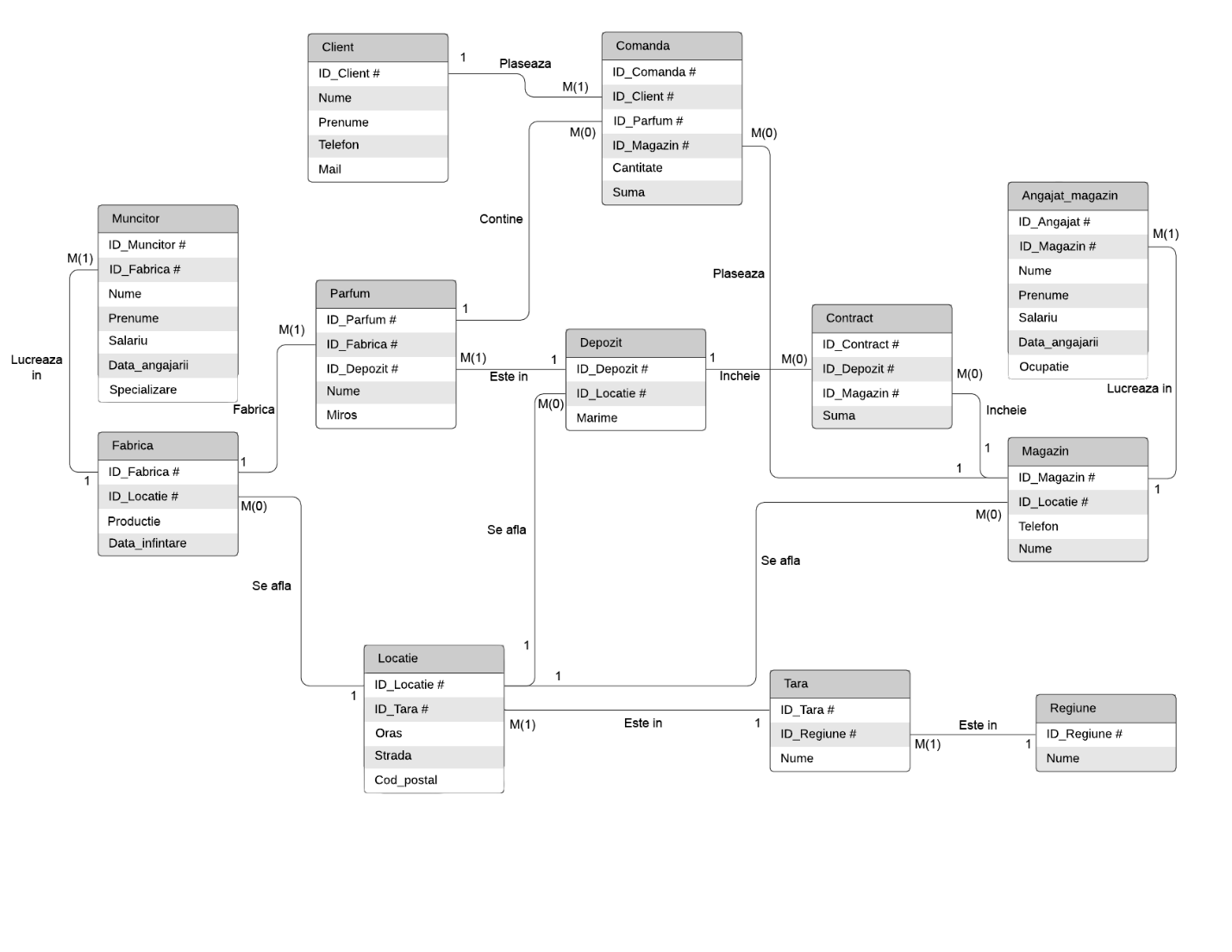
Proiect SGBD

## Alexandru-Daniel Tiganus

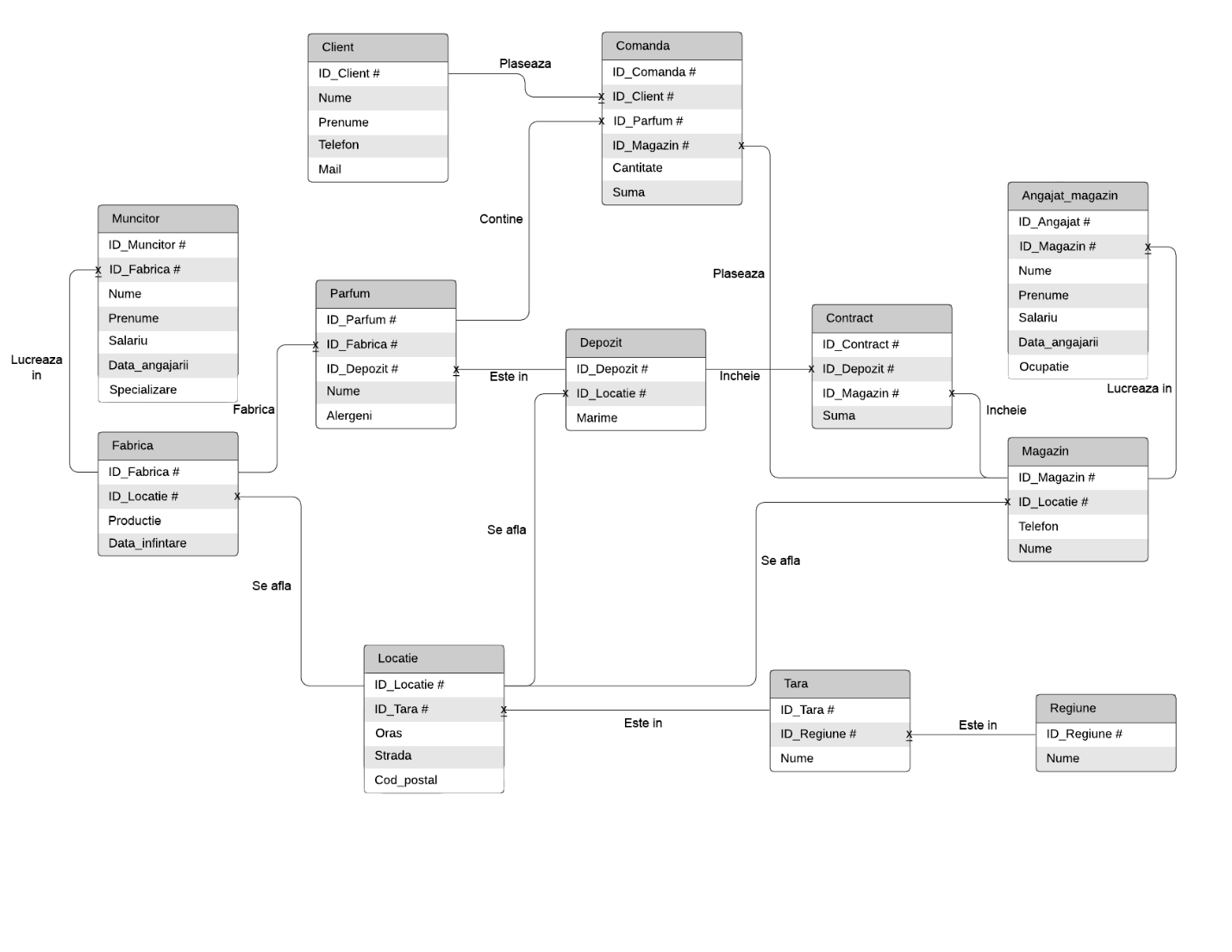
# Utilitatea bazei de date

Modelul descrie o companie care produce si comercializeaza parfumuri. Compania se ocupa atat de fabricarea cat si de depozitarea si de vanzarea acestora. Clientii plaseaza comenzi si pot lua parfumuri facute de companie. Nu exista intermediari, deci costul parfumurilor e redus.

# Diagrama entitate-relatie



# Diagrama conceptuala



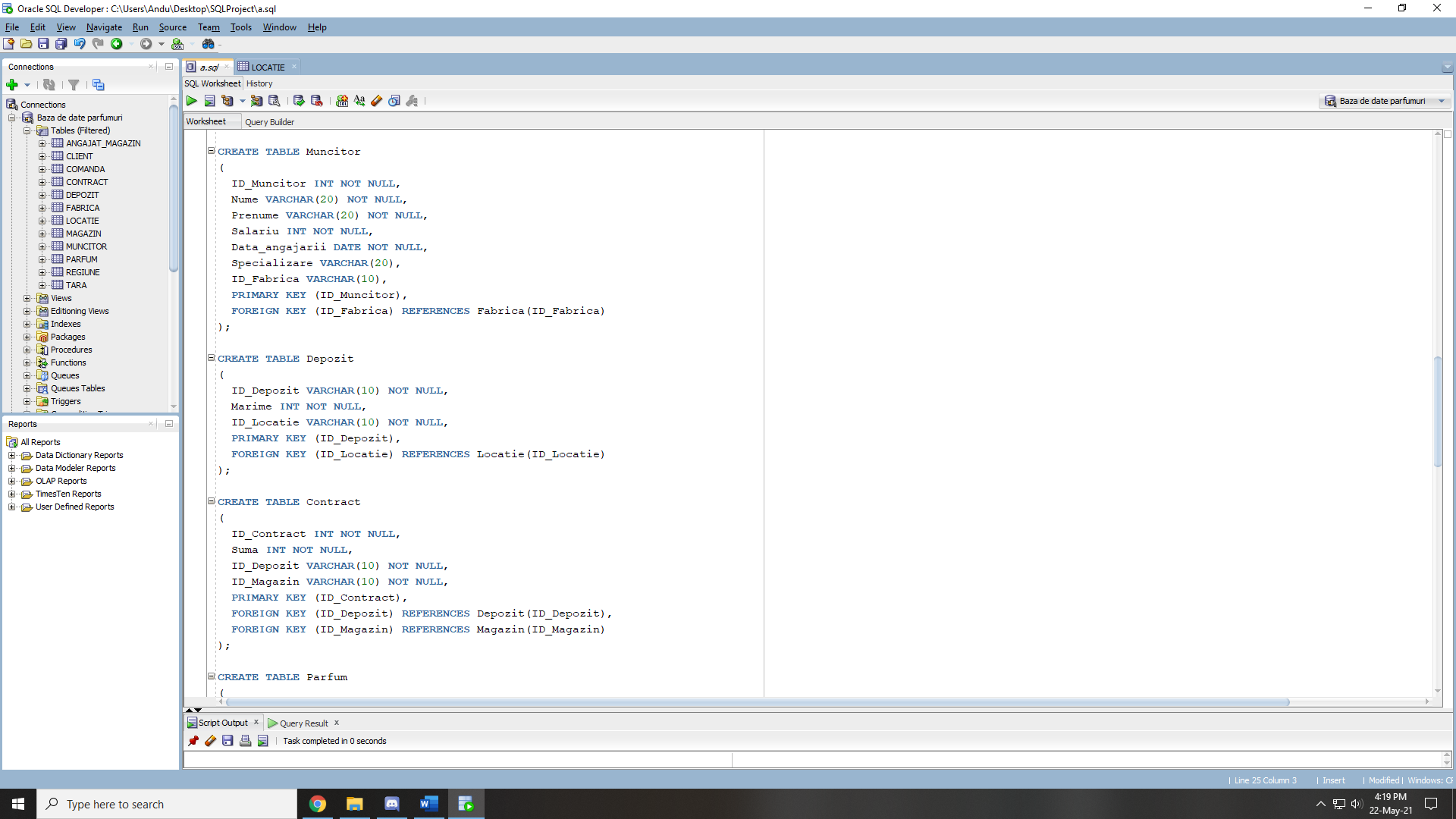
# Implementare model

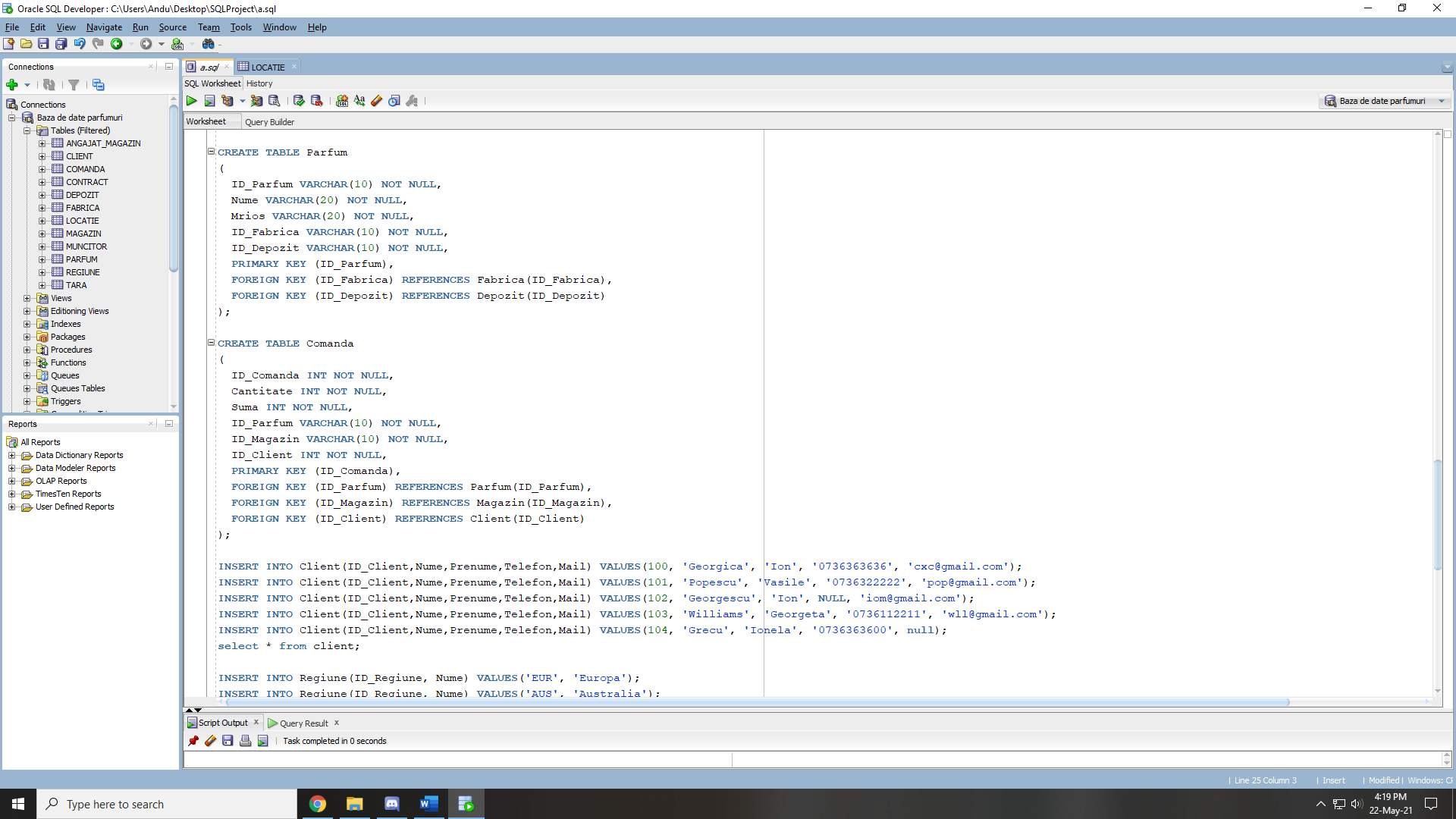
Graphical user interface, text, application

Description automatically generated

Graphical user interface, text, application

Description automatically generated





CREATE TABLE Client

(

ID\_Client INT NOT NULL,

Nume VARCHAR(20) NOT NULL,

Prenume VARCHAR(20),

Telefon VARCHAR(20),

Mail VARCHAR(20),

PRIMARY KEY (ID\_Client)

);

CREATE TABLE Regiune

(

ID\_Regiune VARCHAR(20) NOT NULL,

Nume VARCHAR(20) NOT NULL,

PRIMARY KEY (ID\_Regiune)

);

CREATE TABLE Tara

(

ID\_Tara VARCHAR(10) NOT NULL,

Nume VARCHAR(20) NOT NULL,

ID\_Regiune VARCHAR(20) NOT NULL,

PRIMARY KEY (ID\_Tara),

FOREIGN KEY (ID\_Regiune) REFERENCES Regiune(ID\_Regiune)

);

CREATE TABLE Locatie

(

ID\_Locatie VARCHAR(10) NOT NULL,

Cod\_postal VARCHAR(10) NOT NULL,

Strada VARCHAR(30),

ID\_Tara VARCHAR(10),

Oras VARCHAR(20) NOT NULL,

PRIMARY KEY (ID\_Locatie),

FOREIGN KEY (ID\_Tara) REFERENCES Tara(ID\_Tara)

);

CREATE TABLE Magazin

(

ID\_Magazin VARCHAR(10) NOT NULL,

Telefon VARCHAR(20),

Nume VARCHAR(20) NOT NULL,

ID\_Locatie VARCHAR(10) NOT NULL,

PRIMARY KEY (ID\_Magazin),

FOREIGN KEY (ID\_Locatie) REFERENCES Locatie(ID\_Locatie)

);

CREATE TABLE Angajat\_magazin

(

ID\_Angajat INT NOT NULL,

Nume VARCHAR(20) NOT NULL,

Prenume VARCHAR(20) NOT NULL,

Salariu INT NOT NULL,

Data\_angajarii DATE NOT NULL,

Ocupatie VARCHAR(20),

ID\_Magazin VARCHAR(10) NOT NULL,

PRIMARY KEY (ID\_Angajat),

FOREIGN KEY (ID\_Magazin) REFERENCES Magazin(ID\_Magazin)

);

CREATE TABLE Fabrica

(

ID\_Fabrica VARCHAR(10) NOT NULL,

Productie INT NOT NULL,

Data\_infintare DATE NOT NULL,

ID\_Locatie VARCHAR(10) NOT NULL,

PRIMARY KEY (ID\_Fabrica),

FOREIGN KEY (ID\_Locatie) REFERENCES Locatie(ID\_Locatie)

);

CREATE TABLE Muncitor

(

ID\_Muncitor INT NOT NULL,

Nume VARCHAR(20) NOT NULL,

Prenume VARCHAR(20) NOT NULL,

Salariu INT NOT NULL,

Data\_angajarii DATE NOT NULL,

Specializare VARCHAR(20),

ID\_Fabrica VARCHAR(10),

PRIMARY KEY (ID\_Muncitor),

FOREIGN KEY (ID\_Fabrica) REFERENCES Fabrica(ID\_Fabrica)

);

CREATE TABLE Depozit

(

ID\_Depozit VARCHAR(10) NOT NULL,

Marime INT NOT NULL,

ID\_Locatie VARCHAR(10) NOT NULL,

PRIMARY KEY (ID\_Depozit),

FOREIGN KEY (ID\_Locatie) REFERENCES Locatie(ID\_Locatie)

);

CREATE TABLE Contract

(

ID\_Contract INT NOT NULL,

Suma INT NOT NULL,

ID\_Depozit VARCHAR(10) NOT NULL,

ID\_Magazin VARCHAR(10) NOT NULL,

PRIMARY KEY (ID\_Contract),

FOREIGN KEY (ID\_Depozit) REFERENCES Depozit(ID\_Depozit),

FOREIGN KEY (ID\_Magazin) REFERENCES Magazin(ID\_Magazin)

);

CREATE TABLE Parfum

(

ID\_Parfum VARCHAR(10) NOT NULL,

Nume VARCHAR(20) NOT NULL,

Miros VARCHAR(20) NOT NULL,

ID\_Fabrica VARCHAR(10) NOT NULL,

ID\_Depozit VARCHAR(10) NOT NULL,

PRIMARY KEY (ID\_Parfum),

FOREIGN KEY (ID\_Fabrica) REFERENCES Fabrica(ID\_Fabrica),

FOREIGN KEY (ID\_Depozit) REFERENCES Depozit(ID\_Depozit)

);

CREATE TABLE Comanda

(

ID\_Comanda INT NOT NULL,

Cantitate INT NOT NULL,

Suma INT NOT NULL,

ID\_Parfum VARCHAR(10) NOT NULL,

ID\_Magazin VARCHAR(10) NOT NULL,

ID\_Client INT NOT NULL,

PRIMARY KEY (ID\_Comanda),

FOREIGN KEY (ID\_Parfum) REFERENCES Parfum(ID\_Parfum),

FOREIGN KEY (ID\_Magazin) REFERENCES Magazin(ID\_Magazin),

FOREIGN KEY (ID\_Client) REFERENCES Client(ID\_Client)

);

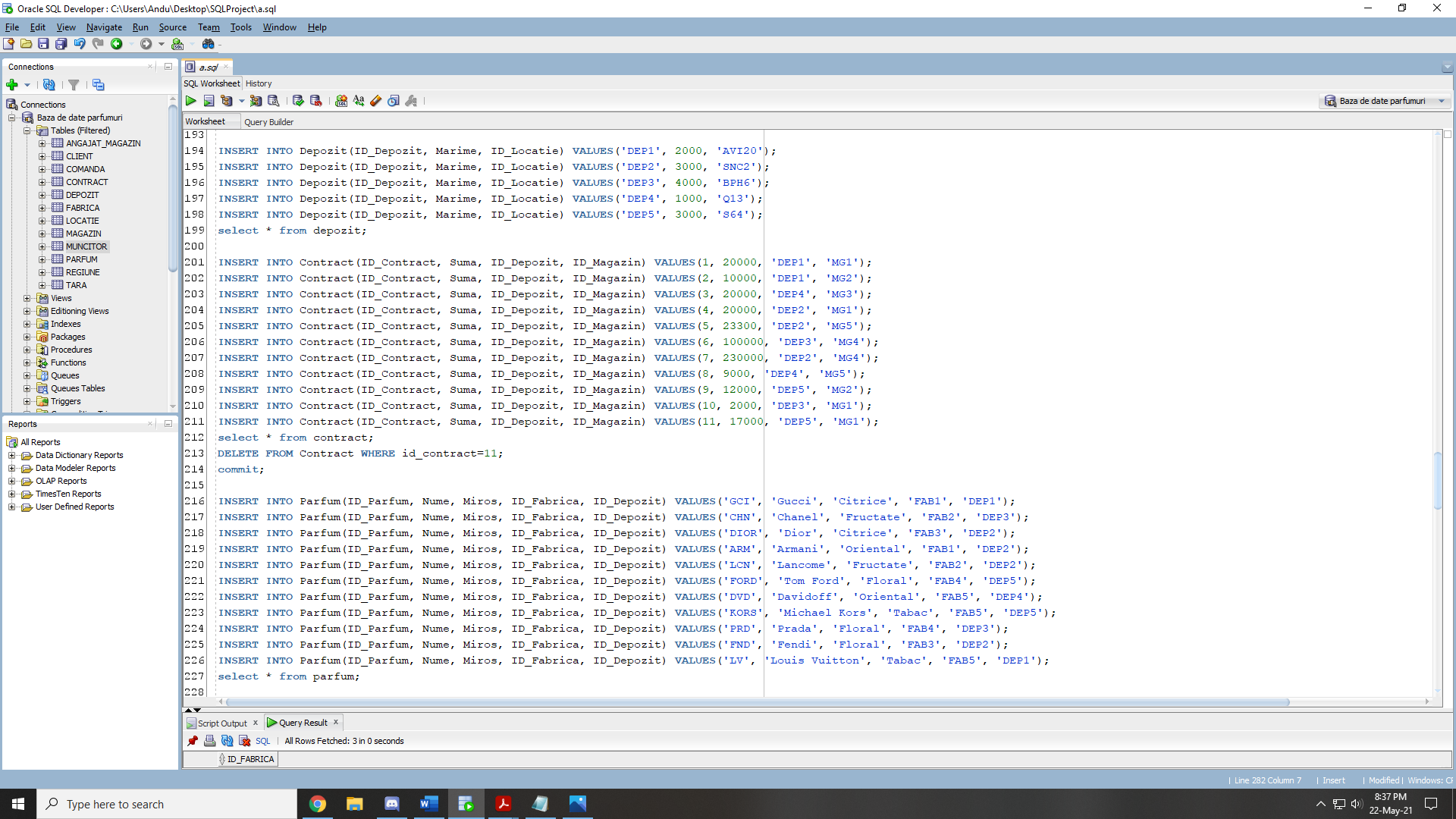
# Adaugare informatii

Graphical user interface, text, application

Description automatically generated

Graphical user interface, text, application

Description automatically generated



Graphical user interface, text, application, email

Description automatically generated

INSERT INTO Client(ID\_Client,Nume,Prenume,Telefon,Mail) VALUES(100, 'Georgica', 'Ion', '0736363636', 'cxc@gmail.com');

INSERT INTO Client(ID\_Client,Nume,Prenume,Telefon,Mail) VALUES(101, 'Popescu', 'Vasile', '0736322222', 'pop@gmail.com');

INSERT INTO Client(ID\_Client,Nume,Prenume,Telefon,Mail) VALUES(102, 'Georgescu', 'Ion', NULL, 'iom@gmail.com');

INSERT INTO Client(ID\_Client,Nume,Prenume,Telefon,Mail) VALUES(103, 'Williams', 'Georgeta', '0736112211', 'wll@gmail.com');

INSERT INTO Client(ID\_Client,Nume,Prenume,Telefon,Mail) VALUES(104, 'Grecu', 'Ionela', '0736363600', null);

select \* from client;

INSERT INTO Regiune(ID\_Regiune, Nume) VALUES('EUR', 'Europa');

INSERT INTO Regiune(ID\_Regiune, Nume) VALUES('AUS', 'Australia');

INSERT INTO Regiune(ID\_Regiune, Nume) VALUES('ASI', 'Asia');

INSERT INTO Regiune(ID\_Regiune, Nume) VALUES('ANO', 'America de Nord');

INSERT INTO Regiune(ID\_Regiune, Nume) VALUES('ASU', 'America de Sud');

select \* from regiune;

INSERT INTO Tara(ID\_Tara, Nume, ID\_Regiune) VALUES('RO', 'Romania', 'EUR');

INSERT INTO Tara(ID\_Tara, Nume, ID\_Regiune) VALUES('DE', 'Germania', 'EUR');

INSERT INTO Tara(ID\_Tara, Nume, ID\_Regiune) VALUES('US', 'Statele Unite', 'ANO');

INSERT INTO Tara(ID\_Tara, Nume, ID\_Regiune) VALUES('CH', 'China', 'ASI');

INSERT INTO Tara(ID\_Tara, Nume, ID\_Regiune) VALUES('BZ', 'Brazilia', 'ASU');

select\* from tara;

INSERT INTO Locatie(ID\_Locatie, Cod\_postal, Strada, ID\_Tara, Oras) VALUES('SHN1', '002233', 'Serban Huinu 1', 'RO', 'Bucuresti');

INSERT INTO Locatie(ID\_Locatie, Cod\_postal, Strada, ID\_Tara, Oras) VALUES('RCC34', '056493', 'Rech Cron 34', 'DE', 'Hamburg');

INSERT INTO Locatie(ID\_Locatie, Cod\_postal, Strada, ID\_Tara, Oras) VALUES('Q13', '53221', 'Qianmen 13', 'CH', 'Beijing');

INSERT INTO Locatie(ID\_Locatie, Cod\_postal, Strada, ID\_Tara, Oras) VALUES('S64', '430222', 'Sloath 41', 'US', 'San Francisco');

INSERT INTO Locatie(ID\_Locatie, Cod\_postal, Strada, ID\_Tara, Oras) VALUES('BPH6', null, 'Bogdan P. Hasdeu 6', 'RO', 'Craiova');

INSERT INTO Locatie(ID\_Locatie, Cod\_postal, Strada, ID\_Tara, Oras) VALUES('BZE3', '212333', 'Bezeor 3', 'BZ', 'Brasilia');

INSERT INTO Locatie(ID\_Locatie, Cod\_postal, Strada, ID\_Tara, Oras) VALUES('SNC2', '346783', 'Sincai 15', 'RO', 'Bucuresti');

INSERT INTO Locatie(ID\_Locatie, Cod\_postal, Strada, ID\_Tara, Oras) VALUES('AVI20', '388543', 'Aviatorilor 20', 'RO', 'Iasi');

select \* from locatie;

INSERT INTO Magazin(ID\_Magazin, Telefon, Nume, ID\_Locatie ) VALUES('MG1', '+8616742264298', 'Beijing parfumes', 'Q13');

INSERT INTO Magazin(ID\_Magazin, Telefon, Nume, ID\_Locatie ) VALUES('MG2', '+40736073135', 'Luxury shop', 'SHN1');

INSERT INTO Magazin(ID\_Magazin, Telefon, Nume, ID\_Locatie ) VALUES('MG3', '+491515553057', 'German quality', 'RCC34');

INSERT INTO Magazin(ID\_Magazin, Telefon, Nume, ID\_Locatie ) VALUES('MG4', '+14842634732', 'San parfume', 'S64');

INSERT INTO Magazin(ID\_Magazin, Telefon, Nume, ID\_Locatie ) VALUES('MG5', '+40722222222', 'Parfumuri de top', 'BPH6');

select \* from magazin;

INSERT INTO Angajat\_magazin(ID\_Angajat, Nume, Prenume, Salariu, Data\_angajarii, Ocupatie, ID\_Magazin) VALUES(1000, 'Ion', 'Croco', 2000, TO\_DATE('17/12/2015', 'DD/MM/YYYY'), 'Vanzator', 'MG1');

INSERT INTO Angajat\_magazin(ID\_Angajat, Nume, Prenume, Salariu, Data\_angajarii, Ocupatie, ID\_Magazin) VALUES(1001, 'Vasile', 'Popescu', 2000, TO\_DATE('12/10/2012', 'DD/MM/YYYY'), 'Vanzator', 'MG2');

INSERT INTO Angajat\_magazin(ID\_Angajat, Nume, Prenume, Salariu, Data\_angajarii, Ocupatie, ID\_Magazin) VALUES(1002, 'Vasile', 'Campineanu', 4000, TO\_DATE('24/03/2001', 'DD/MM/YYYY'), 'Manager', 'MG3');

INSERT INTO Angajat\_magazin(ID\_Angajat, Nume, Prenume, Salariu, Data\_angajarii, Ocupatie, ID\_Magazin) VALUES(1003, 'Herbert', 'Bobbins', 2500, TO\_DATE('11/11/2011', 'DD/MM/YYYY'), 'Vanzator', 'MG4');

INSERT INTO Angajat\_magazin(ID\_Angajat, Nume, Prenume, Salariu, Data\_angajarii, Ocupatie, ID\_Magazin) VALUES(1004, 'George', 'Ionel', 1000, TO\_DATE('10/12/2012', 'DD/MM/YYYY'), 'Ingrijitor', 'MG5');

select \* from angajat\_magazin;

INSERT INTO Fabrica(ID\_Fabrica, Data\_infintare, Productie, ID\_Locatie) VALUES('FAB1', TO\_DATE('10/12/2001', 'DD/MM/YYYY'), 2000, 'AVI20');

INSERT INTO Fabrica(ID\_Fabrica, Data\_infintare, Productie, ID\_Locatie) VALUES('FAB2', TO\_DATE('01/02/1990', 'DD/MM/YYYY'), 1200, 'BZE3');

INSERT INTO Fabrica(ID\_Fabrica, Data\_infintare, Productie, ID\_Locatie) VALUES('FAB3', TO\_DATE('05/06/1984', 'DD/MM/YYYY'), 1000, 'SNC2');

INSERT INTO Fabrica(ID\_Fabrica, Data\_infintare, Productie, ID\_Locatie) VALUES('FAB4', TO\_DATE('12/12/2016', 'DD/MM/YYYY'), 4000, 'Q13');

INSERT INTO Fabrica(ID\_Fabrica, Data\_infintare, Productie, ID\_Locatie) VALUES('FAB5', TO\_DATE('07/09/2012', 'DD/MM/YYYY'), 3200, 'S64');

select \* from fabrica;

INSERT INTO Muncitor(ID\_Muncitor, Nume, Prenume, Salariu, Data\_angajarii, Specializare, ID\_Fabrica) VALUES(2000, 'Alexandru', 'Vasilaru', 800, TO\_DATE('17/12/2015', 'DD/MM/YYYY'), 'Sticlar', 'FAB1');

INSERT INTO Muncitor(ID\_Muncitor, Nume, Prenume, Salariu, Data\_angajarii, Specializare, ID\_Fabrica) VALUES(2001, 'Bob', 'Stocks', 900, TO\_DATE('10/06/2012', 'DD/MM/YYYY'), 'Sticlar', 'FAB2');

INSERT INTO Muncitor(ID\_Muncitor, Nume, Prenume, Salariu, Data\_angajarii, Specializare, ID\_Fabrica) VALUES(2002, 'Shi', 'San Ping', 1700, TO\_DATE('17/12/2018', 'DD/MM/YYYY'), 'Parfumier', 'FAB3');

INSERT INTO Muncitor(ID\_Muncitor, Nume, Prenume, Salariu, Data\_angajarii, Specializare, ID\_Fabrica) VALUES(2003, 'Matei', 'Olteanu', 1700, TO\_DATE('24/03/2014', 'DD/MM/YYYY'), 'Parfumier', 'FAB4');

INSERT INTO Muncitor(ID\_Muncitor, Nume, Prenume, Salariu, Data\_angajarii, Specializare, ID\_Fabrica) VALUES(2004, 'Florin', 'Filimon', 1500, TO\_DATE('30/10/2019', 'DD/MM/YYYY'), 'Parfumier', 'FAB5');

INSERT INTO Muncitor(ID\_Muncitor, Nume, Prenume, Salariu, Data\_angajarii, Specializare, ID\_Fabrica) VALUES(2005, 'Adrian', 'Florinache', 3300, TO\_DATE('05/12/1999', 'DD/MM/YYYY'), 'Maistru', 'FAB1');

INSERT INTO Muncitor(ID\_Muncitor, Nume, Prenume, Salariu, Data\_angajarii, Specializare, ID\_Fabrica) VALUES(2006, 'George', 'Cal', 3300, TO\_DATE('05/10/2006', 'DD/MM/YYYY'), 'Maistru', 'FAB1');

INSERT INTO Muncitor(ID\_Muncitor, Nume, Prenume, Salariu, Data\_angajarii, Specializare, ID\_Fabrica) VALUES(2007, 'Alin', 'Zlotea', 1000, TO\_DATE('28/12/2009', 'DD/MM/YYYY'), 'Sticlar', 'FAB2');

INSERT INTO Muncitor(ID\_Muncitor, Nume, Prenume, Salariu, Data\_angajarii, Specializare, ID\_Fabrica) VALUES(2008, 'Alexandru', 'Marcu', 1600, TO\_DATE('25/04/2009', 'DD/MM/YYYY'), 'Parfumier', 'FAB3');

INSERT INTO Muncitor(ID\_Muncitor, Nume, Prenume, Salariu, Data\_angajarii, Specializare, ID\_Fabrica) VALUES(2009, 'Alexandru', 'Ionel', 1000, TO\_DATE('23/05/2020', 'DD/MM/YYYY'), 'Parfumier', null);

INSERT INTO Muncitor(ID\_Muncitor, Nume, Prenume, Salariu, Data\_angajarii, Specializare, ID\_Fabrica) VALUES(2010, 'Grigore', 'Alexandrescu', 1000, TO\_DATE('03/02/2000', 'DD/MM/YYYY'), 'Parfumier', 'FAB4');

select \* from muncitor;

INSERT INTO Depozit(ID\_Depozit, Marime, ID\_Locatie) VALUES('DEP1', 2000, 'AVI20');

INSERT INTO Depozit(ID\_Depozit, Marime, ID\_Locatie) VALUES('DEP2', 3000, 'SNC2');

INSERT INTO Depozit(ID\_Depozit, Marime, ID\_Locatie) VALUES('DEP3', 4000, 'BPH6');

INSERT INTO Depozit(ID\_Depozit, Marime, ID\_Locatie) VALUES('DEP4', 1000, 'Q13');

INSERT INTO Depozit(ID\_Depozit, Marime, ID\_Locatie) VALUES('DEP5', 3000, 'S64');

select \* from depozit;

INSERT INTO Contract(ID\_Contract, Suma, ID\_Depozit, ID\_Magazin) VALUES(1, 20000, 'DEP1', 'MG1');

INSERT INTO Contract(ID\_Contract, Suma, ID\_Depozit, ID\_Magazin) VALUES(2, 10000, 'DEP1', 'MG2');

INSERT INTO Contract(ID\_Contract, Suma, ID\_Depozit, ID\_Magazin) VALUES(3, 20000, 'DEP4', 'MG3');

INSERT INTO Contract(ID\_Contract, Suma, ID\_Depozit, ID\_Magazin) VALUES(4, 20000, 'DEP2', 'MG1');

INSERT INTO Contract(ID\_Contract, Suma, ID\_Depozit, ID\_Magazin) VALUES(5, 23300, 'DEP2', 'MG5');

INSERT INTO Contract(ID\_Contract, Suma, ID\_Depozit, ID\_Magazin) VALUES(6, 100000, 'DEP3', 'MG4');

INSERT INTO Contract(ID\_Contract, Suma, ID\_Depozit, ID\_Magazin) VALUES(7, 230000, 'DEP2', 'MG4');

INSERT INTO Contract(ID\_Contract, Suma, ID\_Depozit, ID\_Magazin) VALUES(8, 9000, 'DEP4', 'MG5');

INSERT INTO Contract(ID\_Contract, Suma, ID\_Depozit, ID\_Magazin) VALUES(9, 12000, 'DEP5', 'MG2');

INSERT INTO Contract(ID\_Contract, Suma, ID\_Depozit, ID\_Magazin) VALUES(10, 2000, 'DEP3', 'MG1');

INSERT INTO Contract(ID\_Contract, Suma, ID\_Depozit, ID\_Magazin) VALUES(11, 17000, 'DEP5', 'MG1');

select \* from contract;

DELETE FROM Contract WHERE id\_contract=11;

commit;

INSERT INTO Parfum(ID\_Parfum, Nume, Miros, ID\_Fabrica, ID\_Depozit) VALUES('GCI', 'Gucci', 'Citrice', 'FAB1', 'DEP1');

INSERT INTO Parfum(ID\_Parfum, Nume, Miros, ID\_Fabrica, ID\_Depozit) VALUES('CHN', 'Chanel', 'Fructate', 'FAB2', 'DEP3');

INSERT INTO Parfum(ID\_Parfum, Nume, Miros, ID\_Fabrica, ID\_Depozit) VALUES('DIOR', 'Dior', 'Citrice', 'FAB3', 'DEP2');

INSERT INTO Parfum(ID\_Parfum, Nume, Miros, ID\_Fabrica, ID\_Depozit) VALUES('ARM', 'Armani', 'Oriental', 'FAB1', 'DEP2');

INSERT INTO Parfum(ID\_Parfum, Nume, Miros, ID\_Fabrica, ID\_Depozit) VALUES('LCN', 'Lancome', 'Fructate', 'FAB2', 'DEP2');

INSERT INTO Parfum(ID\_Parfum, Nume, Miros, ID\_Fabrica, ID\_Depozit) VALUES('FORD', 'Tom Ford', 'Floral', 'FAB4', 'DEP5');

INSERT INTO Parfum(ID\_Parfum, Nume, Miros, ID\_Fabrica, ID\_Depozit) VALUES('DVD', 'Davidoff', 'Oriental', 'FAB5', 'DEP4');

INSERT INTO Parfum(ID\_Parfum, Nume, Miros, ID\_Fabrica, ID\_Depozit) VALUES('KORS', 'Michael Kors', 'Tabac', 'FAB5', 'DEP5');

INSERT INTO Parfum(ID\_Parfum, Nume, Miros, ID\_Fabrica, ID\_Depozit) VALUES('PRD', 'Prada', 'Floral', 'FAB4', 'DEP3');

INSERT INTO Parfum(ID\_Parfum, Nume, Miros, ID\_Fabrica, ID\_Depozit) VALUES('FND', 'Fendi', 'Floral', 'FAB3', 'DEP2');

INSERT INTO Parfum(ID\_Parfum, Nume, Miros, ID\_Fabrica, ID\_Depozit) VALUES('LV', 'Louis Vuitton', 'Tabac', 'FAB5', 'DEP1');

select \* from parfum;

INSERT INTO Comanda(ID\_Comanda, Cantitate, Suma, ID\_Parfum, ID\_Magazin, ID\_Client) VALUES(1, 2, 200, 'FORD', 'MG4', 100);

INSERT INTO Comanda(ID\_Comanda, Cantitate, Suma, ID\_Parfum, ID\_Magazin, ID\_Client) VALUES(2, 4, 440, 'FND', 'MG5', 101);

INSERT INTO Comanda(ID\_Comanda, Cantitate, Suma, ID\_Parfum, ID\_Magazin, ID\_Client) VALUES(3, 5, 500, 'FORD', 'MG4', 102);

INSERT INTO Comanda(ID\_Comanda, Cantitate, Suma, ID\_Parfum, ID\_Magazin, ID\_Client) VALUES(4, 1, 200, 'GCI', 'MG1', 102);

INSERT INTO Comanda(ID\_Comanda, Cantitate, Suma, ID\_Parfum, ID\_Magazin, ID\_Client) VALUES(5, 2, 240, 'KORS', 'MG2', 100);

INSERT INTO Comanda(ID\_Comanda, Cantitate, Suma, ID\_Parfum, ID\_Magazin, ID\_Client) VALUES(6, 3, 800, 'DIOR', 'MG5', 101);

INSERT INTO Comanda(ID\_Comanda, Cantitate, Suma, ID\_Parfum, ID\_Magazin, ID\_Client) VALUES(7, 1, 140, 'LCN', 'MG5', 100);

INSERT INTO Comanda(ID\_Comanda, Cantitate, Suma, ID\_Parfum, ID\_Magazin, ID\_Client) VALUES(8, 6, 1400, 'ARM', 'MG5', 104);

INSERT INTO Comanda(ID\_Comanda, Cantitate, Suma, ID\_Parfum, ID\_Magazin, ID\_Client) VALUES(9, 24, 2000, 'DVD', 'MG3', 104);

INSERT INTO Comanda(ID\_Comanda, Cantitate, Suma, ID\_Parfum, ID\_Magazin, ID\_Client) VALUES(10, 5, 500, 'FND', 'MG4', 102);

INSERT INTO Comanda(ID\_Comanda, Cantitate, Suma, ID\_Parfum, ID\_Magazin, ID\_Client) VALUES(11, 2, 200, 'FND', 'MG4', 103);

select \* from comanda;

# Problema

Graphical user interface, text, application, email

Description automatically generated

Graphical user interface, text, application, email

Description automatically generated

-- Creati o functie care gaseste numarul angajatilor care au fost angajati printre primii n oameni la firma, si lucreaza intr-una din cele mai noi deschise m fabrici.

CREATE OR REPLACE FUNCTION Numar\_muncitori\_adt(n in number, m in number) return number is rezultat number(10);

TYPE tablou\_muncitori IS TABLE OF muncitor%ROWTYPE INDEX BY BINARY\_INTEGER;

TYPE vector\_fabrici IS VARRAY(100) OF fabrica%ROWTYPE;

muncitori tablou\_muncitori;

fabrici vector\_fabrici;

i binary\_integer;

j binary\_integer;

BEGIN

select \*

bulk collect into muncitori

from( select \* from muncitor order by data\_angajarii )

where rownum <= n;

select \*

bulk collect into fabrici

from( select \* from fabrica order by data\_infintare desc )

where rownum <= m;

i := muncitori.first;

rezultat := 0;

while (i is not null) loop

j := fabrici.first;

while (j is not null) loop

if muncitori(i).id\_fabrica = fabrici(j).id\_fabrica then

rezultat := rezultat + 1;

end if;

j := fabrici.next(j);

end loop;

i := muncitori.next(i);

end loop;

return rezultat;

END Numar\_muncitori\_adt;

/

DECLARE

BEGIN

DBMS\_OUTPUT.PUT\_LINE('Numarul de angajati este '|| Numar\_muncitori\_adt(5, 2));

DBMS\_OUTPUT.PUT\_LINE('Numarul de angajati este '|| Numar\_muncitori\_adt(5, 3));

END;

# Problema

Graphical user interface, text, application, email

Description automatically generated

Graphical user interface, text, application

Description automatically generated

-- Afisati pentru fiecare magazin toti angajatii care lucreaza la el.

CREATE OR REPLACE PROCEDURE Angajati\_magazin\_adt is

CURSOR magazine IS SELECT id\_magazin, nume FROM magazin;

CURSOR angajati IS SELECT id\_magazin, nume, prenume FROM angajat\_magazin;

id\_m magazin.id\_magazin%type;

nume\_m magazin.nume%type;

id\_a angajat\_magazin.id\_magazin%type;

nume\_a angajat\_magazin.nume%type;

prenume\_a angajat\_magazin.prenume%type;

BEGIN

OPEN magazine;

LOOP

FETCH magazine INTO id\_m, nume\_m;

EXIT WHEN magazine%NOTFOUND;

DBMS\_OUTPUT.PUT\_LINE ('Nume magazin: '|| nume\_m);

OPEN angajati;

LOOP

FETCH angajati INTO id\_a, nume\_a, prenume\_a;

EXIT WHEN angajati%NOTFOUND;

IF id\_m = id\_a THEN

DBMS\_OUTPUT.PUT\_LINE(nume\_a || ' ' || prenume\_a);

END IF;

END LOOP;

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

CLOSE angajati;

END LOOP;

CLOSE magazine;

END Angajati\_magazin\_adt;

/

DECLARE

BEGIN

Angajati\_magazin\_adt;

END;

/

# Problema

Graphical user interface, text, application, email

Description automatically generated

Graphical user interface, text, application

Description automatically generated

-- Afisati numarul de muncitori care au lucrat la fabricarea unui parfum dat ca parametru.

CREATE OR REPLACE FUNCTION Muncitori\_parfumuri\_adt(nume\_parfum in parfum.nume%type) return number is rezultat number(10);

nume\_invalid exception;

nume\_check number;

BEGIN

select count(\*)

into nume\_check

from parfum

where nume\_parfum = parfum.nume;

if nume\_check = 0 then

raise nume\_invalid;

end if;

select count(\*)

into rezultat

from parfum p

join fabrica f on p.id\_fabrica = f.id\_fabrica

join muncitor m on f.id\_fabrica = m.id\_fabrica

where lower(p.nume) = lower(nume\_parfum);

return rezultat;

EXCEPTION -- nu prea pot aparea exceptii pentru ca countul mereu va pune ceva in variabile, si daca nu se gaseste nimic in tabele, countul va fi 0, ceea ce e ok

when nume\_invalid then

DBMS\_OUTPUT.PUT\_LINE('Nu exista parfumuri cu numele dat.');

when others then

DBMS\_OUTPUT.PUT\_LINE('Alta eroare: ' || SQLCODE || ' - ' || SQLERRM);

END Muncitori\_parfumuri\_adt;

/

DECLARE

BEGIN

-- alte exceptii nu pot fi obtinute

DBMS\_OUTPUT.PUT\_LINE('Numarul de muncitori este '|| Muncitori\_parfumuri\_adt('Gucci'));

DBMS\_OUTPUT.PUT\_LINE('Numarul de muncitori este '|| Muncitori\_parfumuri\_adt('Guscci'));

END;

/

# Problema

Graphical user interface, text, application, email

Description automatically generated

Graphical user interface, text, application

Description automatically generated

-- Dandu-se id-ul unei comenzi si numele unui parfum aflati din ce tara provine parfumul comandat (din ce tara este magazinul de unde a fost comandat).

CREATE OR REPLACE PROCEDURE Tara\_parfum\_adt(nume\_parfum in parfum.nume%type, idul\_comenzii in comanda.id\_comanda%type) is

nume\_tara tara.nume%type;

BEGIN

select t.nume

into nume\_tara

from tara t

join locatie l on t.id\_tara = l.id\_tara

join magazin m on m.id\_locatie = l.id\_locatie

join comanda c on c.id\_magazin = m.id\_magazin

join parfum p on p.id\_parfum = c.id\_parfum

where lower(p.nume) = lower(nume\_parfum) and c.id\_comanda = idul\_comenzii;

DBMS\_OUTPUT.PUT\_LINE('Tara este ' || nume\_tara);

EXCEPTION

when NO\_DATA\_FOUND then

DBMS\_OUTPUT.PUT\_LINE('Nu exista parfumul pe comanda data sau nu exista comanda: ' || SQLCODE || ' - ' || SQLERRM);

when TOO\_MANY\_ROWS then

DBMS\_OUTPUT.PUT\_LINE('Parfumul are mai multe tari de provenienta: ' || SQLCODE || ' - ' || SQLERRM);

when others then

DBMS\_OUTPUT.PUT\_LINE('Alta eroare: ' || SQLCODE || ' - ' || SQLERRM);

END Tara\_parfum\_adt;

/

DECLARE

BEGIN

-- cred ca doar NO\_DATA\_FOUND poate fi obtinuta cu ce am facut mai sus

Tara\_parfum\_adt('Armani', 8);

Tara\_parfum\_adt('Armani', 7);

Tara\_parfum\_adt('Armani', 37);

END;

/

# Trigger

Graphical user interface, text, application, email

Description automatically generated

-- Creati un trigger care sa permita modificarea tabelului muncitor doar in prima zi a lunii, caci numai atunci se vrea sa se faca angajari/concedieri/schimbari salariale.

CREATE OR REPLACE TRIGGER salariu\_check\_adt

BEFORE INSERT OR UPDATE OR DELETE ON muncitor

BEGIN

IF (TO\_CHAR(SYSDATE,'DD') != '01') THEN

RAISE\_APPLICATION\_ERROR(-20001, 'Tabelul muncitor poate fi modificat doar in prima zi a lunii!');

END IF;

END;

/

DROP TRIGGER salariu\_check\_adt;

-- verificare

INSERT INTO Muncitor(ID\_Muncitor, Nume, Prenume, Salariu, Data\_angajarii, Specializare, ID\_Fabrica) VALUES(3000, 'Alexandru', 'Vasilaru', 800, TO\_DATE('17/12/2015', 'DD/MM/YYYY'), 'Sticlar', 'FAB1');

# Trigger

Graphical user interface, text, application, email

Description automatically generated

-- Creati un trigger care nu permite micsorarea sumei unei comenzi odata ce a fost plasata.

CREATE OR REPLACE TRIGGER comanda\_check\_adt

BEFORE UPDATE OF suma ON comanda FOR EACH ROW

BEGIN

IF (:OLD.suma > :NEW.suma) THEN

RAISE\_APPLICATION\_ERROR(-20002, 'Suma unei comenzi nu poate fi micsorata!');

END IF;

END;

/

DROP TRIGGER comanda\_check\_adt;

-- verificare

UPDATE comanda

SET suma = 4

WHERE id\_comanda = 8;

# Trigger

Graphical user interface, text, application, email

Description automatically generated

-- Creati un tabel care va retine date despre comenzile date de utilizator (numele utilizatorului, data, actiunea si tabelul). Apoi creati un trigger care adauga in tabel fiecare comanda de creare

-- stergere sau alterare de tabel impreuna cu informatiile aferente.

CREATE TABLE istorie\_comenzi(utilizator varchar2(30), data date, actiune varchar2(30), tabel varchar2(30));

CREATE OR REPLACE TRIGGER Completare\_tabel

BEFORE CREATE or ALTER or DROP ON DATABASE

BEGIN

IF ora\_sysevent = 'CREATE' THEN

INSERT INTO istorie\_comenzi VALUES(user, sysdate, 'CREARE', ora\_dict\_obj\_name);

ELSIF ora\_sysevent = 'ALTER' THEN

INSERT INTO istorie\_comenzi VALUES(user, sysdate, 'MODIFICARE', ora\_dict\_obj\_name);

ELSIF ora\_sysevent = 'DROP' THEN

INSERT INTO istorie\_comenzi VALUES(user, sysdate, 'STERGERE' ,ora\_dict\_obj\_name);

END IF;

END;

/

drop trigger completare\_tabel;

create table persoana(nume varchar2(20), prenume varchar2(20));

alter table persoana add varsta number(10);

drop table persoana;

select \* from istorie\_comenzi;

# Pachet

Graphical user interface, text, application

Description automatically generated

Graphical user interface, text, application

Description automatically generated

Graphical user interface, text, application

Description automatically generated

Graphical user interface, text, application, email

Description automatically generated

Graphical user interface, text, application

Description automatically generated

-- Creati un pachet care sa contina toate functiile declarate anterior

CREATE OR REPLACE PACKAGE pachet\_adt AS

FUNCTION Numar\_muncitori\_adt(n in number, m in number) return number;

PROCEDURE Angajati\_magazin\_adt;

FUNCTION Muncitori\_parfumuri\_adt(nume\_parfum in parfum.nume%type) return number;

PROCEDURE Tara\_parfum\_adt(nume\_parfum in parfum.nume%type, idul\_comenzii in comanda.id\_comanda%type);

END pachet\_adt;

/

CREATE OR REPLACE PACKAGE BODY pachet\_adt AS

--

FUNCTION Numar\_muncitori\_adt(n in number, m in number) return number is rezultat number(10);

TYPE tablou\_muncitori IS TABLE OF muncitor%ROWTYPE INDEX BY BINARY\_INTEGER;

TYPE vector\_fabrici IS VARRAY(100) OF fabrica%ROWTYPE;

muncitori tablou\_muncitori;

fabrici vector\_fabrici;

i binary\_integer;

j binary\_integer;

BEGIN

select \*

bulk collect into muncitori

from( select \* from muncitor order by data\_angajarii )

where rownum <= n;

select \*

bulk collect into fabrici

from( select \* from fabrica order by data\_infintare desc )

where rownum <= m;

i := muncitori.first;

rezultat := 0;

while (i is not null) loop

j := fabrici.first;

while (j is not null) loop

if muncitori(i).id\_fabrica = fabrici(j).id\_fabrica then

rezultat := rezultat + 1;

end if;

j := fabrici.next(j);

end loop;

i := muncitori.next(i);

end loop;

return rezultat;

END;

--

PROCEDURE Angajati\_magazin\_adt is

CURSOR magazine IS SELECT id\_magazin, nume FROM magazin;

CURSOR angajati IS SELECT id\_magazin, nume, prenume FROM angajat\_magazin;

id\_m magazin.id\_magazin%type;

nume\_m magazin.nume%type;

id\_a angajat\_magazin.id\_magazin%type;

nume\_a angajat\_magazin.nume%type;

prenume\_a angajat\_magazin.prenume%type;

BEGIN

OPEN magazine;

LOOP

FETCH magazine INTO id\_m, nume\_m;

EXIT WHEN magazine%NOTFOUND;

DBMS\_OUTPUT.PUT\_LINE ('Nume magazin: '|| nume\_m);

OPEN angajati;

LOOP

FETCH angajati INTO id\_a, nume\_a, prenume\_a;

EXIT WHEN angajati%NOTFOUND;

IF id\_m = id\_a THEN

DBMS\_OUTPUT.PUT\_LINE(nume\_a || ' ' || prenume\_a);

END IF;

END LOOP;

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

CLOSE angajati;

END LOOP;

CLOSE magazine;

END;

--

FUNCTION Muncitori\_parfumuri\_adt(nume\_parfum in parfum.nume%type) return number is rezultat number(10);

nume\_invalid exception;

nume\_check number;

BEGIN

select count(\*)

into nume\_check

from parfum

where nume\_parfum = parfum.nume;

if nume\_check = 0 then

raise nume\_invalid;

end if;

select count(\*)

into rezultat

from parfum p

join fabrica f on p.id\_fabrica = f.id\_fabrica

join muncitor m on f.id\_fabrica = m.id\_fabrica

where lower(p.nume) = lower(nume\_parfum);

return rezultat;

EXCEPTION -- nu prea pot aparea exceptii pentru ca countul mereu va pune ceva in variabile

when nume\_invalid then

DBMS\_OUTPUT.PUT\_LINE('Nu exista parfumuri cu numele dat.');

when others then

DBMS\_OUTPUT.PUT\_LINE('Alta eroare: ' || SQLCODE || ' - ' || SQLERRM);

END;

--

PROCEDURE Tara\_parfum\_adt(nume\_parfum in parfum.nume%type, idul\_comenzii in comanda.id\_comanda%type) is

nume\_tara tara.nume%type;

BEGIN

select t.nume

into nume\_tara

from tara t

join locatie l on t.id\_tara = l.id\_tara

join magazin m on m.id\_locatie = l.id\_locatie

join comanda c on c.id\_magazin = m.id\_magazin

join parfum p on p.id\_parfum = c.id\_parfum

where lower(p.nume) = lower(nume\_parfum) and c.id\_comanda = idul\_comenzii;

DBMS\_OUTPUT.PUT\_LINE('Tara este ' || nume\_tara);

EXCEPTION

when NO\_DATA\_FOUND then

DBMS\_OUTPUT.PUT\_LINE('Nu exista parfumul pe comanda data sau nu exista comanda: ' || SQLCODE || ' - ' || SQLERRM);

when TOO\_MANY\_ROWS then

DBMS\_OUTPUT.PUT\_LINE('Parfumul are mai multe tari de provenienta: ' || SQLCODE || ' - ' || SQLERRM);

when others then

DBMS\_OUTPUT.PUT\_LINE('Alta eroare: ' || SQLCODE || ' - ' || SQLERRM);

END;

END pachet\_adt;

/

# Pachet

A screenshot of a computer

Description automatically generated

Graphical user interface, text, application, email

Description automatically generated

Graphical user interface, text, application, email

Description automatically generated

Graphical user interface, text, application

Description automatically generated

Graphical user interface, text, application

Description automatically generated

Graphical user interface, text, application, email

Description automatically generated

Graphical user interface, text, application

Description automatically generated

CREATE OR REPLACE PACKAGE pachet\_aux\_adt AS

TYPE tablou\_muncitori IS TABLE OF muncitor%ROWTYPE INDEX BY BINARY\_INTEGER;

muncitori tablou\_muncitori;

TYPE vector\_angajati IS VARRAY(100) OF angajat\_magazin%ROWTYPE;

angajati vector\_angajati;

PROCEDURE Muncitori\_platiti\_bine\_adt;

PROCEDURE Angajati\_platiti\_bine\_adt(nume\_parfum in parfum.nume%type);

FUNCTION Regiune\_muncitor(nume\_muncitor in muncitor.nume%type) return regiune.nume%type;

FUNCTION Parfum\_tara(nume\_parfum in parfum.nume%type) return varchar2;

END pachet\_aux\_adt;

/

CREATE OR REPLACE PACKAGE BODY pachet\_aux\_adt AS

-- Afisati muncitorii care au salariu mai mare decat media salariilor colegilor de continent, neluand in considerare

-- la medie angajatii angajati inainte de anul 2000.

PROCEDURE Muncitori\_platiti\_bine\_adt is

BEGIN

select a.\*

bulk collect into muncitori

from muncitor a

join fabrica b on a.id\_fabrica=b.id\_fabrica

join locatie c on b.id\_locatie=c.id\_locatie

join tara d on c.id\_tara=d.id\_tara

join regiune e on d.id\_regiune=e.id\_regiune

where salariu>=

(select avg(salariu)

from muncitor y

join fabrica x on y.id\_fabrica=x.id\_fabrica

join locatie w on x.id\_locatie=w.id\_locatie

join tara t on w.id\_tara=t.id\_tara

join regiune r on t.id\_regiune=r.id\_regiune

where e.id\_regiune=r.id\_regiune and TO\_CHAR(y.data\_angajarii, 'YYYY')>='2000');

END;

-- Sa se afiseze numele tuturor angajatilor din magazin care au salariu mai mare decat media salariala a muncitorilor care lucreaza

-- la fabricarea al carui nume e dat ca parametru

PROCEDURE Angajati\_platiti\_bine\_adt(nume\_parfum in parfum.nume%type) is

i binary\_integer;

BEGIN

select \*

bulk collect into angajati

from angajat\_magazin

where salariu>=

(select avg(a.salariu)

from muncitor a

join fabrica b on a.id\_fabrica=b.id\_fabrica

join parfum c on b.id\_fabrica=c.id\_fabrica

where lower(id\_parfum)=lower(nume\_parfum))

order by nume;

i := angajati.first;

WHILE (i is not null) LOOP

DBMS\_OUTPUT.PUT\_LINE(angajati(i).nume);

i := angajati.next(i);

END LOOP;

END;

--Afisati pentru fiecare muncitor numele regiunii in care munceste. Daca nu munceste in nicio regiune, afisati 'Nedefinit'.

FUNCTION Regiune\_muncitor(nume\_muncitor in muncitor.nume%type) return regiune.nume%type is rezultat regiune.nume%type;

BEGIN

select e.nume

into rezultat

from muncitor a

left outer join fabrica b on a.id\_fabrica=b.id\_fabrica

left outer join locatie c on b.id\_locatie=c.id\_locatie

left outer join tara d on c.id\_tara=d.id\_tara

left outer join regiune e on d.id\_regiune=e.id\_regiune

where a.nume = nume\_muncitor;

return rezultat;

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

DBMS\_OUTPUT.PUT\_LINE('Numele introdus este incorect ' || SQLCODE || ' - ' || SQLERRM);

WHEN TOO\_MANY\_ROWS THEN

DBMS\_OUTPUT.PUT\_LINE('Muncitorul lucreaza pe mai multe continente ' || SQLCODE || ' - ' || SQLERRM);

WHEN OTHERS THEN

DBMS\_OUTPUT.PUT\_LINE('Alta eroare! ' || SQLCODE || ' - ' || SQLERRM);

END;

FUNCTION Parfum\_tara(nume\_parfum in parfum.nume%type) return varchar2 is rezultat varchar2(100);

tara\_productie locatie.id\_tara%type;

tara\_depozitare locatie.id\_tara%type;

BEGIN

select l.id\_tara

into tara\_productie

from parfum p

join fabrica f on p.id\_fabrica = f.id\_fabrica

join locatie l on l.id\_locatie = f.id\_locatie

where lower(p.nume) = lower(nume\_parfum);

select l.id\_tara

into tara\_depozitare

from parfum p

join depozit d on p.id\_depozit = d.id\_depozit

join locatie l on d.id\_locatie = l.id\_locatie

where lower(p.nume) = lower(nume\_parfum);

IF tara\_productie = tara\_depozitare THEN

rezultat := 'TRUE';

ELSE

rezultat := 'FALSE';

END IF;

return rezultat;

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

DBMS\_OUTPUT.PUT\_LINE('Nu a fost gasit parfumul ' || SQLCODE || ' - ' || SQLERRM);

WHEN TOO\_MANY\_ROWS THEN

DBMS\_OUTPUT.PUT\_LINE('Parfumul are mai multe tari de provenienta sau de vanzare ' || SQLCODE || ' - ' || SQLERRM);

WHEN OTHERS THEN

DBMS\_OUTPUT.PUT\_LINE('Alta eroare! ' || SQLCODE || ' - ' || SQLERRM);

END;

END pachet\_aux\_adt;

/

-- Trigger ca 'muncitori' sa fie mereu actualizat

CREATE OR REPLACE TRIGGER muncitori\_check

AFTER INSERT OR UPDATE OR DELETE ON muncitor

BEGIN

PACHET\_AUX\_ADT.MUNCITORI\_PLATITI\_BINE\_ADT;

END;

/

-- Testari

DECLARE

i binary\_integer;

BEGIN

PACHET\_AUX\_ADT.MUNCITORI\_PLATITI\_BINE\_ADT;

i := PACHET\_AUX\_ADT.muncitori.first;

WHILE (i is not null) LOOP

DBMS\_OUTPUT.PUT\_LINE(PACHET\_AUX\_ADT.muncitori(i).id\_muncitor);

i := PACHET\_AUX\_ADT.muncitori.next(i);

END LOOP;

END;

/

DECLARE

BEGIN

PACHET\_AUX\_ADT.Angajati\_platiti\_bine\_adt('GCI');

END;

/

DECLARE

BEGIN

DBMS\_OUTPUT.PUT\_LINE('Regiunea este ' || PACHET\_AUX\_ADT.Regiune\_muncitor('Bob'));

END;

/

DECLARE

BEGIN

DBMS\_OUTPUT.PUT\_LINE('Parfumul este fabricat si depozitat in aceeasi tara: ' || PACHET\_AUX\_ADT.Parfum\_tara('Davidoff'));

END;

/