Proiect SGBD

-MAGAZIN ONLINE-

Realizat de OANCEA ELENA-ANTONIA

GRUPA 242

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

In acest proiect voi crea baza de date a unui magazin online. Magazinul online va contine :

* Produse : - produsele pe care le pot comanda clientii
* Comenzi : - comenzile realizate de clienti
* Retururi : - retururile realizate
* Furnizori : - date despre furnizori
* Clienti : - date despre clinti
* Departamente: - retinem numele departamentelor
* Angajati: – date despre angajati
* Posturi : - posturile pe care le pot ocupa angajatii si salariul minim pentru fiecare
* Istoric : - retinem evenimentele ALTER, CREATE, DROP

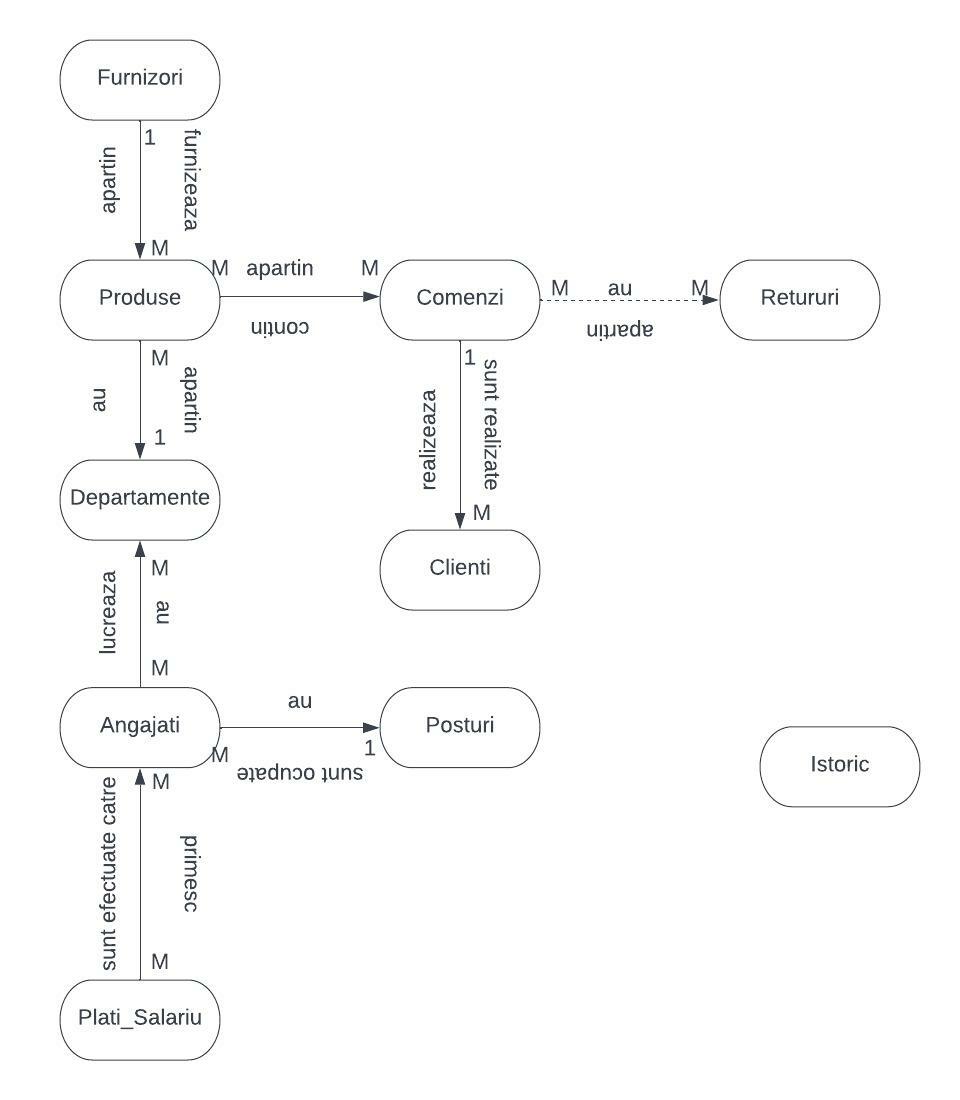
Fiecare produs apartine unui departament.

Comenzile sunt realizate de clienti contin cel putin un produs.

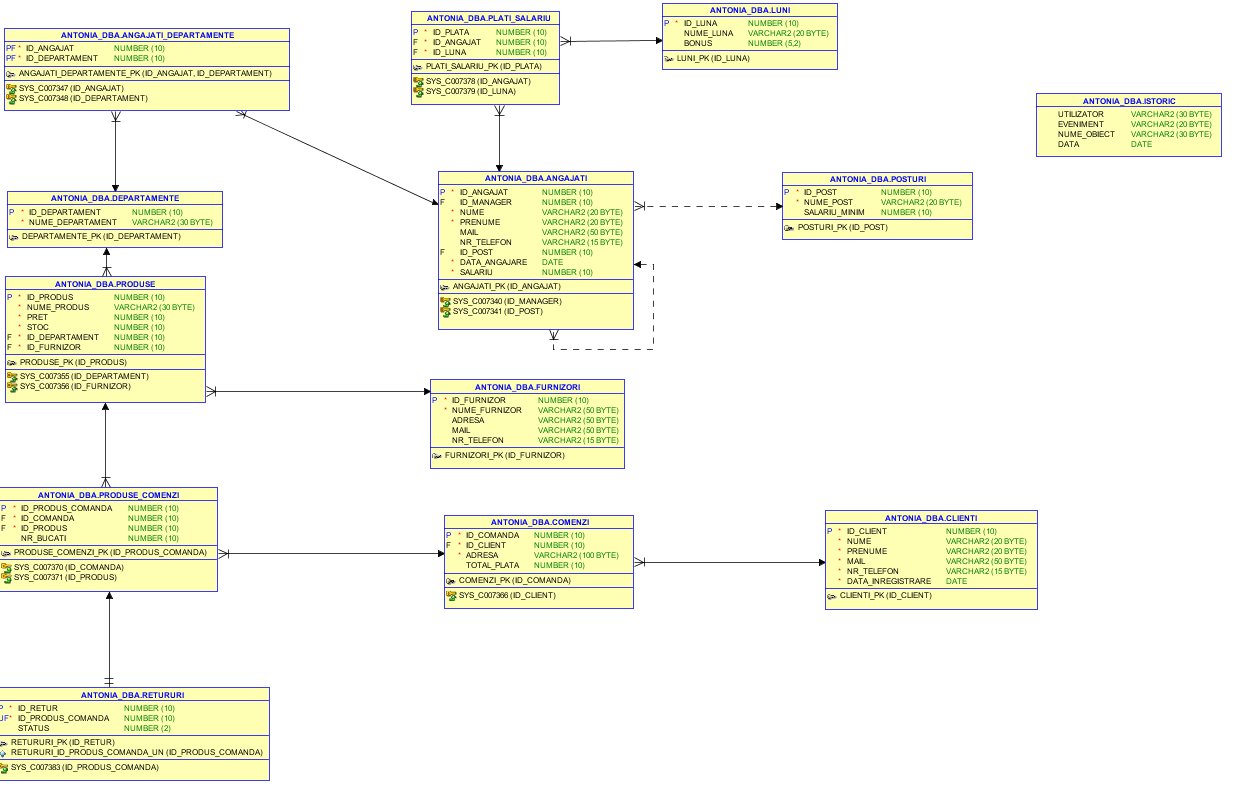
Retururile sunt realizate de clienti asupra unei comenzi existente.

Angajati lucreaza la cel putin un departament.

1. Realizați diagrama entitate-relație (ERD).



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



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

+

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

CREATE SEQUENCE secv

START WITH 100

INCREMENT BY 1

ISTORIC

CREATE TABLE istoric(

utilizator VARCHAR2(30),

eveniment VARCHAR2(20),

nume\_obiect VARCHAR2(30),

data DATE

);

--ADAUGAM EVENIMENTELE DE TIP CREATE, DROP SI ALTER IN TABELUL ISTORIC

CREATE OR REPLACE TRIGGER istoric\_LDD

AFTER CREATE OR DROP OR ALTER ON SCHEMA

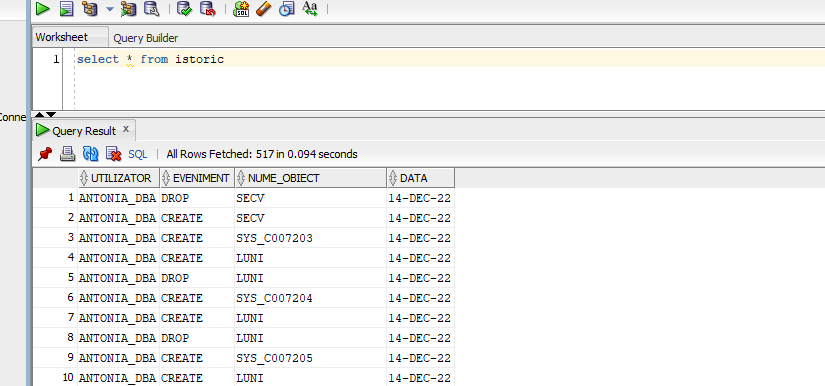
BEGIN

INSERT INTO istoric

VALUES (SYS.LOGIN\_USER, SYS.SYSEVENT, SYS.DICTIONARY\_OBJ\_NAME, SYSDATE);

END;

/



LUNI

create table luni(

id\_luna number(10) NOT NULL ,

nume\_luna varchar2(20)NOT NULL,

bonus numeric(5,2) DEFAULT 0,

CONSTRAINT luni\_pk PRIMARY KEY (id\_luna)

);

insert into luni (id\_luna,nume\_luna,bonus) values (1, 'Ianuarie', 0.1);

insert into luni (id\_luna,nume\_luna) values (2, 'Februarie');

insert into luni (id\_luna,nume\_luna,bonus) values (3, 'Martie', 0.05);

insert into luni (id\_luna,nume\_luna,bonus) values (4, 'Aprilie', 0.02);

insert into luni (id\_luna,nume\_luna) values (5, 'Mai');

insert into luni (id\_luna,nume\_luna,bonus) values (6, 'Iunie', 0.1);

insert into luni (id\_luna,nume\_luna) values (7, 'Iulie');

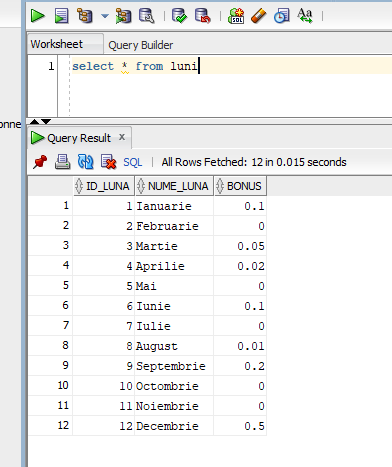
insert into luni (id\_luna,nume\_luna,bonus) values (8, 'August', 0.01);

insert into luni (id\_luna,nume\_luna,bonus) values (9, 'Septembrie', 0.2);

insert into luni (id\_luna,nume\_luna) values (10, 'Octombrie');

insert into luni (id\_luna,nume\_luna) values (11, 'Noiembrie');

insert into luni (id\_luna,nume\_luna,bonus) values (12, 'Decembrie', 0.5);



POSTURI

create table furnizori(

id\_furnizor number(10) NOT NULL,

nume\_furnizor varchar2(50) NOT NULL,

adresa varchar2(50),

mail varchar2(50),

nr\_telefon varchar2(15),

CONSTRAINT furnizori\_pk PRIMARY KEY (id\_furnizor)

);

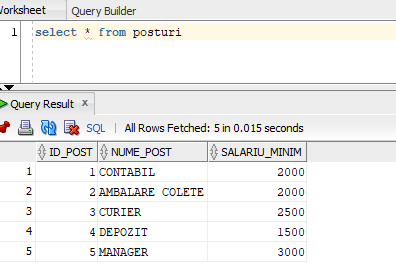
insert into posturi (id\_post, nume\_post,salariu\_minim) values (1,'CONTABIL',2000);

insert into posturi (id\_post, nume\_post,salariu\_minim) values (2,'AMBALARE COLETE',2000);

insert into posturi (id\_post, nume\_post,salariu\_minim) values (3,'CURIER',2500);

insert into posturi (id\_post, nume\_post) values (4,'DEPOZIT');

insert into posturi (id\_post, nume\_post,salariu\_minim) values (5,'MANAGER',3000);



FURNIZORI

create table furnizori(

id\_furnizor number(10) NOT NULL,

nume\_furnizor varchar2(50) NOT NULL,

adresa varchar2(50),

mail varchar2(50),

nr\_telefon varchar2(15),

CONSTRAINT furnizori\_pk PRIMARY KEY (id\_furnizor)

);

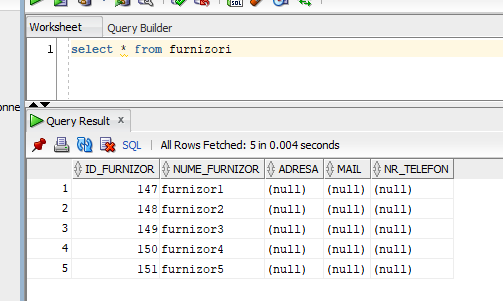
insert into furnizori (id\_furnizor,nume\_furnizor) values (secv.nextval,'furnizor1');

insert into furnizori (id\_furnizor,nume\_furnizor) values (secv.nextval,'furnizor2');

insert into furnizori (id\_furnizor,nume\_furnizor) values (secv.nextval,'furnizor3');

insert into furnizori (id\_furnizor,nume\_furnizor) values (secv.nextval,'furnizor4');

insert into furnizori (id\_furnizor,nume\_furnizor) values (secv.nextval,'furnizor5');



ANGAJATI

create table angajati(

id\_angajat number(10) NOT NULL,

id\_manager number(10),

nume varchar2(20) NOT NULL,

prenume varchar2(20) NOT NULL,

mail varchar2(50),

nr\_telefon varchar2(15),

id\_post number(10),

data\_angajare date NOT NULL,

salariu number(10) NOT NULL,

CONSTRAINT angajati\_pk PRIMARY KEY (id\_angajat),

CONSTRAINT angajati\_fk1 FOREIGN KEY (id\_manager) REFERENCES angajati(id\_angajat),

CONSTRAINT angajati\_fk2 FOREIGN KEY (id\_post) REFERENCES posturi(id\_post)

);

insert into angajati (id\_angajat, id\_manager,nume,prenume,id\_post,data\_angajare,salariu) values (secv.nextval,null,'Popa','Daniel',3,TO\_DATE('2015-07-21','YYYY-MM-DD'),2700);

insert into angajati (id\_angajat, id\_manager,nume,prenume,id\_post,data\_angajare,salariu) values (secv.nextval,null,'Andrei','Marian',1,TO\_DATE('2017-06-02','YYYY-MM-DD'),2100);

insert into angajati (id\_angajat, id\_manager,nume,prenume,id\_post,data\_angajare,salariu) values (secv.nextval,null,'Popescu','Carina',2,TO\_DATE('2018-01-09','YYYY-MM-DD'),2500);

insert into angajati (id\_angajat, id\_manager,nume,prenume,id\_post,data\_angajare,salariu) values (secv.nextval,null,'Ionescu','Sorin',3,TO\_DATE('2019-02-14','YYYY-MM-DD'),2700);

insert into angajati (id\_angajat, id\_manager,nume,prenume,id\_post,data\_angajare,salariu) values (secv.nextval,286,'Popescu','Diana',4,TO\_DATE('2020-03-27','YYYY-MM-DD'),1600);

insert into angajati (id\_angajat, id\_manager,nume,prenume,id\_post,data\_angajare,salariu) values (secv.nextval,287,'Popa','Ion',3,TO\_DATE('2021-07-24','YYYY-MM-DD'),2700);

insert into angajati (id\_angajat, id\_manager,nume,prenume,id\_post,data\_angajare,salariu) values (secv.nextval,288,'David','Marius',1,TO\_DATE('2022-04-19','YYYY-MM-DD'),2700);

insert into angajati (id\_angajat, id\_manager,nume,prenume,id\_post,data\_angajare,salariu) values (secv.nextval,289,'Popovici','Maria',2,TO\_DATE('2016-04-07','YYYY-MM-DD'),2200);

insert into angajati (id\_angajat, id\_manager,nume,prenume,id\_post,data\_angajare,salariu) values (secv.nextval,null,'Simionescu','Sorin',3,TO\_DATE('2017-05-05','YYYY-MM-DD'),2600);

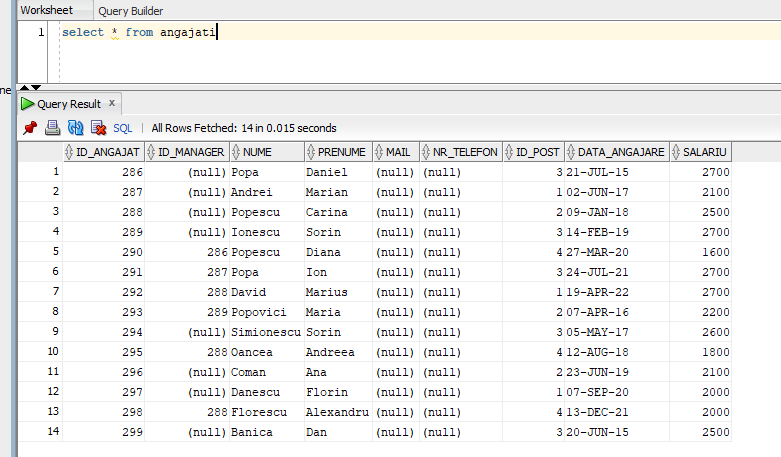
insert into angajati (id\_angajat, id\_manager,nume,prenume,id\_post,data\_angajare,salariu) values (secv.nextval,288,'Oancea','Andreea',4,TO\_DATE('2018-08-12','YYYY-MM-DD'),1800);

insert into angajati (id\_angajat, id\_manager,nume,prenume,id\_post,data\_angajare,salariu) values (secv.nextval,null,'Coman','Ana',2,TO\_DATE('2019-06-23','YYYY-MM-DD'),2100);

insert into angajati (id\_angajat, id\_manager,nume,prenume,id\_post,data\_angajare,salariu) values (secv.nextval,null,'Danescu','Florin',1,TO\_DATE('2020-09-07','YYYY-MM-DD'),2000);

insert into angajati (id\_angajat, id\_manager,nume,prenume,id\_post,data\_angajare,salariu) values (secv.nextval,288,'Florescu','Alexandru',4,TO\_DATE('2021-12-13','YYYY-MM-DD'),2000);

insert into angajati (id\_angajat, id\_manager,nume,prenume,id\_post,data\_angajare,salariu) values (secv.nextval,null,'Banica','Dan',3,TO\_DATE('2015-06-20','YYYY-MM-DD'),2500);



DEPARTAMENTE

create table departamente(

id\_departament number(10) NOT NULL,

nume\_departament varchar2(30) NOT NULL,

CONSTRAINT departamente\_pk PRIMARY KEY (id\_departament)

);

insert into departamente (id\_departament, nume\_departament) values (secv.nextval,'Curatenie');

insert into departamente (id\_departament, nume\_departament) values (secv.nextval,'Electrocasnice');

insert into departamente (id\_departament, nume\_departament) values (secv.nextval,'Ingrijire');

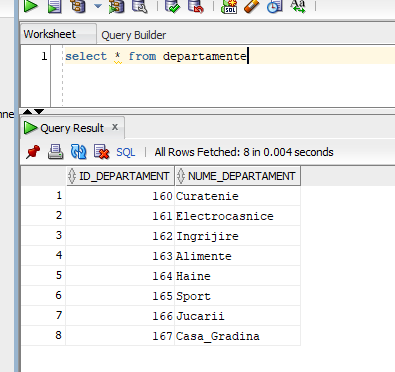
insert into departamente (id\_departament, nume\_departament) values (secv.nextval,'Alimente');

insert into departamente (id\_departament, nume\_departament) values (secv.nextval,'Haine');

insert into departamente (id\_departament, nume\_departament) values (secv.nextval,'Sport');

insert into departamente (id\_departament, nume\_departament) values (secv.nextval,'Jucarii');

insert into departamente (id\_departament, nume\_departament) values (secv.nextval,'Casa\_Gradina');



ANGAJATI\_DEPARTAMENTE

create table angajati\_departamente(

id\_angajat number(10) not null,

id\_departament number(10) not null,

CONSTRAINT ang\_dep\_pk PRIMARY KEY(id\_angajat, id\_departament),

CONSTRAINT ang\_dep\_fk1 FOREIGN KEY (id\_angajat) REFERENCES angajati(id\_angajat),

CONSTRAINT ang\_dep\_fk2 FOREIGN KEY (id\_departament) REFERENCES departamente(id\_departament)

);

insert into angajati\_departamente (id\_angajat,id\_departament) values (286,160);

insert into angajati\_departamente (id\_angajat,id\_departament) values (287,161);

insert into angajati\_departamente (id\_angajat,id\_departament) values (288,162);

insert into angajati\_departamente (id\_angajat,id\_departament) values (289,163);

insert into angajati\_departamente (id\_angajat,id\_departament) values (290,164);

insert into angajati\_departamente (id\_angajat,id\_departament) values (291,165);

insert into angajati\_departamente (id\_angajat,id\_departament) values (292,166);

insert into angajati\_departamente (id\_angajat,id\_departament) values (294,167);

insert into angajati\_departamente (id\_angajat,id\_departament) values (296,160);

insert into angajati\_departamente (id\_angajat,id\_departament) values (297,161);

insert into angajati\_departamente (id\_angajat,id\_departament) values (286,162);

insert into angajati\_departamente (id\_angajat,id\_departament) values (298,163);

insert into angajati\_departamente (id\_angajat,id\_departament) values (299,163);

insert into angajati\_departamente (id\_angajat,id\_departament) values (286,164);

insert into angajati\_departamente (id\_angajat,id\_departament) values (287,164);

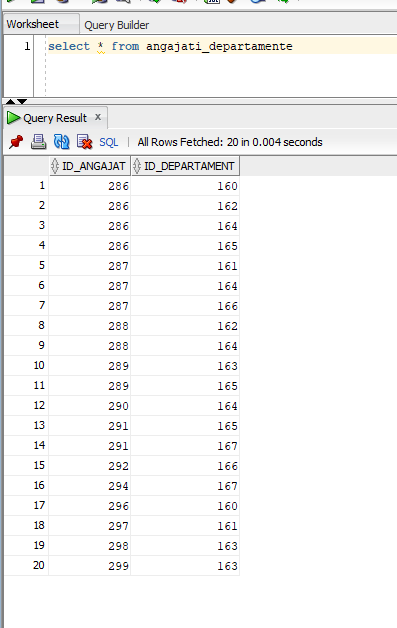
insert into angajati\_departamente (id\_angajat,id\_departament) values (288,164);

insert into angajati\_departamente (id\_angajat,id\_departament) values (289,165);

insert into angajati\_departamente (id\_angajat,id\_departament) values (286,165);

insert into angajati\_departamente (id\_angajat,id\_departament) values (287,166);

insert into angajati\_departamente (id\_angajat,id\_departament) values (291,167);



PRODUSE

create table produse(

id\_produs number(10) NOT NULL,

nume\_produs varchar2(30) NOT NULL,

pret number(10) NOT NULL,

stoc number(10) NOT NULL,

id\_departament NOT NULL,

id\_furnizor NOT NULL,

CONSTRAINT produse\_pk PRIMARY KEY(id\_produs),

CONSTRAINT produse\_fk1 FOREIGN KEY (id\_furnizor) REFERENCES furnizori(id\_furnizor),

CONSTRAINT produse\_fk2 FOREIGN KEY (id\_departament) REFERENCES departamente(id\_departament)

);

insert into produse (id\_produs,nume\_produs,pret,stoc,id\_departament,id\_furnizor) values (secv.nextval,'Scaun',100,4,167,147);

insert into produse (id\_produs,nume\_produs,pret,stoc,id\_departament,id\_furnizor) values (secv.nextval,'Masa',200,2,167,151);

insert into produse (id\_produs,nume\_produs,pret,stoc,id\_departament,id\_furnizor) values (secv.nextval,'Detergent vase',10,5,160,148);

insert into produse (id\_produs,nume\_produs,pret,stoc,id\_departament,id\_furnizor) values (secv.nextval,'Apa',8,20,163,149);

insert into produse (id\_produs,nume\_produs,pret,stoc,id\_departament,id\_furnizor) values (secv.nextval,'Zahar',5,10,163,150);

insert into produse (id\_produs,nume\_produs,pret,stoc,id\_departament,id\_furnizor) values (secv.nextval,'Faina',5,12,163,147);

insert into produse (id\_produs,nume\_produs,pret,stoc,id\_departament,id\_furnizor) values (secv.nextval,'Minge',50,7,165,148);

insert into produse (id\_produs,nume\_produs,pret,stoc,id\_departament,id\_furnizor) values (secv.nextval,'Paleta de tenis',80,3,165,149);

insert into produse (id\_produs,nume\_produs,pret,stoc,id\_departament,id\_furnizor) values (secv.nextval,'Fileu',40,6,165,150);

insert into produse (id\_produs,nume\_produs,pret,stoc,id\_departament,id\_furnizor) values (secv.nextval,'Pantaloni scurti',100,2,164,151);

insert into produse (id\_produs,nume\_produs,pret,stoc,id\_departament,id\_furnizor) values (secv.nextval,'Pantaloni',120,4,164,147);

insert into produse (id\_produs,nume\_produs,pret,stoc,id\_departament,id\_furnizor) values (secv.nextval,'Tricou',70,4,164,148);

insert into produse (id\_produs,nume\_produs,pret,stoc,id\_departament,id\_furnizor) values (secv.nextval,'Hanorac',90,4,164,149);

insert into produse (id\_produs,nume\_produs,pret,stoc,id\_departament,id\_furnizor) values (secv.nextval,'Robot de bucatarie',300,3,161,150);

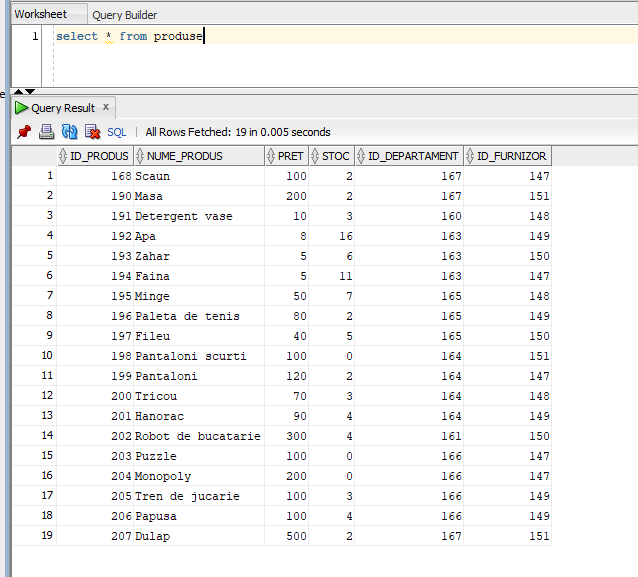
insert into produse (id\_produs,nume\_produs,pret,stoc,id\_departament,id\_furnizor) values (secv.nextval,'Puzzle',100,2,166,147);

insert into produse (id\_produs,nume\_produs,pret,stoc,id\_departament,id\_furnizor) values (secv.nextval,'Monopoly',200,1,166,147);

insert into produse (id\_produs,nume\_produs,pret,stoc,id\_departament,id\_furnizor) values (secv.nextval,'Tren de jucarie',100,3,166,149);

insert into produse (id\_produs,nume\_produs,pret,stoc,id\_departament,id\_furnizor) values (secv.nextval,'Papusa',100,4,166,149);

insert into produse (id\_produs,nume\_produs,pret,stoc,id\_departament,id\_furnizor) values (secv.nextval,'Dulap',500,2,167,151);



CLIENTI

create table clienti(

id\_client number(10) NOT NULL,

nume varchar2(20) NOT NULL,

prenume varchar2(20) NOT NULL,

mail varchar2(50) NOT NULL,

nr\_telefon varchar2(15) NOT NULL,

data\_inregistrare date NOT NULL,

CONSTRAINT clienti\_pk PRIMARY KEY(id\_client)

);

insert into clienti (id\_client,nume,prenume,mail,nr\_telefon,data\_inregistrare) values (secv.nextval,'Popescu','Ion','popescu.ion@yahoo.com','0722334455',TO\_DATE('2019-06-21','YYYY-MM-DD'));

insert into clienti (id\_client,nume,prenume,mail,nr\_telefon,data\_inregistrare) values (secv.nextval,'Popescu','Andrei','popescu.andrei@yahoo.com','0722334456',TO\_DATE('2020-07-01','YYYY-MM-DD'));

insert into clienti (id\_client,nume,prenume,mail,nr\_telefon,data\_inregistrare) values (secv.nextval,'Ionescu','Mihai','ionescu.mihai@yahoo.com','0712334455',TO\_DATE('2022-05-01','YYYY-MM-DD'));

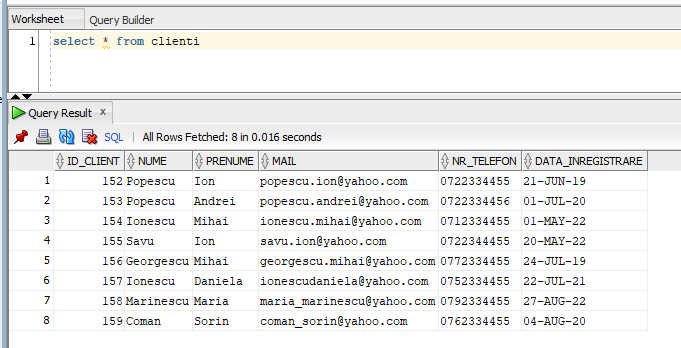
insert into clienti (id\_client,nume,prenume,mail,nr\_telefon,data\_inregistrare) values (secv.nextval,'Savu','Ion','savu.ion@yahoo.com','0722344455',TO\_DATE('2022-05-20','YYYY-MM-DD'));

insert into clienti (id\_client,nume,prenume,mail,nr\_telefon,data\_inregistrare) values (secv.nextval,'Georgescu','Mihai','georgescu.mihai@yahoo.com','0772334455',TO\_DATE('2019-07-24','YYYY-MM-DD'));

insert into clienti (id\_client,nume,prenume,mail,nr\_telefon,data\_inregistrare) values (secv.nextval,'Ionescu','Daniela','ionescudaniela@yahoo.com','0752334455',TO\_DATE('2021-07-22','YYYY-MM-DD'));

insert into clienti (id\_client,nume,prenume,mail,nr\_telefon,data\_inregistrare) values (secv.nextval,'Marinescu','Maria','maria\_marinescu@yahoo.com','0792334455',TO\_DATE('2022-08-27','YYYY-MM-DD'));

insert into clienti (id\_client,nume,prenume,mail,nr\_telefon,data\_inregistrare) values (secv.nextval,'Coman','Sorin','coman\_sorin@yahoo.com','0762334455',TO\_DATE('2020-08-04','YYYY-MM-DD'));



COMENZI

create table comenzi(

id\_comanda number(10) NOT NULL,

id\_client number(10) NOT NULL,

adresa varchar2(100) NOT NULL,

total\_plata number(10) DEFAULT 0,

CONSTRAINT comenzi\_pk PRIMARY KEY(id\_comanda),

CONSTRAINT comenzi\_fk1 FOREIGN KEY (id\_client) REFERENCES clienti(id\_client)

);

insert into comenzi (id\_comanda,id\_client,adresa) values (secv.nextval,152,'Bucuresti');

insert into comenzi (id\_comanda,id\_client,adresa) values (secv.nextval,152,'Bucuresti');

insert into comenzi (id\_comanda,id\_client,adresa) values (secv.nextval,152,'Brasov');

insert into comenzi (id\_comanda,id\_client,adresa) values (secv.nextval,152,'Pitesti');

insert into comenzi (id\_comanda,id\_client,adresa) values (secv.nextval,154,'Bucuresti');

insert into comenzi (id\_comanda,id\_client,adresa) values (secv.nextval,154,'Sibiu');

insert into comenzi (id\_comanda,id\_client,adresa) values (secv.nextval,153,'Sibiu');

insert into comenzi (id\_comanda,id\_client,adresa) values (secv.nextval,156,'Bucuresti');

insert into comenzi (id\_comanda,id\_client,adresa) values (secv.nextval,156,'Bucuresti');

insert into comenzi (id\_comanda,id\_client,adresa) values (secv.nextval,153,'Bucuresti');

insert into comenzi (id\_comanda,id\_client,adresa) values (secv.nextval,154,'Bucuresti');

insert into comenzi (id\_comanda,id\_client,adresa) values (secv.nextval,157,'Bucuresti');

insert into comenzi (id\_comanda,id\_client,adresa) values (secv.nextval,159,'Brasov');

insert into comenzi (id\_comanda,id\_client,adresa) values (secv.nextval,159,'Campulung');

insert into comenzi (id\_comanda,id\_client,adresa) values (secv.nextval,158,'Targoviste');

insert into comenzi (id\_comanda,id\_client,adresa) values (secv.nextval,153,'Brasov');

insert into comenzi (id\_comanda,id\_client,adresa) values (secv.nextval,155,'Cluj');

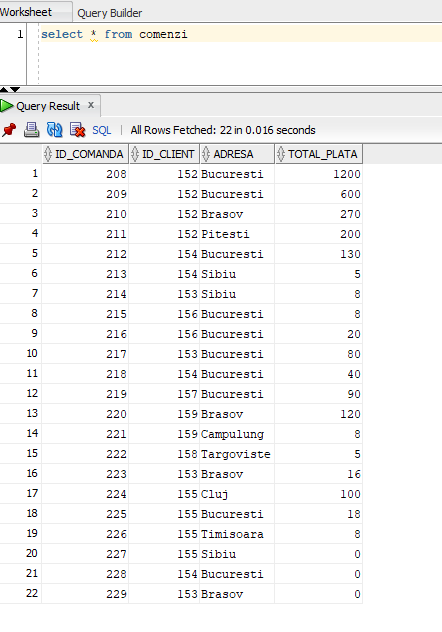
insert into comenzi (id\_comanda,id\_client,adresa) values (secv.nextval,155,'Bucuresti');

insert into comenzi (id\_comanda,id\_client,adresa) values (secv.nextval,155,'Timisoara');

insert into comenzi (id\_comanda,id\_client,adresa) values (secv.nextval,155,'Sibiu');

insert into comenzi (id\_comanda,id\_client,adresa) values (secv.nextval,154,'Bucuresti');

insert into comenzi (id\_comanda,id\_client,adresa) values (secv.nextval,153,'Brasov');



PRODUSE\_COMENZI

create table produse\_comenzi(

id\_produs\_comanda number(10) NOT NULL,

id\_comanda number(10) NOT NULL,

id\_produs number(10) NOT NULL,

nr\_bucati number(10) DEFAULT 1,

CONSTRAINT prod\_com\_pk PRIMARY KEY(id\_produs\_comanda),

CONSTRAINT prod\_com\_fk1 FOREIGN KEY (id\_comanda) REFERENCES comenzi(id\_comanda),

CONSTRAINT prod\_com\_fk2 FOREIGN KEY (id\_produs) REFERENCES produse(id\_produs)

);

insert into produse\_comenzi (id\_produs\_comanda,id\_comanda,id\_produs,nr\_bucati) values (secv.nextval,209,202,1);

insert into produse\_comenzi (id\_produs\_comanda,id\_comanda,id\_produs,nr\_bucati) values (secv.nextval,209,203,2);

insert into produse\_comenzi (id\_produs\_comanda,id\_comanda,id\_produs,nr\_bucati) values (secv.nextval,208,204,1);

insert into produse\_comenzi (id\_produs\_comanda,id\_comanda,id\_produs,nr\_bucati) values (secv.nextval,209,205,1);

insert into produse\_comenzi (id\_produs\_comanda,id\_comanda,id\_produs,nr\_bucati) values (secv.nextval,210,206,1);

insert into produse\_comenzi (id\_produs\_comanda,id\_comanda,id\_produs,nr\_bucati) values (secv.nextval,211,168,1);

insert into produse\_comenzi (id\_produs\_comanda,id\_comanda,id\_produs,nr\_bucati) values (secv.nextval,212,191,1);

insert into produse\_comenzi (id\_produs\_comanda,id\_comanda,id\_produs,nr\_bucati) values (secv.nextval,213,194,1);

insert into produse\_comenzi (id\_produs\_comanda,id\_comanda,id\_produs,nr\_bucati) values (secv.nextval,214,192,1);

insert into produse\_comenzi (id\_produs\_comanda,id\_comanda,id\_produs,nr\_bucati) values (secv.nextval,215,192,1);

insert into produse\_comenzi (id\_produs\_comanda,id\_comanda,id\_produs,nr\_bucati) values (secv.nextval,216,193,4);

insert into produse\_comenzi (id\_produs\_comanda,id\_comanda,id\_produs,nr\_bucati) values (secv.nextval,217,196,1);

insert into produse\_comenzi (id\_produs\_comanda,id\_comanda,id\_produs,nr\_bucati) values (secv.nextval,218,197,1);

insert into produse\_comenzi (id\_produs\_comanda,id\_comanda,id\_produs,nr\_bucati) values (secv.nextval,219,201,1);

insert into produse\_comenzi (id\_produs\_comanda,id\_comanda,id\_produs,nr\_bucati) values (secv.nextval,210,200,1);

insert into produse\_comenzi (id\_produs\_comanda,id\_comanda,id\_produs,nr\_bucati) values (secv.nextval,210,198,1);

insert into produse\_comenzi (id\_produs\_comanda,id\_comanda,id\_produs,nr\_bucati) values (secv.nextval,211,198,1);

insert into produse\_comenzi (id\_produs\_comanda,id\_comanda,id\_produs,nr\_bucati) values (secv.nextval,212,199,1);

insert into produse\_comenzi (id\_produs\_comanda,id\_comanda,id\_produs,nr\_bucati) values (secv.nextval,220,199,1);

insert into produse\_comenzi (id\_produs\_comanda,id\_comanda,id\_produs,nr\_bucati) values (secv.nextval,221,192,1);

insert into produse\_comenzi (id\_produs\_comanda,id\_comanda,id\_produs,nr\_bucati) values (secv.nextval,222,193,1);

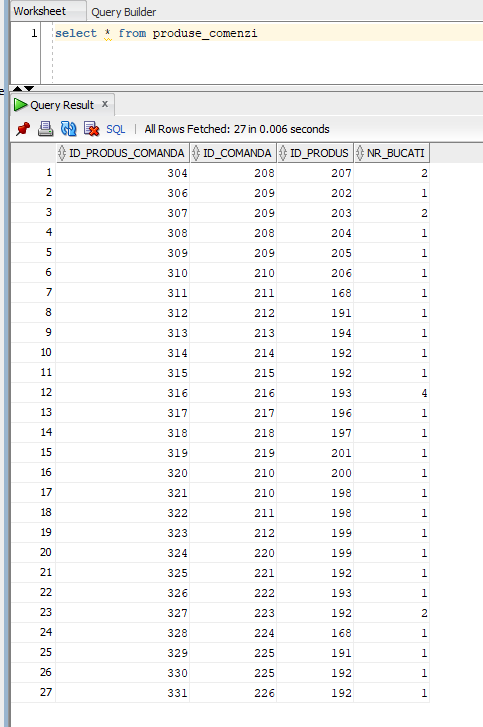
insert into produse\_comenzi (id\_produs\_comanda,id\_comanda,id\_produs,nr\_bucati) values (secv.nextval,223,192,2);

insert into produse\_comenzi (id\_produs\_comanda,id\_comanda,id\_produs,nr\_bucati) values (secv.nextval,224,168,1);

insert into produse\_comenzi (id\_produs\_comanda,id\_comanda,id\_produs,nr\_bucati) values (secv.nextval,225,191,1);

insert into produse\_comenzi (id\_produs\_comanda,id\_comanda,id\_produs,nr\_bucati) values (secv.nextval,225,192,1);

insert into produse\_comenzi (id\_produs\_comanda,id\_comanda,id\_produs,nr\_bucati) values (secv.nextval,226,192,1);



RETURURI

create table retururi(

id\_retur number(10) NOT NULL,

id\_produs\_comanda UNIQUE NOT NULL,

status number(2) DEFAULT 1,

CONSTRAINT retururi\_pk PRIMARY KEY(id\_retur),

CONSTRAINT retururi\_fk1 FOREIGN KEY (id\_produs\_comanda) REFERENCES produse\_comenzi(id\_produs\_comanda)

);

insert into retururi (id\_retur,id\_produs\_comanda,status) VALUES (secv.nextval,306,1);

insert into retururi (id\_retur,id\_produs\_comanda,status) VALUES (secv.nextval,310,1);

insert into retururi (id\_retur,id\_produs\_comanda,status) VALUES (secv.nextval,309,1);

insert into retururi (id\_retur,id\_produs\_comanda,status) VALUES (secv.nextval,319,1);

insert into retururi (id\_retur,id\_produs\_comanda,status) VALUES (secv.nextval,320,0);

insert into retururi (id\_retur,id\_produs\_comanda,status) VALUES (secv.nextval,325,1);

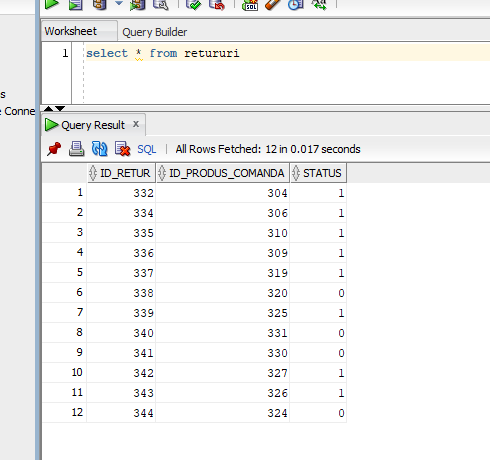
insert into retururi (id\_retur,id\_produs\_comanda,status) VALUES (secv.nextval,331,0);

insert into retururi (id\_retur,id\_produs\_comanda,status) VALUES (secv.nextval,330,0);

insert into retururi (id\_retur,id\_produs\_comanda,status) VALUES (secv.nextval,327,1);

insert into retururi (id\_retur,id\_produs\_comanda,status) VALUES (secv.nextval,326,1);

insert into retururi (id\_retur,id\_produs\_comanda,status) VALUES (secv.nextval,324,0);



PLATI\_SALARIU

create table plati\_salariu(

id\_plata number(10) NOT NULL,

id\_angajat number(10) NOT NULL,

id\_luna number(10) NOT NULL,

CONSTRAINT plati\_sal\_pk PRIMARY KEY(id\_plata),

CONSTRAINT plati\_sal\_fk1 FOREIGN KEY (id\_angajat) REFERENCES angajati(id\_angajat),

CONSTRAINT plati\_sal\_fk2 FOREIGN KEY (id\_luna) REFERENCES luni(id\_luna)

);

insert into plati\_salariu (id\_plata,id\_angajat,id\_luna) values (secv.nextval,286,1);

insert into plati\_salariu (id\_plata,id\_angajat,id\_luna) values (secv.nextval,287,2);

insert into plati\_salariu (id\_plata,id\_angajat,id\_luna) values (secv.nextval,288,3);

insert into plati\_salariu (id\_plata,id\_angajat,id\_luna) values (secv.nextval,289,4);

insert into plati\_salariu (id\_plata,id\_angajat,id\_luna) values (secv.nextval,290,5);

insert into plati\_salariu (id\_plata,id\_angajat,id\_luna) values (secv.nextval,291,6);

insert into plati\_salariu (id\_plata,id\_angajat,id\_luna) values (secv.nextval,292,7);

insert into plati\_salariu (id\_plata,id\_angajat,id\_luna) values (secv.nextval,293,8);

insert into plati\_salariu (id\_plata,id\_angajat,id\_luna) values (secv.nextval,294,9);

insert into plati\_salariu (id\_plata,id\_angajat,id\_luna) values (secv.nextval,295,10);

insert into plati\_salariu (id\_plata,id\_angajat,id\_luna) values (secv.nextval,296,11);

insert into plati\_salariu (id\_plata,id\_angajat,id\_luna) values (secv.nextval,297,12);

insert into plati\_salariu (id\_plata,id\_angajat,id\_luna) values (secv.nextval,298,1);

insert into plati\_salariu (id\_plata,id\_angajat,id\_luna) values (secv.nextval,299,2);

insert into plati\_salariu (id\_plata,id\_angajat,id\_luna) values (secv.nextval,286,3);

insert into plati\_salariu (id\_plata,id\_angajat,id\_luna) values (secv.nextval,287,4);

insert into plati\_salariu (id\_plata,id\_angajat,id\_luna) values (secv.nextval,288,5);

insert into plati\_salariu (id\_plata,id\_angajat,id\_luna) values (secv.nextval,289,6);

insert into plati\_salariu (id\_plata,id\_angajat,id\_luna) values (secv.nextval,286,7);

insert into plati\_salariu (id\_plata,id\_angajat,id\_luna) values (secv.nextval,287,8);

insert into plati\_salariu (id\_plata,id\_angajat,id\_luna) values (secv.nextval,288,9);

insert into plati\_salariu (id\_plata,id\_angajat,id\_luna) values (secv.nextval,289,10);

insert into plati\_salariu (id\_plata,id\_angajat,id\_luna) values (secv.nextval,290,11);

insert into plati\_salariu (id\_plata,id\_angajat,id\_luna) values (secv.nextval,291,12);

insert into plati\_salariu (id\_plata,id\_angajat,id\_luna) values (secv.nextval,292,1);

insert into plati\_salariu (id\_plata,id\_angajat,id\_luna) values (secv.nextval,293,2);

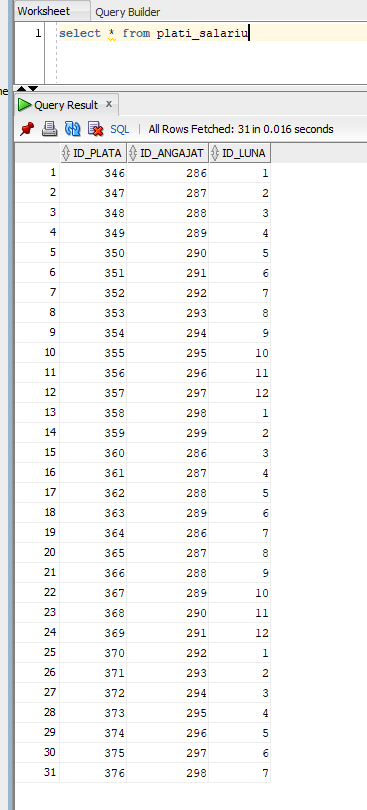
insert into plati\_salariu (id\_plata,id\_angajat,id\_luna) values (secv.nextval,294,3);

insert into plati\_salariu (id\_plata,id\_angajat,id\_luna) values (secv.nextval,295,4);

insert into plati\_salariu (id\_plata,id\_angajat,id\_luna) values (secv.nextval,296,5);

insert into plati\_salariu (id\_plata,id\_angajat,id\_luna) values (secv.nextval,297,6);

insert into plati\_salariu (id\_plata,id\_angajat,id\_luna) values (secv.nextval,298,7);



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.

--Pentru un manager dat ca parametru afisati pentru angajatii pe care ii are in subordine salariile pe care

--le primesc acestia in fiecare luna a anului. Daca id-ul nu corespunde unui manager afisati un mesaj semnificativ.

CREATE OR REPLACE PROCEDURE ANG\_SAL (man angajati.id\_manager%type) AS

TYPE vector IS VARRAY(13) OF NUMBER;

ang vector := vector();

TYPE tablou\_indexat IS TABLE OF luni%rowtype

INDEX BY BINARY\_INTEGER;

t tablou\_indexat;

nume angajati.nume%type;

prenume angajati.prenume%type;

salariu angajati.salariu%type;

salariu\_luna angajati.salariu%type;

ok number:=0;

BEGIN

SELECT COUNT(\*) INTO ok FROM angajati WHERE id\_manager = man;

IF ok = 0 THEN

DBMS\_OUTPUT.PUT\_LINE('ID-ul dat nu corespunde unui manager!');

ELSE

SELECT id\_angajat BULK COLLECT INTO ang FROM angajati WHERE id\_manager=man;

SELECT \* BULK COLLECT INTO t FROM luni;

FOR j IN ang.first..ang.last LOOP

SELECT nume,prenume,trunc(salariu) to\_int INTO nume,prenume,salariu FROM angajati WHERE id\_angajat = ang(j);

FOR i IN 1..12 LOOP

salariu\_luna := salariu+salariu\*t(i).bonus;

DBMS\_OUTPUT.PUT\_LINE(nume||' '||prenume||' va incasa in luna '||t(i).nume\_luna||' '||salariu\_luna);

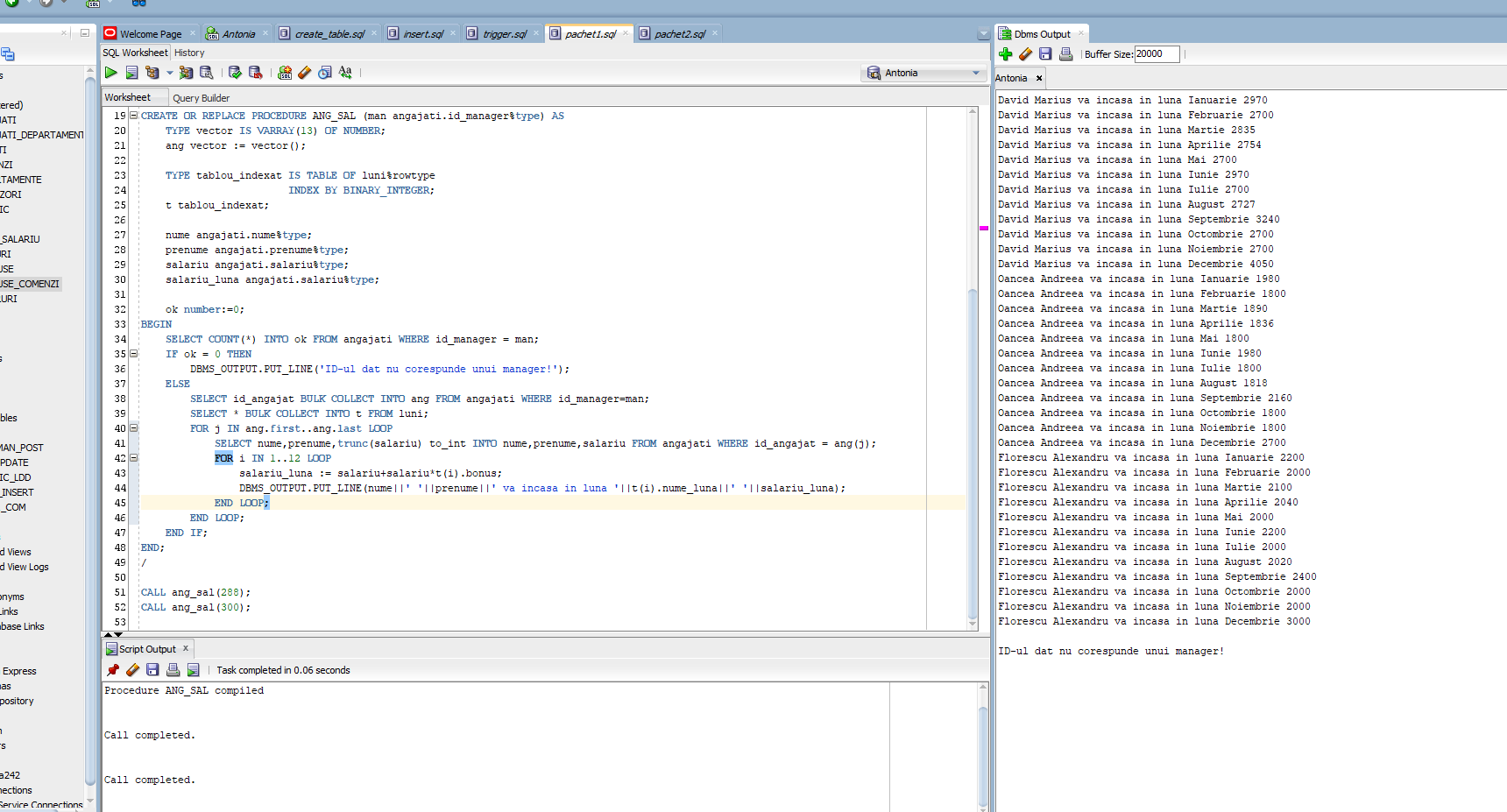
END LOOP;

END LOOP;

END IF;

END;

/



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 departament afisati produsele care au pretul mai mare sau egal decat un parametru.

CREATE OR REPLACE PROCEDURE PROD\_DEP(par number) AS

CURSOR c\_prod (parametru NUMBER) IS

SELECT id\_produs, nume\_produs, pret, id\_furnizor, id\_departament

FROM produse WHERE pret >= parametru;

produs produse.id\_produs%type;

nume produse.nume\_produs%type;

pret produse.pret%type;

furnizor produse.id\_furnizor%type;

departament produse.id\_departament%type;

dep departamente.id\_departament%type;

BEGIN

FOR i IN (SELECT id\_departament FROM departamente) LOOP --ciclu cursor

OPEN c\_prod(par);

DBMS\_OUTPUT.PUT\_LINE('DEPARTAMENTUL '||i.id\_departament);

LOOP

FETCH c\_prod INTO produs,nume,pret,furnizor,departament;

EXIT WHEN c\_prod%NOTFOUND;

IF departament = i.id\_departament THEN

DBMS\_OUTPUT.PUT\_LINE(produs||' : '||nume||',pret = '||pret);

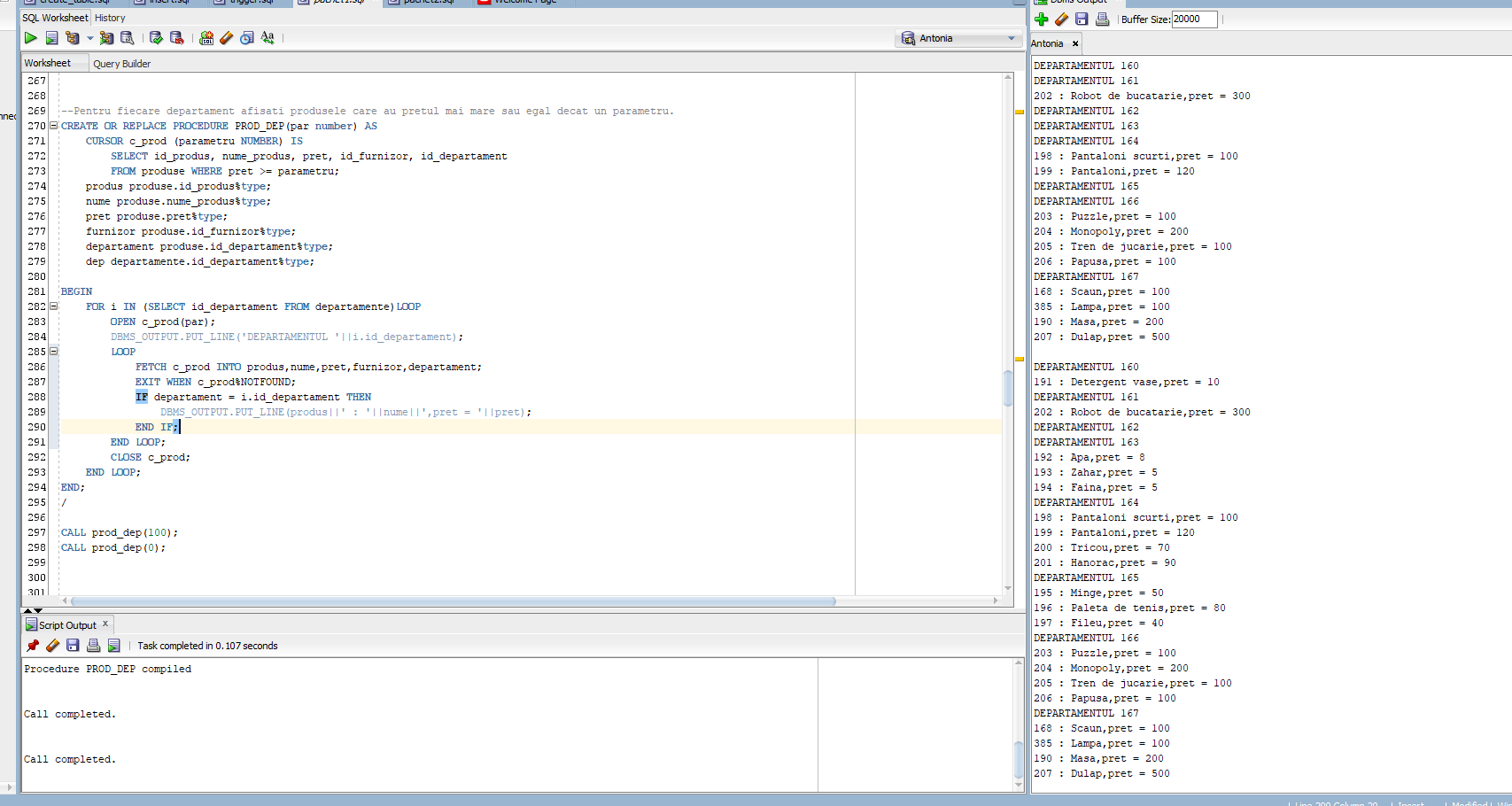
END IF;

END LOOP;

CLOSE c\_prod;

END LOOP;

END;

/

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.

-- Pentru un angajat dat afisati departamentul la care lucreaza.

CREATE OR REPLACE FUNCTION ANG\_DEP (nume\_a angajati.nume%type,prenume\_a angajati.prenume%type)

RETURN departamente.nume\_departament%type IS

dep departamente.nume\_departament%type;

ct number;

MAI\_MULTE\_DEPARTAMENTE EXCEPTION;

BEGIN

SELECT COUNT(\*) INTO ct FROM angajati

WHERE nume = nume\_a AND prenume = prenume\_a;

IF ct = 0 THEN

RAISE NO\_DATA\_FOUND;

END IF;

SELECT nume\_departament INTO dep FROM departamente

WHERE id\_departament = (

SELECT max(id\_departament) FROM angajati\_departamente

WHERE id\_angajat = (

SELECT id\_angajat FROM angajati

WHERE upper(nume) = upper(nume\_a) AND upper(prenume) = upper(prenume\_a)

)

);

SELECT COUNT(\*) INTO ct FROM angajati\_departamente

WHERE id\_angajat = (

SELECT id\_angajat FROM angajati

WHERE upper(nume) = upper(nume\_a) AND upper(prenume) = upper(prenume\_a)

);

IF ct > 1 THEN

RAISE MAI\_MULTE\_DEPARTAMENTE;

ELSE

RETURN dep;

END IF;

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

RAISE\_APPLICATION\_ERROR(-20001,'Nu exista angajatul');

WHEN MAI\_MULTE\_DEPARTAMENTE THEN

RAISE\_APPLICATION\_ERROR(-20002,'Angajatul lucreaza la mai multe departamente,printre care si '||dep);

WHEN OTHERS THEN

RAISE\_APPLICATION\_ERROR(-20003,'Alta eroare!');

END;

/

BEGIN

DBMS\_OUTPUT.PUT\_LINE(ang\_dep('Popa','Daniel')); -- Angajatul lucreaza la mai multe departamente,printre care si Sport

END;

/

BEGIN

DBMS\_OUTPUT.PUT\_LINE(ang\_dep('Popescu','Diana')); -- Haine

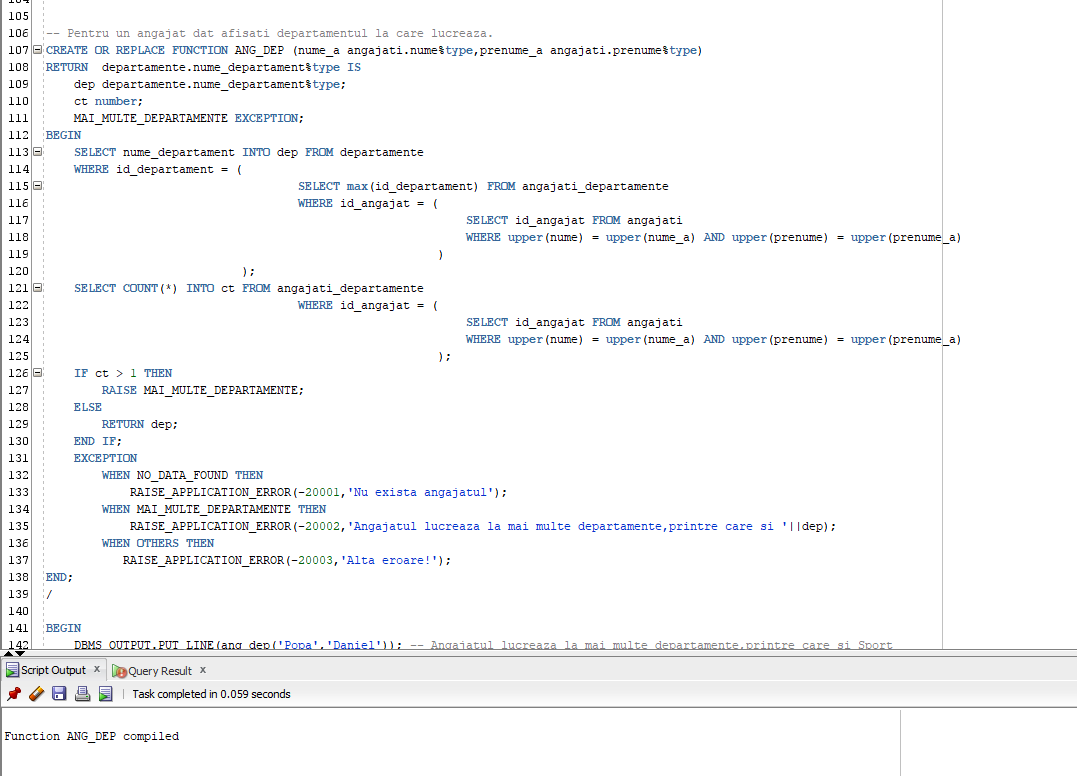
END;

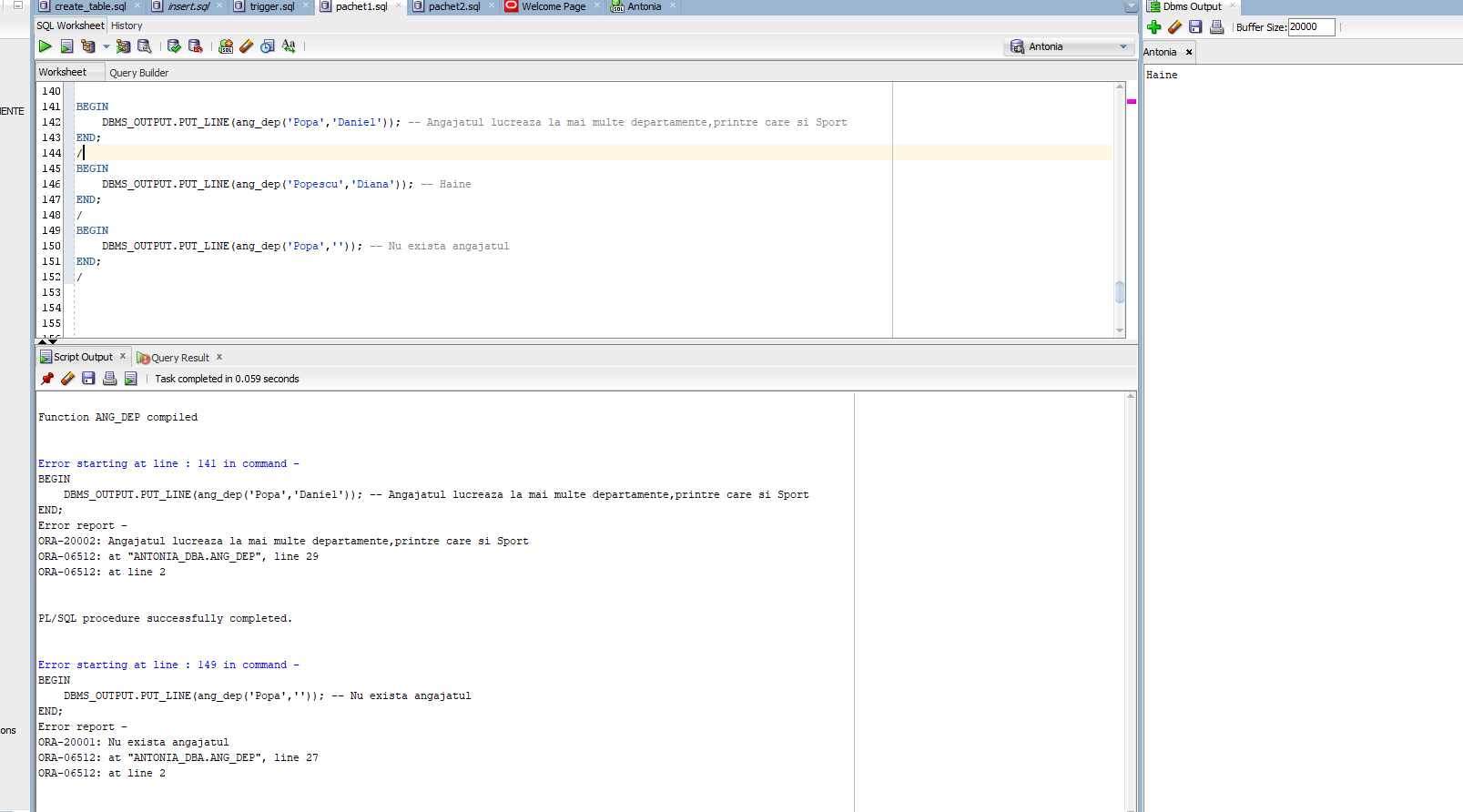
/

BEGIN

DBMS\_OUTPUT.PUT\_LINE(ang\_dep('Popa','')); -- Nu exista angajatul

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 un furnizor dat afisati un client care a cumparat unul din produsele ce apartin furnizorul respectiv.

CREATE OR REPLACE PROCEDURE FURN\_CL (nume\_f furnizori.nume\_furnizor%type) IS

ct number;

cli clienti.nume%type;

furnizor furnizori.id\_furnizor%type;

ACELASI\_NUME EXCEPTION;

NU\_EXISTA\_PRODUSE EXCEPTION;

NU\_EXISTA\_COMENZI EXCEPTION;

BEGIN

SELECT COUNT(\*) INTO ct FROM furnizori

WHERE nume\_furnizor = nume\_f;

IF ct>1 THEN

RAISE ACELASI\_NUME;

END IF;

SELECT id\_furnizor INTO furnizor FROM furnizori

WHERE nume\_furnizor = nume\_f;

SELECT COUNT(\*) INTO ct FROM produse

WHERE id\_furnizor = furnizor;

IF ct=0 THEN

RAISE NU\_EXISTA\_PRODUSE;

END IF;

SELECT count(\*) INTO ct FROM produse\_comenzi

WHERE id\_produs in (

SELECT id\_produs FROM produse

WHERE id\_furnizor = furnizor

);

IF ct=0 THEN

RAISE NU\_EXISTA\_COMENZI;

END IF;

-- Pentru a arunca TOO\_MANY\_ROWS am cautat numele persoanei dupa id, apoi am cautat id-ul dupa nume, fapt ce mi-a returnat mai multe id-uri,

-- apoi am introdus in cli numele angajatului cu id-ul dat , iar in cli pot insera doar unul.

SELECT nume INTO cli FROM clienti

WHERE id\_client in (

SELECT id\_client FROM clienti

where nume = (

SELECT nume FROM clienti

WHERE id\_client in (

SELECT max(id\_client) FROM comenzi

WHERE id\_comanda in (

SELECT id\_comanda FROM produse\_comenzi

WHERE id\_produs in (

SELECT id\_produs FROM produse

WHERE id\_furnizor = (

SELECT id\_furnizor FROM furnizori

WHERE upper(nume\_furnizor) = upper(nume\_f)

)

)

)

)

)

);

DBMS\_OUTPUT.PUT\_LINE('Un produs de la furnizorul: '||nume\_f ||' a fost achizitionat de '||cli);

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

RAISE\_APPLICATION\_ERROR(-20001,'Nu exista furnizorul!');

WHEN NU\_EXISTA\_PRODUSE THEN

RAISE\_APPLICATION\_ERROR(-20002,'Nu exista produse ce apartin furnizorului dat ca parametru!');

WHEN NU\_EXISTA\_COMENZI THEN

RAISE\_APPLICATION\_ERROR(-20003,'Nu exista comenzi ce contin produse ce apartin furnizorului dat ca parametru!');

WHEN ACELASI\_NUME THEN

RAISE\_APPLICATION\_ERROR(-20004,'Exista mai multi furnizori cu acest nume!');

WHEN TOO\_MANY\_ROWS THEN

RAISE\_APPLICATION\_ERROR(-20006,'Mai multi clienti cu acelasi nume!');

END;

/

BEGIN

FURN\_CL('furnizor1'); -- Mai multi clienti cu acelasi nume!

END;

/

BEGIN

FURN\_CL('furnizor'); -- Nu exista furnizorul!

END;

/

BEGIN

FURN\_CL('furnizor5'); -- Popescu

END;

/

BEGIN

FURN\_CL('furnizor4'); -- Exista mai multi furnizori cu acest nume!

END;

/

BEGIN

FURN\_CL('furnizor6'); --Nu exista produse ce apartin furnizorului dat ca parametru!

END;

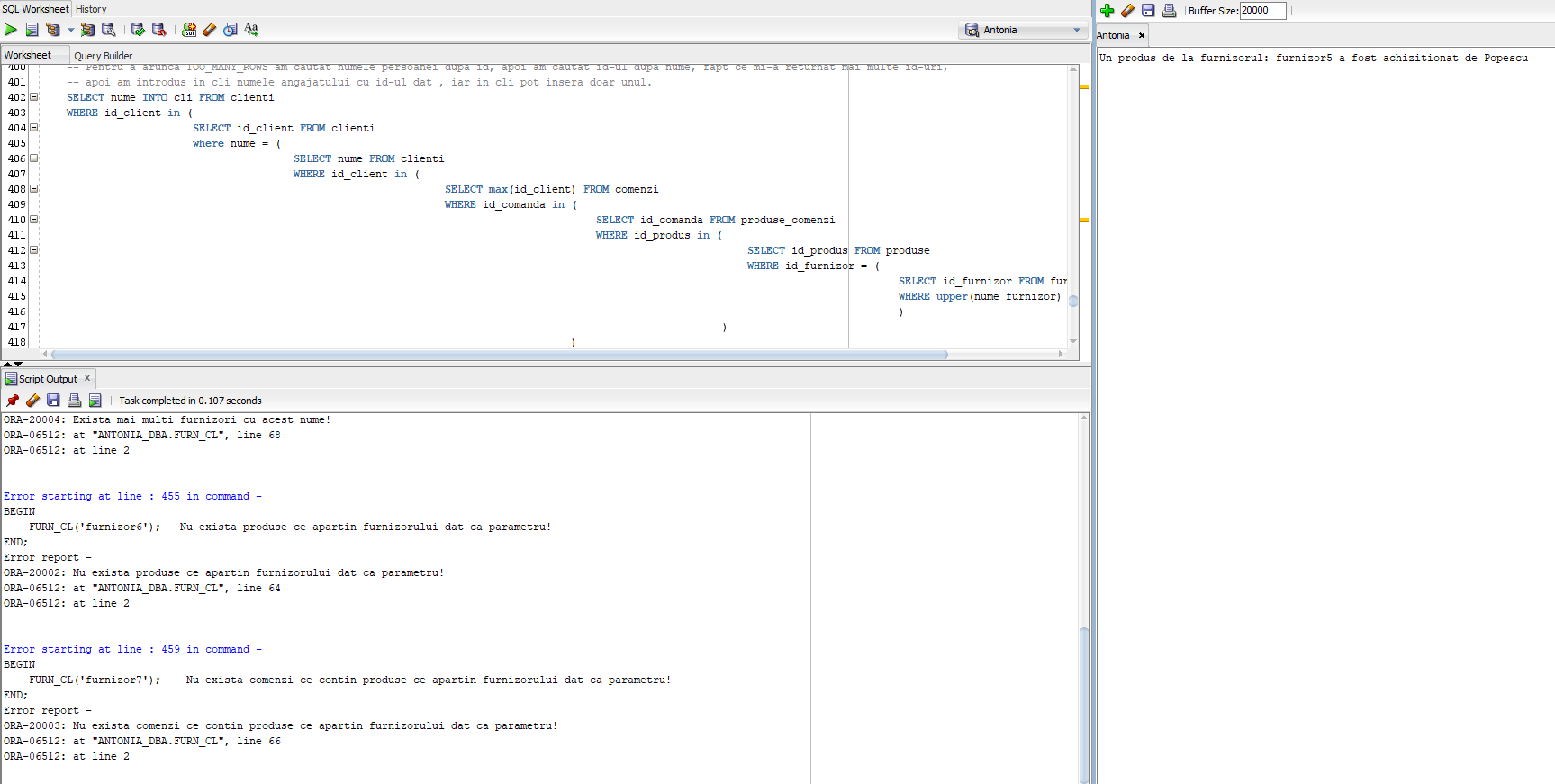
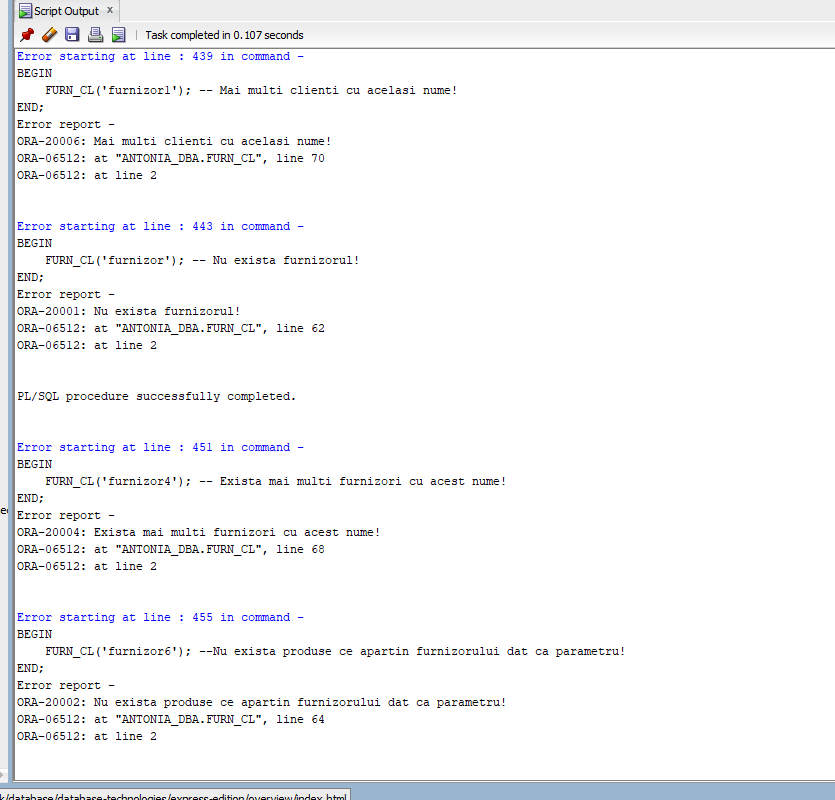
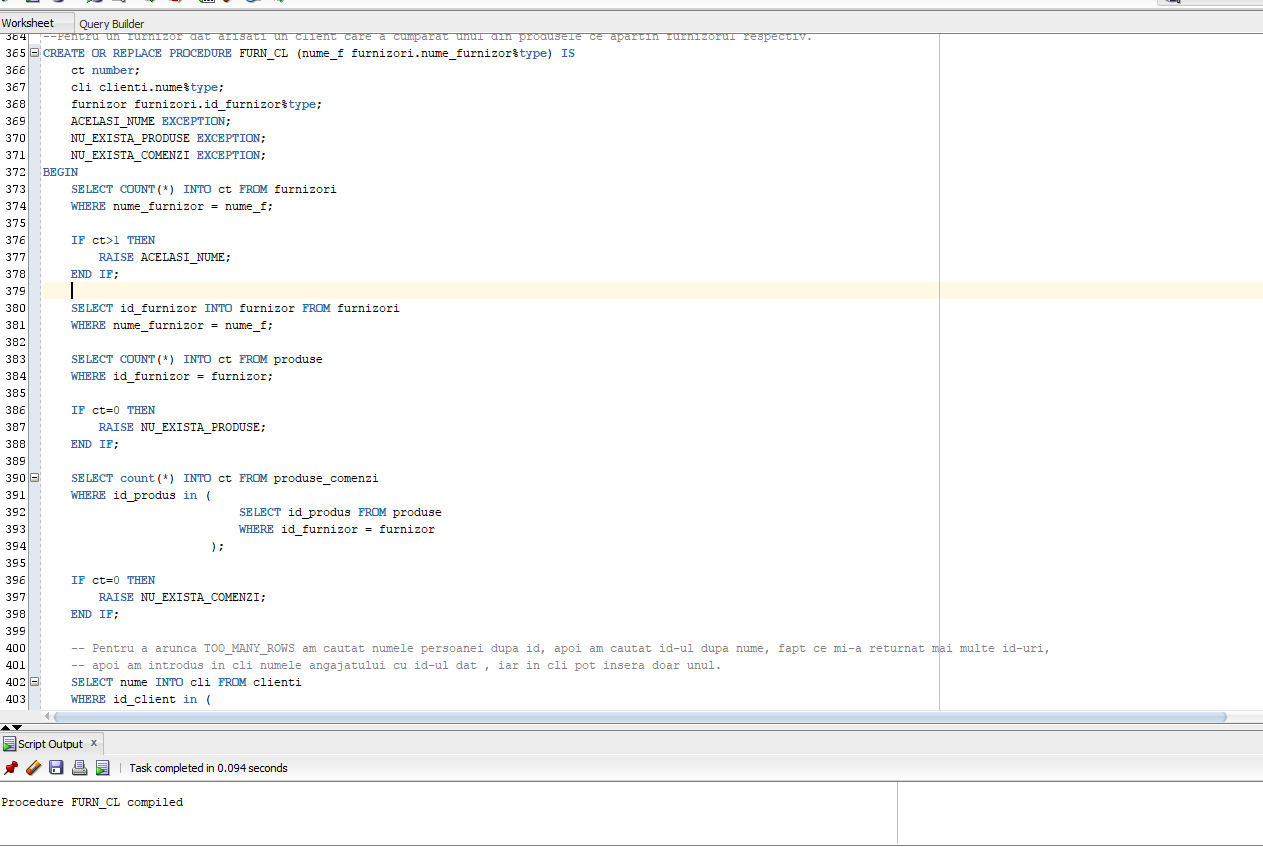
/

BEGIN

FURN\_CL('furnizor7'); -- Nu exista comenzi ce contin produse ce apartin furnizorului dat ca parametru!

END;

/



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

--NU SE PERMITE MODIFICAREA TABELULUI DEPARTAMENTE

CREATE OR REPLACE TRIGGER DEP\_UPDATE

BEFORE UPDATE ON DEPARTAMENTE

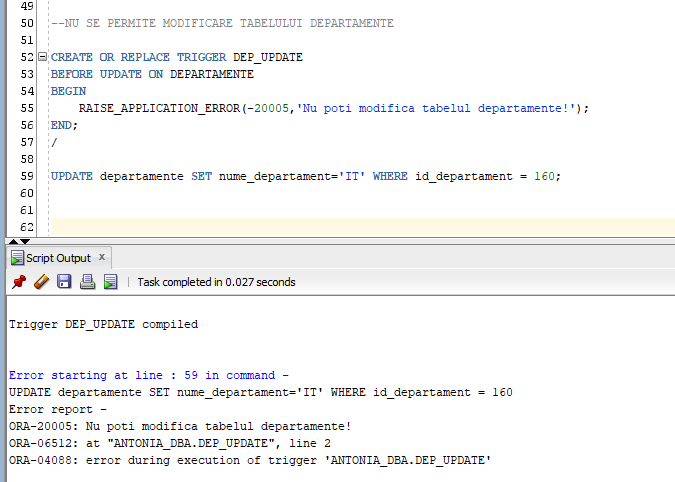
BEGIN

RAISE\_APPLICATION\_ERROR(-20005,'Nu poti modifica tabelul departamente!');

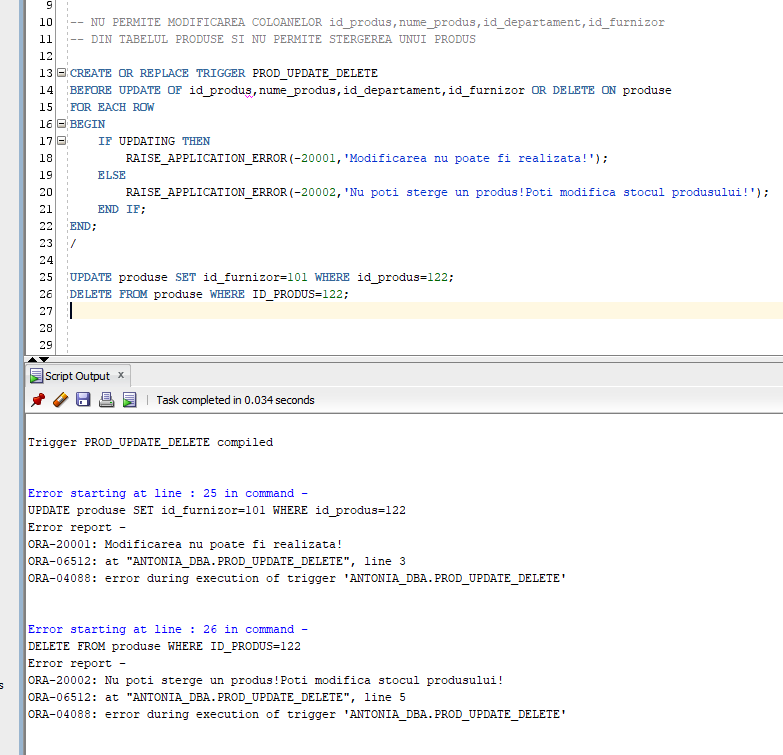
END;

/

--UPDATE departamente SET nume\_departament='IT' WHERE id\_departament = 160;



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



-- NU PERMITE MODIFICAREA COLOANELOR id\_produs,nume\_produs,id\_departament,id\_furnizor

-- DIN TABELUL PRODUSE SI NU PERMITE STERGEREA UNUI PRODUS

CREATE OR REPLACE TRIGGER PROD\_UPDATE\_DELETE

BEFORE UPDATE OF id\_produs,nume\_produs,id\_departament,id\_furnizor OR DELETE ON produse

FOR EACH ROW

DECLARE

nr INT:=null;

BEGIN

IF UPDATING THEN

RAISE\_APPLICATION\_ERROR(-20001,'Modificarea nu poate fi realizata!');

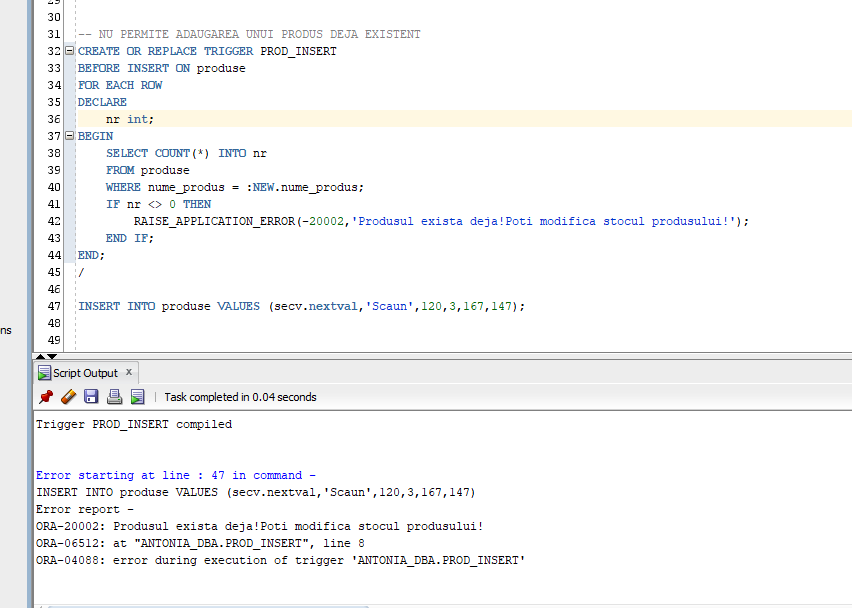
ELSE

RAISE\_APPLICATION\_ERROR(-20002,'Nu poti sterge un produs!Poti modifica stocul produsului!');

END IF;

END;

/



-- NU PERMITE ADAUGAREA UNUI PRODUS DEJA EXISTENT

CREATE OR REPLACE TRIGGER PROD\_INSERT

BEFORE INSERT ON produse

FOR EACH ROW

DECLARE

nr int;

BEGIN

SELECT COUNT(\*) INTO nr

FROM produse

WHERE nume\_produs = :NEW.nume\_produs;

IF nr <> 0 THEN

RAISE\_APPLICATION\_ERROR(-20002,'Produsul exista deja!Poti modifica stocul produsului!');

END IF;

END;

/



--DACA LA INSERAREA UNUI ANGAJAT ID\_POST NU ESTE NULL VERIFICAM DACA SALARIUL ANGAJATULUI ESTE MAI MARE DECAT

--SALARIUL MINIM PENTRU POSTUL RESPECTIV

--UN ANGAJAT NOU NU POATE FI MANAGER

CREATE OR REPLACE TRIGGER ANG\_MAN\_POST

BEFORE INSERT ON angajati

FOR EACH ROW

DECLARE

salariu\_post posturi.salariu\_minim%type;

BEGIN

IF :NEW.id\_post = 5 THEN

RAISE\_APPLICATION\_ERROR(-20007,'Un angajat nou nu poate fi manager!');

END IF;

IF :NEW.id\_post >=1 AND :NEW.id\_post<=5

THEN

SELECT salariu\_minim INTO salariu\_post FROM posturi WHERE id\_post = :NEW.id\_post;

IF salariu\_post > :NEW.salariu THEN

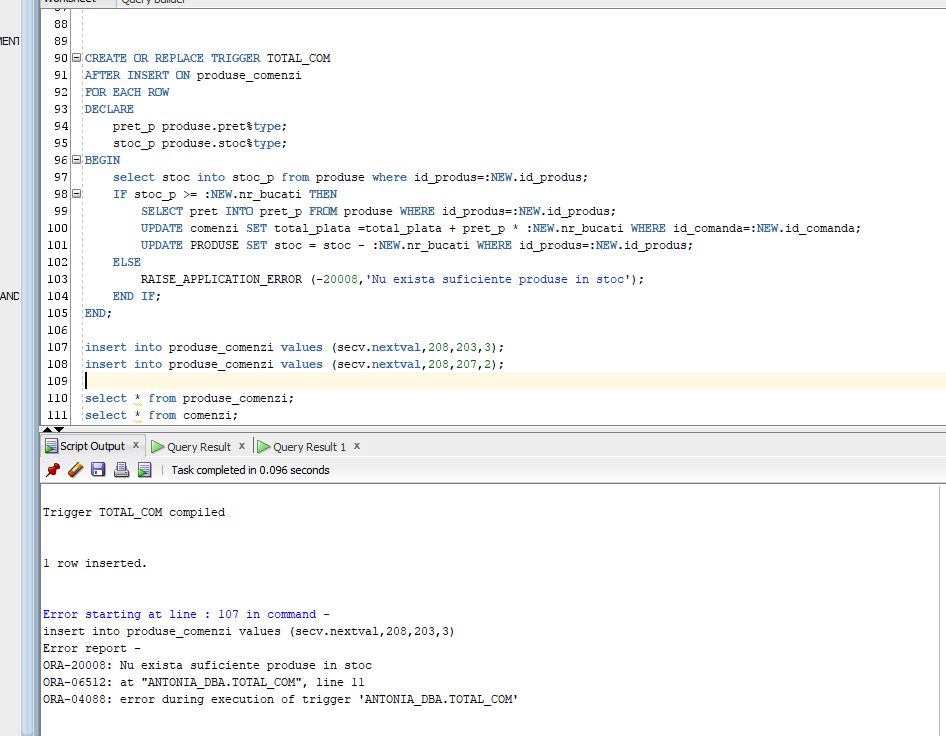
RAISE\_APPLICATION\_ERROR(-20007,'Angajatul trebuie sa aiba salariul mai mare de '||salariu\_post||'!');

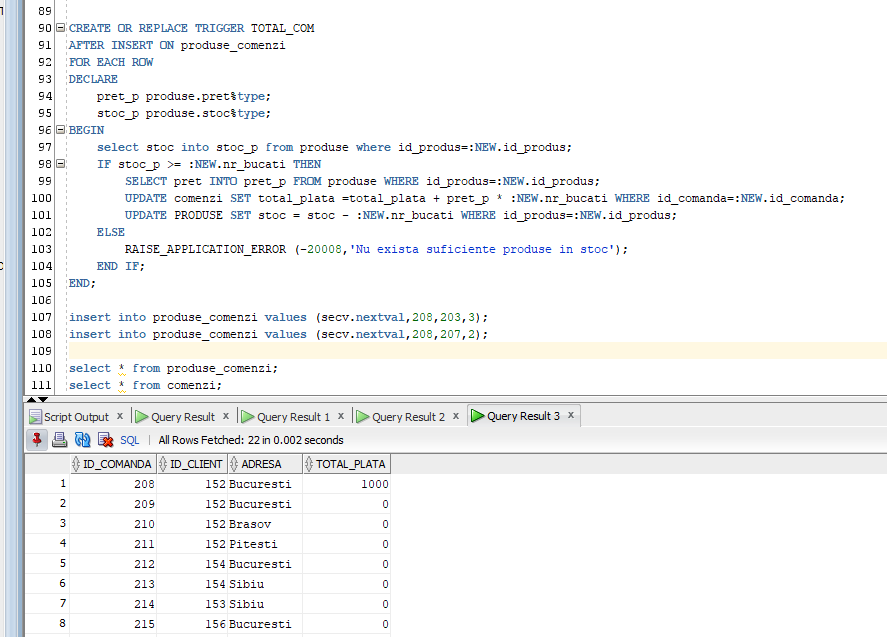
END IF;

END IF;

END;

/





--LA INSERAREA UNUI PRODUS IN COMENZI SA SE REACTUALIZEZE PRETUL TOTAL DIN COMENZI SI STOCUL DIN PRODUSE IN CAZUL IN CARE EXISTA SUFICIENTE PRODUSE IN STOC

CREATE OR REPLACE TRIGGER TOTAL\_COM

AFTER INSERT ON produse\_comenzi

FOR EACH ROW

DECLARE

pret\_p produse.pret%type;

stoc\_p produse.stoc%type;

BEGIN

select stoc into stoc\_p from produse where id\_produs=:NEW.id\_produs;

IF stoc\_p >= :NEW.nr\_bucati THEN

SELECT pret INTO pret\_p FROM produse WHERE id\_produs=:NEW.id\_produs;

UPDATE comenzi SET total\_plata =total\_plata + pret\_p \* :NEW.nr\_bucati WHERE id\_comanda=:NEW.id\_comanda;

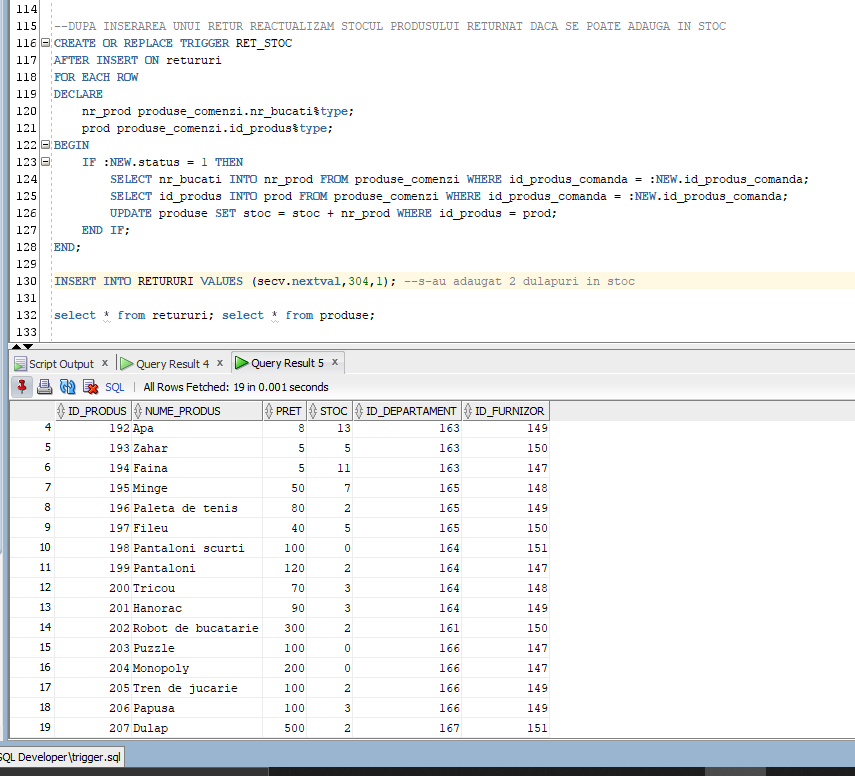
UPDATE PRODUSE SET stoc = stoc - :NEW.nr\_bucati WHERE id\_produs=:NEW.id\_produs;

ELSE

RAISE\_APPLICATION\_ERROR (-20008,'Nu exista suficiente produse in stoc');

END IF;

END;



--DUPA INSERAREA UNUI RETUR REACTUALIZAM STOCUL PRODUSULUI RETURNAT DACA SE POATE ADAUGA IN STOC

CREATE OR REPLACE TRIGGER RET\_STOC

AFTER INSERT ON retururi

FOR EACH ROW

DECLARE

nr\_prod produse\_comenzi.nr\_bucati%type;

prod produse\_comenzi.id\_produs%type;

BEGIN

IF :NEW.status = 1 THEN

SELECT nr\_bucati INTO nr\_prod FROM produse\_comenzi WHERE id\_produs\_comanda = :NEW.id\_produs\_comanda;

SELECT id\_produs INTO prod FROM produse\_comenzi WHERE id\_produs\_comanda = :NEW.id\_produs\_comanda;

UPDATE produse SET stoc = stoc + nr\_prod WHERE id\_produs = prod;

END IF;

END;

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

--ADAUGAM EVENIMENTELE DE TIP CREATE, DROP SI ALTER IN TABELUL ISTORIC

CREATE OR REPLACE TRIGGER istoric\_LDD

AFTER CREATE OR DROP OR ALTER ON SCHEMA

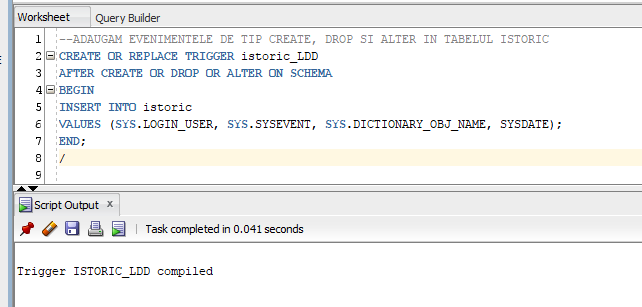
BEGIN

INSERT INTO istoric

VALUES (SYS.LOGIN\_USER, SYS.SYSEVENT, SYS.DICTIONARY\_OBJ\_NAME, SYSDATE);

END;

/



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

CREATE OR REPLACE PACKAGE pachet1

AS

PROCEDURE ANG\_SAL (man angajati.id\_manager%type); --ex6

PROCEDURE PROD\_DEP(par number); --ex7

FUNCTION ANG\_DEP (nume\_a angajati.nume%type,prenume\_a angajati.prenume%type) RETURN departamente.nume\_departament%type; --ex8

PROCEDURE FURN\_CL (nume\_f furnizori.nume\_furnizor%type); --ex9

END pachet1;

/

CREATE OR REPLACE PACKAGE BODY pachet1 AS

--Pentru un manager dat ca parametru afisati pentru angajatii pe care ii are in subordine salariile pe care

--le primesc acestia in fiecare luna a anului. Daca id-ul nu corespunde unui manager afisati un mesaj semnificativ.

PROCEDURE ANG\_SAL (man angajati.id\_manager%type) AS

TYPE vector IS VARRAY(13) OF NUMBER;

ang vector := vector();

TYPE tablou\_indexat IS TABLE OF luni%rowtype

INDEX BY BINARY\_INTEGER;

t tablou\_indexat;

nume angajati.nume%type;

prenume angajati.prenume%type;

salariu angajati.salariu%type;

salariu\_luna angajati.salariu%type;

ok number:=0;

BEGIN

SELECT COUNT(\*) INTO ok FROM angajati WHERE id\_manager = man;

IF ok = 0 THEN

DBMS\_OUTPUT.PUT\_LINE('ID-ul dat nu corespunde unui manager!');

ELSE

SELECT id\_angajat BULK COLLECT INTO ang FROM angajati WHERE id\_manager=man;

SELECT \* BULK COLLECT INTO t FROM luni;

FOR j IN ang.first..ang.last LOOP

SELECT nume,prenume,trunc(salariu) to\_int INTO nume,prenume,salariu FROM angajati WHERE id\_angajat = ang(j);

FOR i IN 1..12 LOOP

salariu\_luna := salariu+salariu\*t(i).bonus;

DBMS\_OUTPUT.PUT\_LINE(nume||' '||prenume||' va incasa in luna '||t(i).nume\_luna||' '||salariu\_luna);

END LOOP;

END LOOP;

END IF;

END ANG\_SAL;

--Pentru fiecare departament afisati produsele care au pretul mai mare sau egal decat un parametru.

PROCEDURE PROD\_DEP(par number) AS

CURSOR c\_prod (parametru NUMBER) IS

SELECT id\_produs, nume\_produs, pret, id\_furnizor, id\_departament

FROM produse WHERE pret >= parametru;

produs produse.id\_produs%type;

nume produse.nume\_produs%type;

pret produse.pret%type;

furnizor produse.id\_furnizor%type;

departament produse.id\_departament%type;

dep departamente.id\_departament%type;

BEGIN

FOR i IN (SELECT id\_departament FROM departamente)LOOP

OPEN c\_prod(par);

DBMS\_OUTPUT.PUT\_LINE('DEPARTAMENTUL '||i.id\_departament);

LOOP

FETCH c\_prod INTO produs,nume,pret,furnizor,departament;

EXIT WHEN c\_prod%NOTFOUND;

IF departament = i.id\_departament THEN

DBMS\_OUTPUT.PUT\_LINE(produs||' : '||nume||',pret = '||pret);

END IF;

END LOOP;

CLOSE c\_prod;

END LOOP;

END PROD\_DEP;

-- Pentru un angajat dat afisati departamentul la care lucreaza.

FUNCTION ANG\_DEP (nume\_a angajati.nume%type,prenume\_a angajati.prenume%type)

RETURN departamente.nume\_departament%type IS

dep departamente.nume\_departament%type;

ct number;

MAI\_MULTE\_DEPARTAMENTE EXCEPTION;

BEGIN

SELECT nume\_departament INTO dep FROM departamente

WHERE id\_departament = (

SELECT max(id\_departament) FROM angajati\_departamente

WHERE id\_angajat = (

SELECT id\_angajat FROM angajati

WHERE upper(nume) = upper(nume\_a) AND upper(prenume) = upper(prenume\_a)

)

);

SELECT COUNT(\*) INTO ct FROM angajati\_departamente

WHERE id\_angajat = (

SELECT id\_angajat FROM angajati

WHERE upper(nume) = upper(nume\_a) AND upper(prenume) = upper(prenume\_a)

);

IF ct > 1 THEN

RAISE MAI\_MULTE\_DEPARTAMENTE;

ELSE

RETURN dep;

END IF;

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

RAISE\_APPLICATION\_ERROR(-20001,'Nu exista angajatul');

WHEN MAI\_MULTE\_DEPARTAMENTE THEN

RAISE\_APPLICATION\_ERROR(-20002,'Angajatul lucreaza la mai multe departamente,printre care si '||dep);

WHEN OTHERS THEN

RAISE\_APPLICATION\_ERROR(-20003,'Alta eroare!');

END ANG\_DEP;

--Pentru un furnizor dat afisati un client care a cumparat unul din produsele ce apartin furnizorul respectiv.

PROCEDURE FURN\_CL (nume\_f furnizori.nume\_furnizor%type) IS

ct number;

cli clienti.nume%type;

furnizor furnizori.id\_furnizor%type;

ACELASI\_NUME EXCEPTION;

NU\_EXISTA\_PRODUSE EXCEPTION;

NU\_EXISTA\_COMENZI EXCEPTION;

BEGIN

SELECT COUNT(\*) INTO ct FROM furnizori

WHERE nume\_furnizor = nume\_f;

IF ct>1 THEN

RAISE ACELASI\_NUME;

END IF;

SELECT id\_furnizor INTO furnizor FROM furnizori

WHERE nume\_furnizor = nume\_f;

SELECT COUNT(\*) INTO ct FROM produse

WHERE id\_furnizor = furnizor;

IF ct=0 THEN

RAISE NU\_EXISTA\_PRODUSE;

END IF;

SELECT count(\*) INTO ct FROM produse\_comenzi

WHERE id\_produs in (

SELECT id\_produs FROM produse

WHERE id\_furnizor = furnizor

);

IF ct=0 THEN

RAISE NU\_EXISTA\_COMENZI;

END IF;

-- Pentru a arunca TOO\_MANY\_ROWS am cautat numele persoanei dupa id, apoi am cautat id-ul dupa nume, fapt ce mi-a returnat mai multe id-uri,

-- apoi am introdus in cli numele angajatului cu id-ul dat , iar in cli pot insera doar unul.

SELECT nume INTO cli FROM clienti

WHERE id\_client in (

SELECT id\_client FROM clienti

where nume = (

SELECT nume FROM clienti

WHERE id\_client in (

SELECT max(id\_client) FROM comenzi

WHERE id\_comanda in (

SELECT id\_comanda FROM produse\_comenzi

WHERE id\_produs in (

SELECT id\_produs FROM produse

WHERE id\_furnizor = (

SELECT id\_furnizor FROM furnizori

WHERE upper(nume\_furnizor) = upper(nume\_f)

)

)

)

)

)

);

DBMS\_OUTPUT.PUT\_LINE('Un produs de la furnizorul: '||nume\_f ||' a fost achizitionat de '||cli);

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

RAISE\_APPLICATION\_ERROR(-20001,'Nu exista furnizorul!');

WHEN NU\_EXISTA\_PRODUSE THEN

RAISE\_APPLICATION\_ERROR(-20002,'Nu exista produse ce apartin furnizorului dat ca parametru!');

WHEN NU\_EXISTA\_COMENZI THEN

RAISE\_APPLICATION\_ERROR(-20003,'Nu exista comenzi ce contin produse ce apartin furnizorului dat ca parametru!');

WHEN ACELASI\_NUME THEN

RAISE\_APPLICATION\_ERROR(-20004,'Exista mai multi furnizori cu acest nume!');

WHEN TOO\_MANY\_ROWS THEN

RAISE\_APPLICATION\_ERROR(-20006,'Mai multi clienti cu acelasi nume!');

END FURN\_CL;

END pachet1;

/

--testam pachetul

CALL pachet1.ang\_sal(288);

CALL pachet1.prod\_dep(100);

BEGIN

DBMS\_OUTPUT.PUT\_LINE(pachet1.ang\_dep('Popescu','Diana')); -- Haine

END;

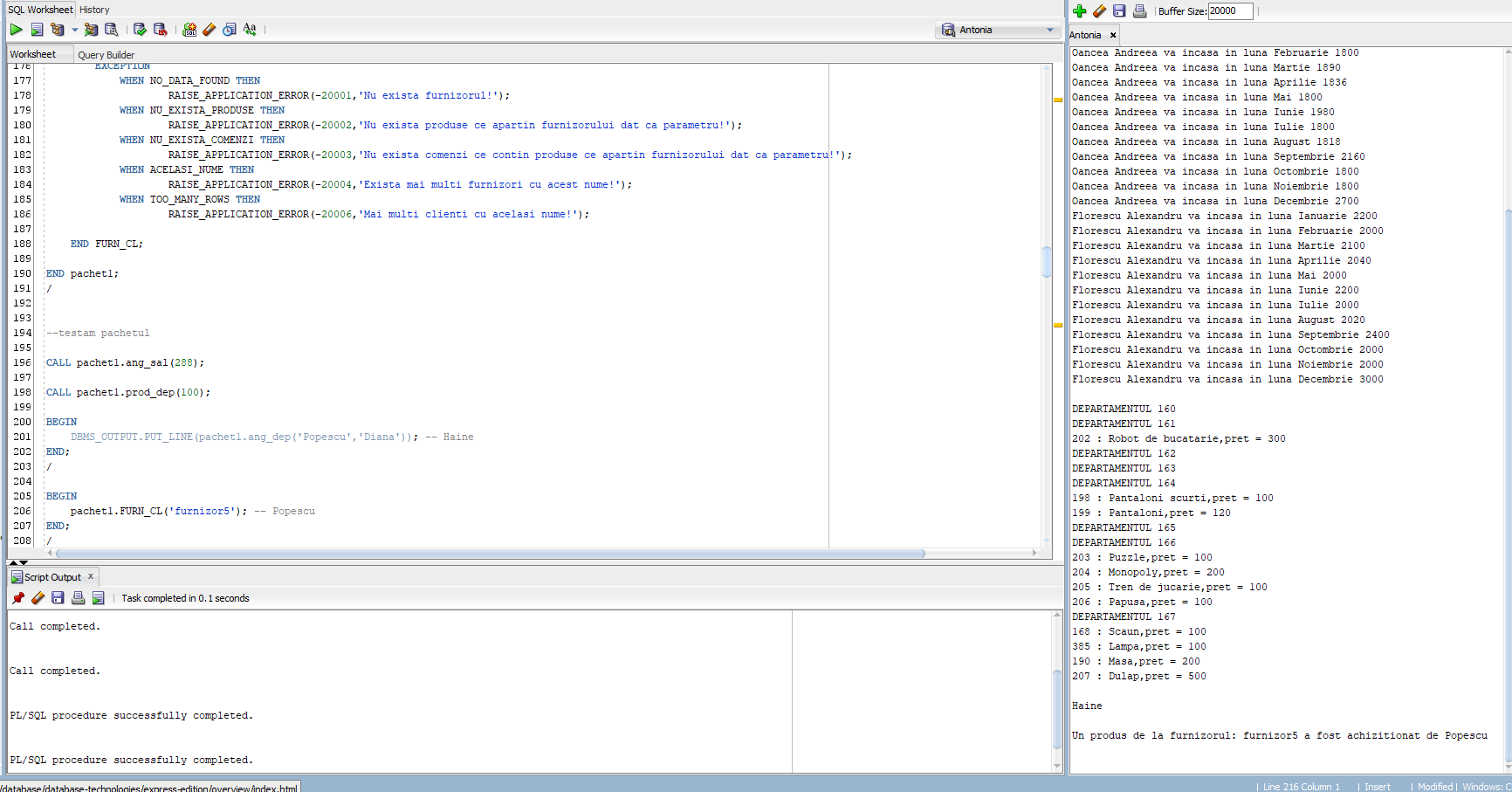
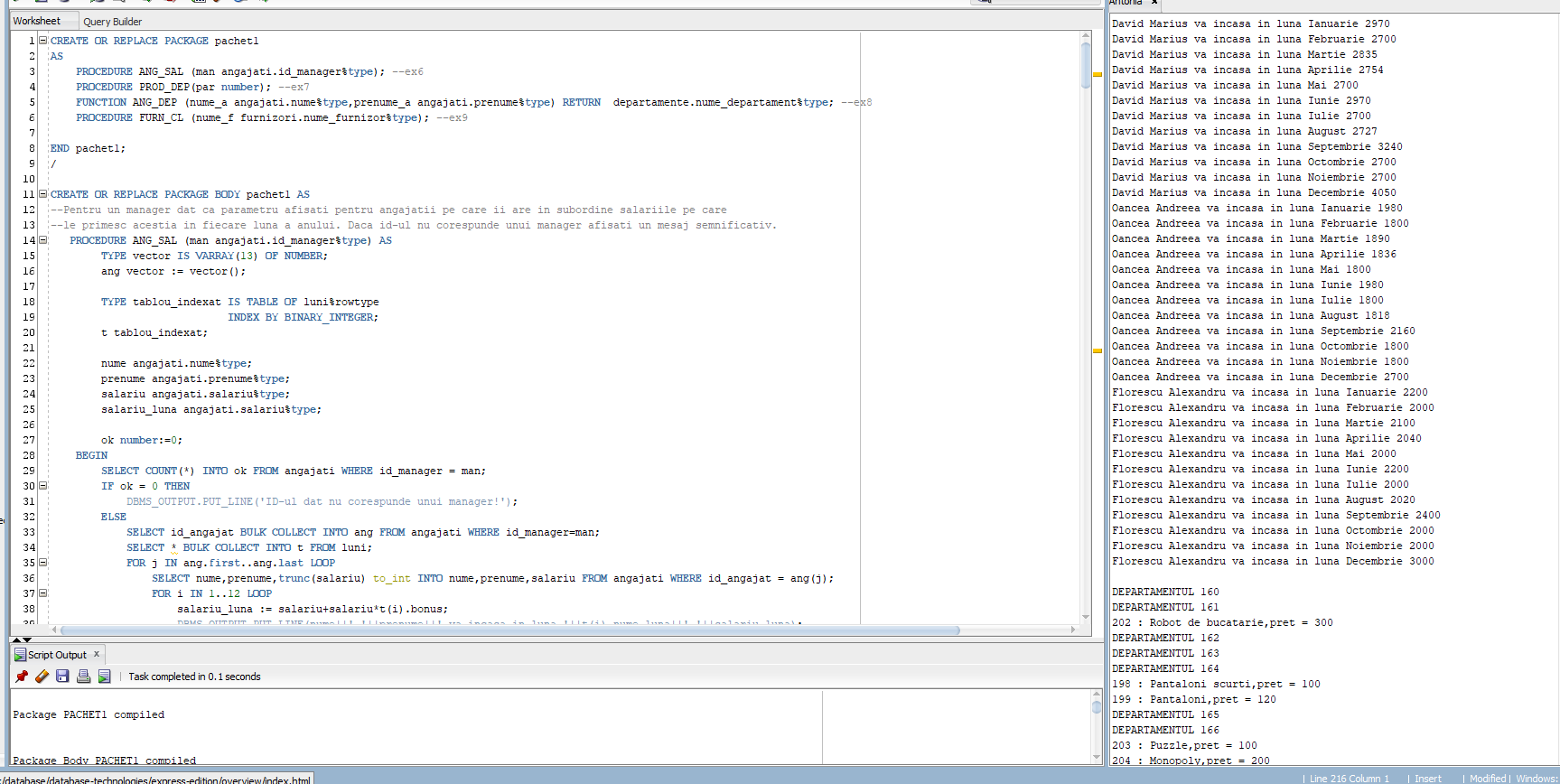
/

BEGIN

pachet1.FURN\_CL('furnizor5'); -- Popescu

END;

/



1. 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 pachet2

AS

TYPE tablou\_indexat IS TABLE OF angajati%rowtype INDEX BY BINARY\_INTEGER;

TYPE produs IS RECORD (id\_prod produse.id\_produs%type,

nume\_prod produse.nume\_produs%type,

stoc\_prod produse.stoc%type,

pret\_prod produse.pret%type);

PROCEDURE UPD\_MAN(ang angajati.id\_angajat%type);

PROCEDURE ALL\_MAN;

PROCEDURE COM\_PR (comanda comenzi.id\_comanda%type);

FUNCTION RET\_NR(comanda comenzi.id\_comanda%type) RETURN NUMBER;

FUNCTION RET\_PROD RETURN NUMBER;

FUNCTION GET\_NUME\_PRODUS(pr produse.id\_produs%type) RETURN produse.nume\_produs%type;

END pachet2;

/

CREATE OR REPLACE PACKAGE BODY pachet2 AS

-- setati postul MANAGER pentru angajati

PROCEDURE UPD\_MAN(ang angajati.id\_angajat%type) AS

BEGIN

UPDATE angajati SET id\_post = 5 WHERE id\_angajat = ang;

END UPD\_MAN;

-- pentru toti angajati care apar ca si id\_manager pentru un angajat setati postul MANAGER

PROCEDURE ALL\_MAN AS

emp tablou\_indexat;

BEGIN

SELECT \* BULK COLLECT INTO emp FROM angajati;

FOR i IN 1..emp.last LOOP

IF emp(i).id\_manager IS NOT NULL THEN

UPD\_MAN(emp(i).id\_manager);

END IF;

END LOOP;

END ALL\_MAN;

-- returnati numarul de produse comandate pentru o comanda data ca parametru

FUNCTION RET\_NR(comanda comenzi.id\_comanda%TYPE ) RETURN NUMBER AS

ct NUMBER:=0;

BEGIN

SELECT SUM(nr\_bucati) INTO ct FROM produse\_comenzi

WHERE id\_comanda = comanda;

return ct;

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

RAISE\_APPLICATION\_ERROR(-20020,'Nu exista comanda cu numarul '||comanda||'!');

END RET\_NR;

-- returnati produsul pentru care au fost comandate cele mai multe exemplare

-- daca sunt mai multe afisati-l pe oricare dintre ele

FUNCTION RET\_PROD RETURN NUMBER AS

pr PRODUS;

id\_pr produse.id\_produs%type;

maxim NUMBER;

BEGIN

SELECT \* INTO id\_pr,maxim

FROM (SELECT id\_produs,SUM(nr\_bucati) FROM produse\_comenzi

GROUP BY id\_produs

ORDER BY SUM(nr\_bucati) DESC

)

WHERE ROWNUM=1;

RETURN id\_pr;

END RET\_PROD;

-- returnati numele unui produs in functie de id-ul sau

FUNCTION GET\_NUME\_PRODUS(pr produse.id\_produs%type)

RETURN produse.nume\_produs%type IS

nume produse.nume\_produs%type;

BEGIN

SELECT nume\_produs INTO nume FROM produse

WHERE id\_produs = pr;

RETURN nume;

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

RAISE\_APPLICATION\_ERROR(-20001,'Nu exista produsul cu codul '||pr);

END;

--pentru o comanda data prin id-ul sau afisati produsele , iar daca nu contine produse afisati un mesaj sugestiv

PROCEDURE COM\_PR (comanda comenzi.id\_comanda%type) AS

CURSOR c IS SELECT \* FROM produse\_comenzi

WHERE id\_comanda = comanda;

ct NUMBER:=0;

ok NUMBER:=0;

BEGIN

SELECT COUNT(\*) INTO ok FROM comenzi

WHERE id\_comanda = comanda;

IF ok=0 THEN

RAISE NO\_DATA\_FOUND;

END IF;

ct:=ret\_nr(comanda);

IF ct=0 THEN

DBMS\_OUTPUT.PUT\_LINE('Comanda cu numarul '||comanda||' nu contine produse momentan!');

ELSE IF ct=1 THEN

DBMS\_OUTPUT.PUT\_LINE('Comanda cu numarul '||comanda||' contine un produs!');

ELSE

DBMS\_OUTPUT.PUT\_LINE('Comanda cu numarul '||comanda||' contine '|| ct||' produse!');

END IF;

END IF;

FOR i in c LOOP

DBMS\_OUTPUT.PUT\_LINE(GET\_NUME\_PRODUS(i.id\_produs)||' x'||i.nr\_bucati);

END LOOP;

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

RAISE\_APPLICATION\_ERROR(-20001,'Nu exista comanda cu numarul '||comanda);

END COM\_PR;

END pachet2;

/

--testam pachetul

BEGIN

pachet2.all\_man;

END;

/

CALL pachet2.upd\_man(286);

BEGIN

DBMS\_OUTPUT.PUT\_LINE(pachet2.ret\_prod||' : '||pachet2.get\_nume\_produs(pachet2.ret\_prod)||' este produsul pentru care s-au vandut cele mai multe exemplare'); --192 : Apa este produsul pentru care s-au vandut cele mai multe exemplare

END;

/

BEGIN

DBMS\_OUTPUT.PUT\_LINE('Comanda cu numarul 208 are '||pachet2.ret\_nr(208) ||' produse');--Comanda cu numarul 208 are 3 produse

END;

/

BEGIN

DBMS\_OUTPUT.PUT\_LINE('Produsul cu id-ul 190 este '||pachet2.get\_nume\_produs(190));--Produsul cu id-ul 190 este Masa

END;

/

BEGIN

pachet2.com\_pr(250); --ERROR : Nu exista comanda cu numarul 250

END;

/

BEGIN

pachet2.com\_pr(210);

-- Comanda cu numarul 210 contine 3 produse!

-- Papusa x1

-- Tricou x1

-- Pantaloni scurti x1

END;

/

