Facultatea de Matematica si Informatica,
Universitatea din Bucuresti

# PROIECT BAZE DE DATE

## Gestiunea unui complex de clinici veterinare

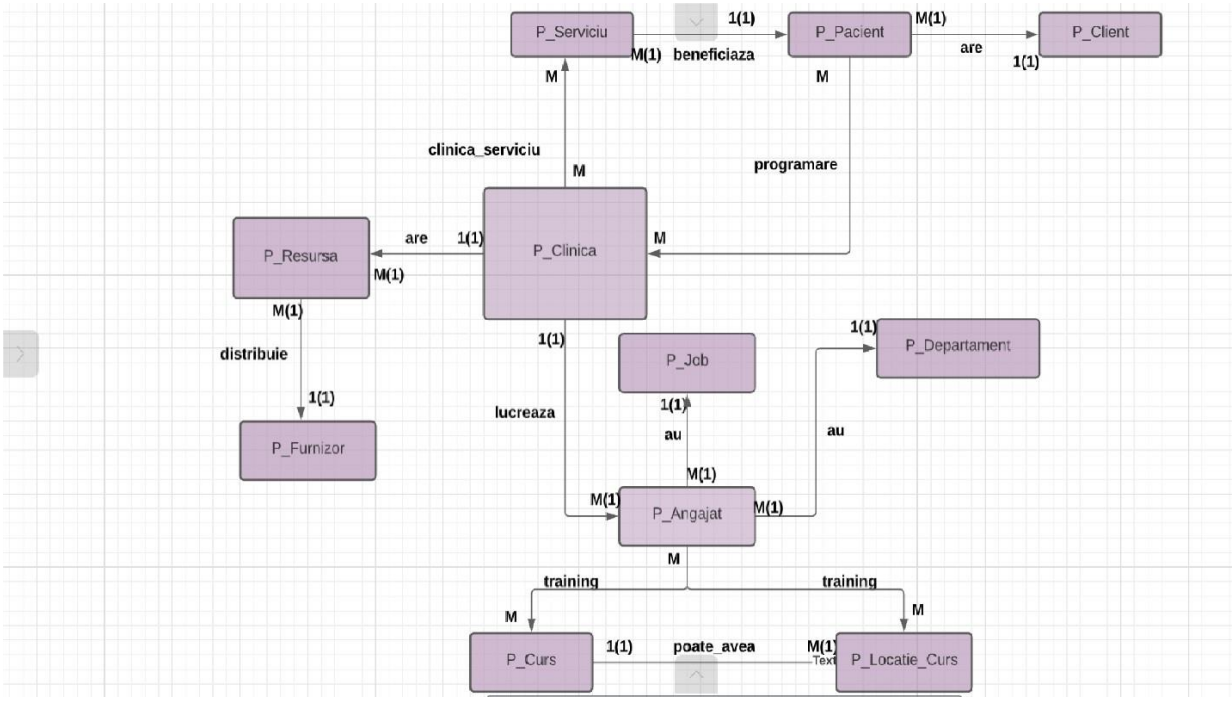
Student

Brandiburu Andreea-Nicoleta

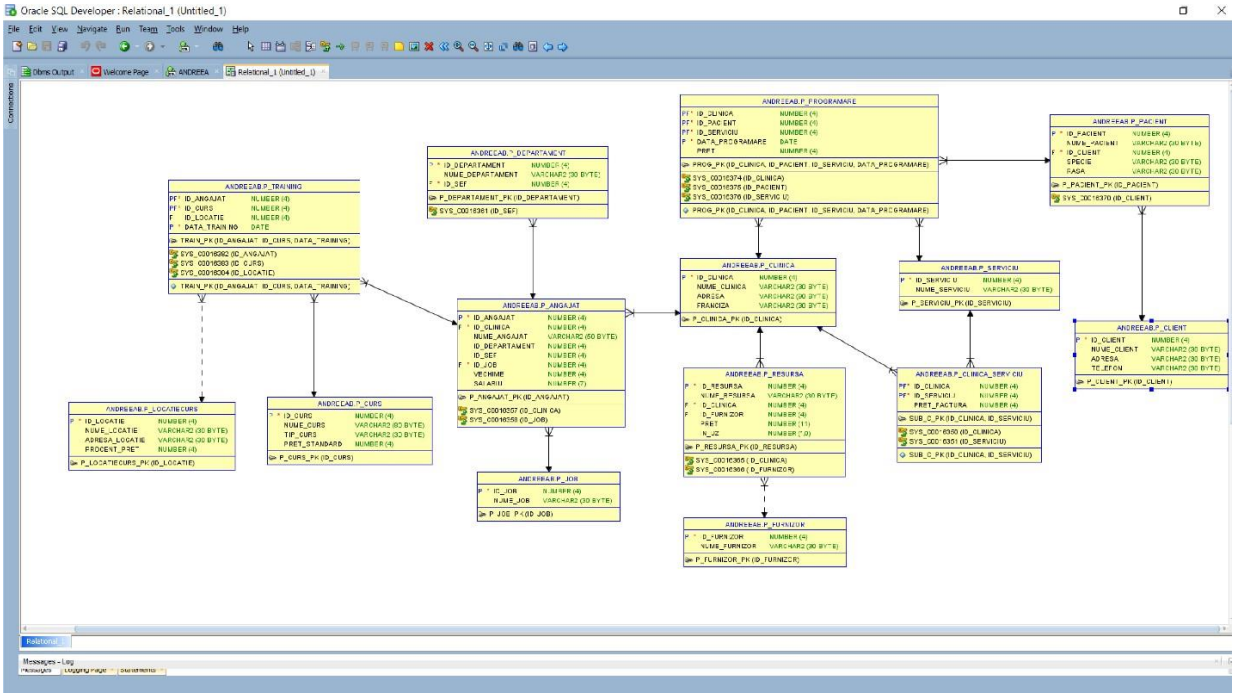
## 1. Utilitatea bazei de date

Sa se implementeze un sistem de gestiune a mai multor clinici medical veterinare. Vrem sa tinem evidenta angajatilor si pozitiilor ocupate de acestia in diferite departmente precum si a diverselor cursuri de formare pe care acestia le urmeaza, a resurselor disponibile in fiecare clinica si a furnizorilor acestora, precum si o lista de clienti si pacienti (animale de companie) si toate programarile facute de acesti clienti. La toate aceste lucruri dorim sa tinem evidenta financiara (salariu/pret).

## 2. Diagrama entitate-relație (ERD).



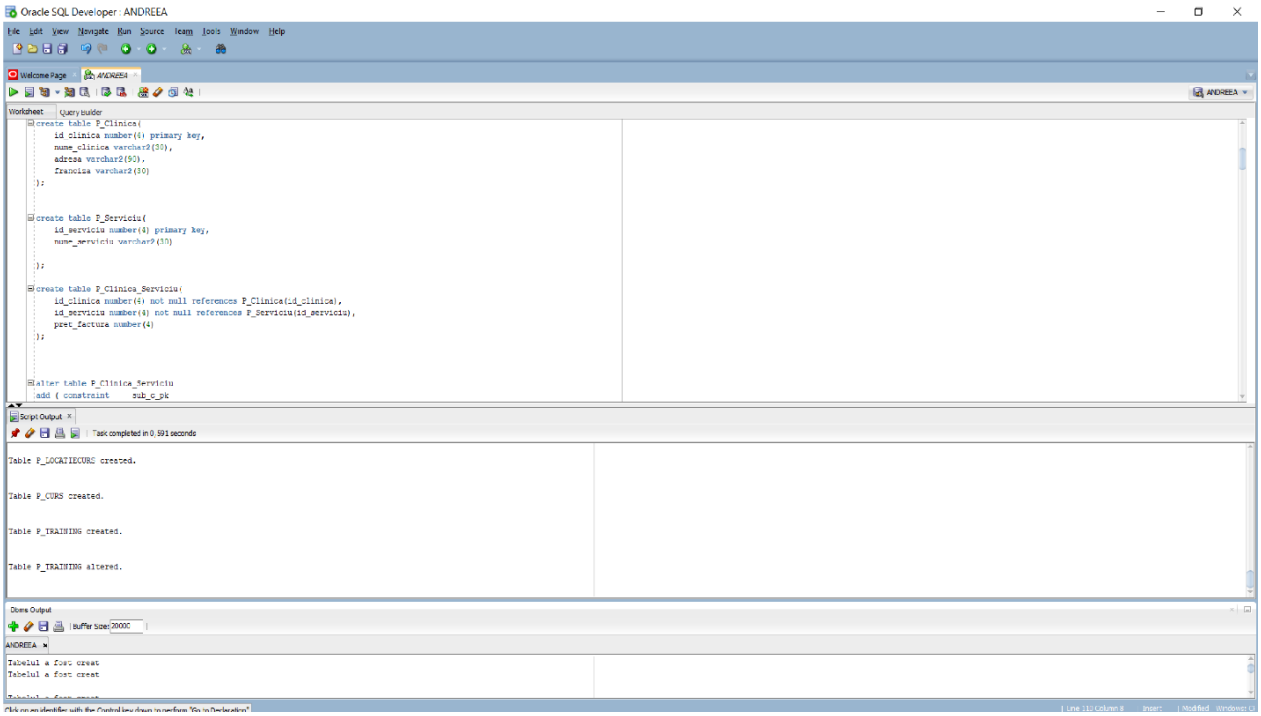
# 3. Diagrama conceptuala



4. **Implementați în Oracle diagrama conceptuală realizată**: definiți toate tabelele, implementând toate constrângerile de integritate necesare (chei primare, cheile externe etc).

In continuare am definit tabelele din schema, am implementat constrangerile de cheie primara si cheie externa pentru tabele.

**Screenshot  create-uri**




**Cod SQL tabele create**

```
drop table P_Training;
drop table P_Curs;
drop table P_LocatieCurs;
drop table P_Programare;
```

```
drop table P_Pacient;
drop table P_Client;
drop table P_Resursa;
drop table P_Furnizor;
drop table P_Departament;
drop table  P_Angajat;
drop table P_Job;
drop table P_Clinica_Serviciu;
drop table P_Serviciu;
drop table P_Clinica;

drop SEQUENCE SEQ_DEPT ;


create table P_Clinica(
        id_clinica number(4) primary key,
        nume_clinica  varchar2(30),
        adresa varchar2(90),
        franciza varchar2(30)
);


create table P_Serviciu(
        id_serviciu number(4) primary key,
        nume_serviciu varchar2(30)

);

create table P_Clinica_Serviciu(
   id_clinica number(4) not null references P_Clinica(id_clinica),
   id_serviciu number(4) not null references P_Serviciu(id_serviciu),
   pret_factura number(4)
);
```

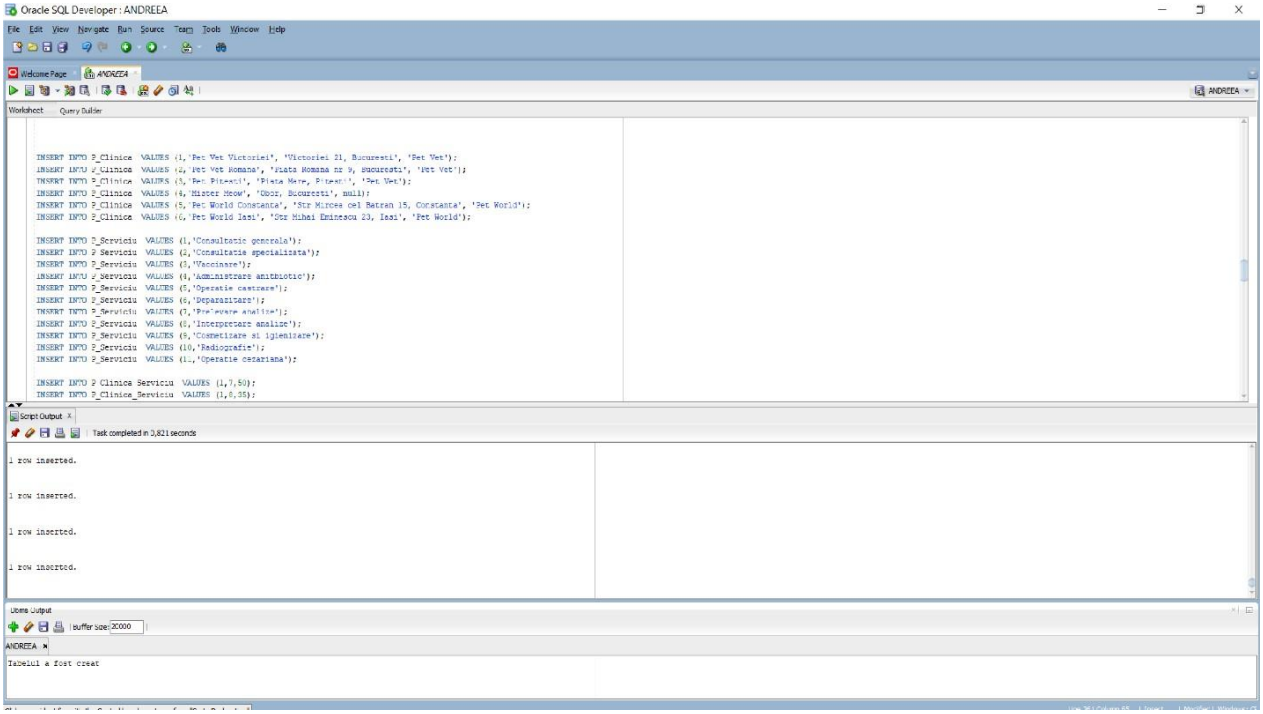
```
alter table P_Clinica_Serviciu
add ( constraint     sub_c_pk
                    primary key (id_clinica, id_serviciu)
   ) ;

create table P_Job(
        id_job number(4) primary key,
        nume_job varchar2(30)
 );

create table P_Angajat(
        id_angajat number(4) primary key,
          id_clinica number(4) not null references P_Clinica(id_clinica),
        nume_angajat varchar2(50),
        id_departament number(4),
        id_sef number(4),
        id_job number(4) not null references P_Job(id_job),
        vechime number(4),
        salariu number(7)
);

create table P_Departament(
        id_departament number(4) primary key,
        nume_departament varchar2(30),
          id_sef number(4) not null references P_Angajat(id_angajat)
);

create table P_Furnizor(
        id_furnizor number(4) primary key,
        nume_furnizor varchar2(30)
);

create table P_Resursa(
        id_resursa number(4) primary key,
        nume_resursa varchar2(30),
        id_clinica number(4) not null references P_Clinica(id_clinica),
        id_furnizor number(4) references P_Furnizor(id_furnizor),
```

```
        pret number(11),
        in_uz int
);

create table P_Client(
        id_client number(4) primary key,
        nume_client  varchar2(30),
        adresa varchar2(30),
        telefon varchar2(30)
);




create table P_Pacient(
        id_pacient number(4) primary key,
        nume_pacient varchar2(30),
        id_client number(4) not null references P_Client(id_client),
        specie varchar2(30),
        rasa varchar2(30)
);

create table P_Programare(
   id_clinica number(4) not null references P_Clinica(id_clinica),
        id_pacient number(4) not null references P_Pacient(id_pacient),
        id_serviciu number(4) not null references P_Serviciu(id_serviciu),
        data_programare date,
        pret number(4)
);

alter table P_Programare
add ( constraint     prog_pk
                primary key (id_clinica,id_pacient,id_serviciu,data_programare)
   ) ;

create table P_LocatieCurs(
        id_locatie number(4) primary key,
```

```
        nume_locatie varchar2(30),
        adresa_locatie varchar2(30),
        procent_pret number(4)
);

create table P_Curs(
        id_curs number(4) primary key,
        nume_curs varchar2(80),
        tip_curs varchar2(30),
        pret_standard number(4)
);

create table P_Training(
        id_angajat number(4) not null references P_Angajat(id_angajat),
        id_curs number(4) not null references P_Curs(id_curs),
        id_locatie number(4) references P_LocatieCurs(id_locatie),
   data_training date

);


alter table P_Training
add ( constraint     train_pk
                primary key (id_angajat,id_curs,data_training)

   ) ;
```

## 5. Adaugarea de informatii coerente in tabelele create

### Screenshot insert-uri




### Cod SQL Insert-uri

INSERT INTO P_Clinica VALUES (1,'Pet Vet Victoriei', 'Victoriei 21, Bucuresti', 'Pet Vet');
INSERT INTO P_Clinica VALUES (2,'Pet Vet Romana', 'Piata Romana nr 9, Bucuresti', 'Pet Vet');
INSERT INTO P_Clinica VALUES (3,'Pet Pitesti', 'Piata Mare, Pitesti', 'Pet Vet');
INSERT INTO P_Clinica VALUES (4,'Mister Meow', 'Obor, Bucuresti', null);
INSERT INTO P_Clinica VALUES (5,'Pet World Constanta', 'Str Mircea cel Batran 15, Constanta', 'Pet World');

```
INSERT INTO P_Clinica  VALUES (6,'Pet World Iasi', 'Str Mihai Eminescu 23,
Iasi', 'Pet World');

INSERT INTO P_Serviciu VALUES (1,'Consultatie generala');
INSERT INTO P_Serviciu  VALUES (2,'Consultatie specializata');
INSERT INTO P_Serviciu  VALUES (3,'Vaccinare');
INSERT INTO P_Serviciu  VALUES (4,'Administrare anitbiotic');
INSERT INTO P_Serviciu VALUES (5,'Operatie castrare');
INSERT INTO P_Serviciu  VALUES (6,'Deparazitare');
INSERT INTO P_Serviciu VALUES (7,'Prelevare analize');
INSERT INTO P_Serviciu VALUES (8,'Interpretare analize');
INSERT INTO P_Serviciu  VALUES (9,'Cosmetizare si igienizare');
INSERT INTO P_Serviciu  VALUES (10,'Radiografie');
INSERT INTO P_Serviciu  VALUES (11,'Operatie cezariana');

INSERT INTO P_Clinica_Serviciu VALUES (1,7,50);
INSERT INTO P_Clinica_Serviciu VALUES (1,8,35);
INSERT INTO P_Clinica_Serviciu  VALUES (1,10,20);
INSERT INTO P_Clinica_Serviciu VALUES (4,6,70);
INSERT INTO P_Clinica_Serviciu VALUES (3,7,50);
INSERT INTO P_Clinica_Serviciu VALUES (6,3,50);
INSERT INTO P_Clinica_Serviciu  VALUES (2,5,150);
INSERT INTO P_Clinica_Serviciu VALUES (4,9,90);
INSERT INTO P_Clinica_Serviciu  VALUES (2,2,100);
INSERT INTO P_Clinica_Serviciu  VALUES (2,4,70);

INSERT INTO P_Job VALUES (1,'Asistent');
INSERT INTO P_Job VALUES (2,'Medic');
INSERT INTO P_Job  VALUES (3,'Manager');
INSERT INTO P_Job VALUES (4,'Electrician');
INSERT INTO P_Job  VALUES (5,'Femeie de servici');
INSERT INTO P_Job  VALUES (6,'Receptioner');

INSERT INTO P_Angajat  VALUES (1,1,'Ion Marin',1,2,1,7,5000);
INSERT INTO P_Angajat  VALUES (2,1,'Codruta Ghenea',1,4,2,15,10000);
INSERT INTO P_Angajat  VALUES (3,1,'Maria David',1,2,1,2,3500);
INSERT INTO P_Angajat  VALUES (4,1,'Andreea Boghici',2,4,3,7,15000);
```

```
INSERT INTO P_Angajat  VALUES (5,1,'Dan Marius',2,4,4,7,4500);
INSERT INTO P_Angajat  VALUES (6,1,'Margareta Burghiu',2,4,5,12,2800);
INSERT INTO P_Angajat VALUES (7,1,'Mariana Despescu',1,2,2,6,7500);
INSERT INTO P_Angajat  VALUES (8,1,'Ioana Maftei',1,2,1,3,4000);
INSERT INTO P_Angajat  VALUES (9,1,'Martin Dobrescu',1,2,1,1,3000);
INSERT INTO P_Angajat VALUES (10,1,'Ilinca Iancu',1,2,2,4,5500);
INSERT INTO P_Angajat VALUES (11,1,'Bogdan Vulpe',1,2,1,2,3500);
INSERT INTO P_Angajat VALUES (12,1,'Florin Ducu',1,2,1,3,4000);
INSERT INTO P_Angajat VALUES (13,1,'Alina Stoia',3,1,3,4,5500);
INSERT INTO P_Angajat VALUES (14,1,'Ana Zare',4,3,3,4,4500);
INSERT INTO P_Angajat  VALUES (15,1,'Mihnea Altoiu',5,5,3,8,7600);


INSERT INTO P_Departament  VALUES (1,'Medical',2);
INSERT INTO P_Departament  VALUES (2,'Admin',4);


CREATE SEQUENCE SEQ_DEPT INCREMENT by 1 START WITH 3
MAXVALUE 99 NOCYCLE; --se insereaza date utilizand o secventa
INSERT INTO P_Departament VALUES (SEQ_DEPT.NEXTVAL, 'IT', 4);
INSERT INTO P_Departament VALUES (SEQ_DEPT.NEXTVAL, 'HR', 4);
INSERT INTO P_Departament VALUES (SEQ_DEPT.NEXTVAL, 'Vanzari', 4);

INSERT INTO P_Furnizor VALUES (1,'Pfizer');
INSERT INTO P_Furnizor  VALUES (2,'Moderna');
INSERT INTO P_Furnizor VALUES (3,'Antibiotice Iasi');
INSERT INTO P_Furnizor VALUES (4,'Mob Expert');
INSERT INTO P_Furnizor  VALUES (5,'Celuloza Braila');
INSERT INTO P_Furnizor  VALUES (6,'Medi Base');

INSERT INTO P_Resursa VALUES (1,'vaccin pisici',1,1,100,0);
INSERT INTO P_Resursa VALUES (2,'manusi sterilizate',1,6,5,0);
INSERT INTO P_Resursa  VALUES (3,'masa de operatie',1,4,2000,1);
INSERT INTO P_Resursa VALUES (4,'bisturiu',1,6,25,0);
INSERT INTO P_Resursa VALUES (5,'antibiotic caini',1,3,35,0);
INSERT INTO P_Resursa  VALUES (6,'pat spitalizare',1,4,3000,1);
INSERT INTO P_Resursa  VALUES (7,'servetel sterilizat',1,5,4,0);
```

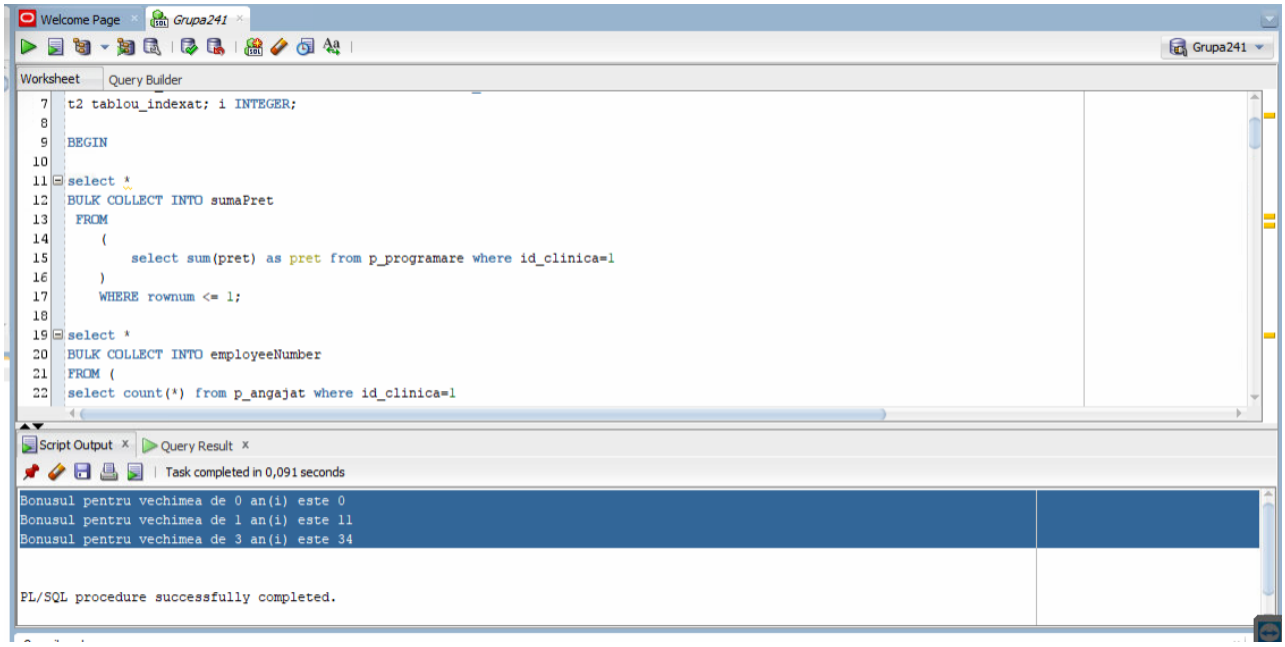
```
INSERT INTO P_Resursa  VALUES (8,'gel dezinfectant',1,6,9,0);
INSERT INTO P_Resursa VALUES (9,'seringa',1,6,15,0);
INSERT INTO P_Resursa  VALUES (10,'branula',1,6,7,1);
INSERT INTO P_Resursa  VALUES (11,'medicament anti paraziti',1,2,85,0);
INSERT INTO P_Resursa  VALUES (12,'scaner CT',1,6,2000000,0);

INSERT INTO P_Client  VALUES (1,'Maria Georgescu','Basarabiei 21,
Bucuresti','0756456653');
INSERT INTO P_Client  VALUES (2,'Vlad Petec','Armeneasca 12,
Bucuresti','0756899653');
INSERT INTO P_Client  VALUES (3,'Ignat Yaki','Bastiliei 36,
Bucuresti','0756489678');
INSERT INTO P_Client  VALUES (4,'Barbara Magheru','Cotroceni 110,
Bucuresti','0756489654');
INSERT INTO P_Client  VALUES (5,'Geanina Moise','Plevnei 211,
Bucuresti','0756488723');
INSERT INTO P_Client  VALUES (6,'Matei Dan','Lujerului 24,
Bucuresti','0758769653');

INSERT INTO P_Pacient VALUES (1,'Jacky',1,'canina','bulldog');
INSERT INTO P_Pacient VALUES (2,'Kitty',1,'felina','birmaneza');
INSERT INTO P_Pacient VALUES (3,'Finny',2,'paun','striat');
INSERT INTO P_Pacient  VALUES (4,'Scott',3,'canina','ciobanesc');
INSERT INTO P_Pacient  VALUES (5,'Geta',4,'felina','siameza');
INSERT INTO P_Pacient  VALUES (6,'Rambo',5,'cameleon','ecuadorian');
INSERT INTO P_Pacient VALUES (7,'JayJay',6,'broasca','testoasa');
INSERT INTO P_Pacient  VALUES (8,'Snakey',6,'sarpe','vipera');
INSERT INTO P_Pacient VALUES (9,'Fifi',1,'papagal','brazilian');
INSERT INTO P_Pacient VALUES (10,'Mark',4,'rozatoare','hamster');
INSERT INTO P_Pacient  VALUES (11,'Mark',2,'canina','corgi');

INSERT INTO P_Programare VALUES (1,1,1,'06-06-21',50);
INSERT INTO P_Programare VALUES (1,2,2,'07-07-21',90);
INSERT INTO P_Programare VALUES (1,3,2,'08-06-21',90);
INSERT INTO P_Programare  VALUES (1,4,3,'09-06-21',150);
INSERT INTO P_Programare  VALUES (1,4,10,'16-06-21',90);
INSERT INTO P_Programare  VALUES (1,6,9,'06-07-21',100);
```

```
INSERT INTO P_Programare VALUES (1,7,10,'07-07-21',90);
INSERT INTO P_Programare  VALUES (1,8,11,'08-07-21',900);
INSERT INTO P_Programare VALUES (1,9,1,'14-07-21',50);
INSERT INTO P_Programare  VALUES (1,10,2,'26-07-21',90);

INSERT INTO P_LocatieCurs  VALUES (1,'Vet Prestige
Amsterdam','Amsterdam, Zuffe 31', 200);
INSERT INTO P_LocatieCurs  VALUES (2,'London Veterinary
University','Londra, Buckingham Road 31', 250);
INSERT INTO P_LocatieCurs  VALUES (3,'Online Vet Academy','online', 75);
INSERT INTO P_LocatieCurs VALUES (4,'Sofia University','Sofia', 85);
INSERT INTO P_LocatieCurs VALUES (5,'Berliner Akamedie','Berlin,
Postadammer Platz 78', 175);

INSERT INTO P_Curs VALUES (1,'Operatia de castrare la feline','fizic', 800);
INSERT INTO P_Curs  VALUES (2,'Probleme specifice la canine','fizic', 1100);
INSERT INTO P_Curs VALUES (3,'Simptomatologia pasarilor in
captivitate','fizic', 850);
INSERT INTO P_Curs VALUES (4,'Anestezia generala veterinara','fizic', 750);
INSERT INTO P_Curs VALUES (5,'Probleme frecvente la speciile de rozatoare
domestice','online', 200);

INSERT INTO P_Training  VALUES (11,4,null,'08-07-21');
INSERT INTO P_Training VALUES (11,5,4,'09-07-21');
INSERT INTO P_Training VALUES (2,3,1,'23-07-21');
INSERT INTO P_Training VALUES (7,1,5,'30-07-21');
INSERT INTO P_Training VALUES (11,2,2,'14-07-21');
INSERT INTO P_Training  VALUES (10,4,null,'08-08-21');
INSERT INTO P_Training  VALUES (10,1,null,'09-08-21');
INSERT INTO P_Training VALUES (9,5,null,'18-09-21');
INSERT INTO P_Training VALUES (9,2,4,'28-09-21');
INSERT INTO P_Training  VALUES (8,5,null,'01-09-21');

commit;
```

6. Angajatii din departamentul medical vor primi un bonus de vechime(10% pentru 1 an vechime, 20% pentru minim 3 ani vechime si 40% pentru minim 5 ani de vechime). Procentul se aplica la media pe numarul de angajati totali a sumei incasate de clinica (prin pretul serviciilor programate). Sa se afiseze cat primeste fiecare angajat in functie de anii de vechime.

```
CREATE OR REPLACE PROCEDURE calcul_bonus IS
TYPE vector IS VARRAY(2) OF NUMBER;
sumaPret vector:= vector();
employeeNumber vector:=vector();
-- Partea de tablou indexat
TYPE tablou_indexat IS TABLE OF NUMBER INDEX BY PLS_INTEGER;
t2 tablou_indexat; i INTEGER;

BEGIN

select *
BULK COLLECT INTO sumaPret
 FROM
   (
       select sum(pret) as pret from p_programare where id_clinica=1
   )
   WHERE rownum <= 1;

select *
BULK COLLECT INTO employeeNumber
FROM (
select count(*) from p_angajat where id_clinica=1
) WHERE rownum <=1;

t2(0) := 0; -- la vechimea 0 bonusul e 0;
t2(1) := round(sumaPret(1)/employeeNumber(1) * 0.10);  -- vechimea 1 , bonusul e
10% t2(2) := round(t(1)/t(2) * 0.10);    -- vechimea 2 , bonusul e 10%

t2(3) := round(sumaPret(1)/employeeNumber(1) * 0.30);  -- vechimea 3 , bonusul e
30% t2(4) := round(t(1)/t(2) * 0.30); -- vechimea 4 , bonusul e 30% t2(5) :=
round(t(1)/t(2) * 0.40); -- vechimea 5 , bonusul e 40%
```

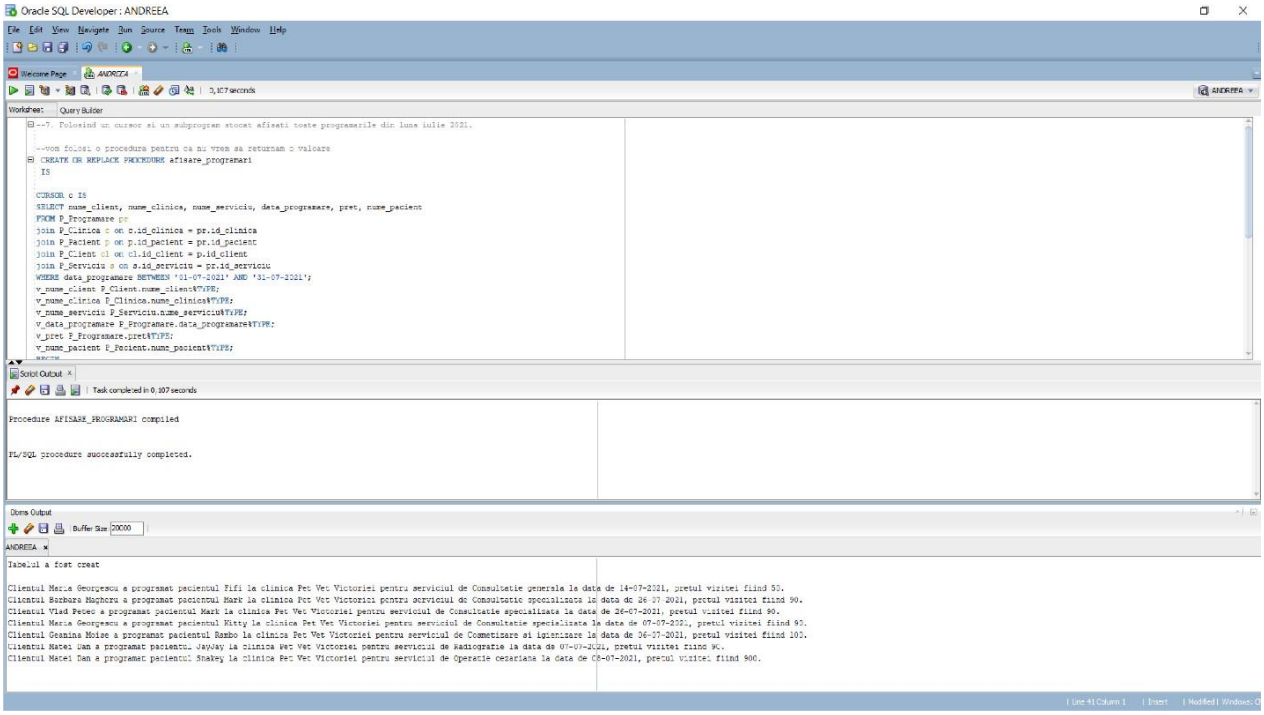
```
--afisare elemente tablou
i := t2.FIRST;
WHILE i <= 5 LOOP
DBMS_OUTPUT.PUT_LINE('Bonusul pentru vechimea de '|| i ||' an(i) este ' || t2(i) );
i := t2.NEXT(i); END LOOP;
t2.delete;


END calcul_bonus;
/

BEGIN
calcul_bonus(); END;
/
```



**7.** Folosind un cursor si un subprogram stocat afisati toate programarile din luna iulie 2021.

```
--vom folosi o procedura pentru ca nu vrem sa returnam o valoare
 CREATE OR REPLACE PROCEDURE afisare_programari
 IS

CURSOR c IS
SELECT nume_client, nume_clinica, nume_serviciu, data_programare, pret,
nume_pacient
FROM P_Programare pr
join P_Clinica c on c.id_clinica = pr.id_clinica
join P_Pacient p on p.id_pacient = pr.id_pacient
join P_Client cl on cl.id_client = p.id_client
join P_Serviciu s on s.id_serviciu = pr.id_serviciu
WHERE data_programare BETWEEN '01-07-2021' AND '31-07-2021';
v_nume_client P_Client.nume_client%TYPE;
```

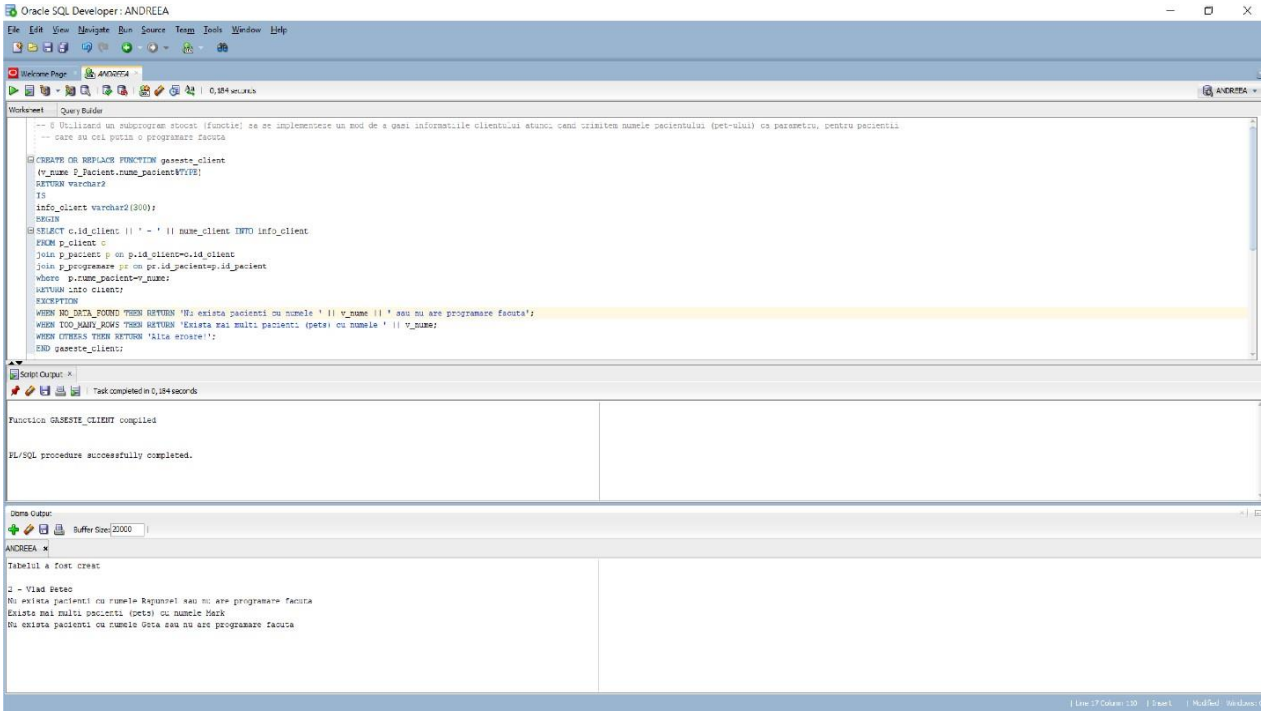
```
v_nume_clinica P_Clinica.nume_clinica%TYPE;
v_nume_serviciu P_Serviciu.nume_serviciu%TYPE;
v_data_programare P_Programare.data_programare%TYPE;
v_pret P_Programare.pret%TYPE;
v_nume_pacient P_Pacient.nume_pacient%TYPE;
BEGIN
BEGIN
OPEN c;
LOOP
FETCH c INTO v_nume_client, v_nume_clinica, v_nume_serviciu,
v_data_programare, v_pret, v_nume_pacient;
EXIT WHEN c%NOTFOUND;
DBMS_OUTPUT.PUT_LINE('Clientul ' || v_nume_client || ' a programat pacientul
' || v_nume_pacient || ' la clinica ' || v_nume_clinica || ' pentru serviciul de ' ||
v_nume_serviciu
|| ' la data de ' || v_data_programare || ', pretul vizitei fiind ' || v_pret || '.');
END LOOP;
CLOSE c;
END;

END afisare_programari;
/
BEGIN
afisare_programari();
END;
/
```




**8.** Utilizand un subprogram stocat (functie) sa se implementeze un mod de a gasi informatiile clientului atunci cand trimitem numele pacientului (pet-ului) ca parametru, pentru pacientii care au cel putin o programare facuta.

```
CREATE OR REPLACE FUNCTION gaseste_client
(v_nume P_Pacient.nume_pacient%TYPE)
RETURN varchar2
IS
info_client varchar2(300);
BEGIN
SELECT c.id_client || ' - ' || nume_client INTO info_client
FROM p_client c
join p_pacient p on p.id_client=c.id_client
join p_programare pr on pr.id_pacient=p.id_pacient
where p.nume_pacient=v_nume;
```

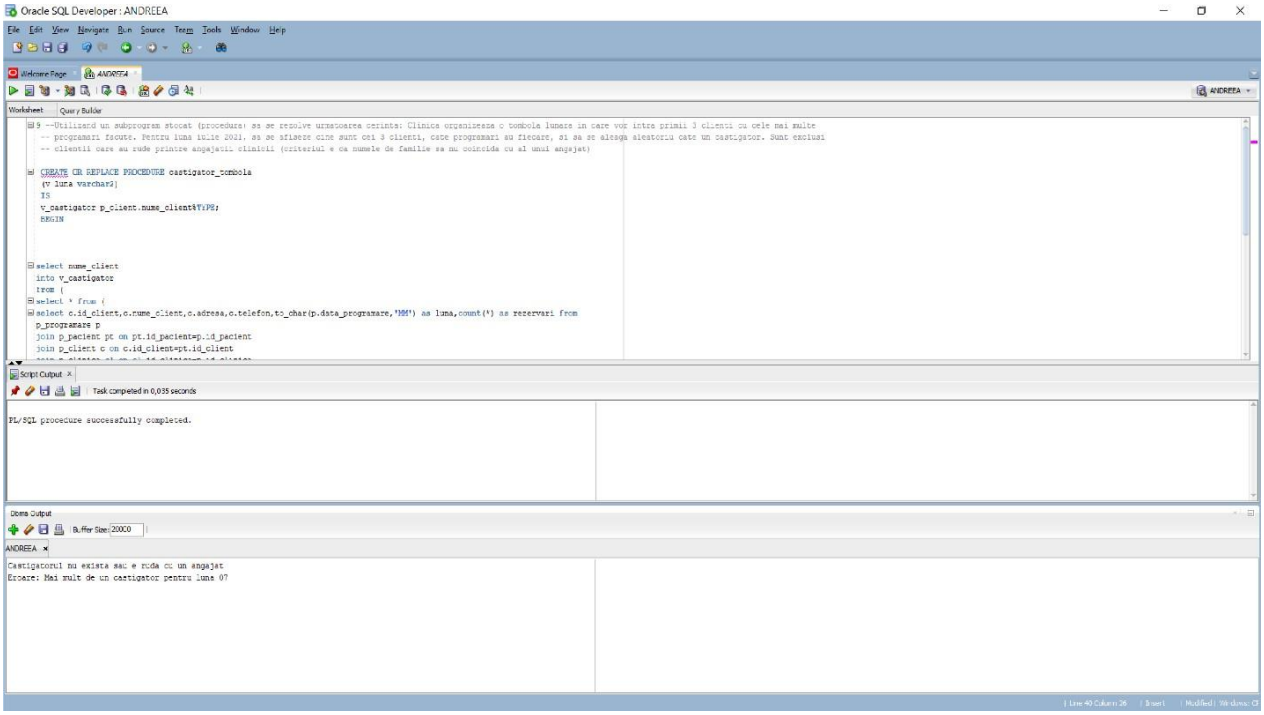
```
RETURN info_client;
EXCEPTION
WHEN NO_DATA_FOUND THEN RETURN 'Nu exista pacienti cu numele ' ||
v_nume || ' sau nu are programare facuta';
WHEN TOO_MANY_ROWS THEN RETURN 'Exista mai multi pacienti (pets)
cu numele ' || v_nume;
WHEN OTHERS THEN RETURN 'Alta eroare!';
END gaseste_client;

/
DECLARE
v_info_client varchar2(300);
BEGIN
v_info_client := gaseste_client('Finny');
DBMS_OUTPUT.PUT_LINE(v_info_client);
v_info_client := gaseste_client('Rapunzel');
DBMS_OUTPUT.PUT_LINE(v_info_client);
v_info_client := gaseste_client('Mark');
DBMS_OUTPUT.PUT_LINE(v_info_client);
v_info_client := gaseste_client('Geta');
DBMS_OUTPUT.PUT_LINE(v_info_client);

END;
/
```




**9.** Utilizand un subprogram stocat (procedura) sa se rezolve urmatoarea cerinta: Clinica organizeaza o tombola lunara in care vor intra primii 3 clienti cu cele mai multe programari facute. Pentru luna iulie 2021, sa se afiseze cine sunt cei 3 clienti, cate programari au fiecare, si sa se aleaga aleatoriu cate un castigator. Sunt exclusi clientii care au rude printre angajatii clinicii (criteriul e ca numele de familie sa nu coincida cu al unui angajat)

```
CREATE OR REPLACE PROCEDURE castigator_tombola
(v_luna varchar2)
IS
v_castigator p_client.nume_client%TYPE;
BEGIN


select nume_client
into v_castigator
```

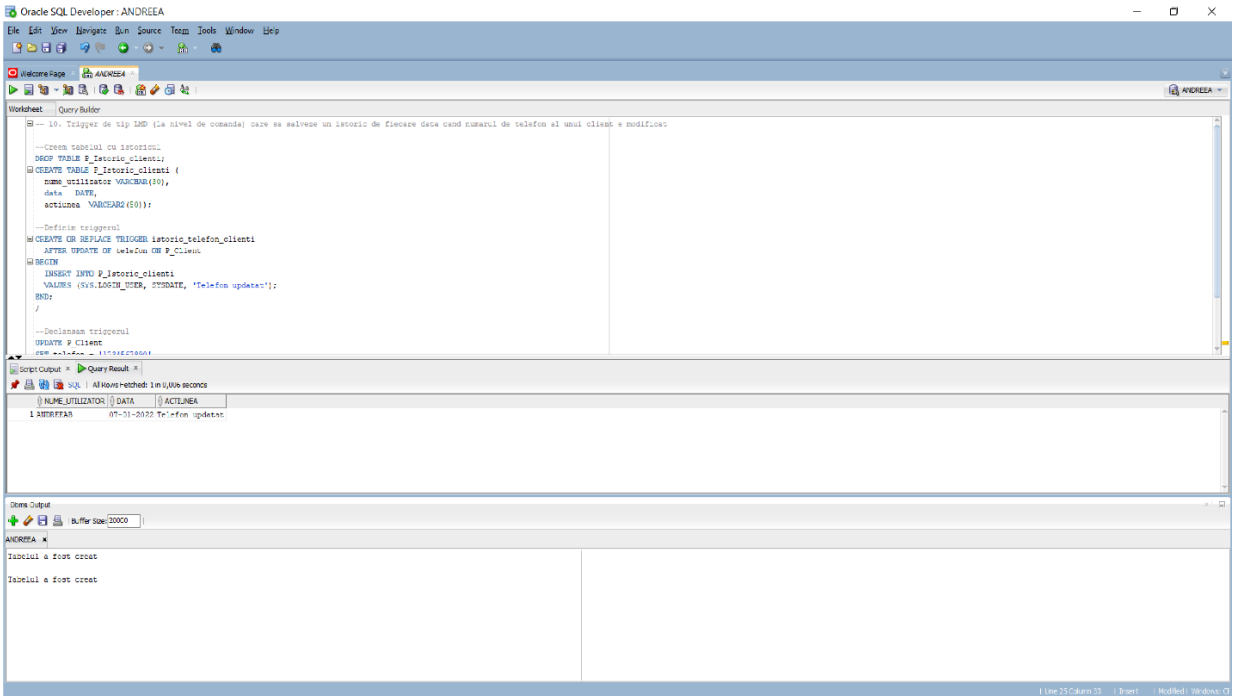
```
from (
select * from (
select
c.id_client,c.nume_client,c.adresa,c.telefon,to_char(p.data_programare,'MM')     as
luna,count(*) as rezervari from
p_programare p
join p_pacient pt on pt.id_pacient=p.id_pacient
join p_client c on c.id_client=pt.id_client
join p_clinica cl on cl.id_clinica=p.id_clinica
left join p_angajat a on a.id_clinica=cl.id_clinica and SubStr(c.nume_client,
InStr(c.nume_client, ' ')+1) = SubStr(a.nume_angajat, InStr(a.nume_angajat, ' ')+1)
where p.id_clinica=1 and
a.id_angajat is null  -- aici verificam ca nu avem angajati cu acelasi nume de
familie ca si clientul
group by
c.id_client,to_char(p.data_programare,'MM'),c.nume_client,c.adresa,c.telefon
order by to_char(p.data_programare,'MM') asc, count(*) desc)
where luna=v_luna and rownum=floor(dbms_random.value(1,3)) order by
rezervari desc);  -- rownum=floor(dbms_random.value(1,3)) genereaza un numar
aleatoriu dintre 1,2,3 si va afisa randul cu rownum egal cu acest numar aleatoriu
drept castigatorul tombolei

 DBMS_OUTPUT.PUT_LINE('Castigatorul pentru luna '  || v_luna || ' este '||
v_castigator);
 EXCEPTION
 WHEN NO_DATA_FOUND THEN DBMS_OUTPUT.PUT_LINE('Castigatorul
nu exista sau e ruda cu un angajat');
 WHEN TOO_MANY_ROWS THEN DBMS_OUTPUT.PUT_LINE('Eroare: Mai
mult de un castigator pentru luna ' || v_luna);
 WHEN OTHERS THEN DBMS_OUTPUT.PUT_LINE('Alta eroare!');
 END castigator_tombola;

 /
BEGIN
castigator_tombola('06');
castigator_tombola('07');
END;
/
```




**10.** Definim un trigger de tip LMD (la nivel de comanda) care sa salveze un istoric de fiecare data cand numarul de telefon al unui client este modificat.

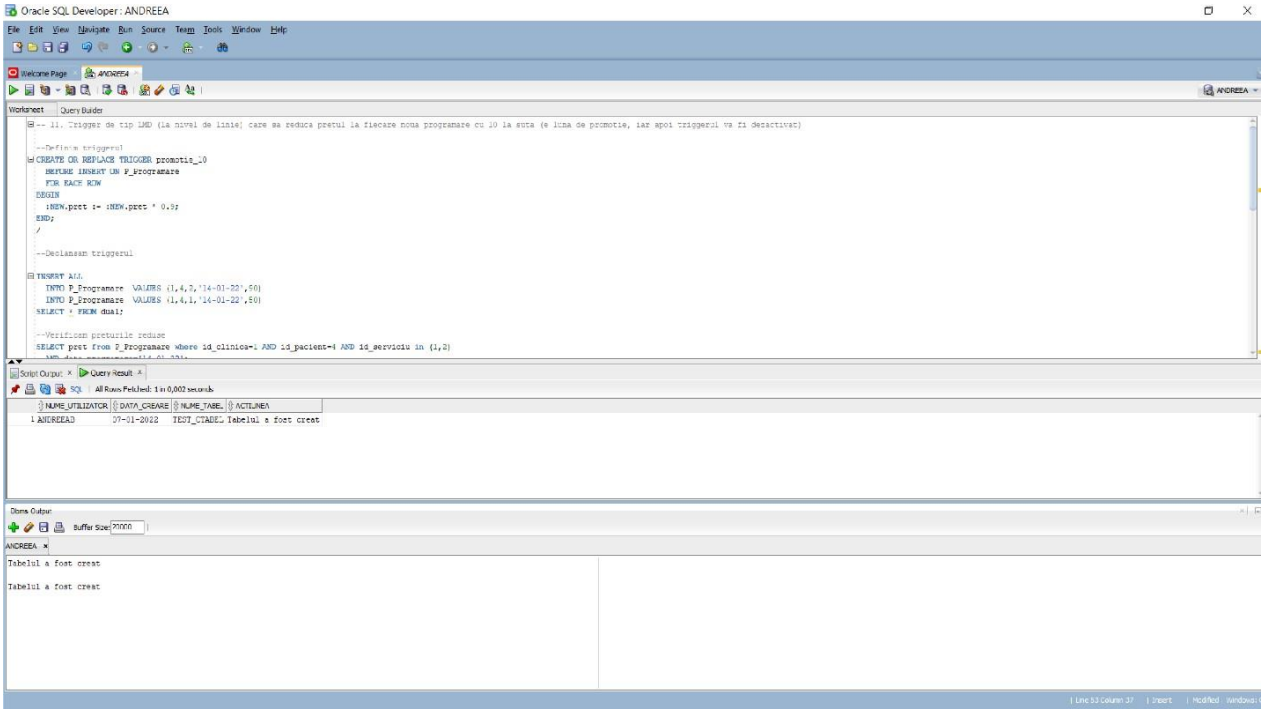
```
--Creem tabelul cu istoricul
DROP TABLE P_Istoric_clienti;
CREATE TABLE P_Istoric_clienti (
  nume_utilizator VARCHAR(30),
  data   DATE,
  actiunea  VARCHAR2(50));
```

```
--Definim triggerul
CREATE OR REPLACE TRIGGER istoric_telefon_clienti
  AFTER UPDATE OF telefon ON P_Client
BEGIN
  INSERT INTO P_Istoric_clienti
  VALUES (SYS.LOGIN_USER, SYSDATE, 'Telefon updatat');
END;
/


--Declansam triggerul
UPDATE P_Client
SET telefon = '1234567890'
WHERE id_client = 2;

--Verificam istoricul
SELECT * from P_Istoric_clienti;
```



**11.**. Definim un trigger de tip LMD (la nivel de linie) care sa reduca pretul la fiecare noua programare cu 10 la suta (e luna de promotie, iar apoi triggerul va fi dezactivat)

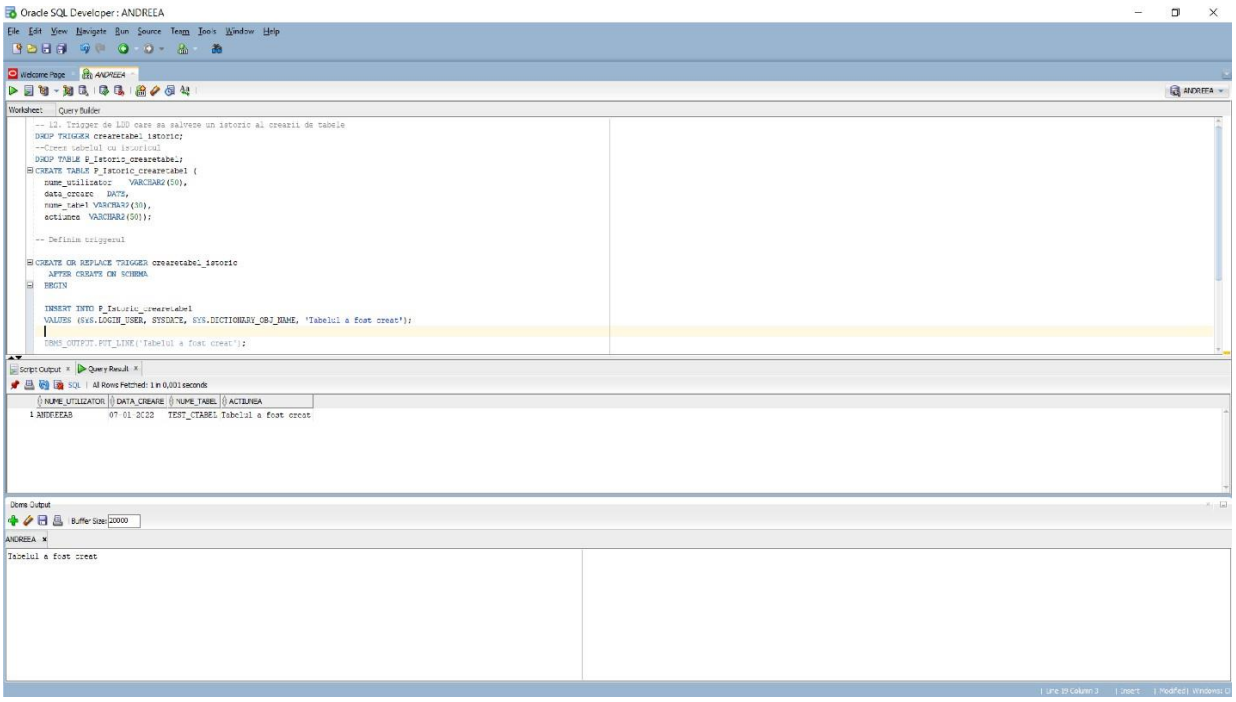
```
--Definim triggerul
CREATE OR REPLACE TRIGGER promotie_10
  BEFORE INSERT ON P_Programare
  FOR EACH ROW
BEGIN
  :NEW.pret := :NEW.pret * 0.9;
END;
/
--Declansam triggerul

INSERT ALL
  INTO P_Programare  VALUES (1,4,2,'14-01-22',90)
  INTO P_Programare  VALUES (1,4,1,'14-01-22',50)
SELECT * FROM dual;

--Verificam preturile reduse
SELECT pret from P_Programare where id_clinica=1 AND id_pacient=4 AND
id_serviciu in (1,2)
  AND data_programare='14-01-22';
```




12. Definiti un *trigger* de tip LDD care sa salveze un istoric al crearii de tabele.

```
DROP TRIGGER crearetabel_istoric;

--Creem tabelul cu istoricul

DROP TABLE P_Istoric_crearetabel;

CREATE TABLE P_Istoric_crearetabel (

nume_utilizator    VARCHAR2(50),

data_creare   DATE,

  nume_tabel VARCHAR2(30),

  actiunea_VARCHAR2(50));


 -- Definim triggerul

 CREATE OR REPLACE TRIGGER crearetabel_istoric
```

```
  AFTER CREATE ON SCHEMA
 BEGIN


 INSERT INTO P_Istoric_crearetabel
 VALUES (SYS.LOGIN_USER, SYSDATE, SYS.DICTIONARY_OBJ_NAME,
'Tabelul a fost creat');
 DBMS_OUTPUT.PUT_LINE('Tabelul a fost creat');
 END;
/
-- Creem un tabel test
 CREATE TABLE test_ctabel (test_coloana1 number(2));
 DROP TABLE test_ctabel;
```



13. Definiti un pachet care sa contina toate obiectele definite in cadrul proiectului.

```
CREATE OR REPLACE PACKAGE pachet AS
PROCEDURE calcul_bonus;
PROCEDURE afisare_programari;
FUNCTION gaseste_client (v_nume P_Pacient.nume_pacient%TYPE)
RETURN varchar2;
PROCEDURE castigator_tombola (v_luna varchar2);
END pachet;

CREATE OR REPLACE PACKAGE BODY pachet AS

--6. Angajatii din departamentul medical vor primi un bonus de vechime(10%
pentru 1 an vechime, 20% pentru minim 3 ani vechime si 40% pentru minim 5 ani
de vechime). Procentul se aplica la media pe numarul de angajati totali a sumei
incasate de clinica (prin pretul serviciilor programate). Sa se afiseze cat primeste
fiecare angajat in functie de anii de vechime.
PROCEDURE calcul_bonus

AS

TYPE vector IS VARRAY(2) OF NUMBER;
t vector:= vector();
-- Partea de tablou indexat
TYPE tablou_indexat IS TABLE OF NUMBER INDEX BY PLS_INTEGER;
t2 tablou_indexat;
i INTEGER;
BEGIN
-- Partea de vector
t.extend;
select sum(pret) INTO t(1) from (select sum(pret) as pret from p_programare where
id_clinica=1); --suma preturilor programarilor pentru clinica noastra
t.extend;
select count(*) INTO t(2) from p_angajat where id_clinica=1; -- numarul total de
angajati pentru clinica noastra
-- Partea de tablou indexat
```

```
t2(0) := 0; -- la vechimea 0 bonusul e 0;
t2(1) := round(t(1)/t(2) * 0.10); -- vechimea 1 , bonusul e 10%
t2(2) := round(t(1)/t(2) * 0.10); -- vechimea 2 , bonusul e 10%
t2(3) := round(t(1)/t(2) * 0.30); -- vechimea 3 , bonusul e 30%
t2(4) := round(t(1)/t(2) * 0.30); -- vechimea 4 , bonusul e 30%
t2(5) := round(t(1)/t(2) * 0.40); -- vechimea 5 , bonusul e 40%
--afisare elemente tablou
i := t2.FIRST;
WHILE i <= 5
LOOP
DBMS_OUTPUT.PUT_LINE('Bonusul pentru vechimea de '|| i ||' an(i) este ' || t2(i)
);
i := t2.NEXT(i);
END LOOP;
t2.delete;
END calcul_bonus;

--7. Folosind un cursor si un subprogram stocat afisati toate programarile din luna
iulie 2021.
--vom folosi o procedura pentru ca nu vrem sa returnam o valoare
PROCEDURE afisare_programari
AS
CURSOR c IS
SELECT nume_client, nume_clinica, nume_serviciu, data_programare, pret,
nume_pacient
FROM P_Programare pr
join P_Clinica c on c.id_clinica = pr.id_clinica
join P_Pacient p on p.id_pacient = pr.id_pacient
join P_Client cl on cl.id_client = p.id_client
join P_Serviciu s on s.id_serviciu = pr.id_serviciu
WHERE data_programare BETWEEN '01-07-2021' AND '31-07-2021';
v_nume_client P_Client.nume_client%TYPE;
v_nume_clinica P_Clinica.nume_clinica%TYPE;
v_nume_serviciu P_Serviciu.nume_serviciu%TYPE;
v_data_programare P_Programare.data_programare%TYPE;
v_pret P_Programare.pret%TYPE;
v_nume_pacient P_Pacient.nume_pacient%TYPE;
```

```
BEGIN
BEGIN
OPEN c;
LOOP
FETCH c INTO v_nume_client, v_nume_clinica, v_nume_serviciu,
v_data_programare, v_pret, v_nume_pacient;
EXIT WHEN c%NOTFOUND;
DBMS_OUTPUT.PUT_LINE('Clientul ' || v_nume_client || ' a programat pacientul '
|| v_nume_pacient || ' la clinica ' || v_nume_clinica || ' pentru serviciul de ' ||
v_nume_serviciu
|| ' la data de ' || v_data_programare || ', pretul vizitei fiind ' || v_pret || '.');
END LOOP;
CLOSE c;
END;
END afisare_programari;

--8. Utilizand un subprogram stocat (functie) sa se implementeze un mod de a gasi
informatiile clientului atunci cand trimitem numele pacientului (pet-ului) ca
parametru, pentru pacientii care au cel putin o programare facuta.
FUNCTION gaseste_client
(v_nume P_Pacient.nume_pacient%TYPE)
RETURN varchar2
IS
info_client varchar2(300);
BEGIN
SELECT c.id_client || ' - ' || nume_client INTO info_client
FROM p_client c
join p_pacient p on p.id_client=c.id_client
join p_programare pr on pr.id_pacient=p.id_pacient
where p.nume_pacient=v_nume;
RETURN info_client;
EXCEPTION
WHEN NO_DATA_FOUND THEN RETURN 'Nu exista pacienti cu numele ' ||
v_nume || ' sau nu are programare facuta';
WHEN TOO_MANY_ROWS THEN RETURN 'Exista mai multi pacienti (pets) cu
numele ' || v_nume;
WHEN OTHERS THEN RETURN 'Alta eroare!';
```

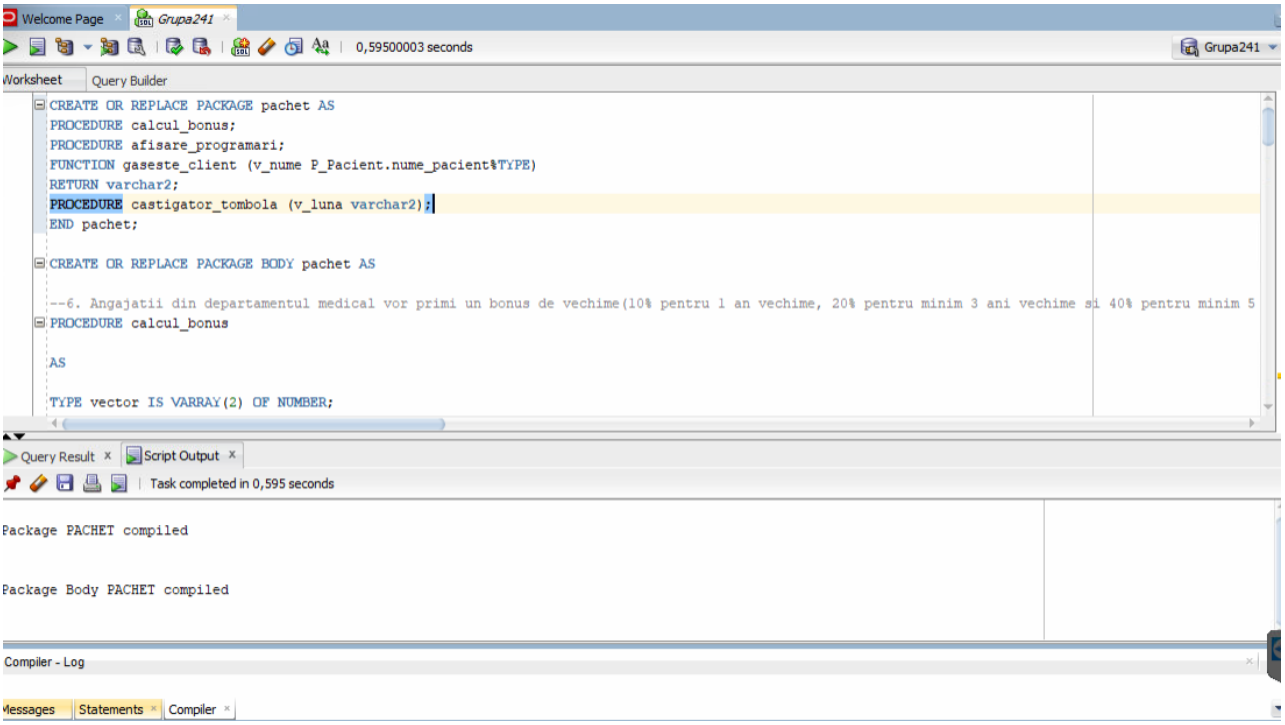
END gaseste_client;

--9. Utilizand un subprogram stocat (procedura) sa se rezolve urmatoarea cerinta: Clinica organizeaza o tombola lunara in care vor intra primii 3 clienti cu cele mai multe programari facute. Pentru luna iulie 2021, sa se afiseze cine sunt cei 3 clienti, cate programari au fiecare, si sa se aleaga aleatoriu cate un castigator. Sunt exclusi clientii care au rude printre angajatii clinicii (criteriul e ca numele de familie sa nu coincida cu al unui angajat)

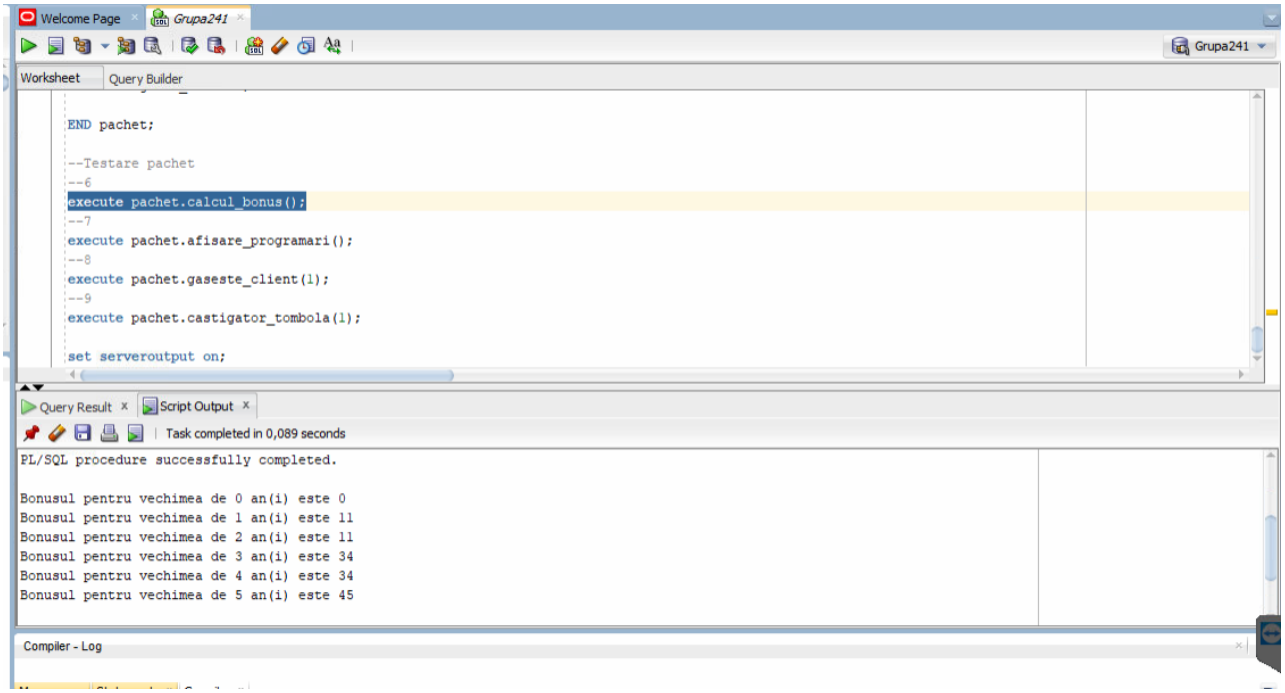
```
PROCEDURE castigator_tombola
(v_luna varchar2)
AS
v_castigator p_client.nume_client%TYPE;
BEGIN
select nume_client
into v_castigator
from (
select * from (
select c.id_client,c.nume_client,c.adresa,c.telefon,to_char(p.data_programare,'MM')
as luna,count(*) as rezervari from
p_programare p
join p_pacient pt on pt.id_pacient=p.id_pacient
join p_client c on c.id_client=pt.id_client
join p_clinica cl on cl.id_clinica=p.id_clinica
left join p_angajat a on a.id_clinica=cl.id_clinica and SubStr(c.nume_client,
InStr(c.nume_client, ' ')+1) = SubStr(a.nume_angajat, InStr(a.nume_angajat, ' ')+1)
where p.id_clinica=1 and
a.id_angajat is null -- aici verificam ca nu avem angajati cu acelasi nume de familie
ca si clientul
group by
c.id_client,to_char(p.data_programare,'MM'),c.nume_client,c.adresa,c.telefon
order by to_char(p.data_programare,'MM') asc, count(*) desc)
where luna=v_luna and rownum=floor(dbms_random.value(1,3)) order by rezervari
desc); -- rownum=floor(dbms_random.value(1,3)) genereaza un numar aleatoriu
dintre 1,2,3 si va afisa randul cu rownum egal cu acest numar aleatoriu drept
castigatorul tombolei
DBMS_OUTPUT.PUT_LINE('Castigatorul pentru luna ' || v_luna || ' este '||
v_castigator);
```

EXCEPTION
WHEN NO_DATA_FOUND THEN DBMS_OUTPUT.PUT_LINE('Castigatorul nu exista sau e ruda cu un angajat');
WHEN TOO_MANY_ROWS THEN DBMS_OUTPUT.PUT_LINE('Eroare: Mai mult de un castigator pentru luna ' || v_luna);
WHEN OTHERS THEN DBMS_OUTPUT.PUT_LINE('Alta eroare!');
END castigator_tombola;

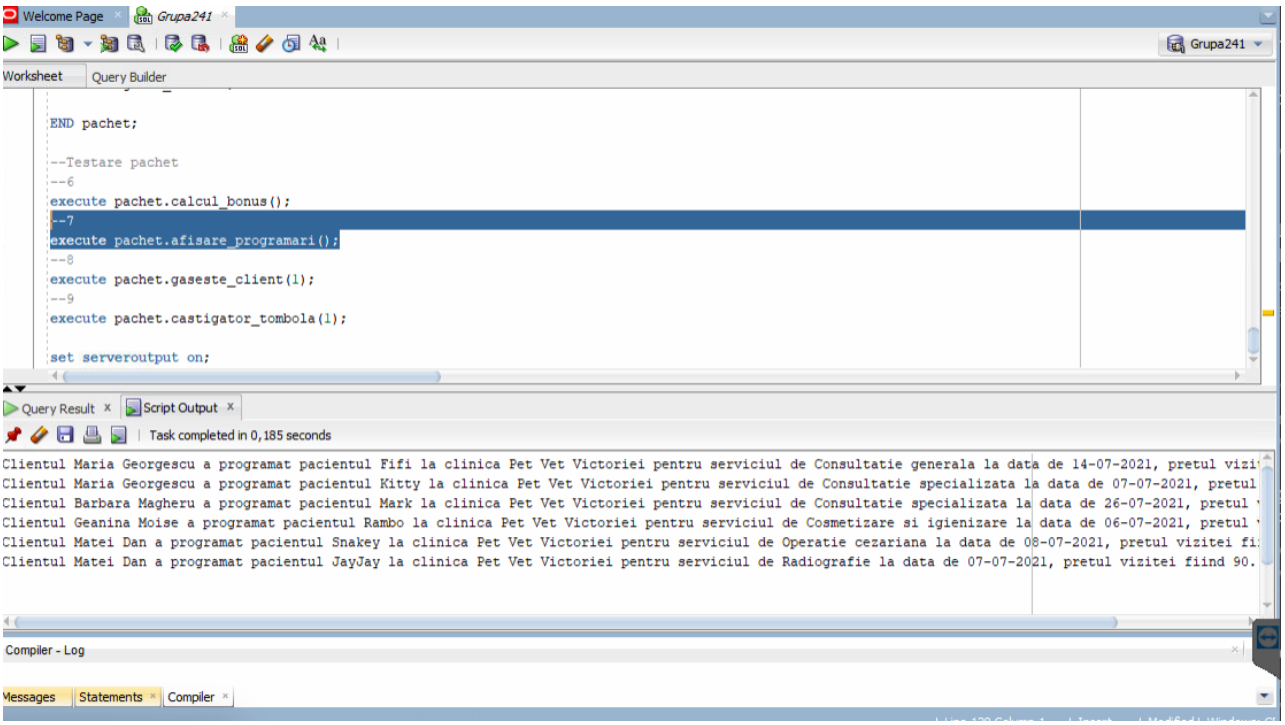
END pachet;



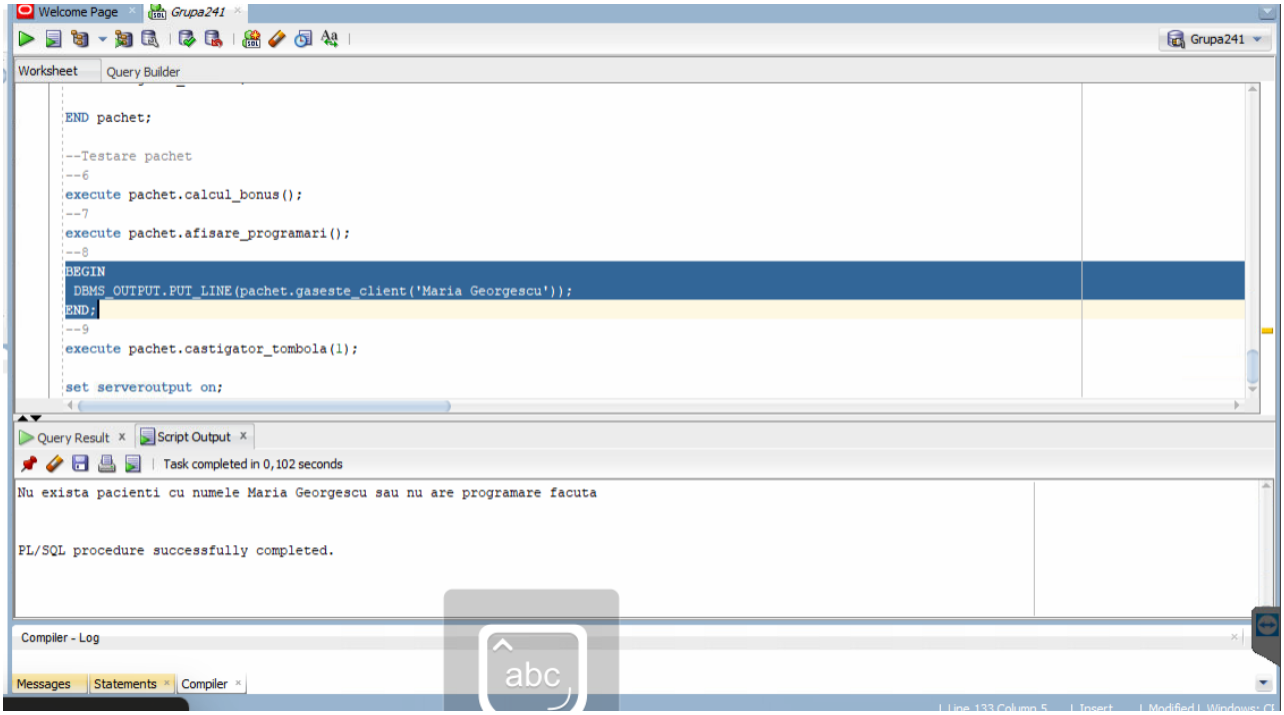
--Testare pachet
--6
execute pachet.calcul_bonus();



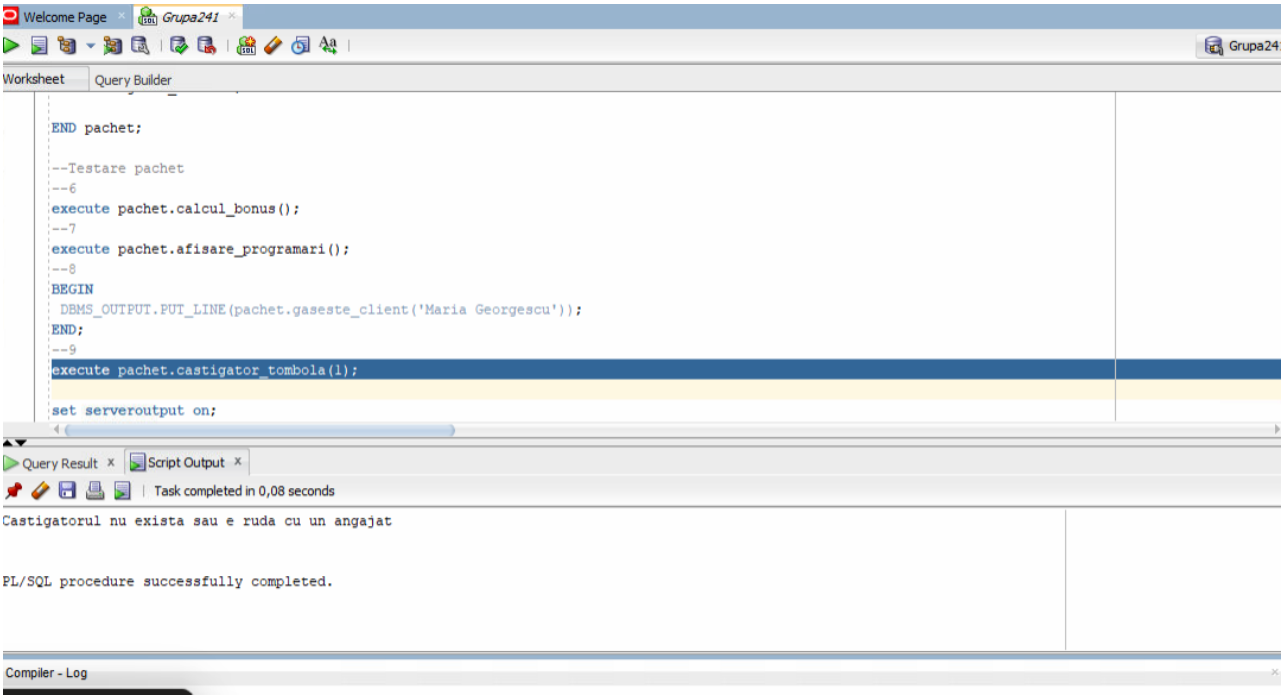

--7
execute pachet.afisare_programari();



--8
BEGIN
 DBMS_OUTPUT.PUT_LINE(pachet.gaseste_client('Maria Georgescu'));
END;




--9
execute pachet.castigator_tombola(1);




set serveroutput on;