**PROIECT SGBD**

**CLINICĂ VETERINARĂ**

Student: Staicu Octavian-Florin

Grupa 231

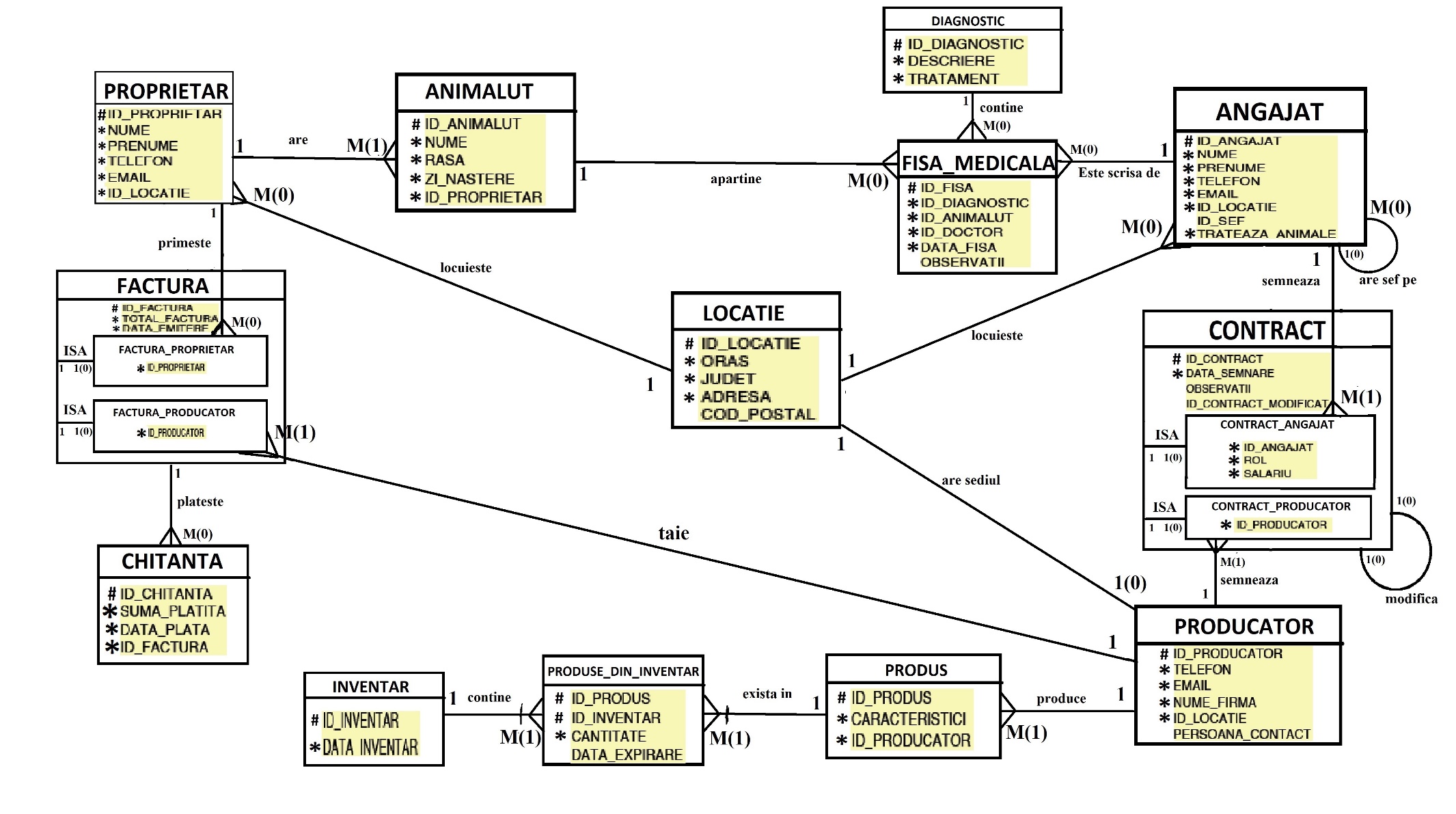
***Clinică veterinară***

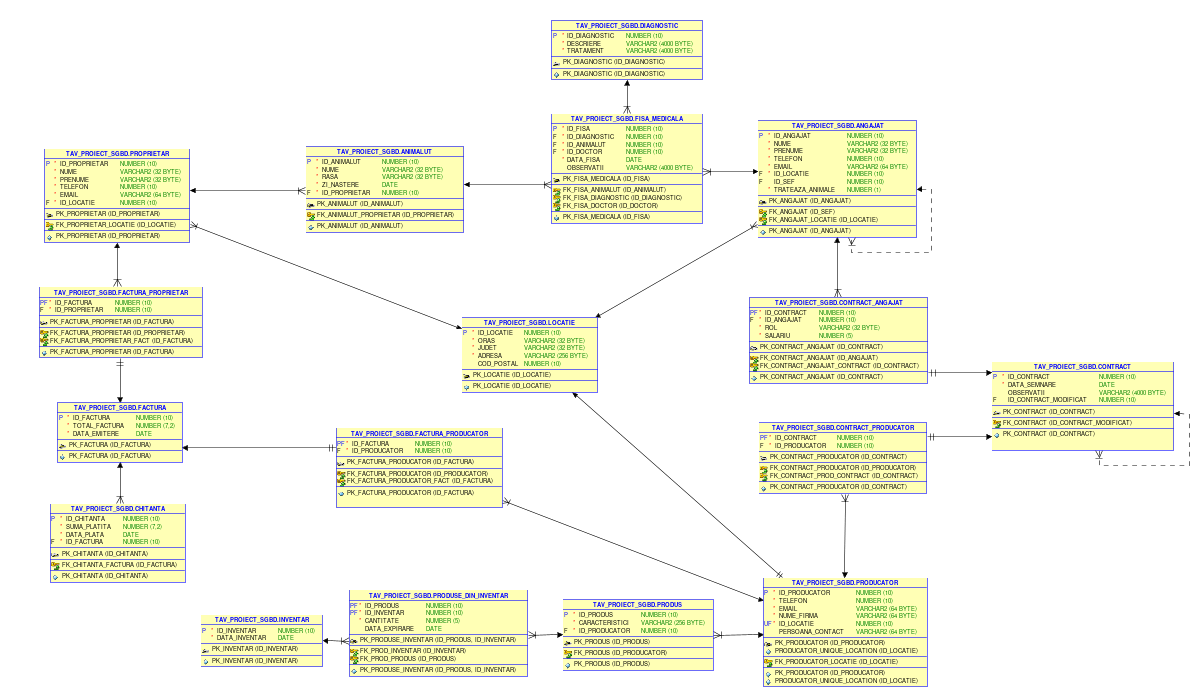
Acest proiect se adresează unei clinici veterinare

# Descrierea Regulilor de Bussines

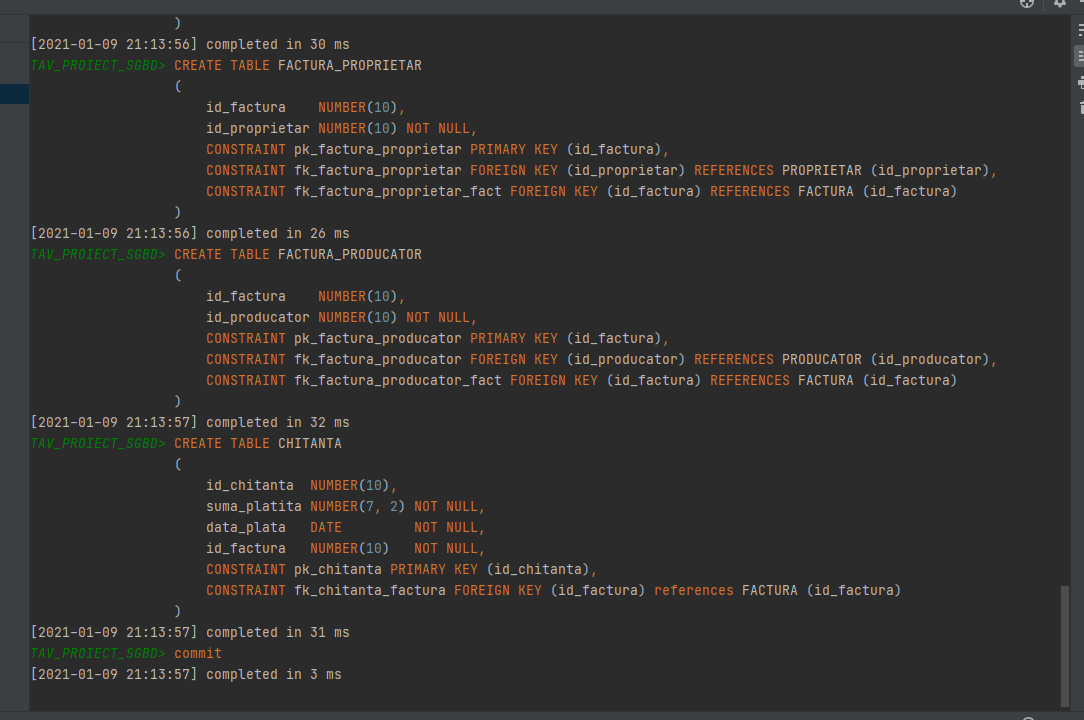
* Un proprietar are mai multe animale. Când un animăluț este tratat la clinică, i se face o nouă fișă medicală (relație de tip 3 între animal, diagnostic și un angajat care are voie să trateze animale).  Pe fișa medicală apare un diagnostic, pentru care se știe tratamentul (sau se adaugă în tabela diagnostic). Un animăluț se poate găsi în baza de date fără să aibă o fișă medicală (spre exemplu, dacă doar s-a microcipat)
* Angajații au unul sau mai multe contracte de muncă (în caz că un angajat vrea să negocieze ulterior ore suplimentare sau măriri de salariu etc., se face un nou contract, care, eventual, poate modifica un contract anterior). Doar unii angajați au voie să trateze animale (precizat in atributul din tabela angajat – ex. doar doctorii și asistentele etc.; întrucât au mai multe contracte, am ales să includ acest atribut pentru a evita o căutare prin toate contractele semnate pentru a verifica dacă există unul care să îi permită să trateze animăluțe ).
* Clinica are mai multe produse: vaccinuri, medicamente, aparatură medicală, mâncare de animale, dezinfectanți etc. Produsele sunt produse de producători, care încheie contracte cu clinica. Produsele se vor găsi în inventarele care se fac o dată la o perioadă de timp îndelungată.
* Clinica emite facturi către proprietarii de animăluțe. Aceștia le pot achita eșalonat, deci există mai multe chitanțe pentru o singură factură.
* Producătorii de produse emit și ei facturi, pe care clinica trebuie să le plătească (la fel, eșalonat).

***Pe următoarele două pagini se află diagrama entitate relație și diagrama conceptuală.***

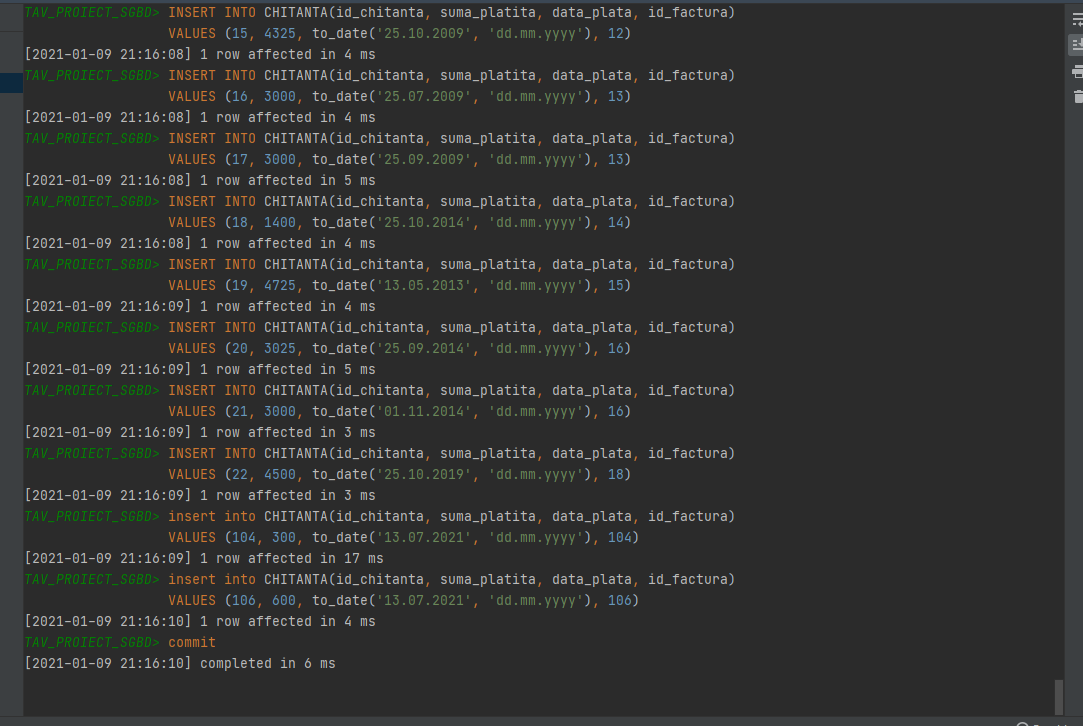




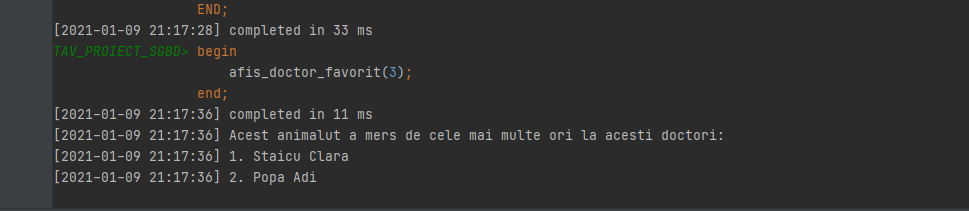
--4  
CREATE TABLE LOCATIE  
(  
 id\_locatie NUMBER(10),  
 oras VARCHAR2(32) NOT NULL,  
 judet VARCHAR2(32) NOT NULL,  
 adresa VARCHAR2(256) NOT NULL,  
 cod\_postal NUMBER(10),  
 CONSTRAINT pk\_locatie PRIMARY KEY (id\_locatie)  
);  
  
CREATE TABLE PROPRIETAR  
(  
 id\_proprietar NUMBER(10),  
 nume VARCHAR2(32) NOT NULL,  
 prenume VARCHAR2(32) NOT NULL,  
 telefon NUMBER(10) NOT NULL,  
 email VARCHAR2(64) NOT NULL,  
 id\_locatie NUMBER(10) NOT NULL,  
 CONSTRAINT pk\_proprietar PRIMARY KEY (id\_proprietar),  
 CONSTRAINT fk\_proprietar\_locatie FOREIGN KEY (id\_locatie) REFERENCES LOCATIE (id\_locatie)  
);  
  
CREATE TABLE ANIMALUT  
(  
 id\_animalut NUMBER(10),  
 nume VARCHAR2(32) NOT NULL,  
 rasa Varchar2(32) NOT NULL,  
 zi\_nastere DATE NOT NULL,  
 id\_proprietar NUMBER(10) NOT NULL,  
 CONSTRAINT pk\_animalut PRIMARY KEY (id\_animalut),  
 CONSTRAINT fk\_animalut\_proprietar FOREIGN KEY (id\_proprietar) REFERENCES PROPRIETAR (id\_proprietar)  
);  
  
CREATE TABLE DIAGNOSTIC  
(  
 id\_diagnostic NUMBER(10),  
 descriere VARCHAR2(4000) NOT NULL,  
 tratament VARCHAR2(4000) NOT NULL,  
 CONSTRAINT pk\_diagnostic PRIMARY KEY (id\_diagnostic)  
);  
  
CREATE TABLE ANGAJAT  
(  
 id\_angajat NUMBER(10),  
 nume VARCHAR2(32) NOT NULL,  
 prenume VARCHAR2(32) NOT NULL,  
 telefon NUMBER(10) NOT NULL,  
 email VARCHAR2(64) NOT NULL,  
 id\_locatie NUMBER(10) NOT NULL,  
 id\_sef NUMBER(10),  
 trateaza\_animale NUMBER(1) NOT NULL,  
 CONSTRAINT pk\_angajat PRIMARY KEY (id\_angajat),  
 CONSTRAINT fk\_angajat\_locatie FOREIGN KEY (id\_locatie) REFERENCES LOCATIE (id\_locatie),  
 CONSTRAINT fk\_angajat FOREIGN KEY (id\_sef) REFERENCES ANGAJAT (id\_angajat)  
);  
  
CREATE TABLE FISA\_MEDICALA  
(  
 id\_fisa NUMBER(10),  
 id\_diagnostic NUMBER(10) NOT NULL,  
 id\_animalut NUMBER(10) NOT NULL,  
 id\_doctor NUMBER(10) NOT NULL,  
 data\_fisa DATE NOT NULL,  
 observatii VARCHAR2(4000),  
 CONSTRAINT pk\_fisa\_medicala PRIMARY KEY (id\_fisa),  
 CONSTRAINT fk\_fisa\_diagnostic FOREIGN KEY (id\_diagnostic) REFERENCES DIAGNOSTIC (id\_diagnostic),  
 CONSTRAINT fk\_fisa\_animalut FOREIGN KEY (id\_animalut) REFERENCES ANIMALUT (id\_animalut),  
 CONSTRAINT fk\_fisa\_doctor FOREIGN KEY (id\_doctor) REFERENCES ANGAJAT (id\_angajat)  
);  
  
CREATE TABLE PRODUCATOR  
(  
 id\_producator NUMBER(10),  
 telefon NUMBER(10) NOT NULL,  
 email VARCHAR2(64) NOT NULL,  
 nume\_firma VARCHAR2(64) NOT NULL,  
 id\_locatie NUMBER(10) NOT NULL,  
 persoana\_contact VARCHAR2(64),  
 CONSTRAINT pk\_producator PRIMARY KEY (id\_producator),  
 CONSTRAINT fk\_producator\_locatie FOREIGN KEY (id\_locatie) REFERENCES LOCATIE (id\_locatie),  
 CONSTRAINT producator\_unique\_location UNIQUE (id\_locatie)  
);  
  
CREATE TABLE CONTRACT  
(  
 id\_contract NUMBER(10),  
 data\_semnare DATE NOT NULL,  
 observatii VARCHAR2(4000),  
 id\_contract\_modificat NUMBER(10),  
 CONSTRAINT pk\_contract PRIMARY KEY (id\_contract),  
 CONSTRAINT fk\_contract FOREIGN KEY (id\_contract\_modificat) REFERENCES CONTRACT (id\_contract)  
);  
  
CREATE TABLE CONTRACT\_ANGAJAT  
(  
 id\_contract NUMBER(10),  
 id\_angajat NUMBER(10) NOT NULL,  
 rol VARCHAR2(32) NOT NULL,  
 salariu NUMBER(5) NOT NULL,  
 CONSTRAINT pk\_contract\_angajat PRIMARY KEY (id\_contract),  
 CONSTRAINT fk\_contract\_angajat\_contract FOREIGN KEY (id\_contract) REFERENCES CONTRACT (id\_contract),  
 CONSTRAINT fk\_contract\_angajat FOREIGN KEY (id\_angajat) REFERENCES ANGAJAT (id\_angajat)  
);  
  
CREATE TABLE CONTRACT\_PRODUCATOR  
(  
 id\_contract NUMBER(10),  
 id\_producator NUMBER(10) NOT NULL,  
 CONSTRAINT pk\_contract\_producator PRIMARY KEY (id\_contract),  
 CONSTRAINT fk\_contract\_prod\_contract FOREIGN KEY (id\_contract) REFERENCES CONTRACT (id\_contract),  
 CONSTRAINT fk\_contract\_producator FOREIGN KEY (id\_producator) REFERENCES PRODUCATOR (id\_producator)  
);  
  
CREATE TABLE PRODUS  
(  
 id\_produs NUMBER(10),  
 caracteristici VARCHAR2(256) NOT NULL,  
 id\_producator NUMBER(10) NOT NULL,  
 CONSTRAINT pk\_produs PRIMARY KEY (id\_produs),  
 CONSTRAINT fk\_produs FOREIGN KEY (id\_producator) REFERENCES PRODUCATOR (id\_producator)  
);  
  
CREATE TABLE INVENTAR  
(  
 id\_inventar NUMBER(10),  
 data\_inventar DATE NOT NULL,  
 CONSTRAINT pk\_inventar PRIMARY KEY (id\_inventar)  
);  
  
CREATE TABLE PRODUSE\_DIN\_INVENTAR  
(  
 id\_produs NUMBER(10),  
 id\_inventar NUMBER(10),  
 cantitate NUMBER(5) NOT NULL,  
 data\_expirare DATE,  
 CONSTRAINT pk\_produse\_inventar PRIMARY KEY (id\_produs, id\_inventar),  
 CONSTRAINT fk\_prod\_inventar FOREIGN KEY (id\_inventar) REFERENCES INVENTAR (id\_inventar),  
 CONSTRAINT fk\_prod\_produs FOREIGN KEY (id\_produs) REFERENCES PRODUS (id\_produs)  
);  
  
CREATE TABLE FACTURA  
(  
 id\_factura NUMBER(10),  
 total\_factura NUMBER(7, 2) NOT NULL,  
 data\_emitere DATE NOT NULL,  
 CONSTRAINT pk\_factura PRIMARY KEY (id\_factura)  
);  
  
CREATE TABLE FACTURA\_PROPRIETAR  
(  
 id\_factura NUMBER(10),  
 id\_proprietar NUMBER(10) NOT NULL,  
 CONSTRAINT pk\_factura\_proprietar PRIMARY KEY (id\_factura),  
 CONSTRAINT fk\_factura\_proprietar FOREIGN KEY (id\_proprietar) REFERENCES PROPRIETAR (id\_proprietar),  
 CONSTRAINT fk\_factura\_proprietar\_fact FOREIGN KEY (id\_factura) REFERENCES FACTURA (id\_factura)  
);  
  
CREATE TABLE FACTURA\_PRODUCATOR  
(  
 id\_factura NUMBER(10),  
 id\_producator NUMBER(10) NOT NULL,  
 CONSTRAINT pk\_factura\_producator PRIMARY KEY (id\_factura),  
 CONSTRAINT fk\_factura\_producator FOREIGN KEY (id\_producator) REFERENCES PRODUCATOR (id\_producator),  
 CONSTRAINT fk\_factura\_producator\_fact FOREIGN KEY (id\_factura) REFERENCES FACTURA (id\_factura)  
);  
  
CREATE TABLE CHITANTA  
(  
 id\_chitanta NUMBER(10),  
 suma\_platita NUMBER(7, 2) NOT NULL,  
 data\_plata DATE NOT NULL,  
 id\_factura NUMBER(10) NOT NULL,  
 CONSTRAINT pk\_chitanta PRIMARY KEY (id\_chitanta),  
 CONSTRAINT fk\_chitanta\_factura FOREIGN KEY (id\_factura) references FACTURA (id\_factura)  
);  
  
commit;



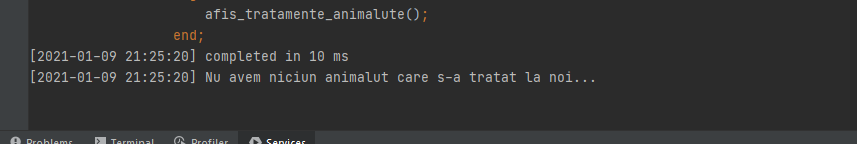
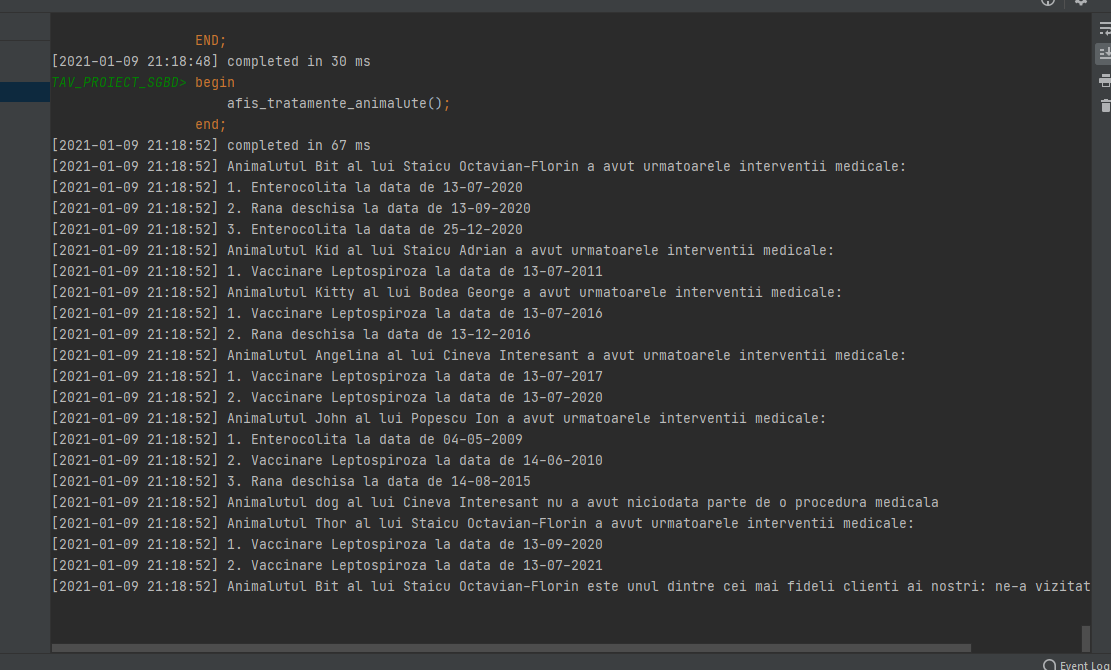
--5  
INSERT INTO LOCATIE(id\_locatie, oras, judet, adresa)  
VALUES (1, 'Bucuresti', 'Sector 2', 'soseaua colentina nr 2');  
INSERT INTO LOCATIE(id\_locatie, oras, judet, adresa, cod\_postal)  
VALUES (2, 'Bucuresti', 'Sector 1', 'soseaua Aviatorilor nr 1', 12345);  
INSERT INTO LOCATIE(id\_locatie, oras, judet, adresa)  
VALUES (3, 'Pantelimon', 'Ilfov', 'soseaua dn2');  
INSERT INTO LOCATIE(id\_locatie, oras, judet, adresa)  
VALUES (4, 'Cluj-Napoca', 'Cluj', 'soseaua jmekera');  
INSERT INTO LOCATIE(id\_locatie, oras, judet, adresa)  
VALUES (5, 'Iasi', 'Iasi', 'soseaua industriei');  
INSERT INTO LOCATIE(id\_locatie, oras, judet, adresa)  
VALUES (6, 'Timisoara', 'Timis', 'soseaua intreprinderii');  
INSERT INTO LOCATIE(id\_locatie, oras, judet, adresa)  
VALUES (7, 'Constanta', 'Constanta', 'soseaua antreprenorilor');  
  
INSERT INTO PROPRIETAR(id\_proprietar, nume, prenume, email, telefon, id\_locatie)  
VALUES (1, 'Staicu', 'Octavian-Florin', 'octavian.staicu@s.unibuc.ro', 0736659442, 1);  
INSERT INTO PROPRIETAR(id\_proprietar, nume, prenume, email, telefon, id\_locatie)  
VALUES (2, 'Bodea', 'George', 'george@yahoo.com', 0737392518, 2);  
INSERT INTO PROPRIETAR(id\_proprietar, nume, prenume, email, telefon, id\_locatie)  
VALUES (3, 'Staicu', 'Adrian', 'adi@gmail.com', 0736659441, 1);  
INSERT INTO PROPRIETAR(id\_proprietar, nume, prenume, email, telefon, id\_locatie)  
VALUES (4, 'Popescu', 'Ion', 'popescu.ion@gmail.com', 0771654789, 3);  
INSERT INTO PROPRIETAR(id\_proprietar, nume, prenume, email, telefon, id\_locatie)  
VALUES (5, 'Cineva', 'Interesant', 'jmekerie@my.fmi.unibuc.ro', 0777111999, 4);  
  
INSERT INTO ANIMALUT(id\_animalut, nume, rasa, zi\_nastere, id\_proprietar)  
VALUES (1, 'Bit', 'Boxer', to\_date('05/05/2017', 'DD/MM/YYYY'), 1);  
INSERT INTO ANIMALUT(id\_animalut, nume, rasa, zi\_nastere, id\_proprietar)  
VALUES (2, 'Kid', 'Ciobanesc German', to\_date('13/07/2010', 'DD/MM/YYYY'), 3);  
INSERT INTO ANIMALUT(id\_animalut, nume, rasa, zi\_nastere, id\_proprietar)  
VALUES (3, 'Kitty', 'Birmaneza', to\_date('25/12/2015', 'DD/MM/YYYY'), 2);  
INSERT INTO ANIMALUT(id\_animalut, nume, rasa, zi\_nastere, id\_proprietar)  
VALUES (4, 'Angelina', 'Pisica Tigrata de Hymalaia', to\_date('06/06/2016', 'DD/MM/YYYY'), 5);  
INSERT INTO ANIMALUT(id\_animalut, nume, rasa, zi\_nastere, id\_proprietar)  
VALUES (5, 'John', 'Metis', to\_date('05/12/2008', 'DD/MM/YYYY'), 4);  
insert into ANIMALUT(id\_animalut, nume, rasa, zi\_nastere, id\_proprietar)  
VALUES (6, 'dog', 'Boxer', to\_date('09.01.2021', 'dd.mm.yyyy'), 5);  
insert into ANIMALUT(id\_animalut, nume, rasa, zi\_nastere, id\_proprietar)  
VALUES (13, 'Thor', 'Golden Retriever', to\_date('13.07.2020', 'dd.mm.yyyy'), 1);  
  
INSERT INTO DIAGNOSTIC(id\_diagnostic, descriere, tratament)  
VALUES (1, 'Vaccinare Leptospiroza', 'Vaccin Leptospiroza');  
INSERT INTO DIAGNOSTIC(id\_diagnostic, descriere, tratament)  
VALUES (2, 'Enterocolita', 'Enteroguarg');  
INSERT INTO DIAGNOSTIC(id\_diagnostic, descriere, tratament)  
VALUES (3, 'Rana deschisa', 'Spray Cicatrizant');  
  
INSERT INTO ANGAJAT(id\_angajat, nume, prenume, telefon, email, id\_locatie, trateaza\_animale)  
VALUES (1, 'Staicu', 'Clara', 0736659440, 'dr.clara@clinicatareanimale.ro', 1, 1);  
INSERT INTO ANGAJAT(id\_angajat, nume, prenume, telefon, email, id\_sef, id\_locatie, trateaza\_animale)  
VALUES (2, 'Popa', 'Adi', 0723145867, 'asistent.adi@clinicatareanimale.ro', 1, 2, 1);  
INSERT INTO ANGAJAT(id\_angajat, nume, prenume, telefon, email, id\_locatie, trateaza\_animale)  
VALUES (3, 'Spalatorul', 'Pop', 0711012070, 'spalatorul.pop@gmail.com', 3, 0);  
  
INSERT INTO FISA\_MEDICALA(id\_fisa, id\_diagnostic, id\_animalut, id\_doctor, data\_fisa, observatii)  
VALUES (1, 2, 1, 1, to\_date('13.07.2020', 'dd.mm.yyyy'), 'Stapanul a dat caineul sa manance tort');  
INSERT INTO FISA\_MEDICALA(id\_fisa, id\_diagnostic, id\_animalut, id\_doctor, data\_fisa)  
VALUES (2, 1, 2, 1, to\_date('13.07.2011', 'dd.mm.yyyy'));  
INSERT INTO FISA\_MEDICALA(id\_fisa, id\_diagnostic, id\_animalut, id\_doctor, data\_fisa)  
VALUES (3, 1, 3, 2, to\_date('13.07.2016', 'dd.mm.yyyy'));  
INSERT INTO FISA\_MEDICALA(id\_fisa, id\_diagnostic, id\_animalut, id\_doctor, data\_fisa, observatii)  
VALUES (4, 3, 1, 2, to\_date('13.09.2020', 'dd.mm.yyyy'), 'Rana deschisa in zona botului');  
INSERT INTO FISA\_MEDICALA(id\_fisa, id\_diagnostic, id\_animalut, id\_doctor, data\_fisa, observatii)  
VALUES (5, 1, 4, 1, to\_date('13.07.2017', 'dd.mm.yyyy'), 'Una din cele mai frumoase specimene');  
INSERT INTO FISA\_MEDICALA(id\_fisa, id\_diagnostic, id\_animalut, id\_doctor, data\_fisa)  
VALUES (6, 2, 5, 1, to\_date('04.05.2009', 'dd.mm.yyyy'));  
INSERT INTO FISA\_MEDICALA(id\_fisa, id\_diagnostic, id\_animalut, id\_doctor, data\_fisa, observatii)  
VALUES (7, 1, 5, 2, to\_date('14.06.2010', 'dd.mm.yyyy'), 'Vaccinare anuala');  
INSERT INTO FISA\_MEDICALA(id\_fisa, id\_diagnostic, id\_animalut, id\_doctor, data\_fisa)  
VALUES (8, 1, 4, 2, to\_date('13.07.2020', 'dd.mm.yyyy'));  
INSERT INTO FISA\_MEDICALA(id\_fisa, id\_diagnostic, id\_animalut, id\_doctor, data\_fisa, observatii)  
VALUES (9, 3, 3, 1, to\_date('13.12.2016', 'dd.mm.yyyy'), 'S-a taiat intr-o conserva de peste');  
INSERT INTO FISA\_MEDICALA(id\_fisa, id\_diagnostic, id\_animalut, id\_doctor, data\_fisa, observatii)  
VALUES (10, 2, 1, 1, to\_date('25.12.2020', 'dd.mm.yyyy'), 'Cainele a mancat din pregaritile de Craciun');  
INSERT INTO FISA\_MEDICALA(id\_fisa, id\_diagnostic, id\_animalut, id\_doctor, data\_fisa)  
VALUES (11, 3, 5, 1, to\_date('14.08.2015', 'dd.mm.yyyy'));  
insert into FISA\_MEDICALA(id\_fisa, id\_diagnostic, id\_animalut, id\_doctor, data\_fisa)  
VALUES (13, 1, 13, 1, to\_date('13.09.2020', 'dd.mm.yyyy'));  
insert into FISA\_MEDICALA(id\_fisa, id\_diagnostic, id\_animalut, id\_doctor, data\_fisa)  
VALUES (12, 1, 13, 1, to\_date('13.07.2021', 'dd.mm.yyyy'));  
  
INSERT INTO PRODUCATOR(id\_producator, nume\_firma, telefon, email, id\_locatie)  
VALUES (1, 'Mancare Catei SA', 0767582563, 'comenzi@mancarecatei.ro', 5);  
INSERT INTO PRODUCATOR(id\_producator, nume\_firma, telefon, email, persoana\_contact, id\_locatie)  
VALUES (2, 'Aparatura Animale SRL', 0798712365, 'aparatus.ion@aparaturaanimale.ro', 'Aparatus Ion', 6);  
INSERT INTO PRODUCATOR(id\_producator, nume\_firma, telefon, email, persoana\_contact, id\_locatie)  
VALUES (3, 'Vaccinam cu Iubire', 0712345678, 'bussines@vacciniubire.ro', 'Ana Vacix', 7);  
  
INSERT INTO CONTRACT(id\_contract, data\_semnare)  
VALUES (1, to\_date('02.01.2005', 'dd.mm.yyyy'));  
INSERT INTO CONTRACT(id\_contract, data\_semnare)  
VALUES (2, to\_date('13.07.2004', 'dd.mm.yyyy'));  
INSERT INTO CONTRACT(id\_contract, data\_semnare)  
VALUES (3, to\_date('09.09.2005', 'dd.mm.yyyy'));  
INSERT INTO CONTRACT(id\_contract, data\_semnare)  
VALUES (4, to\_date('01.05.2007', 'dd.mm.yyyy'));  
INSERT INTO CONTRACT(id\_contract, data\_semnare)  
VALUES (5, to\_date('02.06.2008', 'dd.mm.yyyy'));  
INSERT INTO CONTRACT(id\_contract, data\_semnare)  
VALUES (6, to\_date('02.01.2005', 'dd.mm.yyyy'));  
INSERT INTO CONTRACT(id\_contract, data\_semnare, observatii, id\_contract\_modificat)  
VALUES (7, to\_date('28.12.2020', 'dd.mm.yyyy'), 'Promovare', 2);  
  
INSERT INTO CONTRACT\_ANGAJAT(id\_contract, id\_angajat, rol, salariu)  
VALUES (1, 1, 'doctor veterinar', 6000);  
INSERT INTO CONTRACT\_ANGAJAT(id\_contract, id\_angajat, rol, salariu)  
VALUES (2, 2, 'asistent veterinar', 3000);  
INSERT INTO CONTRACT\_ANGAJAT(id\_contract, id\_angajat, rol, salariu)  
VALUES (3, 3, 'paznic', 1500);  
INSERT INTO CONTRACT\_ANGAJAT(id\_contract, id\_angajat, rol, salariu)  
VALUES (7, 2, 'doctor veterinar', 6000);  
  
INSERT INTO CONTRACT\_PRODUCATOR(id\_contract, id\_producator)  
VALUES (4, 1);  
INSERT INTO CONTRACT\_PRODUCATOR(id\_contract, id\_producator)  
VALUES (5, 2);  
INSERT INTO CONTRACT\_PRODUCATOR(id\_contract, id\_producator)  
VALUES (6, 3);  
INSERT INTO PRODUS(id\_produs, caracteristici, id\_producator)  
VALUES (1, 'boabe de caini cu carne de vita si somon', 1);  
INSERT INTO PRODUS(id\_produs, caracteristici, id\_producator)  
VALUES (2, 'boabe de pisici cu ulei de peste', 1);  
INSERT INTO PRODUS(id\_produs, caracteristici, id\_producator)  
VALUES (3, 'aparat radiografie', 2);  
INSERT INTO PRODUS(id\_produs, caracteristici, id\_producator)  
VALUES (4, 'stetoscop', 2);  
INSERT INTO PRODUS(id\_produs, caracteristici, id\_producator)  
VALUES (5, 'masa operatii', 2);  
INSERT INTO PRODUS(id\_produs, caracteristici, id\_producator)  
VALUES (6, 'vaccin leptospiroza', 3);  
INSERT INTO PRODUS(id\_produs, caracteristici, id\_producator)  
VALUES (7, 'vaccin hexavalent', 3);  
  
INSERT INTO INVENTAR(id\_inventar, data\_inventar)  
VALUES (1, to\_date('05.01.2010', 'dd.mm.yyyy'));  
INSERT INTO INVENTAR(id\_inventar, data\_inventar)  
VALUES (2, to\_date('05.01.2015', 'dd.mm.yyyy'));  
INSERT INTO INVENTAR(id\_inventar, data\_inventar)  
VALUES (3, to\_date('05.01.2020', 'dd.mm.yyyy'));  
  
INSERT INTO PRODUSE\_DIN\_INVENTAR(id\_produs, id\_inventar, cantitate, data\_expirare)  
VALUES (1, 1, 10, to\_date('05.01.2011', 'dd.mm.yyyy'));  
INSERT INTO PRODUSE\_DIN\_INVENTAR(id\_produs, id\_inventar, cantitate, data\_expirare)  
VALUES (2, 1, 15, to\_date('05.01.2011', 'dd.mm.yyyy'));  
INSERT INTO PRODUSE\_DIN\_INVENTAR(id\_produs, id\_inventar, cantitate, data\_expirare)  
VALUES (1, 2, 20, to\_date('05.01.2016', 'dd.mm.yyyy'));  
INSERT INTO PRODUSE\_DIN\_INVENTAR(id\_produs, id\_inventar, cantitate, data\_expirare)  
VALUES (2, 2, 25, to\_date('05.01.2016', 'dd.mm.yyyy'));  
INSERT INTO PRODUSE\_DIN\_INVENTAR(id\_produs, id\_inventar, cantitate, data\_expirare)  
VALUES (1, 3, 35, to\_date('05.01.2021', 'dd.mm.yyyy'));  
INSERT INTO PRODUSE\_DIN\_INVENTAR(id\_produs, id\_inventar, cantitate, data\_expirare)  
VALUES (2, 3, 30, to\_date('05.01.2021', 'dd.mm.yyyy'));  
INSERT INTO PRODUSE\_DIN\_INVENTAR(id\_produs, id\_inventar, cantitate)  
VALUES (3, 1, 1);  
INSERT INTO PRODUSE\_DIN\_INVENTAR(id\_produs, id\_inventar, cantitate)  
VALUES (3, 2, 1);  
INSERT INTO PRODUSE\_DIN\_INVENTAR(id\_produs, id\_inventar, cantitate)  
VALUES (3, 3, 2);  
INSERT INTO PRODUSE\_DIN\_INVENTAR(id\_produs, id\_inventar, cantitate)  
VALUES (4, 1, 5);  
INSERT INTO PRODUSE\_DIN\_INVENTAR(id\_produs, id\_inventar, cantitate)  
VALUES (4, 2, 10);  
INSERT INTO PRODUSE\_DIN\_INVENTAR(id\_produs, id\_inventar, cantitate)  
VALUES (4, 3, 15);  
INSERT INTO PRODUSE\_DIN\_INVENTAR(id\_produs, id\_inventar, cantitate)  
VALUES (5, 1, 1);  
INSERT INTO PRODUSE\_DIN\_INVENTAR(id\_produs, id\_inventar, cantitate)  
VALUES (5, 2, 1);  
INSERT INTO PRODUSE\_DIN\_INVENTAR(id\_produs, id\_inventar, cantitate)  
VALUES (5, 3, 2);  
INSERT INTO PRODUSE\_DIN\_INVENTAR(id\_produs, id\_inventar, cantitate, data\_expirare)  
VALUES (6, 1, 100, to\_date('13.07.2011', 'dd.mm.yyyy'));  
INSERT INTO PRODUSE\_DIN\_INVENTAR(id\_produs, id\_inventar, cantitate, data\_expirare)  
VALUES (7, 1, 150, to\_date('13.07.2011', 'dd.mm.yyyy'));  
INSERT INTO PRODUSE\_DIN\_INVENTAR(id\_produs, id\_inventar, cantitate, data\_expirare)  
VALUES (6, 2, 100, to\_date('13.07.2011', 'dd.mm.yyyy'));  
INSERT INTO PRODUSE\_DIN\_INVENTAR(id\_produs, id\_inventar, cantitate, data\_expirare)  
VALUES (7, 2, 150, to\_date('13.07.2011', 'dd.mm.yyyy'));  
INSERT INTO PRODUSE\_DIN\_INVENTAR(id\_produs, id\_inventar, cantitate, data\_expirare)  
VALUES (6, 3, 234, to\_date('13.07.2011', 'dd.mm.yyyy'));  
INSERT INTO PRODUSE\_DIN\_INVENTAR(id\_produs, id\_inventar, cantitate, data\_expirare)  
VALUES (7, 3, 512, to\_date('13.07.2011', 'dd.mm.yyyy'));  
  
INSERT INTO FACTURA(ID\_FACTURA, TOTAL\_FACTURA, DATA\_EMITERE)  
VALUES (1, 300, to\_date('13.07.2020', 'dd.mm.yyyy'));  
INSERT INTO FACTURA(ID\_FACTURA, TOTAL\_FACTURA, DATA\_EMITERE)  
VALUES (2, 300, to\_date('13.07.2011', 'dd.mm.yyyy'));  
INSERT INTO FACTURA(ID\_FACTURA, TOTAL\_FACTURA, DATA\_EMITERE)  
VALUES (3, 300, to\_date('13.07.2015', 'dd.mm.yyyy'));  
INSERT INTO FACTURA(ID\_FACTURA, TOTAL\_FACTURA, DATA\_EMITERE)  
VALUES (4, 300, to\_date('13.09.2020', 'dd.mm.yyyy'));  
INSERT INTO FACTURA(ID\_FACTURA, TOTAL\_FACTURA, DATA\_EMITERE)  
VALUES (5, 300, to\_date('13.07.2017', 'dd.mm.yyyy'));  
INSERT INTO FACTURA(ID\_FACTURA, TOTAL\_FACTURA, DATA\_EMITERE)  
VALUES (6, 300, to\_date('04.05.2009', 'dd.mm.yyyy'));  
INSERT INTO FACTURA(ID\_FACTURA, TOTAL\_FACTURA, DATA\_EMITERE)  
VALUES (7, 300, to\_date('14.06.2010', 'dd.mm.yyyy'));  
INSERT INTO FACTURA(ID\_FACTURA, TOTAL\_FACTURA, DATA\_EMITERE)  
VALUES (8, 300, to\_date('13.07.2020', 'dd.mm.yyyy'));  
INSERT INTO FACTURA(ID\_FACTURA, TOTAL\_FACTURA, DATA\_EMITERE)  
VALUES (9, 300, to\_date('13.12.2016', 'dd.mm.yyyy'));  
INSERT INTO FACTURA(ID\_FACTURA, TOTAL\_FACTURA, DATA\_EMITERE)  
VALUES (10, 300, to\_date('25.12.2020', 'dd.mm.yyyy'));  
insert into FACTURA(id\_factura, total\_factura, data\_emitere)  
VALUES (104, 300, to\_date('13.07.2021', 'dd.mm.yyyy'));  
insert into FACTURA(id\_factura, total\_factura, data\_emitere)  
VALUES (105, 300, to\_date('13.07.2021', 'dd.mm.yyyy'));  
insert into FACTURA(id\_factura, total\_factura, data\_emitere)  
VALUES (106, 600, to\_date('12.05.2021', 'dd.mm.yyyy'));  
Insert into FACTURA(id\_factura, total\_factura, data\_emitere)  
VALUES (1000, 1000, sysdate);  
  
INSERT INTO FACTURA\_PROPRIETAR(id\_factura, id\_proprietar)  
VALUES (1, 1);  
INSERT INTO FACTURA\_PROPRIETAR(id\_factura, id\_proprietar)  
VALUES (2, 3);  
INSERT INTO FACTURA\_PROPRIETAR(id\_factura, id\_proprietar)  
VALUES (3, 2);  
INSERT INTO FACTURA\_PROPRIETAR(id\_factura, id\_proprietar)  
VALUES (4, 1);  
INSERT INTO FACTURA\_PROPRIETAR(id\_factura, id\_proprietar)  
VALUES (5, 5);  
INSERT INTO FACTURA\_PROPRIETAR(id\_factura, id\_proprietar)  
VALUES (6, 4);  
INSERT INTO FACTURA\_PROPRIETAR(id\_factura, id\_proprietar)  
VALUES (7, 4);  
INSERT INTO FACTURA\_PROPRIETAR(id\_factura, id\_proprietar)  
VALUES (8, 5);  
INSERT INTO FACTURA\_PROPRIETAR(id\_factura, id\_proprietar)  
VALUES (9, 2);  
INSERT INTO FACTURA\_PROPRIETAR(id\_factura, id\_proprietar)  
VALUES (10, 1);  
insert into FACTURA\_PROPRIETAR(id\_factura, id\_proprietar)  
VALUES (104, 1);  
insert into FACTURA\_PROPRIETAR(id\_factura, id\_proprietar)  
VALUES (105, 1);  
insert into FACTURA\_PROPRIETAR(id\_factura, id\_proprietar)  
VALUES (106, 2);  
INSERT INTO FACTURA\_PROPRIETAR(id\_factura, id\_proprietar)  
VALUES (1000, 5);  
  
INSERT INTO FACTURA(ID\_FACTURA, TOTAL\_FACTURA, DATA\_EMITERE)  
VALUES (11, 1500, to\_date('25.10.2009', 'dd.mm.yyyy'));  
INSERT INTO FACTURA(ID\_FACTURA, TOTAL\_FACTURA, DATA\_EMITERE)  
VALUES (12, 4325, to\_date('13.11.2009', 'dd.mm.yyyy'));  
INSERT INTO FACTURA(ID\_FACTURA, TOTAL\_FACTURA, DATA\_EMITERE)  
VALUES (13, 6000, to\_date('25.07.2009', 'dd.mm.yyyy'));  
INSERT INTO FACTURA(ID\_FACTURA, TOTAL\_FACTURA, DATA\_EMITERE)  
VALUES (14, 1400, to\_date('25.10.2014', 'dd.mm.yyyy'));  
INSERT INTO FACTURA(ID\_FACTURA, TOTAL\_FACTURA, DATA\_EMITERE)  
VALUES (15, 4725, to\_date('13.04.2013', 'dd.mm.yyyy'));  
INSERT INTO FACTURA(ID\_FACTURA, TOTAL\_FACTURA, DATA\_EMITERE)  
VALUES (16, 6025, to\_date('25.09.2014', 'dd.mm.yyyy'));  
INSERT INTO FACTURA(ID\_FACTURA, TOTAL\_FACTURA, DATA\_EMITERE)  
VALUES (17, 5000, to\_date('13.11.2019', 'dd.mm.yyyy'));  
INSERT INTO FACTURA(ID\_FACTURA, TOTAL\_FACTURA, DATA\_EMITERE)  
VALUES (18, 7500, to\_date('25.09.2019', 'dd.mm.yyyy'));  
INSERT INTO FACTURA(ID\_FACTURA, TOTAL\_FACTURA, DATA\_EMITERE)  
VALUES (19, 9256.35, to\_date('12.04.2019', 'dd.mm.yyyy'));  
  
  
INSERT INTO FACTURA\_PRODUCATOR(id\_factura, id\_producator)  
VALUES (11, 1);  
INSERT INTO FACTURA\_PRODUCATOR(id\_factura, id\_producator)  
VALUES (12, 2);  
INSERT INTO FACTURA\_PRODUCATOR(id\_factura, id\_producator)  
VALUES (13, 3);  
INSERT INTO FACTURA\_PRODUCATOR(id\_factura, id\_producator)  
VALUES (14, 1);  
INSERT INTO FACTURA\_PRODUCATOR(id\_factura, id\_producator)  
VALUES (15, 2);  
INSERT INTO FACTURA\_PRODUCATOR(id\_factura, id\_producator)  
VALUES (16, 3);  
INSERT INTO FACTURA\_PRODUCATOR(id\_factura, id\_producator)  
VALUES (17, 1);  
INSERT INTO FACTURA\_PRODUCATOR(id\_factura, id\_producator)  
VALUES (18, 2);  
INSERT INTO FACTURA\_PRODUCATOR(id\_factura, id\_producator)  
VALUES (19, 3);  
  
INSERT INTO CHITANTA(id\_chitanta, suma\_platita, data\_plata, id\_factura)  
VALUES (1, 300, to\_date('13.07.2020', 'dd.mm.yyyy'), 1);  
INSERT INTO CHITANTA(id\_chitanta, suma\_platita, data\_plata, id\_factura)  
VALUES (2, 300, to\_date('13.07.2011', 'dd.mm.yyyy'), 2);  
INSERT INTO CHITANTA(id\_chitanta, suma\_platita, data\_plata, id\_factura)  
VALUES (3, 300, to\_date('13.08.2015', 'dd.mm.yyyy'), 3);  
INSERT INTO CHITANTA(id\_chitanta, suma\_platita, data\_plata, id\_factura)  
VALUES (4, 300, to\_date('03.01.2021', 'dd.mm.yyyy'), 4);  
INSERT INTO CHITANTA(id\_chitanta, suma\_platita, data\_plata, id\_factura)  
VALUES (5, 150, to\_date('13.07.2017', 'dd.mm.yyyy'), 5);  
INSERT INTO CHITANTA(id\_chitanta, suma\_platita, data\_plata, id\_factura)  
VALUES (6, 75.25, to\_date('13.08.2017', 'dd.mm.yyyy'), 5);  
INSERT INTO CHITANTA(id\_chitanta, suma\_platita, data\_plata, id\_factura)  
VALUES (7, 75.75, to\_date('13.09.2017', 'dd.mm.yyyy'), 5);  
INSERT INTO CHITANTA(id\_chitanta, suma\_platita, data\_plata, id\_factura)  
VALUES (8, 100, to\_date('13.07.2009', 'dd.mm.yyyy'), 6);  
INSERT INTO CHITANTA(id\_chitanta, suma\_platita, data\_plata, id\_factura)  
VALUES (9, 200, to\_date('17.08.2009', 'dd.mm.yyyy'), 6);  
INSERT INTO CHITANTA(id\_chitanta, suma\_platita, data\_plata, id\_factura)  
VALUES (10, 300, to\_date('13.07.2010', 'dd.mm.yyyy'), 7);  
INSERT INTO CHITANTA(id\_chitanta, suma\_platita, data\_plata, id\_factura)  
VALUES (11, 300, to\_date('13.09.2020', 'dd.mm.yyyy'), 8);  
INSERT INTO CHITANTA(id\_chitanta, suma\_platita, data\_plata, id\_factura)  
VALUES (12, 300, to\_date('13.12.2016', 'dd.mm.yyyy'), 9);  
INSERT INTO CHITANTA(id\_chitanta, suma\_platita, data\_plata, id\_factura)  
VALUES (13, 300, to\_date('15.01.2021', 'dd.mm.yyyy'), 10);  
INSERT INTO CHITANTA(id\_chitanta, suma\_platita, data\_plata, id\_factura)  
VALUES (14, 1500, to\_date('25.10.2009', 'dd.mm.yyyy'), 11);  
INSERT INTO CHITANTA(id\_chitanta, suma\_platita, data\_plata, id\_factura)  
VALUES (15, 4325, to\_date('25.10.2009', 'dd.mm.yyyy'), 12);  
INSERT INTO CHITANTA(id\_chitanta, suma\_platita, data\_plata, id\_factura)  
VALUES (16, 3000, to\_date('25.07.2009', 'dd.mm.yyyy'), 13);  
INSERT INTO CHITANTA(id\_chitanta, suma\_platita, data\_plata, id\_factura)  
VALUES (17, 3000, to\_date('25.09.2009', 'dd.mm.yyyy'), 13);  
INSERT INTO CHITANTA(id\_chitanta, suma\_platita, data\_plata, id\_factura)  
VALUES (18, 1400, to\_date('25.10.2014', 'dd.mm.yyyy'), 14);  
INSERT INTO CHITANTA(id\_chitanta, suma\_platita, data\_plata, id\_factura)  
VALUES (19, 4725, to\_date('13.05.2013', 'dd.mm.yyyy'), 15);  
INSERT INTO CHITANTA(id\_chitanta, suma\_platita, data\_plata, id\_factura)  
VALUES (20, 3025, to\_date('25.09.2014', 'dd.mm.yyyy'), 16);  
INSERT INTO CHITANTA(id\_chitanta, suma\_platita, data\_plata, id\_factura)  
VALUES (21, 3000, to\_date('01.11.2014', 'dd.mm.yyyy'), 16);  
INSERT INTO CHITANTA(id\_chitanta, suma\_platita, data\_plata, id\_factura)  
VALUES (22, 4500, to\_date('25.10.2019', 'dd.mm.yyyy'), 18);  
insert into CHITANTA(id\_chitanta, suma\_platita, data\_plata, id\_factura)  
VALUES (104, 300, to\_date('13.07.2021', 'dd.mm.yyyy'), 104);  
insert into CHITANTA(id\_chitanta, suma\_platita, data\_plata, id\_factura)  
VALUES (106, 600, to\_date('13.07.2021', 'dd.mm.yyyy'), 106);  
  
commit;



--6  
CREATE OR REPLACE PROCEDURE *afis\_doctor\_favorit*(id\_animal ANIMALUT.id\_animalut%type)  
 IS  
 TYPE tab\_ind IS TABLE OF PLS\_INTEGER INDEX BY PLS\_INTEGER;  
 cnt tab\_ind;  
 index\_max pls\_integer;  
 ok binary\_integer := 0;  
 id pls\_integer;  
 num ANGAJAT.nume%type;  
 prenum ANGAJAT.prenume%type;  
 co pls\_integer := 0;  
BEGIN  
 for i in (select fm.id\_doctor  
 from FISA\_MEDICALA fm  
 where fm.id\_animalut = id\_animal)  
 loop  
 begin  
 cnt(i.id\_doctor) := cnt(i.id\_doctor) + 1;  
 if ok = 1 then  
 if cnt(i.id\_doctor) > cnt(index\_max) then  
 index\_max := i.id\_doctor;  
 end if;  
 end if;  
 EXCEPTION  
 when NO\_DATA\_FOUND then  
 cnt(i.id\_doctor) := 1;  
 if ok = 0 then  
 ok := 1;  
 index\_max := i.id\_doctor;  
 end if;  
 end;  
 end loop;  
 if ok = 0 then  
 DBMS\_OUTPUT.*PUT\_LINE*('Acest animalut nu are nicio fisa medicala in sistem');  
 else  
 id := cnt.FIRST;  
 DBMS\_OUTPUT.*PUT\_LINE*('Acest animalut a mers de cele mai multe ori la acesti doctori:');  
 loop  
 exit when id is null;  
 if cnt(id) = cnt(index\_max) then  
 select a.nume, a.prenume  
 into num, prenum  
 from ANGAJAT a  
 where a.id\_angajat = id;  
 co := co + 1;  
 DBMS\_OUTPUT.*PUT\_LINE*(co || '. ' || num || ' ' || prenum);  
 end if;  
 id := cnt.NEXT(id);  
 end loop;  
 end if;  
END;  
/  
  
begin  
 *afis\_doctor\_favorit*(3);  
end;  
/



--7  
CREATE OR REPLACE PROCEDURE *afis\_tratamente\_animalute* IS  
 TYPE refcursor IS REF CURSOR;  
 cursor mycursor is select a.nume,  
 p.nume,  
 p.prenume,  
 cursor (select diag.descriere, fm.data\_fisa  
 from FISA\_MEDICALA fm,  
 DIAGNOSTIC diag  
 where fm.id\_diagnostic = diag.id\_diagnostic  
 and a.id\_animalut = fm.id\_animalut  
 )  
 from ANIMALUT a,  
 PROPRIETAR p  
 where a.id\_proprietar = p.id\_proprietar;  
 v\_cursor refcursor;  
 nume\_animalut ANIMALUT.nume%type;  
 nume\_proprietar PROPRIETAR.nume%type;  
 prenume\_proprietar PROPRIETAR.prenume%type;  
 diag DIAGNOSTIC.descriere%type;  
 data\_diagnostic FISA\_MEDICALA.data\_fisa%type;  
 nume\_animalut\_max ANIMALUT.nume%type;  
 nume\_proprietar\_max PROPRIETAR.nume%type;  
 prenume\_proprietar\_max PROPRIETAR.prenume%type;  
 co\_max pls\_integer := 0;  
 faliment EXCEPTION;  
BEGIN  
 open mycursor;  
 loop  
 fetch mycursor into nume\_animalut,nume\_proprietar,prenume\_proprietar,v\_cursor;  
 exit when mycursor%notfound;  
 fetch v\_cursor into diag, data\_diagnostic;  
 if v\_cursor%notfound then  
 DBMS\_OUTPUT.*PUT\_LINE*(  
 'Animalutul ' || nume\_animalut || ' al lui ' || nume\_proprietar || ' ' || prenume\_proprietar ||  
 ' nu a avut niciodata parte de o procedura medicala');  
 DBMS\_OUTPUT.*NEW\_LINE*();  
 else  
 DBMS\_OUTPUT.*PUT\_LINE*(  
 'Animalutul ' || nume\_animalut || ' al lui ' || nume\_proprietar || ' ' || prenume\_proprietar ||  
 ' a avut urmatoarele interventii medicale:');  
 loop  
 exit when v\_cursor%notfound;  
 DBMS\_OUTPUT.*PUT\_LINE*(v\_cursor%rowcount || '. ' || diag || ' la data de ' || data\_diagnostic);  
 fetch v\_cursor into diag, data\_diagnostic;  
 end loop;  
 if v\_cursor%rowcount > co\_max then  
 co\_max := v\_cursor%rowcount;  
 nume\_animalut\_max := nume\_animalut;  
 nume\_proprietar\_max := nume\_proprietar;  
 prenume\_proprietar\_max := prenume\_proprietar;  
 end if;  
 DBMS\_OUTPUT.*NEW\_LINE*();  
 end if;  
 end loop;  
 close mycursor;  
 if co\_max = 0 then  
 raise faliment;  
 end if;  
 DBMS\_OUTPUT.*PUT\_LINE*('Animalutul ' || nume\_animalut\_max || ' al lui ' || nume\_proprietar\_max || ' ' ||  
 prenume\_proprietar\_max ||  
 ' este unul dintre cei mai fideli clienti ai nostri: ne-a vizitat de ' || co\_max || ' ori :)');  
  
EXCEPTION  
 when faliment then  
 DBMS\_OUTPUT.*PUT\_LINE*('Nu avem niciun animalut care s-a tratat la noi...');  
  
END;  
/  
  
begin  
 *afis\_tratamente\_animalute*();  
end;  
/



\*in a doua poză s-a rulat procedura fără nicio valoare inserată în tabelele din baza de date, pentru a ilustra excepția

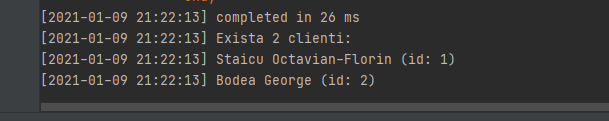
\*\*în cazul în care sunt mai multe animăluțe cu același număr de tratamente, se afișează doar unul

--8  
--Intoarcem id-ul clientilor cu cea mai mare suma achitata (nu datorata)  
--Daca sunt mai multi, ii intoarcem pe toti

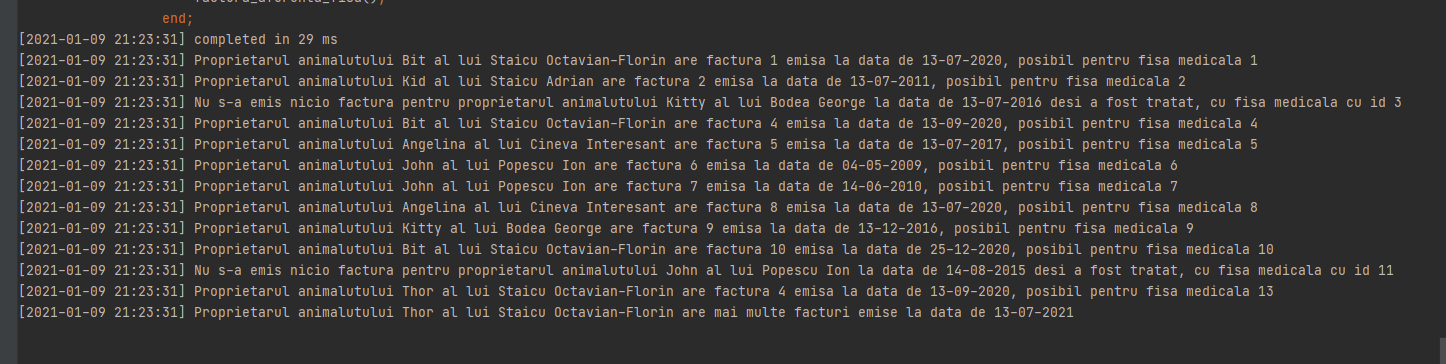
-- Obs: Desi cerinta 8 cerea sa se trateze exceptiile, aceasta functie nu genereaza exceptii. Astfel, functia 6 este

-- cea in care este evidentiata o tratare de exceptii foarte utila. Multumesc pentru intelegere :)  
CREATE or REPLACE TYPE tab\_imb is TABLE OF NUMBER(10);  
/  
  
CREATE or REPLACE FUNCTION *topclienti*(top\_clienti out tab\_imb) return pls\_integer  
 IS  
 TYPE refcursor IS REF CURSOR;  
 cursor mycursor is select f.id\_proprietar,  
 cursor (select c.suma\_platita  
 from CHITANTA c  
 where c.id\_factura = f.id\_factura  
 )  
 from FACTURA\_PROPRIETAR f,  
 PROPRIETAR p  
 where f.id\_proprietar = p.id\_proprietar  
 order by p.id\_proprietar;  
 v\_cursor refcursor;  
 id FACTURA\_PROPRIETAR.id\_factura%type := null;  
 suma pls\_integer := 0;  
 total\_suma pls\_integer := 0;  
 max\_suma pls\_integer := 0;  
 last\_id PROPRIETAR.id\_proprietar%type := null;  
 co pls\_integer := 0;  
BEGIN  
 top\_clienti := tab\_imb();  
 open mycursor;  
 loop  
 fetch mycursor into id, v\_cursor;  
 if last\_id is null then  
 last\_id := id;  
 else  
 if mycursor%notfound or id <> last\_id then  
 if total\_suma > max\_suma then  
 top\_clienti.DELETE;  
 top\_clienti.EXTEND;  
 co := 1;  
 top\_clienti(1) := last\_id;  
 max\_suma := total\_suma;  
 elsif total\_suma = max\_suma then  
 top\_clienti.EXTEND;  
 co := co + 1;  
 top\_clienti(co) := last\_id;  
 end if;  
 last\_id := id;  
 total\_suma := 0;  
 end if;  
 end if;  
 exit when mycursor%notfound;  
 loop  
 fetch v\_cursor into suma;  
 exit when v\_cursor%notfound;  
 total\_suma := total\_suma + suma;  
 end loop;  
 end loop;  
 close mycursor;  
 return co;  
  
END;  
/  
--functia returneaza nr de clienti, iar parametrul intoarce colectia  
--vom scrie un bloc anonim mai complex, pt a verifica corectitudinea

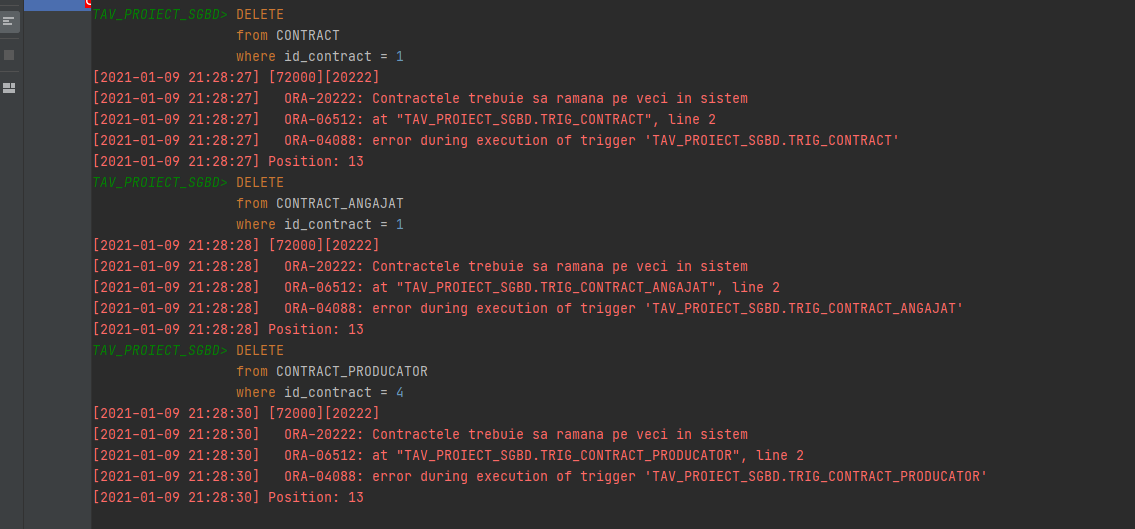
--functiei  
declare  
 co pls\_integer;  
 id NUMBER(10);  
 mytab tab\_imb;  
 num PROPRIETAR.nume%type;  
 prenum PROPRIETAR.prenume%type;  
begin  
 co := *TOPCLIENTI*(mytab);  
 DBMS\_OUTPUT.*PUT\_LINE*('Exista ' || co || ' clienti:');  
  
 id := mytab.FIRST;  
 loop  
 exit when id is null;  
 select nume, prenume  
 into num, prenum  
 from PROPRIETAR  
 where id\_proprietar = mytab(id);  
 DBMS\_OUTPUT.*PUT\_LINE*(num || ' ' || prenum || ' (id: ' || mytab(id) || ')');  
 id := mytab.NEXT(id);  
 end loop;  
end;  
/



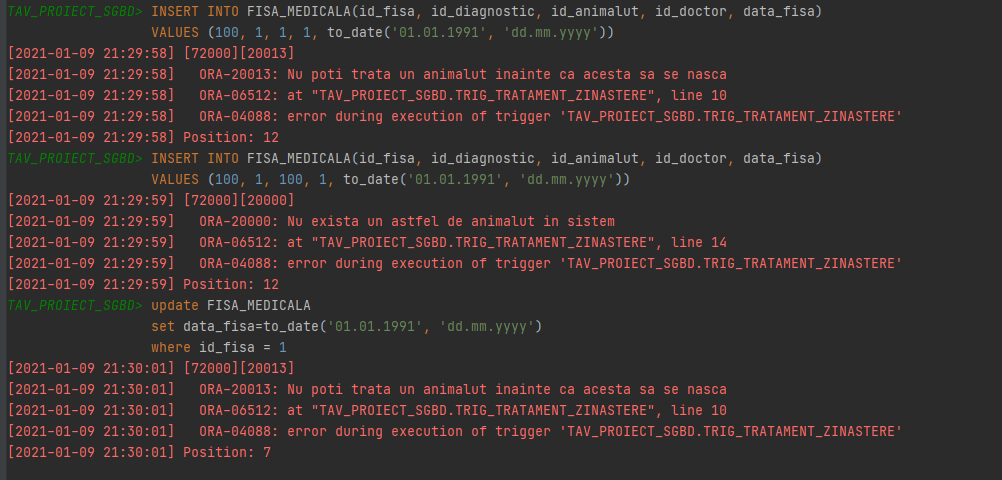
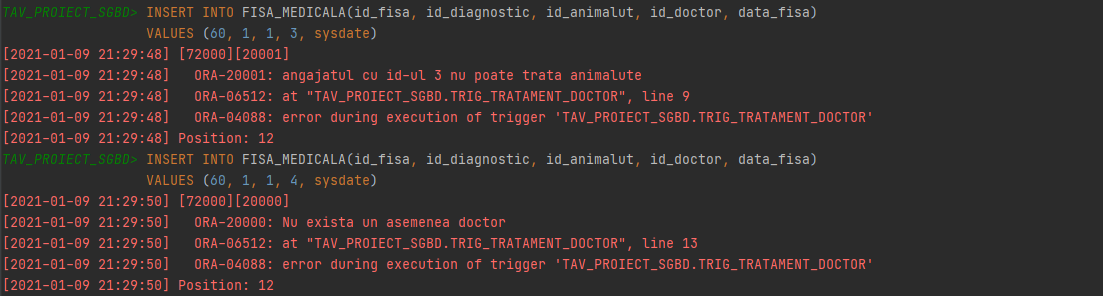
--9  
--Modelul de bussines nu cere sa se tina ce factura carui tratament apartine  
--(adica nu exista un fk intre factura si fisa medicala)  
--acesta e un algoritm probabilistic care gaseste daca stapanul unui animalut are o factura emisa la data cand --acesta are o fisa  
CREATE OR REPLACE PROCEDURE *factura\_aferenta\_fisa* IS  
 id FACTURA\_PROPRIETAR.id\_factura%type;  
BEGIN  
 for i in (select p.id\_proprietar, p.nume propnum, p.prenume, a.nume animnume, fm.data\_fisa, fm.id\_fisa  
 from FISA\_MEDICALA fm,  
 ANIMALUT a,  
 PROPRIETAR p  
 where fm.id\_animalut = a.id\_animalut  
 and a.id\_proprietar = p.id\_proprietar)  
 loop  
 BEGIN  
 select fp.id\_factura  
 into id  
 from FACTURA\_PROPRIETAR fp,  
 FACTURA f  
 where fp.id\_proprietar = i.id\_proprietar  
 and fp.id\_factura = f.id\_factura  
 and i.data\_fisa = f.data\_emitere;  
 DBMS\_OUTPUT.*PUT\_LINE*('Proprietarul animalutului ' || i.animnume || ' al lui ' ||  
 i.propnum || ' ' || i.prenume || ' are factura ' || id || ' emisa la data de ' ||  
 i.data\_fisa || ', posibil pentru fisa medicala ' || i.id\_fisa);  
 EXCEPTION  
 when NO\_DATA\_FOUND then  
 DBMS\_OUTPUT.*PUT\_LINE*('Nu s-a emis nicio factura pentru proprietarul animalutului ' || i.animnume ||  
 ' al lui ' ||  
 i.propnum || ' ' || i.prenume || ' la data de ' || i.data\_fisa ||  
 ' desi a fost tratat, cu fisa medicala cu id ' || i.id\_fisa);  
 when TOO\_MANY\_ROWS then  
 DBMS\_OUTPUT.*PUT\_LINE*('Proprietarul animalutului ' || i.animnume || ' al lui ' ||  
 i.propnum || ' ' || i.prenume || ' are mai multe facturi emise la data de ' ||  
 i.data\_fisa);  
 END;  
 end loop;  
end;  
/  
  
BEGIN  
 *factura\_aferenta\_fisa*();  
end;  
/



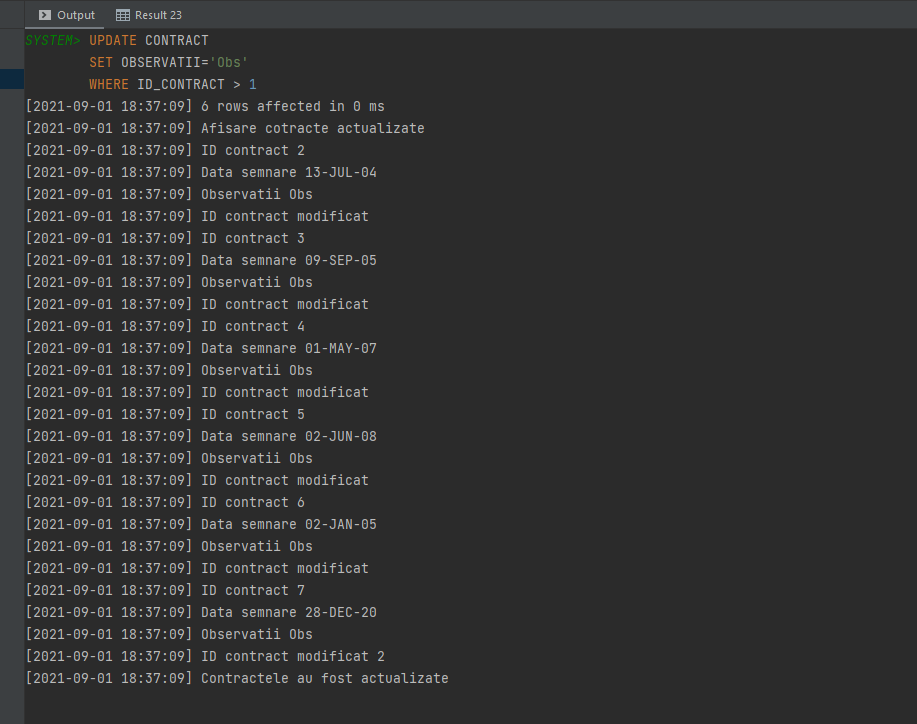
--10  
CREATE OR REPLACE TRIGGER trig\_contract  
 BEFORE DELETE  
 on CONTRACT  
BEGIN  
 RAISE\_APPLICATION\_ERROR(-20222, 'Contractele trebuie sa ramana pe veci in sistem');  
END;  
/  
  
CREATE OR REPLACE TRIGGER trig\_contract\_angajat  
 BEFORE DELETE  
 on CONTRACT\_ANGAJAT  
BEGIN  
 RAISE\_APPLICATION\_ERROR(-20222, 'Contractele trebuie sa ramana pe veci in sistem');  
END;  
/  
  
CREATE OR REPLACE TRIGGER trig\_contract\_producator  
 BEFORE DELETE  
 on CONTRACT\_PRODUCATOR  
BEGIN  
 RAISE\_APPLICATION\_ERROR(-20222, 'Contractele trebuie sa ramana pe veci in sistem');  
END;  
/  
  
DELETE  
from CONTRACT  
where id\_contract = 1;  
DELETE  
from CONTRACT\_ANGAJAT  
where id\_contract = 1;  
DELETE  
from CONTRACT\_PRODUCATOR  
where id\_contract = 4;



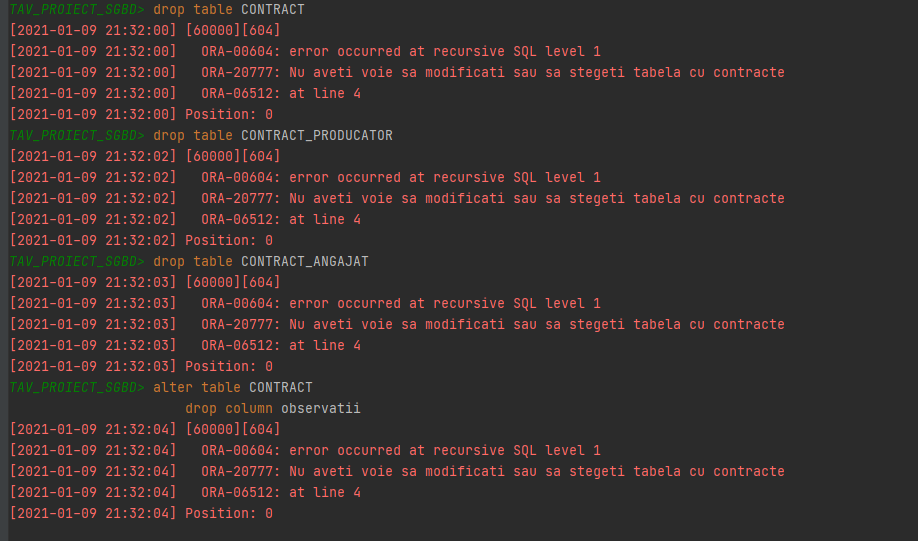
--11  
CREATE OR REPLACE TRIGGER trig\_tratament\_doctor  
 BEFORE INSERT OR UPDATE of id\_doctor  
 on FISA\_MEDICALA  
 FOR EACH ROW  
DECLARE  
 ok ANGAJAT.trateaza\_animale%type;  
BEGIN  
 select trateaza\_animale  
 into ok  
 from ANGAJAT  
 where id\_angajat = :NEW.id\_doctor;  
 if ok <> 1 then  
 RAISE\_APPLICATION\_ERROR(-20001, 'angajatul cu id-ul ' || :NEW.id\_doctor || ' nu poate trata animalute');  
 end if;  
EXCEPTION  
 when NO\_DATA\_FOUND then  
 RAISE\_APPLICATION\_ERROR(-20000, 'Nu exista un asemenea doctor');  
end;  
/  
  
INSERT INTO FISA\_MEDICALA(id\_fisa, id\_diagnostic, id\_animalut, id\_doctor, data\_fisa)  
VALUES (60, 1, 1, 3, sysdate);  
INSERT INTO FISA\_MEDICALA(id\_fisa, id\_diagnostic, id\_animalut, id\_doctor, data\_fisa)  
VALUES (60, 1, 1, 4, sysdate);  
  
CREATE OR REPLACE TRIGGER trig\_tratament\_zinastere  
 BEFORE INSERT OR UPDATE  
 on FISA\_MEDICALA  
 FOR EACH ROW  
DECLARE  
 zinastere ANIMALUT.zi\_nastere%type;  
BEGIN  
 select zi\_nastere  
 into zinastere  
 from ANIMALUT  
 where id\_animalut = :NEW.id\_animalut;  
  
 if :NEW.data\_fisa < zinastere then  
 RAISE\_APPLICATION\_ERROR(-20013, 'Nu poti trata un animalut inainte ca acesta sa se nasca');  
 end if;  
EXCEPTION  
 when NO\_DATA\_FOUND then  
 RAISE\_APPLICATION\_ERROR(-20000, 'Nu exista un astfel de animalut in sistem');  
end;  
/  
  
INSERT INTO FISA\_MEDICALA(id\_fisa, id\_diagnostic, id\_animalut, id\_doctor, data\_fisa)  
VALUES (100, 1, 1, 1, to\_date('01.01.1991', 'dd.mm.yyyy'));  
INSERT INTO FISA\_MEDICALA(id\_fisa, id\_diagnostic, id\_animalut, id\_doctor, data\_fisa)  
VALUES (100, 1, 100, 1, to\_date('01.01.1991', 'dd.mm.yyyy'));  
update FISA\_MEDICALA  
set data\_fisa=to\_date('01.01.1991', 'dd.mm.yyyy')  
where id\_fisa = 1;



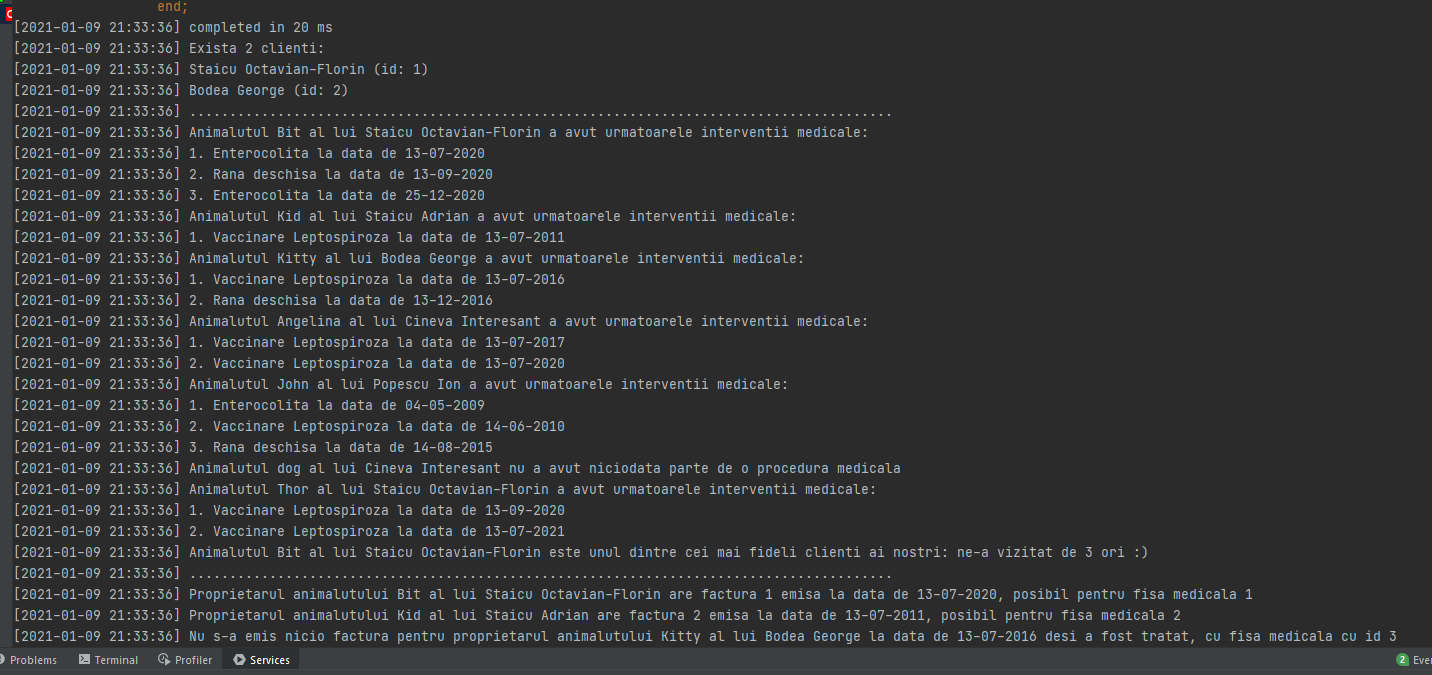
--compound trigger  
CREATE OR REPLACE TRIGGER trig\_update\_contracte  
 FOR UPDATE  
 ON CONTRACT COMPOUND TRIGGER  
  
BEFORE STATEMENT IS  
BEGIN  
 DBMS\_OUTPUT.PUT\_LINE('Afisare cotracte actualizate');  
END BEFORE STATEMENT;  
  
 AFTER EACH ROW IS  
 BEGIN  
 DBMS\_OUTPUT.PUT\_LINE('ID contract ' || :new.id\_contract);  
 DBMS\_OUTPUT.PUT\_LINE('Data semnare ' || :new.data\_semnare);  
 DBMS\_OUTPUT.PUT\_LINE('Observatii ' || :new.observatii);  
 DBMS\_OUTPUT.PUT\_LINE('ID contract modificat ' || :new.id\_contract\_modificat);  
 END AFTER EACH ROW;  
  
 AFTER STATEMENT IS  
 BEGIN  
 DBMS\_OUTPUT.PUT\_LINE('Contractele au fost actualizate');  
 END AFTER STATEMENT;  
 END trig\_update\_contracte;  
/  
  
UPDATE CONTRACT  
SET OBSERVATII='Obs'  
WHERE ID\_CONTRACT > 1;

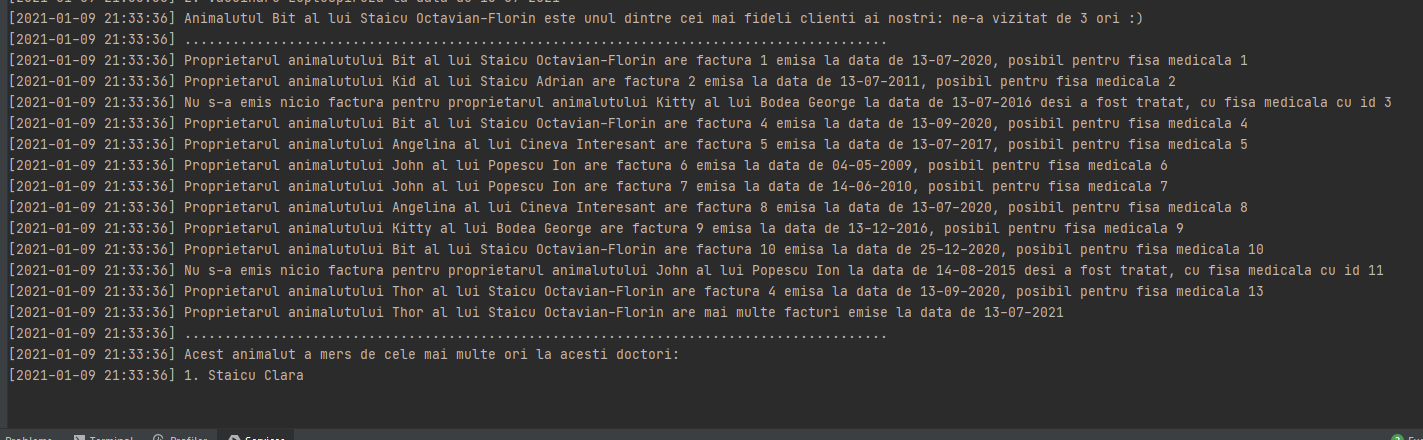


--12  
--blocheaza stergerea sau alterarea tabelelor cu contracte  
CREATE OR REPLACE TRIGGER trig\_ldd\_contract  
 BEFORE DROP OR ALTER  
 on schema  
begin  
 if lower(ORA\_DICT\_OBJ\_NAME) = lower('Contract') or lower(ORA\_DICT\_OBJ\_NAME) = lower('Contract\_angajat') or  
 lower(ORA\_DICT\_OBJ\_NAME) = lower('Contract\_producator') then  
 RAISE\_APPLICATION\_ERROR(-20777, 'Nu aveti voie sa modificati sau sa stegeti tabela cu contracte');  
 end if;  
end;  
/  
  
drop table CONTRACT;  
drop table CONTRACT\_PRODUCATOR;  
drop table CONTRACT\_ANGAJAT;  
alter table CONTRACT  
 drop column observatii;



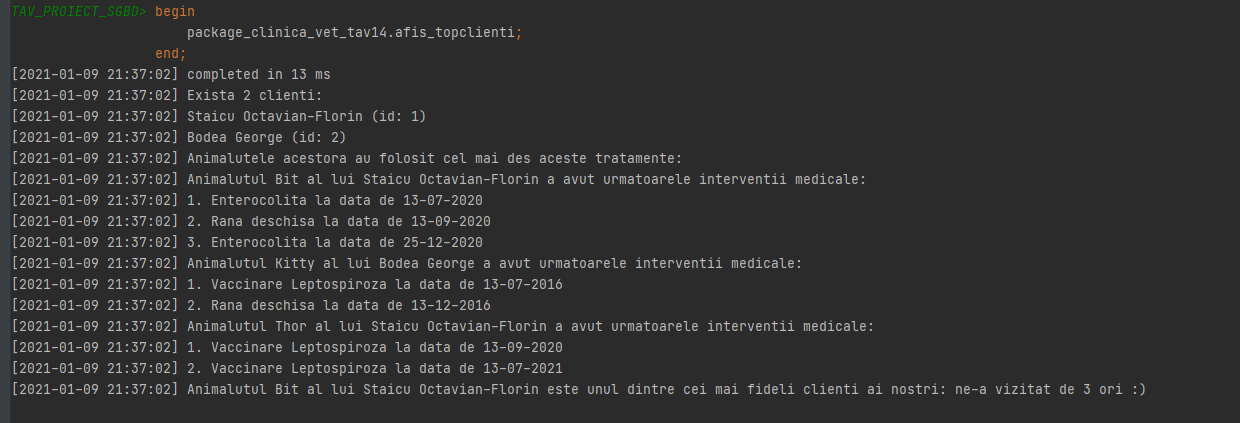
--13  
CREATE OR REPLACE PACKAGE package\_clinica\_vet\_tav13  
AS  
 PROCEDURE *afis\_doctor\_favorit*(id\_animal ANIMALUT.id\_animalut%type);  
 PROCEDURE *afis\_tratamente\_animalute*;  
 TYPE tab\_imb is TABLE OF NUMBER(10);  
 FUNCTION *topclienti*(top\_clienti out tab\_imb) return pls\_integer;  
 PROCEDURE *factura\_aferenta\_fisa*;  
END package\_clinica\_vet\_tav13;  
/  
  
CREATE OR REPLACE PACKAGE BODY package\_clinica\_vet\_tav13  
AS  
 PROCEDURE *afis\_doctor\_favorit*(id\_animal ANIMALUT.id\_animalut%type)  
 IS  
 TYPE tab\_ind IS TABLE OF PLS\_INTEGER INDEX BY PLS\_INTEGER;  
 cnt tab\_ind;  
 index\_max pls\_integer;  
 ok binary\_integer := 0;  
 id pls\_integer;  
 num ANGAJAT.nume%type;  
 prenum ANGAJAT.prenume%type;  
 co pls\_integer := 0;  
 BEGIN  
 for i in (select fm.id\_doctor  
 from FISA\_MEDICALA fm  
 where fm.id\_animalut = id\_animal)  
 loop  
 begin  
 cnt(i.id\_doctor) := cnt(i.id\_doctor) + 1;  
 if ok = 1 then  
 if cnt(i.id\_doctor) > cnt(index\_max) then  
 index\_max := i.id\_doctor;  
 end if;  
 end if;  
 EXCEPTION  
 when NO\_DATA\_FOUND then  
 cnt(i.id\_doctor) := 1;  
 if ok = 0 then  
 ok := 1;  
 index\_max := i.id\_doctor;  
 end if;  
 end;  
 end loop;  
 if ok = 0 then  
 DBMS\_OUTPUT.*PUT\_LINE*('Acest animalut nu are nicio fisa medicala in sistem');  
 else  
 id := cnt.FIRST;  
 DBMS\_OUTPUT.*PUT\_LINE*('Acest animalut a mers de cele mai multe ori la acesti doctori:');  
 loop  
 exit when id is null;  
 if cnt(id) = cnt(index\_max) then  
 select a.nume, a.prenume  
 into num, prenum  
 from ANGAJAT a  
 where a.id\_angajat = id;  
 co := co + 1;  
 DBMS\_OUTPUT.*PUT\_LINE*(co || '. ' || num || ' ' || prenum);  
 end if;  
 id := cnt.NEXT(id);  
 end loop;  
 end if;  
 END *afis\_doctor\_favorit*;  
  
 PROCEDURE *afis\_tratamente\_animalute* IS  
 TYPE refcursor IS REF CURSOR;  
 cursor mycursor is select a.nume,  
 p.nume,  
 p.prenume,  
 cursor (select diag.descriere, fm.data\_fisa  
 from FISA\_MEDICALA fm,  
 DIAGNOSTIC diag  
 where fm.id\_diagnostic = diag.id\_diagnostic  
 and a.id\_animalut = fm.id\_animalut  
 )  
 from ANIMALUT a,  
 PROPRIETAR p  
 where a.id\_proprietar = p.id\_proprietar;  
 v\_cursor refcursor;  
 nume\_animalut ANIMALUT.nume%type;  
 nume\_proprietar PROPRIETAR.nume%type;  
 prenume\_proprietar PROPRIETAR.prenume%type;  
 diag DIAGNOSTIC.descriere%type;  
 data\_diagnostic FISA\_MEDICALA.data\_fisa%type;  
 nume\_animalut\_max ANIMALUT.nume%type;  
 nume\_proprietar\_max PROPRIETAR.nume%type;  
 prenume\_proprietar\_max PROPRIETAR.prenume%type;  
 co\_max pls\_integer := 0;  
 faliment EXCEPTION;  
 BEGIN  
 open mycursor;  
 loop  
 fetch mycursor into nume\_animalut,nume\_proprietar,prenume\_proprietar,v\_cursor;  
 exit when mycursor%notfound;  
 fetch v\_cursor into diag, data\_diagnostic;  
 if v\_cursor%notfound then  
 DBMS\_OUTPUT.*PUT\_LINE*(  
 'Animalutul ' || nume\_animalut || ' al lui ' || nume\_proprietar || ' ' ||  
 prenume\_proprietar ||  
 ' nu a avut niciodata parte de o procedura medicala');  
 DBMS\_OUTPUT.*NEW\_LINE*();  
 else  
 DBMS\_OUTPUT.*PUT\_LINE*(  
 'Animalutul ' || nume\_animalut || ' al lui ' || nume\_proprietar || ' ' ||  
 prenume\_proprietar ||  
 ' a avut urmatoarele interventii medicale:');  
 loop  
 exit when v\_cursor%notfound;  
 DBMS\_OUTPUT.*PUT\_LINE*(v\_cursor%rowcount || '. ' || diag || ' la data de ' || data\_diagnostic);  
 fetch v\_cursor into diag, data\_diagnostic;  
 end loop;  
 if v\_cursor%rowcount > co\_max then  
 co\_max := v\_cursor%rowcount;  
 nume\_animalut\_max := nume\_animalut;  
 nume\_proprietar\_max := nume\_proprietar;  
 prenume\_proprietar\_max := prenume\_proprietar;  
 end if;  
 DBMS\_OUTPUT.*NEW\_LINE*();  
 end if;  
 end loop;  
 close mycursor;  
 if co\_max = 0 then  
 raise faliment;  
 end if;  
 DBMS\_OUTPUT.*PUT\_LINE*('Animalutul ' || nume\_animalut\_max || ' al lui ' || nume\_proprietar\_max || ' ' ||  
 prenume\_proprietar\_max ||  
 ' este unul dintre cei mai fideli clienti ai nostri: ne-a vizitat de ' || co\_max ||  
 ' ori :)');  
  
 EXCEPTION  
 when faliment then  
 DBMS\_OUTPUT.*PUT\_LINE*('Nu avem niciun animalut care s-a tratat la noi...');  
  
 END *afis\_tratamente\_animalute*;  
  
 FUNCTION *topclienti*(top\_clienti out tab\_imb) return pls\_integer  
 IS  
 TYPE refcursor IS REF CURSOR;  
 cursor mycursor is select f.id\_proprietar,  
 cursor (select c.suma\_platita  
 from CHITANTA c  
 where c.id\_factura = f.id\_factura  
 )  
 from FACTURA\_PROPRIETAR f,  
 PROPRIETAR p  
 where f.id\_proprietar = p.id\_proprietar  
 order by p.id\_proprietar;  
 v\_cursor refcursor;  
 id FACTURA\_PROPRIETAR.id\_factura%type := null;  
 suma pls\_integer := 0;  
 total\_suma pls\_integer := 0;  
 max\_suma pls\_integer := 0;  
 last\_id PROPRIETAR.id\_proprietar%type := null;  
 co pls\_integer := 0;  
 BEGIN  
 top\_clienti := tab\_imb();  
 open mycursor;  
 loop  
 fetch mycursor into id, v\_cursor;  
 if last\_id is null then  
 last\_id := id;  
 else  
 if mycursor%notfound or id <> last\_id then  
 if total\_suma > max\_suma then  
 top\_clienti.DELETE;  
 top\_clienti.EXTEND;  
 co := 1;  
 top\_clienti(1) := last\_id;  
 max\_suma := total\_suma;  
 elsif total\_suma = max\_suma then  
 top\_clienti.EXTEND;  
 co := co + 1;  
 top\_clienti(co) := last\_id;  
 end if;  
 last\_id := id;  
 total\_suma := 0;  
 end if;  
 end if;  
 exit when mycursor%notfound;  
 loop  
 fetch v\_cursor into suma;  
 exit when v\_cursor%notfound;  
 total\_suma := total\_suma + suma;  
 end loop;  
 end loop;  
 close mycursor;  
 return co;  
  
 END *topclienti*;  
  
 PROCEDURE *factura\_aferenta\_fisa* IS  
 id FACTURA\_PROPRIETAR.id\_factura%type;  
 BEGIN  
 for i in (select p.id\_proprietar, p.nume propnum, p.prenume, a.nume animnume, fm.data\_fisa, fm.id\_fisa  
 from FISA\_MEDICALA fm,  
 ANIMALUT a,  
 PROPRIETAR p  
 where fm.id\_animalut = a.id\_animalut  
 and a.id\_proprietar = p.id\_proprietar)  
 loop  
 BEGIN  
 select fp.id\_factura  
 into id  
 from FACTURA\_PROPRIETAR fp,  
 FACTURA f  
 where fp.id\_proprietar = i.id\_proprietar  
 and fp.id\_factura = f.id\_factura  
 and i.data\_fisa = f.data\_emitere;  
 DBMS\_OUTPUT.*PUT\_LINE*('Proprietarul animalutului ' || i.animnume || ' al lui ' ||  
 i.propnum || ' ' || i.prenume || ' are factura ' || id ||  
 ' emisa la data de ' ||  
 i.data\_fisa || ', posibil pentru fisa medicala ' || i.id\_fisa);  
 EXCEPTION  
 when NO\_DATA\_FOUND then  
 DBMS\_OUTPUT.*PUT\_LINE*(  
 'Nu s-a emis nicio factura pentru proprietarul animalutului ' || i.animnume ||  
 ' al lui ' ||  
 i.propnum || ' ' || i.prenume || ' la data de ' || i.data\_fisa ||  
 ' desi a fost tratat, cu fisa medicala cu id ' || i.id\_fisa);  
 when TOO\_MANY\_ROWS then  
 DBMS\_OUTPUT.*PUT\_LINE*('Proprietarul animalutului ' || i.animnume || ' al lui ' ||  
 i.propnum || ' ' || i.prenume ||  
 ' are mai multe facturi emise la data de ' ||  
 i.data\_fisa);  
 END;  
 end loop;  
 end *factura\_aferenta\_fisa*;  
  
END package\_clinica\_vet\_tav13;  
/  
  
begin  
 declare  
 co pls\_integer;  
 id NUMBER(10);  
 mytab package\_clinica\_vet\_tav13.tab\_imb;  
 num PROPRIETAR.nume%type;  
 prenum PROPRIETAR.prenume%type;  
 begin  
 co := package\_clinica\_vet\_tav13.*TOPCLIENTI*(mytab);  
 DBMS\_OUTPUT.*PUT\_LINE*('Exista ' || co || ' clienti:');  
  
 id := mytab.FIRST;  
 loop  
 exit when id is null;  
 select nume, prenume  
 into num, prenum  
 from PROPRIETAR  
 where id\_proprietar = mytab(id);  
 DBMS\_OUTPUT.*PUT\_LINE*(num || ' ' || prenum || ' (id: ' || mytab(id) || ')');  
 id := mytab.NEXT(id);  
 end loop;  
 end;  
 DBMS\_OUTPUT.*PUT\_LINE*('........................................................................................');  
 package\_clinica\_vet\_tav13.*afis\_tratamente\_animalute*();  
 DBMS\_OUTPUT.*PUT\_LINE*('........................................................................................');  
 package\_clinica\_vet\_tav13.*factura\_aferenta\_fisa*();  
 DBMS\_OUTPUT.*PUT\_LINE*('........................................................................................');  
 package\_clinica\_vet\_tav13.*afis\_doctor\_favorit*(1);  
end;  
/





--14  
CREATE OR REPLACE PACKAGE package\_clinica\_vet\_tav14 AS

TYPE tab\_imb is TABLE OF NUMBER(10); --A se citi obs din josul paginii  
 top\_clienti tab\_imb := tab\_imb();  
 FUNCTION *topclienti* RETURN PLS\_INTEGER;  
 PROCEDURE *afis\_topclienti*;  
 PROCEDURE *afis\_tratamente\_animalute*;  
 TYPE refcursor IS REF CURSOR;  
 type rec is record  
 (  
 nume\_animalut ANIMALUT.nume%type,  
 nume\_proprietar PROPRIETAR.nume%type,  
 prenume\_proprietar PROPRIETAR.prenume%type  
 );  
END package\_clinica\_vet\_tav14;  
/  
  
CREATE OR REPLACE PACKAGE BODY package\_clinica\_vet\_tav14 AS  
  
 FUNCTION *topclienti* RETURN PLS\_INTEGER  
 IS  
 cursor mycursor is select f.id\_proprietar,  
 cursor (select c.suma\_platita  
 from CHITANTA c  
 where c.id\_factura = f.id\_factura  
 )  
 from FACTURA\_PROPRIETAR f,  
 PROPRIETAR p  
 where f.id\_proprietar = p.id\_proprietar  
 order by p.id\_proprietar;  
 v\_cursor refcursor;  
 id FACTURA\_PROPRIETAR.id\_factura%type := null;  
 suma pls\_integer := 0;  
 total\_suma pls\_integer := 0;  
 max\_suma pls\_integer := 0;  
 last\_id PROPRIETAR.id\_proprietar%type := null;  
 co pls\_integer := 0;  
 BEGIN  
 open mycursor;  
 loop  
 fetch mycursor into id, v\_cursor;  
 if last\_id is null then  
 last\_id := id;  
 else  
 if id <> last\_id or mycursor%notfound then  
 if total\_suma > max\_suma then  
 top\_clienti.DELETE;  
 top\_clienti.EXTEND;  
 co := 1;  
 top\_clienti(1) := last\_id;  
 max\_suma := total\_suma;  
 elsif total\_suma = max\_suma then  
 top\_clienti.EXTEND;  
 co := co + 1;  
 top\_clienti(co) := last\_id;  
 end if;  
 last\_id := id;  
 total\_suma := 0;  
 end if;  
 end if;  
 exit when mycursor%notfound;  
 loop  
 fetch v\_cursor into suma;  
 exit when v\_cursor%notfound;  
 total\_suma := total\_suma + suma;  
 end loop;  
 end loop;  
 close mycursor;  
 return co;  
  
 END *topclienti*;  
  
 PROCEDURE *afis\_tratamente\_animalute* is  
 cursor mycursor is select a.nume,  
 p.nume,  
 p.prenume,  
 cursor (select diag.descriere, fm.data\_fisa  
 from FISA\_MEDICALA fm,  
 DIAGNOSTIC diag  
 where fm.id\_diagnostic = diag.id\_diagnostic  
 and a.id\_animalut = fm.id\_animalut  
 )  
 from ANIMALUT a,  
 PROPRIETAR p  
 where a.id\_proprietar = p.id\_proprietar  
 and p.id\_proprietar in (select \* from table (top\_clienti));  
 v\_cursor refcursor;  
 diag DIAGNOSTIC.descriere%type;  
 data\_diagnostic FISA\_MEDICALA.data\_fisa%type;  
 myrec rec;  
 nume\_animalut\_max ANIMALUT.nume%type;  
 nume\_proprietar\_max PROPRIETAR.nume%type;  
 prenume\_proprietar\_max PROPRIETAR.prenume%type;  
 co\_max pls\_integer := 0;  
 faliment EXCEPTION;  
 BEGIN  
 open mycursor;  
 loop  
 fetch mycursor into myrec.nume\_animalut, myrec.nume\_proprietar, myrec.prenume\_proprietar,v\_cursor;  
 exit when mycursor%notfound;  
 fetch v\_cursor into diag, data\_diagnostic;  
 if v\_cursor%notfound then  
 DBMS\_OUTPUT.*PUT\_LINE*(  
 'Animalutul ' || myrec.nume\_animalut || ' al lui ' || myrec.nume\_proprietar || ' ' ||  
 myrec.prenume\_proprietar ||  
 ' nu a avut niciodata parte de o procedura medicala');  
 DBMS\_OUTPUT.*NEW\_LINE*();  
 else  
 DBMS\_OUTPUT.*PUT\_LINE*(  
 'Animalutul ' || myrec.nume\_animalut || ' al lui ' || myrec.nume\_proprietar || ' ' ||  
 myrec.prenume\_proprietar ||  
 ' a avut urmatoarele interventii medicale:');  
 loop  
 exit when v\_cursor%notfound;  
 DBMS\_OUTPUT.*PUT\_LINE*(v\_cursor%rowcount || '. ' || diag || ' la data de ' || data\_diagnostic);  
 fetch v\_cursor into diag, data\_diagnostic;  
 end loop;  
 if v\_cursor%rowcount > co\_max then  
 co\_max := v\_cursor%rowcount;  
 nume\_animalut\_max := myrec.nume\_animalut;  
 nume\_proprietar\_max := myrec.nume\_proprietar;  
 prenume\_proprietar\_max := myrec.prenume\_proprietar;  
 end if;  
 DBMS\_OUTPUT.*NEW\_LINE*();  
 end if;  
 end loop;  
 close mycursor;  
 if co\_max = 0 then  
 raise faliment;  
 end if;  
 DBMS\_OUTPUT.*PUT\_LINE*('Animalutul ' || nume\_animalut\_max || ' al lui ' || nume\_proprietar\_max || ' ' ||  
 prenume\_proprietar\_max ||  
 ' este unul dintre cei mai fideli clienti ai nostri: ne-a vizitat de ' || co\_max ||  
 ' ori :)');  
  
 EXCEPTION  
 when faliment then  
 DBMS\_OUTPUT.*PUT\_LINE*('Nu avem niciun animalut care s-a tratat la noi...');  
  
 END *afis\_tratamente\_animalute*;  
  
 PROCEDURE *afis\_topclienti* IS  
 co pls\_integer;  
 id NUMBER(10);  
 num PROPRIETAR.nume%type;  
 prenum PROPRIETAR.prenume%type;  
 begin  
 co := package\_clinica\_vet\_tav14.*TOPCLIENTI*;  
 DBMS\_OUTPUT.*PUT\_LINE*('Exista ' || co || ' clienti:');  
 id := top\_clienti.FIRST;  
 loop  
 exit when id is null;  
 select nume, prenume  
 into num, prenum  
 from PROPRIETAR  
 where id\_proprietar = top\_clienti(id);  
 DBMS\_OUTPUT.*PUT\_LINE*(num || ' ' || prenum || ' (id: ' || top\_clienti(id) || ')');  
 id := top\_clienti.NEXT(id);  
 end loop;  
 DBMS\_OUTPUT.*PUT\_LINE*('Animalutele acestora au folosit cel mai des aceste tratamente:');  
 package\_clinica\_vet\_tav14.*AFIS\_TRATAMENTE\_ANIMALUTE*();  
 end *afis\_topclienti*;  
  
END package\_clinica\_vet\_tav14;  
/



Obs. Pachetul a fost rulat pe 1 septembrie 2021 pe Windows folosind Oracle Database 18c Express Edition și a rulat. Pe 9 ianuaria 2021 (când a fost făcut screenshot-ul) codul a fost testat pe Linux cu versiunea o11g si avea o eroare dubioasă la rulare („Object does not exist or is marked for delete”). Soluția la acel moment a fost să exclud din pachet linia TYPE tab\_imb is TABLE OF NUMBER(10); și să folosesc definiția acestui tip de la un excerțiu anterior. Aparent, aceasta a fost o problemă de la versiunea o11g, pentru că acum am putut reintroduce această linie în pachet și funcționeaza ok.

CREATE OR REPLACE PACKAGE package\_clinica\_vet\_utilitare  
IS  
  
 type t\_idx is table of pls\_integer index by pls\_integer;  
 type array is VARRAY (1000) of pls\_integer;  
 type myrec\_datorii is record  
 (  
 id\_prop PROPRIETAR.id\_proprietar%type,  
 nume\_prop PROPRIETAR.nume%type,  
 prenume\_prop PROPRIETAR.prenume%type,  
 id\_fac FACTURA\_PROPRIETAR.id\_factura%type,  
 total\_fac FACTURA.total\_factura%type,  
 suma\_platita CHITANTA.suma\_platita%type  
 );  
 type datorii\_imb is table of myrec\_datorii;  
 FUNCTION *timp\_minim\_de\_asteptare*(v in out array) RETURN t\_idx;  
 PROCEDURE *afis\_datorii*;  
END package\_clinica\_vet\_utilitare;  
/  
  
CREATE OR REPLACE PACKAGE BODY package\_clinica\_vet\_utilitare  
IS  
  
 FUNCTION *timp\_minim\_de\_asteptare*(v in out array) RETURN t\_idx  
 Is --Primind un vector cu timpul in care dureaza fiecare procedura medicala pt oamenii din acea zi  
 --se stabileste in ce ordine vor intra in cabinet, ai timpul mediu de asteptare sa fie minim  
 poz\_min pls\_integer;  
 sol t\_idx;  
 co pls\_integer := 0;  
 BEGIN  
 if v.FIRST is null then  
 raise NO\_DATA\_FOUND;  
 end if;  
 loop  
 poz\_min := 1;  
 for i in v.FIRST..v.LAST  
 loop  
 if v(i) < v(poz\_min) then  
 poz\_min := i;  
 end if;  
 end loop;  
 exit when v(poz\_min) = 1001;  
 co := co + 1;  
 sol(poz\_min) := co;  
 v(poz\_min) := 1001;  
 end loop;  
 return sol;  
 EXCEPTION  
 when NO\_DATA\_FOUND then  
 return sol;  
 END *timp\_minim\_de\_asteptare*;  
  
 PROCEDURE *afis\_datorii* is  
 my\_t datorii\_imb;  
 s pls\_integer;  
 s\_dat pls\_integer := 0;  
 idx pls\_integer;  
 last\_prop PROPRIETAR.id\_proprietar%type := null;  
 Begin  
  
 select p.id\_proprietar,  
 p.nume,  
 p.prenume,  
 fp.id\_factura,  
 f.total\_factura,  
 sum(c.suma\_platita)  
 bulk collect  
 into my\_t  
 from PROPRIETAR p,  
 FACTURA\_PROPRIETAR fp,  
 FACTURA f,  
 CHITANTA c  
 where p.id\_proprietar = fp.id\_proprietar  
 and fp.id\_factura = f.id\_factura  
 and f.id\_factura = c.id\_factura(+)  
 group by p.id\_proprietar, p.nume, p.prenume, fp.id\_factura, f.total\_factura  
 order by p.id\_proprietar;  
 idx := my\_t.FIRST;  
 loop  
 if last\_prop is null then  
 if idx is not null then  
 last\_prop := my\_t(idx).id\_prop;  
 DBMS\_OUTPUT.*PUT\_LINE*(  
 'Proprietarul ' || my\_t(idx).nume\_prop || ' ' || my\_t(idx).prenume\_prop || ' (id:' ||  
 my\_t(idx).id\_prop || '):');  
 end if;  
 else  
 if idx is null or my\_t(idx).id\_prop <> last\_prop then  
 DBMS\_OUTPUT.*PUT\_LINE*('Total de plata: ' || s\_dat);  
 DBMS\_OUTPUT.*NEW\_LINE*();  
 s\_dat := 0;  
 if idx is not null then  
 last\_prop := my\_t(idx).id\_prop;  
 DBMS\_OUTPUT.*PUT\_LINE*(  
 'Proprietarul ' || my\_t(idx).nume\_prop || ' ' || my\_t(idx).prenume\_prop ||  
 ' (id:' ||  
 my\_t(idx).id\_prop || '):');  
 end if;  
 end if;  
 end if;  
 exit when idx is null;  
 s := my\_t(idx).total\_fac - nvl(my\_t(idx).suma\_platita, 0);  
 s\_dat := s\_dat + s;  
 if s > 0 then  
 DBMS\_OUTPUT.*PUT\_LINE*('La factura ' || my\_t(idx).id\_fac || ' mai are de platit: ' || s);  
 end if;  
 idx := my\_t.NEXT(idx);  
 end loop;  
  
 end *afis\_datorii*;  
  
END package\_clinica\_vet\_utilitare;  
/  
  
begin  
 package\_clinica\_vet\_utilitare.*afis\_datorii*;  
end;  
/  
  
DECLARE  
 v package\_clinica\_vet\_utilitare.array := package\_clinica\_vet\_utilitare.array(7,1,2,5,3);  
 sol package\_clinica\_vet\_utilitare.t\_idx;  
BEGIN  
  
 sol := package\_clinica\_vet\_utilitare.*timp\_minim\_de\_asteptare*(v);  
 if sol.FIRST is not null then  
 for i in sol.FIRST..sol.LAST  
 loop  
 DBMS\_OUTPUT.*PUT\_LINE*(i || '. ' || sol(i));  
 end loop;  
 else  
 DBMS\_OUTPUT.*PUT\_LINE*('Niciun Client Programat');  
 end if;  
end;  
/

