Gestionarea unui lant de centre de adoptie pentru animale

Student: Tudoroiu Simona

Grupa: 251

1. ***Prezentați pe scurt baza de date (utilitatea ei).***
2. ***Descrierea:***

Modelul de date ofera informatii despre modul de functionare al unui lant de centre de adoptie pentru animale, oferind detalii despre centrul de adoptie, animalele disponibile, personalul centrului de adoptie impreuna cu departamentele si joburile disponibile.

Pot exista animale ce sufera de anumite boli. Aceste vor beneficia de un tratament special fata de restul animalelor, primind astfel un medicament adecvat bolii.

Animalele detin un carnet de vaccinuri. Acesta va retine istoricul vaccinurilor pana in momentul de fata.

Un animal poate fi de rasa. Acest lucru se poate atesta printr-un pedigree.

In functie de animal, centrul de adoptie trebuie sa retina anumite tipuri de hrana, speciala pentru acel animal.

Pentru ca centrul de adoptie sa functioneze, este nevoie de un personal calificat. Personalul este impartit in mai multe departamente, de exemplu: departamentul doctorilor veterinari, departamentul de curatenie, departamentul de administratie etc., fiind disponibile mai multe joburi.

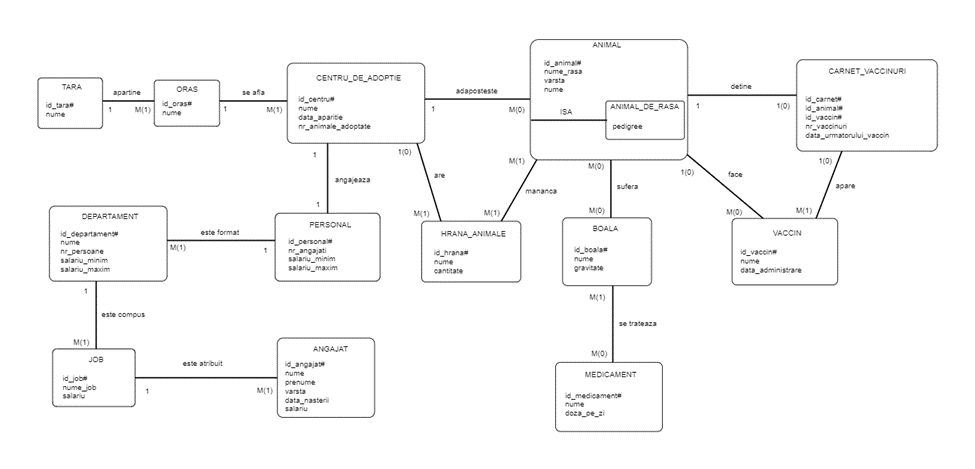
1. ***Regulile de funcționare ale modelului de date:***

* Un centru de adoptie pentru animale are o locatie stabilita, adica un oras din care face parte, si o tara, de unde apartine orasul.
* Un centru de adoptie poate adaposti mai multe animale, dar in cazul in care toate animalele sunt adoptate, atunci centrul nu mai adaposteste niciun animal pana in momentul aparitiei altui animal. Un animal poate fi adapostit de un singur centru de adoptie.
* Un animal poate detine un singur carnet de vaccinuri, acesta reprezentand istoricul de vaccinuri facute de acel animal. Insa animalul nu este obligat sa aiba un carnet de vaccinuri, fiind posibil ca persoanele ce adopta animalul sa il faca.
* Unui animal i se poate face mai multe vaccinuri, sau niciunul. In cazul in care unui animal ii este administrat cel putin un vaccin, atunci i se intocmeste un carnet de vaccinuri, unde vor aparea vaccinurile facute pana in acel moment.
* Un animal poate suferi de o boala pe care doctorii veterinari din centrul de adoptie o detecteaza. Aceasta boala se trateaza in centru folosind un anume medicament. Poate exista situatia in care un animal ce sufera de o boala nu poate fi tratat cu niciun medicament.
* Centrul de adoptie trebuie sa detina hrana pentru animale, pentru a putea ingriji animalele adapostite pana cand acestea sunt adoptate.
* Centrul de adoptie poate angaja maxim un personal pentru ca acesta sa functioneze.
* Personalul este impartit in mai multe departamente, de exemplu: departamentul doctorilor veterinar, cel de curatenie, de administratie etc.
* Pentru fiecare departament exista mai multe joburi, dar este necesar sa existe minim un job la fiecare departament.
* Pentru fiecare job sunt angajate mai multe persoane, dar este necesar sa fie angajate minim o persoane la fiecare job disponibil.

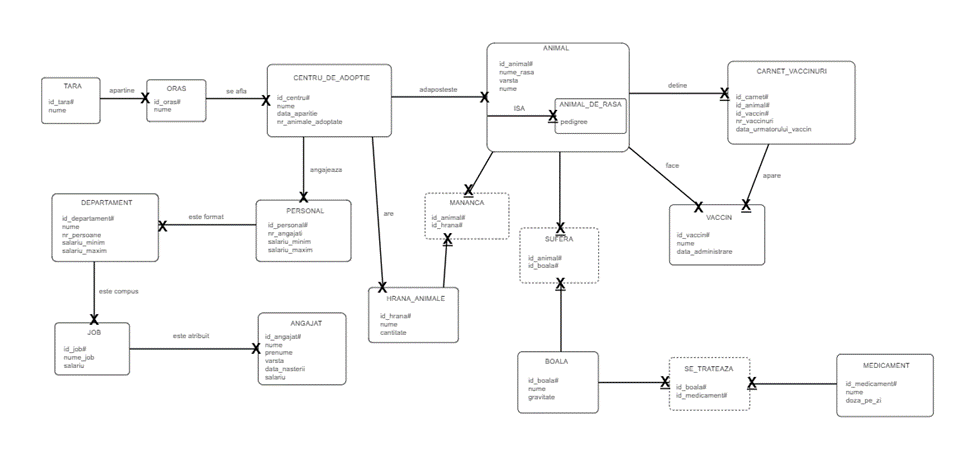
1. ***Constrangeri:***

* Un centru de adoptie trebuie sa aiba minim o hrana de animale.
* Un animal detine doar un carnet de vaccinuri.
* Un animal trebuie sa fie adapostit de cel putin un centru de adoptie.
* Intr-un carnet de vaccinuri trebuie sa existe cel putin un vaccin pentru ca aceste sa fie valid.
* Un medicament trebuie sa trateze cel putin o boala detectata.
* Un centru de adoptie poate sa angajeze maxim un personal.
* Un personal este format din minim un departament.
* Un departament trebuie sa fie compus din minim un job.
* La un job trebuie sa lucreze minim un angajat.

1. ***Realizați diagrama entitate-relație (ERD).***



***3. Pornind de la diagrama entitate-relație realizați diagrama conceptuală a modelului propus, integrând toate atributele necesare.***

******

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

**Crearea tabelelor:**

create table tara

(

id\_tara number(5),

nume varchar(25) constraint nume\_tara\_nn not null,

unique (nume)

);

alter table tara

add constraint id\_tara\_pk primary key(id\_tara);

create table oras

(

id\_oras number(5),

nume varchar(25) constraint nume\_oras\_nn not null,

id\_tara number(5),

unique (nume)

);

alter table oras

add constraint id\_oras\_pk primary key(id\_oras);

alter table oras

add constraint id\_tara\_fk foreign key(id\_tara) references tara(id\_tara);

create table centru\_de\_adoptie

(

id\_centru number(5),

nume varchar(50) constraint nume\_centru\_nn not null,

data\_aparitie date constraint aparitie\_nn not null,

nr\_animale\_adoptate number(10),

id\_oras number(5),

unique (nume)

);

alter table centru\_de\_adoptie

add constraint id\_centru\_pk primary key(id\_centru);

alter table centru\_de\_adoptie

add constraint id\_oras\_fk foreign key(id\_oras) references oras(id\_oras);

create table hrana\_animale

(

id\_hrana number(5),

nume varchar(30) constraint nume\_hrana\_nn not null,

cantitate number(3) constraint cantitate\_hrana\_nn not null,

id\_centru number(5),

constraint id\_hrana\_pk primary key(id\_hrana),

constraint id\_centru\_fk foreign key(id\_centru) references centru\_de\_adoptie(id\_centru)

);

create table animal

(

id\_animal number(5),

nume\_rasa varchar(25),

varsta number(3) constraint varsta\_animal\_nn not null,

nume varchar(25),

id\_centru number(5),

pedigree varchar(10) constraint pedigree\_nn not null,

constraint id\_animal\_pk primary key(id\_animal),

constraint id\_centru\_2\_fk foreign key(id\_centru) references centru\_de\_adoptie(id\_centru),

constraint pedigree\_a check (pedigree='are' or pedigree='nu are')

);

create table vaccin

(

id\_vaccin number(5),

nume varchar(30) constraint nume\_vaccin\_nn not null,

data\_administrare date constraint data\_vaccin\_nn not null,

id\_animal number(5),

constraint id\_vaccin\_pk primary key(id\_vaccin),

constraint id\_animal\_fk foreign key(id\_animal) references animal(id\_animal)

);

create table carnet\_vaccinuri

(

id\_carnet number(5),

id\_animal number(5),

id\_vaccin number(5),

nr\_vaccinuri number(2) constraint nr\_vaccinuri\_nn not null,

data\_urmatorului\_vaccin date,

constraint pk\_carnet\_animal\_vaccin primary key(id\_carnet, id\_animal, id\_vaccin),

constraint id\_animal\_2\_fk foreign key (id\_animal) references animal (id\_animal),

constraint id\_vaccin\_fk foreign key (id\_vaccin) references vaccin (id\_vaccin)

);

create table boala

(

id\_boala number(5),

nume varchar(30) constraint nume\_boala\_nn not null,

gravitate number(2) constraint gravitate\_boala\_nn not null,

constraint id\_boala\_pk primary key (id\_boala)

);

create table medicament

(

id\_medicament number(5),

nume varchar(30) constraint nume\_medicament\_nn not null,

doza\_pe\_zi number(2) constraint doza\_medicament\_nn not null,

constraint id\_medicament\_pk primary key (id\_medicament)

);

create table personal

(

id\_personal number(5),

nr\_angajati number(4) constraint nr\_angajati\_nn not null,

salariu\_minim number(5) constraint sal\_min\_p\_nn not null,

salariu\_maxim number(5) constraint sal\_max\_p\_nn not null,

id\_centru number(5),

constraint id\_personal\_pk primary key (id\_personal),

constraint id\_centru\_3\_fk foreign key (id\_centru) references centru\_de\_adoptie (id\_centru)

);

create table departament

(

id\_departament number(5),

nume varchar(25) constraint nume\_departament\_nn not null,

nr\_persoane number(3) constraint nr\_persoane\_d\_nn not null,

salariu\_minim number(5) constraint sal\_min\_d\_nn not null,

salariu\_maxim number(5) constraint sal\_max\_d\_nn not null,

id\_personal number(5),

constraint id\_departament\_pk primary key (id\_departament),

constraint id\_personal\_fk foreign key (id\_personal) references personal (id\_personal)

);

create table job

(

id\_job number(5),

nume varchar(30) constraint nume\_job\_nn not null,

salariu number(5) constraint salariu\_job\_nn not null,

id\_departament number(5),

constraint id\_job\_pk primary key (id\_job),

constraint id\_departament\_fk foreign key (id\_departament) references departament (id\_departament)

);

create table angajat

(

id\_angajat number(5),

nume varchar(30) constraint nume\_angajat\_nn not null,

prenume varchar(30) constraint prenume\_angajat\_nn not null,

varsta number(2) constraint varsta\_angajat\_nn not null,

data\_nasterii date constraint data\_angajat\_nn not null,

salariu number(5) constraint salariu\_angajat\_nn not null,

id\_job number(5),

constraint id\_angajat\_pk primary key(id\_angajat),

constraint id\_job\_fk foreign key(id\_job) references job (id\_job)

);

create table sufera

(

id\_animal number(5),

id\_boala number(5),

constraint pk\_animal\_boala primary key (id\_animal, id\_boala),

constraint id\_animal\_4\_fk foreign key (id\_animal) references animal(id\_animal),

constraint id\_boala\_2\_fk foreign key (id\_boala) references boala (id\_boala)

);

create table se\_treateaza

(

id\_boala number(5),

id\_medicament number(5),

constraint pk\_boala\_medicament primary key (id\_boala, id\_medicament),

constraint id\_boala\_3\_fk foreign key (id\_boala) references boala (id\_boala),

constraint id\_medicament\_fk foreign key (id\_medicament) references medicament(id\_medicament)

);

create table mananca

(

id\_animal number(5),

id\_hrana number(5),

constraint pk\_animal\_hrana primary key (id\_animal, id\_hrana),

constraint id\_animal\_5\_fk foreign key (id\_animal) references animal(id\_animal),

constraint id\_hrana\_2\_fk foreign key (id\_hrana) references hrana\_animale (id\_hrana)

);

***5. Adăugați informații coerente în tabelele create (minim 5 înregistrări pentru fiecare entitate independentă; minim 10 înregistrări pentru tabela asociativă).***

**Inserarea de date:**

create sequence seq\_tara

increment by 1

start with 1

maxvalue 10000

nocycle;

insert into tara

values(seq\_tara.nextval, 'Romania');

insert into tara

values(seq\_tara.nextval, 'Franta');

insert into tara

values(seq\_tara.nextval, 'Germania');

insert into tara

values(seq\_tara.nextval, 'Italia');

insert into tara

values(seq\_tara.nextval, 'Bulgaria');

insert into tara

values(seq\_tara.nextval, 'Ucraina');

insert into tara

values(seq\_tara.nextval, 'Spania');

insert into tara

values(seq\_tara.nextval, 'Ungaria');

insert into tara

values(seq\_tara.nextval, 'Polonia');

insert into tara

values(seq\_tara.nextval, 'Danemarca');

commit;

create sequence seq\_oras

increment by 1

start with 1

maxvalue 10000

nocycle;

insert into oras

values(seq\_oras.nextval, 'Constanta', 1);

insert into oras

values(seq\_oras.nextval, 'Craiova', 1);

insert into oras

values(seq\_oras.nextval, 'Roma', 4);

insert into oras

values(seq\_oras.nextval, 'Barcelona', 7);

insert into oras

values(seq\_oras.nextval, 'Navodari', 1);

insert into oras

values(seq\_oras.nextval, 'Bucuresti', 1);

insert into oras

values(seq\_oras.nextval, 'Verona', 4);

insert into oras

values(seq\_oras.nextval, 'Timisoara', 1);

insert into oras

values(seq\_oras.nextval, 'Alba-Iulia', 1);

insert into oras

values(seq\_oras.nextval, 'Iasi', 1);

commit;

create sequence seq\_centru

increment by 1

start with 1

maxvalue 10000

nocycle;

insert into centru\_de\_adoptie

values(seq\_centru.nextval, 'Animalutele fericite', TO\_DATE('12-03-2004', 'dd-mm-yyyy'), 100, 2);

insert into centru\_de\_adoptie

values(seq\_centru.nextval, 'Adopta un prieten', TO\_DATE('02-11-2010', 'dd-mm-yyyy'), 150, 6);

insert into centru\_de\_adoptie

values(seq\_centru.nextval, 'Animalul tau', TO\_DATE('25-06-2012', 'dd-mm-yyyy'), 98, 10);

insert into centru\_de\_adoptie

values(seq\_centru.nextval, 'Adopta-ma!', TO\_DATE('16-09-2006', 'dd-mm-yyyy'), 200, 4);

insert into centru\_de\_adoptie

values(seq\_centru.nextval, 'Animale prietenoase', TO\_DATE('05-10-2013', 'dd-mm-yyyy'), 59, 5);

insert into centru\_de\_adoptie

values(seq\_centru.nextval, 'Animalutele fericite', TO\_DATE('12-03-2004', 'dd-mm-yyyy'), 100, 2);

insert into centru\_de\_adoptie

values(seq\_centru.nextval, 'Animalia', TO\_DATE('24-09-2001', 'dd-mm-yyyy'), 250, 9);

commit;

create sequence seq\_hrana

increment by 1

start with 1

maxvalue 10000

nocycle;

insert into hrana\_animale

values(seq\_hrana.nextval, 'Chonk', 300, 1);

insert into hrana\_animale

values(seq\_hrana.nextval, 'Cocomel', 579, 2);

insert into hrana\_animale

values(seq\_hrana.nextval, 'Papbun', 400, 3);

insert into hrana\_animale

values(seq\_hrana.nextval, 'Falrel', 280, 4);

insert into hrana\_animale

values(seq\_hrana.nextval, 'Varel', 600, 5);

insert into hrana\_animale

values(seq\_hrana.nextval, 'Chonk', 250, 2);

insert into hrana\_animale

values(seq\_hrana.nextval, 'Chonk', 190, 5);

insert into hrana\_animale

values(seq\_hrana.nextval, 'Cocomel', 398, 1);

insert into hrana\_animale

values(seq\_hrana.nextval, 'Cocomel', 400, 4);

insert into hrana\_animale

values(seq\_hrana.nextval, 'Varel', 387, 3);

insert into hrana\_animale

values(seq\_hrana.nextval, 'Papbun', 376, 1);

insert into hrana\_animale

values(seq\_hrana.nextval, 'Falrel', 578, 5);

commit;

create sequence seq\_animal

increment by 1

start with 1

maxvalue 10000

nocycle;

insert into animal

values(seq\_animal.nextval, 'British', 3, 'Mimi', 2, 'are');

insert into animal

values(seq\_animal.nextval, 'Scotish', 1, 'Misha', 5, 'nu are');

insert into animal

values(seq\_animal.nextval, 'nu stim', 2, 'Cara', 4,'nu are');

insert into animal

values(seq\_animal.nextval, 'Labrador', 4, 'Rex', 1,'are');

insert into animal

values(seq\_animal.nextval, 'Pug', 1, 'Boni', 3,'nu are');

commit;

create sequence seq\_vaccin

increment by 1

start with 1

maxvalue 10000

nocycle;

insert into vaccin

values(seq\_vaccin.nextval, 'Eurican', TO\_DATE('12-04-2005', 'dd-mm-yyyy'),1);

insert into vaccin

values(seq\_vaccin.nextval, 'Rabisim', TO\_DATE('04-03-2010', 'dd-mm-yyyy'),2);

insert into vaccin

values(seq\_vaccin.nextval, 'Purevax', TO\_DATE('24-11-2006', 'dd-mm-yyyy'),3);

insert into vaccin

values(seq\_vaccin.nextval, 'Primodog', TO\_DATE('15-12-2013', 'dd-mm-yyyy'),4);

insert into vaccin

values(seq\_vaccin.nextval, 'Eurican', TO\_DATE('02-09-2012', 'dd-mm-yyyy'),5);

commit;

create sequence seq\_carnet

increment by 1

start with 1

maxvalue 10000

nocycle;

insert into carnet\_vaccinuri

values(seq\_carnet.nextval, 1, 3, 1, TO\_DATE('23-07-2022', 'dd-mm-yyyy'));

insert into carnet\_vaccinuri

values(seq\_carnet.nextval, 2, 1, 3, TO\_DATE('12-09-2022', 'dd-mm-yyyy'));

insert into carnet\_vaccinuri

values(seq\_carnet.nextval, 3, 5, 6, TO\_DATE('15-10-2022', 'dd-mm-yyyy'));

insert into carnet\_vaccinuri

values(seq\_carnet.nextval, 4, 2, 2, TO\_DATE('13-11-2022', 'dd-mm-yyyy'));

insert into carnet\_vaccinuri

values(seq\_carnet.nextval, 5, 4, 1, TO\_DATE('25-08-2022', 'dd-mm-yyyy'));

commit;

create sequence seq\_boala

increment by 1

start with 1

maxvalue 10000

nocycle;

insert into boala

values(seq\_boala.nextval, 'Parazitoza', 3);

insert into boala

values(seq\_boala.nextval, 'Salmoneloza', 6);

insert into boala

values(seq\_boala.nextval, 'Stenoza Aortica', 10);

insert into boala

values(seq\_boala.nextval, 'Toxopalsmoza', 4);

insert into boala

values(seq\_boala.nextval, 'Otita', 5);

commit;

create sequence seq\_medicament

increment by 1

start with 1

maxvalue 10000

nocycle;

insert into medicament

values(seq\_medicament.nextval, 'Brufil', 3);

insert into medicament

values(seq\_medicament.nextval, 'Iosman', 2);

insert into medicament

values(seq\_medicament.nextval, 'Calmarin', 1);

insert into medicament

values(seq\_medicament.nextval, 'Maldon', 1);

insert into medicament

values(seq\_medicament.nextval, 'Aurilo', 2);

commit;

create sequence seq\_personal

increment by 1

start with 1

maxvalue 10000

nocycle;

insert into personal

values(seq\_personal.nextval, 100, 1300, 3000, 1);

insert into personal

values(seq\_personal.nextval, 210, 1250, 2998, 2);

insert into personal

values(seq\_personal.nextval, 87, 1100, 2500, 3);

insert into personal

values(seq\_personal.nextval, 180, 1400, 3100, 4);

insert into personal

values(seq\_personal.nextval, 150, 1247, 2876, 5);

commit;

create sequence seq\_departament

increment by 1

start with 1

maxvalue 10000

nocycle;

insert into departament

values(seq\_departament.nextval, 'Curatenie',30, 1300, 1400, 1);

insert into departament

values(seq\_departament.nextval, 'Curatenie',25, 1250, 1300, 2);

insert into departament

values(seq\_departament.nextval, 'Curatenie',15, 1100, 1278, 3);

insert into departament

values(seq\_departament.nextval, 'Curatenie',40, 1400, 1500, 4);

insert into departament

values(seq\_departament.nextval, 'Curatenie',35, 1247, 1350, 5);

insert into departament

values(seq\_departament.nextval, 'Curatenie',30, 1300, 1400, 1);

insert into departament

values(seq\_departament.nextval, 'Doctori',50, 2500, 3000, 1);

insert into departament

values(seq\_departament.nextval, 'Doctori',100, 2480, 2998, 2);

insert into departament

values(seq\_departament.nextval, 'Doctori',45, 2000, 2500, 3);

insert into departament

values(seq\_departament.nextval, 'Doctori',68, 2657, 3100, 4);

insert into departament

values(seq\_departament.nextval, 'Doctori',48, 2150, 2876, 5);

insert into departament

values(seq\_departament.nextval, 'Administratie',20, 1500, 2500, 1);

insert into departament

values(seq\_departament.nextval, 'Administratie',85, 1458, 2480, 2);

insert into departament

values(seq\_departament.nextval, 'Administratie',27, 1376, 2000, 3);

insert into departament

values(seq\_departament.nextval, 'Administratie',72, 2000, 2657, 4);

insert into departament

values(seq\_departament.nextval, 'Administratie',67, 1500, 2150, 5);

commit;

create sequence seq\_job

increment by 1

start with 1

maxvalue 10000

nocycle;

insert into job

values(seq\_job.nextval, 'Ingrijitor animale',1300, 1);

insert into job

values(seq\_job.nextval, 'Om de serviciu',1400, 1);

insert into job

values(seq\_job.nextval, 'Ingrijitor animale',1250, 2);

insert into job

values(seq\_job.nextval, 'Om de serviciu',1300, 2);

insert into job

values(seq\_job.nextval, 'Ingrijitor animale',1100, 3);

insert into job

values(seq\_job.nextval, 'Om de serviciu',1278, 3);

insert into job

values(seq\_job.nextval, 'Ingrijitor animale',1400, 4);

insert into job

values(seq\_job.nextval, 'Om de serviciu',1500, 4);

insert into job

values(seq\_job.nextval, 'Ingrijitor animale',1247, 5);

insert into job

values(seq\_job.nextval, 'Om de serviciu',1350, 5);

insert into job

values(seq\_job.nextval, 'Medic veterinar',2500, 6);

insert into job

values(seq\_job.nextval, 'Chirurg veterinar',3000, 6);

insert into job

values(seq\_job.nextval, 'Medic veterinar',2480, 7);

insert into job

values(seq\_job.nextval, 'Chirurg veterinar',2998, 7);

insert into job

values(seq\_job.nextval, 'Medic veterinar',2000, 8);

insert into job

values(seq\_job.nextval, 'Chirurg veterinar',2500, 8);

insert into job

values(seq\_job.nextval, 'Medic veterinar',2657, 9);

insert into job

values(seq\_job.nextval, 'Chirurg veterinar',3100, 9);

insert into job

values(seq\_job.nextval, 'Medic veterinar',2150, 10);

insert into job

values(seq\_job.nextval, 'Chirurg veterinar',2876, 10);

insert into job

values(seq\_job.nextval, 'Casier',1500, 11);

insert into job

values(seq\_job.nextval, 'Manager',2500, 11);

insert into job

values(seq\_job.nextval, 'Casier',1458, 12);

insert into job

values(seq\_job.nextval, 'Manager',2480, 12);

insert into job

values(seq\_job.nextval, 'Casier',1376, 13);

insert into job

values(seq\_job.nextval, 'Manager',2000, 13);

insert into job

values(seq\_job.nextval, 'Casier',2000, 14);

insert into job

values(seq\_job.nextval, 'Manager',2657, 14);

insert into job

values(seq\_job.nextval, 'Casier',1500, 15);

insert into job

values(seq\_job.nextval, 'Manager',2150, 15);

commit;

create sequence seq\_angajat

increment by 1

start with 1

maxvalue 10000

nocycle;

insert into angajat

values(seq\_angajat.nextval, 'Agache','Cosmin', 20,TO\_DATE('12-03-2002', 'dd-mm-yyyy'), 1300, 1);

insert into angajat

values(seq\_angajat.nextval, 'Popa','Andrei', 35, TO\_DATE('09-10-1987', 'dd-mm-yyyy'),1400, 2);

insert into angajat

values(seq\_angajat.nextval, 'Marius','Andrei', 48, TO\_DATE('10-02-1974', 'dd-mm-yyyy'), 1250, 3);

insert into angajat

values(seq\_angajat.nextval, 'Marian','Maria', 40, TO\_DATE('20-12-1982', 'dd-mm-yyyy'),1300, 4);

insert into angajat

values(seq\_angajat.nextval, 'Coman','Alexandru', 38, TO\_DATE('24-06-1984', 'dd-mm-yyyy'),1100, 5);

insert into angajat

values(seq\_angajat.nextval, 'Popescu','Alina', 28,TO\_DATE('30-09-1994', 'dd-mm-yyyy'), 1278, 6);

insert into angajat

values(seq\_angajat.nextval, 'Alexandru','Ion', 50,TO\_DATE('14-01-1972', 'dd-mm-yyyy'), 1400, 7);

insert into angajat

values(seq\_angajat.nextval, 'Oleanu','Marian', 43,TO\_DATE('08-08-1987', 'dd-mm-yyyy'), 1500, 8);

insert into angajat

values(seq\_angajat.nextval, 'Cornel','Maria', 35,TO\_DATE('24-06-1987', 'dd-mm-yyyy'), 1247, 9);

insert into angajat

values(seq\_angajat.nextval, 'Gheorghe','Andrei', 46,TO\_DATE('09-06-1976', 'dd-mm-yyyy'), 1350, 10);

insert into angajat

values(seq\_angajat.nextval, 'Stoica','Carina', 40, TO\_DATE('05-11-1982', 'dd-mm-yyyy'),2500, 11);

insert into angajat

values(seq\_angajat.nextval, 'Petru','Matei', 45,TO\_DATE('13-05-1977', 'dd-mm-yyyy'), 3000, 12);

insert into angajat

values(seq\_angajat.nextval, 'Marius','David', 37,TO\_DATE('04-12-1985', 'dd-mm-yyyy'), 2480, 13);

insert into angajat

values(seq\_angajat.nextval, 'Ion','Marian', 50,TO\_DATE('09-06-1972', 'dd-mm-yyyy'), 2998, 14);

insert into angajat

values(seq\_angajat.nextval, 'Cornel','Ionut', 34, TO\_DATE('25-10-1988', 'dd-mm-yyyy'),2000, 15);

insert into angajat

values(seq\_angajat.nextval, 'Gheorghe','Ana', 39,TO\_DATE('29-11-1983', 'dd-mm-yyyy'), 2500, 16);

insert into angajat

values(seq\_angajat.nextval, 'Pavel','George', 51, TO\_DATE('03-05-1971', 'dd-mm-yyyy'),2657, 17);

insert into angajat

values(seq\_angajat.nextval, 'Marius','Ion', 30,TO\_DATE('08-12-1992', 'dd-mm-yyyy'), 3100, 18);

insert into angajat

values(seq\_angajat.nextval, 'Agache','Maria', 40, TO\_DATE('04-10-1982', 'dd-mm-yyyy'),2150, 19);

insert into angajat

values(seq\_angajat.nextval, 'Petru','Costel', 35,TO\_DATE('04-05-1987', 'dd-mm-yyyy'), 2876, 20);

insert into angajat

values(seq\_angajat.nextval, 'Alexandru','Ioana', 25, TO\_DATE('10-10-1997', 'dd-mm-yyyy'),1500, 21);

insert into angajat

values(seq\_angajat.nextval, 'Ionescu','Carmen', 30,TO\_DATE('05-11-1992', 'dd-mm-yyyy'), 2500, 22);

insert into angajat

values(seq\_angajat.nextval, 'Pavel','Mirel', 22,TO\_DATE('10-12-2000', 'dd-mm-yyyy'), 1458, 23);

insert into angajat

values(seq\_angajat.nextval, 'Mitrel','Ionut', 28,TO\_DATE('12-09-1994', 'dd-mm-yyyy'), 2480, 24);

insert into angajat

values(seq\_angajat.nextval, 'Cordean','Marin', 32,TO\_DATE('12-02-1990', 'dd-mm-yyyy'), 1376, 25);

insert into angajat

values(seq\_angajat.nextval, 'Ionescu','Alexandra', 34, TO\_DATE('04-03-1988', 'dd-mm-yyyy'),2000, 26);

insert into angajat

values(seq\_angajat.nextval, 'Petre','Alin', 27,TO\_DATE('05-11-1995', 'dd-mm-yyyy'), 2000, 27);

insert into angajat

values(seq\_angajat.nextval, 'Cornel','Ioan', 34,TO\_DATE('10-10-1988', 'dd-mm-yyyy'), 2657, 28);

insert into angajat

values(seq\_angajat.nextval, 'Ion','Ana', 24,TO\_DATE('03-04-1998', 'dd-mm-yyyy'), 1500, 29);

insert into angajat

values(seq\_angajat.nextval, 'Popescu','Elena', 34, TO\_DATE('12-10-1988', 'dd-mm-yyyy'),2150, 30);

commit;

insert into sufera

values(1, 1);

insert into sufera

values(1, 2);

insert into sufera

values(1, 4);

insert into sufera

values(2, 1);

insert into sufera

values(2, 3);

insert into sufera

values(3, 1);

insert into sufera

values(3, 2);

insert into sufera

values(3, 3);

insert into sufera

values(3, 5);

insert into sufera

values(5, 1);

commit;

insert into se\_treateaza

values(1, 1);

insert into se\_treateaza

values(1, 2);

insert into se\_treateaza

values(1, 4);

insert into se\_treateaza

values(2, 1);

insert into se\_treateaza

values(2, 3);

insert into se\_treateaza

values(3, 1);

insert into se\_treateaza

values(3, 2);

insert into se\_treateaza

values(3, 3);

insert into se\_treateaza

values(3, 5);

insert into se\_treateaza

values(5, 1);

commit;

insert into mananca

values(1, 1);

insert into mananca

values(1, 12);

insert into mananca

values(1, 4);

insert into mananca

values(2, 10);

insert into mananca

values(2, 3);

insert into mananca

values(3, 9);

insert into mananca

values(3, 11);

insert into mananca

values(3, 2);

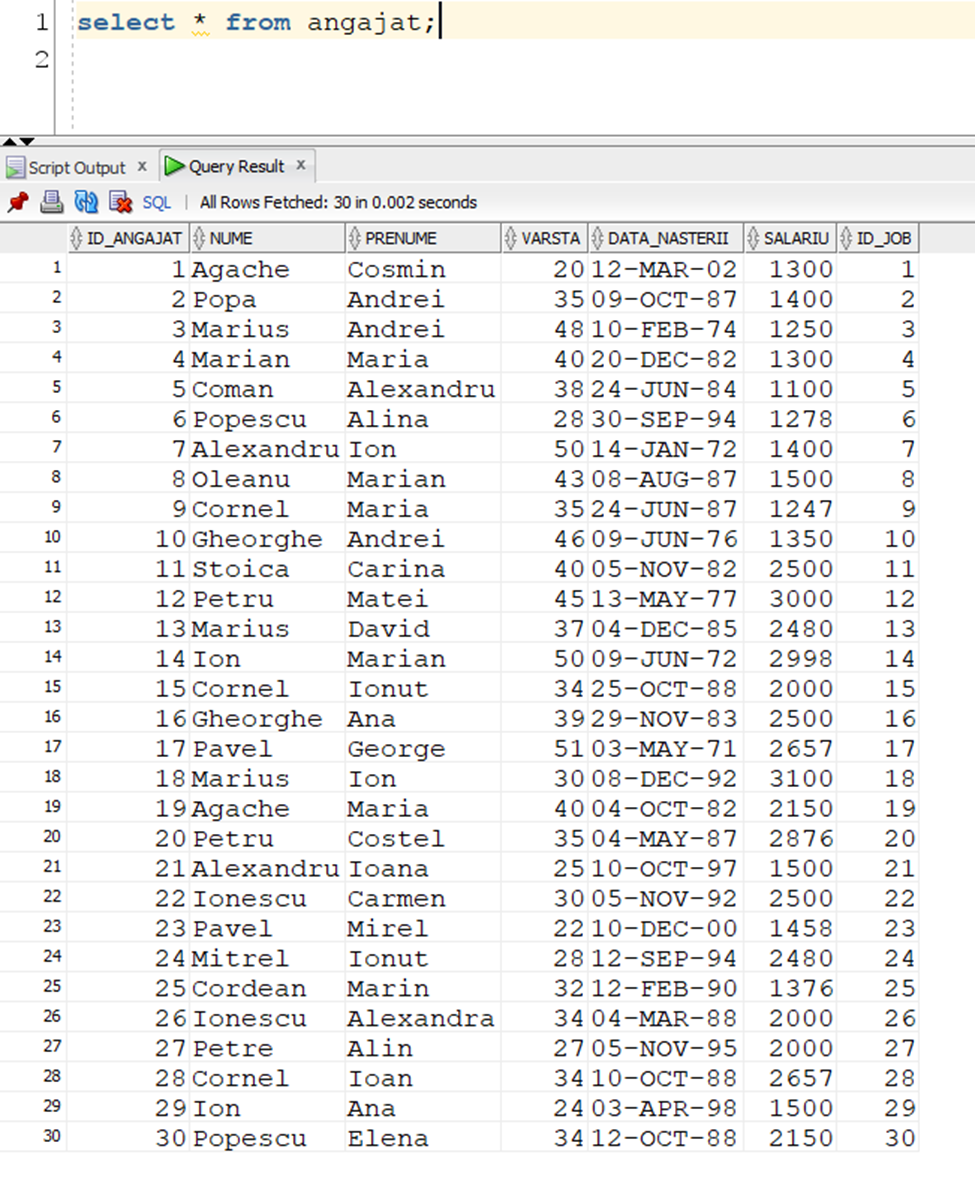
insert into mananca

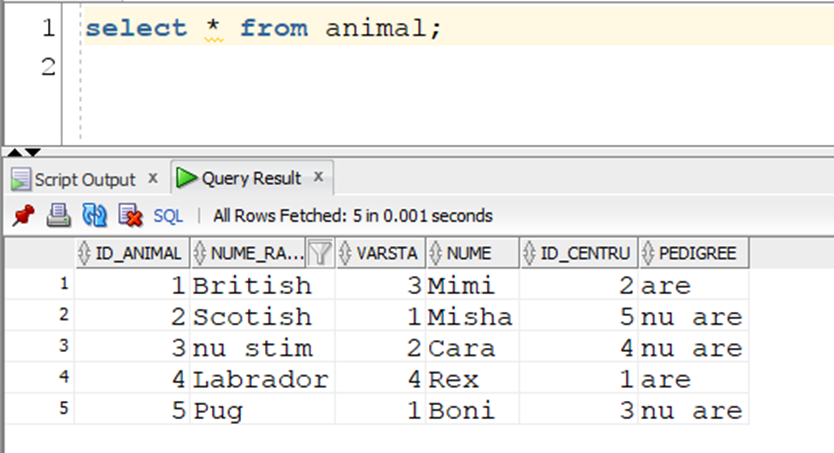
values(3, 12);

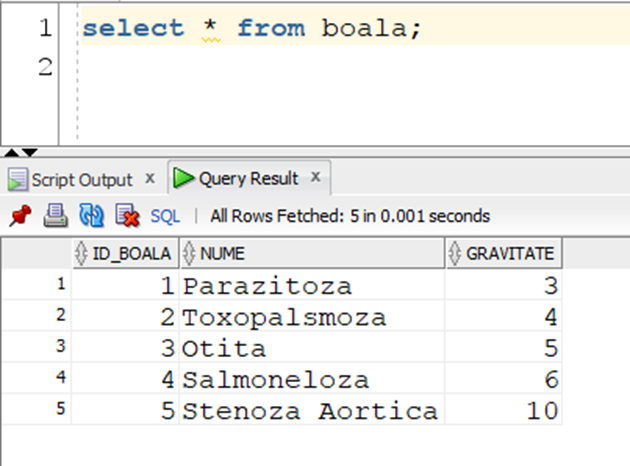
insert into mananca

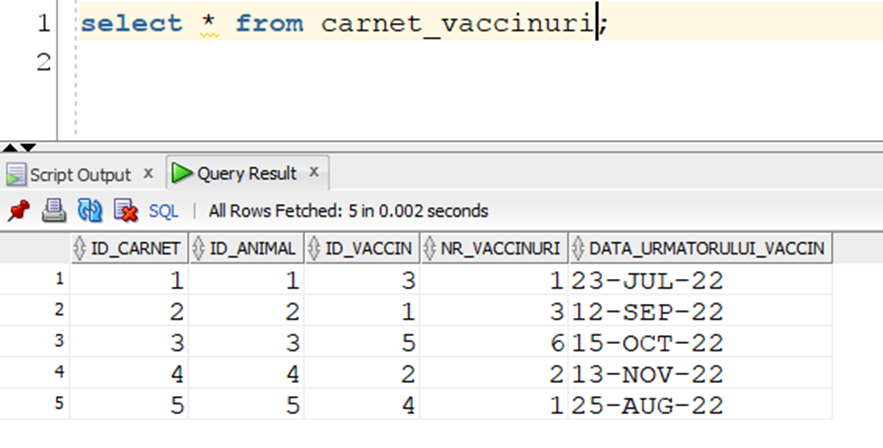
values(5, 10);

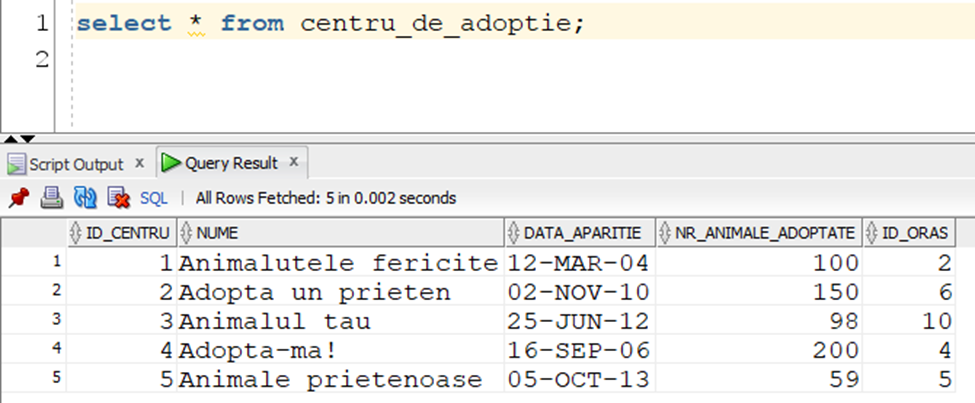
commit;

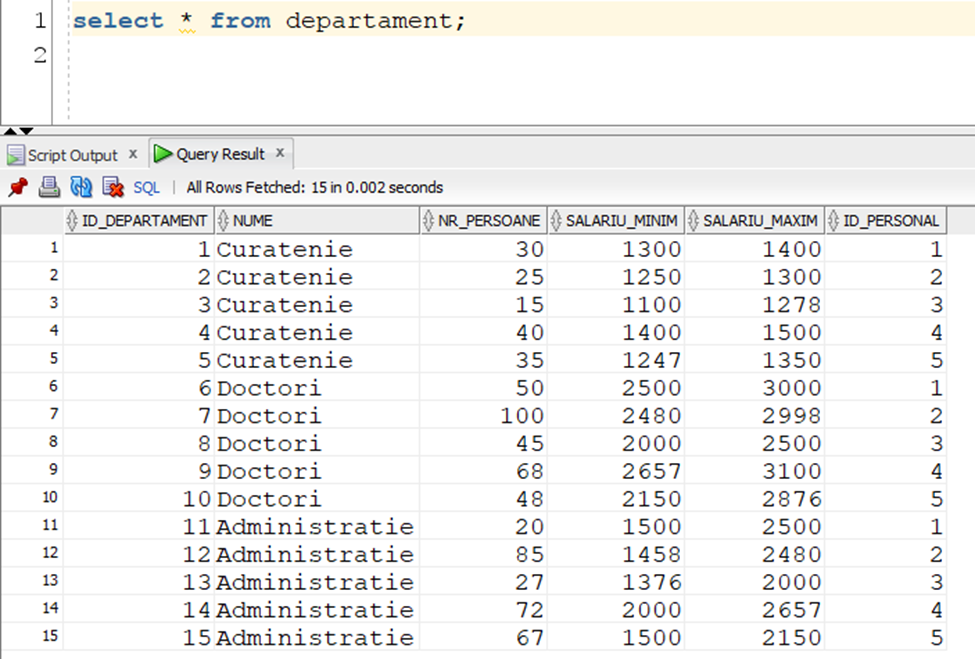


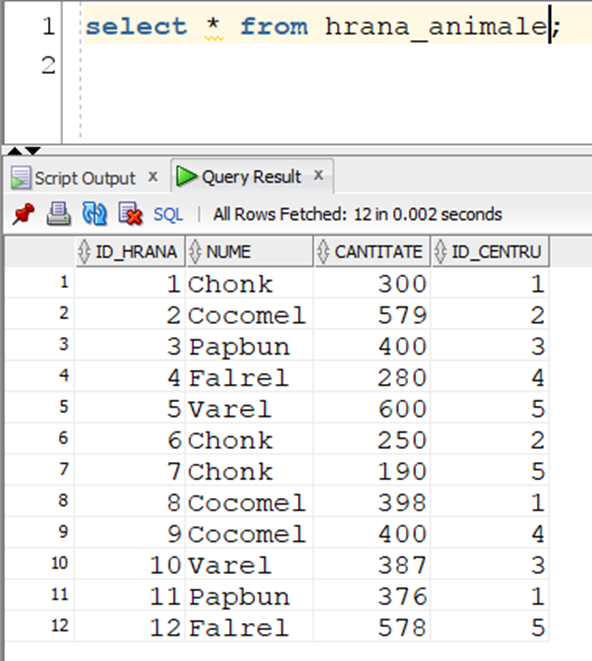


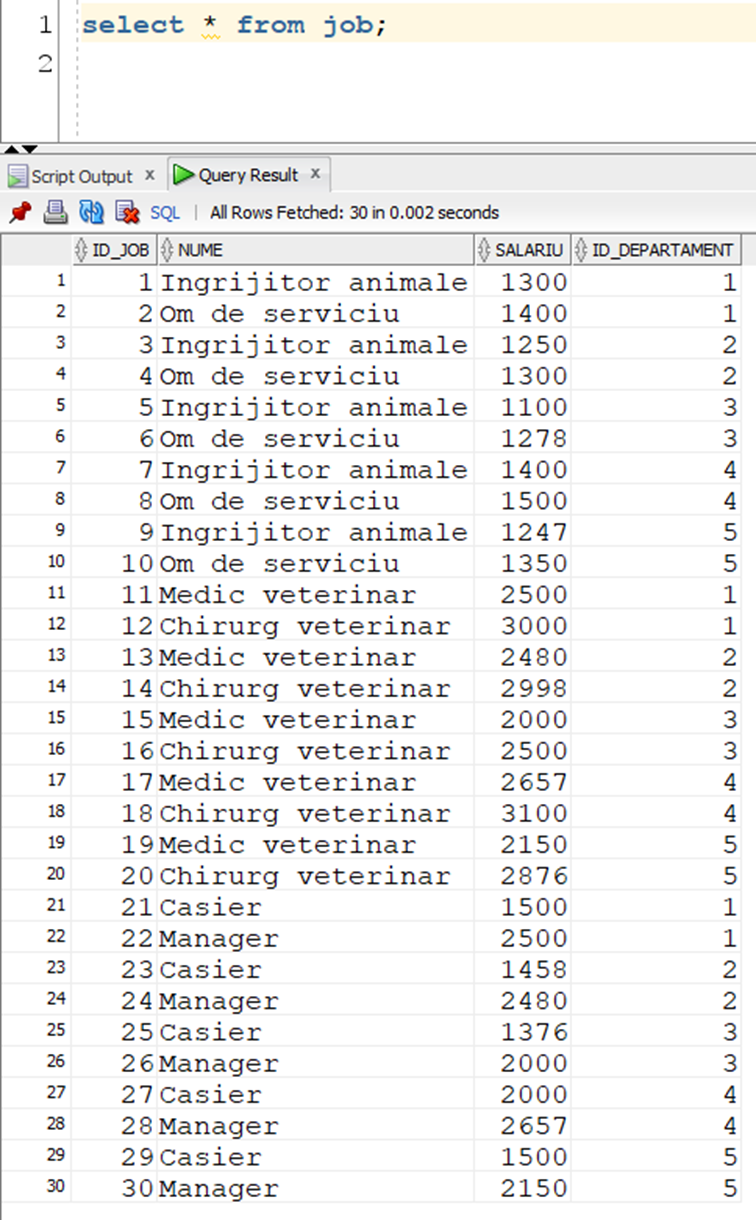




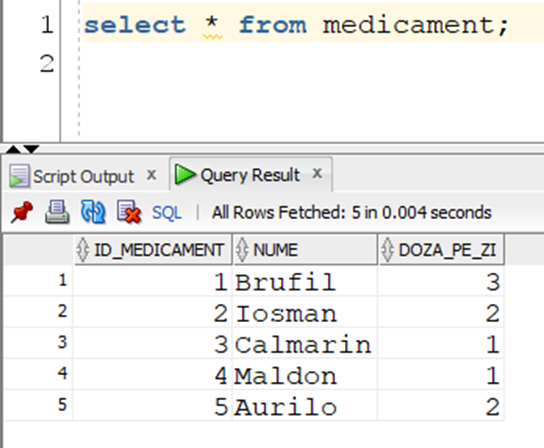


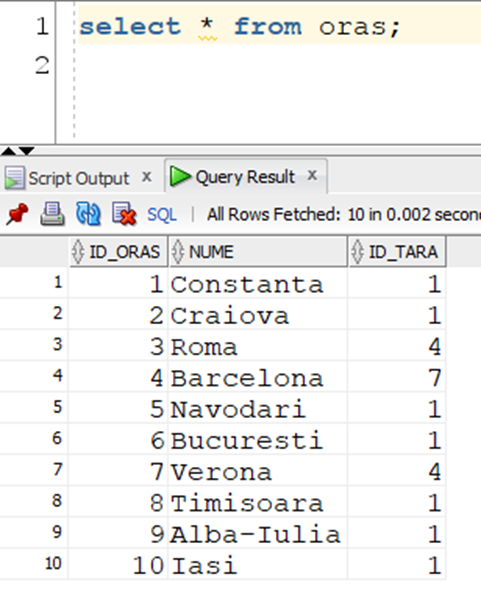


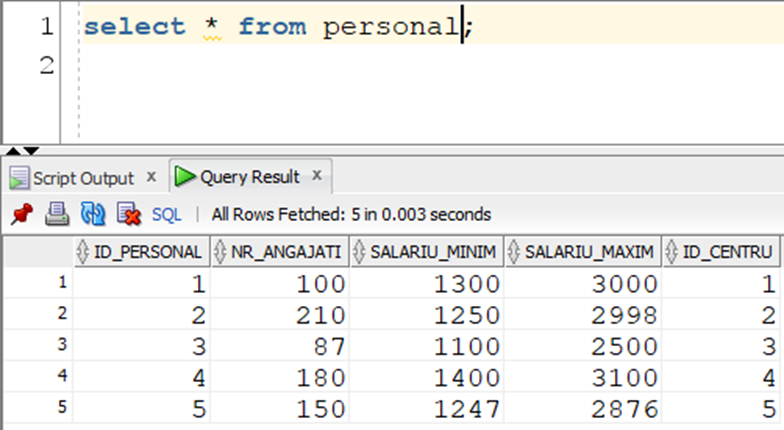


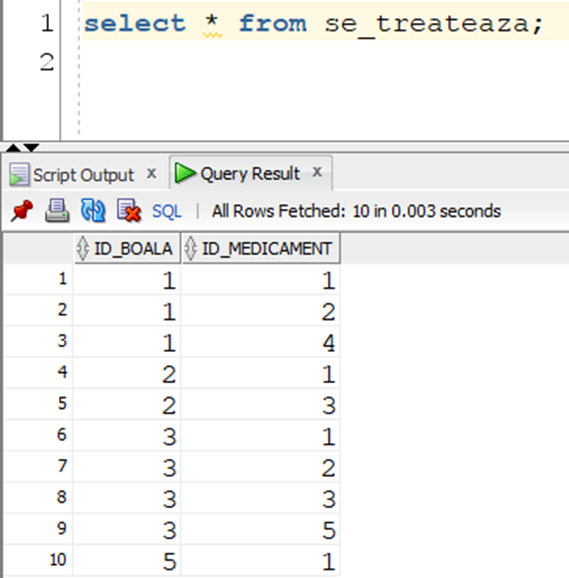


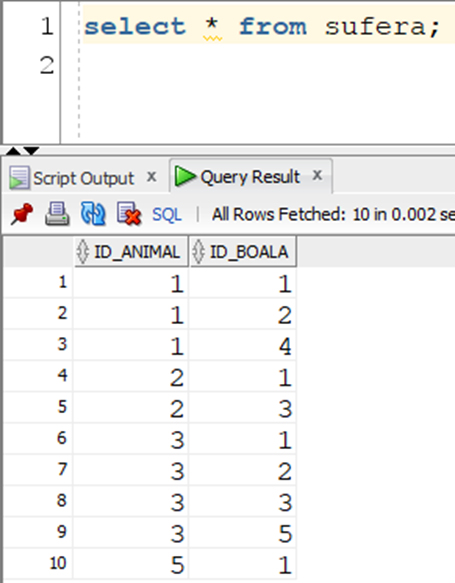


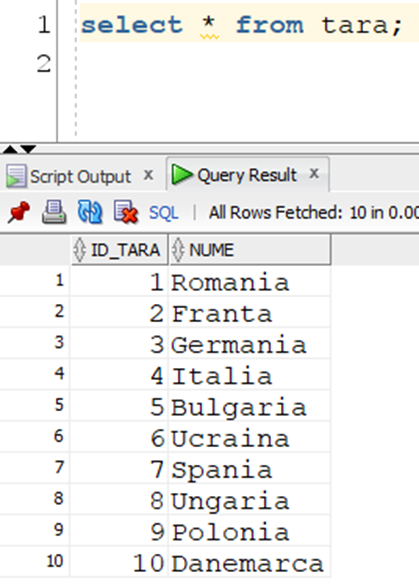


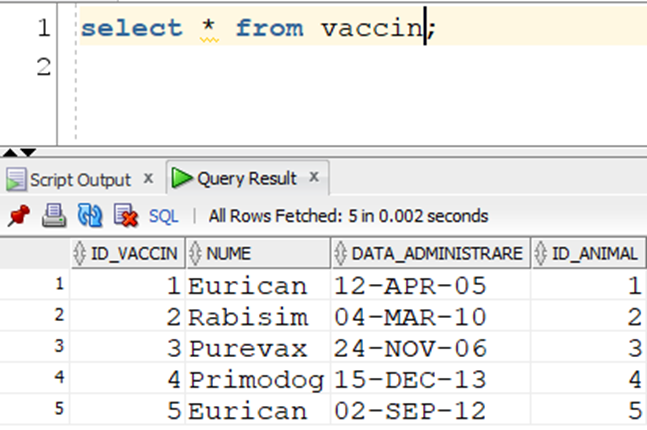












***6. Formulați în limbaj natural o problemă pe care să o rezolvați folosind un subprogram stocat independent care să utilizeze două tipuri de colecție studiate. Apelați subprogramul.***

--Afisati numele animalelor si numele bolii/bolilor de care sufera

--pentru animalele ale caror varsta este mai mica sau egala cu gravitatea bolii.

--Folositi tablouri indexate si tablouri imbricate pentru a rezolva problema data.

create or replace procedure ex6

is

type t1 is table of varchar2(30);

nume\_animale t1 := t1();

type t2 is table of varchar2(30) index by pls\_integer;

nume\_boli t2;

begin

select distinct a.nume

bulk collect into nume\_animale

from animal a, sufera s, boala b

where a.id\_animal = s.id\_animal and s.id\_boala = b.id\_boala and

a.varsta in (select b2.gravitate

from animal a2 join sufera s2 on (a2.id\_animal = s2.id\_animal)

join boala b2 on (s2.id\_boala = b2.id\_boala)

where b2.gravitate <= a.varsta)

group by a.nume, b.nume, b.gravitate;

select distinct b.nume

bulk collect into nume\_boli

from animal a, sufera s, boala b

where a.id\_animal = s.id\_animal and s.id\_boala = b.id\_boala and

a.varsta in (select b2.gravitate

from animal a2 join sufera s2 on (a2.id\_animal = s2.id\_animal)

join boala b2 on (s2.id\_boala = b2.id\_boala)

where b2.gravitate <= a.varsta)

group by a.nume, b.nume, b.gravitate;

for i in nume\_animale.first..nume\_animale.last loop

dbms\_output.put\_line(nume\_animale(i) || ' sufera de:');

for j in nume\_boli.first..nume\_boli.last loop

dbms\_output.put\_line(nume\_boli(j));

end loop;

end loop;

exception

when NO\_DATA\_FOUND then

raise\_application\_error(-20000, 'Nu exista niciun animal cu aceste proprietati');

when TOO\_MANY\_ROWS then

raise\_application\_error(-20001, 'Exista mai multe animale cu aceste proprietati');

when OTHERS then

raise\_application\_error(-20002, 'Alta eroare!');

end;

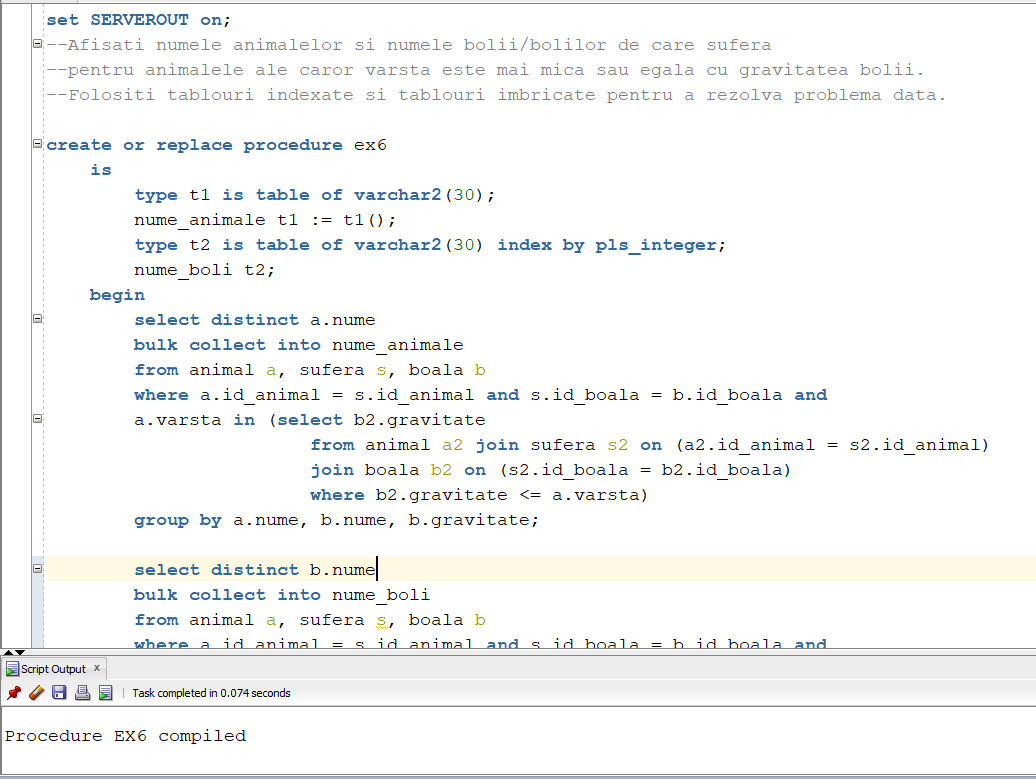
/

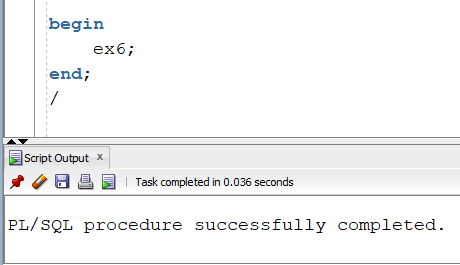
begin

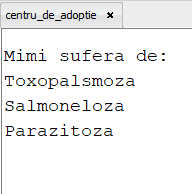
ex6;

end;

/







***7. Formulați în limbaj natural o problemă pe care să o rezolvați folosind un subprogram stocat independent care să utilizeze 2 tipuri de cursoare studiate, unul dintre acestea fiind cursor parametrizat. Apelați subprogramul.***

--In luna august personalul din centrele de adoptie sunt in

--vacanta. Vrem sa stim ce animale au primit vaccinul inaine de

--vacanta, si ce animale au primit vaccinul dupa vacanta, in

--anul trecut, pentru animalele care au mai mult de un numar

--dat de vaccinuri.

--Afisati numele animalului , varsta lui, iar pentru fiecare

--situatie, afisati un text semnificativ. Folositi un cursor

--explicit si un ciclu cursor cu subcereri pentru a rezolva problema.

create or replace procedure ex7

(v\_nr\_vaccinuri carnet\_vaccinuri.nr\_vaccinuri%type)

is

v\_id animal.id\_animal%type;

v\_nume animal.nume%type;

v\_varsta animal.varsta%type;

v\_nume2 animal.nume%type;

ok number := 0;

cursor c is

select id\_animal, nume , varsta

from animal;

begin

open c;

loop

fetch c into v\_id, v\_nume, v\_varsta;

exit when c%notfound;

for j in ( select c.data\_urmatorului\_vaccin data\_v, c.nr\_vaccinuri nr

from carnet\_vaccinuri c

where c.id\_animal = v\_id) loop

if j.nr >v\_nr\_vaccinuri then

ok := 1;

dbms\_output.put\_line(v\_nume || '. Varsta: ' || v\_varsta );

if extract(year from j.data\_v) = extract(year from sysdate) - 1 then

if extract(month from j.data\_v) < 8 then

dbms\_output.put\_line('Vaccinul a fost facut inainte de vacanta');

elsif extract(month from j.data\_v) > 8 then

dbms\_output.put\_line('Vaccinul a fost facut dupa vacanta');

else

dbms\_output.put\_line('Vaccinul trebuia facut in vacanta!');

end if;

end if;

end if;

end loop;

end loop;

if ok = 0 then

raise NO\_DATA\_FOUND;

end if;

exception

when NO\_DATA\_FOUND then

raise\_application\_error(-20000, 'Nu exista animale cu acel numar de vaccinuri sau mai mare');

when TOO\_MANY\_ROWS then

raise\_application\_error(-20001, 'Sunt prea multe animale care indeplinesc criteriul');

when OTHERS then

raise\_application\_error(-20002, 'Alta eroare!');

end;

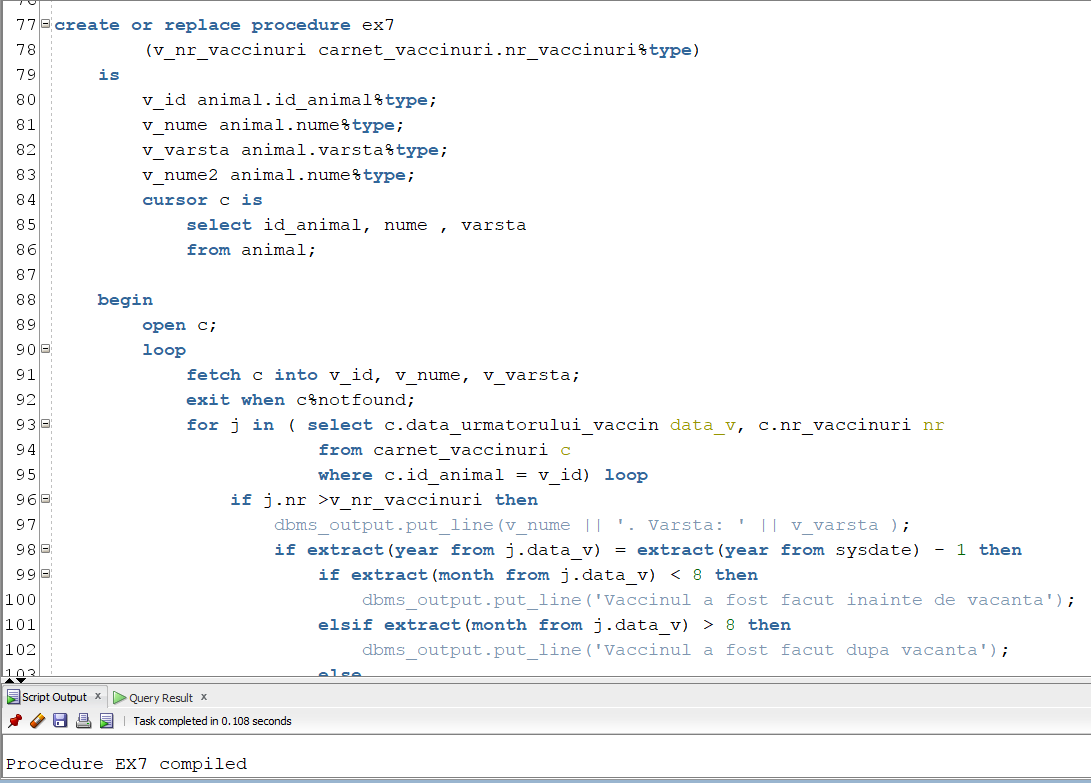
/

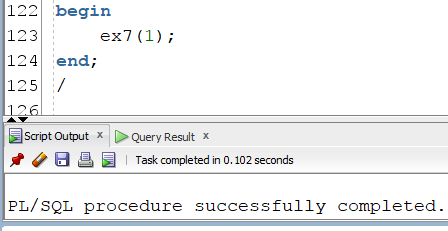
begin

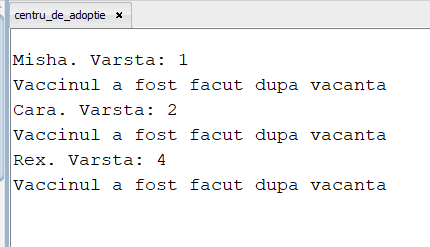
ex7(1);

end;

/







***8. Formulați în limbaj natural o problemă pe care să o rezolvați folosind un subprogram stocat independent de tip funcție care să utilizeze într-o singură comandă SQL 3 dintre tabelele definite. Definiți minim 2 excepții. Apelați subprogramul astfel încât să evidențiați toate cazurile tratate.***

--8

--Pentru un oras dat, afisati numarul de animalute date spre

--adoptie ce sunt bagate in baza noastra de date. Daca orasul

--dat ca parametru nu face parte din Romania, tratati exceptia

--cu un mesaj corespunzator. Daca nu sunt animale date spre adoptie

--in baza noastra de date din orasul dat ca parametru, tratati exceptia

--cu un mesaj corespunzator.

create or replace function ex8

(v\_oras oras.nume%type default 'Craiova')

return number

is

nr\_animale number;

exceptie1 exception;

exceptie2 exception;

v\_tara tara.nume%type;

begin

select t.nume

into v\_tara

from tara t join oras o on (t.id\_tara = o.id\_tara)

where o.nume = v\_oras;

if v\_tara != 'Romania' then

raise exceptie1;

end if;

select count(a.nume)

into nr\_animale

from animal a join centru\_de\_adoptie ca on (a.id\_centru = ca.id\_centru)

join oras o on (ca.id\_oras = o.id\_oras)

where o.nume = v\_oras;

if nr\_animale = 0 then

raise exceptie2;

end if;

return nr\_animale;

exception

when exceptie1 then

raise\_application\_error(-20000,'Orasul transmis ca parametru nu face parte din Romania');

when exceptie2 then

raise\_application\_error(-20001,'Nu exista animale date spre adoptie in baza noastra de date din orasul dat ca parametru.');

end;

/

begin

dbms\_output.put\_line('Este/sunt ' || ex8('Navodari') || ' animalut/animalute de adoptat!');

end;

/

begin

dbms\_output.put\_line('Este/sunt ' || ex8('Roma') || ' animalut/animalute de adoptat!');

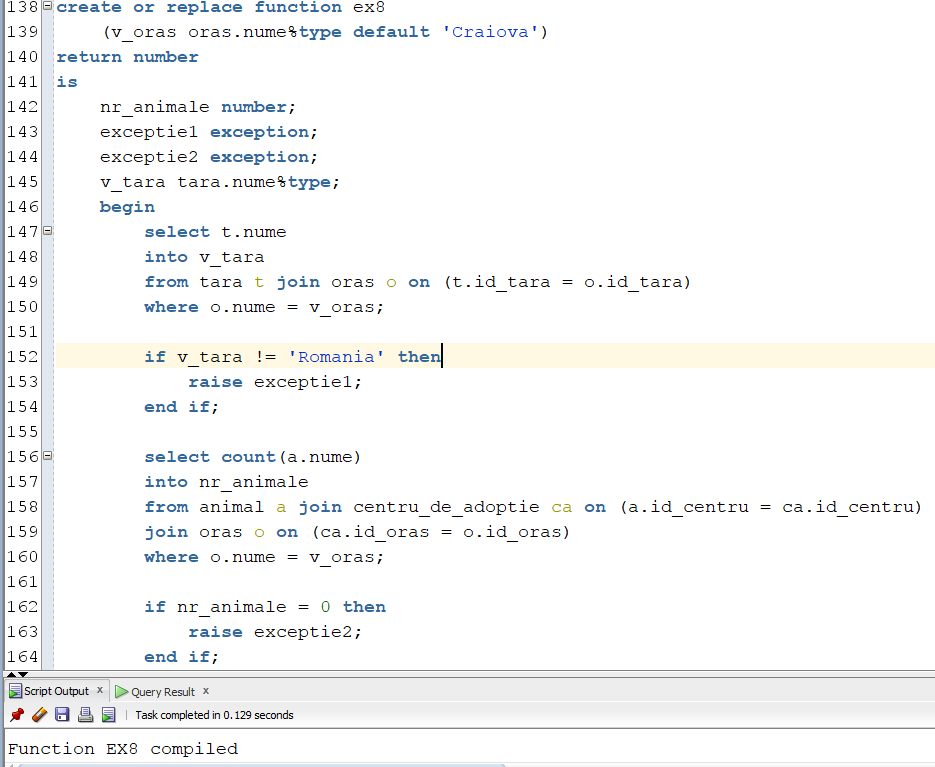
end;

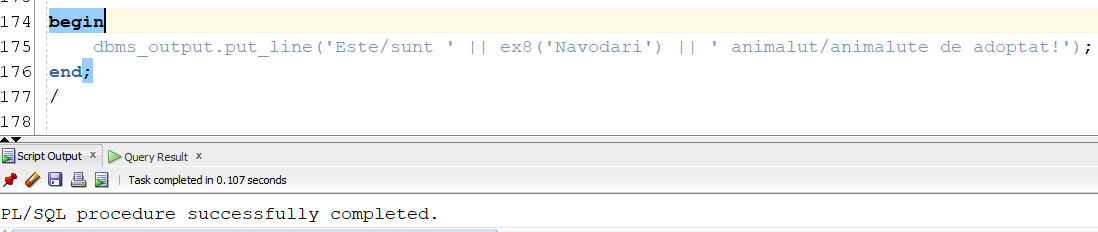
/

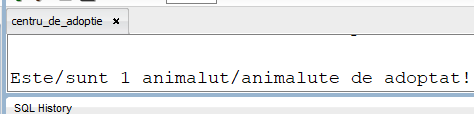
begin

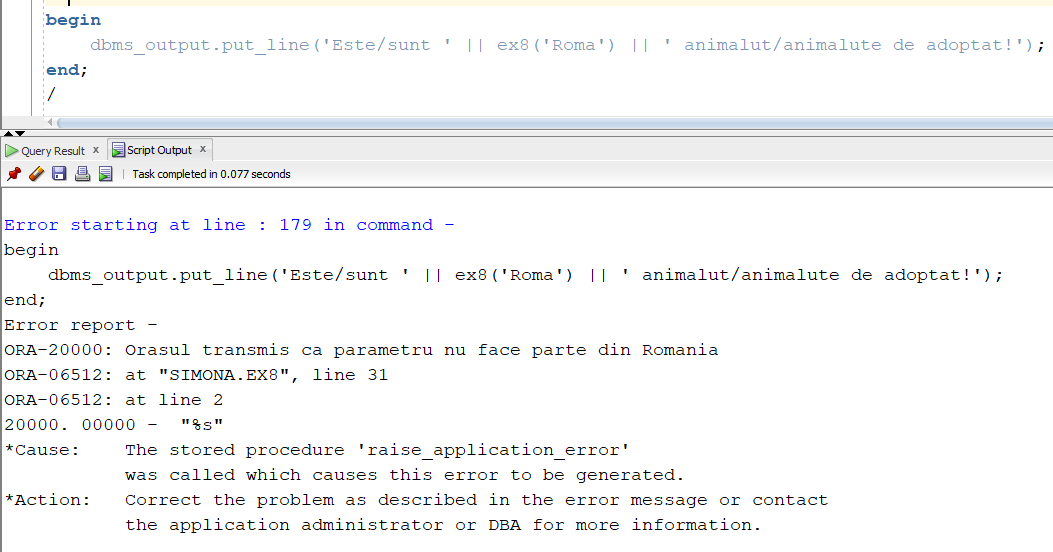
dbms\_output.put\_line('Este/sunt ' ||ex8('Constanta')||' animalut/animalute de adoptat!');

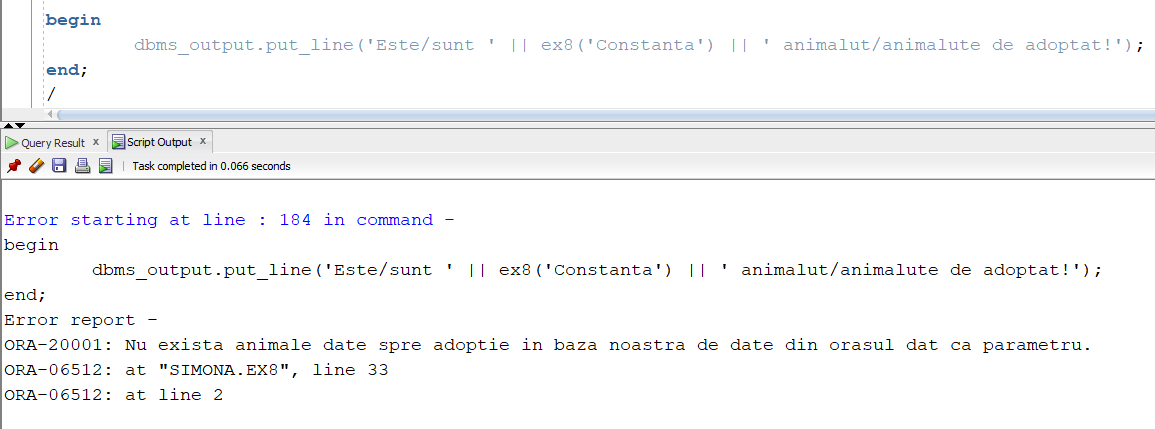
end;

/









***9. Formulați în limbaj natural o problemă pe care să o rezolvați folosind un subprogram stocat independent de tip procedură care să utilizeze într-o singură comandă SQL 5 dintre tabelele definite. Tratați toate excepțiile care pot apărea, incluzând excepțiile NO\_DATA\_FOUND și TOO\_MANY\_ROWS. Apelați subprogramul astfel încât să evidențiați toate cazurile tratate.***

--Pentru un centru de adoptie dat, realizati un top 3 angajati ordonati dupa salariu

--al departamentului cu cei mai multi angajati. Tratati cazurile ce pot duce

--la erori.

create or replace procedure ex9

(v\_centru centru\_de\_adoptie.nume%type)

is

type tab\_centre is table of centru\_de\_adoptie.nume%type;

var\_centre tab\_centre;

ok number :=0;

top number :=0;

var\_departament departament.id\_departament%type;

var\_max number;

var\_nume\_dep departament.nume%type;

exceptie exception;

begin

select nume

bulk collect into var\_centre

from centru\_de\_adoptie;

for i in var\_centre.first..var\_centre.last loop

if v\_centru = var\_centre(i) then

ok := 1;

exit;

end if;

end loop;

if ok = 0 then

raise exceptie;

end if;

select max(count(a.id\_angajat))

into var\_max

from departament d left outer join job j on (d.id\_departament = j.id\_departament)

left outer join angajat a on (j.id\_job = a.id\_job) , personal p, centru\_de\_adoptie ca

where d.id\_personal = p.id\_personal and ca.id\_centru = p.id\_centru and ca.nume = v\_centru

group by d.nume;

select distinct d.id\_departament , d.nume

into var\_departament, var\_nume\_dep

from departament d left outer join job j on (d.id\_departament = j.id\_departament)

left outer join angajat a on (j.id\_job = a.id\_job) , personal p, centru\_de\_adoptie ca

where d.id\_personal = p.id\_personal and ca.id\_centru = p.id\_centru and ca.nume = v\_centru and

(

select count(a1.id\_angajat)

from departament d1 left outer join job j1 on (d1.id\_departament = j1.id\_departament)

left outer join angajat a1 on (j1.id\_job = a1.id\_job) , personal p1, centru\_de\_adoptie ca1

where d1.id\_personal = p1.id\_personal and ca1.id\_centru = p1.id\_centru and ca1.nume = v\_centru and

d1.id\_departament = d.id\_departament

group by d1.nume) = var\_max;

dbms\_output.put\_line('Departamentul: ' || var\_nume\_dep);

top := 0;

for j in (select a.nume nume, a.prenume prenume, a.salariu salariu

from centru\_de\_adoptie ca join personal p on (ca.id\_centru = p.id\_centru)

join departament d on (p.id\_personal = d.id\_personal)

join job jo on (d.id\_departament = jo.id\_departament)

join angajat a on (jo.id\_job = a.id\_job)

where ca.nume = v\_centru and d.id\_departament = var\_departament

order by a.salariu desc) loop

top := top + 1;

dbms\_output.put\_line(top || '. ' || j.nume || ' ' || j.prenume || ': ' || j.salariu);

exit when top = 3;

end loop;

exception

when exceptie then

raise\_application\_error(-20000, 'Acest centru de adoptie nu este introdus in baza noastra de date');

when NO\_DATA\_FOUND then

raise\_application\_error(-20001, 'In centrul de adoptie introdus inca nu s-au angajat oameni');

when TOO\_MANY\_ROWS then

raise\_application\_error(-20002, 'Prea multi angajati');

when OTHERS then

raise\_application\_error(-20003, 'Alta eroare!');

end;

/

begin

ex9('Animalutele fericite');

end;

/

begin

ex9('Animalia');

end;

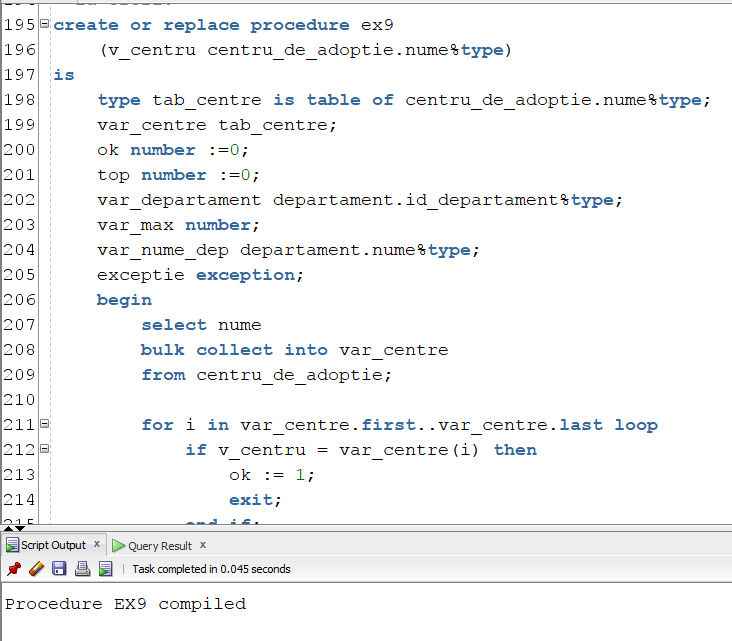
/

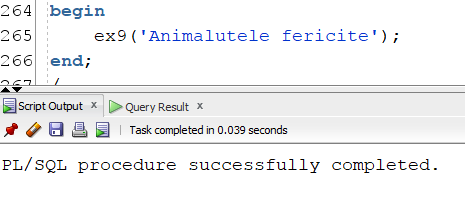
begin

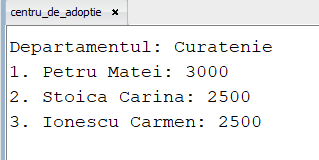
ex9('abs');

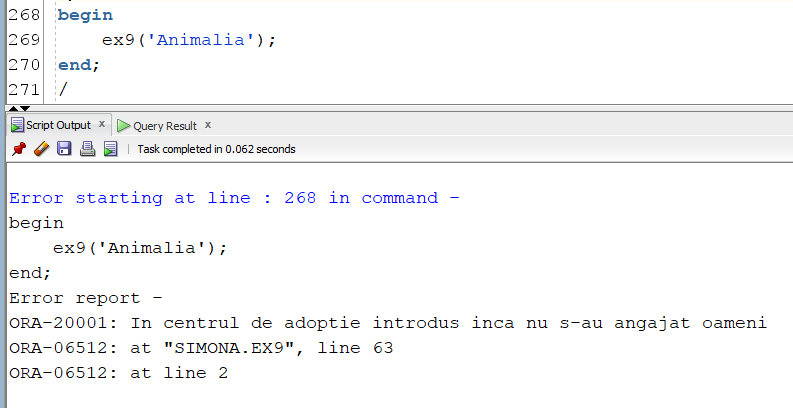
end;

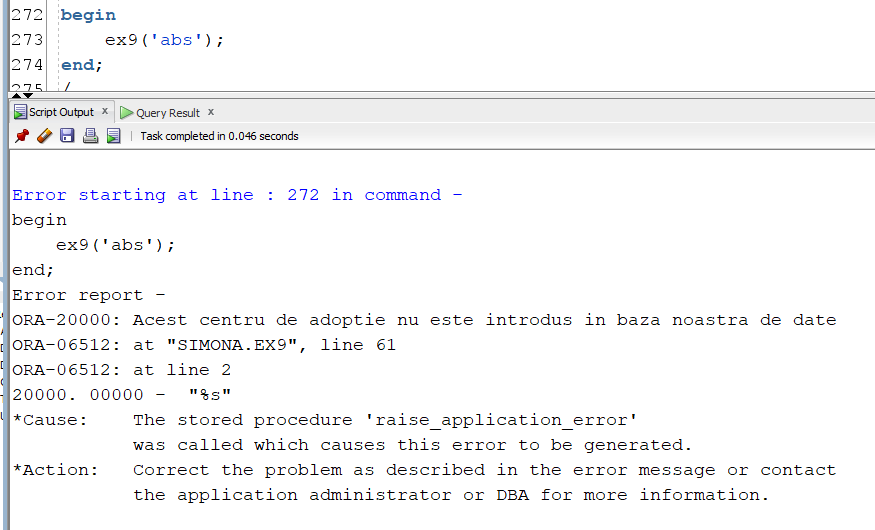
/











***10. Definiți un trigger de tip LMD la nivel de comandă. Declanșați trigger-ul.***

--Pentru a putea adauga un alt centru de adoptie in baza de date,

--trebuie ca toate centrele de adoptie sa isi atinga capacitatea

--maxima, aceasta fiind de 10 angajati. Daca cel putin un centru

--nu indeplineste aceasta conditie, atunci nu se poate adauga alt centru.

--Creeati un trigger ce previne acest lucru.

create or replace trigger ex10

before insert on centru\_de\_adoptie

declare

type tab\_nr is table of number;

nr\_angajati tab\_nr;

begin

select count(a.nume)

bulk collect into nr\_angajati

from centru\_de\_adoptie ca join personal p on (ca.id\_centru = p.id\_centru)

join departament d on (p.id\_personal = d.id\_personal)

join job j on (d.id\_departament = j.id\_departament)

join angajat a on (j.id\_job = a.id\_job)

group by ca.id\_centru;

for i in nr\_angajati.first..nr\_angajati.last loop

if nr\_angajati(i) < 10 then

raise\_application\_error(-20001, 'Centrele nu si-au atins capacitatea maxima de angajati');

end if;

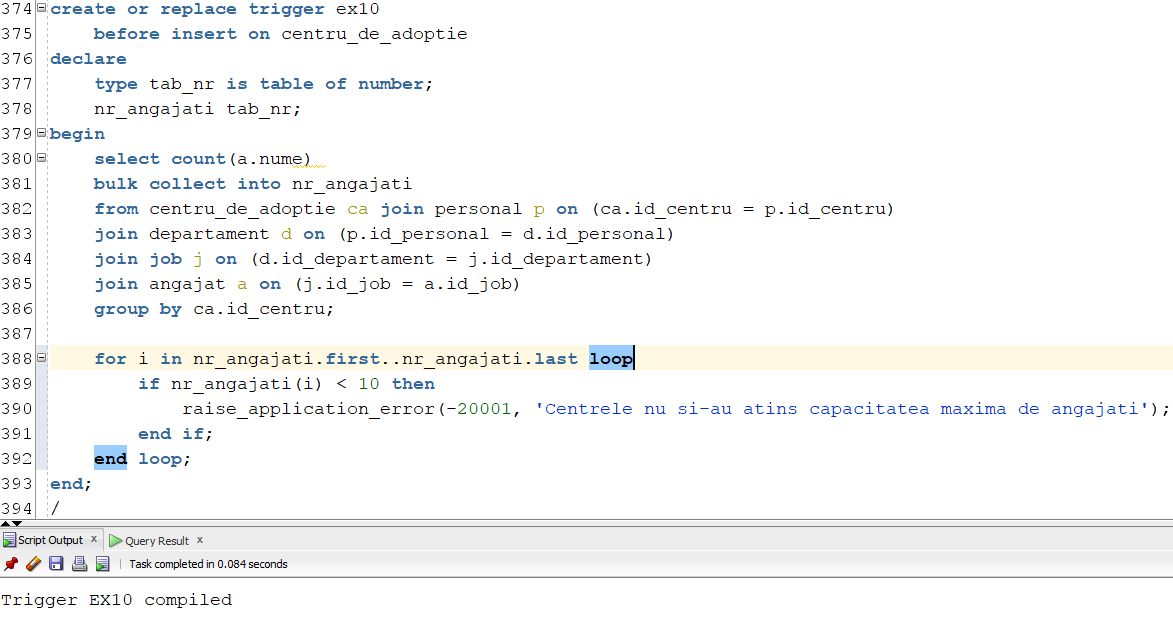
end loop;

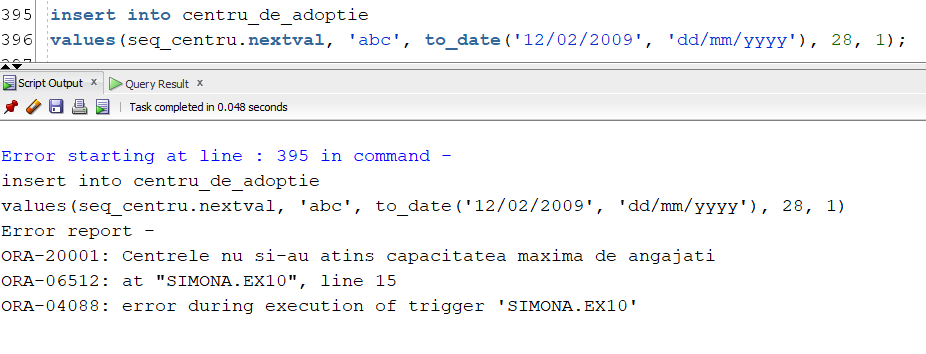
end;

/

insert into centru\_de\_adoptie

values(seq\_centru.nextval, 'abc', to\_date('12/02/2009', 'dd/mm/yyyy'), 28, 1);





***11. Definiți un trigger de tip LMD la nivel de linie. Declanșați trigger-ul.***

--La fiecare adaugare sau stergere a unui animal in baza noastra

--de date, au de suferit atat tabelul ce retine animalele, cat si altele.

--La fiecare stergere a unui animal, trebuie reactualizat numarul de animale

--adoptate din centrul de adoptie in care a fost animalul. La fiecare

--introducere a unui nou animal, trebuie reactualizat numarul de animale

--adoptate din centru de adoptie in care ajunge animalul.

--Orice nou animal primeste ca hrana mancare denumita 'Chonk',

--primeste vaccinul 'Eurican', si i se face un carnet de vaccinuri.

--Realizati un trigger ce actualizeaza automat aceste date.

create or replace trigger ex11

after delete or insert on animal

for each row

declare

id\_v vaccin.id\_vaccin%type;

begin

if deleting then

update centru\_de\_adoptie

set nr\_animale\_adoptate = nr\_animale\_adoptate - 1

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

else

update centru\_de\_adoptie

set nr\_animale\_adoptate = nr\_animale\_adoptate + 1

where id\_centru = :new.id\_centru;

insert into mananca

values(:new.id\_animal, 1);

id\_v := seq\_vaccin.nextval;

insert into vaccin

values(id\_v, 'Eurican', to\_date(to\_char(sysdate, 'dd-mm-yyyy'), 'dd-mm-yyyy'), :new.id\_animal);

insert into carnet\_vaccinuri

values(seq\_carnet.nextval, :new.id\_animal, id\_v, 1, to\_date(to\_char(sysdate + 365, 'dd-mm-yyyy'), 'dd-mm-yyyy'));

end if;

end;

/

select \* from centru\_de\_adoptie;

delete from animal

where id\_animal = 1;

select \* from centru\_de\_adoptie;

insert into animal

values(seq\_animal.nextval, 'nu stim', 5, 'Coco', 3, 'nu are');

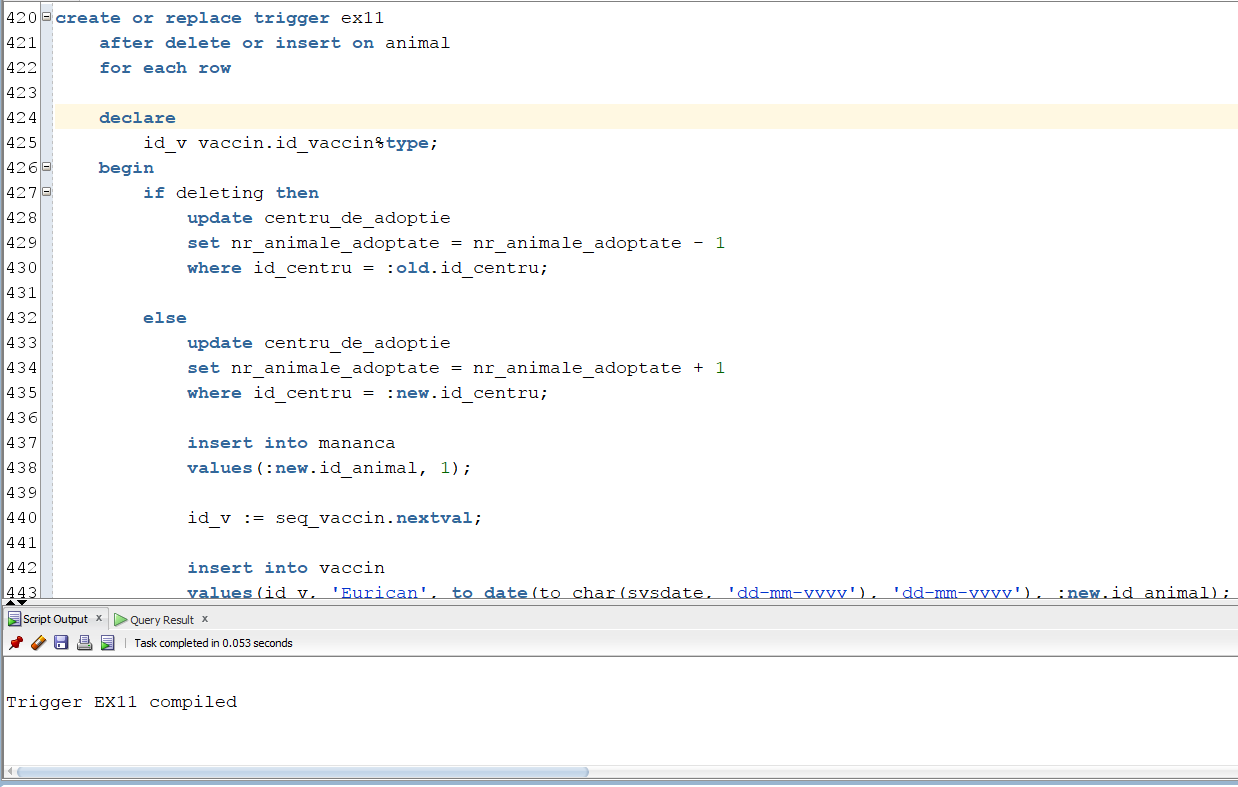
select \* from centru\_de\_adoptie;

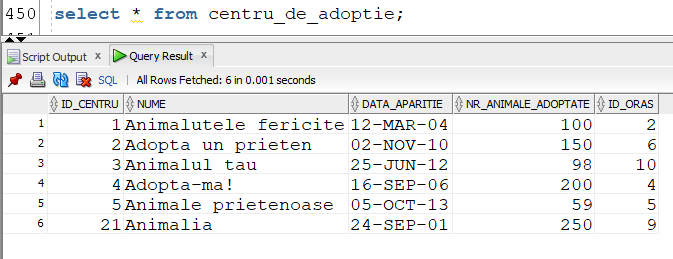
select \* from animal;

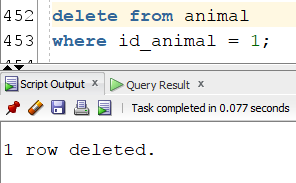
select \* from carnet\_vaccinuri;

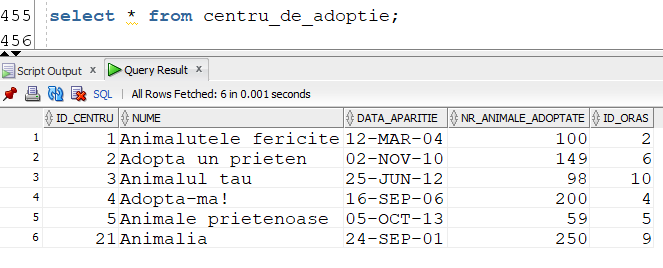
select \* from mananca;

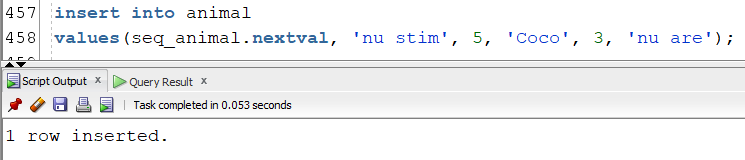
select \* from vaccin;

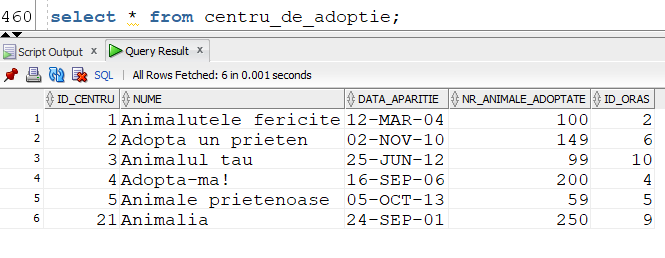


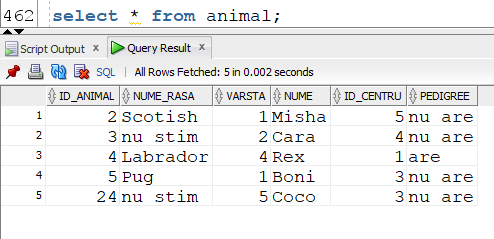


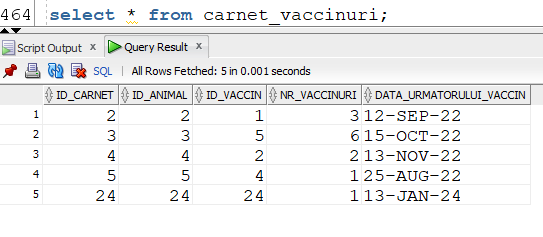




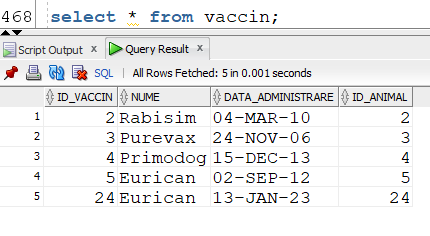












***12. Definiți un trigger de tip LDD. Declanșați trigger-ul.***

create table utilizatori

(

user\_name varchar2(30),

comanda varchar2(30),

nume\_tabel varchar2(4000),

data\_executiei date

);

create or replace trigger ex12

before create or alter or drop on schema

begin

insert into utilizatori

values (sys.login\_user, sys.sysevent, ora\_dict\_obj\_name, sysdate);

end;

/

create table test

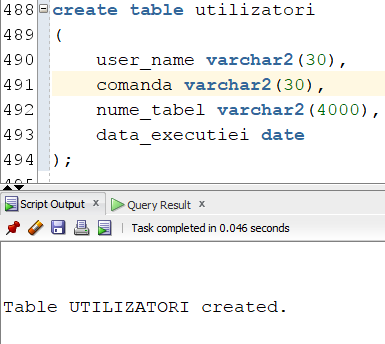
(

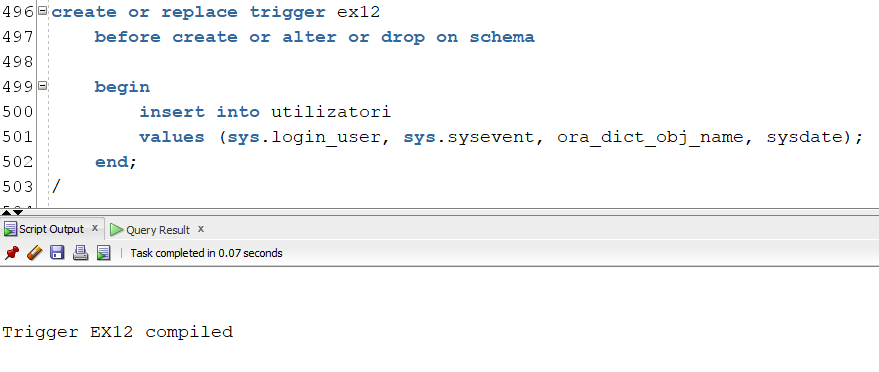
nume varchar2(30)

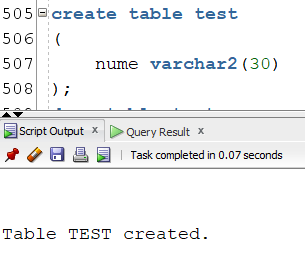
);

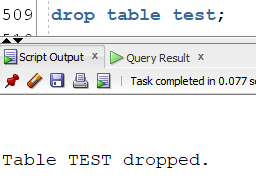
drop table test;

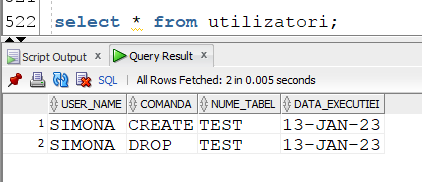
select \* from utilizatori;











***13. Definiți un pachet care să conțină toate obiectele definite în cadrul proiectului.***

create or replace package ex13 as

procedure ex6;

procedure ex7 (v\_nr\_vaccinuri carnet\_vaccinuri.nr\_vaccinuri%type);

function ex8 (v\_oras oras.nume%type default 'Craiova')

return number;

procedure ex9 (v\_centru centru\_de\_adoptie.nume%type);

end ex13;

/

create or replace package body ex13 as

procedure ex6

is

type t1 is table of varchar2(30);

nume\_animale t1 := t1();

type t2 is table of varchar2(30) index by pls\_integer;

nume\_boli t2;

begin

select distinct a.nume

bulk collect into nume\_animale

from animal a, sufera s, boala b

where a.id\_animal = s.id\_animal and s.id\_boala = b.id\_boala and

a.varsta in (select b2.gravitate

from animal a2 join sufera s2 on (a2.id\_animal = s2.id\_animal)

join boala b2 on (s2.id\_boala = b2.id\_boala)

where b2.gravitate <= a.varsta)

group by a.nume, b.nume, b.gravitate;

select distinct b.nume

bulk collect into nume\_boli

from animal a, sufera s, boala b

where a.id\_animal = s.id\_animal and s.id\_boala = b.id\_boala and

a.varsta in (select b2.gravitate

from animal a2 join sufera s2 on (a2.id\_animal = s2.id\_animal)

join boala b2 on (s2.id\_boala = b2.id\_boala)

where b2.gravitate <= a.varsta)

group by a.nume, b.nume, b.gravitate;

for i in nume\_animale.first..nume\_animale.last loop

dbms\_output.put\_line(nume\_animale(i) || ' sufera de:');

for j in nume\_boli.first..nume\_boli.last loop

dbms\_output.put\_line(nume\_boli(j));

end loop;

end loop;

exception

when NO\_DATA\_FOUND then

raise\_application\_error(-20000, 'Nu exista niciun animal cu aceste proprietati');

when TOO\_MANY\_ROWS then

raise\_application\_error(-20001, 'Exista mai multe animale cu aceste proprietati');

when OTHERS then

raise\_application\_error(-20002, 'Alta eroare!');

end;

procedure ex7

(v\_nr\_vaccinuri carnet\_vaccinuri.nr\_vaccinuri%type)

is

v\_id animal.id\_animal%type;

v\_nume animal.nume%type;

v\_varsta animal.varsta%type;

v\_nume2 animal.nume%type;

cursor c is

select id\_animal, nume , varsta

from animal;

begin

open c;

loop

fetch c into v\_id, v\_nume, v\_varsta;

exit when c%notfound;

for j in ( select c.data\_urmatorului\_vaccin data\_v, c.nr\_vaccinuri nr

from carnet\_vaccinuri c

where c.id\_animal = v\_id) loop

if j.nr >v\_nr\_vaccinuri then

dbms\_output.put\_line(v\_nume || '. Varsta: ' || v\_varsta );

if extract(year from j.data\_v) = extract(year from sysdate) - 1 then

if extract(month from j.data\_v) < 8 then

dbms\_output.put\_line('Vaccinul a fost facut inainte de vacanta');

elsif extract(month from j.data\_v) > 8 then

dbms\_output.put\_line('Vaccinul a fost facut dupa vacanta');

else

dbms\_output.put\_line('Vaccinul trebuia facut in vacanta!');

end if;

end if;

end if;

end loop;

end loop;

exception

when NO\_DATA\_FOUND then

raise\_application\_error(-20000, 'Nu exista animale cu acel numar de vaccinuri sau mai mare');

when TOO\_MANY\_ROWS then

raise\_application\_error(-20001, 'Sunt prea multe animale care indeplinesc criteriul');

when OTHERS then

raise\_application\_error(-20002, 'Alta eroare!');

end;

function ex8

(v\_oras oras.nume%type default 'Craiova')

return number

is

nr\_animale number;

exceptie1 exception;

exceptie2 exception;

v\_tara tara.nume%type;

begin

select t.nume

into v\_tara

from tara t join oras o on (t.id\_tara = o.id\_tara)

where o.nume = v\_oras;

if v\_tara != 'Romania' then

raise exceptie1;

end if;

select count(a.nume)

into nr\_animale

from animal a join centru\_de\_adoptie ca on (a.id\_centru = ca.id\_centru)

join oras o on (ca.id\_oras = o.id\_oras)

where o.nume = v\_oras;

if nr\_animale = 0 then

raise exceptie2;

end if;

return nr\_animale;

exception

when exceptie1 then

raise\_application\_error(-20000,'Orasul transmis ca parametru nu face parte din Romania');

when exceptie2 then

raise\_application\_error(-20001,'Nu exista animale date spre adoptie in baza noastra de date din orasul dat ca parametru.');

end;

procedure ex9

(v\_centru centru\_de\_adoptie.nume%type)

is

type tab\_centre is table of centru\_de\_adoptie.nume%type;

var\_centre tab\_centre;

ok number :=0;

top number :=0;

var\_departament departament.id\_departament%type;

var\_max number;

var\_nume\_dep departament.nume%type;

exceptie exception;

begin

select nume

bulk collect into var\_centre

from centru\_de\_adoptie;

for i in var\_centre.first..var\_centre.last loop

if v\_centru = var\_centre(i) then

ok := 1;

exit;

end if;

end loop;

if ok = 0 then

raise exceptie;

end if;

select max(count(a.id\_angajat))

into var\_max

from departament d left outer join job j on (d.id\_departament = j.id\_departament)

left outer join angajat a on (j.id\_job = a.id\_job) , personal p, centru\_de\_adoptie ca

where d.id\_personal = p.id\_personal and ca.id\_centru = p.id\_centru and ca.nume = v\_centru

group by d.nume;

select distinct d.id\_departament , d.nume

into var\_departament, var\_nume\_dep

from departament d left outer join job j on (d.id\_departament = j.id\_departament)

left outer join angajat a on (j.id\_job = a.id\_job) , personal p, centru\_de\_adoptie ca

where d.id\_personal = p.id\_personal and ca.id\_centru = p.id\_centru and ca.nume = v\_centru and

(

select count(a1.id\_angajat)

from departament d1 left outer join job j1 on (d1.id\_departament = j1.id\_departament)

left outer join angajat a1 on (j1.id\_job = a1.id\_job) , personal p1, centru\_de\_adoptie ca1

where d1.id\_personal = p1.id\_personal and ca1.id\_centru = p1.id\_centru and ca1.nume = v\_centru and

d1.id\_departament = d.id\_departament

group by d1.nume) = var\_max;

dbms\_output.put\_line('Departamentul: ' || var\_nume\_dep);

top := 0;

for j in (select a.nume nume, a.prenume prenume, a.salariu salariu

from centru\_de\_adoptie ca join personal p on (ca.id\_centru = p.id\_centru)

join departament d on (p.id\_personal = d.id\_personal)

join job jo on (d.id\_departament = jo.id\_departament)

join angajat a on (jo.id\_job = a.id\_job)

where ca.nume = v\_centru and d.id\_departament = var\_departament

order by a.salariu desc) loop

top := top + 1;

dbms\_output.put\_line(top || '. ' || j.nume || ' ' || j.prenume || ': ' || j.salariu);

exit when top = 3;

end loop;

exception

when exceptie then

raise\_application\_error(-20000, 'Acest centru de adoptie nu este introdus in baza noastra de date');

when NO\_DATA\_FOUND then

raise\_application\_error(-20001, 'In centrul de adoptie introdus inca nu s-au angajat oameni');

when TOO\_MANY\_ROWS then

raise\_application\_error(-20002, 'Prea multi angajati');

when OTHERS then

raise\_application\_error(-20003, 'Alta eroare!');

end;

end ex13;

/

begin

ex13.ex6;

dbms\_output.new\_line();

ex13.ex7(1);

dbms\_output.new\_line();

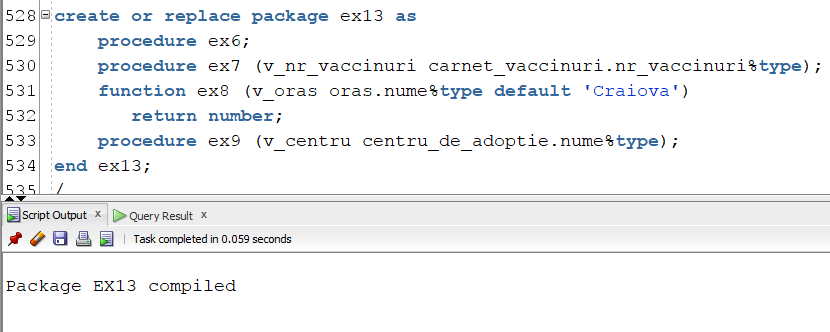
dbms\_output.put\_line('Este/sunt ' || ex8('Navodari') || ' animalut/animalute de adoptat!');

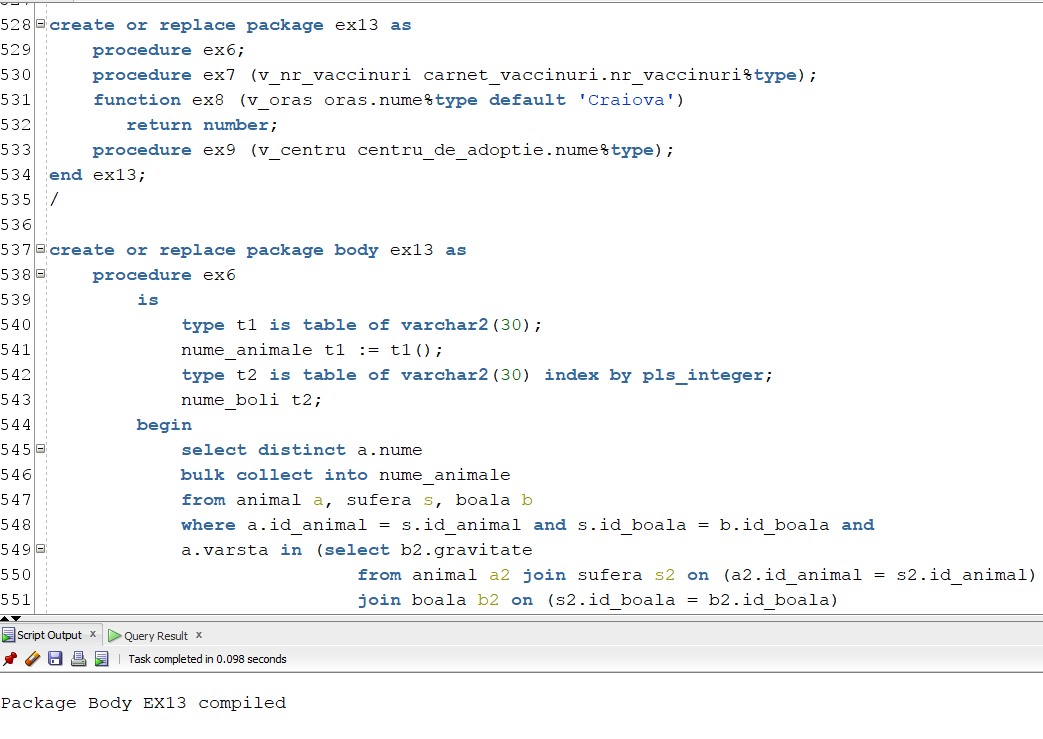
dbms\_output.new\_line();

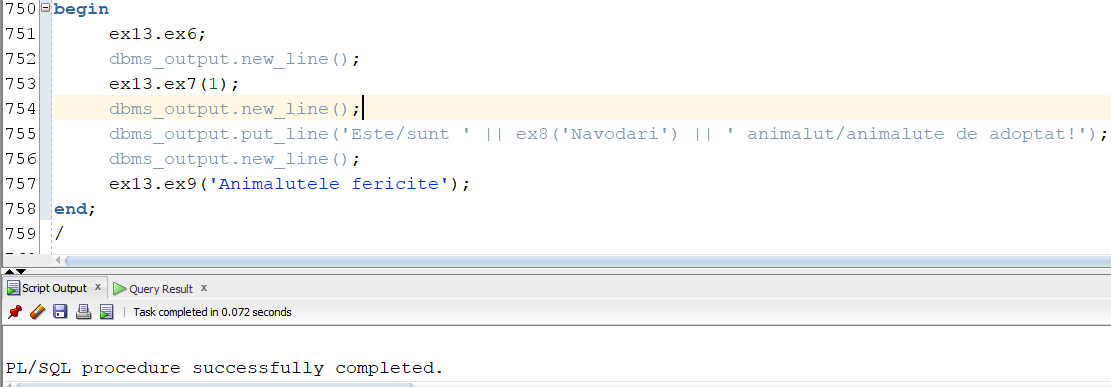
ex13.ex9('Animalutele fericite');

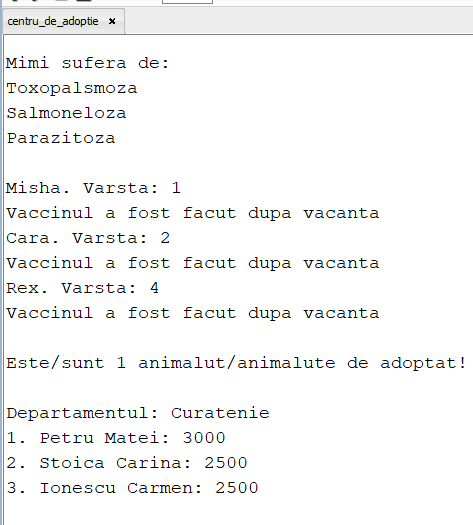
end;

/









14. Definiți un pachet care să includă tipuri de date complexe și obiecte necesare unui flux de acțiuni

integrate, specifice bazei de date definite (minim 2 tipuri de date, minim 2 funcții, minim 2 proceduri).

create or replace package ex14 as

type tab\_animal is varray(100) of animal.nume%type;

type tab\_hrana is table of hrana\_animale.nume%type;

type tab\_medicament is table of medicament.nume%type;

type tab\_nr is table of number;

procedure p1 (v\_centru centru\_de\_adoptie.nume%type);

procedure p2 (v\_animal animal.nume%type);

procedure p3 (v\_boala boala.nume%type);

function f1 (v\_centru centru\_de\_adoptie.nume%type)

return number;

function f2 (v\_id centru\_de\_adoptie.id\_centru%type)

return number;

-- function f3 (v\_animal animal.nume%type)

-- return number;

end ex14;

/

create or replace package body ex14 as

var\_animal tab\_animal := tab\_animal();

var\_hrana tab\_hrana := tab\_hrana();

var\_medicament tab\_medicament := tab\_medicament();

var\_nr tab\_nr := tab\_nr();

procedure p1(v\_centru centru\_de\_adoptie.nume%type) is

begin

select a.nume

bulk collect into var\_animal

from centru\_de\_adoptie ca, animal a

where ca.id\_centru = a.id\_centru and ca.nume = v\_centru;

dbms\_output.put\_line('Animalele din centrul de adoptie ' || v\_centru || ' sunt urmatoarele: ');

for i in var\_animal.first..var\_animal.last loop

dbms\_output.put\_line(var\_animal(i));

end loop;

exception

when no\_data\_found then

raise\_application\_error(-20001, 'Niciun animal gasit in acest centru');

end p1;

procedure p2(v\_animal animal.nume%type) is

begin

select h.nume

bulk collect into var\_hrana

from animal a, mananca m, hrana\_animale h

where a.id\_animal = m.id\_animal and m.id\_hrana = h.id\_hrana and a.nume = v\_animal;

dbms\_output.put\_line(v\_animal || ' mananca urmatoarele: ');

for i in var\_hrana.first..var\_hrana.last loop

dbms\_output.put\_line(var\_hrana(i));

end loop;

exception

when no\_data\_found then

raise\_application\_error(-20001, 'Nu s-a introdus inca nimic in legatura cu mancarea acestui animal');

end p2;

procedure p3(v\_boala boala.nume%type) is

begin

select m.nume

bulk collect into var\_medicament

from boala b, se\_treateaza t, medicament m

where b.id\_boala = t.id\_boala and t.id\_medicament = m.id\_medicament and b.nume = v\_boala;

dbms\_output.put\_line(v\_boala || 'se treateaza cu urmatoarele medicamente:');

for i in var\_medicament.first..var\_medicament.last loop

dbms\_output.put\_line(var\_medicament(i));

end loop;

exception

when no\_data\_found then

raise\_application\_error(-20001, 'Nu s-a gasit niciun medicament ce poate trata aceasta boala');

end p3;

function f1 (v\_centru centru\_de\_adoptie.nume%type)

return number is

buget number := 0;

begin

select a.salariu

bulk collect into var\_nr

from centru\_de\_adoptie ca, personal p, departament d, job j, angajat a

where ca.id\_centru = p.id\_centru and p.id\_personal = d.id\_personal and

d.id\_departament = j.id\_departament and j.id\_job = a.id\_job and

ca.nume = v\_centru;

for i in var\_nr.first..var\_nr.last loop

buget := buget + var\_nr(i);

end loop;

return buget;

exception

when no\_data\_found then

raise\_application\_error(-20001, 'Nu sunt angajati in acest centru');

end f1;

function f2 (v\_id centru\_de\_adoptie.id\_centru%type)

return number is

nr number(2) :=0;

begin

select count(\*)

into nr

from animal

where id\_centru = v\_id;

return nr;

end f2;

end ex14;

/

begin

ex14.p1('Animalutele fericite');

end;

/

begin

ex14.p2('Misha');

end;

/

begin

ex14.p3('Parazitoza');

end;

/

begin

dbms\_output.put\_line('Bugetul este de ' || ex14.f1('Animalutele fericite') || ' lei');

end;

/

begin

dbms\_output.put\_line( ex14.f2(1) );

end;

/

