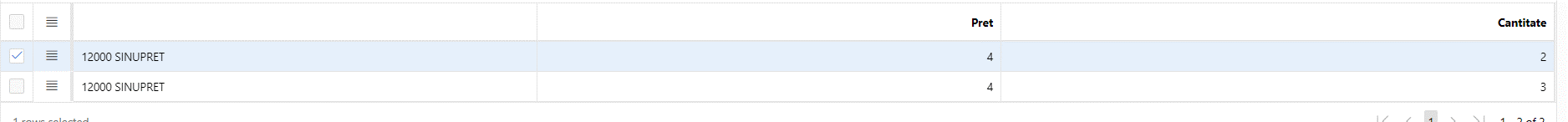
select id\_medicament||' '||denumire as display\_value,

id\_medicament as return\_value

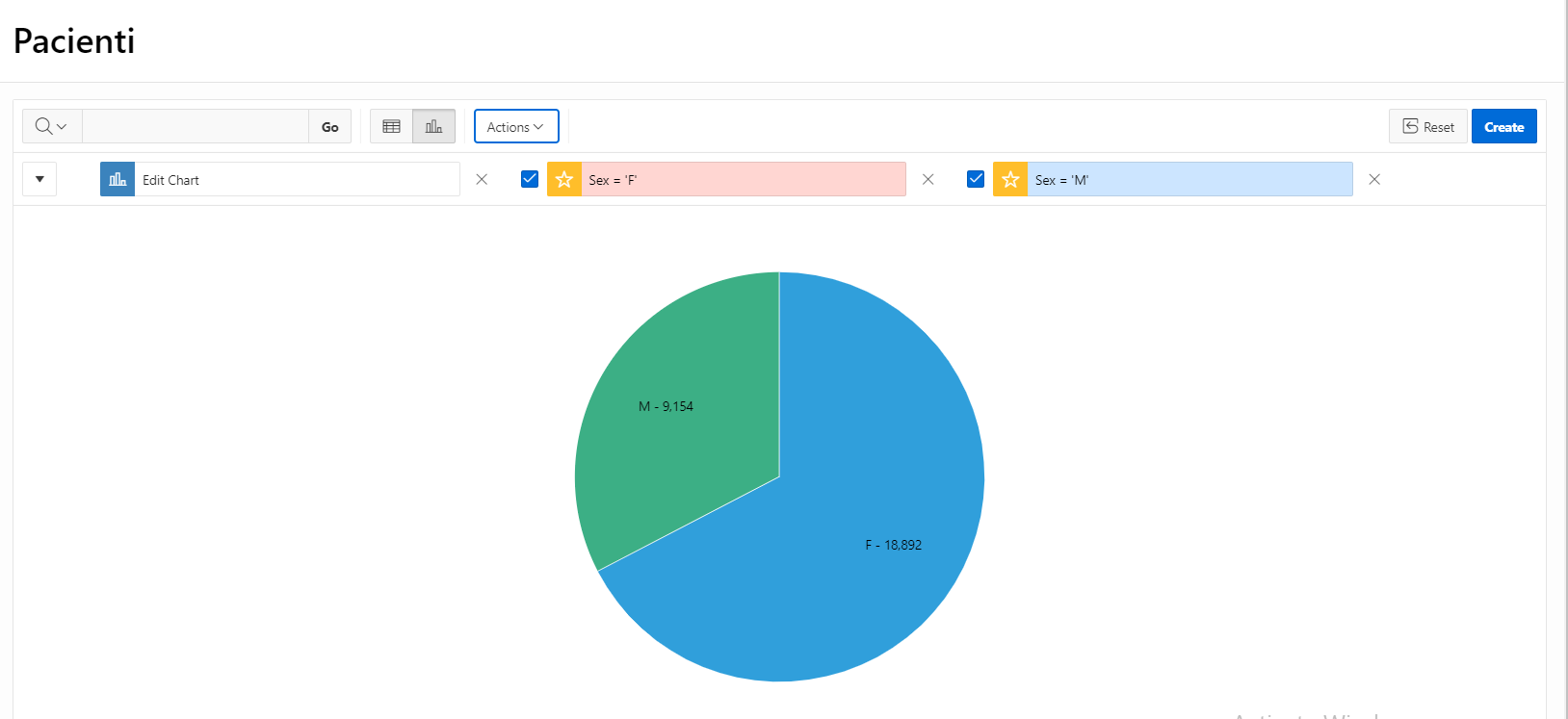
from medicamente

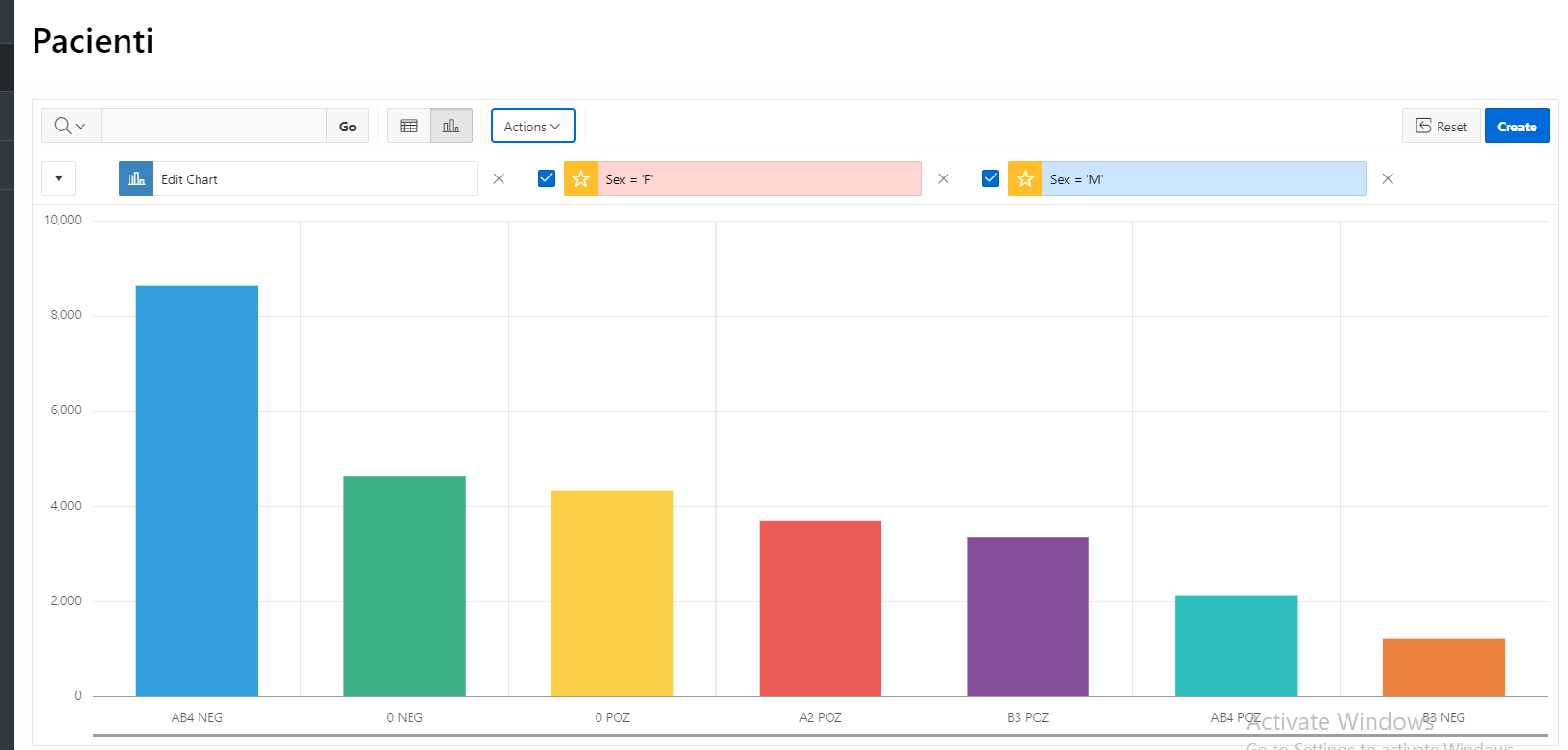
order by id\_medicament

**REZULTATUL:**

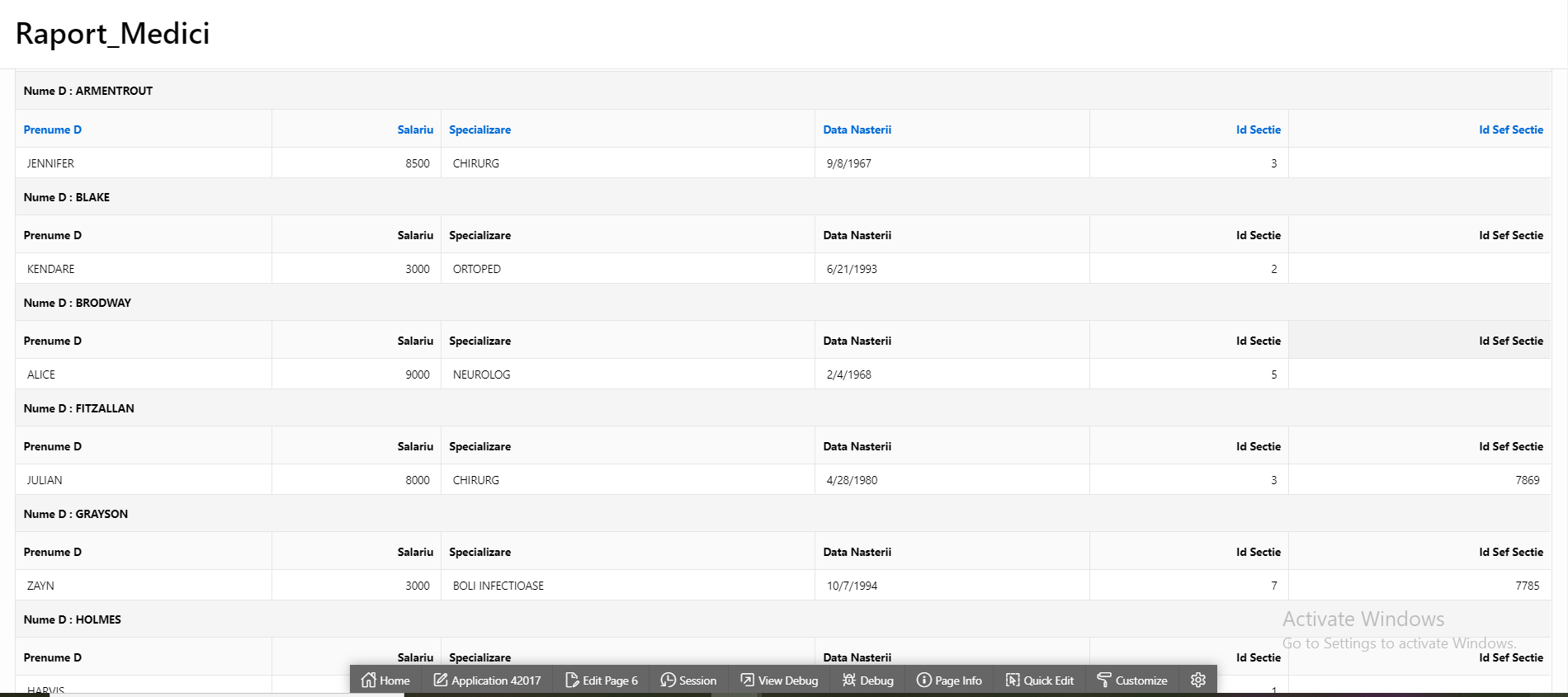


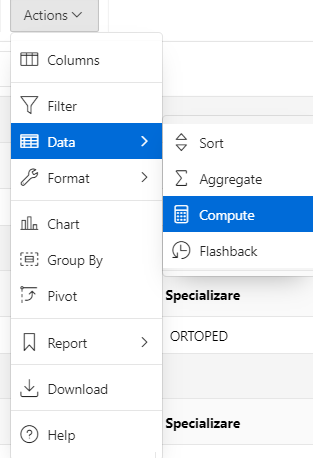
În final am generat două grafice, unul de tip **PIE** (care înfăţişează ponderea sexelor)şi unul de tip **BARS** (care înfăţişează ponderea grupelor de sânge).

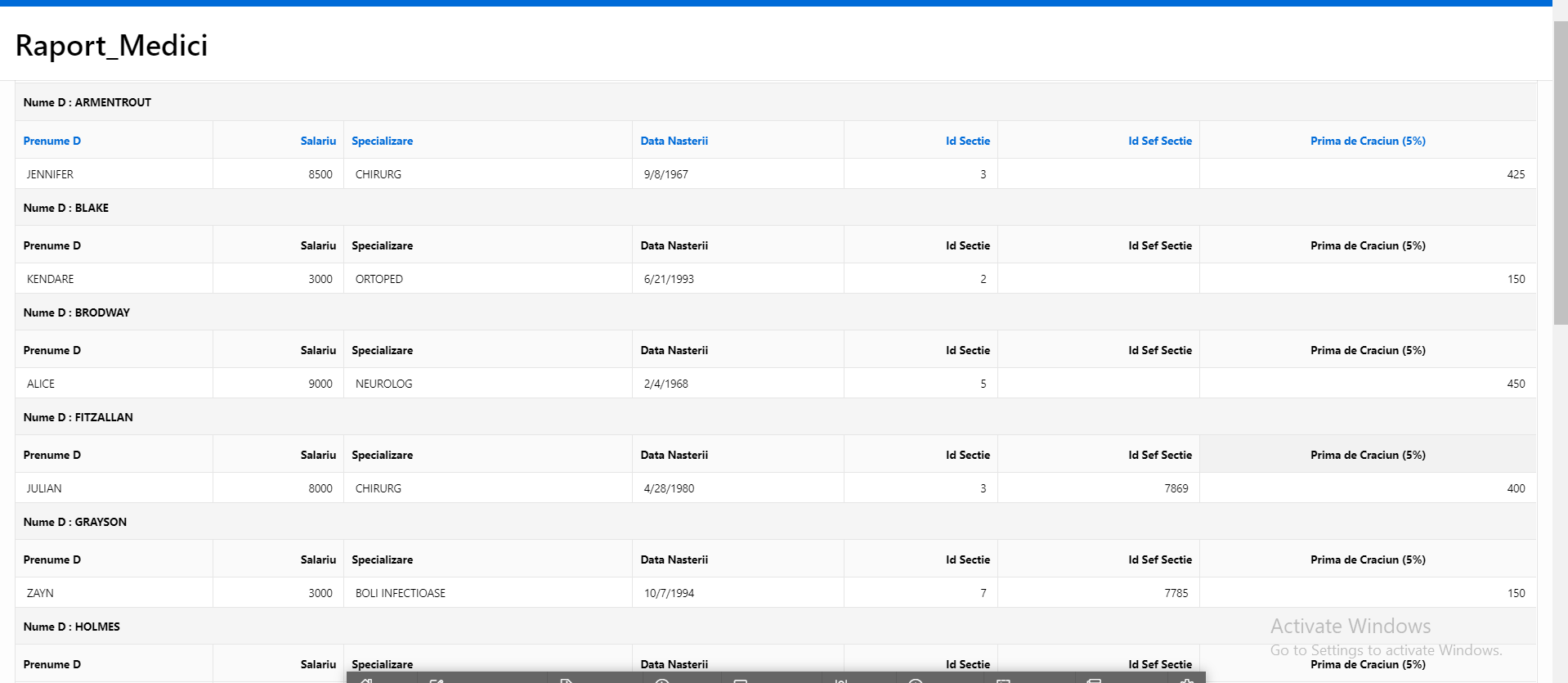


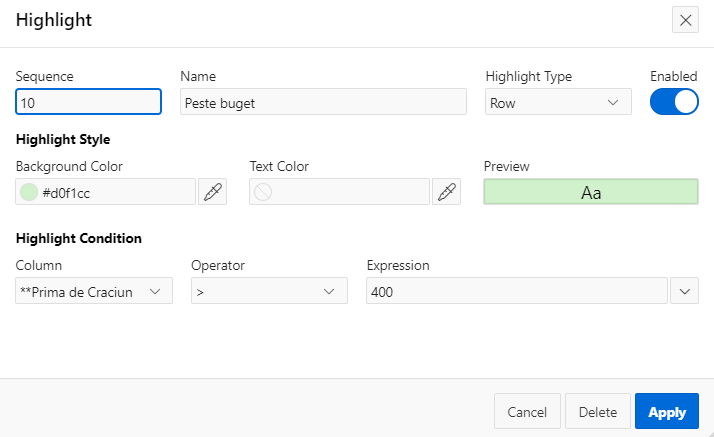


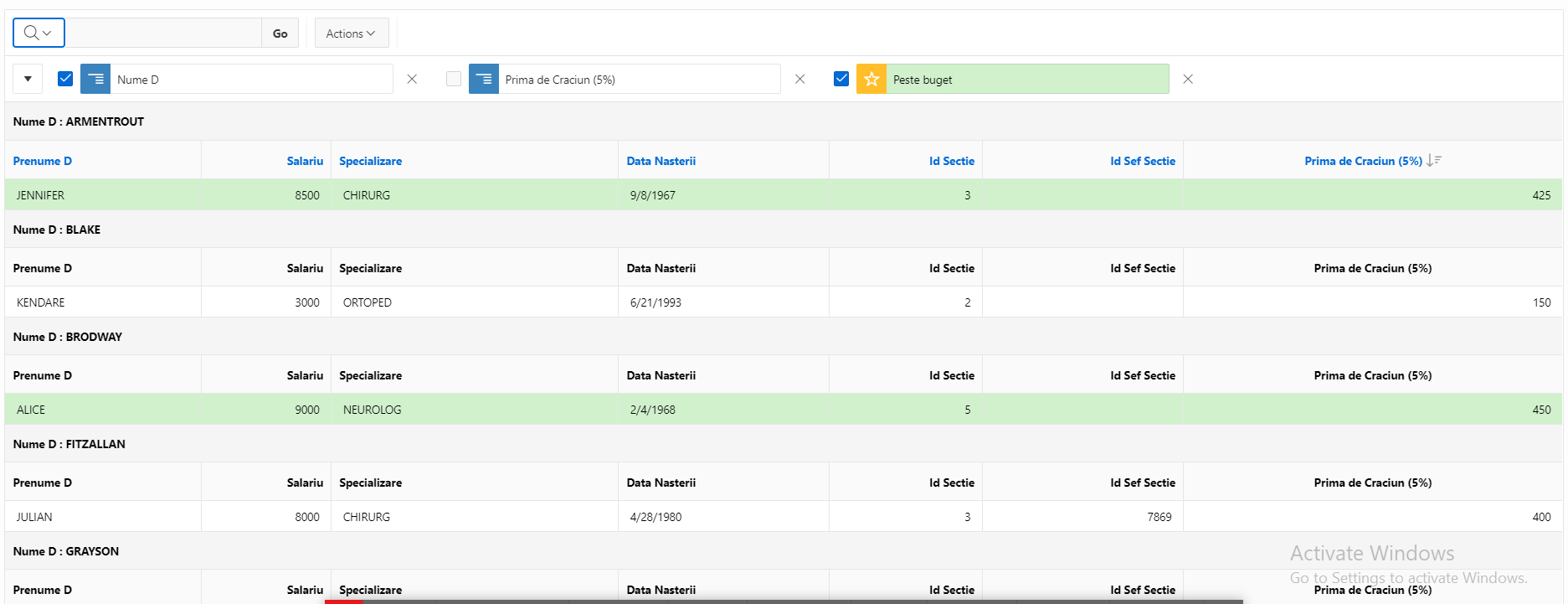
Am creat un Interactive report pentru tabela Doctori.



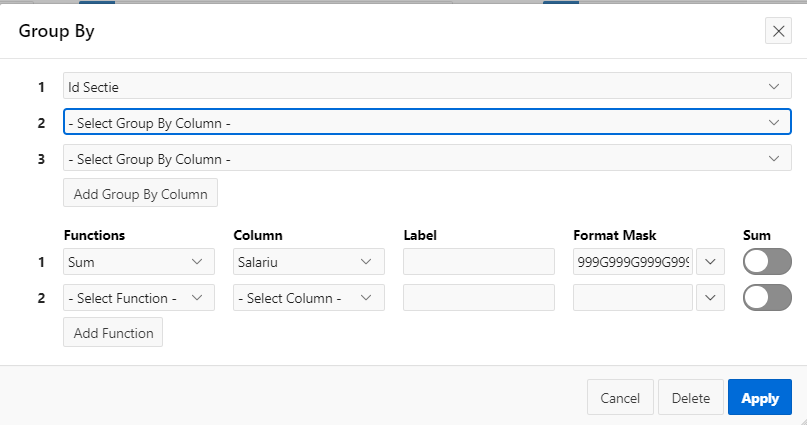
 Folosind funcţia de Compute am adaugat o coloană ce calculează cât înseamnă prima de Crăciun pentru fiecare Medic. Prima reprezintă 5% din salariu.

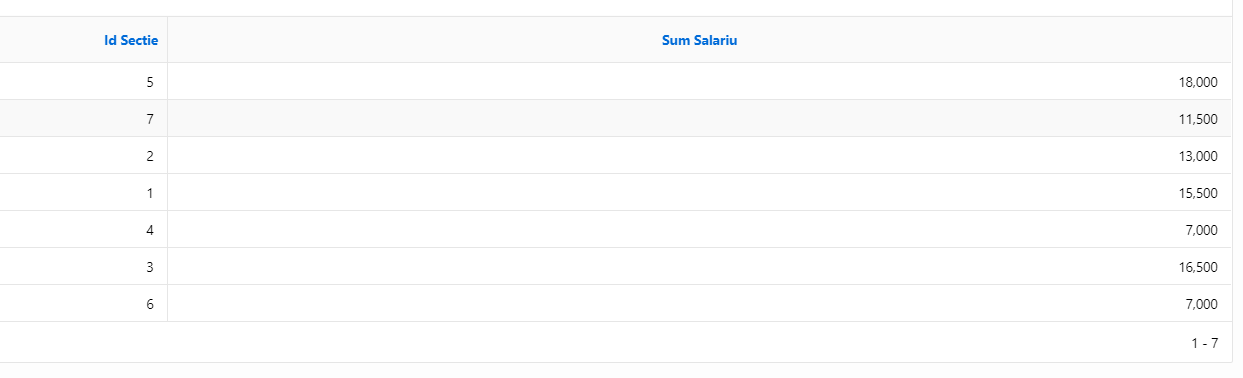


 Utilizând opţiunea de Highlight am evidenţiat cu verde primele care trec de suma 400, deoarece depăşesc bugetul.

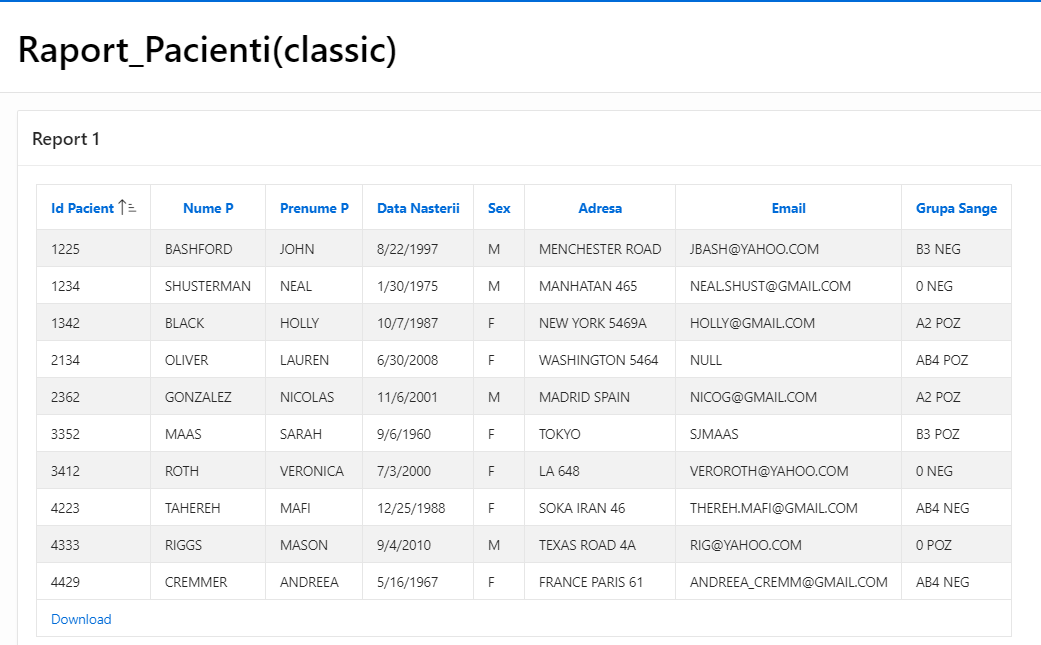


Am folosit apoi funcţia Group by pentru a vedea valoarea salariilor de platit pe fiecare sectie.





Classic report pentru tabela Pacienţi:



ANEXE

# Construirea bazei de date- tabele și restricții de integritate:

**Tabela pacienți:**

CREATE TABLE PACIENTI(

ID\_PACIENT NUMBER(5),

NUME\_P VARCHAR2(10) NOT NULL,

PRENUME\_P VARCHAR2(10) NOT NULL,

DATA\_NASTERII DATE NOT NULL,

SEX VARCHAR2(2),

ADRESA VARCHAR2(15),

EMAIL VARCHAR2(10),

GRUPA\_SANGE VARCHAR2(5),

CONSTRAINT PACIENTI\_PK PRIMARY KEY(ID\_PACIENT)

);

**Tabela doctori**

CREATE TABLE DOCTORI(

ID\_DOCTOR NUMBER(4),

NUME\_D VARCHAR2(10) NOT NULL,

PRENUME\_D VARCHAR2(10) NOT NULL,

SALARIU NUMBER(7) NOT NULL,

SPECIALIZARE VARCHAR2(15),

DATA\_NASTERII DATE,

ID\_SECTIE NUMBER(2),

CONSTRAINT DOCTORI\_PK PRIMARY KEY(ID\_DOCTOR),

CONSTRAINT FK\_ID\_SECTIE FOREIGN KEY (ID\_SECTIE) REFERENCES SECTII(ID\_SECTIE)

);

**Tabela secții:**

CREATE TABLE SECTII(

ID\_SECTIE NUMBER(2),

DENUMIRE VARCHAR2(20) NOT NULL,

CONSTRAINT SECTII\_PK PRIMARY KEY(ID\_SECTIE)

);

**Tabela fișe\_medicale:**

CREATE TABLE FISE\_MEDICALE(

COD\_FISA NUMBER (5),

ID\_PACIENT NUMBER(4),

ID\_DOCTOR NUMBER(4),

ID\_SECTIE NUMBER(2),

DIAGNOSTIC VARCHAR2(20) NOT NULL,

DATA\_INTERNARII DATE,

DATA\_EXTERNARII DATE,

CONSTRAINT FISE\_PK PRIMARY KEY(COD\_FISA),

CONSTRAINT FK\_ID\_PACIENT FOREIGN KEY (ID\_PACIENT) REFERENCES PACIENTI(ID\_PACIENT),

CONSTRAINT FK\_ID\_DOCTOR FOREIGN KEY (ID\_DOCTOR) REFERENCES DOCTORI(ID\_DOCTOR),

CONSTRAINT FK\_ID\_SECTIE FOREIGN KEY (ID\_SECTIE) REFERENCES SECTII(ID\_SECTIE)

);

**Tabela rețete:**

CREATE TABLE RETETE(

COD\_RETETA NUMBER(3),

ID\_DOCTOR NUMBER(4),

COD\_FISA NUMBER(5),

CONSTRAINT RETETE\_PK PRIMARY KEY(COD RETETA),

CONSTRAINT FK\_COD\_FISA FOREIGN KEY (COD\_FISA) REFERENCES FISE\_MEDICALE(COD\_FISA),

CONSTRAINT FK\_ID\_DOCTOR FOREIGN KEY (ID\_DOCTOR) REFERENCES DOCTORI(ID\_DOCTOR)

);

**Tabela medicamente:**

CREATE TABLE MEDICAMENTE(

ID\_MEDICAMENT NUMBER(6),

DENUMIRE VARCHAR2(15),

ADMINISTRARE VARCHAR2(30),

CONSTRAINT MEDICAMENTE\_PK PRIMARY KEY(ID\_MEDICAMENT);

**Tabela tratamente:**

CREATE TABLE TRATAMENTE(

COD\_TRATAMENT NUMBER(2),

ID\_MEDICAMENT NUMBER(5),

COD\_RETETA NUMBER(3),

CONSTRAINT TRATAMENTE\_PK PRIMARY KEY(COD\_TRATAMENT),

CONSTRAINT FK\_COD\_RETETA FOREIGN KEY (COD\_RETETA) REFERENCES RETETE(COD\_RETETA),

CONSTRAINT FK\_ID\_MEDICAMENT FOREIGN KEY (ID\_MEDICAMENT) REFERENCES MEDICAMENTE(ID\_MEDICAMENT));

# Adăugarea înregistrărilor:

**PACIENȚI**

--SA ADAUGAM VALORI

ALTER TABLE PACIENTI MODIFY EMAIL VARCHAR2(25);

ALTER TABLE PACIENTI MODIFY GRUPA\_SANGE VARCHAR2(7);

INSERT INTO PACIENTI (ID\_PACIENT, NUME\_P,PRENUME\_P,DATA\_NASTERII, SEX, ADRESA, EMAIL,GRUPA\_SANGE) VALUES (1234, 'SHUSTERMAN', 'NEAL', TO\_DATE('30.01.1975','DD.MM.YYYY'),'M','MANHATAN 465','NEAL.SHUST@GMAIL.COM' ,'0 NEG');

INSERT INTO PACIENTI (ID\_PACIENT, NUME\_P,PRENUME\_P,DATA\_NASTERII, SEX, ADRESA, EMAIL,GRUPA\_SANGE) VALUES (1342, 'BLACK', 'HOLLY', TO\_DATE('07.10.1987','DD.MM.YYYY'), 'F','NEW YORK 5469A','HOLLY@GMAIL.COM' , 'A2 POZ');

INSERT INTO PACIENTI (ID\_PACIENT,NUME\_P,PRENUME\_P,DATA\_NASTERII, SEX, ADRESA, EMAIL,GRUPA\_SANGE) VALUES (3412, 'ROTH', 'VERONICA', TO\_DATE('03.07.2000','DD.MM.YYYY'),'F','LA 648','VEROROTH@YAHOO.COM' , '0 NEG');

INSERT INTO PACIENTI (ID\_PACIENT,NUME\_P,PRENUME\_P,DATA\_NASTERII, SEX, ADRESA, EMAIL,GRUPA\_SANGE) VALUES (2134, 'OLIVER', 'LAUREN', TO\_DATE('30.06.2008','DD.MM.YYYY'),'F','WASHINGTON 5464','NULL' , 'AB4 POZ');

INSERT INTO PACIENTI (ID\_PACIENT,NUME\_P,PRENUME\_P,DATA\_NASTERII, SEX, ADRESA, EMAIL,GRUPA\_SANGE) VALUES (4223, 'TAHEREH', 'MAFI', TO\_DATE('25.12.1988','DD.MM.YYYY'),'F','SOKA IRAN 46','THEREH.MAFI@GMAIL.COM' , 'AB4 NEG');

INSERT INTO PACIENTI (ID\_PACIENT,NUME\_P,PRENUME\_P,DATA\_NASTERII, SEX, ADRESA, EMAIL,GRUPA\_SANGE) VALUES (3352, 'MAAS', 'SARAH', TO\_DATE('06.09.1960','DD.MM.YYYY'),'F','TOKYO','SJMAAS','B3 POZ');

INSERT INTO PACIENTI (ID\_PACIENT,NUME\_P,PRENUME\_P,DATA\_NASTERII, SEX, ADRESA, EMAIL,GRUPA\_SANGE) VALUES (4333, 'RIGGS', 'MASON', TO\_DATE('04.09.2010','DD.MM.YYYY'),'M','TEXAS ROAD 4A','RIG@YAHOO.COM' , '0 POZ');

INSERT INTO PACIENTI (ID\_PACIENT,NUME\_P,PRENUME\_P,DATA\_NASTERII, SEX, ADRESA, EMAIL,GRUPA\_SANGE) VALUES (1225, 'BASHFORD', 'JOHN', TO\_DATE('22.08.1997','DD.MM.YYYY'),'M','MENCHESTER ROAD','JBASH@YAHOO.COM' , 'B3 NEG');

INSERT INTO PACIENTI (ID\_PACIENT,NUME\_P,PRENUME\_P,DATA\_NASTERII, SEX, ADRESA, EMAIL,GRUPA\_SANGE) VALUES (2362, 'GONZALEZ', 'NICOLAS', TO\_DATE('06.11.2001','DD.MM.YYYY'),'M','MADRID SPAIN','NICOG@GMAIL.COM' , 'A2 POZ');

INSERT INTO PACIENTI (ID\_PACIENT,NUME\_P,PRENUME\_P,DATA\_NASTERII, SEX, ADRESA, EMAIL,GRUPA\_SANGE) VALUES (4429, 'CREMMER', 'ANDREEA',TO\_DATE( '16.05.1967','DD.MM.YYYY'),'F','FRANCE PARIS 61','ANDREEA\_CREMM@GMAIL.COM' , 'AB4 NEG');

**DOCTORI**

ALTER TABLE DOCTORI MODIFY SPECIALIZARE VARCHAR(17);

INSERT INTO DOCTORI (ID\_DOCTOR,NUME\_D,PRENUME\_D, SALARIU, SPECIALIZARE, DATA\_NASTERII, ID\_SECTIE) VALUES (9876, 'WHITEFIELD', 'ANDREAS', 7000,'ORTOPED',TO\_DATE('01.09.1985','DD.MM.YYYY'),2);

INSERT INTO DOCTORI (ID\_DOCTOR,NUME\_D,PRENUME\_D, SALARIU, SPECIALIZARE, DATA\_NASTERII, ID\_SECTIE) VALUES (9867, 'SMITH', 'RICHARD', 3000,'ORTOPED',TO\_DATE('09.04.1995','DD.MM.YYYY'),2);

INSERT INTO DOCTORI (ID\_DOCTOR,NUME\_D,PRENUME\_D, SALARIU, SPECIALIZARE, DATA\_NASTERII, ID\_SECTIE) VALUES (9768, 'BLAKE', 'KENDARE',3000,'ORTOPED',TO\_DATE('21.06.1993','DD.MM.YYYY'),2);

**INSERT INTO DOCTORI (ID\_DOCTOR,NUME\_D,PRENUME\_D, SALARIU, SPECIALIZARE, DATA\_NASTERII, ID\_SECTIE) VALUES (8976, 'SIMION', 'MARIA', 7500,'PEDIATRU',TO\_DATE('03.07.1990','DD.MM.YYYY'),1);**

INSERT INTO DOCTORI (ID\_DOCTOR,NUME\_D,PRENUME\_D, SALARIU, SPECIALIZARE, DATA\_NASTERII, ID\_SECTIE) VALUES (8769, 'HOLMES', 'HARVIS', 8000,'PEDIATRU',TO\_DATE('14.05.1965','DD.MM.YYYY'),1);

INSERT INTO DOCTORI (ID\_DOCTOR,NUME\_D,PRENUME\_D, SALARIU, SPECIALIZARE, DATA\_NASTERII, ID\_SECTIE) VALUES (8697, 'FITZALLAN', 'JULIAN', 8000,'CHIRURG',TO\_DATE('28.04.1980','DD.MM.YYYY'),3);

INSERT INTO DOCTORI (ID\_DOCTOR,NUME\_D,PRENUME\_D, SALARIU, SPECIALIZARE, DATA\_NASTERII, ID\_SECTIE) VALUES (7869, 'ARMENTROUT', 'JENNIFER', 8500,'CHIRURG',TO\_DATE('08.09.1967','DD.MM.YYYY'),3);

INSERT INTO DOCTORI (ID\_DOCTOR,NUME\_D,PRENUME\_D, SALARIU, SPECIALIZARE, DATA\_NASTERII, ID\_SECTIE) VALUES (7688, 'PURDIE', 'KATHERINE', 7000,'ORL-IST',TO\_DATE('30.06.1975','DD.MM.YYYY'),4);

INSERT INTO DOCTORI (ID\_DOCTOR,NUME\_D,PRENUME\_D, SALARIU, SPECIALIZARE, DATA\_NASTERII, ID\_SECTIE) VALUES (9877, 'LEWIS', 'CAROL', 9000,'NEUROLOG',TO\_DATE('18.07.1970','DD.MM.YYYY'),5);

INSERT INTO DOCTORI (ID\_DOCTOR,NUME\_D,PRENUME\_D, SALARIU, SPECIALIZARE, DATA\_NASTERII, ID\_SECTIE) VALUES (6789, 'BRODWAY', 'ALICE', 9000,'NEUROLOG',TO\_DATE('04.02.1968','DD.MM.YYYY'),5);

INSERT INTO DOCTORI (ID\_DOCTOR,NUME\_D,PRENUME\_D, SALARIU, SPECIALIZARE, DATA\_NASTERII, ID\_SECTIE) VALUES (6679, 'SEBASTIAN', 'LAURA', 7000,'DERMATOLOG',TO\_DATE('08.03.1975','DD.MM.YYYY'),6);

INSERT INTO DOCTORI (ID\_DOCTOR,NUME\_D,PRENUME\_D, SALARIU, SPECIALIZARE, DATA\_NASTERII, ID\_SECTIE) VALUES (7785, 'ROSS', 'REBECCA', 8500,'BOLI INFECTIOASE',TO\_DATE('13.04.1974','DD.MM.YYYY'),7);

INSERT INTO DOCTORI (ID\_DOCTOR,NUME\_D,PRENUME\_D, SALARIU, SPECIALIZARE, DATA\_NASTERII, ID\_SECTIE) VALUES (7774, 'GRAYSON', 'ZAYN', 3000,'BOLI INFECTIOASE',TO\_DATE('07.10.1994','DD.MM.YYYY'),7);

**SECȚII**

INSERT INTO SECTII(ID\_SECTIE, DENUMIRE) VALUES (1,'PEDIATRIE');

INSERT INTO SECTII(ID\_SECTIE, DENUMIRE) VALUES (2,'ORTOPEDIE');

INSERT INTO SECTII(ID\_SECTIE, DENUMIRE) VALUES (3,'CHIRURGIE');

INSERT INTO SECTII(ID\_SECTIE, DENUMIRE) VALUES (4,'ORL');

INSERT INTO SECTII(ID\_SECTIE, DENUMIRE) VALUES (5,'NEUROLOGIE');

INSERT INTO SECTII(ID\_SECTIE, DENUMIRE) VALUES (6,'DERMATOLOGIE');

INSERT INTO SECTII(ID\_SECTIE, DENUMIRE) VALUES (7,'BOLI INFECTIOASE');

**FIȘE\_MEDICALE**

INSERT INTO FISE\_MEDICALE(COD\_FISA, ID\_PACIENT, ID\_DOCTOR, DIAGNOSTIC, DATA\_INTERNARII, DATA\_EXTERNARII, ID\_SECTIE) VALUES(455,1234,7688,'OTITA',TO\_DATE('01.09.2020','DD.MM.YYYY'),TO\_DATE('01.09.2020','DD.MM.YYYY'),4);

INSERT INTO FISE\_MEDICALE(COD\_FISA, ID\_PACIENT, ID\_DOCTOR, DIAGNOSTIC, DATA\_INTERNARII, DATA\_EXTERNARII, ID\_SECTIE) VALUES(435,1342,6679,'ACNEE SEVERA',TO\_DATE('16.10.2020','DD.MM.YYYY'),TO\_DATE('16.10.2020','DD.MM.YYYY'),6);

INSERT INTO FISE\_MEDICALE(COD\_FISA, ID\_PACIENT, ID\_DOCTOR, DIAGNOSTIC, DATA\_INTERNARII, DATA\_EXTERNARII, ID\_SECTIE) VALUES(445,3412,9867,'FRACTURA USOARA',TO\_DATE('21.06.2020','DD.MM.YYYY'),TO\_DATE('21.06.2020','DD.MM.YYYY'),2);

INSERT INTO FISE\_MEDICALE(COD\_FISA, ID\_PACIENT, ID\_DOCTOR, DIAGNOSTIC, DATA\_INTERNARII, DATA\_EXTERNARII, ID\_SECTIE) VALUES(432,2134,9876,'RUPTURA DE LIGAMENT',TO\_DATE('19.08.2020','DD.MM.YYYY'),TO\_DATE('23.08.2020','DD.MM.YYYY'),2);

INSERT INTO FISE\_MEDICALE(COD\_FISA, ID\_PACIENT, ID\_DOCTOR, DIAGNOSTIC, DATA\_INTERNARII, DATA\_EXTERNARII, ID\_SECTIE) VALUES(456,4223,6679,'DERMATITA SEBOREICA',TO\_DATE('21.04.2020','DD.MM.YYYY'),TO\_DATE('21.04.2020','DD.MM.YYYY')6),;

INSERT INTO FISE\_MEDICALE(COD\_FISA, ID\_PACIENT, ID\_DOCTOR, DIAGNOSTIC, DATA\_INTERNARII, DATA\_EXTERNARII, ID\_SECTIE) VALUES(566,3352,8697,'PLAGA INJUNGHIERE',TO\_DATE('04.01.2020','DD.MM.YYYY'),TO\_DATE('11.01.2020','DD.MM.YYYY'),3);

INSERT INTO FISE\_MEDICALE(COD\_FISA, ID\_PACIENT, ID\_DOCTOR, DIAGNOSTIC, DATA\_INTERNARII, DATA\_EXTERNARII, ID\_SECTIE) VALUES(663,4333,7774,'GRIPA',TO\_DATE('14.02.2020','DD.MM.YYYY'),TO\_DATE('17.02.2020','DD.MM.YYYY'),7);

INSERT INTO FISE\_MEDICALE(COD\_FISA, ID\_PACIENT, ID\_DOCTOR, DIAGNOSTIC, DATA\_INTERNARII, DATA\_EXTERNARII, ID\_SECTIE) VALUES(345,1225,9768,'ENTORSA',TO\_DATE('07.12.2020','DD.MM.YYYY'),TO\_DATE('07.12.2020','DD.MM.YYYY'),2);

INSERT INTO FISE\_MEDICALE(COD\_FISA, ID\_PACIENT, ID\_DOCTOR, DIAGNOSTIC, DATA\_INTERNARII, DATA\_EXTERNARII, ID\_SECTIE) VALUES(114, 2362,7869,'APENDICITA',TO\_DATE('05.12.2019','DD.MM.YYYY'),TO\_DATE('08.12.2019','DD.MM.YYYY'),3);

INSERT INTO FISE\_MEDICALE(COD\_FISA, ID\_PACIENT, ID\_DOCTOR, DIAGNOSTIC, DATA\_INTERNARII, DATA\_EXTERNARII, ID\_SECTIE) VALUES(156, 4429,7785,'HEPATITA',TO\_DATE('30.03.2019','DD.MM.YYYY'),TO\_DATE('13.04.2019','DD.MM.YYYY'),7);

**REȚETE**

INSERT INTO RETETE(COD\_RETETA, ID\_DOCTOR,COD\_FISA) VALUES (123, 7785, 156);

INSERT INTO RETETE(COD\_RETETA, ID\_DOCTOR,COD\_FISA) VALUES (132,7869,114);

INSERT INTO RETETE(COD\_RETETA, ID\_DOCTOR,COD\_FISA) VALUES (321,9768,345);

INSERT INTO RETETE(COD\_RETETA, ID\_DOCTOR,COD\_FISA) VALUES (312,7774,663);

INSERT INTO RETETE(COD\_RETETA, ID\_DOCTOR,COD\_FISA) VALUES (112,8697,556);

INSERT INTO RETETE(COD\_RETETA, ID\_DOCTOR,COD\_FISA) VALUES (113,6679,456);

INSERT INTO RETETE(COD\_RETETA, ID\_DOCTOR,COD\_FISA) VALUES (122,9876,432);

INSERT INTO RETETE(COD\_RETETA, ID\_DOCTOR,COD\_FISA) VALUES (332,9867,445);

INSERT INTO RETETE(COD\_RETETA, ID\_DOCTOR,COD\_FISA) VALUES (213,6679,435);

INSERT INTO RETETE(COD\_RETETA, ID\_DOCTOR,COD\_FISA) VALUES (223,7688,455);

**MEDICAMENTE**

INSERT INTO MEDICAMENTE(ID\_MEDICAMENT, DENUMIRE, ADMINISTRARE) VALUES(12000, 'SINUPRET', '1/ZI SEARA');

INSERT INTO MEDICAMENTE(ID\_MEDICAMENT, DENUMIRE, ADMINISTRARE) VALUES(12001,'ACUSTIVUM','2/ZI');

INSERT INTO MEDICAMENTE(ID\_MEDICAMENT, DENUMIRE, ADMINISTRARE) VALUES(12002,'ROACUTANE 100MG','1/ZI DUPA MASA');

INSERT INTO MEDICAMENTE(ID\_MEDICAMENT, DENUMIRE, ADMINISTRARE) VALUES(12003,'LAROCHE POSAY', 'APLICARE CREMA SEARA');

INSERT INTO MEDICAMENTE(ID\_MEDICAMENT, DENUMIRE, ADMINISTRARE) VALUES(12004,'ANTIINFLAMATOR','LA DURERE');

INSERT INTO MEDICAMENTE(ID\_MEDICAMENT, DENUMIRE, ADMINISTRARE) VALUES(12005,'KERIUM SAMPON','2 SAPT LA FIECARE SPALARE');

INSERT INTO MEDICAMENTE(ID\_MEDICAMENT, DENUMIRE, ADMINISTRARE) VALUES(12006, 'DIPROSALIC', 'PE SCALP SEARA');

INSERT INTO MEDICAMENTE(ID\_MEDICAMENT, DENUMIRE, ADMINISTRARE) VALUES(12007,'ANALGEZIC','2/ZI TIMP DE 3 ZILE');

INSERT INTO MEDICAMENTE(ID\_MEDICAMENT, DENUMIRE, ADMINISTRARE) VALUES(12008, 'MORFINA','2 FIOLE/ZI PENTRU 3 ZILE');

INSERT INTO MEDICAMENTE(ID\_MEDICAMENT, DENUMIRE, ADMINISTRARE) VALUES(12009, 'AUGUMENTIN', '2/ZI INAINTE DE MASA');

INSERT INTO MEDICAMENTE(ID\_MEDICAMENT, DENUMIRE, ADMINISTRARE) VALUES(12010,'DICLOFENAC','SEARA');

INSERT INTO MEDICAMENTE(ID\_MEDICAMENT, DENUMIRE, ADMINISTRARE) VALUES(12011, 'ANALGEZIC', '2/ZI DIMINEATA SI SEARA');

INSERT INTO MEDICAMENTE(ID\_MEDICAMENT, DENUMIRE, ADMINISTRARE) VALUES(12012, 'HEPATOPROTECT','10 ZILE, 2/ZI');

SELECT \* FROM MEDICAMENTE ORDER BY ID\_MEDICAMENT;

**TRATAMENTE**

INSERT INTO TRATAMENTE(COD\_TRATAMENT, ID\_MEDICAMENT, COD\_RETETA) VALUES(10,12000,223);

INSERT INTO TRATAMENTE(COD\_TRATAMENT, ID\_MEDICAMENT, COD\_RETETA) VALUES(11,12002,213);

INSERT INTO TRATAMENTE(COD\_TRATAMENT, ID\_MEDICAMENT, COD\_RETETA) VALUES(12,12003,213);

INSERT INTO TRATAMENTE(COD\_TRATAMENT, ID\_MEDICAMENT, COD\_RETETA) VALUES(13,12004,332);

INSERT INTO TRATAMENTE(COD\_TRATAMENT, ID\_MEDICAMENT, COD\_RETETA) VALUES(14,12005,112);

INSERT INTO TRATAMENTE(COD\_TRATAMENT, ID\_MEDICAMENT, COD\_RETETA) VALUES(15,12006,113);

INSERT INTO TRATAMENTE(COD\_TRATAMENT, ID\_MEDICAMENT, COD\_RETETA) VALUES(16,12007,122);

INSERT INTO TRATAMENTE(COD\_TRATAMENT, ID\_MEDICAMENT, COD\_RETETA) VALUES(17,12008,112);

INSERT INTO TRATAMENTE(COD\_TRATAMENT, ID\_MEDICAMENT, COD\_RETETA) VALUES(18,12009,312);

INSERT INTO TRATAMENTE(COD\_TRATAMENT, ID\_MEDICAMENT, COD\_RETETA) VALUES(19,12010,321);

INSERT INTO TRATAMENTE(COD\_TRATAMENT, ID\_MEDICAMENT, COD\_RETETA) VALUES(20,12011,132);

INSERT INTO TRATAMENTE(COD\_TRATAMENT, ID\_MEDICAMENT, COD\_RETETA) VALUES(21,12012,123);

INSERT INTO TRATAMENTE(COD\_TRATAMENT, ID\_MEDICAMENT, COD\_RETETA) VALUES(25,12001,223);

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Interogări:**

-- Să se mărească grupa de sânge la 20.

ALTER TABLE PACIENTI GRUPA SANGE VARCHAR2(20);

--Să se adauge o restricţie de tip check şi una de tip unique la coloana email în tabela pacienţi

ALTER TABLE PACIENTI

ADD CONSTRAINT CHK\_MAIL CHECK (EMAIL LIKE '%@%');

ALTER TABLE PACIENTI ADD CONSTRAINT uqMail UNIQUE (email);

-- Să se afişeze numele şi prenumele doctorilor din departamentul pediatrie. Să se folosească un alias.

SELECT NUME\_D ||' '||

PRENUME\_D AS "NUME DOCTORI"

FROM DOCTORI

WHERE ID\_SECTIE=1;

--Să se afişeze codurile şi denumirile secţiilor, ordonându-se după denumire.

SELECT ID\_SECTIE ||' '||

DENUMIRE AS "INFORMATII SECTIE"

FROM SECTII

ORDER BY DENUMIRE;

--Să se afişeze grupele de sânge eliminându-se duplicatele

SELECT DISTINCT GRUPA\_SANGE

FROM PACIENTI;

--Să se afişeze câte zile au fost internaţi pacienţii

SELECT p.nume\_p, p.prenume\_p, (DATA\_EXTERNARII-DATA\_INTERNARII)

AS "ZILE INTERNARE" from pacienti p join fise\_medicale f on p.id\_pacient=f.id\_pacient;

-- Să se afişeze numele şi vârsta tutror doctorilor

select nume\_d, prenume\_d, round((sysdate-DATA\_NASTERII)/365)|| 'years' as AGE from doctori;

--Să se afişeze câte zile au fost internaţi pacienţii din secţia ortopedie

  SELECT p.nume\_p, p.prenume\_p, (DATA\_EXTERNARII-DATA\_INTERNARII), f.id\_sectie

AS "ZILE INTERNARE" from pacienti p join fise\_medicale f on p.id\_pacient=f.id\_pacient

where f.id\_sectie='2';

--Să se afişeze pentru fiecare pacient numărul zilelor de spitalizare şi doctorul care l-a tratat.

select nume\_p, prenume\_p, (data\_externarii-data\_internarii) AS Zile\_de\_internare, nume\_d||' '||prenume\_d DOCTOR

from doctori m, pacienti p, fise\_medicale f

where m.id\_doctor=f.id\_doctor and p.id\_pacient=f.id\_pacient;

-- Să se afişeze pentru pacientul cu numele de familie BLACK diagnosticul, medicamentele prescrise şi modul de administrare

select nume\_p, prenume\_p, diagnostic, denumire, administrare

from pacienti p, fise\_medicale f, retete r, tratamente t, medicamente m

where nume\_p='BLACK' and p.id\_pacient=f.id\_pacient and f.cod\_fisa=r.cod\_fisa and r.cod\_reteta=t.cod\_reteta and m.id\_medicament=t.id\_medicament;



--Să se afişeze pentru fiecare medic numărul pacienţilor pe care i-a consultat

select d.id\_doctor,d.nume\_d, d.prenume\_d,count(f.id\_pacient) Numar\_pacienti

from fise\_medicale f, doctori d

where d.id\_doctor=f.id\_doctor

group by d.id\_doctor,d.nume\_d, d.prenume\_d

order by  Numar\_pacienti;

--Să se afişeze secţiile cu cel mai mare număr de doctori

select s.id\_sectie, s.denumire

from doctori d, sectii s

where d.id\_sectie=s.id\_sectie

group by s.id\_sectie,s.denumire

having count(\*)=(select max(count(\*))

                  from doctori group by id\_sectie);

-- Să se afişeze medicamentele ce conţin două vocale 'a' în nume

select \*

from medicamente

where lower(denumire) like '%a%a%';

--Să se creeze un clasament al pacienţilor în funcţie de numărul de zile de spitalizare. Dacă numărul este mai mic de 5, atunci pacientul va avea statusul de ‘Pacient in stare buna’, între 5-10 zile ‘Pacient în stare medie’ iar peste 10 ‘Pacient în stare gravă’. Să se ordoneze crescator pacienţii în funcţie de numărul zilelor de spitalizare.

select nume\_p||' '||prenume\_p as Pacient,data\_externarii-data\_internarii as Zile\_spitalizare,

case

when (data\_externarii-data\_internarii)<3 then 'Pacient in stare buna'

when (data\_externarii-data\_internarii) between 3 and 7 then 'Pacient in stare medie'

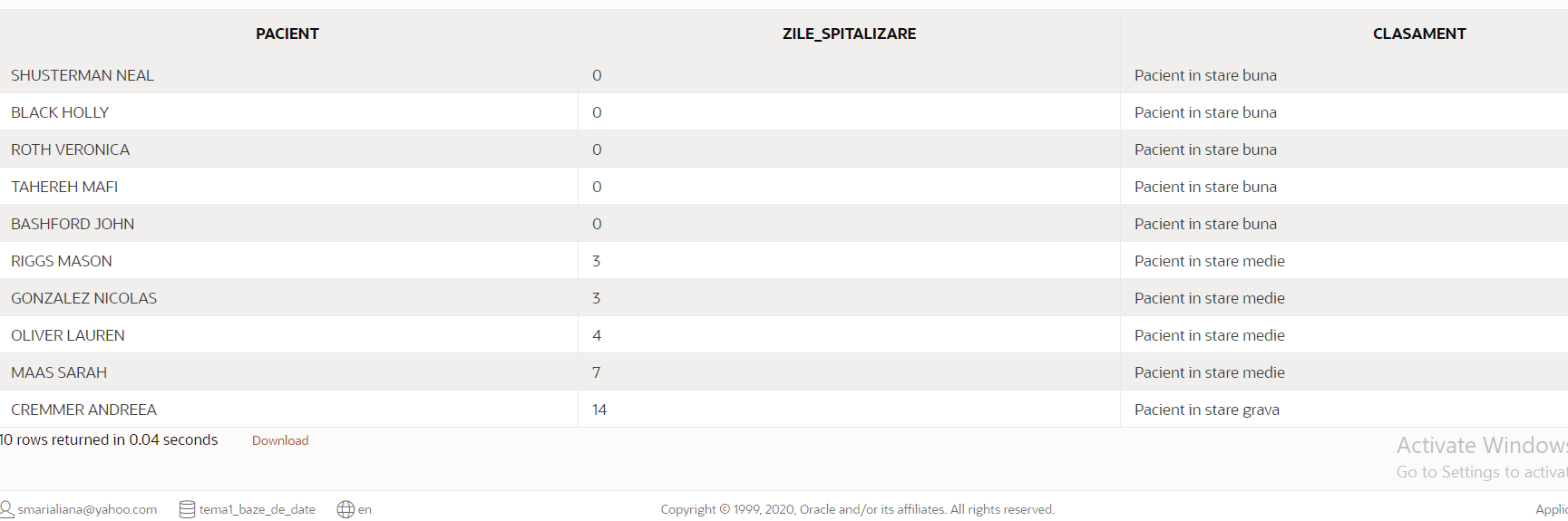
else 'Pacient in stare grava'

end clasament

from pacienti p, fise\_medicale f

where f.id\_pacient=p.id\_pacient

order by Zile\_spitalizare asc;



--Să se adauge o secţie de neonatologie

insert into sectii (id\_sectie, denumire) values (8,'NEONATOLOGIE');

--Să se afişeze pentru fiecare secţie număul de doctori(inclusiv secţiile fără doctori)

select s.id\_sectie, s.denumire, count(d.id\_doctor) Numar\_doctori

from doctori d, sectii s

where d.id\_sectie(+)=s.id\_sectie

group by s.id\_sectie, s.denumire

order by Numar\_doctori asc;

--Să se afişeze medicamentele cu o denumire mai mare de 10, iar lungimea intrucţiunilor de administrare mai mare de 10.

select \*

from medicamente

where length(DENUMIRE)>10 and length(ADMINISTRARE)>15;

--Să se afişeze data cand a părăsit spitalul, dar şi id-ul fişei sale, pacientul cu emailul THEREH.MAFI@GMAIL.COM.

select cod\_fisa, data\_internarii

from fise\_medicale

where id\_pacient=(select id\_pacient from pacienti where email like '%THEREH.MAFI@GMAIL.COM%');

--Să se afişeze numele secţiilor care au id-ul între 1 şi 8 fără cei care au id-ul 3 şi 5.

SELECT id\_sectie, denumire FROM sectii WHERE id\_sectie BETWEEN 1 AND 8

MINUS

SELECT id\_sectie, denumire FROM sectii WHERE id\_sectie IN (3, 5);

--Să se afişeze secţiile care nu au medici încă, dar şi secţiile care au 2 medici

select s.id\_sectie, s.DENUMIRE, count(d.id\_doctor) Numar\_doctori

from doctori d, sectii s

where d.id\_sectie(+)=s.id\_sectie

group by s.id\_sectie, s.DENUMIRE

having count(d.id\_doctor)=0

UNION

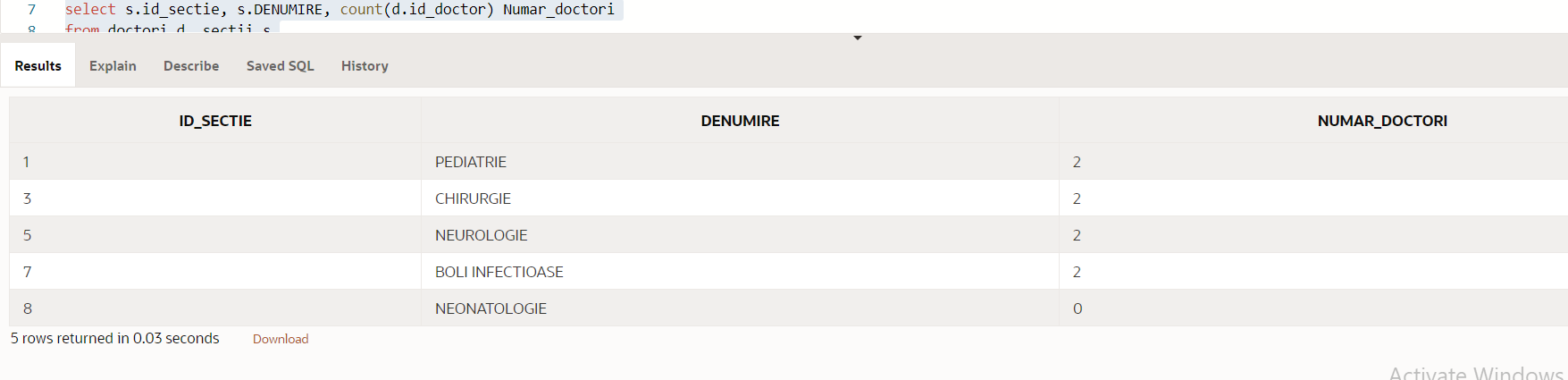
select s.id\_sectie, s.DENUMIRE, count(d.id\_doctor) Numar\_doctori

from doctori d, sectii s

where d.id\_sectie(+)=s.id\_sectie

group by s.id\_sectie, s.DENUMIRE

having count(d.id\_doctor)=2;



**--Să se afişeze numele doctorilor din secţia pediatrie**

**SELECT NUME\_D, PRENUME\_D, SPECIALIZARE FROM DOCTORI WHERE ID\_SECTIE=1;**



--Să se schimbe grupa de sânge în DONATOR UNIVERSAL pentru 0 şi PRIMITOR UNIVERSAL pentru AB4.

UPDATE PACIENTI

SET GRUPA\_SANGE=

(CASE

WHEN (GRUPA\_SANGE LIKE'0%') THEN 'DONATOR UNIVERSAL'

WHEN (GRUPA\_SANGE LIKE'AB4%') THEN 'PRIMITOR UNIVERSAL'

ELSE (GRUPA\_SANGE)

END

);

--Să se treacă toţi pacienţii sub 18 ani în secţia pediatrie

UPDATE (

    SELECT p.nume\_p, p.id\_pacient, p.DATA\_NASTERII, f.id\_sectie

    FROM pacienti p

    join fise\_medicale f on p.id\_pacient=f.id\_pacient

    WHERE(extract(year from sysdate)-extract(year from to\_date(DATA\_NASTERII))<18)

)

SET ID\_SECTIE='1';

--Să se modifice specializarea doctorilor sub 29 de ani în REZIDENT

UPDATE DOCTORI

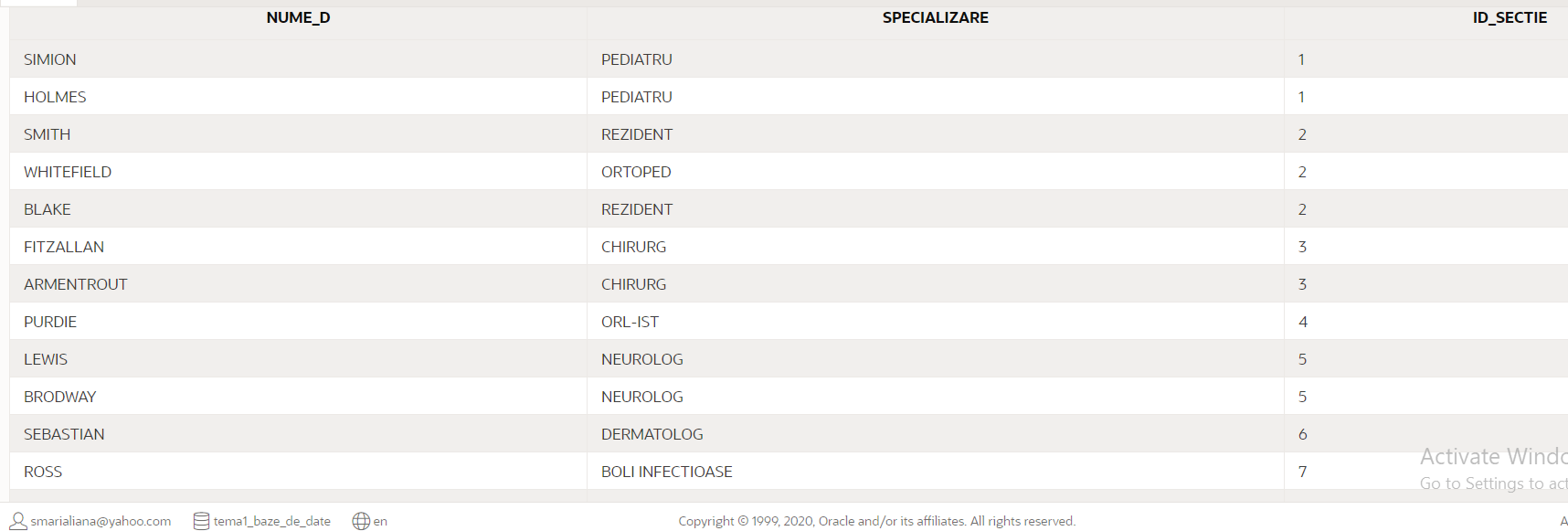
SET SPECIALIZARE = 'REZIDENT'

WHERE ( extract(year from sysdate)-extract(year from to\_date(DATA\_NASTERII))<29);

select nume\_d, prenume\_d, id\_sectie, specializare from doctori order by id\_sectie;

select nume\_d, specializare, id\_sectie from doctori

order by id\_sectie;



--Să se marească cu 10% salariul doctorilor din secţia neurologie

update doctori

set salariu= salariu\*1.1

where id\_sectie='5';

select salariu from doctori where id\_sectie='5';

--Să se mărească cu 25% salariul rezidenţilor din secţia ortopedie

update doctori

set salariu= 1.25\*salariu

where id\_sectie='2' and specializare ='REZIDENT';

SELECT \* FROM DOCTORI WHERE id\_sectie='2' AND specializare='REZIDENT';

--Să se actualizeze diagnosticul persoanei înjunghiate în STARE CRITICĂ

UPDATE FISE\_MEDICALE

SET DIAGNOSTIC='STARE CRITICA'

WHERE DIAGNOSTIC ='PLAGA INJUNGHIERE';

--Să se afiseze toti subordonatii doctorului cu id-ul ‘9867’

alter table doctori add ID\_SEF number(4);

alter table doctori add constraint sef\_doctori\_fk foreign key(id\_sef) references doctori(id\_doctor);

select id\_doctor,level, lpad('\*',level\*2-2,'\*')||nume\_d||' '||prenume\_d nume\_complet,sys\_connect\_by\_path(nume\_d,'/') cale

from doctori

connect by prior id\_doctor=id\_sef

start with id\_doctor=9867;

--Să se realizeze o tabelă virtuală cu fiecare pacient şi numărul de zile de spitalizare.

create or replace view Virtual\_table\_pacienti as

select nume\_p,data\_externarii-data\_internarii Zile\_spitalizare

from fise\_medicale f, pacienti p

where p.id\_pacient=f.id\_pacient;

desc Virtual\_table\_pacienti;

--Să se creeze un index pe tabela medicamente, pe coloana denumire şi să se şteargă apoi

Create index index\_medicamente

on medicamente(DENUMIRE);

 drop INDEX index\_medicamente;

--Să se creeze un sinonim pentru tabela tratamente şi unul pentru tabela medicamente

 CREATE SYNONYM tt FOR TRATAMENTE;

 CREATE SYNONYM med FOR MEDICAMENTE;

--Să se folosească sinonimele create pentru a afişa codul tratamentului, id-ul medicamentului şi denumirea acestuia

select m.id\_medicament, m.denumire, t.cod\_tratament from med m join tt t on m.id\_medicament=t.id\_medicament;

--Să se creeze o secvenţă pentru asigurarea unicitătii cheii primare din tabela Pacienţi.

CREATE SEQUENCE seq\_pacienti

start with 1234 increment by 5

maxvalue 5000 nocycle;

- Să se şteargă din tabela medicamente medicamentele ce conţin cuvântul MASA  în coloana administrare

DELETE FROM MEDICAMENTE WHERE ADMINISTRARE LIKE '%MASA';

--Să se şteargă toţi pacienţii care au stat mai puţin de o zi.

DELETE FROM FISE\_MEDICALE WHERE DATA\_INTERNARE=DATA\_EXTERNARE;

--Să se şteargă tabelele create

drop table PACIENTI;

drop table DOCTORI;

drop table SECTII;

drop table FISE\_MEDICALE;

drop table MEDICAMENTE;

drop table TRATAMENTE;

drop table RETETE;

A fost nevoie sa creez si tabela FUNCTII\_MEDICI:

**ALTER TABLE doctori**

**ADD ID\_FUNCTIE number(2);**

**create table FUNCTII\_MEDICI(**

**ID\_FUNCTIE NUMBER(2),**

**DENUMIRE VARCHAR2(10),**

**SALARIU\_MIN NUMBER(7),**

**SALARIU\_MAX NUMBER(7)**

**);**

**alter table FUNCTII\_MEDICI**

**add constraint ID\_FUNCTIE\_MED\_PK primary key (ID\_FUNCTIE);**

**alter table DOCTORI**

**add constraint DOC\_FUNCTIE\_FK foreign key (ID\_FUNCTIE)**

**references FUNCTII\_MEDICI (ID\_FUNCTIE);**

**INSERT INTO FUNCTII\_MEDICI (ID\_FUNCTIE, DENUMIRE, SALARIU\_MIN, SALARIU\_MAX) VALUES (1, 'REZIDENT', 3000, 4500);**

**INSERT INTO FUNCTII\_MEDICI(ID\_FUNCTIE, DENUMIRE, SALARIU\_MIN, SALARIU\_MAX) VALUES (2, 'SPECIALIST', 5000, 7000);**

**INSERT INTO FUNCTII\_MEDICI (ID\_FUNCTIE, DENUMIRE, SALARIU\_MIN, SALARIU\_MAX) VALUES (3, 'PRIMAR', 7500, 10000);**

**UPDATE DOCTORI SET ID\_FUNCTIE = 3 WHERE ID\_DOCTOR= 9877;**

**UPDATE DOCTORI SET ID\_FUNCTIE = 3 WHERE ID\_DOCTOR=94;**

**UPDATE DOCTORI SET ID\_FUNCTIE = 3 WHERE ID\_DOCTOR=93;**

**UPDATE DOCTORI SET ID\_FUNCTIE = 3 WHERE ID\_DOCTOR=95;**

**UPDATE DOCTORI SET ID\_FUNCTIE = 3 WHERE ID\_DOCTOR=8697;**

**UPDATE DOCTORI SET ID\_FUNCTIE = 3 WHERE ID\_DOCTOR=8976;**

**UPDATE DOCTORI SET ID\_FUNCTIE = 3 WHERE ID\_DOCTOR=91;**

**UPDATE DOCTORI SET ID\_FUNCTIE = 3 WHERE ID\_DOCTOR=6789;**

**UPDATE DOCTORI SET ID\_FUNCTIE = 3 WHERE ID\_DOCTOR=7869;**

**UPDATE DOCTORI SET ID\_FUNCTIE = 3 WHERE ID\_DOCTOR=8769;**

**UPDATE DOCTORI SET ID\_FUNCTIE = 3 WHERE ID\_DOCTOR=7785;**

**UPDATE DOCTORI SET ID\_FUNCTIE = 3 WHERE ID\_DOCTOR=96;**

**UPDATE DOCTORI SET ID\_FUNCTIE = 2 WHERE ID\_DOCTOR=9876;**

**UPDATE DOCTORI SET ID\_FUNCTIE = 2 WHERE ID\_DOCTOR=90;**

**UPDATE DOCTORI SET ID\_FUNCTIE = 2 WHERE ID\_DOCTOR=92;**

**UPDATE DOCTORI SET ID\_FUNCTIE = 2 WHERE ID\_DOCTOR=97;**

**UPDATE DOCTORI SET ID\_FUNCTIE = 2 WHERE ID\_DOCTOR=100;**

**UPDATE DOCTORI SET ID\_FUNCTIE = 2 WHERE ID\_DOCTOR=7688;**

**UPDATE DOCTORI SET ID\_FUNCTIE = 2 WHERE ID\_DOCTOR=6679;**

**UPDATE DOCTORI SET ID\_FUNCTIE = 1 WHERE ID\_DOCTOR=9768;**

**UPDATE DOCTORI SET ID\_FUNCTIE = 1 WHERE ID\_DOCTOR=9867;**

**UPDATE DOCTORI SET ID\_FUNCTIE = 1 WHERE ID\_DOCTOR=7774;**