PROIECT SGBD

FABRICA DE MOBILA

Nume: Cucos Maria-Marianita

Grupa: 243

1. **Prezentați pe scurt baza de date.**
2. Descrierea modelului real, a utilității acestuia și a regulilor de funcționare.

O fabrica de mobila, a carei locatie si nume il cunoastem, dispune de un numar de hale.

In interiorul acestora sunt organizate departamente diferite.

In evidenta angajatilor, regasim toate informatiile necesare despre acestia.

Fabrica dispune de un stoc al materialelor si de un catalog, in care clientii pot vedea ce modele de mobila sunt produse.

In momentul in care cantitatea unui material sau obiect ce intra in alcatuirea constructiei mobilei scade sub 5, se efectueaza o comanda catre furnizor.

Clientii pot comanda produse din catalog, sau pot da o comanda personalizata. Pentru comanda personalizata se va atasa un link catre un document ce cuprinde informatiile necesare. Cererea le poate fi acceptata sau nu. Dupa plasarea comenzii, vor primi termenul limita pana la care le va fi expediata.

Transportul mobilei este realizata de un curier.

1. Prezentarea constrângerilor (restricții, reguli) impuse asupra modelului.

Fabrica cuprinde mai multe hali.

O hala cuprinde unul sau mai multe departamente.

Un department apartine unei singure hali.

Un angajat lucreaza intr-un singur departament.

Un anumit tip de material se achizitioneaza de la un singur furnizor.

Furnizorii vand mai multe tipuri de materiale.

Clientii comanda unul sau mai multe obiecte.

Fiecare comanda personalizata primește un calificativ: admis sau respins.

Curierul livreaza una sau mai multe comenzi finalizate.

* 1. **ERD & Diagrama conceptuala**

**lucreaza**

M(1)

ANGAJAT

FURNIZOR

1

1

**livreaza**

DEPARTAMENT

FACTURA

CURIER

CLIENT

PRODUS

MATERIAL

HALA

FABRICA

**vinde**

M(1)

M(1)

**cuprinde**

1

M(1)

M(1)

1

1

**are**

**cuprinde**

1

**proceseaza**

COMANDA\_PRODUS

M(1)

M(1)

1

1(0)

STANDARD

M(1)

**cuprinde**

M(0)

1(0)

ISA

1

M(1)

M

ISA

PERSONALIZATA

**efectueaza**

1

**insumeaza**

M

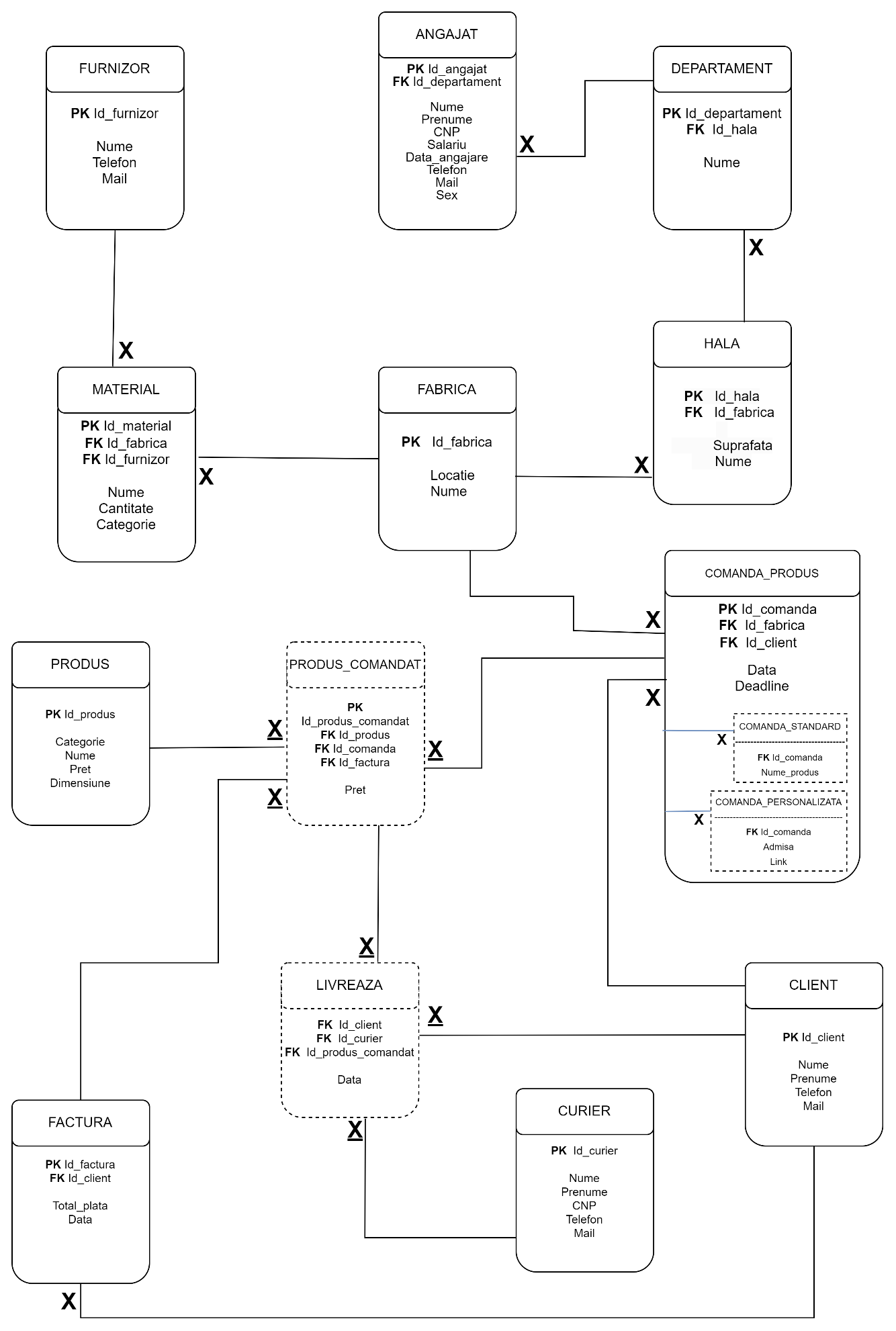
1

M(0)

M

M(1)

**plateste**



1. **Implementați în Oracle diagrama conceptuală realizată: definiți toate tabelele, implementând toate constrângerile de integritate necesare (chei primare, cheile externe etc).**
2. **Adăugați informații coerente în tabelele create (minim 5 înregistrări pentru fiecare entitate independentă; minim 10 înregistrări pentru tabela asociativă).**

CREATE TABLE FABRICA (

Id\_fabrica number GENERATED BY DEFAULT ON NULL AS IDENTITY START WITH 1 INCREMENT BY 1,

Locatie varchar2(200) NOT NULL,

Nume varchar2(100) NOT NULL,

CONSTRAINT PKEY\_FABRICA PRIMARY KEY(Id\_fabrica)

);

INSERT INTO FABRICA(locatie,nume)

VALUES ('Bucuresti','Mob Expert');

INSERT INTO FABRICA(locatie,nume)

VALUES ('Ploiesti','Mob Expert');

INSERT INTO FABRICA(locatie,nume)

VALUES ('Bucuresti','The home');

INSERT INTO FABRICA(locatie,nume)

VALUES ('Constanta','Your furniture');

INSERT INTO FABRICA(locatie,nume)

VALUES ('Brasov','Mob Design');

select \* from FABRICA;

CREATE TABLE HALA(

Id\_hala number GENERATED BY DEFAULT ON NULL AS IDENTITY START WITH 1 INCREMENT BY 1,

Suprafata FLOAT NOT NULL,

Nume VARCHAR2(100) NOT NULL,

Id\_fabrica number,

CONSTRAINT SUPRAFATA\_HALA CHECK (Suprafata>=150 AND Suprafata<=1000),

CONSTRAINT FK\_HALA FOREIGN KEY(Id\_fabrica) REFERENCES FABRICA(Id\_fabrica),

CONSTRAINT PKEY\_HALA PRIMARY KEY (Id\_hala)

);

INSERT INTO HALA(suprafata, nume, id\_fabrica)

VALUES (200,'corp A',1 );

INSERT INTO HALA(suprafata, nume, id\_fabrica)

VALUES (500,'corp AB',1 );

INSERT INTO HALA(suprafata, nume, id\_fabrica)

VALUES (250,'corp B',2 );

INSERT INTO HALA(suprafata, nume, id\_fabrica)

VALUES (900,'corp 1',3 );

INSERT INTO HALA(suprafata, nume, id\_fabrica)

VALUES (300,'corp 2',3 );

INSERT INTO HALA(suprafata, nume, id\_fabrica)

VALUES (500,'corp A',4 );

INSERT INTO HALA(suprafata, nume, id\_fabrica)

VALUES (680,'corp B',5 );

SELECT \* FROM HALA;

CREATE TABLE DEPARTAMENT(

Id\_departament number GENERATED BY DEFAULT ON NULL AS IDENTITY START WITH 1 INCREMENT BY 1,

Nume VARCHAR2(100) NOT NULL,

Id\_hala INT,

CONSTRAINT PKEY\_DEPARTAMENT PRIMARY KEY(Id\_departament),

CONSTRAINT FK\_DEPARTAMENT FOREIGN KEY(Id\_hala) REFERENCES HALA(Id\_hala) );

INSERT INTO DEPARTAMENT(nume, id\_hala)

VALUES ('impachetare', 2);

INSERT INTO DEPARTAMENT(nume, id\_hala)

VALUES ('modelare', 2);

INSERT INTO DEPARTAMENT(nume, id\_hala)

VALUES ('ansamblare', 3);

INSERT INTO DEPARTAMENT(nume, id\_hala)

VALUES ('design', 4);

INSERT INTO DEPARTAMENT(nume, id\_hala)

VALUES ('finisare', 3);

INSERT INTO DEPARTAMENT(nume, id\_hala)

VALUES ('livrare', 6);

INSERT INTO DEPARTAMENT(nume, id\_hala)

VALUES ('taiere', 7);

INSERT INTO DEPARTAMENT(nume, id\_hala)

VALUES ('design', 7);

SELECT \* FROM DEPARTAMENT;

CREATE TABLE ANGAJAT(

Id\_angajat number GENERATED BY DEFAULT ON NULL AS IDENTITY START WITH 1 INCREMENT BY 1,

Nume VARCHAR2(50) NOT NULL,

Prenume VARCHAR2(100) NOT NULL,

CNP varchar2(20) NOT NULL,

Salariu FLOAT DEFAULT 0,

Data\_angajare DATE NOT NULL,

Telefon varchar2(20) NOT NULL,

Sex VARCHAR2(10) NOT NULL,

Mail VARCHAR2(100),

Id\_departament INT,

CONSTRAINT PKEY\_ANGAJAT PRIMARY KEY(Id\_angajat),

CONSTRAINT CNP\_ANGAJAT UNIQUE (CNP), CHECK (LENGTH(CNP)=13),

CONSTRAINT SEX\_ANGAJAT CHECK (Sex in ('F','M')),

CONSTRAINT FK\_ANGAJAT FOREIGN KEY(Id\_departament) REFERENCES DEPARTAMENT(Id\_departament)

);

INSERT INTO ANGAJAT (nume, prenume, cnp, salariu,data\_angajare,telefon,sex,mail,id\_departament)

VALUES ('Popa','Alex', 1760523567832, 3000, to\_date('2010-07-16', 'yyyy-mm-dd') , '0763247942', 'M', 'popaalex@yahoo.com',2);

INSERT INTO ANGAJAT (nume, prenume, cnp, salariu,data\_angajare,telefon,sex,mail,id\_departament)

VALUES ('Mincu','Elena', 2950319789954 , 4500, to\_date('2015-10-18','yyyy-mm-dd'), '0764674378', 'F', 'm\_elena@gmail.com',4);

INSERT INTO ANGAJAT (nume, prenume, cnp, salariu,data\_angajare,telefon,sex,mail,id\_departament)

VALUES ('Lazar','Claudiu', 5000219479822, 4000, to\_date('2021-09-01','yyyy-mm-dd'), '0762748895', 'M', 'claudiu\_lazar@gmail.com',4);

INSERT INTO ANGAJAT(nume, prenume, cnp, salariu,data\_angajare,telefon,sex,mail,id\_departament)

VALUES ('Zapachi','Mihai', 1850416452896, 3200, to\_date('2008-03-05','yyyy-mm-dd'), '0761487529', 'M', 'zapachi@gmail.com',6);

INSERT INTO ANGAJAT(nume, prenume, cnp, salariu,data\_angajare,telefon,sex,mail,id\_departament)

VALUES ('Constantin','Andrei', 1860421678942, 3000, to\_date('2019-07-12','yyyy-mm-dd'), '0764267481', 'M', null,5);

INSERT INTO ANGAJAT(nume, prenume, cnp, salariu,data\_angajare,telefon,sex,mail,id\_departament)

VALUES ('Alecu','Raluca', 2890627898429, 2800, to\_date('2020-11-08','yyyy-mm-dd'), '0762485571', 'F', 'ralu.alecu@yahoo.com',5);

INSERT INTO ANGAJAT(nume, prenume, cnp, salariu,data\_angajare,telefon,sex,mail,id\_departament)

VALUES ('Popa','Marian', 1990514589617, 5000, to\_date('2022-06-01','yyyy-mm-dd'), '0762458697', 'M', null,8);

INSERT INTO ANGAJAT (nume, prenume, cnp, salariu,data\_angajare,telefon,sex,mail,id\_departament)

VALUES ('Panaite','Valentin', 1850419526379, 4200, to\_date('2021-09-16','yyyy-mm-dd'), '0762154786', 'M', null,5);

SELECT \* FROM ANGAJAT;

drop table angajat;

CREATE TABLE FURNIZOR(

Id\_furnizor number GENERATED BY DEFAULT ON NULL AS IDENTITY START WITH 1 INCREMENT BY 1,

Nume VARCHAR2(100) NOT NULL,

Telefon varchar2(20) NOT NULL,

Mail VARCHAR2(100) NOT NULL,

CONSTRAINT PKEY\_FURNIZOR PRIMARY KEY(Id\_furnizor),

CONSTRAINT MAIL\_FURNIZOR UNIQUE(Mail), CHECK (Mail like '%\_@\_\_%.\_\_%')

);

INSERT INTO FURNIZOR(nume, telefon, mail)

VALUES ('AlexCo','0745267759','alexcoSRL@gmail.com');

INSERT INTO FURNIZOR (nume, telefon, mail)

VALUES ('Androstal','0761254895','androstal@gmail.com');

INSERT INTO FURNIZOR (nume, telefon, mail)

VALUES ('Deccons','0748264818','Deccons@gmail.com');

INSERT INTO FURNIZOR (nume, telefon, mail)

VALUES ('Inoveco','0746958725','InovecoSRL@gmail.com');

INSERT INTO FURNIZOR (nume, telefon, mail)

VALUES ('CRH Romania','0764259583','CRH\_Romania@yahoo.com');

INSERT INTO FURNIZOR (nume, telefon, mail)

VALUES ('Deppo','0761459027','deppoSRL@gmail.com');

SELECT \* FROM FURNIZOR;

drop table furnizor;

CREATE TABLE MATERIAL(

Id\_material number GENERATED BY DEFAULT ON NULL AS IDENTITY START WITH 1 INCREMENT BY 1,

Nume VARCHAR2(100) NOT NULL,

Cantitate number DEFAULT 0,

Categorie VARCHAR2(100) NOT NULL,

Id\_fabrica number,

CONSTRAINT FK1\_MATERIAL FOREIGN KEY(Id\_fabrica) REFERENCES FABRICA(Id\_fabrica),

Id\_furnizor number,

CONSTRAINT FK2\_MATERIAL FOREIGN KEY(Id\_furnizor) REFERENCES FURNIZOR(Id\_furnizor),

CONSTRAINT PKEY\_MATERIAL PRIMARY KEY(Id\_material)

);

INSERT INTO MATERIAL(nume, cantitate, categorie, id\_fabrica, id\_furnizor)

VALUES ('lemn stejar',100,'lemn', 1, 3);

INSERT INTO MATERIAL(nume, cantitate, categorie, id\_fabrica, id\_furnizor)

VALUES ('lemn mahon',120,'lemn', 1, 6);

INSERT INTO MATERIAL(nume, cantitate, categorie, id\_fabrica, id\_furnizor)

VALUES ('lemn cires',105,'lemn', 1, 3);

INSERT INTO MATERIAL(nume, cantitate, categorie, id\_fabrica, id\_furnizor)

VALUES ('sticla cenusie',8,'sticla', 5, 3);

INSERT INTO MATERIAL(nume, cantitate, categorie, id\_fabrica, id\_furnizor)

VALUES ('sticla neagra',50,'sticla', 1, 3);

INSERT INTO MATERIAL(nume, cantitate, categorie, id\_fabrica, id\_furnizor)

VALUES ('surub M5',5,'surub', 4, 5);

INSERT INTO MATERIAL(nume, cantitate, categorie, id\_fabrica, id\_furnizor)

VALUES ('maner auriu',34,'accesorii', 4, 2);

INSERT INTO MATERIAL(nume, cantitate, categorie, id\_fabrica, id\_furnizor)

VALUES ('surub M10',100,'surub', 2, 4);

INSERT INTO MATERIAL(nume, cantitate, categorie, id\_fabrica, id\_furnizor)

VALUES ('surub M30',9,'surub', 2, 4);

SELECT \* FROM MATERIAL;

CREATE TABLE PRODUS(

Id\_produs number GENERATED BY DEFAULT ON NULL AS IDENTITY START WITH 1 INCREMENT BY 1,

Categorie VARCHAR2(100) NOT NULL,

Nume VARCHAR2(100) NOT NULL,

Pret INT NOT NULL,

Dimensiune VARCHAR2(100) NOT NULL,

CONSTRAINT PKEY\_PRODUS PRIMARY KEY(Id\_produs)

);

INSERT INTO PRODUS(categorie, nume, pret, dimensiune)

VALUES ('dormitor', 'sifonier 3 usi', 2500, '250x50x200');

INSERT INTO PRODUS(categorie, nume, pret, dimensiune)

VALUES ('dormitor', 'sifonier 2 usi', 1900, '120x50x200');

INSERT INTO PRODUS(categorie, nume, pret, dimensiune)

VALUES ('dormitor', 'pat mahon', 2000, '200x150x40');

INSERT INTO PRODUS(categorie, nume, pret, dimensiune)

VALUES ('baie', 'dulap cu sertare', 900, '60x40x180');

INSERT INTO PRODUS(categorie, nume, pret, dimensiune)

VALUES ('sufragerie', 'masa cafea', 500, '100x60x50');

INSERT INTO PRODUS(categorie, nume, pret, dimensiune)

VALUES ('sufragerie', 'biblioteca', 4500, '270x40x200');

INSERT INTO PRODUS(categorie, nume, pret, dimensiune)

VALUES ('bucatarie', 'masa stejar', 1200, '200x60x80');

SELECT \* FROM PRODUS;

CREATE TABLE CLIENT(

Id\_client number GENERATED BY DEFAULT ON NULL AS IDENTITY START WITH 1 INCREMENT BY 1,

Nume VARCHAR2(100) NOT NULL,

Prenume VARCHAR2(100) NOT NULL,

Telefon VARCHAR2(20) NOT NULL,

Mail VARCHAR2(100) NOT NULL,

CONSTRAINT PKEY\_CLIENT PRIMARY KEY(Id\_client),

CONSTRAINT MAIL\_CLIENT UNIQUE(Mail), CHECK (Mail like '%\_@\_\_%.\_\_%')

);

INSERT INTO CLIENT(nume, prenume, telefon, mail)

VALUES ('Diaconu','Madalin', '0763479586','m.diaconu78@gmail.com');

INSERT INTO CLIENT(nume, prenume, telefon, mail)

VALUES ('Florea','Andreea', '0761485722','andreea-fl@gmail.com');

INSERT INTO CLIENT(nume, prenume, telefon, mail)

VALUES ('Andrei','Costin', '0762175984','costi@gmail.com');

INSERT INTO CLIENT(nume, prenume, telefon, mail)

VALUES ('Balan','Roxana', '0764598746','RoxBalan@gmail.com');

INSERT INTO CLIENT(nume, prenume, telefon, mail)

VALUES ('Nita','Claudia', '0761457957','nita\_claudia@gmail.com');

INSERT INTO CLIENT(nume, prenume, telefon, mail)

VALUES ('Popa','Radu', '0763473695','poparadu@gmail.com');

SELECT \* FROM CLIENT;

drop table client;

CREATE TABLE FACTURA(

Id\_factura number GENERATED BY DEFAULT ON NULL AS IDENTITY START WITH 1 INCREMENT BY 1,

Total\_plata FLOAT NOT NULL,

Data DATE NOT NULL,

Id\_client INT,

CONSTRAINT FK\_FACTURA FOREIGN KEY(Id\_client) REFERENCES CLIENT(Id\_client),

CONSTRAINT PKEY\_FACTURA PRIMARY KEY(Id\_factura)

);

INSERT INTO FACTURA(total\_plata, data, id\_client)

VALUES (4500, to\_date('2022-03-11', 'yyyy-mm-dd'), 1);

INSERT INTO FACTURA(total\_plata, data, id\_client)

VALUES (7500,to\_date('2022-05-28', 'yyyy-mm-dd'), 2);

INSERT INTO FACTURA(total\_plata, data, id\_client)

VALUES (2500,to\_date('2022-01-22', 'yyyy-mm-dd'), 2);

INSERT INTO FACTURA(total\_plata, data, id\_client)

VALUES (4000,to\_date('2021-10-07', 'yyyy-mm-dd'), 3);

INSERT INTO FACTURA(total\_plata, data, id\_client)

VALUES (900,to\_date('2020-12-03', 'yyyy-mm-dd'), 4);

INSERT INTO FACTURA(total\_plata, data, id\_client)

VALUES (720,to\_date('2022-06-10', 'yyyy-mm-dd'), 5);

INSERT INTO FACTURA(total\_plata, data, id\_client)

VALUES (11000,to\_date('2021-07-13', 'yyyy-mm-dd'), 6);

SELECT \* FROM FACTURA;

drop table factura;

CREATE TABLE COMANDA\_PRODUS(

Id\_comanda number GENERATED BY DEFAULT ON NULL AS IDENTITY START WITH 1 INCREMENT BY 1,

Data DATE NOT NULL,

Deadline DATE NOT NULL,

Id\_fabrica INT,

CONSTRAINT FK1\_COMANDA\_PRODUS FOREIGN KEY(Id\_fabrica) REFERENCES FABRICA(Id\_fabrica),

Id\_client INT,

CONSTRAINT FK2\_COMANDA\_PRODUS FOREIGN KEY(Id\_client) REFERENCES CLIENT(Id\_client),

CONSTRAINT PKEY\_COMANDA\_PRODUS PRIMARY KEY(Id\_comanda)

);

INSERT INTO COMANDA\_PRODUS(data, deadline, id\_fabrica, id\_client)

VALUES(to\_date('20220415', 'yyyy-mm-dd'),to\_date('20220615', 'yyyy-mm-dd'), 1, 3);

INSERT INTO COMANDA\_PRODUS(data, deadline, id\_fabrica, id\_client)

VALUES(to\_date('20210810', 'yyyy-mm-dd'),to\_date('20211101', 'yyyy-mm-dd'), 1, 2);

INSERT INTO COMANDA\_PRODUS(data, deadline, id\_fabrica, id\_client)

VALUES(to\_date('20200415', 'yyyy-mm-dd'),to\_date('20200515', 'yyyy-mm-dd'), 1, 5);

INSERT INTO COMANDA\_PRODUS(data, deadline, id\_fabrica, id\_client)

VALUES(to\_date('20220127', 'yyyy-mm-dd'),to\_date('20220301', 'yyyy-mm-dd'), 3, 6);

INSERT INTO COMANDA\_PRODUS(data, deadline, id\_fabrica, id\_client)

VALUES(to\_date('20211117', 'yyyy-mm-dd'),to\_date('20220128', 'yyyy-mm-dd'), 3, 1);

INSERT INTO COMANDA\_PRODUS(data, deadline, id\_fabrica, id\_client)

VALUES(to\_date('20210801', 'yyyy-mm-dd'),to\_date('20210915', 'yyyy-mm-dd'), 1, 3);

INSERT INTO COMANDA\_PRODUS(data, deadline, id\_fabrica, id\_client)

VALUES(to\_date('20210508', 'yyyy-mm-dd'),to\_date('20210805', 'yyyy-mm-dd'), 3, 4);

INSERT INTO COMANDA\_PRODUS(data, deadline, id\_fabrica, id\_client)

VALUES(to\_date('20210219', 'yyyy-mm-dd'),to\_date('20210401', 'yyyy-mm-dd'), 4, 3);

INSERT INTO COMANDA\_PRODUS(data, deadline, id\_fabrica, id\_client)

VALUES(to\_date('20220108', 'yyyy-mm-dd'),to\_date('20220301', 'yyyy-mm-dd'), 2, 2);

INSERT INTO COMANDA\_PRODUS(data, deadline, id\_fabrica, id\_client)

VALUES(to\_date('20210510', 'yyyy-mm-dd'),to\_date('20210707', 'yyyy-mm-dd'), 2, 5);

INSERT INTO COMANDA\_PRODUS(data, deadline, id\_fabrica, id\_client)

VALUES(to\_date('20220504', 'yyyy-mm-dd'),to\_date('20220601', 'yyyy-mm-dd'), 4, 3);

INSERT INTO COMANDA\_PRODUS(data, deadline, id\_fabrica, id\_client)

VALUES(to\_date('20210725', 'yyyy-mm-dd'),to\_date('20210815', 'yyyy-mm-dd'), 5, 1);

SELECT \* FROM COMANDA\_PRODUS;

drop table COMANDA\_PRODUS;

CREATE TABLE COMANDA\_STANDARD(

Id\_comanda INT,

CONSTRAINT FK\_COMANDA\_STANDARD FOREIGN KEY(Id\_comanda) REFERENCES COMANDA\_PRODUS(Id\_comanda),

Nume\_produs VARCHAR2(100) NOT NULL);

INSERT INTO COMANDA\_STANDARD

VALUES(2, 'pat mahon');

INSERT INTO COMANDA\_STANDARD

VALUES(3, 'sifonier 3 usi');

INSERT INTO COMANDA\_STANDARD

VALUES(6, 'dulap baie');

INSERT INTO COMANDA\_STANDARD

VALUES(7, 'biblioteca');

INSERT INTO COMANDA\_STANDARD

VALUES(10, 'masa cafea');

SELECT \* FROM COMANDA\_STANDARD;

CREATE TABLE COMANDA\_PERSONALIZATA(

Id\_comanda INT,

CONSTRAINT FK\_COMANDA\_PERSONALIZATA FOREIGN KEY(Id\_comanda) REFERENCES COMANDA\_PRODUS(Id\_comanda),

Admisa VARCHAR2(20) NOT NULL,

CONSTRAINT ADMIS\_COMANDA CHECK (Admisa in ('F','T')),

Link VARCHAR2(300) NOT NULL );

INSERT INTO COMANDA\_PERSONALIZATA

VALUES(1, 'T', 'https://drive.google.com/file/d/13U8KQCObP8ke');

INSERT INTO COMANDA\_PERSONALIZATA

VALUES(4, 'T', 'https://drive.google.com/file/d/1er54g6g4');

INSERT INTO COMANDA\_PERSONALIZATA

VALUES(8, 'F', 'https://drive.google.com/file/d/5jydj4srj5j4sr');

INSERT INTO COMANDA\_PERSONALIZATA

VALUES(9, 'T', 'https://drive.google.com/file/d/tj4gejnzbs8md');

INSERT INTO COMANDA\_PERSONALIZATA

VALUES(5, 'F', 'https://drive.google.com/file/d/45hj1j54nb5fs2');

INSERT INTO COMANDA\_PERSONALIZATA

VALUES(11, 'T', 'https://drive.google.com/file/d/5e46b44b4v');

INSERT INTO COMANDA\_PERSONALIZATA

VALUES(12, 'T', 'https://drive.google.com/file/d/n4asdfghjkecu');

SELECT \* FROM COMANDA\_PERSONALIZATA;

drop table COMANDA\_PERSONALIZATA;

CREATE TABLE PRODUS\_COMANDAT(

Id\_produs\_comandat number GENERATED BY DEFAULT ON NULL AS IDENTITY START WITH 1 INCREMENT BY 1,

Pret FLOAT NOT NULL,

Id\_produs INT,

CONSTRAINT FK1\_PRODUS\_COMANDAT FOREIGN KEY(Id\_produs) REFERENCES PRODUS(Id\_produs),

Id\_comanda INT,

CONSTRAINT FK2\_PRODUS\_COMANDAT FOREIGN KEY(Id\_comanda) REFERENCES COMANDA\_PRODUS(Id\_comanda),

Id\_factura INT,

CONSTRAINT FK3\_PRODUS\_COMANDAT FOREIGN KEY(Id\_factura) REFERENCES FACTURA(Id\_factura),

CONSTRAINT PKEY\_PRODUS\_COMANDAT PRIMARY KEY(Id\_produs\_comandat)

);

INSERT INTO PRODUS\_COMANDAT(pret,id\_produs,id\_comanda, id\_factura)

VALUES(2000, 3, 2, 1);

INSERT INTO PRODUS\_COMANDAT(pret,id\_produs,id\_comanda, id\_factura)

VALUES(2500, 1, 3, 1);

INSERT INTO PRODUS\_COMANDAT(pret,id\_produs,id\_comanda, id\_factura)

VALUES(900, 4, 6, 5);

INSERT INTO PRODUS\_COMANDAT(pret,id\_produs,id\_comanda, id\_factura)

VALUES(4500, 6, 7, 7);

INSERT INTO PRODUS\_COMANDAT(pret,id\_produs,id\_comanda, id\_factura)

VALUES(500, 5, 10, 7);

INSERT INTO PRODUS\_COMANDAT(pret,id\_produs,id\_comanda, id\_factura)

VALUES(6000, 3, 1, 7);

INSERT INTO PRODUS\_COMANDAT(pret,id\_produs,id\_comanda, id\_factura)

VALUES(720, 7, 4, 6);

INSERT INTO PRODUS\_COMANDAT(pret,id\_produs,id\_comanda, id\_factura)

VALUES(2500, 3, 9, 3);

INSERT INTO PRODUS\_COMANDAT(pret,id\_produs,id\_comanda, id\_factura)

VALUES(4000, 2, 11, 4);

INSERT INTO PRODUS\_COMANDAT(pret,id\_produs,id\_comanda, id\_factura)

VALUES(7500, 6, 12, 2);

INSERT INTO PRODUS\_COMANDAT(pret,id\_produs,id\_comanda, id\_factura)

VALUES(720, 4, 4, 6);

SELECT \* FROM PRODUS\_COMANDAT;

CREATE TABLE CURIER(

Id\_curier number GENERATED BY DEFAULT ON NULL AS IDENTITY START WITH 1 INCREMENT BY 1,

Nume VARCHAR2(100) NOT NULL,

Prenume VARCHAR2(100) NOT NULL,

CNP VARCHAR2(20) NOT NULL,

Telefon VARCHAR2(20) NOT NULL,

Mail VARCHAR2(100),

CONSTRAINT PKEY\_CURIER PRIMARY KEY(Id\_curier),

CONSTRAINT CNP\_CURIER UNIQUE(CNP), CHECK (LENGTH(CNP)=13)

);

INSERT INTO CURIER(nume, prenume, cnp, telefon, mail)

VALUES('Popescu','Mihnea', '1780514589642','0764589316',null);

INSERT INTO CURIER(nume, prenume, cnp, telefon, mail)

VALUES('Marinescu','Razvan', '1840618562849','0746259873','razvan.marinescu@yahoo.com');

INSERT INTO CURIER(nume, prenume, cnp, telefon, mail)

VALUES('Birau','Alex', '1891008457892','0743298562','abirau@gmail.com');

INSERT INTO CURIER(nume, prenume, cnp, telefon, mail)

VALUES('Popescu','Robert', '1901118596248','0764458961','popescu\_robert@gmail.com');

INSERT INTO CURIER(nume, prenume, cnp, telefon, mail)

VALUES('Mincu','Catalin', '1850128125968','0741259862','mincu.cata@gmail.com');

SELECT \* FROM CURIER;

CREATE TABLE LIVREAZA(

Data DATE NOT NULL,

Id\_client INT,

CONSTRAINT FK1\_LIVRARE FOREIGN KEY(Id\_client) REFERENCES CLIENT(Id\_client),

Id\_curier INT,

CONSTRAINT FK2\_LIVRARE FOREIGN KEY(Id\_curier) REFERENCES CURIER(Id\_curier),

Id\_produs\_comandat INT,

CONSTRAINT FK3\_LIVRARE FOREIGN KEY(Id\_produs\_comandat) REFERENCES PRODUS\_COMANDAT (Id\_produs\_comandat) );

INSERT INTO LIVREAZA

VALUES(to\_date('2022-04-15', 'yyyy-mm-dd'),1,3,1);

INSERT INTO LIVREAZA

VALUES(to\_date('2022-04-15', 'yyyy-mm-dd'),1,3,2);

INSERT INTO LIVREAZA

VALUES(to\_date('2021-10-27', 'yyyy-mm-dd'), 2, 1, 10);

INSERT INTO LIVREAZA

VALUES(to\_date('2021-11-01', 'yyyy-mm-dd'), 3, 5, 9);

INSERT INTO LIVREAZA

VALUES(to\_date('2022-03-09', 'yyyy-mm-dd'), 2, 4, 8);

INSERT INTO LIVREAZA

VALUES(to\_date('2021-12-14', 'yyyy-mm-dd'), 4, 1, 3);

INSERT INTO LIVREAZA

VALUES(to\_date('2022-05-19', 'yyyy-mm-dd'), 5, 2, 7);

INSERT INTO LIVREAZA

VALUES(to\_date('2021-09-21', 'yyyy-mm-dd'), 6, 2, 5);

INSERT INTO LIVREAZA

VALUES(to\_date('2021-09-21', 'yyyy-mm-dd'), 6, 5, 4);

INSERT INTO LIVREAZA

VALUES(to\_date('2021-09-21', 'yyyy-mm-dd'), 6, 3, 6);

INSERT INTO LIVREAZA

VALUES(to\_date('2022-02-15', 'yyyy-mm-dd'), 5, 2, 21);

SELECT \* FROM LIVREAZA;

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

-- Sa se afiseze denumirea materialelor a caror cantitate este mai mica si egala cu 5, dar si numele si telefonul furnizorului de la care pot fi cumparate. De asemenea, pentru produsele comandate al carui pret este mai mare decat 500 si factura este emisa in luna decembrie, sa se aplice o reducere de 5%.

CREATE OR REPLACE PROCEDURE ex6

IS

TYPE tablou\_imbricat IS TABLE OF NUMBER;

t tablou\_imbricat := tablou\_imbricat();

TYPE tablou\_indexat IS TABLE OF produs\_comandat.id\_produs\_comandat%TYPE INDEX BY PLS\_INTEGER;

d tablou\_indexat;

nume\_material material.nume%TYPE;

nume\_furnizor furnizor.nume%TYPE;

telefon\_furnizor furnizor.telefon%TYPE;

pret\_vechi produs\_comandat.pret%TYPE;

pret\_nou produs\_comandat.pret%TYPE;

nr number;

exceptie\_1 exception;

exceptie\_2 exception;

BEGIN

select count(Id\_material)

into nr

from material

where cantitate<=5;

if nr = 0

then raise exceptie\_1;

end if;

select Id\_material

bulk collect into t

from material

where cantitate<=5;

dbms\_output.put\_line('Materiale insuficiente in stoc: ');

for i in t.first..t.last loop

select m.nume, f.nume, f.telefon

into nume\_material, nume\_furnizor, telefon\_furnizor

from material m, furnizor f

where m.Id\_furnizor = f.Id\_furnizor

and m.id\_material = t(i);

DBMS\_OUTPUT.PUT('ID material: '|| t(i) );

DBMS\_OUTPUT.NEW\_LINE;

DBMS\_OUTPUT.PUT('material: '||nume\_material||', furnizor: '||nume\_furnizor||' '||telefon\_furnizor );

DBMS\_OUTPUT.NEW\_LINE;

end loop;

DBMS\_OUTPUT.NEW\_LINE;

----------------------------------------------------------------------

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

select count(p.Id\_produs\_comandat)

into nr

from produs\_comandat p, factura f

where p.id\_factura = f.id\_factura

and p.pret > 500

and EXTRACT(MONTH FROM f.data) = 12;

if nr = 0

then raise exceptie\_2;

end if;

select p.Id\_produs\_comandat

bulk collect into d

from produs\_comandat p, factura f

where p.id\_factura = f.id\_factura

and p.pret > 500

and EXTRACT(MONTH FROM f.data) = 12;

FOR i IN d.FIRST..d.LAST LOOP

select pret

into pret\_vechi

from produs\_comandat

where id\_produs\_comandat = d(i);

DBMS\_OUTPUT.PUT('Pentru produsul comandat cu id-ul: '|| d(i) );

DBMS\_OUTPUT.NEW\_LINE;

DBMS\_OUTPUT.PUT('pret vechi: '||pret\_vechi || ' ');

DBMS\_OUTPUT.NEW\_LINE;

--maresc cu 5%

update produs\_comandat

set pret = pret - (pret \* 5/100)

where id\_produs\_comandat = d(i)

RETURNING pret

INTO pret\_nou;

DBMS\_OUTPUT.PUT('pret nou: '||pret\_nou );

DBMS\_OUTPUT.NEW\_LINE;

END LOOP;

EXCEPTION

WHEN exceptie\_1

THEN DBMS\_OUTPUT.PUT\_LINE('Nu exista materiale cu stoc insuficient.');

WHEN exceptie\_2

THEN DBMS\_OUTPUT.PUT\_LINE('Nu exista produse comandate care sa respecte conditiile pentru reducere.');

WHEN OTHERS

THEN DBMS\_OUTPUT.PUT\_LINE('Alta eroare!');

END ex6;

/

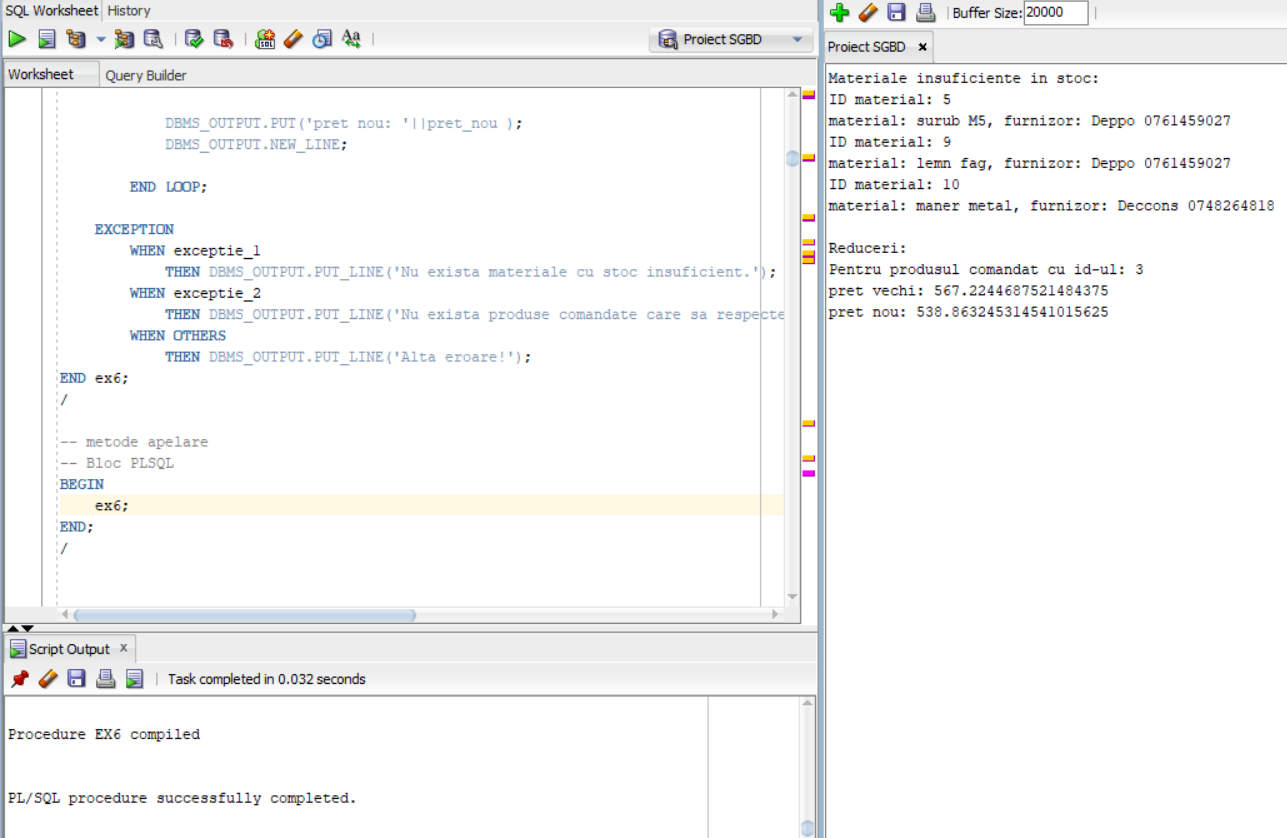
-- apelare

BEGIN

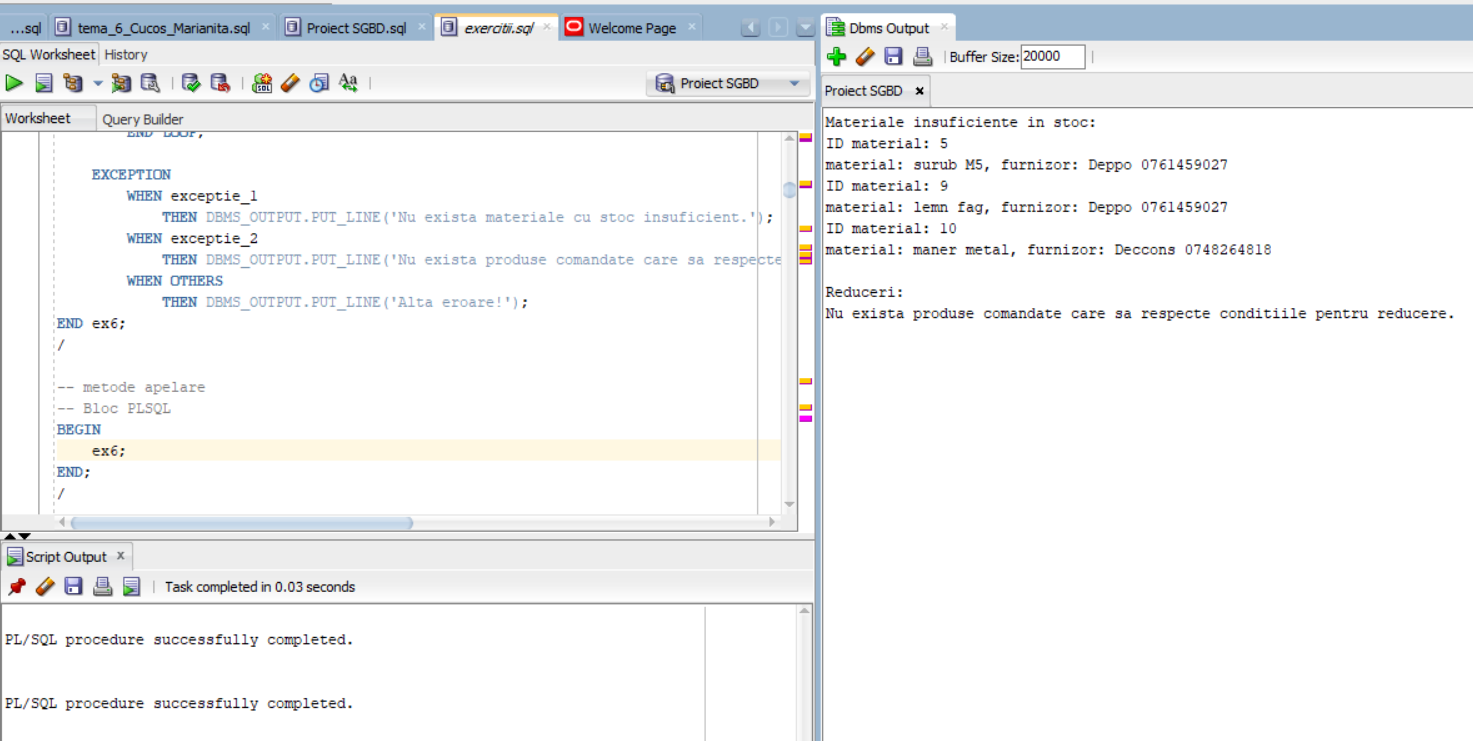
ex6;

END;

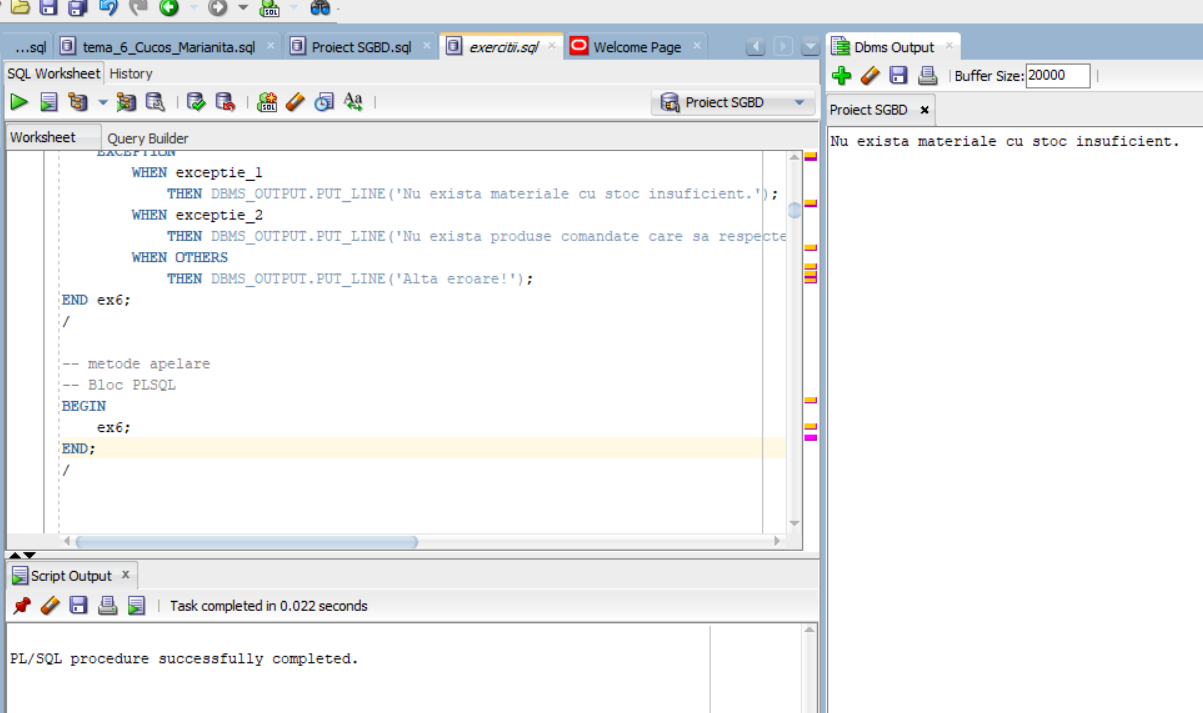
/

****

* Conditie: pret mai mic decat 500:

****

* Conditie: stoc sub o bucata

****

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

-- Pentru fiecare fabrica si fiecare departament al acesteia sa se afiseze numarul de angajati, suma salariilor si valoarea medie a veniturilor angajatilor

CREATE OR REPLACE PROCEDURE ex7

(fabrica\_id fabrica.id\_fabrica%TYPE)

IS

id\_dep departament.Id\_departament%TYPE;

nume\_dep departament.nume%TYPE;

a\_nume angajat.nume%TYPE;

a\_prenume angajat.prenume%TYPE;

a\_salariu angajat.salariu%TYPE;

n\_fab fabrica.nume%TYPE;

l\_fab fabrica.locatie%TYPE;

CURSOR c2(id departament.id\_departament%TYPE) IS

SELECT nume nume, prenume prenume, salariu salariu

FROM angajat

WHERE id\_departament = id;

ang number;

v\_lunar number;

v\_medie number;

nr number;

exc exception;

total\_ang number := 0;

total\_v\_lunar number := 0;

total\_v\_medie number(9,2) := 0;

BEGIN

select count(\*)

into nr

from fabrica

where id\_fabrica = fabrica\_id;

if nr = 0

then raise exc;

end if;

select nume, locatie

into n\_fab, l\_fab

from fabrica

where id\_fabrica = fabrica\_id;

DBMS\_OUTPUT.PUT\_LINE('Pentru fabrica '||n\_fab||' din '||l\_fab);

FOR i IN ( select d.id\_departament id\_dep, d.nume nume\_dep

from departament d, hala h

where h.id\_fabrica = fabrica\_id

and d.id\_hala = h.id\_hala)

LOOP

DBMS\_OUTPUT.NEW\_LINE();

DBMS\_OUTPUT.PUT\_LINE('Departament: ' || upper(i.nume\_dep) || '. Angajati:');

OPEN c2(i.id\_dep);

ang:=0;

v\_lunar := 0;

v\_medie := 0;

LOOP

FETCH c2 into a\_nume, a\_prenume, a\_salariu;

EXIT WHEN c2%NOTFOUND;

ang := ang +1;

v\_lunar := v\_lunar + a\_salariu;

IF ang != 0

THEN DBMS\_OUTPUT.PUT\_LINE(' - '|| ang || '. ' || a\_nume || ' ' || a\_prenume || ', salariu ' || a\_salariu);

END IF;

END LOOP;

close c2;

IF ang = 0

THEN DBMS\_OUTPUT.PUT\_LINE(' Nu sunt angajati aici.');

else

DBMS\_OUTPUT.NEW\_LINE();

DBMS\_OUTPUT.PUT\_LINE('Numarul de angajati este: '|| ang);

DBMS\_OUTPUT.PUT\_LINE('Valoarea lunara a veniturilor angajatilor este: '|| v\_lunar);

v\_medie := v\_lunar / ang;

DBMS\_OUTPUT.PUT\_LINE('Valoarea medie a veniturilor angajatilor este: '|| v\_medie);

END IF;

total\_ang := total\_ang + ang;

total\_v\_lunar := total\_v\_lunar + v\_lunar;

END LOOP;

DBMS\_OUTPUT.NEW\_LINE();

DBMS\_OUTPUT.NEW\_LINE();

DBMS\_OUTPUT.PUT\_LINE('Numarul total de angajati din fabrica este: '|| total\_ang);

DBMS\_OUTPUT.PUT\_LINE('Valoarea lunara totala a veniturilor angajatilor este: '|| total\_v\_lunar);

total\_v\_medie := total\_v\_lunar / total\_ang;

DBMS\_OUTPUT.PUT\_LINE('Valoarea medie totala a veniturilor angajatilor este: '|| total\_v\_medie);

EXCEPTION

WHEN exc

THEN DBMS\_OUTPUT.PUT\_LINE('Nu exista fabrica cu id-ul dat.');

WHEN ZERO\_DIVIDE

THEN DBMS\_OUTPUT.PUT\_LINE('Impartirea la 0 nu este permisa');

WHEN CURSOR\_ALREADY\_OPEN

THEN DBMS\_OUTPUT.PUT\_LINE('Cursor deja deschis.');

WHEN OTHERS

THEN DBMS\_OUTPUT.PUT\_LINE('Alta eroare!');

END ex7;

/

-- apelare

BEGIN

ex7('1');

DBMS\_OUTPUT.NEW\_LINE();

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

ex7('2');

DBMS\_OUTPUT.NEW\_LINE();

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

ex7('3');

DBMS\_OUTPUT.NEW\_LINE();

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

ex7('4');

DBMS\_OUTPUT.NEW\_LINE();

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

ex7('5');

DBMS\_OUTPUT.NEW\_LINE();

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

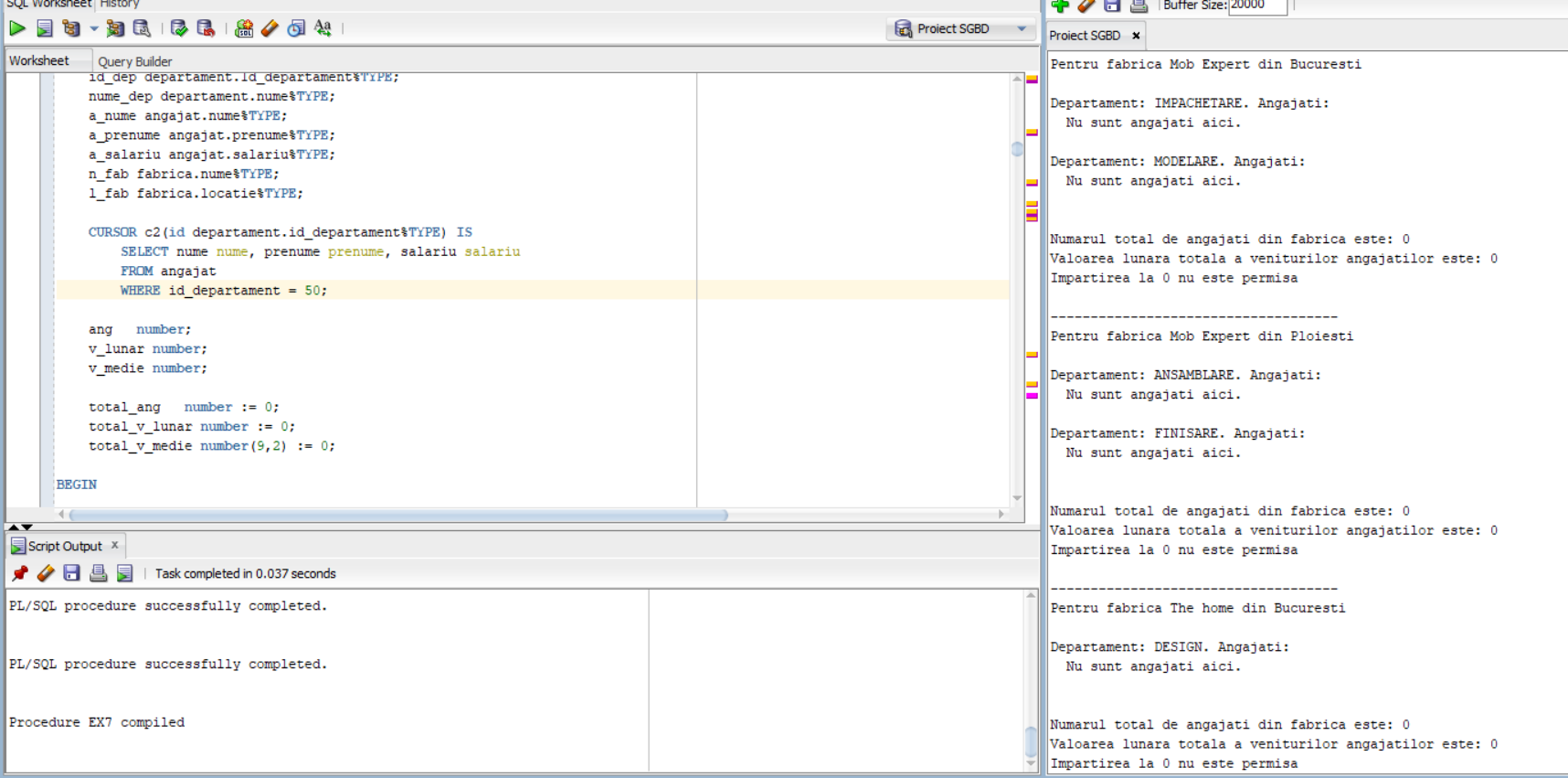
ex7('6'); -- id gresit

END;

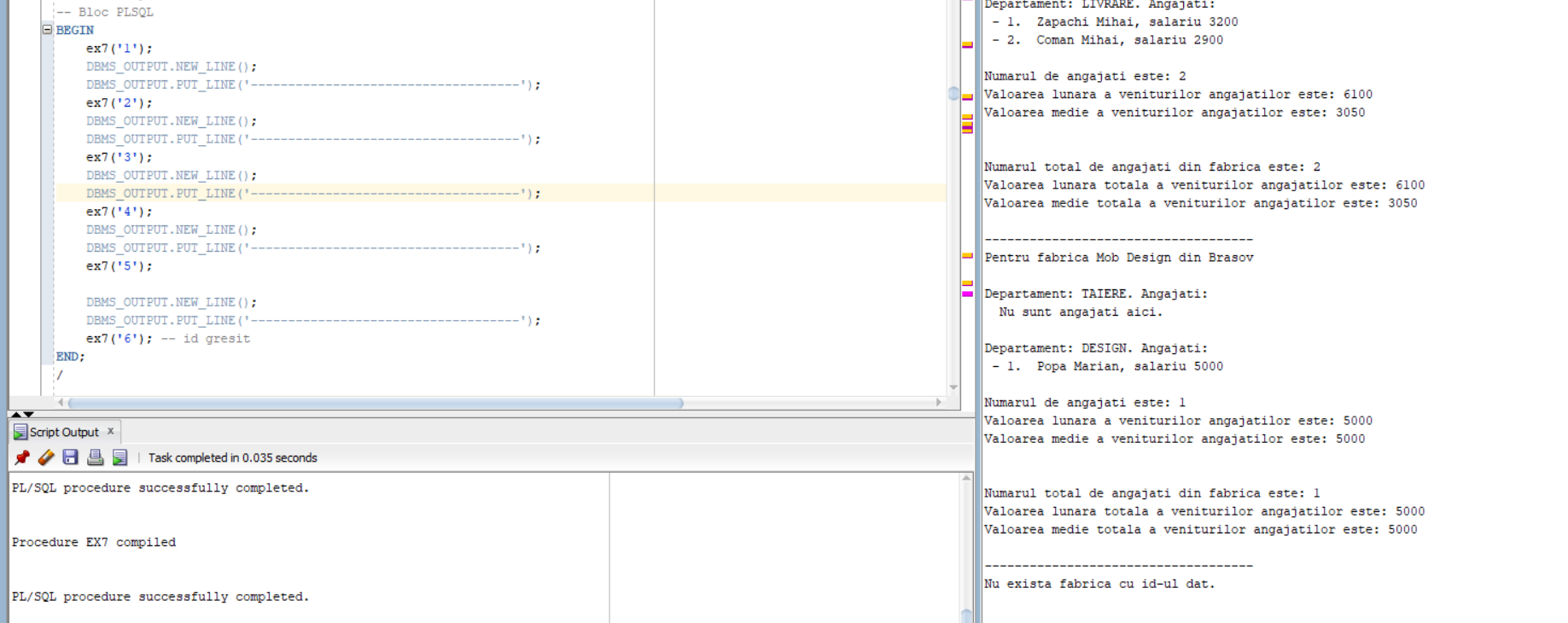
/



* Cursor gol

****

* id gresit

****

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

-- Se da numele unui client si se cere sa se calculeze totalul de plata de pe facturile acestuia, iar in cazul in care suma actuala difera, sa se actualizeze.

CREATE OR REPLACE FUNCTION ex8

(v\_nume client.nume%TYPE,

id\_fac factura.Id\_factura%type)

RETURN NUMBER

IS

pret\_total factura.total\_plata%type;

pret\_prod produs\_comandat.pret%type;

nr number;

exceptie1 exception;

exceptie2 exception;

BEGIN

select count(\*)

into nr

from client

where upper(nume) = upper(v\_nume);

if nr > 1

then raise exceptie1;

elsif nr = 0

then raise exceptie2;

end if;

SELECT SUM(p.pret)

INTO pret\_total

FROM produs\_comandat p, client c, factura f

WHERE c.Id\_client = f.Id\_client

and f.Id\_factura = p.Id\_factura

and UPPER(c.nume) = UPPER(v\_nume)

and f.id\_factura = id\_fac;

RETURN pret\_total;

EXCEPTION

WHEN exceptie1

THEN DBMS\_OUTPUT.PUT\_LINE('Exista mai multi clienti cu numele dat');

return 0;

WHEN exceptie2

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

return 0;

WHEN OTHERS

THEN DBMS\_OUTPUT.PUT\_LINE('Alta eroare!');

return 0;

END ex8;

/

--pt too many rows

INSERT INTO CLIENT(nume, prenume, telefon, mail)

VALUES ('Popa','Roxana', '0762567435','popa@gmail.com');

declare

total factura.total\_plata%TYPE;

v\_nume client.nume%TYPE := '&\_nume\_client';

val\_fct number;

exceptie exception;

nr number;

begin

select count(\*)

into nr

from client

where upper(nume) = upper(v\_nume);

if nr = 0

then raise exceptie;

end if;

for i in (select f.id\_factura id\_fac

from factura f, client c

where c.Id\_client = f.Id\_client

and UPPER(c.nume) = UPPER(v\_nume)) loop

DBMS\_OUTPUT.PUT\_LINE('Nume client: '||v\_nume);

select ex8(v\_nume,i.id\_fac) into val\_fct from dual;

select f.total\_plata

into total

from factura f, client c

where c.Id\_client = f.Id\_client

and UPPER(c.nume) = UPPER(v\_nume)

and f.id\_factura = i.id\_fac;

if total != val\_fct and val\_fct != 0 then

update factura

set total\_plata = val\_fct

where Id\_client = ( select f.id\_client

from client c, factura f

where c.Id\_client = f.Id\_client

and UPPER(c.nume) = UPPER(v\_nume)

and f.id\_factura = i.id\_fac);

DBMS\_OUTPUT.put\_line('Totalul a fost actualizat!');

elsif total = val\_fct and val\_fct != 0 then

DBMS\_OUTPUT.put\_line('Id factura: '||i.id\_fac);

DBMS\_OUTPUT.put\_line('Valorile sunt egale!');

end if;

end loop;

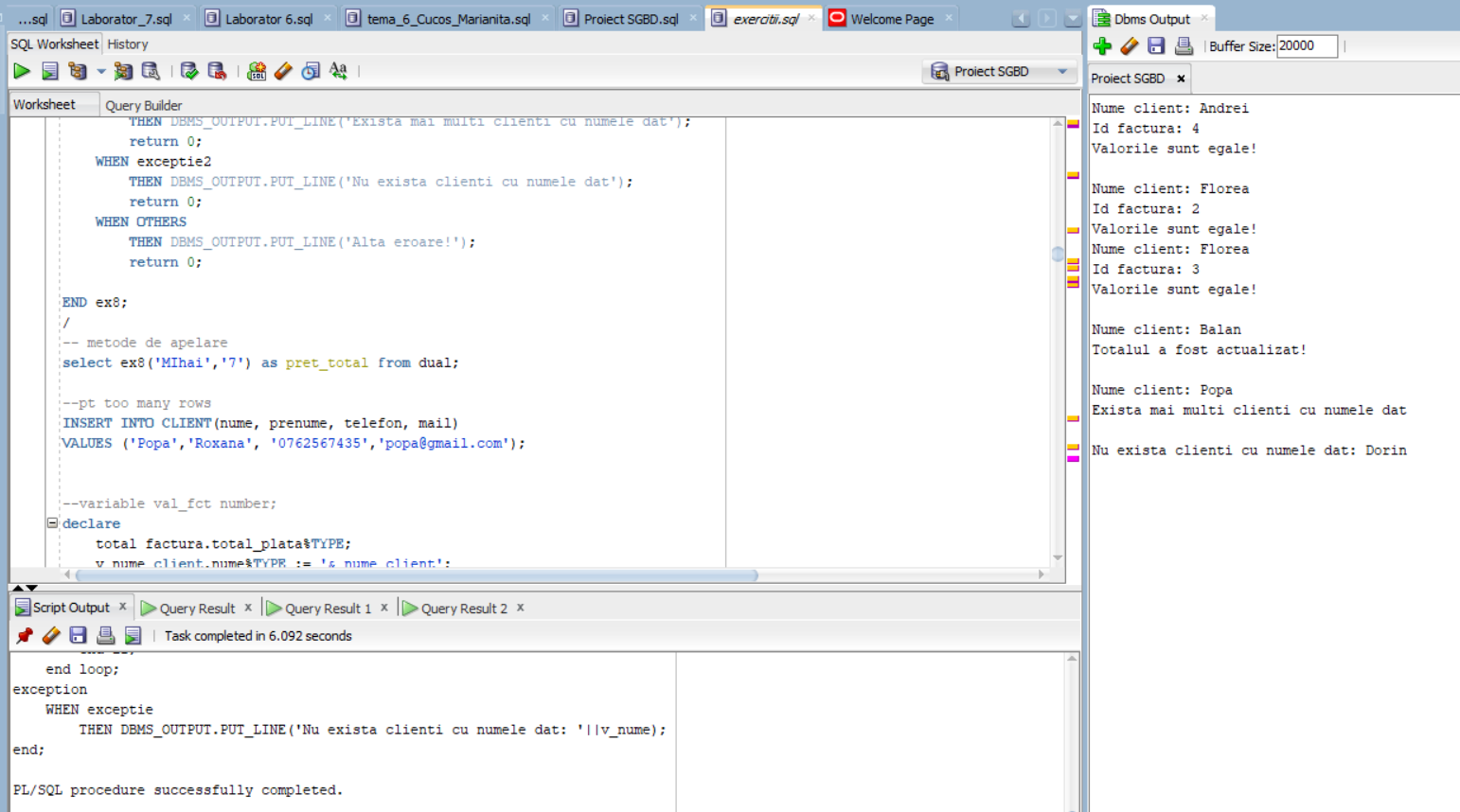
exception

WHEN exceptie

THEN DBMS\_OUTPUT.PUT\_LINE('Nu exista clienti cu numele dat: '||v\_nume);

end;

/

****

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

--Pentru prenumele unui curier dat ca parametru, sa se afiseze cate produse livreaza unui client, data livrarii si totalul de plata.

CREATE OR REPLACE PROCEDURE ex9

(c\_prenume curier.prenume%TYPE)

IS

total factura.total\_plata%TYPE;

data\_livrare livreaza.data%TYPE;

nr\_prod number;

curier varchar2(100);

client varchar2(100);

BEGIN

select max(c.nume||' '||c.prenume) curier, max(cl.nume||' '||cl.prenume) client,l.data, count(p.id\_factura) nr\_produse, f.total\_plata

into curier, client, data\_livrare, nr\_prod, total

from curier c JOIN livreaza l ON (l.id\_curier = c.id\_curier)

JOIN client cl ON (l.id\_client = cl.id\_client)

JOIN produs\_comandat p ON (l.id\_produs\_comandat = p.id\_produs\_comandat)

JOIN factura f ON (f.id\_client = cl.id\_client

and p.id\_factura = f.id\_factura)

where c.prenume = c\_prenume

group by p.id\_factura,f.total\_plata, l.data;

dbms\_output.put\_line('Curierul '||curier||' livreaza la data de '||data\_livrare||' clientului '||

client||' '||nr\_prod||' produse, cu un total de plata de '||total);

EXCEPTION

WHEN NO\_DATA\_FOUND

THEN dbms\_output.put\_line('Nu exista curieri cu prenumele introdus.');

WHEN TOO\_MANY\_ROWS

THEN dbms\_output.put\_line('Exista mai multi curieri cu prenumele introdus.');

WHEN OTHERS

THEN dbms\_output.put\_line('Alta eroare!');

END ex9;

/

-- apelare

BEGIN

ex9('Robert');

ex9('Calin');

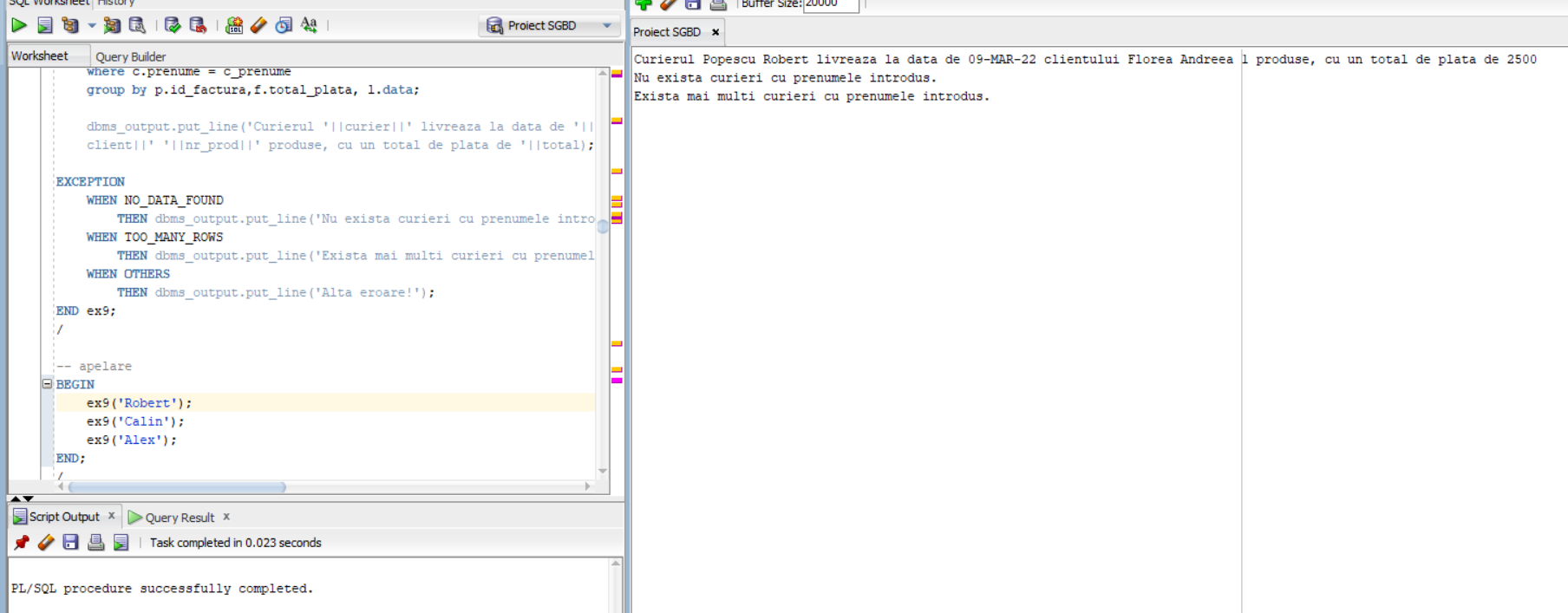
ex9('Alex');

END;

/

INSERT INTO CURIER(nume, prenume, cnp, telefon, mail)

VALUES('Moraru','Alex', '1891008457867','0743258562','alex\_m@gmail.com');

****

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

--trigger care permite inserarea, stergerea sau updatarea de informatii in tabela produs doar in intervalul orar 8-20, de luni pana vineri.

CREATE OR REPLACE TRIGGER trig\_ex10

BEFORE INSERT OR DELETE OR UPDATE on produs

BEGIN

IF (TO\_CHAR(SYSDATE,'D') = 1 or TO\_CHAR(SYSDATE,'D') = 7) OR (TO\_CHAR(SYSDATE,'HH24') NOT BETWEEN 8 AND 20)

THEN

IF INSERTING THEN

RAISE\_APPLICATION\_ERROR(-20001,'Inserarea in tabel este permisa doar in timpul programului de lucru!');

ELSIF DELETING THEN

RAISE\_APPLICATION\_ERROR(-20002,'Stergerea din tabel este permisa doar in timpul programului de lucru!');

ELSE

RAISE\_APPLICATION\_ERROR(-20003,'Actualizarile in tabel sunt permise doar in timpul programului de lucru!');

END IF;

END IF;

END;

/

INSERT INTO PRODUS(categorie, nume, pret, dimensiune)

VALUES ('bucatarie', 'scaun cires', 150, '100x40x40');

delete

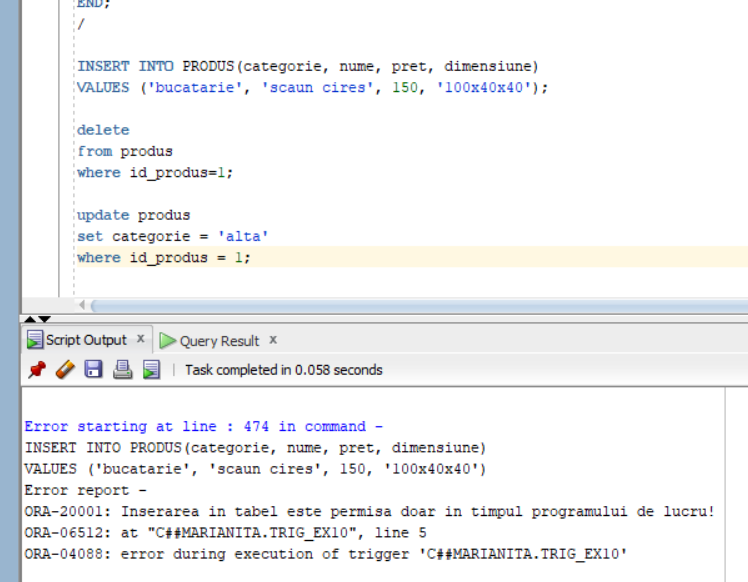
from produs

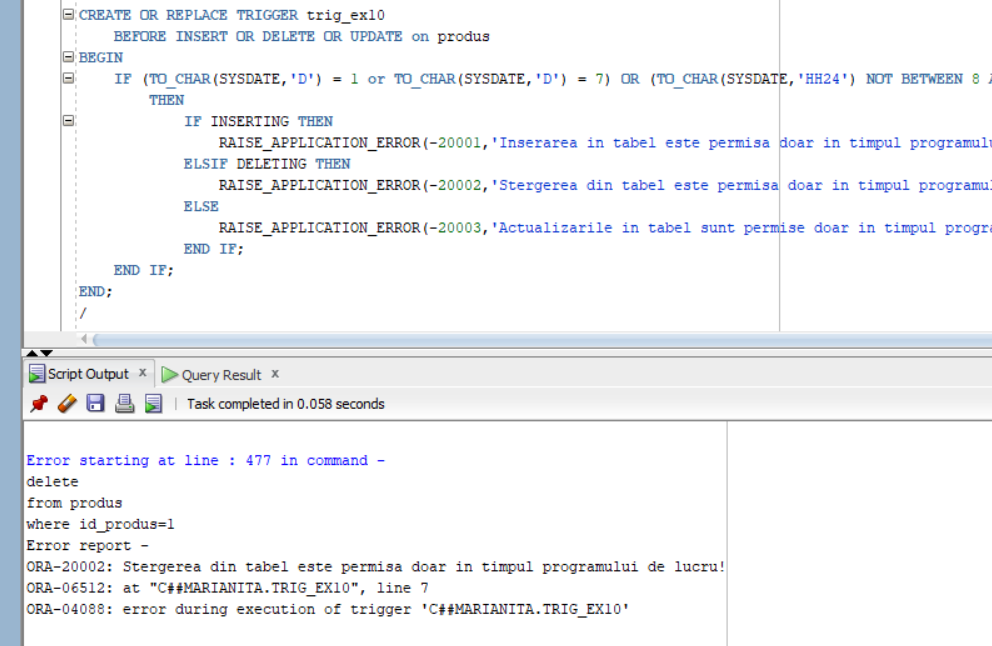
where id\_produs=1;

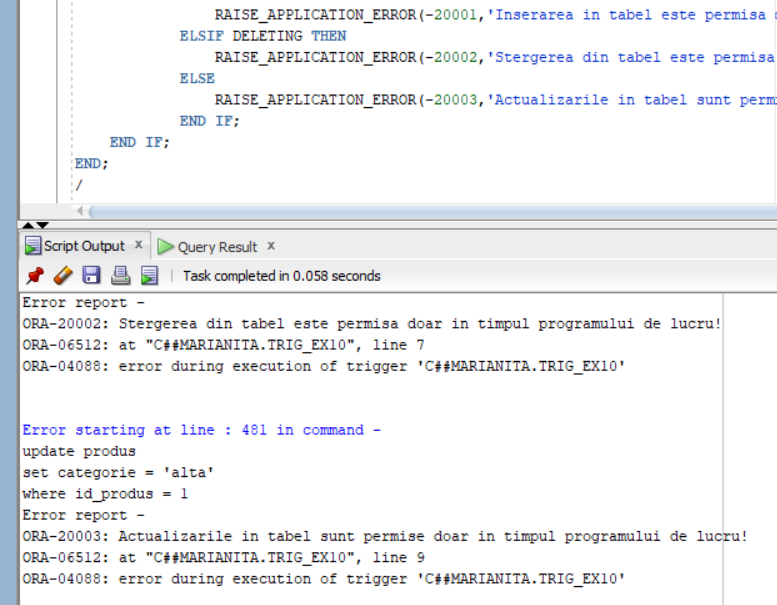
update produs

set categorie = 'alta'

where id\_produs = 1;

****

****

****

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

**--**trigger care nu permite updatarea categoriei unui produs

CREATE OR REPLACE TRIGGER trig\_ex11

BEFORE UPDATE OF categorie ON produs

FOR EACH ROW

WHEN (NEW.categorie <> OLD.categorie)

BEGIN

RAISE\_APPLICATION\_ERROR (-20000, 'Nu puteti modifica categoria unui produs din catalog!');

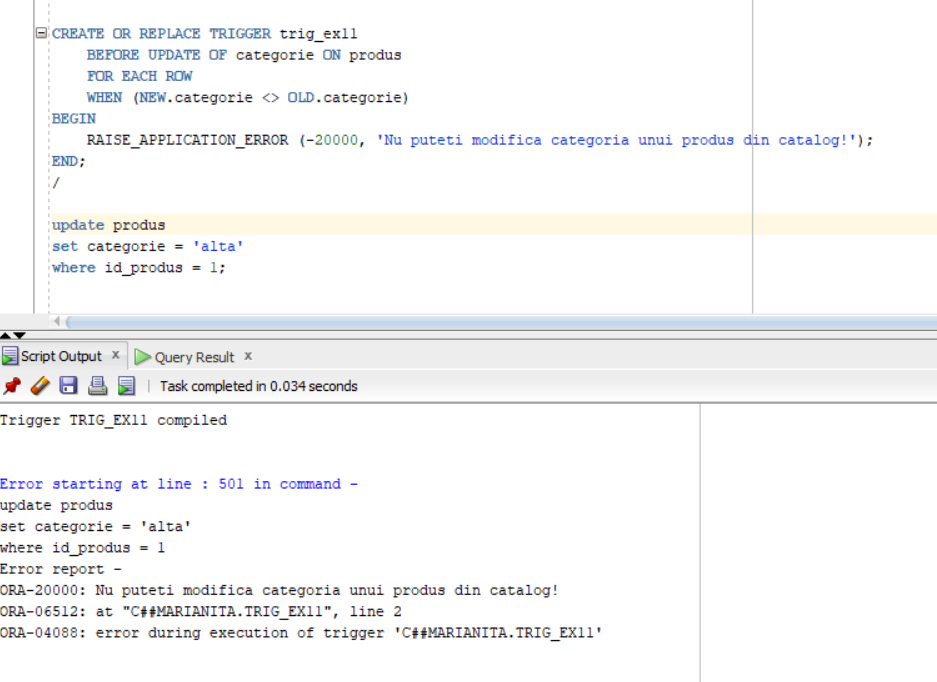
END;

/

update produs

set categorie = 'alta'

where id\_produs = 1;

****

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

--trigger prin intermediul caruia memorez intr-un tabel datele crearii, stergerii si modificarii unui tabel/functii/proceduri/trigger etc.

CREATE TABLE user\_action

(nume\_bd VARCHAR2(50),

user\_logat VARCHAR2(30),

eveniment VARCHAR2(20),

tip\_obiect\_referit VARCHAR2(30),

nume\_obiect\_referit VARCHAR2(30),

data TIMESTAMP(3));

CREATE OR REPLACE TRIGGER modificari

AFTER CREATE OR DROP OR ALTER ON SCHEMA

BEGIN

INSERT INTO user\_action

VALUES (SYS.DATABASE\_NAME, SYS.LOGIN\_USER,

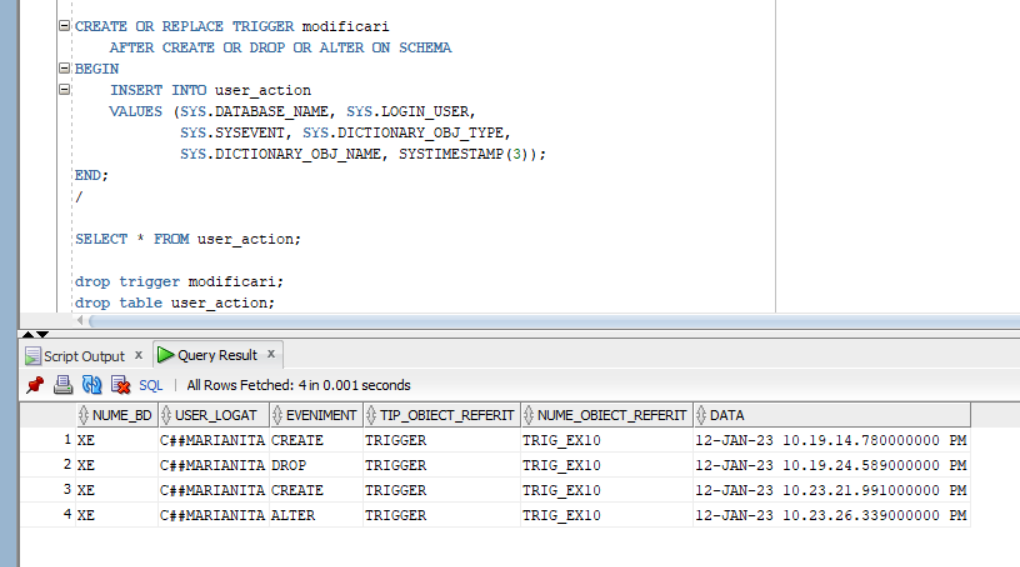
SYS.SYSEVENT, SYS.DICTIONARY\_OBJ\_TYPE,

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

END;

/

SELECT \* FROM user\_action;

****

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

CREATE OR REPLACE PACKAGE pachet\_ex13 AS

PROCEDURE ex6 ;

PROCEDURE ex7 (fabrica\_id fabrica.id\_fabrica%TYPE);

FUNCTION ex8 (v\_nume client.nume%TYPE, id\_fac factura.Id\_factura%type)

RETURN NUMBER ;

PROCEDURE ex9 (c\_prenume curier.prenume%TYPE);

end pachet\_ex13;

/

CREATE OR REPLACE PACKAGE BODY pachet\_ex13 AS

PROCEDURE ex6

IS

TYPE tablou\_imbricat IS TABLE OF NUMBER;

t tablou\_imbricat := tablou\_imbricat();

TYPE tablou\_indexat IS TABLE OF produs\_comandat.id\_produs\_comandat%TYPE INDEX BY PLS\_INTEGER;

d tablou\_indexat;

nume\_material material.nume%TYPE;

nume\_furnizor furnizor.nume%TYPE;

telefon\_furnizor furnizor.telefon%TYPE;

pret\_vechi produs\_comandat.pret%TYPE;

pret\_nou produs\_comandat.pret%TYPE;

nr number;

exceptie\_1 exception;

exceptie\_2 exception;

BEGIN

select count(Id\_material)

into nr

from material

where cantitate<=5;

if nr = 0

then raise exceptie\_1;

end if;

select Id\_material

bulk collect into t

from material

where cantitate<=5;

dbms\_output.put\_line('Materiale insuficiente in stoc: ');

for i in t.first..t.last loop

select m.nume, f.nume, f.telefon

into nume\_material, nume\_furnizor, telefon\_furnizor

from material m, furnizor f

where m.Id\_furnizor = f.Id\_furnizor

and m.id\_material = t(i);

DBMS\_OUTPUT.PUT('ID material: '|| t(i) );

DBMS\_OUTPUT.NEW\_LINE;

DBMS\_OUTPUT.PUT('material: '||nume\_material||', furnizor: '||nume\_furnizor||' '||telefon\_furnizor );

DBMS\_OUTPUT.NEW\_LINE;

end loop;

DBMS\_OUTPUT.NEW\_LINE;

----------------------------------------------------------------------

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

select count(p.Id\_produs\_comandat)

into nr

from produs\_comandat p, factura f

where p.id\_factura = f.id\_factura

and p.pret > 500

and EXTRACT(MONTH FROM f.data) = 12;

if nr = 0

then raise exceptie\_2;

end if;

select p.Id\_produs\_comandat

bulk collect into d

from produs\_comandat p, factura f

where p.id\_factura = f.id\_factura

and p.pret > 500

and EXTRACT(MONTH FROM f.data) = 12;

FOR i IN d.FIRST..d.LAST LOOP

select pret

into pret\_vechi

from produs\_comandat

where id\_produs\_comandat = d(i);

DBMS\_OUTPUT.PUT('Pentru produsul comandat cu id-ul: '|| d(i) );

DBMS\_OUTPUT.NEW\_LINE;

DBMS\_OUTPUT.PUT('pret vechi: '||pret\_vechi || ' ');

DBMS\_OUTPUT.NEW\_LINE;

--maresc cu 5%

update produs\_comandat

set pret = pret - (pret \* 5/100)

where id\_produs\_comandat = d(i)

RETURNING pret

INTO pret\_nou;

DBMS\_OUTPUT.PUT('pret nou: '||pret\_nou );

DBMS\_OUTPUT.NEW\_LINE;

END LOOP;

EXCEPTION

WHEN exceptie\_1

THEN DBMS\_OUTPUT.PUT\_LINE('Nu exista materiale cu stoc insuficient.');

WHEN exceptie\_2

THEN DBMS\_OUTPUT.PUT\_LINE('Nu exista produse comandate care sa respecte conditiile pentru reducere.');

WHEN OTHERS

THEN DBMS\_OUTPUT.PUT\_LINE('Alta eroare!');

END ex6;

PROCEDURE ex7

(fabrica\_id fabrica.id\_fabrica%TYPE)

IS

id\_dep departament.Id\_departament%TYPE;

nume\_dep departament.nume%TYPE;

a\_nume angajat.nume%TYPE;

a\_prenume angajat.prenume%TYPE;

a\_salariu angajat.salariu%TYPE;

n\_fab fabrica.nume%TYPE;

l\_fab fabrica.locatie%TYPE;

CURSOR c2(id departament.id\_departament%TYPE) IS

SELECT nume nume, prenume prenume, salariu salariu

FROM angajat

WHERE id\_departament = id;

ang number;

v\_lunar number;

v\_medie number;

nr number;

exc exception;

total\_ang number := 0;

total\_v\_lunar number := 0;

total\_v\_medie number(9,2) := 0;

BEGIN

select count(\*)

into nr

from fabrica

where id\_fabrica = fabrica\_id;

if nr = 0

then raise exc;

end if;

select nume, locatie

into n\_fab, l\_fab

from fabrica

where id\_fabrica = fabrica\_id;

DBMS\_OUTPUT.PUT\_LINE('Pentru fabrica '||n\_fab||' din '||l\_fab);

FOR i IN ( select d.id\_departament id\_dep, d.nume nume\_dep

from departament d, hala h

where h.id\_fabrica = fabrica\_id

and d.id\_hala = h.id\_hala)

LOOP

DBMS\_OUTPUT.NEW\_LINE();

DBMS\_OUTPUT.PUT\_LINE('Departament: ' || upper(i.nume\_dep) || '. Angajati:');

OPEN c2(i.id\_dep);

ang:=0;

v\_lunar := 0;

v\_medie := 0;

LOOP

FETCH c2 into a\_nume, a\_prenume, a\_salariu;

EXIT WHEN c2%NOTFOUND;

ang := ang +1;

v\_lunar := v\_lunar + a\_salariu;

IF ang != 0

THEN DBMS\_OUTPUT.PUT\_LINE(' - '|| ang || '. ' || a\_nume || ' ' || a\_prenume || ', salariu ' || a\_salariu);

END IF;

END LOOP;

close c2;

IF ang = 0

THEN DBMS\_OUTPUT.PUT\_LINE(' Nu sunt angajati aici.');

else

DBMS\_OUTPUT.NEW\_LINE();

DBMS\_OUTPUT.PUT\_LINE('Numarul de angajati este: '|| ang);

DBMS\_OUTPUT.PUT\_LINE('Valoarea lunara a veniturilor angajatilor este: '|| v\_lunar);

v\_medie := v\_lunar / ang;

DBMS\_OUTPUT.PUT\_LINE('Valoarea medie a veniturilor angajatilor este: '|| v\_medie);

END IF;

total\_ang := total\_ang + ang;

total\_v\_lunar := total\_v\_lunar + v\_lunar;

END LOOP;

DBMS\_OUTPUT.NEW\_LINE();

DBMS\_OUTPUT.NEW\_LINE();

DBMS\_OUTPUT.PUT\_LINE('Numarul total de angajati din fabrica este: '|| total\_ang);

DBMS\_OUTPUT.PUT\_LINE('Valoarea lunara totala a veniturilor angajatilor este: '|| total\_v\_lunar);

total\_v\_medie := total\_v\_lunar / total\_ang;

DBMS\_OUTPUT.PUT\_LINE('Valoarea medie totala a veniturilor angajatilor este: '|| total\_v\_medie);

EXCEPTION

WHEN exc

THEN DBMS\_OUTPUT.PUT\_LINE('Nu exista fabrica cu id-ul dat.');

WHEN ZERO\_DIVIDE

THEN DBMS\_OUTPUT.PUT\_LINE('Impartirea la 0 nu este permisa');

WHEN CURSOR\_ALREADY\_OPEN

THEN DBMS\_OUTPUT.PUT\_LINE('Cursor deja deschis.');

WHEN OTHERS

THEN DBMS\_OUTPUT.PUT\_LINE('Alta eroare!');

END ex7;

FUNCTION ex8

(v\_nume client.nume%TYPE,

id\_fac factura.Id\_factura%type)

RETURN NUMBER

IS

pret\_total factura.total\_plata%type;

pret\_prod produs\_comandat.pret%type;

nr number;

exceptie1 exception;

exceptie2 exception;

BEGIN

select count(\*)

into nr

from client

where upper(nume) = upper(v\_nume);

if nr > 1

then raise exceptie1;

elsif nr = 0

then raise exceptie2;

end if;

SELECT SUM(p.pret)

INTO pret\_total

FROM produs\_comandat p, client c, factura f

WHERE c.Id\_client = f.Id\_client

and f.Id\_factura = p.Id\_factura

and UPPER(c.nume) = UPPER(v\_nume)

and f.id\_factura = id\_fac;

RETURN pret\_total;

EXCEPTION

WHEN exceptie1

THEN DBMS\_OUTPUT.PUT\_LINE('Exista mai multi clienti cu numele dat');

return 0;

WHEN exceptie2

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

return 0;

WHEN OTHERS

THEN DBMS\_OUTPUT.PUT\_LINE('Alta eroare!');

return 0;

END ex8;

PROCEDURE ex9

(c\_prenume curier.prenume%TYPE)

IS

total factura.total\_plata%TYPE;

data\_livrare livreaza.data%TYPE;

nr\_prod number;

curier varchar2(100);

client varchar2(100);

BEGIN

select max(c.nume||' '||c.prenume) curier, max(cl.nume||' '||cl.prenume) client,l.data, count(p.id\_factura) nr\_produse, f.total\_plata

into curier, client, data\_livrare, nr\_prod, total

from curier c JOIN livreaza l ON (l.id\_curier = c.id\_curier)

JOIN client cl ON (l.id\_client = cl.id\_client)

JOIN produs\_comandat p ON (l.id\_produs\_comandat = p.id\_produs\_comandat)

JOIN factura f ON (f.id\_client = cl.id\_client

and p.id\_factura = f.id\_factura)

where c.prenume = c\_prenume

group by p.id\_factura,f.total\_plata, l.data;

dbms\_output.put\_line('Curierul '||curier||' livreaza la data de '||data\_livrare||' clientului '||

client||' '||nr\_prod||' produse, cu un total de plata de '||total);

EXCEPTION

WHEN NO\_DATA\_FOUND

THEN dbms\_output.put\_line('Nu exista curieri cu prenumele introdus.');

WHEN TOO\_MANY\_ROWS

THEN dbms\_output.put\_line('Exista mai multi curieri cu prenumele introdus.');

WHEN OTHERS

THEN dbms\_output.put\_line('Alta eroare!');

END ex9;

end pachet\_ex13;

/

-- apelare

declare

total factura.total\_plata%TYPE;

v\_nume client.nume%TYPE := '&\_nume\_client';

val\_fct number;

exceptie exception;

nr number;

BEGIN

DBMS\_OUTPUT.PUT\_LINE('----------------- ex 6 -------------------');

pachet\_ex13.ex6;

DBMS\_OUTPUT.NEW\_LINE();

DBMS\_OUTPUT.PUT\_LINE('----------------- ex 7 -------------------');

DBMS\_OUTPUT.NEW\_LINE();

pachet\_ex13.ex7('1');

DBMS\_OUTPUT.NEW\_LINE();

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

pachet\_ex13.ex7('2');

DBMS\_OUTPUT.NEW\_LINE();

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

pachet\_ex13.ex7('3');

DBMS\_OUTPUT.NEW\_LINE();

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

pachet\_ex13.ex7('4');

DBMS\_OUTPUT.NEW\_LINE();

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

pachet\_ex13.ex7('5');

DBMS\_OUTPUT.NEW\_LINE();

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

pachet\_ex13.ex7('6'); -- id gresit

DBMS\_OUTPUT.NEW\_LINE();

DBMS\_OUTPUT.PUT\_LINE('----------------- ex 8 -------------------');

DBMS\_OUTPUT.NEW\_LINE();

select count(\*)

into nr

from client

where upper(nume) = upper(v\_nume);

if nr = 0

then DBMS\_OUTPUT.PUT\_LINE('Nu exista clienti cu numele dat: '||v\_nume);

end if;

for i in (select f.id\_factura id\_fac

from factura f, client c

where c.Id\_client = f.Id\_client

and UPPER(c.nume) = UPPER(v\_nume)) loop

DBMS\_OUTPUT.PUT\_LINE('Nume client: '||v\_nume);

select pachet\_ex13.ex8(v\_nume,i.id\_fac) into val\_fct from dual;

select f.total\_plata

into total

from factura f, client c

where c.Id\_client = f.Id\_client

and UPPER(c.nume) = UPPER(v\_nume)

and f.id\_factura = i.id\_fac;

if total != val\_fct and val\_fct != 0 then

update factura

set total\_plata = val\_fct

where Id\_client = ( select f.id\_client

from client c, factura f

where c.Id\_client = f.Id\_client

and UPPER(c.nume) = UPPER(v\_nume)

and f.id\_factura = i.id\_fac);

DBMS\_OUTPUT.put\_line('Totalul a fost actualizat!');

elsif total = val\_fct and val\_fct != 0 then

DBMS\_OUTPUT.put\_line('Id factura: '||i.id\_fac);

DBMS\_OUTPUT.put\_line('Valorile sunt egale!');

end if;

end loop;

DBMS\_OUTPUT.NEW\_LINE();

DBMS\_OUTPUT.PUT\_LINE('----------------- ex 9 -------------------');

DBMS\_OUTPUT.NEW\_LINE();

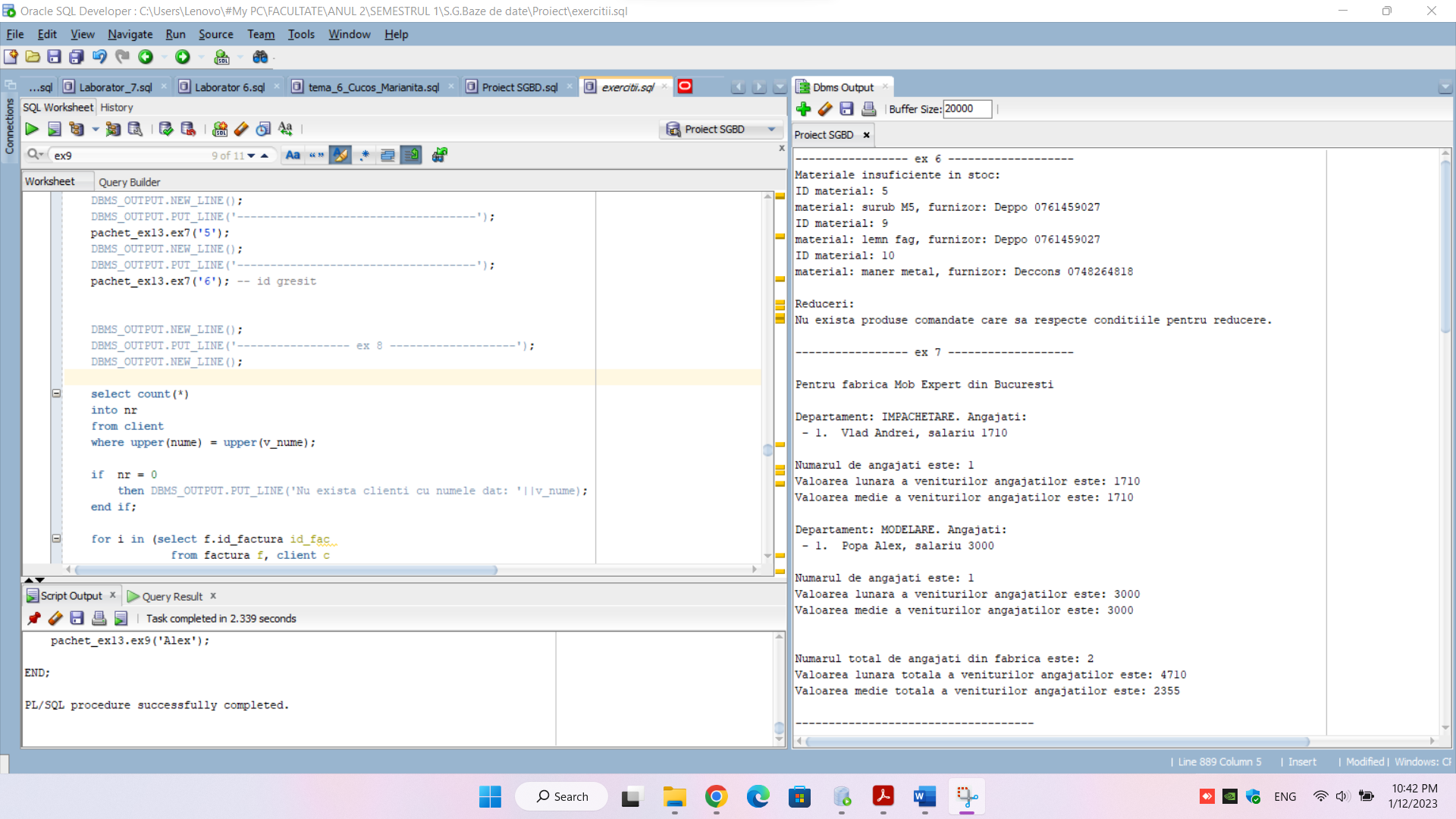
pachet\_ex13.ex9('Robert');

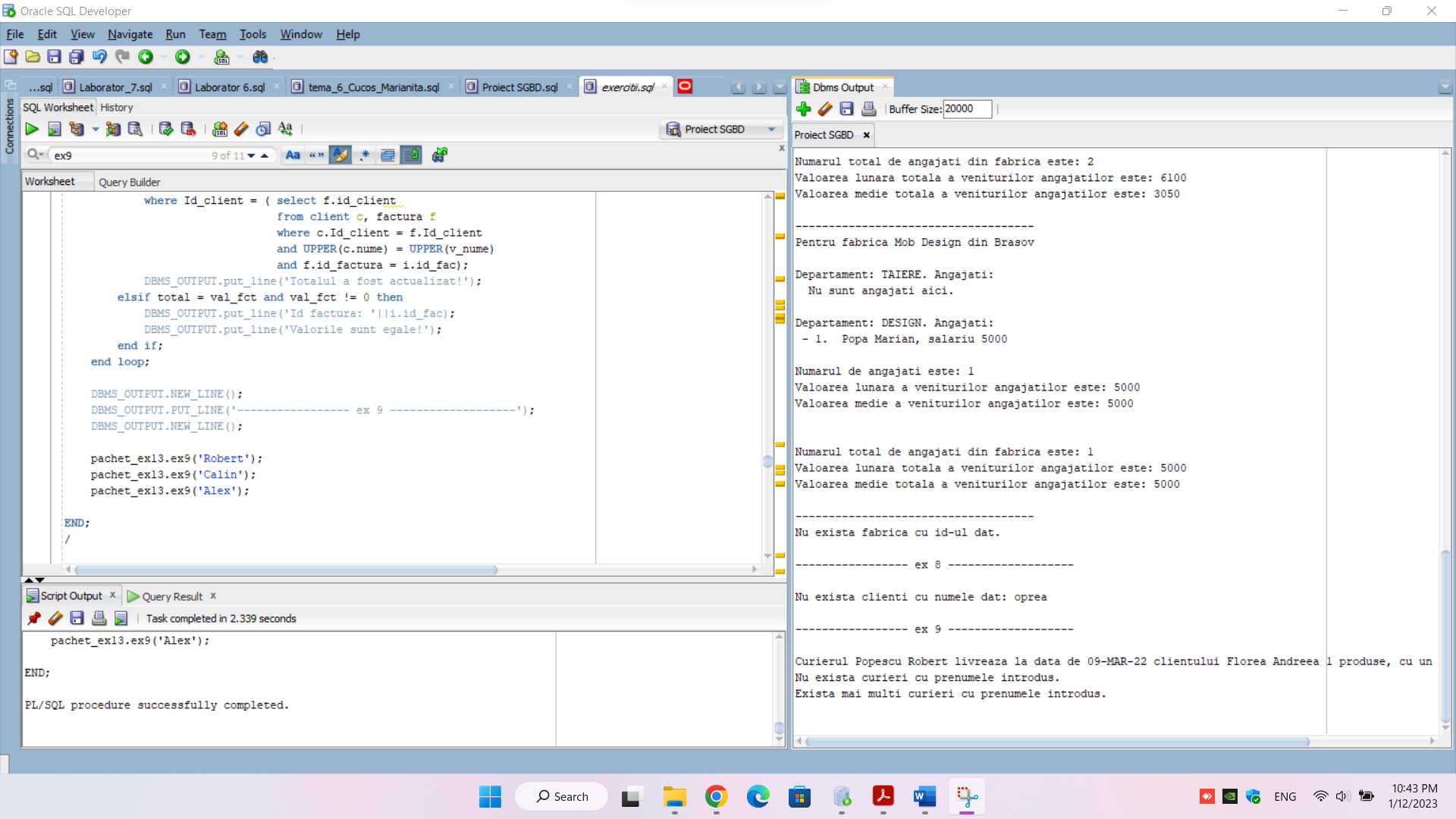
pachet\_ex13.ex9('Calin');

pachet\_ex13.ex9('Alex');

END;

/

****

****

**14. Definiți un pachet care să includă tipuri de date complexe și obiecte necesare unui flux de acțiuni integrate, specifice bazei de date definite (minim 2 tipuri de date, minim 2 funcții, minim 2 proceduri).**

CREATE OR REPLACE PACKAGE pachet\_ex14 IS

TYPE t\_idx IS TABLE OF hala%rowtype INDEX BY PLS\_INTEGER;

hali t\_idx;

cursor departamente\_fab (hala\_id hala.id\_hala%type, denumirefab fabrica.nume%type) return departament%rowtype;

TYPE inf\_angajat IS RECORD

(nume varchar2(100),

cnp angajat.cnp%TYPE,

tel angajat.telefon%TYPE,

exceptie varchar2(100));

-- functie care returneaza datele unui angajat

function info\_angajati

(nume\_ang angajat.nume%type,

prenume\_ang angajat.prenume%TYPE)

return inf\_angajat;

-- functie obtinere cod fabrica dupa numele acesteia

FUNCTION GetFabrica (denumire\_fab fabrica.nume%type)

RETURN fabrica.id\_fabrica%type ;

--- angajatii dintr-un departament ---

procedure ang\_dep (departament\_id departament.id\_departament%TYPE);

-- pentru o fabrica al carei nume este dat, pentru fiecare hala, afisati angajatii care lucreaza in departamentele ce apartin halei curente sau afisati 'nu exista'

-- daca pentru departamentul x nu exista

PROCEDURE fabrica\_ang (denumirefab fabrica.nume%TYPE);

END pachet\_ex14;

/

CREATE OR REPLACE PACKAGE BODY pachet\_ex14 IS

cursor departamente\_fab (hala\_id hala.id\_hala%type, denumirefab fabrica.nume%type) return departament%rowtype

is

SELECT d.Id\_departament id\_dep, d.nume nume, d.id\_hala

FROM departament d, hala h, fabrica f

WHERE f.Id\_fabrica=h.id\_fabrica and h.id\_hala=d.id\_hala

and h.id\_hala = hala\_id

AND UPPER(f.nume) LIKE UPPER(denumirefab);

-- functie obtinere cod fabrica dupa numele acesteia

FUNCTION GetFabrica (denumire\_fab fabrica.nume%type)

RETURN fabrica.id\_fabrica%type

AS

v\_fabrica\_id fabrica.id\_fabrica%type;

nr NUMBER(6);

BEGIN

SELECT COUNT(id\_fabrica)

INTO nr

FROM fabrica f

WHERE upper(f.nume) = upper(denumire\_fab);

IF nr > 1 THEN

DBMS\_OUTPUT.PUT\_LINE('Exista mai multe fabrici cu acest nume.');

RETURN 0;

ELSIF nr = 0 THEN

DBMS\_OUTPUT.PUT\_LINE('Nu exista fabrici cu acest nume.');

RETURN 0;

ELSE

SELECT id\_fabrica

INTO v\_fabrica\_id

FROM fabrica f

WHERE upper(f.nume) = upper(denumire\_fab);

RETURN v\_fabrica\_id;

END IF;

END GetFabrica;

-- functie care returneaza datele unui angajat

function info\_angajati

(nume\_ang angajat.nume%type,

prenume\_ang angajat.prenume%TYPE)

return inf\_angajat

is

rezultat inf\_angajat;

begin

select nume\_ang||' '||prenume\_ang, cnp, telefon, '-'

into rezultat

from angajat

where id\_angajat = (select id\_angajat

from angajat a

where a.nume = nume\_ang

and a.prenume = prenume\_ang);

return rezultat;

exception

when NO\_DATA\_FOUND then

select '-','-','-','no data found'

into rezultat

from dual;

return rezultat;

when TOO\_MANY\_ROWS then

select '-','-','-','too many rows'

into rezultat

from dual;

return rezultat;

end info\_angajati;

--- angajatii dintr-un departament ---

procedure ang\_dep (departament\_id departament.id\_departament%TYPE)

IS

TYPE refcursor IS REF CURSOR;

CURSOR c\_dept IS

SELECT nume,

CURSOR (SELECT nume n, prenume p

FROM angajat a

WHERE a.id\_departament = d.id\_departament)

FROM departament d

WHERE id\_departament = departament\_id;

v\_nume\_dept departament.nume%TYPE;

v\_cursor refcursor;

v\_nume varchar2(100);

v\_prenume varchar2(100);

nr number;

BEGIN

OPEN c\_dept;

LOOP

FETCH c\_dept

INTO v\_nume\_dept, v\_cursor;

EXIT WHEN c\_dept%NOTFOUND;

DBMS\_OUTPUT.PUT\_LINE ('DEPARTAMENT: '||v\_nume\_dept);

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

nr := 0;

LOOP

FETCH v\_cursor

INTO v\_nume, v\_prenume; --angajati din departamentul curent

EXIT WHEN v\_cursor%NOTFOUND;

nr := nr+1;

if info\_angajati(v\_nume, v\_prenume).exceptie = 'no data found'

then dbms\_output.put\_line('exceptie: no data found');

elsif info\_angajati(v\_nume, v\_prenume).exceptie = 'too many rows'

then dbms\_output.put\_line('exceptie: too many rows');

else

dbms\_output.put\_line('-Nume: '||info\_angajati(v\_nume, v\_prenume).nume);

dbms\_output.put\_line('-CNP: '||info\_angajati(v\_nume, v\_prenume).cnp);

dbms\_output.put\_line('-Telefon: '||info\_angajati(v\_nume, v\_prenume).tel);

end if;

dbms\_output.new\_line();

END LOOP;

if nr = 0

then dbms\_output.put\_line('Nu exista angajati in departament.');

DBMS\_OUTPUT.NEW\_LINE;

end if;

END LOOP;

CLOSE c\_dept;

END ang\_dep;

-- pentru o fabrica al carei nume este dat, pentru fiecare hala, afisati angajatii care lucreaza in departamentele ce apartin halei curente sau afisati 'nu exista'

-- daca pentru departamentul x nu exista

PROCEDURE fabrica\_ang

(denumirefab fabrica.nume%TYPE)

AS

numar NUMBER(6);

id\_fab number;

BEGIN

DBMS\_OUTPUT.PUT\_LINE('Nume fabrica: ' || denumirefab);

if GetFabrica(denumirefab) != 0 then

id\_fab := GetFabrica(denumirefab);

SELECT h.Id\_hala, h.suprafata, h.nume, h.id\_fabrica

BULK COLLECT INTO hali

FROM hala h, fabrica f

WHERE h.id\_fabrica = f.Id\_fabrica

AND f.id\_fabrica = id\_fab;

FOR i IN hali.first..hali.last LOOP

DBMS\_OUTPUT.PUT\_LINE('HALA: ' || hali(i).nume);

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

SELECT COUNT(\*)

INTO numar

FROM departament d, hala h, fabrica f

WHERE f.Id\_fabrica=h.id\_fabrica and h.id\_hala=d.id\_hala

and h.id\_hala = hali(i).id\_hala

AND f.id\_fabrica = id\_fab;

if numar = 0

then DBMS\_OUTPUT.PUT\_LINE('Nu exista departamente in hala');

else

FOR j IN departamente\_fab(hali(i).id\_hala, denumirefab) LOOP

ang\_dep(j.id\_departament); -- apelare procedura anterioara

END LOOP;

end if;

END LOOP;

end if;

DBMS\_OUTPUT.NEW\_LINE;

END fabrica\_ang;

END pachet\_ex14;

/

-- apelare

begin

pachet\_ex14.fabrica\_ang('the home');

DBMS\_OUTPUT.NEW\_LINE;

pachet\_ex14.fabrica\_ang('your furniture');

DBMS\_OUTPUT.NEW\_LINE;

pachet\_ex14.fabrica\_ang('mob design');

DBMS\_OUTPUT.NEW\_LINE;

pachet\_ex14.fabrica\_ang('mob expert'); -- too many rows

DBMS\_OUTPUT.NEW\_LINE;

pachet\_ex14.fabrica\_ang('mobila'); -- no data found

end; /

