**Gestionarea Restaurantelor**

**dintr-un Oraș**

**Popescu Vasile-Alin**

**Grupa 244**

**Proiect SGBD anul II semestrul 1**

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

Mai multe restaurante dintr-un oraș comercializeaza diverse preparate, pe care le pot livra la o locație stabilita de client.

Dacă același magazin are doua sau mai multe locații,restaurantul va fi inserat de mai multe ori și ele pot avea aceleași preparate dar și preparate diferite.

Pentru fiecare preparat se știu: denumirea, ingredientele, prețul, numărul de calorii, greutatea și timpul de preparare.

Ingredientele pot fi locale sau importate din alte țări. Pentru fiecare ingredient se cunosc: denumirea, țara de origine, preț per sută de grame și data expirarii;

Țările sunt caracterizate prin: nume, monedă și continentul pe care se află.

Un client poate plasa una sau mai multe comenzi.

O comandă poate conține unul sau mai multe produse (nu neapărat toate de la același restaurant).

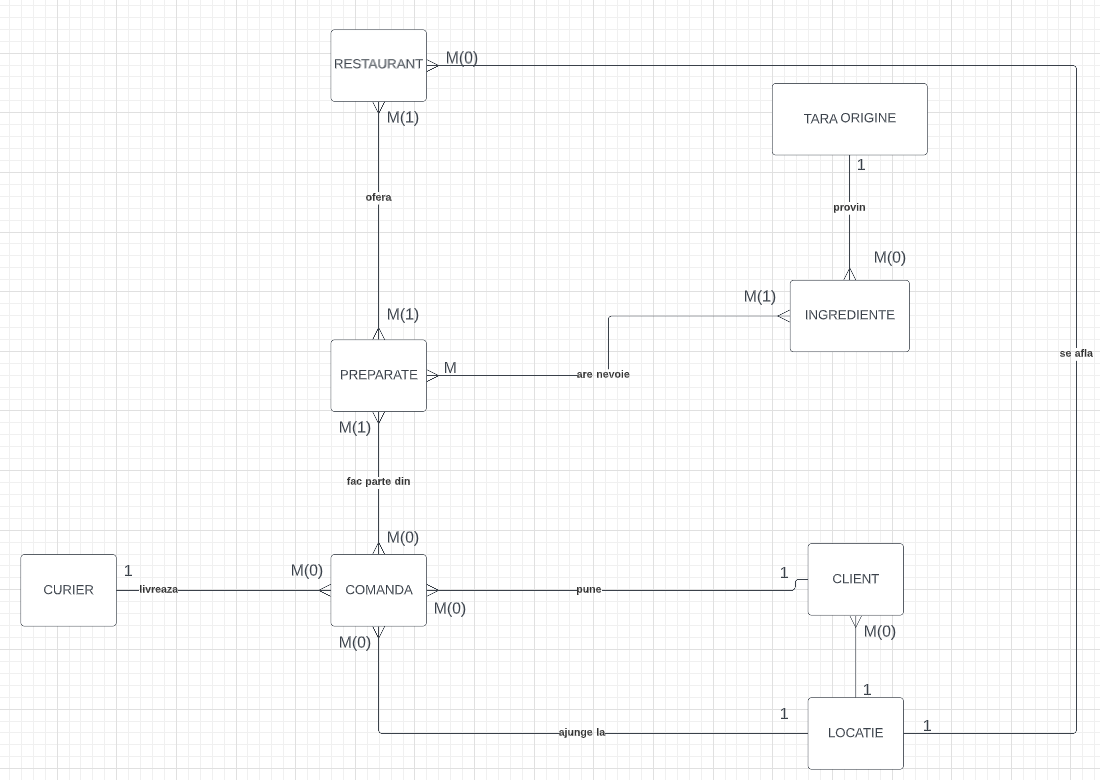
Livrarea se face la o locație aleasă de client, în ziua în care s-a efectuat comanda.

Clienții sunt caracterizați prin , nume prenume, număr de telefon și adresa.

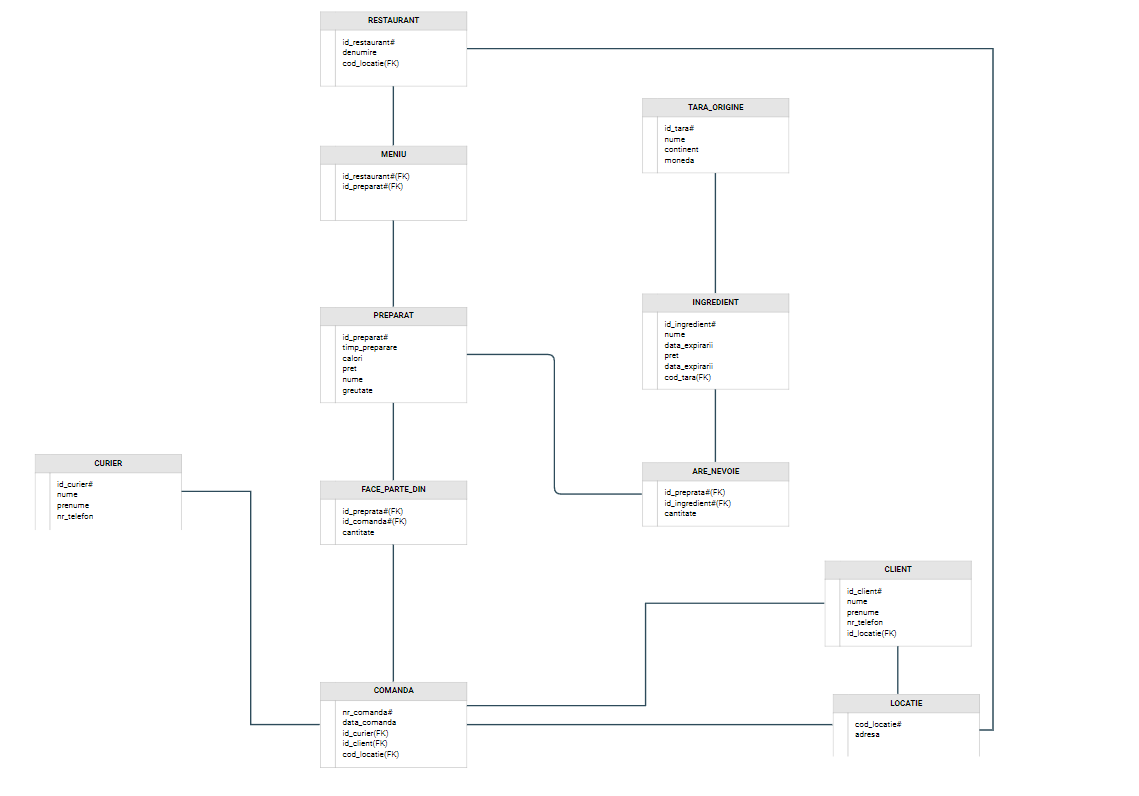
Comenzile sunt livrate de curieri, aceștia fiind caracterizati prin nume, prenume,număr de telefon și salariu.

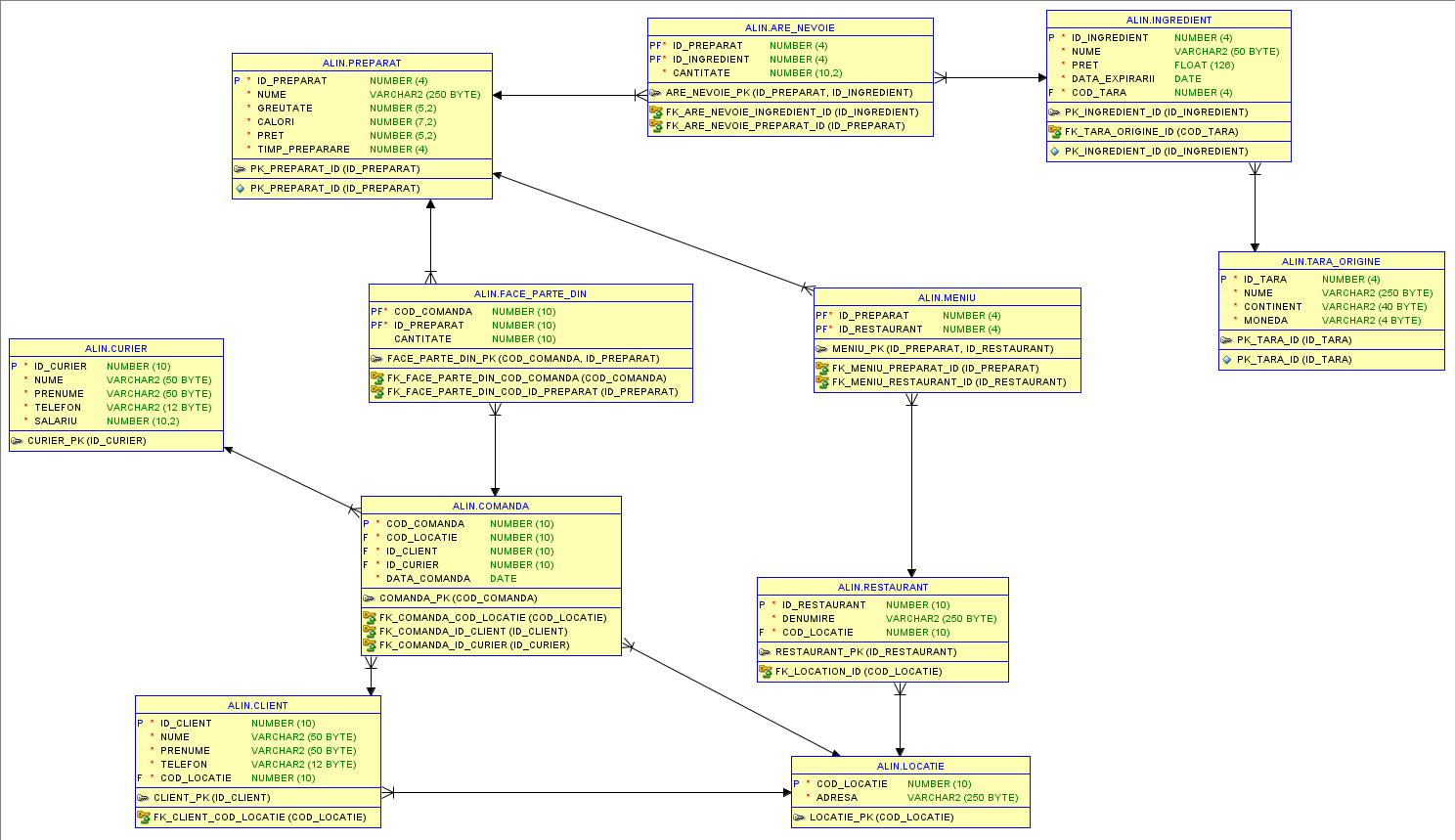
La o anumită locație se pot afla mai multe restaurante/clienți sau se pot livra mai multe comenzi.

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

****

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

****

****

**4 și 5. Implementați în Oracle diagrama conceptuală realizată și adăugarea informațiilor.**

CREATE TABLE PREPARAT(

id\_preparat number(4) CONSTRAINT PK\_PREPARAT\_ID PRIMARY KEY,

nume VARCHAR2(250) NOT NULL,

greutate number(5,2) DEFAULT 0 NOT NULL,

calori number(7,2) DEFAULT 0 NOT NULL,

pret number(5,2) DEFAULT 0 NOT NULL,

timp\_preparare number(4) DEFAULT 0 NOT NULL

);

INSERT INTO preparat

VALUES (1,'Sarmalute in foi de vita',200, 2000, 5.2,120);

INSERT INTO preparat

VALUES (2,'Cartofi copti',250, 2500, 7,30);

INSERT INTO preparat

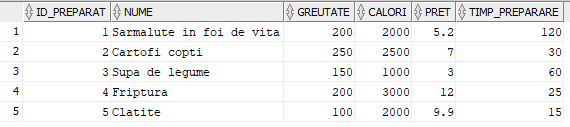
VALUES (3,'Supa de legume',150, 1000, 3 ,60);

INSERT INTO preparat

VALUES (4,'Friptura',200, 3000, 12,25);

INSERT INTO preparat

VALUES (5,'Clatite',100, 2000, 9.90,15);

****

CREATE TABLE TARA\_ORIGINE(

id\_tara number(4) CONSTRAINT PK\_TARA\_ID PRIMARY KEY ,

nume VARCHAR2(250) NOT NULL,

continent VARCHAR2(40) NOT NULL,

moneda VARCHAR(4) NOT NULL

);

INSERT INTO tara\_origine

VALUES (1,'Romania','Europa', 'RON');

INSERT INTO tara\_origine

VALUES (2,'Italia','Europa', 'EUR');

INSERT INTO tara\_origine

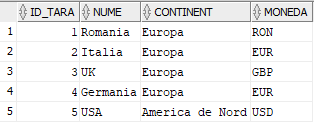
VALUES (3,'UK','Europa', 'GBP');

INSERT INTO tara\_origine

VALUES (4,'Germania','Europa', 'EUR');

INSERT INTO tara\_origine

VALUES (5,'USA','America de Nord', 'USD');



CREATE TABLE INGREDIENT(

id\_ingredient number(4) CONSTRAINT PK\_INGREDIENT\_ID PRIMARY KEY ,

nume VARCHAR2(50) NOT NULL,

pret float DEFAULT 0 NOT NULL,

data\_expirarii date NOT NULL,

cod\_tara number(4) NOT NULL

);

--adaugam cheia externa

--legam tara de origine de ingredient

alter table INGREDIENT

add constraint FK\_TARA\_ORIGINE\_ID

foreign key (cod\_tara) references TARA\_ORIGINE(id\_tara);

INSERT INTO INGREDIENT

VALUES (1,'Sare', 0.5,'10/may/24',5);

INSERT INTO INGREDIENT

VALUES (2,'Otet', 1,'1/may/23',4);

INSERT INTO INGREDIENT

VALUES (3,'Faina', 5,'10/JUN/23',5);

INSERT INTO INGREDIENT

VALUES (4,'Carne', 1,'29/JAN/24',5);

INSERT INTO INGREDIENT

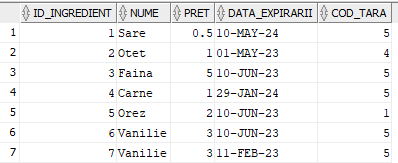
VALUES (5,'Orez', 2,'10/JUN/23',1);

INSERT INTO INGREDIENT

VALUES (6,'Vanilie', 3,'10/JUN/23',5);

INSERT INTO INGREDIENT

VALUES (7,'Vanilie', 3,sysdate+30,5);



--Tabel Asociativ Ingredient-Preparat

--ARE\_NEVOIE

CREATE TABLE ARE\_NEVOIE(

id\_preparat number(4) NOT NULL ,

id\_ingredient number(4) NOT NULL,

cantitate number(10,2) NOT NULL,

PRIMARY KEY(id\_preparat,id\_ingredient)

);

--adaugam cheia externa

--legam ingredientul de origine de tabelul asociativ

alter table ARE\_NEVOIE

add constraint FK\_ARE\_NEVOIE\_INGREDIENT\_ID

foreign key (id\_ingredient) references INGREDIENT(id\_ingredient);

--adaugam cheia externa

--legam ingredientul de origine de tabelul asociativ

alter table ARE\_NEVOIE

add constraint FK\_ARE\_NEVOIE\_PREPARAT\_ID

foreign key (id\_preparat) references PREPARAT(id\_preparat);

INSERT INTO are\_nevoie VALUES(1,1,20);

INSERT INTO are\_nevoie VALUES(2,4,5);

INSERT INTO are\_nevoie VALUES(1,4,6);

INSERT INTO are\_nevoie VALUES(3,2,10);

INSERT INTO are\_nevoie VALUES(2,2,23);

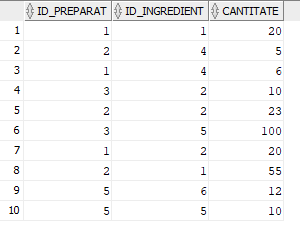
INSERT INTO are\_nevoie VALUES(3,5,100);

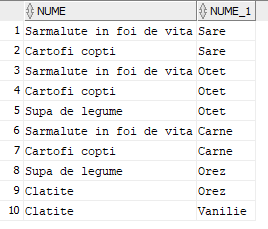
INSERT INTO are\_nevoie VALUES(1,2,20);

INSERT INTO are\_nevoie VALUES(2,1,55);

INSERT INTO are\_nevoie VALUES(5,6,12);

INSERT INTO are\_nevoie VALUES(5,5,10);





---LOCATIE--

CREATE TABLE LOCATIE(

cod\_locatie number(10) ,

adresa VARCHAR2(250) NOT NULL,

PRIMARY KEY(cod\_locatie)

);

INSERT INTO locatie

VALUES(1,'Str. Valea cu Tei, Pitesti');

INSERT INTO locatie

VALUES(2,'Bulevardul I.C. Bratianu, Pitesti');

INSERT INTO locatie

VALUES(3,'Str. Dimitrie Butculescu, Pitesti');

INSERT INTO locatie

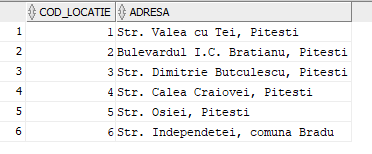
VALUES(4,'Str. Calea Craiovei, Pitesti');

INSERT INTO locatie

VALUES(5,'Str. Osiei, Pitesti');

INSERT INTO locatie

VALUES(6,'Str. Independetei, comuna Bradu');



--RESTAURANT--

CREATE TABLE RESTAURANT(

id\_restaurant number(10) ,

denumire VARCHAR2(250) NOT NULL,

cod\_locatie number(10) NOT NULL,

PRIMARY KEY(id\_restaurant)

);

--adaugam cheia externa pentru locatie

alter table RESTAURANT

add constraint FK\_LOCATION\_ID

foreign key (cod\_locatie) references LOCATIE(cod\_locatie);

INSERT INTO restaurant

VALUES (1,'Pizza Luca Pitesti',2);

INSERT INTO restaurant

VALUES (2,'Atelier',1);

INSERT INTO restaurant

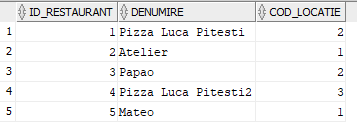
VALUES (3,'Papao',2);

INSERT INTO restaurant

VALUES (4,'Pizza Luca Pitesti2',3);

INSERT INTO restaurant

VALUES (5,'Mateo',1);



--Tabel asociativ restaurant preparat.

--Meniu

CREATE TABLE MENIU(

id\_preparat number(4) NOT NULL ,

id\_restaurant number(4) NOT NULL,

PRIMARY KEY(id\_preparat,id\_restaurant)

);

alter table MENIU

add constraint FK\_MENIU\_PREPARAT\_ID

foreign key (id\_preparat) references PREPARAT(id\_preparat);

alter table MENIU

add constraint FK\_MENIU\_RESTAURANT\_ID

foreign key (id\_restaurant) references RESTAURANT(id\_restaurant);

insert into MENIU VALUES(3, 1);

insert into MENIU VALUES(2, 1);

insert into MENIU VALUES(3, 4);

insert into MENIU VALUES(2, 4);

insert into MENIU VALUES(1, 5);

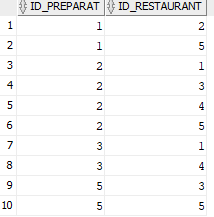
insert into MENIU VALUES(1, 2);

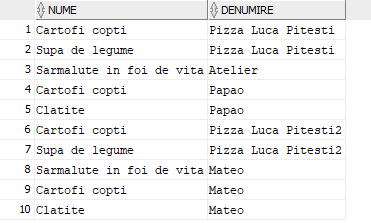
insert into MENIU VALUES( 2, 5);

insert into MENIU VALUES(5, 5);

insert into MENIU VALUES( 2, 3);

insert into MENIU VALUES( 5, 3);





--CURIER--

CREATE TABLE CURIER(

id\_curier number(10) ,

nume VARCHAR2(50) NOT NULL,

prenume VARCHAR2(50) NOT NULL,

telefon VARCHAR2(12) NOT NULL,

salariu number(10,2) NOT NULL,

PRIMARY KEY(id\_curier)

);

INSERT INTO CURIER

VALUES (1,'Dragomirescu','Claudia','0772722321',1500);

INSERT INTO CURIER

VALUES (2,'Popescu','Marcus','0736969697',2000);

INSERT INTO CURIER

VALUES (3,'Ion','Catalin','0736661111',2300.25);

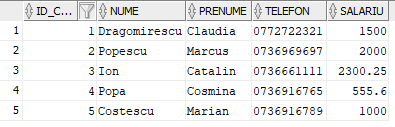
INSERT INTO CURIER

VALUES (4,'Popa','Cosmina','0736916765',555.60);

INSERT INTO CURIER

VALUES (5,'Costescu','Marian','0736916789',1000);

select \* from CURIER;



--CLIENT--

CREATE TABLE CLIENT(

id\_client number(10) ,

nume VARCHAR2(50) CONSTRAINT last\_name\_client\_nn NOT NULL,

prenume VARCHAR2(50) CONSTRAINT fistr\_name\_client\_nn NOT NULL,

telefon VARCHAR2(12) CONSTRAINT telefon\_client\_nn NOT NULL,

cod\_locatie number(10) NOT NULL,

PRIMARY KEY(id\_client)

);

alter table CLIENT

add constraint FK\_CLIENT\_COD\_LOCATIE

foreign key (cod\_locatie) references LOCATIE(cod\_locatie);

INSERT INTO CLIENT

VALUES (1,'Dragomirescu','Claudia','0772744621',1);

INSERT INTO CLIENT

VALUES (2,'Popescu','Catalin','0734469697',5);

INSERT INTO CLIENT

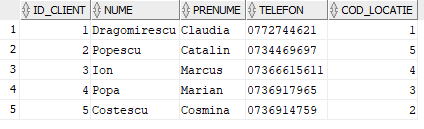
VALUES (3,'Ion','Marcus','07366615611',4);

INSERT INTO CLIENT

VALUES (4,'Popa','Marian','0736917965',3);

INSERT INTO CLIENT

VALUES (5,'Costescu','Cosmina','0736914759',2);



---COMANDA---

CREATE TABLE COMANDA(

cod\_comanda number(10) ,

cod\_locatie number(10) NOT NULL,

id\_client number(10) NOT NULL,

id\_curier number(10) NOT NULL,

data\_comanda date NOT NULL,

PRIMARY KEY(cod\_comanda)

);

alter table COMANDA

add constraint FK\_COMANDA\_COD\_LOCATIE

foreign key (cod\_locatie) references LOCATIE(cod\_locatie);

alter table COMANDA

add constraint FK\_COMANDA\_id\_client

foreign key (id\_client) references CLIENT(id\_client);

alter table COMANDA

add constraint FK\_COMANDA\_id\_curier

foreign key (id\_curier) references CURIER(id\_curier);

INSERT INTO COMANDA

VALUES(1,1,2,1,sysdate-33);

INSERT INTO COMANDA

VALUES(2,2,1,1,sysdate-13);

INSERT INTO COMANDA

VALUES(3,4,2,5,sysdate-39);

INSERT INTO COMANDA

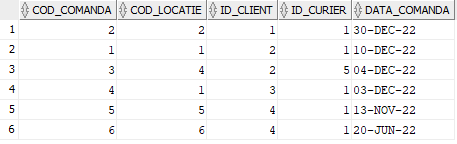
VALUES(4,1,3,1,sysdate-40);

INSERT INTO COMANDA

VALUES(5,5,4,1,sysdate-60);

INSERT INTO COMANDA

VALUES(6,6,4,1,'20-JUN-22');



-- tabela asociativa Preparat-Comanda

CREATE TABLE FACE\_PARTE\_DIN(

cod\_comanda number(10) NOT NULL,

id\_preparat number(10) NOT NULL,

cantitate number(10) default 1,

PRIMARY KEY(cod\_comanda,id\_preparat)

);

alter table FACE\_PARTE\_DIN

add constraint FK\_FACE\_PARTE\_DIN\_COD\_ID\_PREPARAT

foreign key (id\_preparat) references PREPARAT(id\_preparat);

alter table FACE\_PARTE\_DIN

add constraint FK\_FACE\_PARTE\_DIN\_cod\_comanda

foreign key (cod\_comanda) references COMANDA(cod\_comanda);

INSERT INTO FACE\_PARTE\_DIN

VALUES(4,5,1);

INSERT INTO FACE\_PARTE\_DIN

VALUES(1,2,1);

INSERT INTO FACE\_PARTE\_DIN

VALUES(2,1,3);

INSERT INTO FACE\_PARTE\_DIN

VALUES(3,3,1);

INSERT INTO FACE\_PARTE\_DIN

VALUES(3,2,4);

INSERT INTO FACE\_PARTE\_DIN

VALUES(3,5,2);

INSERT INTO FACE\_PARTE\_DIN

VALUES(1,5,1);

INSERT INTO FACE\_PARTE\_DIN

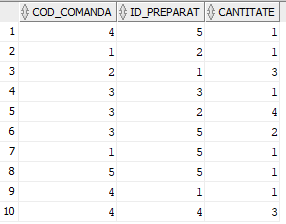
VALUES(5,5,1);

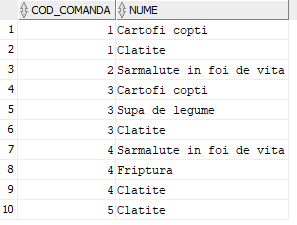
INSERT INTO FACE\_PARTE\_DIN

VALUES(4,1,1);

INSERT INTO FACE\_PARTE\_DIN

VALUES(4,4,3);





**6.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 fiecare client sa se efiseze numele, prenumele si o lista cu toate locatiile la care a dorit livrat comenzi cat si data livrarii

--(precizam ca o comanda este livrata in ziua in care a fost plasata)

--colectii folosite:vector si tablou indexat

create or replace procedure ex\_6

is

type v\_client IS RECORD(id client.id\_client%type,

nume client.nume%type,

prenume client.prenume%type

);

type v\_adresa IS RECORD(adresa locatie.adresa%type,

data\_livrare date

);

TYPE v\_clienti IS TABLE OF v\_client INDEX BY PLS\_INTEGER;

TYPE v\_adrese IS varray(200) OF v\_adresa;

clienti v\_clienti :=v\_clienti();

adrese v\_adrese :=v\_adrese();

sem number(1):=0;

begin

select id\_client, nume, prenume bulk collect into clienti

from client;

for i in clienti.first..clienti.last loop

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

select adresa,data\_comanda bulk collect into adrese

from comanda join locatie using(cod\_locatie)

where id\_client=clienti(i).id;

if adrese.count()>0 then

for j in adrese.first..adrese.last loop

DBMS\_OUTPUT.PUT\_LINE(' ' || adrese(j).adresa ||' la data de:'|| adrese(j).data\_livrare);

end loop;

else

DBMS\_OUTPUT.PUT\_LINE(' acest client nu a a facut nicio comanda.');

end if;

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

end loop;

end ex\_6;

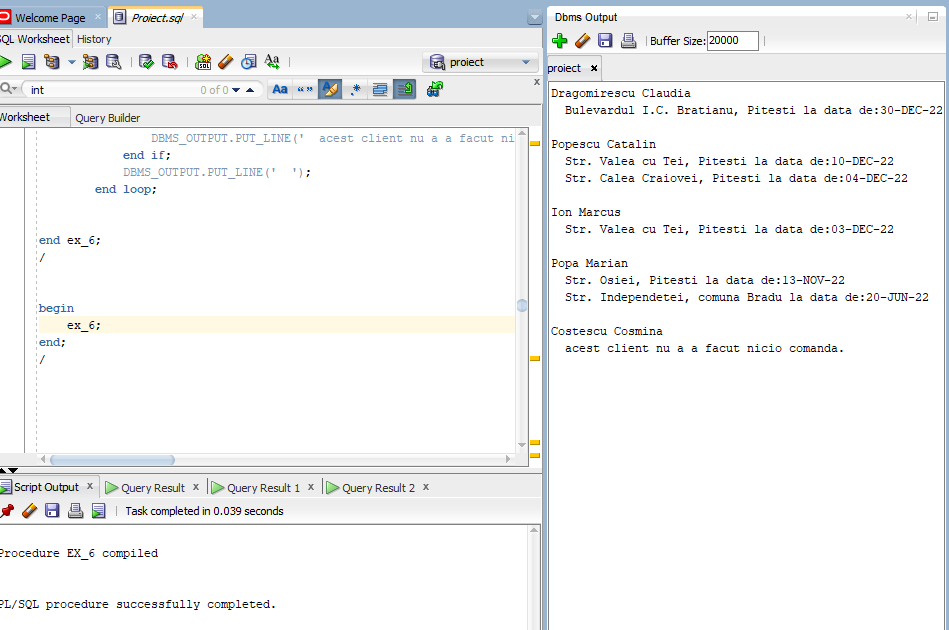
/

begin

ex\_6;

end;

/



**7.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**

--Generati toate meniurile stocate in baza de date, numele restaurantului si locatia sa vor fi specificat o singura data la inceput, iar pentru fiecare preparat se vor afisa si ingredientele.

--Cursoare folosite:ref-cursor si ciclu cursor cu subcerei

create or replace procedure cerinta\_7

is

TYPE refcursor IS REF CURSOR;

cursor preparat\_c( id\_rest meniu.id\_restaurant%type) is

select nume, greutate, calori,pret, cursor (select nume from ingredient join are\_nevoie using(id\_ingredient) where id\_preparat=p.id\_preparat )

from preparat p join meniu m on(p.id\_preparat=m.id\_preparat)

where id\_restaurant=id\_rest;

ingr refcursor;

nume preparat.nume%type;

greutate preparat.greutate%type;

calori preparat.calori%type;

pret preparat.pret%type;

nume\_ingr ingredient.nume%type;

begin

for rest in (select denumire,id\_restaurant,adresa from restaurant join locatie using(cod\_locatie)) loop

DBMS\_OUTPUT.PUT\_LINE ('Restaurantul : ' || rest.denumire || ', aflat la adresa : ' || rest.adresa );

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

open preparat\_c(rest.id\_restaurant);

loop

FETCH preparat\_c INTO nume, greutate, calori, pret, ingr;

exit when preparat\_c %notfound;

DBMS\_OUTPUT.PUT\_LINE ( nume || ' pret: ' || pret || ' calori ' || calori || ' greutate ' || greutate);

loop

FETCH ingr INTO nume\_ingr;

exit when ingr %notfound;

DBMS\_OUTPUT.PUT ( nume\_ingr || ' ' );

end loop;

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

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

end loop;

close preparat\_c;

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

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

end loop;

end cerinta\_7;

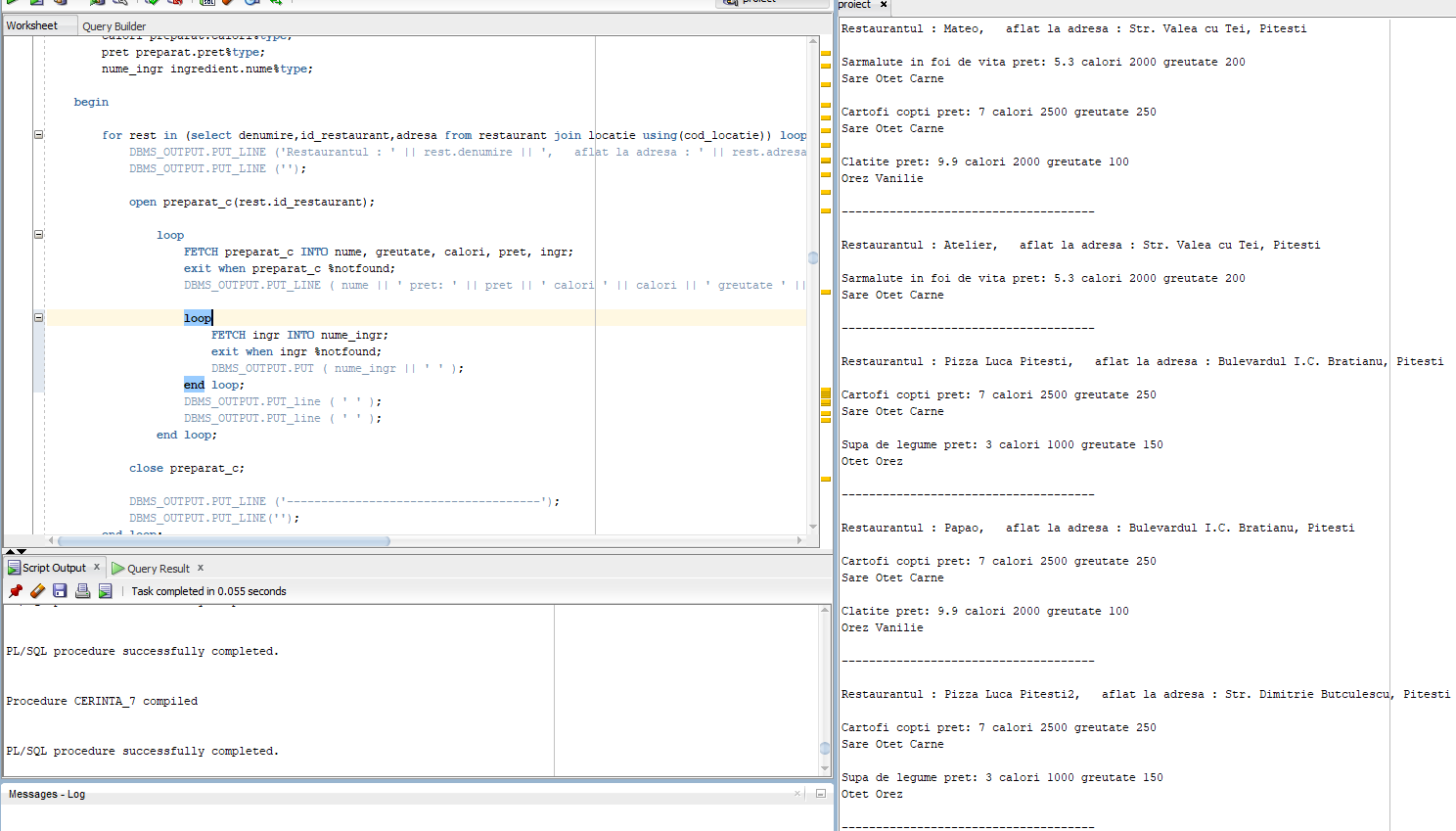
/

begin

cerinta\_7;

end;

/



**8.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**

--Sa se determine id-ul si valoarea celei mai scumpe comenzi livrate de un curier,raportul dintre valoarea determinata si valoarea totala a comenzilor, cat si adresa la care a fost livrata comanda ,

--functia are ca parametru doar prenumele unui curier

--tabele folosite:curier, comanda, locatie, preparat, face\_parte\_din

--inseram date pentru evidentierea erorilor

insert into curier values (6,'test','Marian','0000',0);

insert into curier values (7,'test','test2','0000',0);

insert into curier values (8,'test','test3','0000',0);

INSERT INTO COMANDA VALUES(11,6,4,7,'20-JUN-22');

INSERT INTO COMANDA VALUES(12,6,4,7,'20-JUN-22');

INSERT INTO FACE\_PARTE\_DIN VALUES(11,2,1);

INSERT INTO FACE\_PARTE\_DIN VALUES(12,2,1);

create or replace function cerinta\_8(prenume\_c curier.prenume%type) return varchar2

is

id number(10);

comanda number(10):=-1;

suma\_comanda number(10,2):=0;

adresa locatie.adresa%type;

suma\_totala number(10,2):=0;

counter number(10):=0;

zero\_comenzi exception;

prea\_multe\_comenzi exception;

begin

select id\_curier into id

from curier

where upper(prenume)=upper(prenume\_c);

for i in ( select id\_curier, cod\_comanda ,sum(cantitate \* pret ) as suma, adresa

from comanda join face\_parte\_din using(cod\_comanda)

join preparat using(id\_preparat)

join locatie using(cod\_locatie)

group by cod\_comanda,id\_curier,adresa) loop

suma\_totala:=suma\_totala+i.suma;

if i.id\_curier=id and i.suma>suma\_comanda then

counter:=1;

suma\_comanda:=i.suma;

comanda:=i.cod\_comanda;

adresa:=i.adresa;

elsif i.id\_curier=id and i.suma=suma\_comanda then

counter:=counter+1;

end if;

end loop;

if counter=0 then

raise zero\_comenzi;

elsif counter>1 then

raise prea\_multe\_comenzi;

end if;

return 'comanda: ' || comanda ||', cu valoarea: ' || suma\_comanda || ', reprezinta :'|| round(suma\_comanda\*100/suma\_totala,3) ||'% , livrtata la adresa : ' || adresa ;

exception

when too\_many\_rows then DBMS\_OUTPUT.PUT\_LINE('prea multi curieri cu prenumele :' ||prenume\_c) ;return null;

when no\_data\_found then DBMS\_OUTPUT.PUT\_LINE('nu exista curieri cu prenumele :' ||prenume\_c) ;return null;

when zero\_comenzi then DBMS\_OUTPUT.PUT\_LINE('curierul cu prenumele :' ||prenume\_c || ' nu a livrat comenzi') ;return null;

when prea\_multe\_comenzi then DBMS\_OUTPUT.PUT\_LINE('curierul cu prenumele :' ||prenume\_c || ' a livrat mai multe comenzi cu valoare maxima : '|| suma\_comanda ) ;return null;

end cerinta\_8;

/

begin

DBMS\_OUTPUT.PUT\_LINE('too many rows');

DBMS\_OUTPUT.PUT\_LINE(cerinta\_8('Marian'));

DBMS\_OUTPUT.PUT\_LINE('no data found');

DBMS\_OUTPUT.PUT\_LINE(cerinta\_8('Test'));

DBMS\_OUTPUT.PUT\_LINE('mai multe comenzi cu aceeasi suma');

DBMS\_OUTPUT.PUT\_LINE(cerinta\_8('Test2'));

DBMS\_OUTPUT.PUT\_LINE('curier care nu a livrat comenzi');

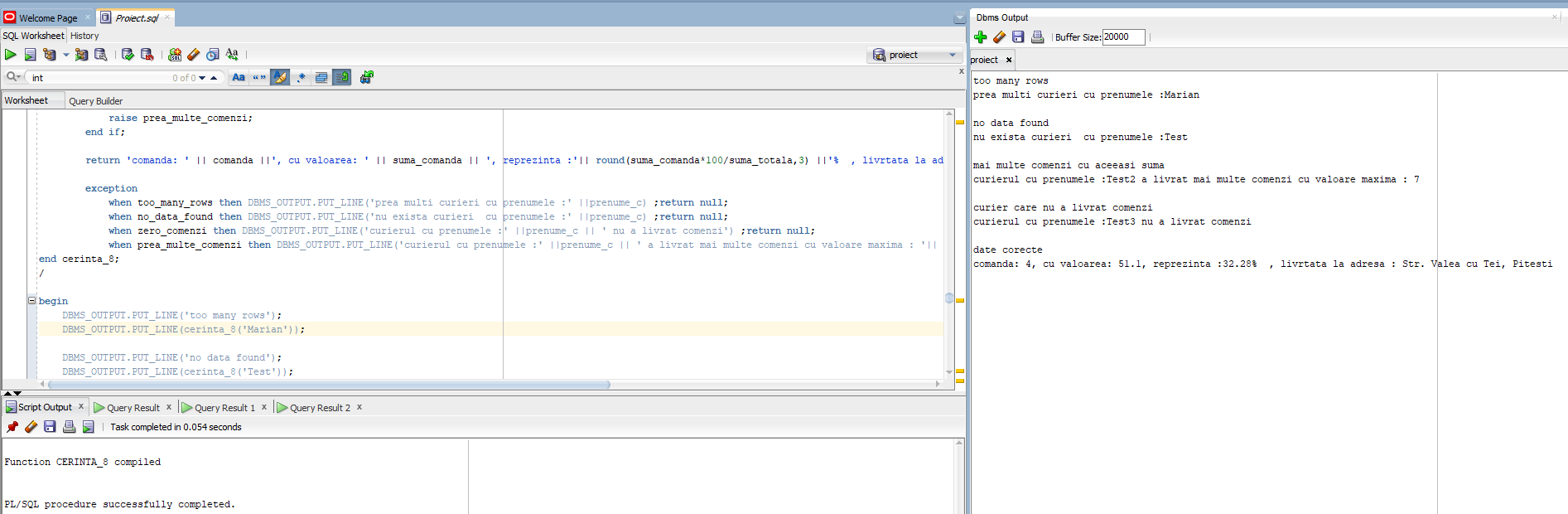
DBMS\_OUTPUT.PUT\_LINE(cerinta\_8('Test3'));

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

DBMS\_OUTPUT.PUT\_LINE(cerinta\_8('Claudia'));

end;

/



**9. 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.**

--Sa se afiseze adresa, numele și nr de telefon al tuturor clienților care au comandat în ultimele 30 de zile un produs care conținea un anumit ingredeint dintr-o anumita tara.Se vor primi ca parametri numele tarii si denumirea ingredientului.

--tabele folosite: tara\_origine, ingredient, preparat, comanda, client, locatie, face\_parte\_din

--vom face insert la noi date pentru evidentierea erorilor.

INSERT INTO INGREDIENT

VALUES (11,'ORZ', 3,sysdate+70,5);

create or replace procedure cerinta\_9 (tara tara\_origine.nume%type,ingr ingredient.nume%type)

is

id\_ingr number(10);

id\_tara number(10);

counter number(10):=0;

INGREDIENT\_NEFOLOSIT exception;

PREPARAT\_NELIVRAT exception;

begin

select id\_ingredient , id\_tara into id\_ingr,id\_tara

from tara\_origine t join ingredient i on(i.cod\_tara=t.id\_tara)

where upper(t.nume)=upper(tara) and upper(i.nume)=upper(ingr) ;

select count(\*) into counter

from are\_nevoie where id\_ingredient = id\_ingr;

if counter=0 then

raise INGREDIENT\_NEFOLOSIT;

end if;

counter:=0;

for i in (select distinct c.nume,c.prenume,telefon,cl.adresa

from preparat join face\_parte\_din using(id\_preparat)

join comanda com using (cod\_comanda)

join client c using (id\_client)

join locatie cl on (c.cod\_locatie=cl.cod\_locatie)

where com.data\_comanda>sysdate-30 and id\_preparat in (select id\_preparat from are\_nevoie where id\_ingredient = id\_ingr )) loop

counter:=counter+1;

DBMS\_OUTPUT.PUT\_LINE ('Client:' || i.nume||' ' || i.prenume || ' cu nr de telefon :' || i.telefon || ' cu adresa: ' ||i.adresa);

end loop;

if counter=0 then

raise PREPARAT\_NELIVRAT;

end if;

exception

when no\_data\_found then DBMS\_OUTPUT.PUT\_LINE ('Nu a fost gasit in baza de date ingredientul: ' || ingr || ' din tara:' || tara);

when too\_many\_rows then DBMS\_OUTPUT.PUT\_LINE ('In baza de date exista de mai multe ori ingredientul : ' || ingr || ' din tara:' || tara);

when INGREDIENT\_NEFOLOSIT then DBMS\_OUTPUT.PUT\_LINE ('Ingredientul : ' || ingr || ' din tara:' || tara || ' nu a fost folosit in niciun preparat');

when PREPARAT\_NELIVRAT then DBMS\_OUTPUT.PUT\_LINE ('Ingredientul : ' || ingr || ' din tara:' || tara || ' nu a fost inclus in niciun preparat livrat in ultimele 30 de zile');

end cerinta\_9;

/

begin

DBMS\_OUTPUT.PUT\_LINE('eroarea de no\_data\_found');

cerinta\_9('usa','muschi');

DBMS\_OUTPUT.PUT\_LINE('too\_many\_rows');

cerinta\_9('usa','vanilie');

DBMS\_OUTPUT.PUT\_LINE('exceptia ingredient nefolosit');

cerinta\_9('usa','orz');

DBMS\_OUTPUT.PUT\_LINE('exceptia PREPARAT\_NELIVRAT');

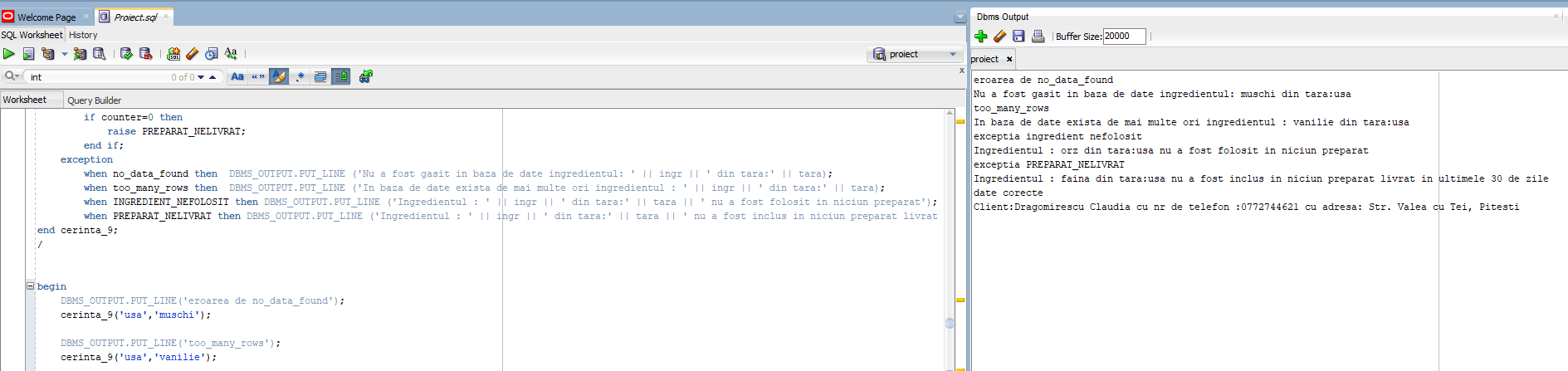
cerinta\_9('usa','sare');

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

cerinta\_9('romania','orez');

end;

/

****

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

--dorim sa creeam un triger care sa perminate angajarea/concediere curierilor doar in zilele de luni si mari in intervalul 8:00-20:00

create or replace trigger ex\_10 before INSERT or DELETE ON curier

BEGIN

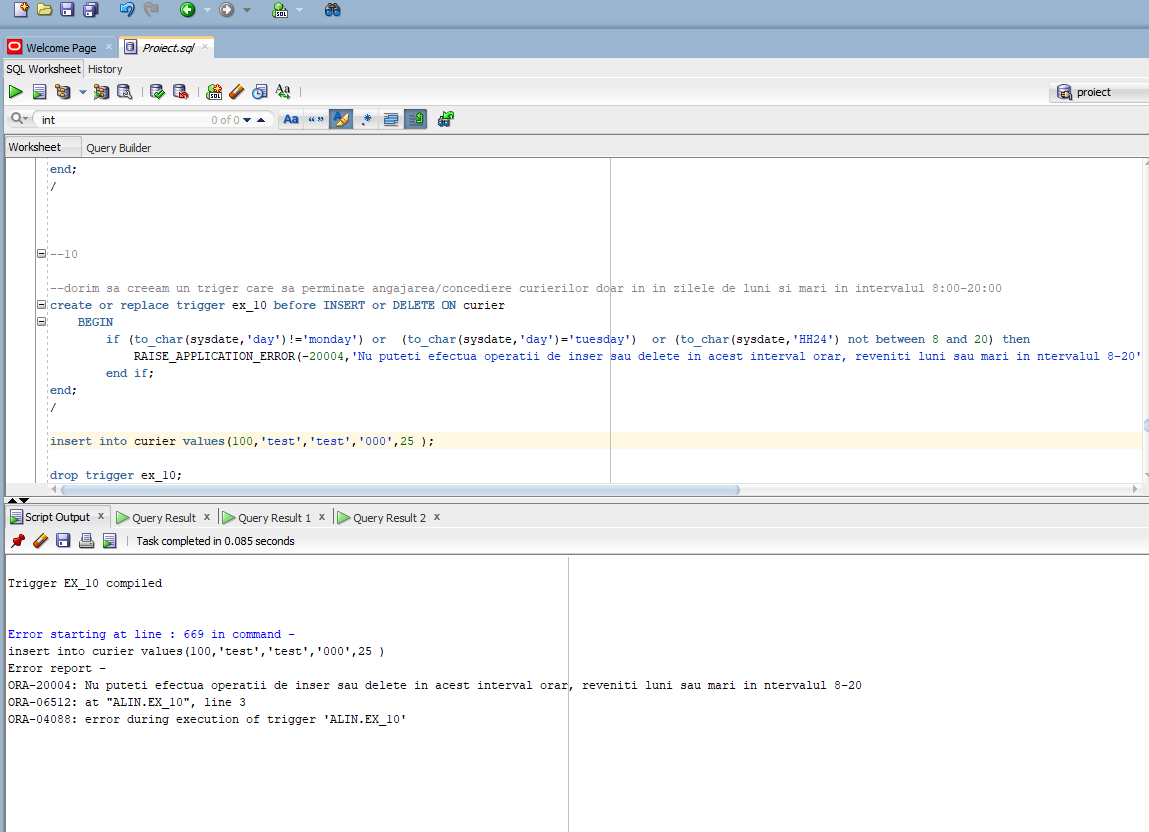
if (to\_char(sysdate,'day')!='monday') or (to\_char(sysdate,'day')='tuesday') or (to\_char(sysdate,'HH24') not between 8 and 20) then

RAISE\_APPLICATION\_ERROR(-20004,'Nu puteti efectua operatii de inser sau delete in acest interval orar, reveniti luni sau mari in intervalul 8-20');

end if;

end;

/



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

--Pretul unui preparat se va modifica in functie de pretul ingredientelor. Vom creea un trigger pentru tabela asociativa dintre preparat

-- si ingredient astfel in cat pretul preparatului sa se modifice in functie de pret\_ingredient \* cantitate/100;

-- un ingredient poate fi adaugat/sters pt un preparat sau i se poate modifica cantitatea

--iar cantitatea noua trebuie sa fie mai mare ca 0;

create or replace trigger ex\_11 before DELETE OR UPDATE OR INSERT ON are\_nevoie

FOR EACH ROW

DECLARE

p\_old number(10,2):=0;

p\_new number(10,2):=0;

BEGIN

if deleting then

select pret into p\_old from ingredient where id\_ingredient = :old.id\_ingredient;

p\_old:= p\_old \* :old.cantitate /100 ;--calculez valoarea pe care trebuie sa o scad.

update preparat set

pret=pret-p\_old

where id\_preparat= :old.id\_preparat;

elsif updating then

if :new.cantitate<=0 then

RAISE\_APPLICATION\_ERROR(-20002,'Cantitatea nu poate fi un nr negativ sau 0');

elsif :old.id\_preparat = :new.id\_preparat and :old.id\_ingredient=:new.id\_ingredient then

select pret into p\_old from ingredient where id\_ingredient = :old.id\_ingredient;

p\_old:= p\_old \* :old.cantitate /100 ;--calculez valoarea pe care trebuie sa o scad.

select pret into p\_new from ingredient where id\_ingredient = :new.id\_ingredient;

p\_new:= p\_new \* :new.cantitate /100 ;--calculez valoarea pe care trebuie sa o adun.

update preparat set

pret=pret + p\_new - p\_old

where id\_preparat= :new.id\_preparat;

else

RAISE\_APPLICATION\_ERROR(-20001,'Nu poti modifica id\_preparat sau id\_ingredient!');

end if;

else

if :new.cantitate<=0 then

RAISE\_APPLICATION\_ERROR(-20002,'Cantitatea nu poate fi un nr negativ sau 0');

end if;

select pret into p\_new from ingredient where id\_ingredient = :new.id\_ingredient;

p\_new:= p\_new \* :new.cantitate /100 ;--calculez valoarea pe care trebuie sa o adun.

update preparat set

pret=pret + p\_new

where id\_preparat= :new.id\_preparat;

end if;

exception

when no\_data\_found then RAISE\_APPLICATION\_ERROR(-20010,'nu exista ingredient cu acest id');

end;

/

--Teste:

--date înaintea modificarilor

select \* from preparat where id\_preparat=1;



delete from are\_nevoie where id\_preparat=1 and id\_ingredient=1;

select \* from preparat where id\_preparat=1;

--date după stergerea unui ingredient



insert into are\_nevoie values(1,1,20);

select \* from preparat where id\_preparat=1;

--inseram ingredientul



update are\_nevoie set

cantitate=40

where id\_preparat=1 and id\_ingredient=1;

select \* from preparat where id\_preparat=1;



--evidentiem faprtul ca nu putem modifica cantitate in 0 sau sa modificăm

--id\_ingredient/id\_preparat

update are\_nevoie set

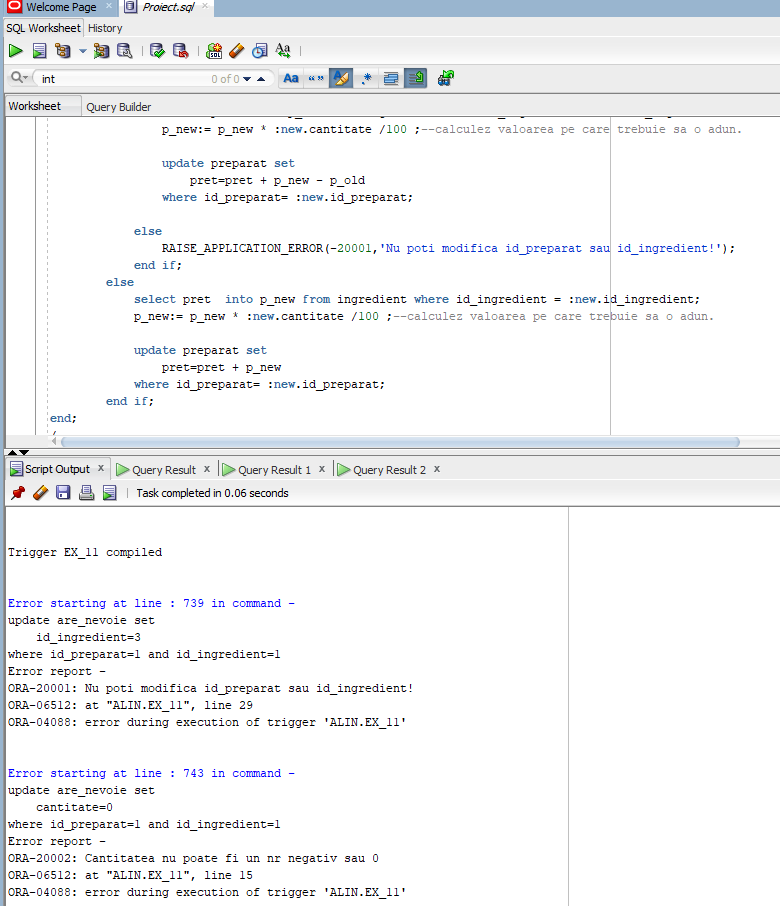
id\_ingredient=3

where id\_preparat=1 and id\_ingredient=1;

update are\_nevoie set

cantitate=0

where id\_preparat=1 and id\_ingredient=1;

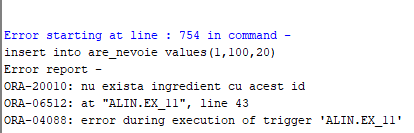


-- daca introducem date incorecte:

insert into are\_nevoie values(1,100,20);

select \* from preparat where id\_preparat=1;

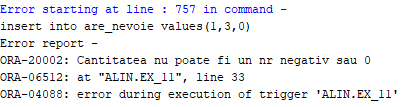




insert into are\_nevoie values(1,3,0);

select \* from preparat where id\_preparat=1;





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

-dorim sa memoram intr o tabela toate operatiile de create alter sau drop facute pe baza de date.

--se vor memora intr-o tabela de istoric

create table istoric(

id number(10),

data\_efectuare\_operatie date,

utilizator varchar(200),

operatie varchar(30),

tip\_obiect varchar(200),

nume\_obiect varchar(200),

PRIMARY KEY(id)

);

/

create or replace trigger istoric\_operati AFTER CREATE OR ALTER OR DROP ON SCHEMA

DECLARE

BEGIN

INSERT INTO istoric VALUES( (select count(\*)+1 from istoric), --id generat automat

sysdate,

user,

SYS.SYSEVENT,

SYS.DICTIONARY\_OBJ\_TYPE,

SYS.DICTIONARY\_OBJ\_NAME);

END;

/

– teste care arată funcționare trigger-ului

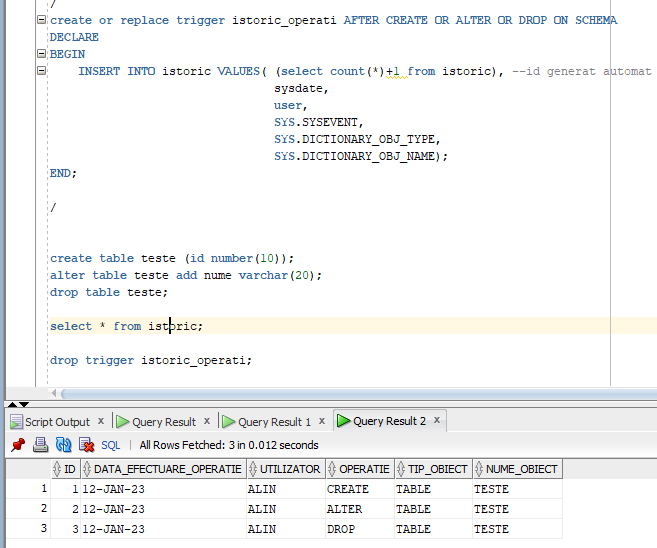
create table teste (id number(10));

alter table teste add nume varchar(20);

drop table teste;

select \* from istoric;

drop trigger istoric\_operati;



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

CREATE OR REPLACE PACKAGE ex\_13 AS

procedure ex\_6 ;

procedure cerinta\_7 ;

function cerinta\_8(prenume\_c curier.prenume%type) return varchar2 ;

procedure cerinta\_9 (tara tara\_origine.nume%type,ingr ingredient.nume%type);

end ex\_13;

/

create or replace package body ex\_13 as

procedure ex\_6

is

type v\_client IS RECORD(id client.id\_client%type,

nume client.nume%type,

prenume client.prenume%type

);

type v\_adresa IS RECORD(adresa locatie.adresa%type,

data\_livrare date

);

TYPE v\_clienti IS TABLE OF v\_client INDEX BY PLS\_INTEGER;

TYPE v\_adrese IS varray(200) OF v\_adresa;

clienti v\_clienti :=v\_clienti();

adrese v\_adrese :=v\_adrese();

sem number(1):=0;

begin

select id\_client, nume, prenume bulk collect into clienti

from client;

for i in clienti.first..clienti.last loop

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

select adresa,data\_comanda bulk collect into adrese

from comanda join locatie using(cod\_locatie)

where id\_client=clienti(i).id;

if adrese.count()>0 then

for j in adrese.first..adrese.last loop

DBMS\_OUTPUT.PUT\_LINE(' ' || adrese(j).adresa ||' la data de:'|| adrese(j).data\_livrare);

end loop;

else

DBMS\_OUTPUT.PUT\_LINE(' acest client nu a a facut nicio comanda.');

end if;

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

end loop;

end ex\_6;

procedure cerinta\_7

is

TYPE refcursor IS REF CURSOR;

cursor preparat\_c( id\_rest meniu.id\_restaurant%type) is

select nume, greutate, calori,pret, cursor (select nume from ingredient join are\_nevoie using(id\_ingredient) where id\_preparat=p.id\_preparat )

from preparat p join meniu m on(p.id\_preparat=m.id\_preparat)

where id\_restaurant=id\_rest;

ingr refcursor;

nume preparat.nume%type;

greutate preparat.greutate%type;

calori preparat.calori%type;

pret preparat.pret%type;

nume\_ingr ingredient.nume%type;

begin

for rest in (select denumire,id\_restaurant,adresa from restaurant join locatie using(cod\_locatie)) loop

DBMS\_OUTPUT.PUT\_LINE ('Restaurantul : ' || rest.denumire || ', aflat la adresa : ' || rest.adresa );

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

open preparat\_c(rest.id\_restaurant);

loop

FETCH preparat\_c INTO nume, greutate, calori, pret, ingr;

exit when preparat\_c %notfound;

DBMS\_OUTPUT.PUT\_LINE ( nume || ' pret: ' || pret || ' calori ' || calori || ' greutate ' || greutate);

loop

FETCH ingr INTO nume\_ingr;

exit when ingr %notfound;

DBMS\_OUTPUT.PUT ( nume\_ingr || ' ' );

end loop;

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

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

end loop;

close preparat\_c;

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

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

end loop;

end cerinta\_7;

function cerinta\_8(prenume\_c curier.prenume%type) return varchar2

is

id number(10);

comanda number(10):=-1;

suma\_comanda number(10,2):=0;

adresa locatie.adresa%type;

suma\_totala number(10,2):=0;

counter number(10):=0;

zero\_comenzi exception;

prea\_multe\_comenzi exception;

begin

select id\_curier into id

from curier

where upper(prenume)=upper(prenume\_c);

for i in ( select id\_curier, cod\_comanda ,sum(cantitate \* pret ) as suma, adresa

from comanda join face\_parte\_din using(cod\_comanda)

join preparat using(id\_preparat)

join locatie using(cod\_locatie)

group by cod\_comanda,id\_curier,adresa) loop

suma\_totala:=suma\_totala+i.suma;

if i.id\_curier=id and i.suma>suma\_comanda then

counter:=1;

suma\_comanda:=i.suma;

comanda:=i.cod\_comanda;

adresa:=i.adresa;

elsif i.id\_curier=id and i.suma=suma\_comanda then

counter:=counter+1;

end if;

end loop;

if counter=0 then

raise zero\_comenzi;

elsif counter>1 then

raise prea\_multe\_comenzi;

end if;

return 'comanda: ' || comanda ||', cu valoarea: ' || suma\_comanda || ', reprezinta :'|| round(suma\_comanda\*100/suma\_totala,3) ||'% , livrtata la adresa : ' || adresa ;

exception

when too\_many\_rows then DBMS\_OUTPUT.PUT\_LINE('prea multi curieri cu prenumele :' ||prenume\_c) ;return null;

when no\_data\_found then DBMS\_OUTPUT.PUT\_LINE('nu exista curieri cu prenumele :' ||prenume\_c) ;return null;

when zero\_comenzi then DBMS\_OUTPUT.PUT\_LINE('curierul cu prenumele :' ||prenume\_c || ' nu a livrat comenzi') ;return null;

when prea\_multe\_comenzi then DBMS\_OUTPUT.PUT\_LINE('curierul cu prenumele :' ||prenume\_c || ' a livrat mai multe comenzi cu valoare maxima : '|| suma\_comanda ) ;return null;

end cerinta\_8;

procedure cerinta\_9 (tara tara\_origine.nume%type,ingr ingredient.nume%type)

is

id\_ingr number(10);

id\_tara number(10);

counter number(10):=0;

INGREDIENT\_NEFOLOSIT exception;

PREPARAT\_NELIVRAT exception;

begin

select id\_ingredient , id\_tara into id\_ingr,id\_tara

from tara\_origine t join ingredient i on(i.cod\_tara=t.id\_tara)

where upper(t.nume)=upper(tara) and upper(i.nume)=upper(ingr) ;

select count(\*) into counter

from are\_nevoie where id\_ingredient = id\_ingr;

if counter=0 then

raise INGREDIENT\_NEFOLOSIT;

end if;

counter:=0;

for i in (select distinct c.nume,c.prenume,telefon,cl.adresa

from preparat join face\_parte\_din using(id\_preparat)

join comanda com using (cod\_comanda)

join client c using (id\_client)

join locatie cl on (c.cod\_locatie=cl.cod\_locatie)

where com.data\_comanda>sysdate-30 and id\_preparat in (select id\_preparat from are\_nevoie where id\_ingredient = id\_ingr )) loop

counter:=counter+1;

DBMS\_OUTPUT.PUT\_LINE ('Client:' || i.nume||' ' || i.prenume || ' cu nr de telefon :' || i.telefon || ' cu adresa: ' ||i.adresa);

end loop;

if counter=0 then

raise PREPARAT\_NELIVRAT;

end if;

exception

when no\_data\_found then DBMS\_OUTPUT.PUT\_LINE ('Nu a fost gasit in baza de date ingredientul: ' || ingr || ' din tara:' || tara);

when too\_many\_rows then DBMS\_OUTPUT.PUT\_LINE ('In baza de date exista de mai multe ori ingredientul : ' || ingr || ' din tara:' || tara);

when INGREDIENT\_NEFOLOSIT then DBMS\_OUTPUT.PUT\_LINE ('Ingredientul : ' || ingr || ' din tara:' || tara || ' nu a fost folosit in niciun preparat');

when PREPARAT\_NELIVRAT then DBMS\_OUTPUT.PUT\_LINE ('Ingredientul : ' || ingr || ' din tara:' || tara || ' nu a fost inclus in niciun preparat livrat in ultimele 30 de zile');

end cerinta\_9;

end ex\_13;

/

begin

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

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

DBMS\_OUTPUT.PUT\_LINE('exercitiu 6');

ex\_13.ex\_6;

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

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

DBMS\_OUTPUT.PUT\_LINE('exercitiu 7');

ex\_13.cerinta\_7;

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

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

DBMS\_OUTPUT.PUT\_LINE('exercitiu 8');

DBMS\_OUTPUT.PUT\_LINE('too many rows');

DBMS\_OUTPUT.PUT\_LINE(ex\_13.cerinta\_8('Marian'));

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

DBMS\_OUTPUT.PUT\_LINE('no data found');

DBMS\_OUTPUT.PUT\_LINE(ex\_13.cerinta\_8('Test'));

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

DBMS\_OUTPUT.PUT\_LINE('mai multe comenzi cu aceeasi suma');

DBMS\_OUTPUT.PUT\_LINE(ex\_13.cerinta\_8('Test2'));

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

DBMS\_OUTPUT.PUT\_LINE('curier care nu a livrat comenzi');

DBMS\_OUTPUT.PUT\_LINE(ex\_13.cerinta\_8('Test3'));

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

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

DBMS\_OUTPUT.PUT\_LINE(ex\_13.cerinta\_8('Claudia'));

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

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

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

DBMS\_OUTPUT.PUT\_LINE('exercitiu 9');

DBMS\_OUTPUT.PUT\_LINE('eroarea de no\_data\_found');

ex\_13.cerinta\_9('usa','muschi');

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

DBMS\_OUTPUT.PUT\_LINE('too\_many\_rows');

ex\_13.cerinta\_9('usa','vanilie');

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

DBMS\_OUTPUT.PUT\_LINE('exceptia ingredient nefolosit');

ex\_13.cerinta\_9('usa','orz');

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

DBMS\_OUTPUT.PUT\_LINE('exceptia PREPARAT\_NELIVRAT');

ex\_13.cerinta\_9('usa','faina');

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

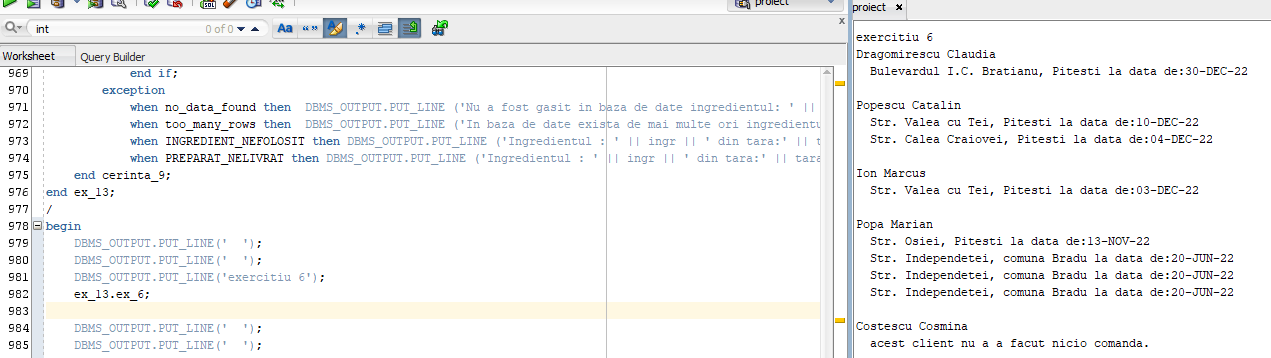
DBMS\_OUTPUT.PUT\_LINE('date corecte');

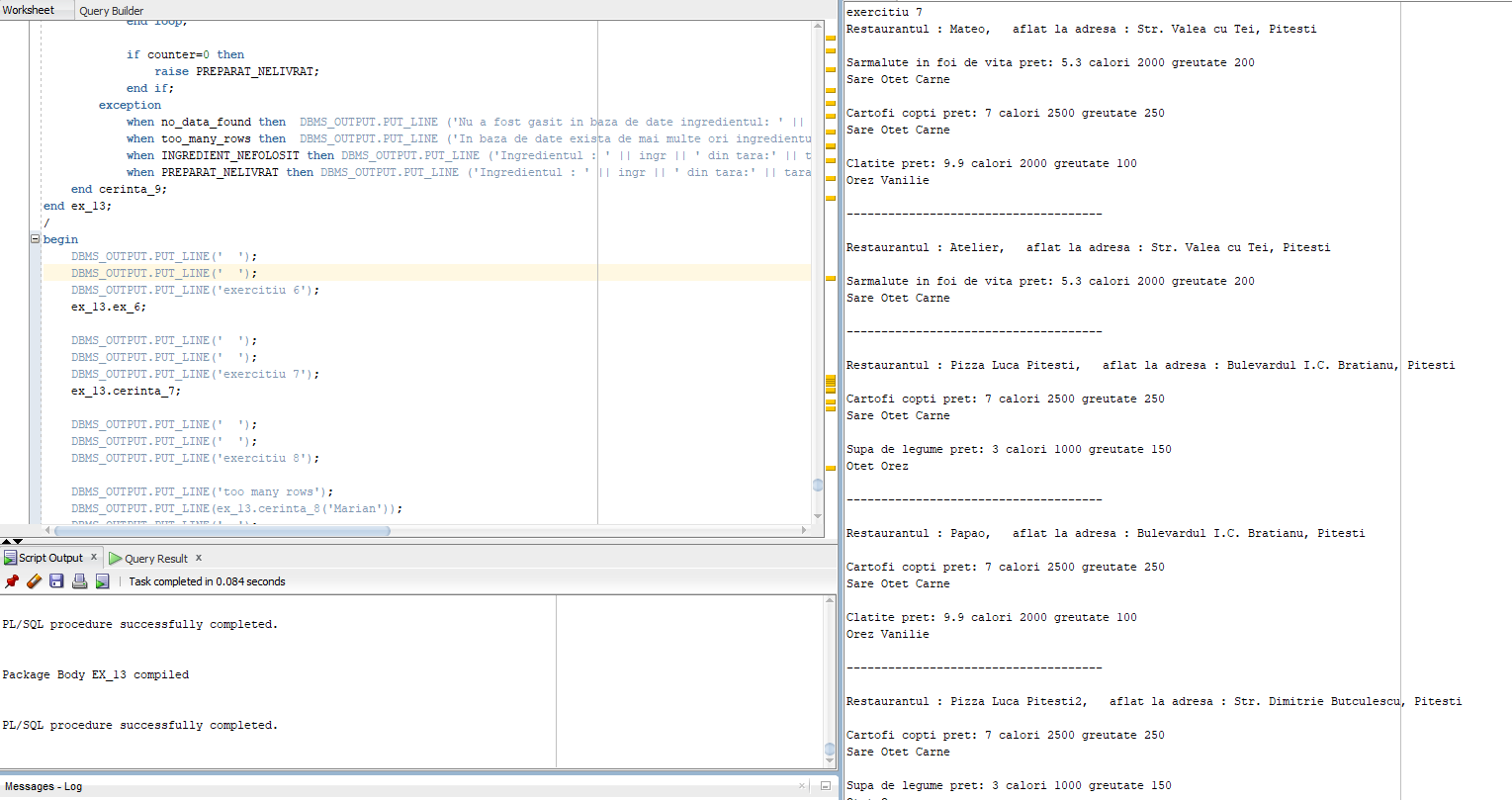
ex\_13.cerinta\_9('USA','sare');

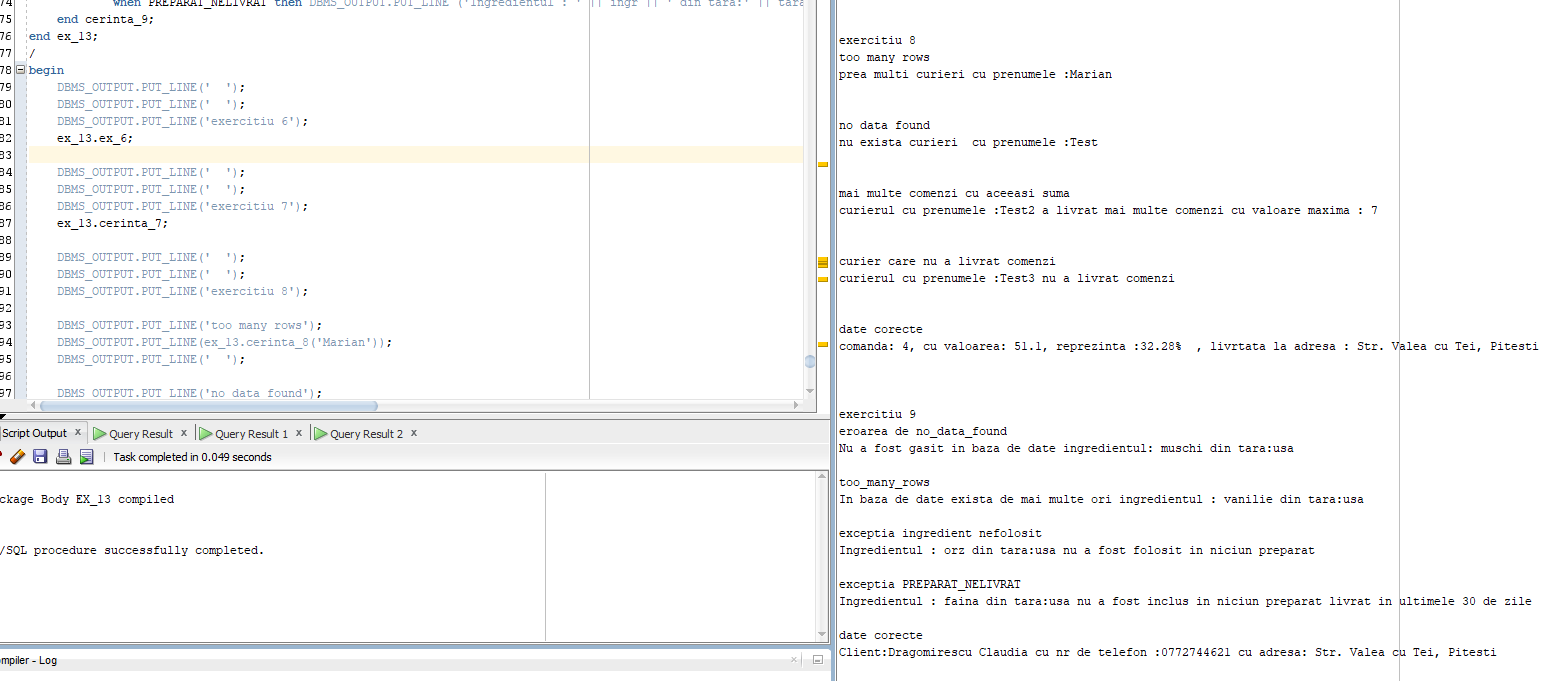
DBMS\_OUTPUT.PUT\_LINE(' ');

end;

/

****

****

****