# Proiect SGBD – Salon de infrumusetare

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

**Modelul de date:**

Modelul de date va gestiona informaţii legate de organizarea si funcţionarea unui lant de saloane de infrumusetare. Exista manageri care se ocupă de buna desfasurare a unei locatii. Fiecare locatie are un contabil, angajat al unei firme de contabilitate si angajati pe diferite domenii(make-up artist, hair stylist, manichiurist, cosmetolog).

Clientii au programari la diferite servicii prestate de salariati, clientii pot avea un card de fidelitate. Pentru prestarea serviciilor este nevoie de anumite produse(fixativ, paleta de ochi, pensule etc) furnizate de anumiti furnizori.

De asemenea, modelul analizeaza informaţii legate de adresa locatiilor, firmele de contabilitate, a contabililor, a managerilor, a salariatilor, a clientilor, a firmelor de furnizare, a produselor si a serviciilor.

**Modelul de date respecta anumite restrictii de functionare:**

* O locatie poate fi condusa de un manager.
* Un manager poate conduce mai multe locatii.
* Un local poate avea un contabil.
* Un contabil poate tine contabilitatea mai multor locatii.
* Un salariat poate lucra la o singura locatie.
* O locatie poate avea mai multi salariatii.
* Un client poate avea mai multe programari.
* O programare are un singur client.
* O programare se face pentru un singur serviciu.
* Un serviciu poate avea mai multe programari.
* Un client are doar un abonament.
* Un abonament poate fi cumparat de un singur client.
* Un serviciu poate fi prestat de mai multi salariati.
* Un salariat poate presta mai multe servicii.
* Un furnizor poate furniza mai multe produse.
* Un produs este furnizat de un singur furnizor.
* Un serviciu poate utiliza mai multe produse.
* Un produs poate fi utilizat pentru mai multe servicii.

**Entitati:**

Pentru modelul de date referitor saloanele de infrumusetare , structurile MANAGER, CONTABIL, FIRMA\_CONTABILITATE, LOCATIE, SALARIAT, SERVICII, PRODUSE, FURNIZOR, CLIENT, CARD\_FIDELITATE reprezinta entitati.

**Relatii:**

MANAGER\_conduce\_LOCATIE = relatie ce leaga entitatile MANAGER si LOCATIE (ce manager conduce ce locatie). Deoarece intr-o locatie poate sa fie doar un manager,un manager poate conduce mai multe locatii relatia are cardinalitatea minima 1:1 si cardinalitatea maxima n:1.

CONTABIL\_are\_LOCATIE= relatie ce leaga entitatile CONTABIL si LOCATIE (ce contabil tine contabilitatea carei locatii). Deoarece intr-o locatie poate sa fie doar un contabil, un contabil poate lucra la mai multe locatii relatia are cardinalitatea minima 1:1 si cardinalitatea maxima n:1.

SALARIAT\_lucreaza\_la\_LOCATIE= relatie ce leaga entitatile SALARIAT si LOCATIE (la ce locatie lucreaza un salariat). Deoarece intr-o locatie poat fi mai multi salariati si un salariat lucreaza la o singura locatie , relatia are cardinalitatea minima 1:1 si cardinalitatea maxima n:1.

PROGRAMARE\_la\_LOCATIE= relatie ce leaga entitatile PROGRAMARE si LOCATIE (unde are loc programarea). Deoarece intr-o locatie poate sa fie mai multe programari, relatia are cardinalitatea minima 0:1 si cardinalitatea maxima n:1.

CONTABIL\_lucreaza\_la\_FIRMA\_CONTABILITATE=relatie dintre entitatile CONTABIL si FIRMA\_CONTABILITATE (la ce firma este angajat un contabil). Relatia are cardinalitatea minima 1:1 si cardinalitatea maxima n:1.

SALARIAT\_presteaza\_SERVICII=relatie de tip many-to-many dintre entitatile SALARIAT si SERVICII (ce salariat presteaza ce servicii). Relatia are cardinalitatea minima 1:1 si cardinalitatea maxima m:n.

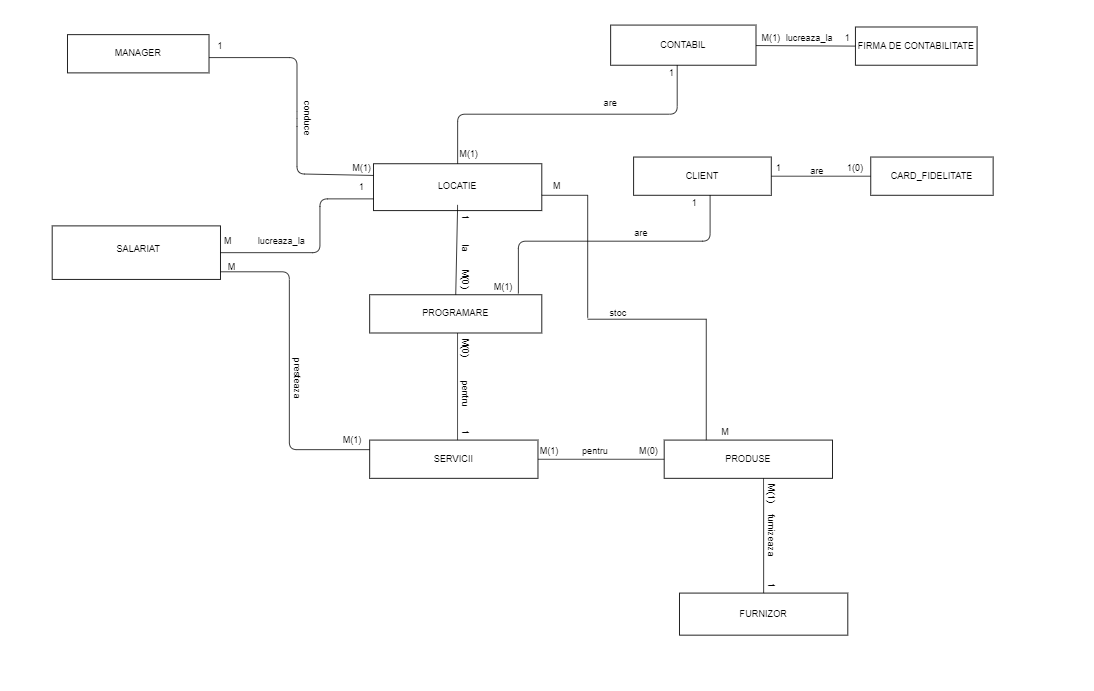
PROGRAMARE\_pentru\_SERVICII=relatie ce leaga entitatile PROGRAMARE si SERVICII(programarea pentru serviciu). Relatia are cardinalitatea minima 0:1 si cardinalitatea maxima n:1.

PRODUSE\_pentru\_SERVICII= relatie de tip many-to-many ce leaga entitatile PRODUSE si SERVICII(pentru ce servicii se folosesc produsele). Relatia are cardinalitatea minima 0:1 si cardinalitatea maxima m:n.

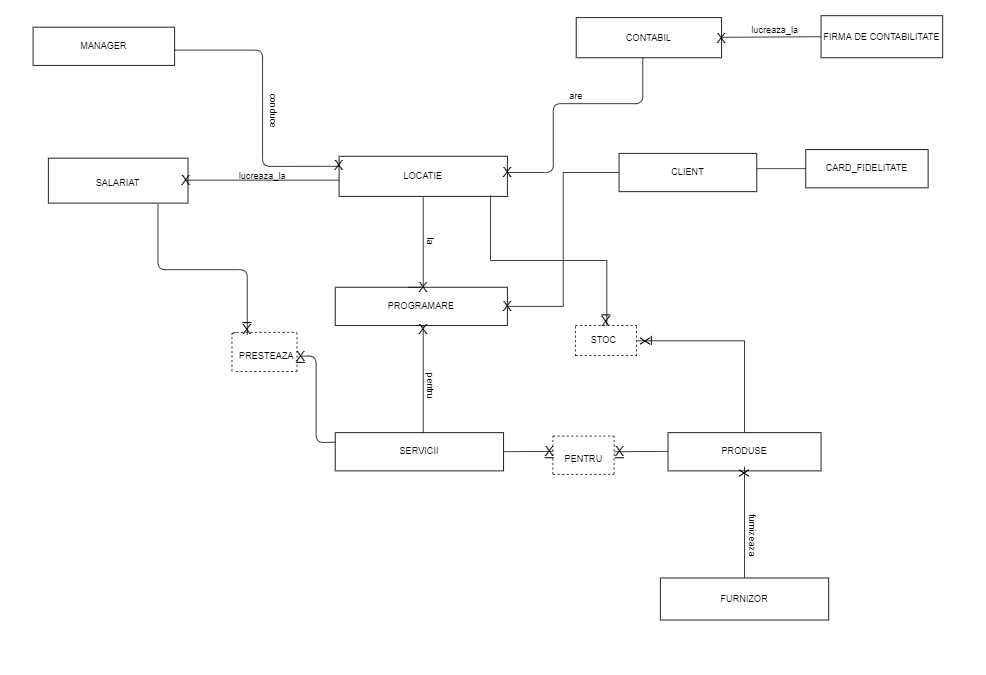
FURNIZOR\_furnizeaza\_PRODUSE=relatie ce leaga entitatile FURNIZOR si PRODUSE. Relatia are cardinalitatea minima 1:1 si cardinalitatea maxima 1:n.

CLIENT\_are\_ABONAMENT= relatie ce leaga entitatile CLIENT si ABONAMENT. Relatia are cardinalitatea minima 1:0 si cardinalitatea maxima 1:1.

CLIENT \_are\_PROGRAMARE= relatie ce leaga entitatile CLIENT si PROGRAMARE. Relatia are cardinalitatea minima 1:1 si cardinalitatea maxima 1:m.

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

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



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

CREATE TABLE FURNIZOR

( cod\_furnizor NUMBER(4) PRIMARY KEY,nume VARCHAR2(25) CONSTRAINT nume\_mm NOT NULL)

ALTER TABLE furnizor

ADD nr\_tel NUMBER(10);

CREATE TABLE produse

( cod\_produs NUMBER(5) PRIMARY KEY, denumire VARCHAR2(25) CONSTRAINT denumire\_nn NOT NULL,cod\_furnizor NUMBER(4) NOT NULL,

CONSTRAINT cod\_furnizor\_fk FOREIGN KEY (cod\_furnizor)REFERENCES FURNIZOR(cod\_furnizor) )

CREATE TABLE manageri

( cod\_manager NUMBER(3) PRIMARY KEY , nume VARCHAR(25) CONSTRAINT nume\_rr NOT NULL, prenume VARCHAR(25) CONSTRAINT

prenume\_rr NOT NULL, salariu NUMBER(12))

CREATE TABLE client

( cod\_client NUMBER(3) PRIMARY KEY , nume VARCHAR(25) CONSTRAINT nume NOT NULL, prenume VARCHAR(25) CONSTRAINT

prenume NOT NULL, varsta NUMBER(2))

CREATE TABLE firma\_contabilitate

(cod\_firma\_contabilitate NUMBER(4) PRIMARY KEY, nume VARCHAR(25))

CREATE TABLE contabil

(cod\_contabil NUMBER(2) PRIMARY KEY, nume VARCHAR(25)CONSTRAINT nume\_pp NOT NULL , prenume VARCHAR(25)

CONSTRAINT prenume\_pp NOT NULL,cod\_firma\_contabilitate NUMBER(4) NOT NULL,CONSTRAINT cod\_firma\_contabilitate\_fk

FOREIGN KEY (cod\_firma\_contabilitate)REFERENCES firma\_contabilitate(cod\_firma\_contabilitate))

CREATE TABLE locatie

(cod\_locatie NUMBER(4) PRIMARY KEY, adresa VARCHAR(50) CONSTRAINT adresa NOT NULL, telefon NUMBER (10),

cod\_manager NUMBER(3) NOT NULL,CONSTRAINT cod\_manager\_fk FOREIGN KEY (cod\_manager)REFERENCES manageri(cod\_manager),

cod\_contabil NUMBER(2) NOT NULL,CONSTRAINT cod\_contabil\_fk FOREIGN KEY (cod\_contabil)REFERENCES contabil(cod\_contabil));

CREATE TABLE servicii

(cod\_servicii NUMBER(5) PRIMARY KEY,

tip VARCHAR2(20) CONSTRAINT tip\_nn NOT NULL, pret NUMBER(3))

CREATE TABLE programare

(cod\_programare NUMBER(4) PRIMARY KEY, data\_ora TIMESTAMP, cod\_locatie NUMBER(4) NOT NULL,

CONSTRAINT cod\_locatie\_fk FOREIGN KEY (cod\_locatie)REFERENCES locatie(cod\_locatie), cod\_servicii NUMBER(5) NOT NULL,

CONSTRAINT cod\_servicii\_fk FOREIGN KEY (cod\_servicii)REFERENCES servicii(cod\_servicii), cod\_client NUMBER(3) NOT NULL,

CONSTRAINT cod\_client\_ffk FOREIGN KEY (cod\_client)REFERENCES client(cod\_client));

CREATE TABLE card\_fidelitate

(cod\_card NUMBER(4) PRIMARY KEY, data\_activare DATE, data\_exp DATE, cod\_client NUMBER(3) NOT NULL,

CONSTRAINT cod\_client\_fk FOREIGN KEY (cod\_client)REFERENCES client(cod\_client),

CONSTRAINT cod\_client\_um UNIQUE (cod\_client))

CREATE TABLE salariat

(cod\_salariat NUMBER(3) PRIMARY KEY,tip VARCHAR2(25)

CHECK (tip in('cosmetolog', 'makeup\_artist', 'manichiurist', 'hair\_stylist')) CONSTRAINT tip\_zz NOT NULL,

cod\_locatie NUMBER(4) NOT NULL,

CONSTRAINT cod\_locaties\_fk FOREIGN KEY (cod\_locatie) REFERENCES locatie(cod\_locatie),

nume VARCHAR(25) CONSTRAINT nume\_zz NOT NULL, prenume VARCHAR(25) CONSTRAINT prenume\_zz NOT NULL ,

salariu NUMBER(4) CONSTRAINT salariu\_zz NOT NULL)

CREATE TABLE stoc

(cod\_locatie NUMBER(4) NOT NULL REFERENCES locatie(cod\_locatie),

cod\_produs NUMBER(5) NOT NULL REFERENCES produse(cod\_produs),

CONSTRAINT stoc\_pk PRIMARY KEY (cod\_locatie, cod\_produs), nr\_stoc NUMBER(4))

CREATE TABLE pentru

(cod\_servicii NUMBER(5) NOT NULL REFERENCES servicii(cod\_servicii),

cod\_produs NUMBER(5) NOT NULL REFERENCES produse(cod\_produs),

CONSTRAINT pentru\_pk PRIMARY KEY (cod\_servicii, cod\_produs))

CREATE TABLE presteaza

(cod\_servicii NUMBER(5) NOT NULL REFERENCES servicii(cod\_servicii),

cod\_salariat NUMBER(3) NOT NULL REFERENCES salariat(cod\_salariat),

CONSTRAINT presteza\_pk PRIMARY KEY (cod\_servicii, cod\_salariat))

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

INSERT INTO manageri VALUES (100, 'Popa','Andrei', 35000);

INSERT INTO manageri VALUES (110, 'Stan','Mihai', 30000);

INSERT INTO manageri VALUES (120, 'Ion','Ana', 35000);

INSERT INTO manageri VALUES (130, 'Nae','Andrei', 40000);

INSERT INTO manageri VALUES (140, 'Matei','Laura', 40000);

INSERT INTO furnizor VALUES (1843, 'WELLA', '0788655');

INSERT INTO furnizor VALUES (2572, 'NYX','0787655' );

INSERT INTO furnizor VALUES (3678, 'Olaplex','0785322');

INSERT INTO furnizor VALUES (6893, 'Heliocare','07857642' );

INSERT INTO furnizor VALUES (9567, 'Cupio', '07859567');

INSERT INTO CLIENT VALUES(105, 'Stan', 'Maria', 23);

INSERT INTO CLIENT VALUES(115, 'Mincu', 'Ana', 35);

INSERT INTO CLIENT VALUES(125, 'Popescu', 'Dana', 19);

INSERT INTO CLIENT VALUES(135, 'Pop', 'Ioana', 32);

INSERT INTO CLIENT VALUES(145, 'Ion', 'Maria', 25);

INSERT INTO CLIENT VALUES(155, 'Ale', 'Mihai', 23);

INSERT INTO CLIENT VALUES(165, 'Mirea', 'Andreea', 30);

INSERT INTO CLIENT VALUES(175, 'Pop', 'Diana', 34);

INSERT INTO CLIENT VALUES(185, 'Anghel', 'Ioana', 32);

INSERT INTO CLIENT VALUES(195, 'Ion', 'Mara', 27);

INSERT INTO produse VALUES (10785, 'Oja semiper', 9567);

INSERT INTO produse VALUES (10790, 'Oja simpla', 9567);

INSERT INTO produse VALUES (10795, 'Tratament unghii', 9567);

INSERT INTO produse VALUES (15670, 'Vopsea', 1843);

INSERT INTO produse VALUES (15675, 'Fixativ', 1843);

INSERT INTO produse VALUES (19000, 'Tratament keratina', 3678);

INSERT INTO produse VALUES (19050, 'Sampon', 3678);

INSERT INTO produse VALUES (19055, 'Balsam', 3678);

INSERT INTO produse VALUES (30000, 'Paleta ochi', 2572);

INSERT INTO produse VALUES (30100, 'Paleta fata', 2572);

INSERT INTO produse VALUES (30200, 'Fond de ten', 2572);

INSERT INTO produse VALUES (30300, 'Pensule', 2572);

INSERT INTO produse VALUES (67000, 'Ser hidratant', 6893);

INSERT INTO produse VALUES (67050, 'Exfoliant', 6893);

INSERT INTO produse VALUES (67100, 'Masca', 6893);

INSERT INTO firma\_contabilitate VALUES (4070, 'GP ACCOUNTING');

INSERT INTO firma\_contabilitate VALUES (4071, 'TERWHOO');

INSERT INTO firma\_contabilitate VALUES (4072, 'CONTELLO');

INSERT INTO firma\_contabilitate VALUES (4073, 'MIVEXPERT');

INSERT INTO firma\_contabilitate VALUES (4074, 'PREST');

INSERT INTO contabil VALUES (10, 'Popescu', 'Ana',4070);

INSERT INTO contabil VALUES (20, 'Ana', 'Andreea',4072);

INSERT INTO contabil VALUES (30, 'Anghel', 'Ion',4073);

INSERT INTO contabil VALUES (40, 'Matei', 'Elena',4071);

INSERT INTO contabil VALUES (50, 'Stan', 'Ovidiu',4074);

INSERT INTO locatie VALUES (0001, 'Strada Popa Savu, nr.77', 0723657899,100, 50);

INSERT INTO locatie VALUES (0002, 'Strada Fabrica de chibrituri 13-15', 0723657839,110, 20);

INSERT INTO locatie VALUES (0003, 'Strada Petre S. Aurelian nr. 54', 0723657839,110, 30);

INSERT INTO locatie VALUES (0004, 'Soseaua Virtutii Nr. 1 E', 0723657836,120, 40);

INSERT INTO locatie VALUES (0005, 'Str. Heliade intre Vii nr. 29', 0723657879,130, 10);

INSERT INTO locatie VALUES (0006, 'Str. Stelea Spatarul nr 27', 0723657779,140, 50);

INSERT INTO stoc VALUES(0001,10785, 30);

INSERT INTO stoc VALUES(0001,10790, 45);

INSERT INTO stoc VALUES(0001,,10795, 12);

INSERT INTO stoc VALUES(0001,15670, 50);

INSERT INTO stoc VALUES(0001,15675, 30);

INSERT INTO stoc VALUES(0001,19000, 12);

INSERT INTO stoc VALUES(0001,19050, 8);

INSERT INTO stoc VALUES(0001,19055, 13);

INSERT INTO stoc VALUES(0001,30000, 20);

INSERT INTO stoc VALUES(0001,30100, 12);

INSERT INTO stoc VALUES(0001,30200, 20);

INSERT INTO stoc VALUES(0001,30300, 13);

INSERT INTO stoc VALUES(0001,67000, 0);

INSERT INTO stoc VALUES(0001,67050, 12);

INSERT INTO stoc VALUES(0001,67100, 0);

INSERT INTO stoc VALUES(0002,10785, 30);

INSERT INTO stoc VALUES(0002,10790, 25);

INSERT INTO stoc VALUES(0002,10795, 7);

INSERT INTO stoc VALUES(0002,15670, 20);

INSERT INTO stoc VALUES(0002,15675, 4);

INSERT INTO stoc VALUES(0002,19000, 12);

INSERT INTO stoc VALUES(0002,19050, 12);

INSERT INTO stoc VALUES(0002,19055, 10);

INSERT INTO stoc VALUES(0002,30000, 12);

INSERT INTO stoc VALUES(0002,30100, 13);

INSERT INTO stoc VALUES(0002,30200, 20);

INSERT INTO stoc VALUES(0002,30300, 15);

INSERT INTO stoc VALUES(0002,67000, 5);

INSERT INTO stoc VALUES(0002,67050, 7);

INSERT INTO stoc VALUES(0002,67100, 19);

INSERT INTO stoc VALUES(0003,10785, 30);

INSERT INTO stoc VALUES(0003,10790, 25);

INSERT INTO stoc VALUES(0003,10795, 12);

INSERT INTO stoc VALUES(0003,15670, 2);

INSERT INTO stoc VALUES(0003,15675, 4);

INSERT INTO stoc VALUES(0003,19000, 12);

INSERT INTO stoc VALUES(0003,19050, 12);

INSERT INTO stoc VALUES(0003,19055, 0);

INSERT INTO stoc VALUES(0003,30000, 12);

INSERT INTO stoc VALUES(0003,30100, 13);

INSERT INTO stoc VALUES(0003,30200, 14);

INSERT INTO stoc VALUES(0003,30300, 15);

INSERT INTO stoc VALUES(0003,67000, 5);

INSERT INTO stoc VALUES(0003,67050, 17);

INSERT INTO stoc VALUES(0003,67100, 19);

INSERT INTO stoc VALUES(0004,10785, 13);

INSERT INTO stoc VALUES(0004,10790, 15);

INSERT INTO stoc VALUES(0004,10795, 12);

INSERT INTO stoc VALUES(0004,15670, 0);

INSERT INTO stoc VALUES(0004,15675, 14);

INSERT INTO stoc VALUES(0004,19000, 0);

INSERT INTO stoc VALUES(0004,19050, 12);

INSERT INTO stoc VALUES(0004,19055, 20);

INSERT INTO stoc VALUES(0004,30000, 12);

INSERT INTO stoc VALUES(0004,30100, 13);

INSERT INTO stoc VALUES(0004,30200, 14);

INSERT INTO stoc VALUES(0004,30300, 15);

INSERT INTO stoc VALUES(0004,67000, 15);

INSERT INTO stoc VALUES(0004,67050, 17);

INSERT INTO stoc VALUES(0004,67100, 19);

INSERT INTO stoc VALUES(0005,10785, 13);

INSERT INTO stoc VALUES(0005,10790, 15);

INSERT INTO stoc VALUES(0005,10795, 12);

INSERT INTO stoc VALUES(0005,15670, 20);

INSERT INTO stoc VALUES(0005,15675, 14);

INSERT INTO stoc VALUES(0005,19000, 40);

INSERT INTO stoc VALUES(0005,19050, 12);

INSERT INTO stoc VALUES(0005,19055, 20);

INSERT INTO stoc VALUES(0005,30000, 32);

INSERT INTO stoc VALUES(0005,30100, 43);

INSERT INTO stoc VALUES(0005,30200, 14);

INSERT INTO stoc VALUES(0005,30300, 15);

INSERT INTO stoc VALUES(0005,67000, 18);

INSERT INTO stoc VALUES(0005,67050, 17);

INSERT INTO stoc VALUES(0005,67100, 27);

INSERT INTO stoc VALUES(0006,10785, 13);

INSERT INTO stoc VALUES(0006,10790, 15);

INSERT INTO stoc VALUES(0006,10795, 12);

INSERT INTO stoc VALUES(0006,15670, 20);

INSERT INTO stoc VALUES(0006,15675, 14);

INSERT INTO stoc VALUES(0006,19000, 0);

INSERT INTO stoc VALUES(0006,19050, 12);

INSERT INTO stoc VALUES(0006,19055, 20);

INSERT INTO stoc VALUES(0006,30000, 32);

INSERT INTO stoc VALUES(0006,30100, 43);

INSERT INTO stoc VALUES(0006,30200, 14);

INSERT INTO stoc VALUES(0006,30300, 15);

INSERT INTO stoc VALUES(0006,67000, 18);

INSERT INTO stoc VALUES(0006,67050, 17);

INSERT INTO stoc VALUES(0006,67100, 7);

INSERT INTO servicii VALUES (00010, 'Pedichiura',50);

INSERT INTO servicii VALUES (00020, 'Manichiura',70);

INSERT INTO servicii VALUES (00030, 'Vopsit',150);

INSERT INTO servicii VALUES (00040, 'Tuns',50);

INSERT INTO servicii VALUES (00050, 'Tratament par',150);

INSERT INTO servicii VALUES (00060, 'Tratament facial',300);

INSERT INTO servicii VALUES (00070, 'Machiaj',350);

INSERT INTO servicii VALUES (00080, 'Manichiura semiper',140);

INSERT INTO servicii VALUES (00090, 'Machiaj de seara',100);

INSERT INTO servicii VALUES (00100, 'Machiaj de zi',70);

INSERT INTO programare VALUES (5001, to\_timestamp('04/02/2022 12:00:00','MM/DD/YYYY HH24:MI:SS.FF3'), 0005, 00080,105);

INSERT INTO programare VALUES (5002, to\_timestamp('03/16/2022 14:30:00','MM/DD/YYYY HH24:MI:SS.FF3'), 0002, 00060,115);

INSERT INTO programare VALUES (5003, to\_timestamp('01/12/2022 14:30:00','MM/DD/YYYY HH24:MI:SS.FF3'), 0004, 00080, 125);

INSERT INTO programare VALUES (5004, to\_timestamp('02/02/2022 13:00:00','MM/DD/YYYY HH24:MI:SS.FF3'), 0001, 00010,135);

INSERT INTO programare VALUES (5005, to\_timestamp('03/20/2022 14:30:00','MM/DD/YYYY HH24:MI:SS.FF3'), 0002, 00050, 155);

INSERT INTO programare VALUES (5006, to\_timestamp('03/15/2022 18:00:00','MM/DD/YYYY HH24:MI:SS.FF3'), 0003, 00030, 145);

INSERT INTO programare VALUES (5007, to\_timestamp('01/06/2022 09:30:00','MM/DD/YYYY HH24:MI:SS.FF3'), 0001, 00020, 165);

INSERT INTO programare VALUES (5008, to\_timestamp('04/15/2022 09:30:00','MM/DD/YYYY HH24:MI:SS.FF3'), 0002, 00070,175);

INSERT INTO programare VALUES (5009, to\_timestamp('05/22/2022 12:00:00','MM/DD/YYYY HH24:MI:SS.FF3'), 0005, 00080,185);

INSERT INTO programare VALUES (5010, to\_timestamp('03/02/2022 08:00:00','MM/DD/YYYY HH24:MI:SS.FF3'), 0005, 00080,195);

INSERT INTO programare VALUES (5011, to\_timestamp('05/12/2022 17:30:00','MM/DD/YYYY HH24:MI:SS.FF3'), 0006, 00090,115);

INSERT INTO programare VALUES (5012, to\_timestamp('04/17/2022 10:00:00','MM/DD/YYYY HH24:MI:SS.FF3'), 0006, 00070,125);

INSERT INTO programare VALUES (5013, to\_timestamp('05/22/2022 12:00:00','MM/DD/YYYY HH24:MI:SS.FF3'), 0005, 00080,195);

INSERT INTO programare VALUES (5014, to\_timestamp('03/30/2022 08:00:00','MM/DD/YYYY HH24:MI:SS.FF3'), 0003, 00080,115);

INSERT INTO programare VALUES (5015, to\_timestamp('04/02/2022 12:00:00','MM/DD/YYYY HH24:MI:SS.FF3'), 0001, 00060,105);

INSERT INTO programare VALUES (5016, to\_timestamp('01/11/2022 12:00:00','MM/DD/YYYY HH24:MI:SS.FF3'), 0001, 00060,105);

INSERT INTO card\_fidelitate VALUES(4000, to\_date('02/11/2022','MM/DD/YYYY'),to\_date('03/11/2022','MM/DD/YYYY'),115);

INSERT INTO card\_fidelitate VALUES(4001, to\_date('01/23/2022','MM/DD/YYYY'),to\_date('03/23/2022','MM/DD/YYYY'),145);

INSERT INTO card\_fidelitate VALUES(4002, to\_date('02/25/2022','MM/DD/YYYY'),to\_date('05/25/2022','MM/DD/YYYY'),135);

INSERT INTO card\_fidelitate VALUES(4003, to\_date('02/01/2022','MM/DD/YYYY'),to\_date('03/01/2022','MM/DD/YYYY'),195);

INSERT INTO card\_fidelitate VALUES(4004, to\_date('12/22/2021','MM/DD/YYYY'),to\_date('02/22/2022','MM/DD/YYYY'),175);

INSERT INTO card\_fidelitate VALUES(4005, to\_date('02/12/2022','MM/DD/YYYY'),to\_date('03/12/2022','MM/DD/YYYY'),105);

INSERT INTO salariat VALUES(360, 'manichiurist',0002, 'Popescu', 'Elena', 1450);

INSERT INTO salariat VALUES(370, 'makeup\_artist',0002, 'Ana', 'Laura', 1450);

INSERT INTO salariat VALUES(380, 'cosmetolog',0002, 'Stancu', 'Ana', 1450);

INSERT INTO salariat VALUES(390, 'hair\_stylist',0002, 'Stan', 'Dorina', 1450);

INSERT INTO salariat VALUES(400, 'manichiurist',0002, 'Stancu', 'Mihaela', 1450);

INSERT INTO salariat VALUES(410, 'hair\_stylist',0002, 'Ion', 'Maria', 1500);

INSERT INTO salariat VALUES(420, 'manichiurist',0003, 'Stanescu', 'Ana', 1550);

INSERT INTO salariat VALUES(430, 'makeup\_artist',0003, 'Mincu', 'Lana', 1600);

INSERT INTO salariat VALUES(440, 'cosmetolog',0003, 'Marina', 'Anastasia', 1700);

INSERT INTO salariat VALUES(450, 'hair\_stylist',0003, 'Ion', 'Maria', 1300);

INSERT INTO salariat VALUES(460, 'manichiurist',0003, 'Stan', 'Mihnea', 1650);

INSERT INTO salariat VALUES(470, 'makeup\_artist',0003, 'Banea', 'Nicoleta', 1700);

INSERT INTO salariat VALUES(480, 'manichiurist',0004, 'Dan', 'Nicoleta', 1750);

INSERT INTO salariat VALUES(490, 'makeup\_artist',0004, 'Nae', 'Ana', 1650);

INSERT INTO salariat VALUES(500, 'cosmetolog',0004, 'Manea', 'Andra', 1700);

INSERT INTO salariat VALUES(510, 'hair\_stylist',0004, 'Nae', 'Maria', 1500);

INSERT INTO salariat VALUES(520, 'manichiurist',0004, 'Stan', 'Mara', 1650);

INSERT INTO salariat VALUES(530, 'makeup\_artist',0004, 'Mincu', 'Nicoleta', 1700);

INSERT INTO salariat VALUES(540, 'manichiurist',0005, 'Manea', 'Denisa', 1500);

INSERT INTO salariat VALUES(550, 'makeup\_artist',0005, 'Pancu', 'Raluca', 1550);

INSERT INTO salariat VALUES(560, 'cosmetolog',0005, 'Manea', 'Andra', 1700);

INSERT INTO salariat VALUES(570, 'hair\_stylist',0005, 'Pop', 'Nadia', 1550);

INSERT INTO salariat VALUES(580, 'manichiurist',0005, 'Cristescu', 'Diana', 1650);

INSERT INTO salariat VALUES(590, 'cosmetolog',0005, 'Mincu', 'Dana', 1750);

INSERT INTO salariat VALUES(600, 'manichiurist',0006, 'Cristescu', 'Denisa', 1500);

INSERT INTO salariat VALUES(610, 'makeup\_artist',0006, 'Pana', 'Ana', 1600);

INSERT INTO salariat VALUES(620, 'cosmetolog',0006, 'Mina', 'Nadia', 1700);

INSERT INTO salariat VALUES(630, 'hair\_stylist',0006, 'Stan', 'Andra', 1550);

INSERT INTO salariat VALUES(640, 'manichiurist',0006, 'Sincu', 'Alexandra', 1600);

INSERT INTO salariat VALUES(650, 'cosmetolog',0006, 'Paun', 'Raluca', 1800);

INSERT INTO PENTRU VALUES(10, 10790);

INSERT INTO PENTRU VALUES(10, 10795);

INSERT INTO PENTRU VALUES(20, 10790);

INSERT INTO PENTRU VALUES(20, 10795);

INSERT INTO PENTRU VALUES(30, 15670);

INSERT INTO PENTRU VALUES(30, 19050);

INSERT INTO PENTRU VALUES(30, 19055);

INSERT INTO PENTRU VALUES(40, 19050);

INSERT INTO PENTRU VALUES(40, 19055);

INSERT INTO PENTRU VALUES(50, 19050);

INSERT INTO PENTRU VALUES(50, 19055);

INSERT INTO PENTRU VALUES(50, 19000);

INSERT INTO PENTRU VALUES(60, 67000);

INSERT INTO PENTRU VALUES(60, 67050);

INSERT INTO PENTRU VALUES(60, 67100);

INSERT INTO PENTRU VALUES(70, 30100);

INSERT INTO PENTRU VALUES(70, 30200);

INSERT INTO PENTRU VALUES(70, 30300);

INSERT INTO PENTRU VALUES(80, 10790);

INSERT INTO PENTRU VALUES(80, 10785);

INSERT INTO PENTRU VALUES(90, 30000);

INSERT INTO PENTRU VALUES(90, 30100);

INSERT INTO PENTRU VALUES(90, 30200);

INSERT INTO PENTRU VALUES(90, 30300);

INSERT INTO PENTRU VALUES(100, 30000);

INSERT INTO PENTRU VALUES(100, 30100);

INSERT INTO PENTRU VALUES(100, 30200);

INSERT INTO PENTRU VALUES(100, 30300);

INSERT INTO presteaza VALUES(10, 300);

INSERT INTO presteaza VALUES(10, 340);

INSERT INTO presteaza VALUES(10, 360);

INSERT INTO presteaza VALUES(10, 400);

INSERT INTO presteaza VALUES(10, 420);

INSERT INTO presteaza VALUES(10, 460);

INSERT INTO presteaza VALUES(10, 480);

INSERT INTO presteaza VALUES(10, 520);

INSERT INTO presteaza VALUES(10, 540);

INSERT INTO presteaza VALUES(10, 580);

INSERT INTO presteaza VALUES(10, 600);

INSERT INTO presteaza VALUES(10, 640);

INSERT INTO presteaza VALUES(20, 300);

INSERT INTO presteaza VALUES(20, 340);

INSERT INTO presteaza VALUES(20, 360);

INSERT INTO presteaza VALUES(20, 400);

INSERT INTO presteaza VALUES(20, 420);

INSERT INTO presteaza VALUES(20, 460);

INSERT INTO presteaza VALUES(20, 480);

INSERT INTO presteaza VALUES(20, 520);

INSERT INTO presteaza VALUES(20, 540);

INSERT INTO presteaza VALUES(20, 580);

INSERT INTO presteaza VALUES(20, 600);

INSERT INTO presteaza VALUES(20, 640);

INSERT INTO presteaza VALUES(30, 330);

INSERT INTO presteaza VALUES(30, 350);

INSERT INTO presteaza VALUES(30, 390);

INSERT INTO presteaza VALUES(30, 410);

INSERT INTO presteaza VALUES(40, 330);

INSERT INTO presteaza VALUES(40, 350);

INSERT INTO presteaza VALUES(40, 390);

INSERT INTO presteaza VALUES(40, 410);

INSERT INTO presteaza VALUES(40, 450);

INSERT INTO presteaza VALUES(40, 510);

INSERT INTO presteaza VALUES(40, 570);

INSERT INTO presteaza VALUES(40, 630);

INSERT INTO presteaza VALUES(50, 390);

INSERT INTO presteaza VALUES(50, 410);

INSERT INTO presteaza VALUES(50, 450);

INSERT INTO presteaza VALUES(50, 510);

INSERT INTO presteaza VALUES(50, 570);

INSERT INTO presteaza VALUES(50, 630);

INSERT INTO presteaza VALUES(60, 320);

INSERT INTO presteaza VALUES(60, 440);

INSERT INTO presteaza VALUES(60, 380);

INSERT INTO presteaza VALUES(60, 500);

INSERT INTO presteaza VALUES(60, 560);

INSERT INTO presteaza VALUES(60, 590);

INSERT INTO presteaza VALUES(60, 620);

INSERT INTO presteaza VALUES(60, 650);

INSERT INTO presteaza VALUES(70, 310);

INSERT INTO presteaza VALUES(70, 370);

INSERT INTO presteaza VALUES(70, 430);

INSERT INTO presteaza VALUES(70, 470);

INSERT INTO presteaza VALUES(70, 490);

INSERT INTO presteaza VALUES(70, 530);

INSERT INTO presteaza VALUES(70, 550);

INSERT INTO presteaza VALUES(70, 610);

INSERT INTO presteaza VALUES(80, 600);

INSERT INTO presteaza VALUES(80, 640);

INSERT INTO presteaza VALUES(80, 300);

INSERT INTO presteaza VALUES(80, 340);

INSERT INTO presteaza VALUES(80, 360);

INSERT INTO presteaza VALUES(80, 400);

INSERT INTO presteaza VALUES(80, 420);

INSERT INTO presteaza VALUES(90, 310);

INSERT INTO presteaza VALUES(90, 370);

INSERT INTO presteaza VALUES(90, 430);

INSERT INTO presteaza VALUES(90, 470);

INSERT INTO presteaza VALUES(90, 490);

INSERT INTO presteaza VALUES(90, 530);

INSERT INTO presteaza VALUES(90, 550);

INSERT INTO presteaza VALUES(90, 610);

INSERT INTO presteaza VALUES(100, 310);

INSERT INTO presteaza VALUES(100, 370);

INSERT INTO presteaza VALUES(100, 430);

INSERT INTO presteaza VALUES(100, 470);

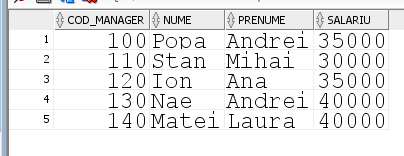
INSERT INTO presteaza VALUES(100, 490);

INSERT INTO presteaza VALUES(100, 530);

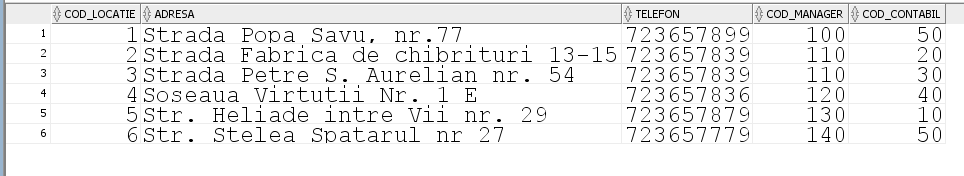
INSERT INTO presteaza VALUES(100, 550);

INSERT INTO presteaza VALUES(100, 610);

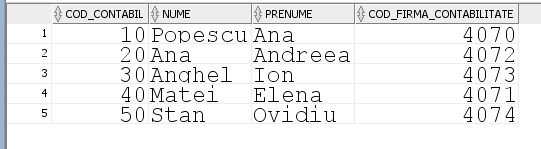
* MANAGER



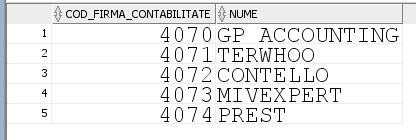
* LOCATIE



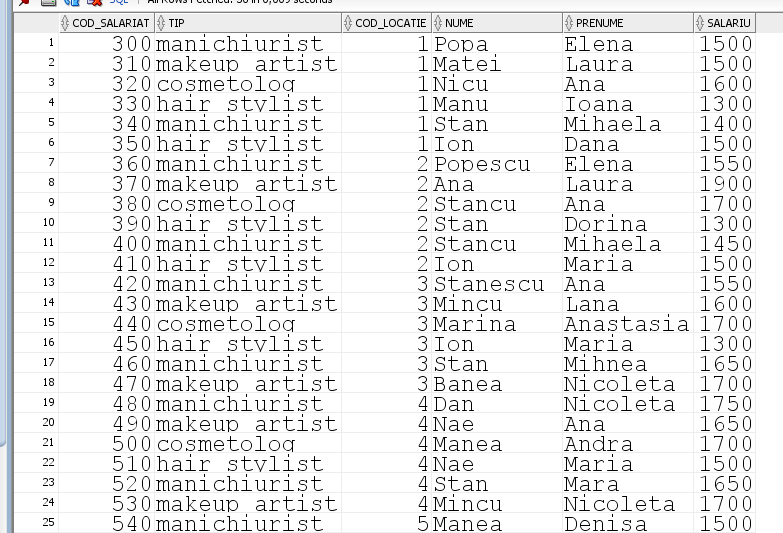
* CONTABIL

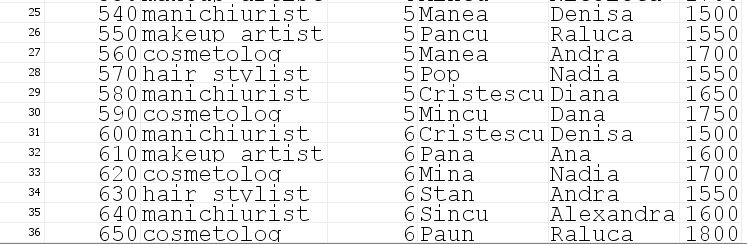


* FIRMA\_CONTABILITATE

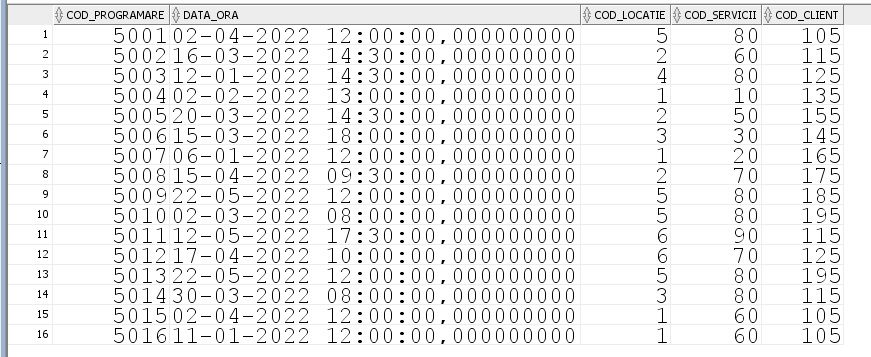


* SALARIAT

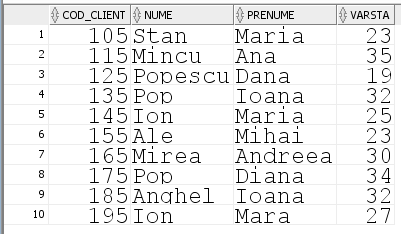




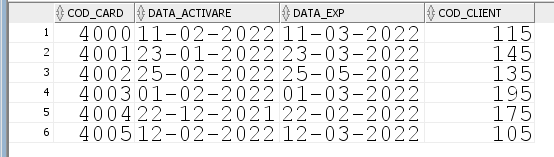
* PROGRAMARE



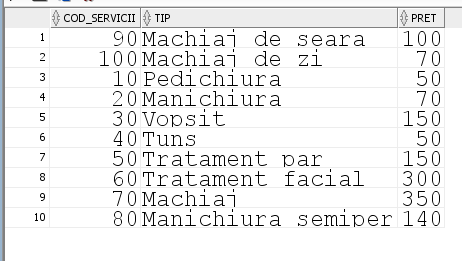
* CLIENT



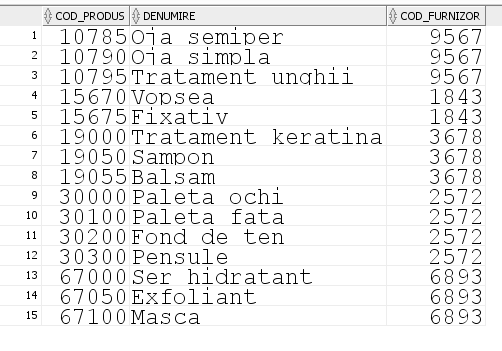
* CARD\_FIDELITATE



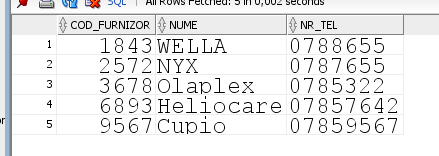
* SERVICII



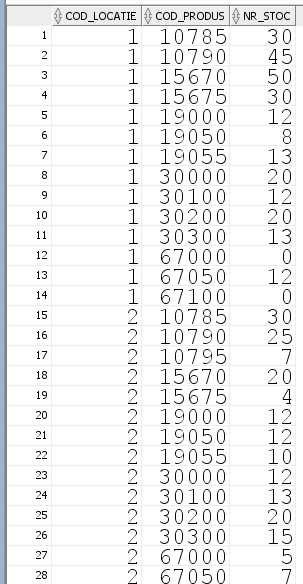
* PRODUSE

**

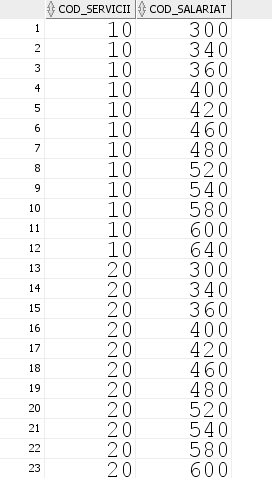
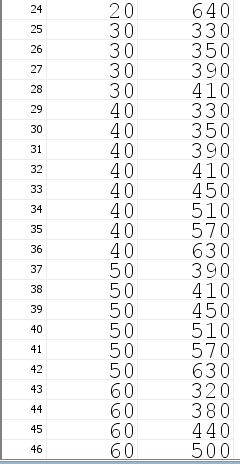
* FURNIZOR



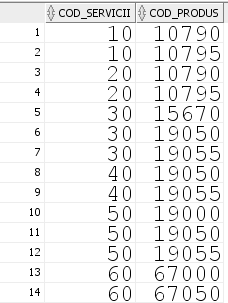
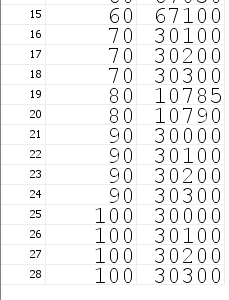
* *STOC*



* PRESTEAZA

* PENTRU

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

Subprogramul primeste de la tastatura un cod al unui client si apoi afiseaza: locul, data, ora, tipul de serviciu, pretul si angajatii disponibili pentru fiecare programare a clientului

--Cod:

set SERVEROUTPUT ON

CREATE OR REPLACE PROCEDURE ex6

(pcod programare.cod\_client%TYPE )

IS

TYPE tabel\_indexat IS TABLE OF VARCHAR(50) INDEX BY PLS\_INTEGER;

t\_prog tabel\_indexat;

TYPE vector IS VARRAY(10) OF programare%ROWTYPE;

v vector:= vector();

adresa VARCHAR(50);

pret NUMBER(3);

card NUMBER(1);

tip\_serv VARCHAR(20);

BEGIN

select count(\*) into card from card\_fidelitate where cod\_client=pcod;

SELECT \* BULK COLLECT INTO v

FROM programare where cod\_client= pcod ;

for i in v.FIRST..v.LAST loop

select locatie.adresa into adresa

from locatie where locatie.cod\_locatie=v(i).cod\_locatie;

select servicii.tip into tip\_serv from

servicii where servicii.cod\_servicii=v(i).cod\_servicii;

select servicii.pret into pret from

servicii where servicii.cod\_servicii=v(i).cod\_servicii;

DBMS\_OUTPUT.PUT(i||'. Programarea cu codul '||v(i).cod\_programare||' are loc la data de '||TO\_CHAR(v(i).data\_ora, 'dd.mm.yyyy')

||' ora '||TO\_CHAR(v(i).data\_ora,'hh:mm')||' la adresa '||adresa);

DBMS\_OUTPUT.NEW\_LINE;

DBMS\_OUTPUT.PUT('---Tip serviciu: '||tip\_serv );

IF (pret>70 AND card<>0) THEN

pret:=pret-15;

DBMS\_OUTPUT.NEW\_LINE;

DBMS\_OUTPUT.PUT('---Pret redus pentru clientii fideli: '||pret );

ELSE

DBMS\_OUTPUT.NEW\_LINE;

DBMS\_OUTPUT.PUT('---Pret: '||pret );

end if;

DBMS\_OUTPUT.NEW\_LINE;

DBMS\_OUTPUT.PUT('---Angajati disponibili: ' );

SELECT nume ||' ' || prenume BULK COLLECT INTO t\_prog

FROM salariat

WHERE cod\_salariat in

(select cod\_salariat from presteaza where cod\_servicii=v(i).cod\_servicii )and cod\_locatie =v(i).cod\_locatie;

FOR i IN t\_prog.FIRST..t\_prog.LAST LOOP

DBMS\_OUTPUT.PUT(t\_prog(i) || ' ');

END LOOP;

DBMS\_OUTPUT.NEW\_LINE;

end loop;

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

RAISE\_APPLICATION\_ERROR(-20001,'Nu exista programari cu acest cod al clientului');

WHEN OTHERS THEN

RAISE\_APPLICATION\_ERROR(-20002,'Alta eroare!');

END ex6 ;

/

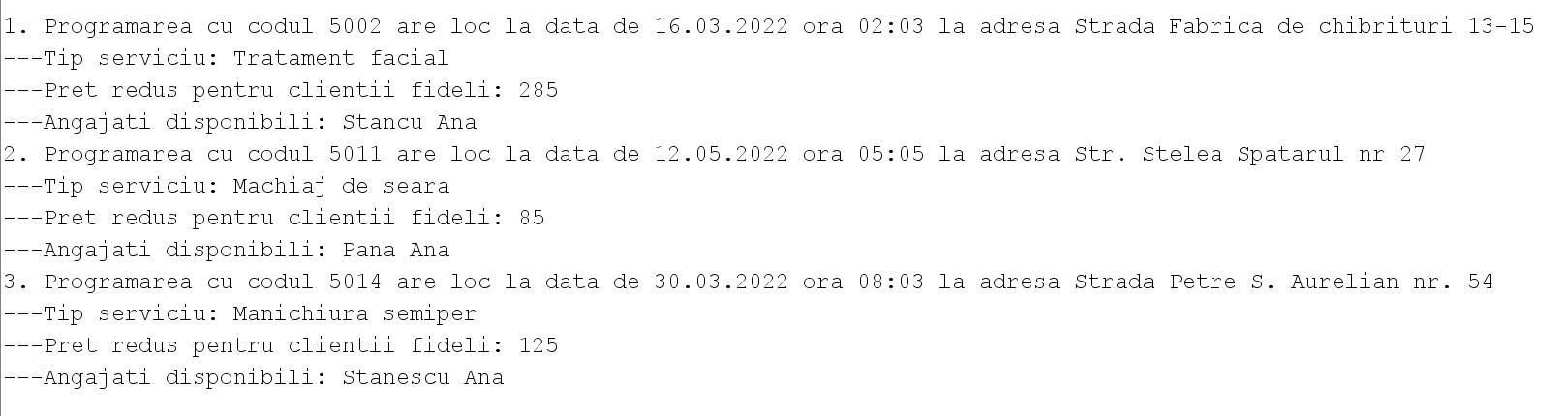
--Testare:

begin

ex6(115);

end;

/

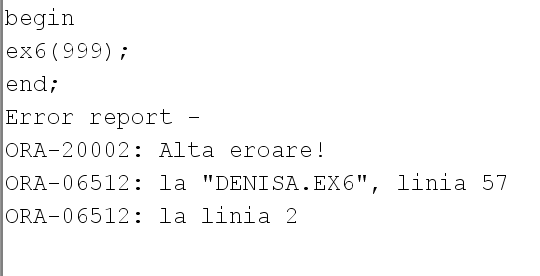


begin

ex6(999);

end;

/



**7. Formulați în limbaj natural o problemă pe care să o rezolvați folosind un subprogram stocat care să utilizeze un tip de cursor studiat. Apelați subprogramul.**

Procedura arata toate informatiile locatiilor(manager, contabil, angajati). Angajatii sunt impartiti pe categorii(cosmetolog, make\_up artist etc) si la final se afiseaza si numarul de programari de la fiecare locatie.

set SERVEROUTPUT ON

CREATE OR REPLACE PROCEDURE ex7

IS

contor NUMBER;

pr NUMBER;

CURSOR cursor\_job IS SELECT cod\_locatie, adresa, cod\_contabil, cod\_manager FROM locatie;

v\_man VARCHAR(50);

v\_con VARCHAR(150);

v\_loc locatie.cod\_locatie%TYPE;

v\_adresa locatie.adresa%TYPE;

v\_contabil locatie.cod\_contabil%TYPE;

v\_manager locatie.cod\_manager%TYPE;

BEGIN

OPEN cursor\_job;

LOOP

contor:=1;

FETCH cursor\_job INTO v\_loc,v\_adresa, v\_contabil, v\_manager;

EXIT WHEN cursor\_job%NOTFOUND;

DBMS\_OUTPUT.PUT\_LINE('-------------------------------------------------------------');

DBMS\_OUTPUT.PUT\_LINE (' locatie : '||v\_loc ||'('||v\_adresa||')');

DBMS\_OUTPUT.PUT\_LINE('-------------------------------------------------------------');

SELECT nume || ' ' || prenume INTO v\_man FROM manager WHERE cod\_manager=v\_manager;

DBMS\_OUTPUT.PUT\_LINE ('MANAGER : '||v\_man );

DBMS\_OUTPUT.PUT\_LINE('-------------------------------------------------------------');

SELECT c.nume || ' ' || c.prenume||'-'||f.nume INTO v\_con FROM contabil c

JOIN firma\_contabilitate f ON c.cod\_firma\_contabilitate=f.cod\_firma\_contabilitate

WHERE cod\_contabil=v\_contabil;

DBMS\_OUTPUT.PUT\_LINE ('CONTABIL : '||v\_con);

DBMS\_OUTPUT.PUT\_LINE('-------------------------------------------------------------');

DBMS\_OUTPUT.PUT\_LINE('ANGAJATI:');

FOR t IN (SELECT DISTINCT tip FROM salariat) LOOP

DBMS\_OUTPUT.NEW\_LINE;

DBMS\_OUTPUT.PUT\_LINE(UPPER(t.tip)|| ' :');

FOR v\_emp IN (SELECT nume, prenume, salariu FROM salariat WHERE cod\_locatie=v\_loc and tip=t.tip) LOOP

DBMS\_OUTPUT.PUT\_LINE(contor||'. '||v\_emp.nume || ' ' || v\_emp.prenume||' are salariul: ' ||v\_emp.salariu||' lei');

contor:=contor+1;

END LOOP;

END LOOP;

select COUNT(\*)into pr from programare where cod\_locatie=v\_loc;

DBMS\_OUTPUT.NEW\_LINE;

DBMS\_OUTPUT.PUT\_LINE( 'NR PROGRAMARI:'|| pr);

END LOOP;

CLOSE cursor\_job;

END ex7;

/

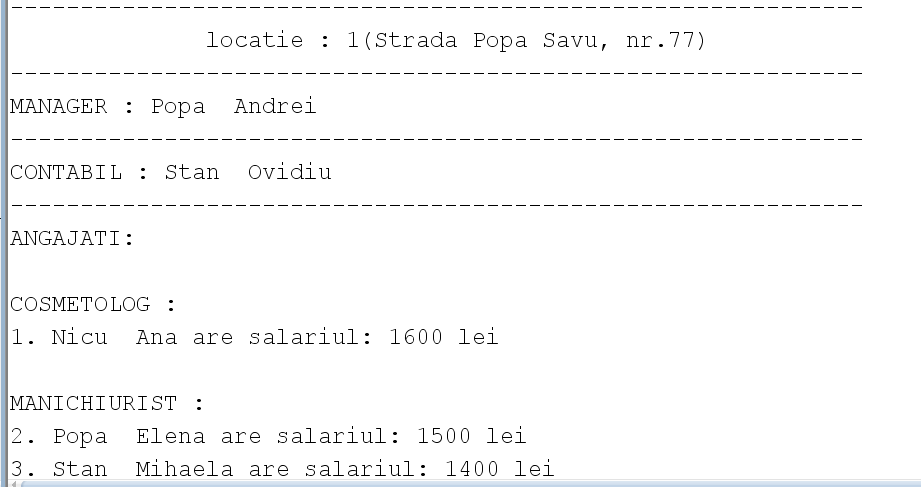
--Testare:

begin

ex7();

end;

/



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

Introducand numele de familie al unui angajat functia gaseste numarul de programari ce pot fi indeplinite de acesta tinand cont de cod\_servicii si cod\_locatie. Functia numara programarile din ziua curenta dupa ora in care este apelata(De exemplu, daca apelam functia la ora 14:00 nu va lua in calcul programarile facute in aceeasi zi mai devreme de ora 14:00)

--cod:

set SERVEROUTPUT ON

CREATE OR REPLACE FUNCTION f\_ex8

(v\_numesal salariat.nume%TYPE DEFAULT 'xyz')

RETURN NUMBER IS

nr\_prog NUMBER(2);

v\_cod salariat.cod\_salariat%TYPE;

BEGIN

SELECT cod\_salariat INTO v\_cod

FROM salariat

WHERE nume=v\_numesal;

select COUNT(\*) into nr\_prog from programare p

join presteaza s on p.cod\_servicii=s.cod\_servicii

join salariat l on s.cod\_salariat = l.cod\_salariat and p.cod\_locatie=l.cod\_locatie

where l.cod\_salariat=v\_cod and (TO\_CHAR(data\_ora,'dd/mm/yyyy')=TO\_CHAR(sysdate, 'dd/mm/yyyy') and

(To\_number(TO\_CHAR(data\_ora, 'HH24MI'))-To\_number(TO\_CHAR(sysdate, 'HH24MI')))>0);

RETURN nr\_prog;

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

RAISE\_APPLICATION\_ERROR(-20000,

'Nu exista angajati cu numele dat');

WHEN TOO\_MANY\_ROWS THEN

RAISE\_APPLICATION\_ERROR(-20001,'Exista mai multi angajati cu acelasi nume');

WHEN OTHERS THEN

RAISE\_APPLICATION\_ERROR(-20002,'Alta eroare!');

END f\_ex8;

/

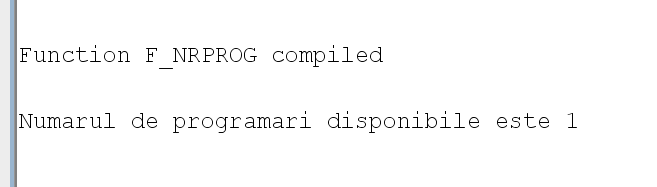
--Testare:

begin

DBMS\_OUTPUT.PUT\_LINE('Numarul de programari disponibile este '|| f\_ex8('Popa'));

end;

/

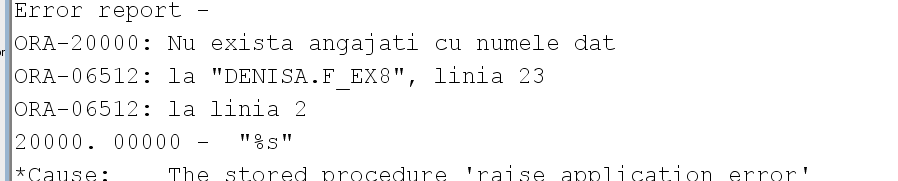


begin

DBMS\_OUTPUT.PUT\_LINE('Numarul de programari disponibile sunt '|| f\_ex8('Joe'));

end;

/

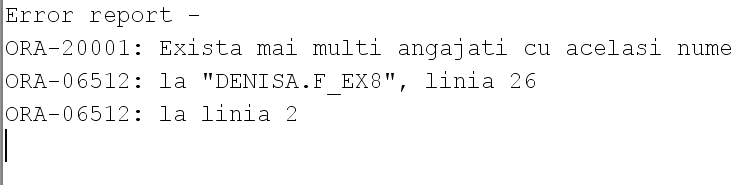


begin

DBMS\_OUTPUT.PUT\_LINE('Numarul de programari disponibile este '|| f\_ex8('Ion'));

end;

/



**9. Formulați în limbaj natural o problemă pe care să o rezolvați folosind un subprogram stocat 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.**

Procedura verifica daca produsele necesare unei programari ale carui cod il introducem sunt in stoc. Daca sunt se afiseaza mesajul „TOATE PRODUSELE SUNT IN STOC”, daca nu, exista doua cazuri:

1.Daca programarea este la minim 7 zile distanta produsele vor fi livrate pana atunci

2. Daca programarea este la o distanta mai mica de 7 zile, atunci se va sterge programarea din tabelul PROGRAMARE si se va afisa mesajul “PROGRAMAREA A FOST ANULATA” insotit de codul programarii si numele clientului. Am tratat si exceptiile no\_data\_found si others

--cod

set SERVEROUTPUT ON

create or replace procedure prog\_stoc (id\_prog programare.cod\_programare%type default -1)

is

TYPE rec IS RECORD (

id\_prod produse.cod\_produs%TYPE,

nume\_produs produse.denumire%TYPE,

nr stoc.nr\_stoc%TYPE,

num\_fur furnizor.nume%TYPE,

nr\_tel furnizor.nr\_tel%TYPE);

TYPE prod\_t IS TABLE OF rec INDEX BY BINARY\_INTEGER;

pr\_table prod\_t;

c NUMBER(2);

nume\_client VARCHAR(50);

data programare.data\_ora%TYPE;

begin

c:=0;

select p.cod\_produs, p.denumire, st.nr\_stoc, f.nume , f.nr\_tel BULK COLLECT INTO pr\_table from produse p

join pentru t on p.cod\_produs=t.cod\_produs

join programare g on t.cod\_servicii=g.cod\_servicii

join stoc st on g.cod\_locatie=st.cod\_locatie and p.cod\_produs=st.cod\_produs

join furnizor f on p.cod\_furnizor=f.cod\_furnizor

where g.cod\_programare=id\_prog;

select nume||' '||prenume into nume\_client from client where cod\_client=(select cod\_client from programare where cod\_programare=id\_prog);

select data\_ora into data from programare where cod\_programare=id\_prog;

FOR i IN pr\_table.FIRST..pr\_table.LAST LOOP

IF pr\_table(i).nr=0 THEN

DBMS\_OUTPUT.NEW\_LINE;

DBMS\_OUTPUT.PUT(pr\_table(i).id\_prod || '. '||pr\_table(i).nume\_produs||' nu este in stoc.');

DBMS\_OUTPUT.NEW\_LINE;

DBMS\_OUTPUT.PUT(' DETALII COMANDA: ');

DBMS\_OUTPUT.NEW\_LINE;

DBMS\_OUTPUT.PUT(' Furnizor: '||pr\_table(i).num\_fur||', nr tel: '||pr\_table(i).nr\_tel||' ');

DBMS\_OUTPUT.NEW\_LINE;

c:=c+1;

END IF;

END LOOP;

IF c>0 and trunc(to\_date(to\_char(data, 'yyyy-mm-dd'),'yyyy-mm-dd')-sysdate )<7 THEN

DBMS\_OUTPUT.PUT('Programarea cu codul: '||id\_prog||' a clientului '||nume\_client|| 'a fost anulata');

DELETE FROM programare WHERE cod\_programare = id\_prog;

ELSIF c>0 and trunc(to\_date(to\_char(data, 'yyyy-mm-dd'),'yyyy-mm-dd')-sysdate )>7 THEN

DBMS\_OUTPUT.PUT('Produsele vor fi livrate pana la programare');

ELSE

DBMS\_OUTPUT.PUT('Toate produsele sunt in stoc ');

END IF;

DBMS\_OUTPUT.NEW\_LINE;

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

RAISE\_APPLICATION\_ERROR(-20000,

'Nu exista programari cu acest cod');

WHEN OTHERS THEN

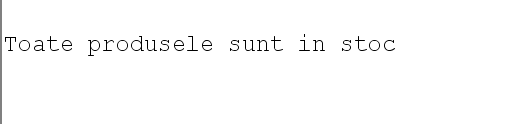
RAISE\_APPLICATION\_ERROR(-20002,'Alta eroare!');

END prog\_stoc;

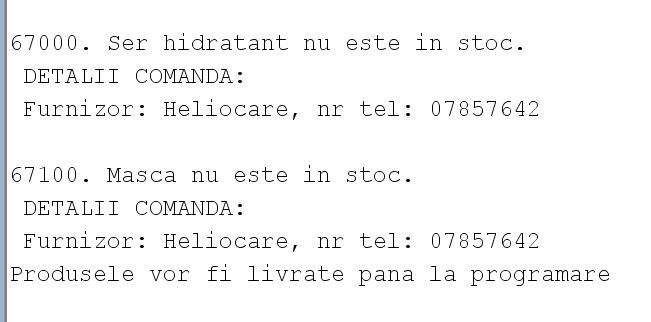
/

--testare:

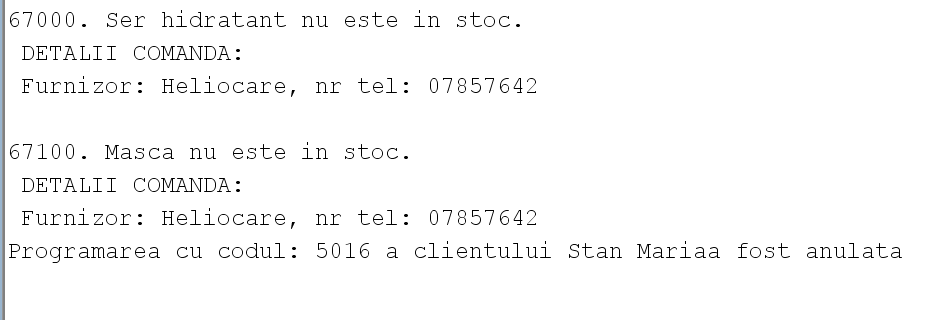
EXECUTE prog\_stoc(5001);



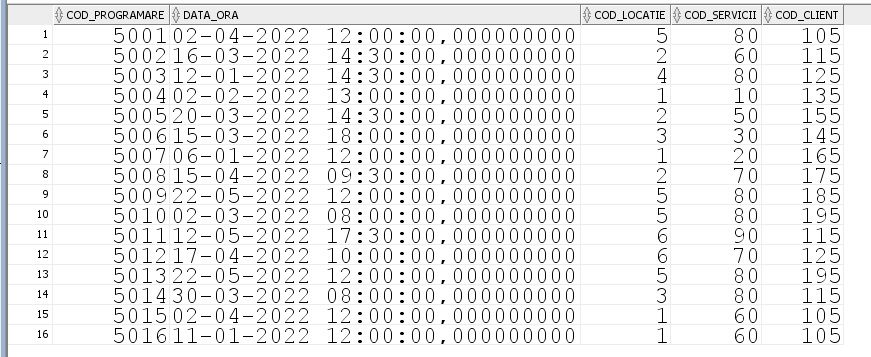
EXECUTE prog\_stoc(5015);

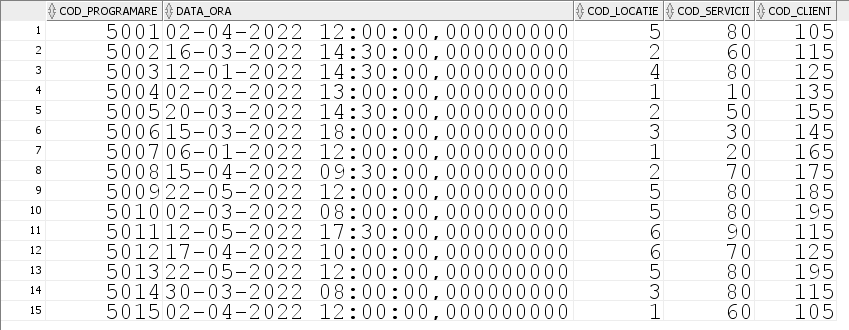


EXECUTE prog\_stoc(5016);

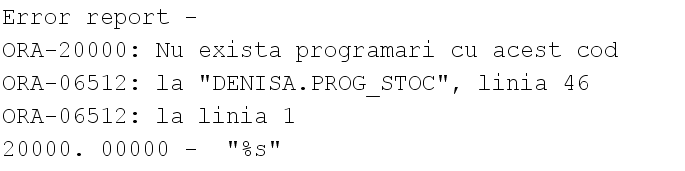


--Programarea 5016 a fost stearsa din table:





EXECUTE prog\_stoc(5999);



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

Actualizarile, stergerile si inserarile in tabelul stoc se pot face doar vinerea de la 8:00 la 14:00.

--cod

CREATE OR REPLACE TRIGGER stoc\_trig

BEFORE INSERT OR DELETE OR UPDATE ON stoc

BEGIN

IF (TO\_CHAR(SYSDATE,'DAY')<> 'VINERI')

AND (TO\_CHAR(SYSDATE,'HH24') NOT BETWEEN 8 AND 14)

THEN

IF INSERTING THEN

RAISE\_APPLICATION\_ERROR(-20001,'Inserarea in tabel doar in ziua inventarului(vineri) intre orele 8:00-14:00');

ELSIF DELETING THEN

RAISE\_APPLICATION\_ERROR(-20002,'Stergerea din tabeleste permisa doar in ziua inventarului(vineri) intre orele 8:00-14:00');

ELSE

RAISE\_APPLICATION\_ERROR(-20003,'Actualizarile in tabel sunt permise doar in ziua inventarului(vineri) intre orele 8:00-14:00');

END IF;

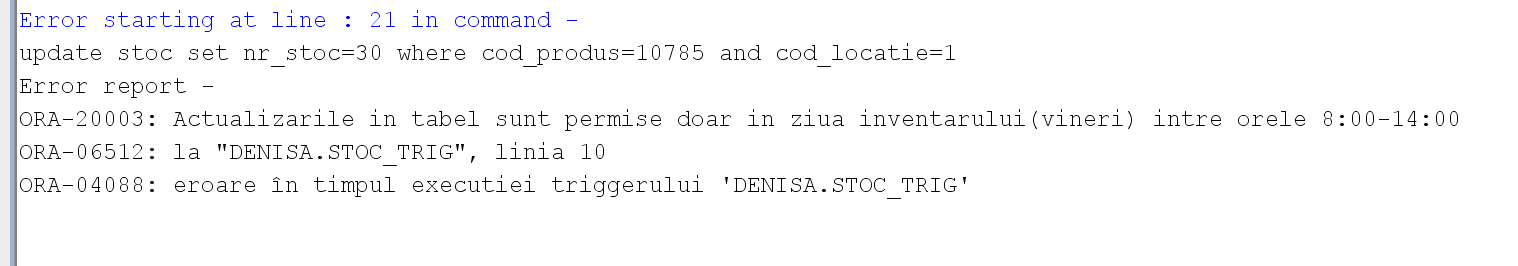
END IF;

END;

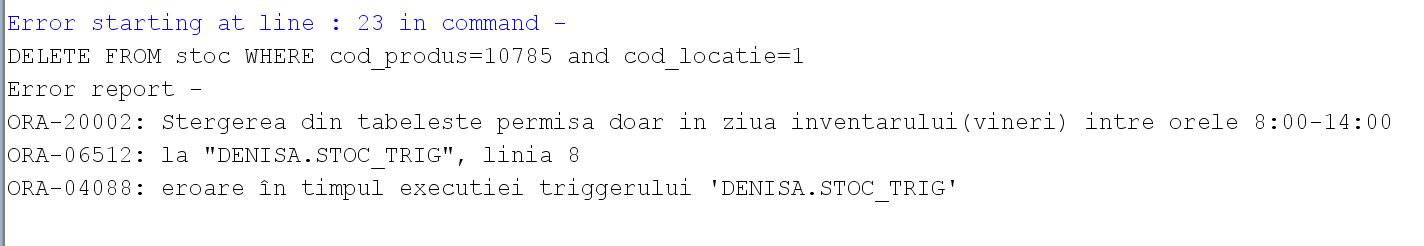
/

--testare:

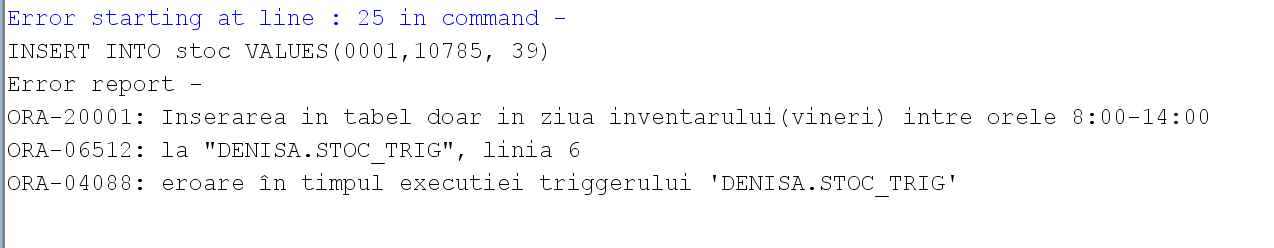
update stoc set nr\_stoc=30 where cod\_produs=10785 and cod\_locatie=1;



DELETE FROM stoc WHERE cod\_produs=10785 and cod\_locatie=1;



INSERT INTO stoc VALUES(0001,10785, 39);



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

--Trigger-ul asigura ca ora inserata sau actualizata este in timpul orelor de munca

--cod:

CREATE OR REPLACE TRIGGER ex11\_trigger

AFTER INSERT OR UPDATE OF DATA\_ORA ON programare

REFERENCING NEW AS New OLD AS Old

FOR EACH ROW

DECLARE

nr\_ang NUMBER(2);

BEGIN

IF INSERTING THEN

IF TO\_CHAR(:NEW.DATA\_ORA,'HH24MI') NOT BETWEEN 800 AND 1830 --am pus 'hh24mi' pt a verifica si minutele

THEN RAISE\_APPLICATION\_ERROR(-20000,'Ora nu se afla in programul de lucru. Program de lucru: 8:00-18:30');

END IF;

ELSIF UPDATING('DATA\_ORA') THEN

IF TO\_CHAR(:NEW.DATA\_ORA,'HH24MI') NOT BETWEEN 800 AND 1830 --am pus 'hh24mi' pt a verifica si minutele

THEN RAISE\_APPLICATION\_ERROR(-20000,'Ora nu se afla in programul de lucru. Program de lucru: 8:00-18:30');

END IF;

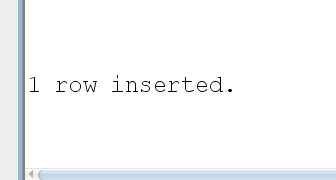
END IF;

END;

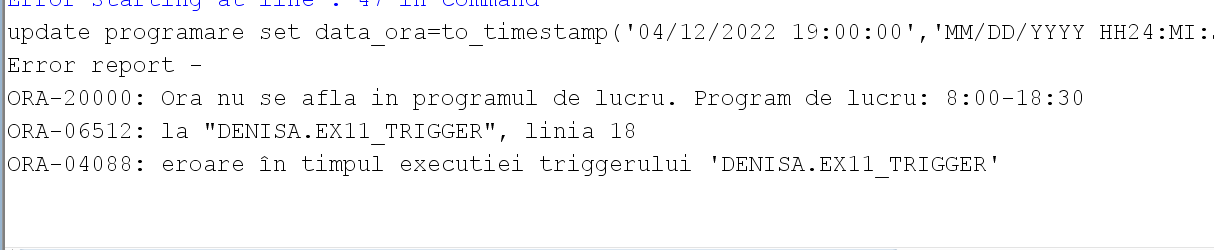
/

--test:

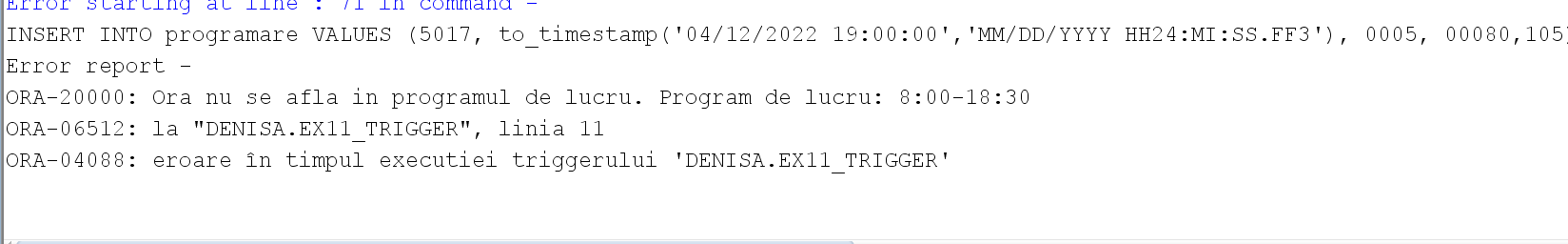
INSERT INTO programare VALUES (5017, to\_timestamp('04/12/2022 12:00:00','MM/DD/YYYY HH24:MI:SS.FF3'), 0005, 00080,125);



update programare set data\_ora=to\_timestamp('04/12/2022 19:00:00','MM/DD/YYYY HH24:MI:SS.FF3')where cod\_programare=5001



INSERT INTO programare VALUES (5017, to\_timestamp('04/12/2022 19:00:00','MM/DD/YYYY HH24:MI:SS.FF3'), 0005, 00080,105)



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

Am creat un tabel in care se va insera fiecare schimbare (create, alter, drop) cu ajutorul trigger-ului LDD

--cod:

CREATE TABLE istoric\_schimbari

(persoana VARCHAR2(30),

eveniment VARCHAR2(20),

tip VARCHAR2(30),

nume VARCHAR2(30),

data TIMESTAMP(3));

CREATE OR REPLACE TRIGGER istoric

AFTER CREATE OR DROP OR ALTER ON SCHEMA

BEGIN

INSERT INTO istoric\_schimbari

VALUES (SYS.LOGIN\_USER,

SYS.SYSEVENT, SYS.DICTIONARY\_OBJ\_TYPE,

SYS.DICTIONARY\_OBJ\_NAME, SYSTIMESTAMP(3));

END;

/

--TESTARE:

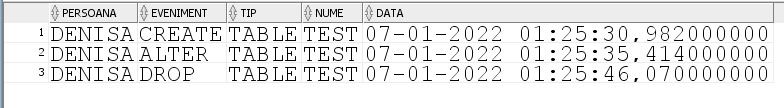
CREATE TABLE test (c1 number(3));

ALTER TABLE test ADD (c2 VARCHAR(2));

INSERT INTO test VALUES (1,'c');

DROP TABLE test;

SELECT \* FROM istoric\_schimbari;



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

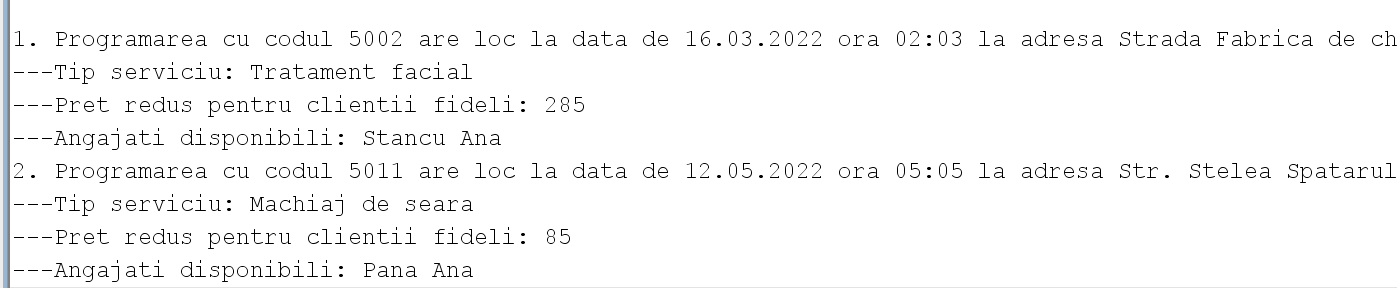
--Testare:

begin

pack\_ex13.ex6(115);

end;

/

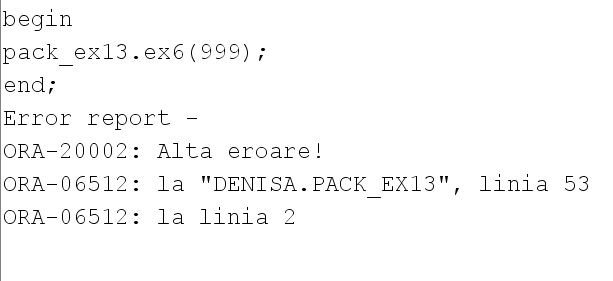


begin

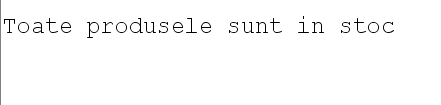
pack\_ex13.ex6(999);

end;

/



EXECUTE pack\_ex13.prog\_stoc(5001);

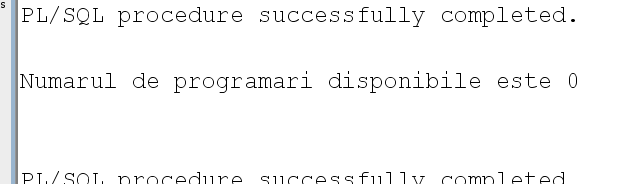


begin

DBMS\_OUTPUT.PUT\_LINE('Numarul de programari disponibile este '|| pack\_ex13.f\_ex8('Popa'));

end;

/



begin

ex7();

end;

/

