**Baza de date a unui lant de pizzerii**

Realizat de: Suditu Darius-Cosmin

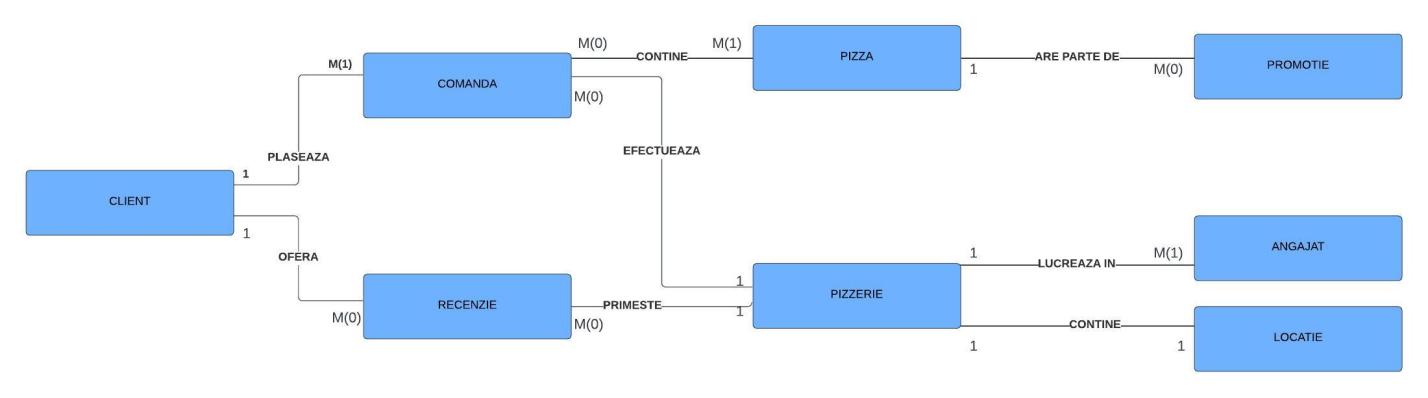
1. Prezentați pe scurt baza de date (utilitatea ei).

Pentru acest proiect, am ales realizarea bazei de date a unui lant de pizzerii, ce are mai multe locatii in diferite orase precum Bucuresti, Constanta, Cluj. Clientii pot comanda diverse preparate din meniu, acest lucru realizandu-se cu ajutorul angajatilor, care au diferite specializari, precum bucatari, ospatari sau menajere. Totodata, clientii pot oferi recenzii in functie de cat de satisfacuti au fost de cele comandate si primi diferite promotii avantajoase. Utilitatea principala a acestei baze de date este gestionarea optima a informatiilor referitoare la clienti, comenzi, angajati si preparate in cadrul lantului de pizzerii. Ea permite inregistrarea si stocarea datelor relevante pentru functionarea eficienta a afacerii.

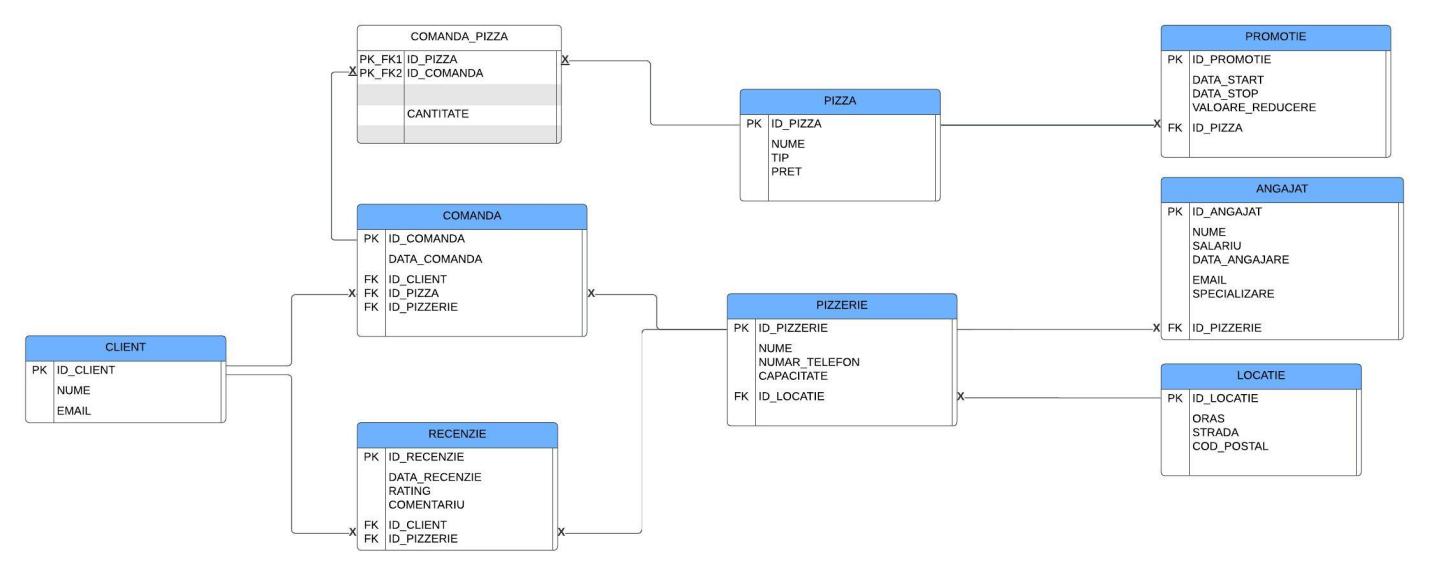
Restrictii si reguli:

* Fiecare angajat lucreaza intr-o singura locatie, iar fiecare locatie are cel putin un angajat. Angajatul este identifiabil prin ID.
* Un client trebuie sa fi plasat cel putin o comanda.
* Comenzile sunt unice si pot fi identificate printr-un ID.
* Pizzeriile se afla in diverse locatii din diferite orase, identifiabile prin ID.
* Salariul angajatilor este direct proportional cu experienta lor.
* Clientii pot lasa recenzii unice ce pot fi identificate printr-un ID.
* Pizzelor le sunt aplicate promotii in functie de ofertele curente.
* Preparatele sunt aceleasi pentru fiecare pizzerie in parte si pot fi identificate printr-un ID.

1. Realizați diagrama entitate-relație (ERD).



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



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

CREATE TABLE Pizza

(

ID\_Pizza VARCHAR2(10) NOT NULL,

Nume VARCHAR2(20) NOT NULL,

Tip VARCHAR2(20) NOT NULL,

Pret int not null,

PRIMARY KEY (ID\_Pizza)

);

CREATE TABLE Locatie

(

ID\_Locatie VARCHAR2(10) NOT NULL,

Oras VARCHAR2(20) NOT NULL,

Strada VARCHAR2(30),

Cod\_postal VARCHAR2(10) NOT NULL,

PRIMARY KEY (ID\_Locatie)

);

CREATE TABLE Pizzerie

(

ID\_PIZZERIE VARCHAR2(20) NOT NULL,

Nume VARCHAR2(10) NOT NULL,

numar\_telefon VARCHAR2(10),

Capacitate int not null,

ID\_Locatie VARCHAR2(10) NOT NULL,

PRIMARY KEY (ID\_Pizzerie),

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

);

CREATE TABLE Client

(

ID\_Client INT NOT NULL,

Nume VARCHAR2(20) NOT NULL,

Email VARCHAR2(30),

PRIMARY KEY (ID\_Client)

);

create TABLE Angajat(

ID\_Angajat int not null,

Nume VARCHAR2(50) not null,

Specializare VARCHAR2(20) not null,

Salariu int not null,

ID\_Pizzerie VARCHAR2(10) NOT NULL,

Data\_Angajare date not null,

Email VARCHAR2(50),

PRIMARY KEY (ID\_Angajat),

FOREIGN KEY (ID\_Pizzerie) REFERENCES Pizzerie(ID\_Pizzerie)

);

create TABLE Recenzie(

ID\_Recenzie int not null primary key,

Data\_Recenzie date,

Rating int not null,

Comentariu VARCHAR2(50) NOT NULL,

ID\_Client int not null,

ID\_Pizzerie VARCHAR2(10) not null,

foreign key(ID\_Client) references Client(ID\_Client),

foreign key(ID\_Pizzerie) references Pizzerie(ID\_Pizzerie)

);

create TABLE Promotie(

ID\_Promotie int not null primary key,

Data\_Start date,

Data\_Stop date,

Valoare\_Reducere int not null,

ID\_Pizza VARCHAR2(10) not null,

foreign key(ID\_Pizza) references Pizza(ID\_Pizza)

);

CREATE TABLE Comanda

(

ID\_Comanda INT NOT NULL,

Data\_Comanda date,

ID\_Pizzerie VARCHAR2(10) NOT NULL,

ID\_Client INT NOT NULL,

PRIMARY KEY (ID\_Comanda),

FOREIGN KEY (ID\_Pizzerie) REFERENCES Pizzerie(ID\_Pizzerie),

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

);

CREATE TABLE COMANDA\_PIZZA (

ID\_Pizza VARCHAR2(10) NOT NULL,

ID\_Comanda INT NOT NULL,

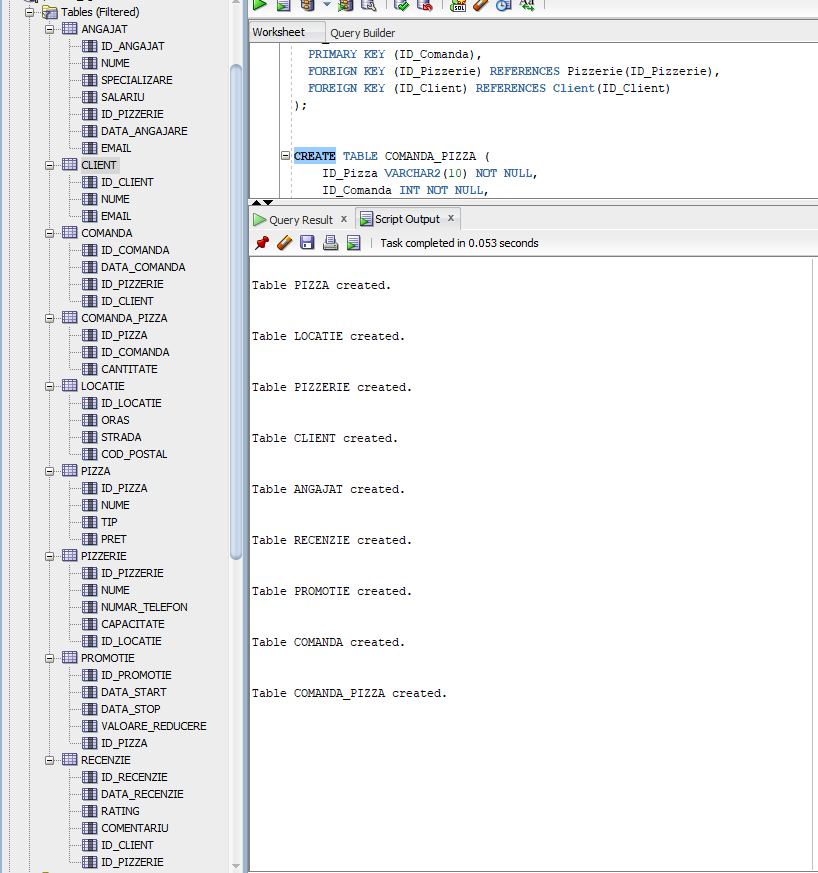
Cantitate INT NOT NULL,

PRIMARY KEY (ID\_Pizza, ID\_Comanda),

FOREIGN KEY (ID\_Pizza) REFERENCES Pizza(ID\_Pizza),

FOREIGN KEY (ID\_Comanda) REFERENCES Comanda(ID\_Comanda)

);



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

insert into Client (ID\_Client, nume, email) VALUES (1, 'Ana Popescu', 'ana.popescu@example.com');

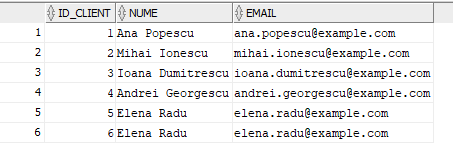
insert into Client (ID\_Client, nume, email) VALUES(2, 'Mihai Ionescu', 'mihai.ionescu@example.com');

insert into Client (ID\_Client, nume, email) VALUES(3, 'Ioana Dumitrescu', 'ioana.dumitrescu@example.com');

insert into Client (ID\_Client, nume, email) VALUES(4, 'Andrei Georgescu', 'andrei.georgescu@example.com');

insert into Client (ID\_Client, nume, email) VALUES(5, 'Elena Radu', 'elena.radu@example.com');

insert into Client (ID\_Client, nume, email) VALUES(6, 'Elena Radu', ['elena.radu@example.com');](mailto:'elena.radu@example.com');)



insert into Locatie (ID\_Locatie, Oras, Strada, Cod\_postal) VALUES('RO001', 'Bucharest', 'Nordului', '010101');

insert into Locatie (ID\_Locatie, Oras, Strada, Cod\_postal) VALUES('RO002', 'Cluj-Napoca', 'Gheorghe Titeica', '020202');

insert into Locatie (ID\_Locatie, Oras, Strada, Cod\_postal) VALUES('RO003', 'Timisoara', 'Brailita', '030303');

insert into Locatie (ID\_Locatie, Oras, Strada, Cod\_postal) VALUES('RO004', 'Iasi', 'Mugur Mugurel', '040404');

insert into Locatie (ID\_Locatie, Oras, Strada, Cod\_postal) VALUES('RO005', 'Constanta', 'Poenaru', '050505');

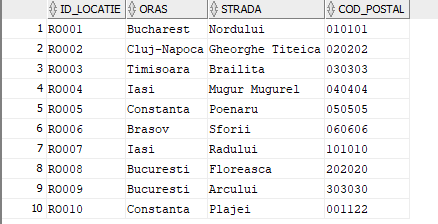
insert into Locatie (ID\_Locatie, Oras, Strada, Cod\_postal) VALUES('RO006', 'Brasov', 'Sforii', '060606');

insert into Locatie (ID\_Locatie, Oras, Strada, Cod\_postal) VALUES('RO007','Iasi', 'Radului', '101010');

insert into Locatie (ID\_Locatie, Oras, Strada, Cod\_postal) VALUES('RO008', 'Bucuresti', 'Floreasca', '202020');

insert into Locatie (ID\_Locatie, Oras, Strada, Cod\_postal) VALUES('RO009', 'Bucuresti', 'Arcului', '303030');

insert into Locatie (ID\_Locatie, Oras, Strada, Cod\_postal) VALUES('RO010', 'Constanta', 'Plajei', '001122');



insert into Pizzerie (ID\_Pizzerie, Nume, numar\_telefon,Capacitate, ID\_Locatie) VALUES ('PIZZ001', 'Pizzerie1', '1234567890',88, 'RO001');

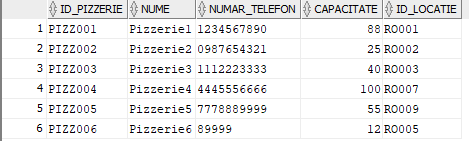
insert into Pizzerie (ID\_Pizzerie, Nume, numar\_telefon,Capacitate, ID\_Locatie) VALUES('PIZZ002', 'Pizzerie2', '0987654321',25, 'RO002');

insert into Pizzerie (ID\_Pizzerie, Nume, numar\_telefon,Capacitate, ID\_Locatie) VALUES('PIZZ003', 'Pizzerie3', '1112223333',40, 'RO003');

insert into Pizzerie (ID\_Pizzerie, Nume, numar\_telefon,Capacitate, ID\_Locatie) VALUES('PIZZ004', 'Pizzerie4', '4445556666',100, 'RO007');

insert into Pizzerie (ID\_Pizzerie, Nume, numar\_telefon,Capacitate, ID\_Locatie) VALUES('PIZZ005', 'Pizzerie5', '7778889999',55, 'RO009');

insert into Pizzerie (ID\_Pizzerie, Nume, numar\_telefon,Capacitate, ID\_Locatie) VALUES('PIZZ006', 'Pizzerie6', '89999',12, 'RO005');



INSERT INTO Angajat (ID\_Angajat, Nume, Specializare, Salariu, ID\_Pizzerie, Data\_Angajare, Email) VALUES (1, 'Ion Popescu', 'Buc?tar', 3000, 'PIZZ001', TO\_DATE('2023-01-10', 'YYYY-MM-DD'), 'ion.p@pizzerie1.ro');

INSERT INTO Angajat (ID\_Angajat, Nume, Specializare, Salariu, ID\_Pizzerie, Data\_Angajare, Email) VALUES (2, 'Maria Ionescu', 'Menajera', 2500, 'PIZZ001', TO\_DATE('2023-02-15', 'YYYY-MM-DD'), 'maria.i@pizzerie1.ro');

INSERT INTO Angajat (ID\_Angajat, Nume, Specializare, Salariu, ID\_Pizzerie, Data\_Angajare, Email) VALUES (3, 'Alexandru Georgescu', 'Ospatar', 2200, 'PIZZ001', TO\_DATE('2023-03-20', 'YYYY-MM-DD'), 'alex.g@pizzerie1.ro');

INSERT INTO Angajat (ID\_Angajat, Nume, Specializare, Salariu, ID\_Pizzerie, Data\_Angajare, Email) VALUES (4, 'Andrei Popa', 'Buc?tar', 3200, 'PIZZ002', TO\_DATE('2023-01-12', 'YYYY-MM-DD'), 'andrei.p@pizzerie2.ro');

INSERT INTO Angajat (ID\_Angajat, Nume, Specializare, Salariu, ID\_Pizzerie, Data\_Angajare, Email) VALUES (5, 'Elena Stoica', 'Menajera', 2400, 'PIZZ002', TO\_DATE('2023-03-18', 'YYYY-MM-DD'), 'elena.s@pizzerie2.ro');

INSERT INTO Angajat (ID\_Angajat, Nume, Specializare, Salariu, ID\_Pizzerie, Data\_Angajare, Email) VALUES (6, 'Mihai Radu', 'Ospatar', 2100, 'PIZZ002', TO\_DATE('2023-04-25', 'YYYY-MM-DD'), 'mihai.r@pizzerie2.ro');

INSERT INTO Angajat (ID\_Angajat, Nume, Specializare, Salariu, ID\_Pizzerie, Data\_Angajare, Email) VALUES (7, 'Gabriel Dumitru', 'Buc?tar', 3300, 'PIZZ003', TO\_DATE('2023-02-01', 'YYYY-MM-DD'), 'gabriel.d@pizzerie3.ro');

INSERT INTO Angajat (ID\_Angajat, Nume, Specializare, Salariu, ID\_Pizzerie, Data\_Angajare, Email) VALUES (8, 'Ana Matei', 'Menajera', 2600, 'PIZZ003', TO\_DATE('2023-03-10', 'YYYY-MM-DD'), 'ana.m@pizzerie3.ro');

INSERT INTO Angajat (ID\_Angajat, Nume, Specializare, Salariu, ID\_Pizzerie, Data\_Angajare, Email) VALUES (9, 'George Marinescu', 'Ospatar', 2300, 'PIZZ003', TO\_DATE('2023-05-30', 'YYYY-MM-DD'), 'george.m@pizzerie3.ro');

INSERT INTO Angajat (ID\_Angajat, Nume, Specializare, Salariu, ID\_Pizzerie, Data\_Angajare, Email) VALUES (10, 'Florin Vasilescu', 'Buc?tar', 3400, 'PIZZ004', TO\_DATE('2023-01-20', 'YYYY-MM-DD'), 'florin.v@pizzerie4.ro');

INSERT INTO Angajat (ID\_Angajat, Nume, Specializare, Salariu, ID\_Pizzerie, Data\_Angajare, Email) VALUES (11, 'Cristina Tudor', 'Menajera', 2550, 'PIZZ004', TO\_DATE('2023-02-28', 'YYYY-MM-DD'), 'cristina.t@pizzerie4.ro');

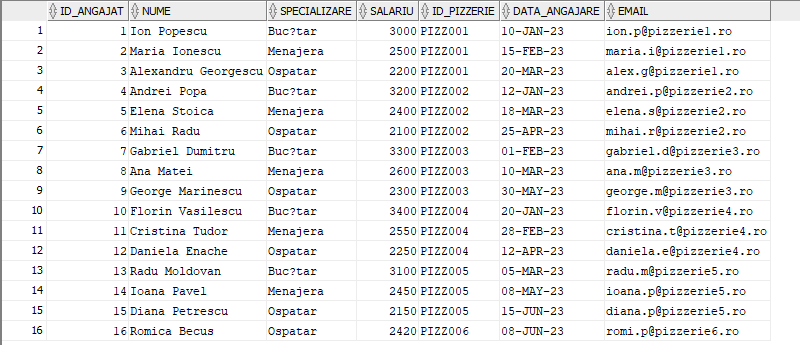
INSERT INTO Angajat (ID\_Angajat, Nume, Specializare, Salariu, ID\_Pizzerie, Data\_Angajare, Email) VALUES (12, 'Daniela Enache', 'Ospatar', 2250, 'PIZZ004', TO\_DATE('2023-04-12', 'YYYY-MM-DD'), 'daniela.e@pizzerie4.ro');

INSERT INTO Angajat (ID\_Angajat, Nume, Specializare, Salariu, ID\_Pizzerie, Data\_Angajare, Email) VALUES (13, 'Radu Moldovan', 'Buc?tar', 3100, 'PIZZ005', TO\_DATE('2023-03-05', 'YYYY-MM-DD'), 'radu.m@pizzerie5.ro');

INSERT INTO Angajat (ID\_Angajat, Nume, Specializare, Salariu, ID\_Pizzerie, Data\_Angajare, Email) VALUES (14, 'Ioana Pavel', 'Menajera', 2450, 'PIZZ005', TO\_DATE('2023-05-08', 'YYYY-MM-DD'), 'ioana.p@pizzerie5.ro');

INSERT INTO Angajat (ID\_Angajat, Nume, Specializare, Salariu, ID\_Pizzerie, Data\_Angajare, Email) VALUES (15, 'Diana Petrescu', 'Ospatar', 2150, 'PIZZ005', TO\_DATE('2023-06-15', 'YYYY-MM-DD'), 'diana.p@pizzerie5.ro');

INSERT INTO Angajat (ID\_Angajat, Nume, Specializare, Salariu, ID\_Pizzerie, Data\_Angajare, Email) VALUES (16, 'Romica Becus', 'Ospatar', 2420, 'PIZZ006', TO\_DATE('2023-06-08', 'YYYY-MM-DD'), ['romi.p@pizzerie6.ro');](mailto:'romi.p@pizzerie6.ro');)



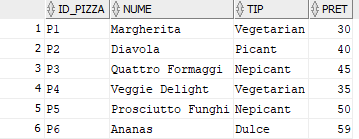
insert into Pizza (ID\_Pizza, Nume, Tip, Pret) VALUES ('P1', 'Margherita', 'Vegetarian', 30);

insert into Pizza (ID\_Pizza, Nume, Tip, Pret) VALUES ('P2', 'Diavola', 'Picant', 40);

insert into Pizza (ID\_Pizza, Nume, Tip, Pret) VALUES ('P3', 'Quattro Formaggi', 'Nepicant', 45);

insert into Pizza (ID\_Pizza, Nume, Tip, Pret) VALUES ('P4', 'Veggie Delight', 'Vegetarian', 35);

insert into Pizza (ID\_Pizza, Nume, Tip, Pret) VALUES ('P5', 'Prosciutto Funghi', 'Nepica



INSERT INTO Comanda (ID\_Comanda, Data\_Comanda, ID\_Pizzerie, ID\_Client)VALUES (1, TO\_DATE('2023-07-10', 'YYYY-MM-DD'), 'PIZZ001', 1);

INSERT INTO Comanda (ID\_Comanda, Data\_Comanda, ID\_Pizzerie, ID\_Client)VALUES (2, TO\_DATE('2023-07-12', 'YYYY-MM-DD'), 'PIZZ002', 2);

INSERT INTO Comanda (ID\_Comanda, Data\_Comanda, ID\_Pizzerie, ID\_Client)VALUES (3, TO\_DATE('2023-07-15', 'YYYY-MM-DD'), 'PIZZ003', 3);

INSERT INTO Comanda (ID\_Comanda, Data\_Comanda, ID\_Pizzerie, ID\_Client)VALUES (4, TO\_DATE('2023-07-18', 'YYYY-MM-DD'), 'PIZZ004', 4);

INSERT INTO Comanda (ID\_Comanda, Data\_Comanda, ID\_Pizzerie, ID\_Client)VALUES (5, TO\_DATE('2023-07-20', 'YYYY-MM-DD'), 'PIZZ005', 5);

INSERT INTO Comanda (ID\_Comanda, Data\_Comanda, ID\_Pizzerie, ID\_Client)VALUES (6, TO\_DATE('2023-07-22', 'YYYY-MM-DD'), 'PIZZ001', 2);

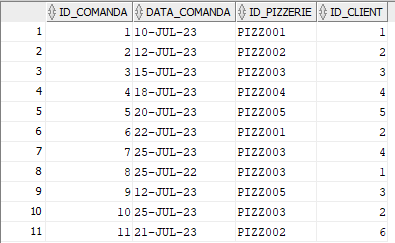
INSERT INTO Comanda (ID\_Comanda, Data\_Comanda, ID\_Pizzerie, ID\_Client)VALUES (7, TO\_DATE('2023-07-25', 'YYYY-MM-DD'), 'PIZZ003', 4);

INSERT INTO Comanda (ID\_Comanda, Data\_Comanda, ID\_Pizzerie, ID\_Client)VALUES (8, TO\_DATE('2022-07-25', 'YYYY-MM-DD'), 'PIZZ003', 1);

INSERT INTO Comanda (ID\_Comanda, Data\_Comanda, ID\_Pizzerie, ID\_Client)VALUES (9, TO\_DATE('2023-07-12', 'YYYY-MM-DD'), 'PIZZ005', 3);

INSERT INTO Comanda (ID\_Comanda, Data\_Comanda, ID\_Pizzerie, ID\_Client)VALUES (10, TO\_DATE('2023-07-25', 'YYYY-MM-DD'), 'PIZZ003', 2);

INSERT INTO Comanda (ID\_Comanda, Data\_Comanda, ID\_Pizzerie, ID\_Client)VALUES (11, TO\_DATE('2023-07-21', 'YYYY-MM-DD'), 'PIZZ002', 6);



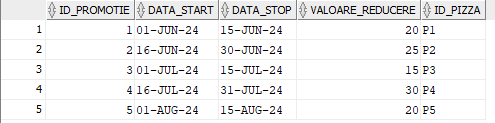
insert into Promotie (ID\_Promotie, Data\_Start, Data\_Stop, Valoare\_Reducere, ID\_Pizza) VALUES (1, TO\_DATE('2024-06-01', 'YYYY-MM-DD'), TO\_DATE('2024-06-15', 'YYYY-MM-DD'), 20, 'P1');

insert into Promotie (ID\_Promotie, Data\_Start, Data\_Stop, Valoare\_Reducere, ID\_Pizza) VALUES (2, TO\_DATE('2024-06-16', 'YYYY-MM-DD'), TO\_DATE('2024-06-30', 'YYYY-MM-DD'), 25, 'P2');

insert into Promotie (ID\_Promotie, Data\_Start, Data\_Stop, Valoare\_Reducere, ID\_Pizza) VALUES (3, TO\_DATE('2024-07-01', 'YYYY-MM-DD'), TO\_DATE('2024-07-15', 'YYYY-MM-DD'), 15, 'P3');

insert into Promotie (ID\_Promotie, Data\_Start, Data\_Stop, Valoare\_Reducere, ID\_Pizza) VALUES (4, TO\_DATE('2024-07-16', 'YYYY-MM-DD'), TO\_DATE('2024-07-31', 'YYYY-MM-DD'), 30, 'P4');

insert into Promotie (ID\_Promotie, Data\_Start, Data\_Stop, Valoare\_Reducere, ID\_Pizza) VALUES (5, TO\_DATE('2024-08-01', 'YYYY-MM-DD'), TO\_DATE('2024-08-15', 'YYYY-MM-DD'), 20, 'P5');



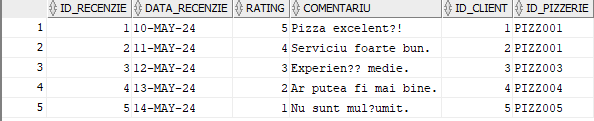
INSERT INTO Recenzie (ID\_Recenzie, Data\_Recenzie, Rating, Comentariu, ID\_Client, ID\_Pizzerie) VALUES (1, TO\_DATE('2024-05-10', 'YYYY-MM-DD'), 5, 'Pizza excelent?!', 1, 'PIZZ001');

INSERT INTO Recenzie (ID\_Recenzie, Data\_Recenzie, Rating, Comentariu, ID\_Client, ID\_Pizzerie) VALUES (2, TO\_DATE('2024-05-11', 'YYYY-MM-DD'), 4, 'Serviciu foarte bun.', 2, 'PIZZ001');

INSERT INTO Recenzie (ID\_Recenzie, Data\_Recenzie, Rating, Comentariu, ID\_Client, ID\_Pizzerie) VALUES (3, TO\_DATE('2024-05-12', 'YYYY-MM-DD'), 3, 'Experien?? medie.', 3, 'PIZZ003');

INSERT INTO Recenzie (ID\_Recenzie, Data\_Recenzie, Rating, Comentariu, ID\_Client, ID\_Pizzerie) VALUES (4, TO\_DATE('2024-05-13', 'YYYY-MM-DD'), 2, 'Ar putea fi mai bine.', 4, 'PIZZ004');

INSERT INTO Recenzie (ID\_Recenzie, Data\_Recenzie, Rating, Comentariu, ID\_Client, ID\_Pizzerie) VALUES (5, TO\_DATE('2024-05-14', 'YYYY-MM-DD'), 1, 'Nu sunt mul?umit.', 5, 'PIZZ005');



INSERT INTO COMANDA\_PIZZA (ID\_Pizza, ID\_Comanda, Cantitate) VALUES ('P1', 1,1);

INSERT INTO COMANDA\_PIZZA (ID\_Pizza, ID\_Comanda, Cantitate) VALUES ('P1', 2,2);

INSERT INTO COMANDA\_PIZZA (ID\_Pizza, ID\_Comanda, Cantitate) VALUES ('P2', 3,1);

INSERT INTO COMANDA\_PIZZA (ID\_Pizza, ID\_Comanda, Cantitate) VALUES ('P2', 4,1);

INSERT INTO COMANDA\_PIZZA (ID\_Pizza, ID\_Comanda, Cantitate) VALUES ('P3', 5,3);

INSERT INTO COMANDA\_PIZZA (ID\_Pizza, ID\_Comanda, Cantitate) VALUES ('P4', 6,2);

INSERT INTO COMANDA\_PIZZA (ID\_Pizza, ID\_Comanda, Cantitate) VALUES ('P4', 1, 1);

INSERT INTO COMANDA\_PIZZA (ID\_Pizza, ID\_Comanda, Cantitate) VALUES ('P5', 1,2);

INSERT INTO COMANDA\_PIZZA (ID\_Pizza, ID\_Comanda, Cantitate) VALUES ('P5', 3,2);

INSERT INTO COMANDA\_PIZZA (ID\_Pizza, ID\_Comanda, Cantitate) VALUES ('P5', 5,4);

INSERT INTO COMANDA\_PIZZA (ID\_Pizza, ID\_Comanda, Cantitate) VALUES ('P2', 2,3);

INSERT INTO COMANDA\_PIZZA (ID\_Pizza, ID\_Comanda, Cantitate) VALUES ('P3', 4,1);

INSERT INTO COMANDA\_PIZZA (ID\_Pizza, ID\_Comanda, Cantitate) VALUES ('P3', 6,2);

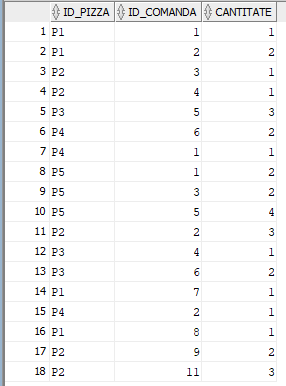
INSERT INTO COMANDA\_PIZZA (ID\_Pizza, ID\_Comanda, Cantitate) VALUES ('P1', 7,1);

INSERT INTO COMANDA\_PIZZA (ID\_Pizza, ID\_Comanda, Cantitate) VALUES ('P4', 2,1);

INSERT INTO COMANDA\_PIZZA (ID\_Pizza, ID\_Comanda, Cantitate) VALUES ('P1', 8,1);

INSERT INTO COMANDA\_PIZZA (ID\_Pizza, ID\_Comanda, Cantitate) VALUES ('P2', 9,2);

INSERT INTO COMANDA\_PIZZA (ID\_Pizza, ID\_Comanda, Cantitate) VALUES ('P2', 11,3);



1. Formulați în limbaj natural o problemă pe care să o rezolvați folosind un subprogram stocat independent care să utilizeze toate cele 3 tipuri de colecții studiate. Apelați subprogramul.

**Sa se afiseze, pentru fiecare client, toate pizzele comandate de acesta impreuna cu pizzeria in care a efectuat comanda pentru comenzie efectuate intr-o anumita perioada de timp.**

CREATE OR REPLACE PROCEDURE EX\_6 (

p\_data\_start DATE,

p\_data\_stop DATE

) IS

TYPE Tablou\_Pizzas IS TABLE OF VARCHAR2(50);

TYPE Tablou\_Clienti\_Pizzerii IS TABLE OF VARCHAR2(50) INDEX BY PLS\_INTEGER;

TYPE Vector\_Comenzi IS VARRAY(100) OF VARCHAR2(100);

clienti\_pizzerii Tablou\_Clienti\_Pizzerii;

v\_comenzi Vector\_Comenzi := Vector\_Comenzi();

pizzas Tablou\_Pizzas := Tablou\_Pizzas();

v\_index PLS\_INTEGER := 0;

v\_comanda VARCHAR2(100);

BEGIN

FOR comanda IN (

SELECT c.nume AS nume\_client, p.nume AS nume\_pizzerie, z.nume AS nume\_pizza

FROM Comanda o

JOIN Client c ON o.ID\_Client = c.ID\_Client

JOIN Pizzerie p ON o.ID\_Pizzerie = p.ID\_Pizzerie

JOIN COMANDA\_PIZZA cp ON o.ID\_Comanda = cp.ID\_Comanda

JOIN Pizza z ON cp.ID\_Pizza = z.ID\_Pizza

WHERE o.Data\_Comanda BETWEEN p\_data\_start AND p\_data\_stop

)

LOOP

v\_index := v\_index + 1;

clienti\_pizzerii(v\_index) := comanda.nume\_client || ' la ' || comanda.nume\_pizzerie;

pizzas.EXTEND;

pizzas(pizzas.COUNT) := comanda.nume\_pizza;

v\_comanda := clienti\_pizzerii(v\_index) || ': ' || pizzas(pizzas.COUNT);

v\_comenzi.EXTEND;

v\_comenzi(v\_comenzi.COUNT) := v\_comanda;

END LOOP;

DBMS\_OUTPUT.PUT\_LINE('Lista comenzilor:');

FOR i IN 1 .. v\_comenzi.COUNT LOOP

DBMS\_OUTPUT.PUT\_LINE(v\_comenzi(i));

END LOOP;

END;

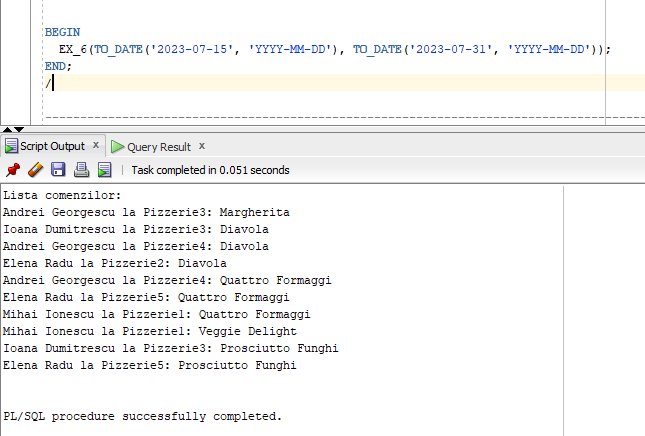
/

BEGIN

EX\_6(TO\_DATE('2023-07-15', 'YYYY-MM-DD'), TO\_DATE('2023-07-31', 'YYYY-MM-DD'));

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, dependent de celălalt cursor. Apelați subprogramul.

**Pentru fiecare pizzerie in parte, sa se afiseze cea mai comandata pizza, tipul acesteia, de cate ori a fost comandata si data ultimei comenzi efectuate.**

CREATE OR REPLACE PROCEDURE EX\_7

IS

CURSOR c\_pizzerii IS

SELECT ID\_Pizzerie, Nume

FROM Pizzerie;

CURSOR c\_pizza\_populara(p\_id\_pizzerie VARCHAR2) IS

SELECT p.Nume, p.Tip, SUM(cp.Cantitate) AS Total\_Comenzi, MAX(c.Data\_Comanda) AS Ultima\_Comanda

FROM Pizza p

JOIN COMANDA\_PIZZA cp ON p.ID\_Pizza = cp.ID\_Pizza

JOIN Comanda c ON cp.ID\_Comanda = c.ID\_Comanda

WHERE c.ID\_Pizzerie = p\_id\_pizzerie

GROUP BY p.Nume, p.Tip

ORDER BY Total\_Comenzi DESC;

v\_Nume Pizza.Nume%TYPE;

v\_Tip Pizza.Tip%TYPE;

v\_Total\_Comenzi NUMBER;

v\_Ultima\_Comanda DATE;

BEGIN

FOR pizzerie IN c\_pizzerii LOOP

DBMS\_OUTPUT.PUT\_LINE('Pizzerie: ' || pizzerie.Nume);

OPEN c\_pizza\_populara(pizzerie.ID\_Pizzerie);

FETCH c\_pizza\_populara INTO v\_Nume, v\_Tip, v\_Total\_Comenzi, v\_Ultima\_Comanda;

IF c\_pizza\_populara%FOUND THEN

DBMS\_OUTPUT.PUT\_LINE(' Pizza Populara: ' || v\_Nume);

DBMS\_OUTPUT.PUT\_LINE(' Tip: ' || v\_Tip);

DBMS\_OUTPUT.PUT\_LINE(' Total Comenzi: ' || v\_Total\_Comenzi);

DBMS\_OUTPUT.PUT\_LINE(' Ultima Comanda: ' || TO\_CHAR(v\_Ultima\_Comanda, 'DD-MON-YYYY'));

ELSE

DBMS\_OUTPUT.PUT\_LINE(' Nu s-au gasit comenzi pentru aceasta pizzerie.');

END IF;

CLOSE c\_pizza\_populara;

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

END LOOP;

END EX\_7;

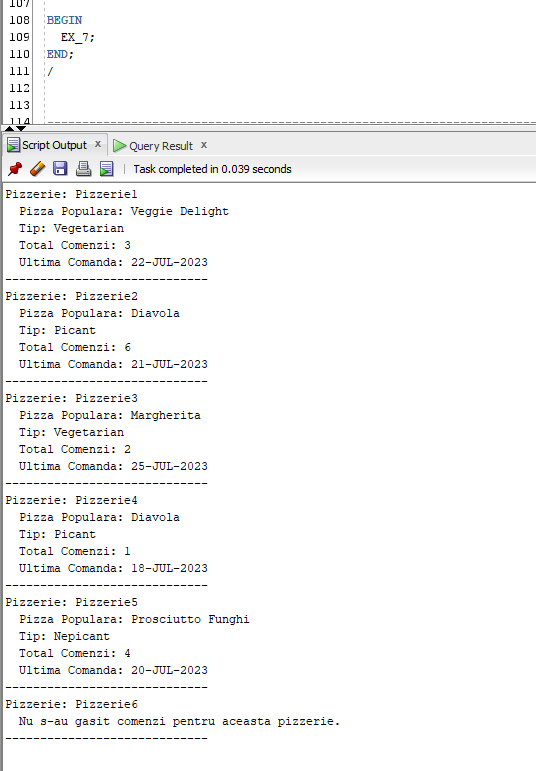
/

BEGIN

EX\_7;

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 proprii. Apelați subprogramul astfel încât să evidențiați toate cazurile definite și tratate.

**Pentru fiecare pizzerie, sa se afiseze nr de recenzii, respectiv de comenzi efectuate in cadrul acesteia si valoarea totala a comenzilor. Sa se trateze cazurile in care pizzeria fie nu exista, fie nu au fost efectuate comenzi si recenzii.**

CREATE OR REPLACE FUNCTION EX\_8(p\_id\_pizzerie VARCHAR2)

RETURN VARCHAR2

IS

v\_nume\_pizzerie VARCHAR2(10);

v\_nr\_recenzii INT;

v\_nr\_comenzi INT;

v\_suma\_totala INT;

Pizzerie\_Inexistenta EXCEPTION;

Fara\_Activitate EXCEPTION;

BEGIN

BEGIN

SELECT Nume INTO v\_nume\_pizzerie

FROM Pizzerie

WHERE ID\_Pizzerie = p\_id\_pizzerie;

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

RAISE Pizzerie\_Inexistenta;

END;

SELECT COUNT(\*) INTO v\_nr\_recenzii

FROM Recenzie

WHERE ID\_Pizzerie = p\_id\_pizzerie;

SELECT COUNT(\*) INTO v\_nr\_comenzi

FROM Comanda

WHERE ID\_Pizzerie = p\_id\_pizzerie;

SELECT NVL(SUM(cp.Cantitate \* p.Pret), 0) INTO v\_suma\_totala

FROM COMANDA\_PIZZA cp

JOIN Pizza p ON cp.ID\_Pizza = p.ID\_Pizza

JOIN Comanda c ON cp.ID\_Comanda = c.ID\_Comanda

WHERE c.ID\_Pizzerie = p\_id\_pizzerie;

IF v\_nr\_recenzii = 0 AND v\_nr\_comenzi = 0 THEN

RAISE Fara\_Activitate;

END IF;

RETURN 'Pizzerie: ' || v\_nume\_pizzerie || ', Recenzii: ' || v\_nr\_recenzii || ', Comenzi: ' || v\_nr\_comenzi || ', Suma totala: ' || v\_suma\_totala || ' lei';

EXCEPTION

WHEN Pizzerie\_Inexistenta THEN

RETURN 'Pizzeria cu ID-ul ' || p\_id\_pizzerie || ' nu exista.';

WHEN Fara\_Activitate THEN

RETURN 'Pizzeria ' || v\_nume\_pizzerie || ' nu are nicio activitate (fara comenzi sau recenzii).';

END;

/

BEGIN

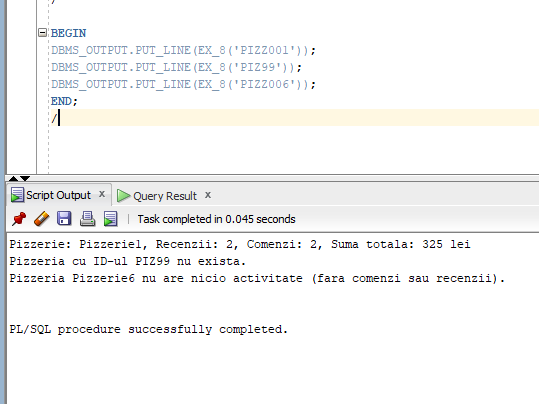
DBMS\_OUTPUT.PUT\_LINE(EX\_8('PIZZ001'));

DBMS\_OUTPUT.PUT\_LINE(EX\_8('PIZ99'));

DBMS\_OUTPUT.PUT\_LINE(EX\_8('PIZZ006'));

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 client dat dupa nume, sa se afiseze toate pizzele comandate de acesta, respectiv in ce pizzerii au fost comandate intr-o perioada specificata.**

CREATE OR REPLACE PROCEDURE EX\_9(

p\_Nume\_Client IN Client.Nume%TYPE,

p\_Data\_Start IN DATE,

p\_Data\_Stop IN DATE

)

IS

v\_Nume\_Pizzerie Pizzerie.Nume%TYPE;

v\_Nume\_Pizza Pizza.Nume%TYPE;

v\_Tip\_Pizza Pizza.Tip%TYPE;

v\_Cursor SYS\_REFCURSOR;

v\_Client\_Count INT;

invalid\_characters EXCEPTION;

PRAGMA EXCEPTION\_INIT(invalid\_characters, -20001);

BEGIN

IF NOT REGEXP\_LIKE(p\_Nume\_Client, '^[A-Za-z ]+$') THEN

RAISE invalid\_characters;

END IF;

SELECT COUNT(\*)

INTO v\_Client\_Count

FROM Client

WHERE Nume = p\_Nume\_Client;

IF v\_Client\_Count > 1 THEN

RAISE TOO\_MANY\_ROWS;

ELSIF v\_Client\_Count = 0 THEN

RAISE NO\_DATA\_FOUND;

END IF;

OPEN v\_Cursor FOR

SELECT pz.Nume AS Nume\_Pizzerie,

pi.Nume AS Nume\_Pizza,

pi.Tip AS Tip\_Pizza

FROM Client c

JOIN Comanda co ON c.ID\_Client = co.ID\_Client

JOIN Pizzerie pz ON co.ID\_Pizzerie = pz.ID\_Pizzerie

JOIN COMANDA\_PIZZA cp ON co.ID\_Comanda = cp.ID\_Comanda

JOIN Pizza pi ON cp.ID\_Pizza = pi.ID\_Pizza

WHERE c.Nume = p\_Nume\_Client

AND co.Data\_Comanda BETWEEN p\_Data\_Start AND p\_Data\_Stop;

FETCH v\_Cursor INTO v\_Nume\_Pizzerie, v\_Nume\_Pizza, v\_Tip\_Pizza;

IF v\_Cursor%NOTFOUND THEN

RAISE NO\_DATA\_FOUND;

END IF;

LOOP

EXIT WHEN v\_Cursor%NOTFOUND;

DBMS\_OUTPUT.PUT\_LINE('Pizzerie: ' || v\_Nume\_Pizzerie || ', Pizza: ' || v\_Nume\_Pizza || ' (' || v\_Tip\_Pizza || ')');

FETCH v\_Cursor INTO v\_Nume\_Pizzerie, v\_Nume\_Pizza, v\_Tip\_Pizza;

END LOOP;

CLOSE v\_Cursor;

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

DBMS\_OUTPUT.PUT\_LINE('Niciun client cu numele ' || p\_Nume\_Client || ' nu a plasat vreo comanda in perioada specificata.');

WHEN TOO\_MANY\_ROWS THEN

DBMS\_OUTPUT.PUT\_LINE('Exista mai multi clienti cu numele ' || p\_Nume\_Client || '.');

WHEN invalid\_characters THEN

DBMS\_OUTPUT.PUT\_LINE('Numele clientului contine caractere invalide.');

END EX\_9;

/

BEGIN

EX\_9('Mihai Ionescu', TO\_DATE('2023-01-01', 'YYYY-MM-DD'), TO\_DATE('2023-12-31', 'YYYY-MM-DD'));

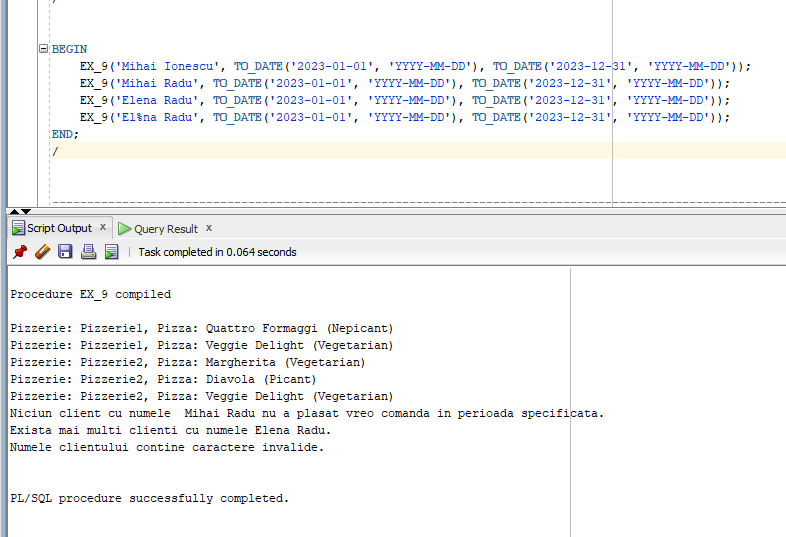
EX\_9('Mihai Radu', TO\_DATE('2023-01-01', 'YYYY-MM-DD'), TO\_DATE('2023-12-31', 'YYYY-MM-DD'));

EX\_9('Elena Radu', TO\_DATE('2023-01-01', 'YYYY-MM-DD'), TO\_DATE('2023-12-31', 'YYYY-MM-DD'));

EX\_9('El%na Radu', TO\_DATE('2023-01-01', 'YYYY-MM-DD'), TO\_DATE('2023-12-31', 'YYYY-MM-DD'));

END;

/



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

**Creati un trigger care interzice angajarea bucatarilor in luna iulie 2024.**

CREATE OR REPLACE TRIGGER EX\_10

BEFORE INSERT OR UPDATE ON Angajat

FOR EACH ROW

BEGIN

IF (EXTRACT(YEAR FROM :NEW.Data\_Angajare) = 2024 AND EXTRACT(MONTH FROM :NEW.Data\_Angajare) = 7) AND

(:NEW.Specializare = 'Bucatar') THEN

RAISE\_APPLICATION\_ERROR(-20001, 'Nu se pot angaja bucatari în luna iulie 2024. Este prea cald !');

END IF;

END;

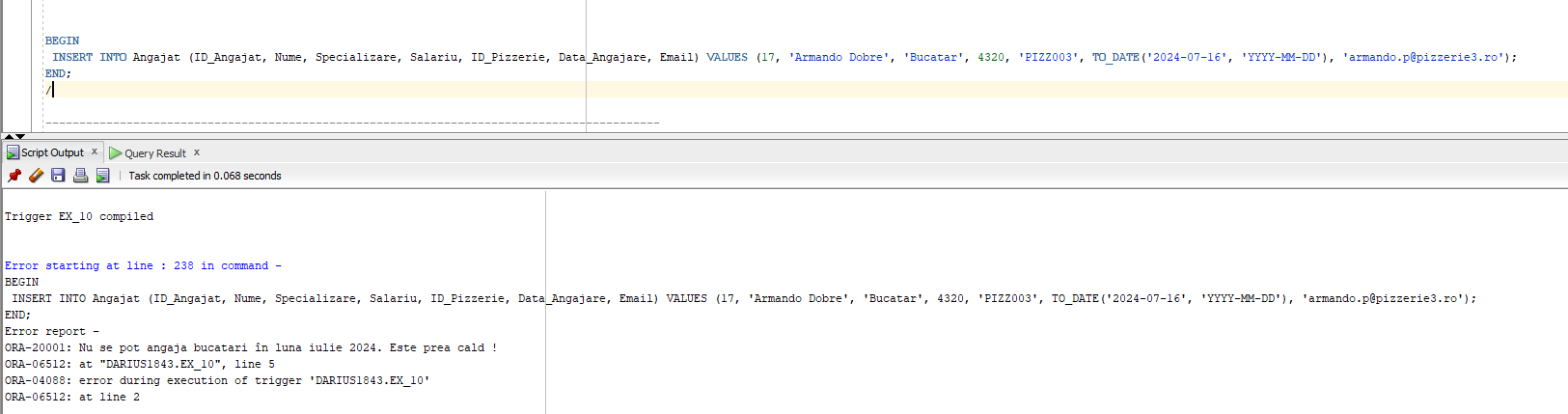
/

BEGIN

INSERT INTO Angajat (ID\_Angajat, Nume, Specializare, Salariu, ID\_Pizzerie, Data\_Angajare, Email) VALUES (17, 'Armando Dobre', 'Bucatar', 4320, 'PIZZ003', TO\_DATE('2024-07-16', 'YYYY-MM-DD'), 'armando.p@pizzerie3.ro');

END;

/



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

Creati un trigger care interzice modificarea notelor acordate in recenzie.

CREATE OR REPLACE TRIGGER EX\_11

BEFORE UPDATE ON Recenzie

FOR EACH ROW

BEGIN

IF :OLD.Rating != :NEW.Rating THEN

RAISE\_APPLICATION\_ERROR(-20001, 'Nu se poate modifica rating-ul acordat în review-uri.');

END IF;

END;

/

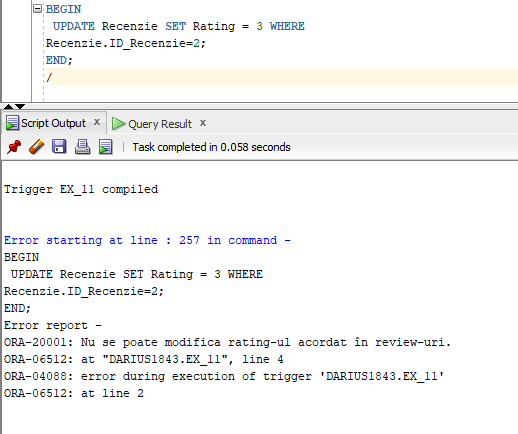
BEGIN

UPDATE Recenzie SET Rating = 3 WHERE

Recenzie.ID\_Recenzie=2;

END;

/



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

**Creati un trigger care sa permita modificarea pretului pizzelor doar de utilizatorul darius1843.**

CREATE OR REPLACE TRIGGER EX\_12

BEFORE UPDATE OF Pret ON Pizza

FOR EACH ROW

BEGIN

IF SYS\_CONTEXT('USERENV', 'SESSION\_USER') != 'DARIUS1843' THEN

RAISE\_APPLICATION\_ERROR(-20001, 'Numai administratorul pizzeriei poate modifica pretul pizzei.');

END IF;

END;

/

select \* from pizza;

UPDATE Pizza

SET Pret = 31

WHERE ID\_Pizza = 'P1';

