

PROIECT SGDB

- MAGAZIN ONLINE -

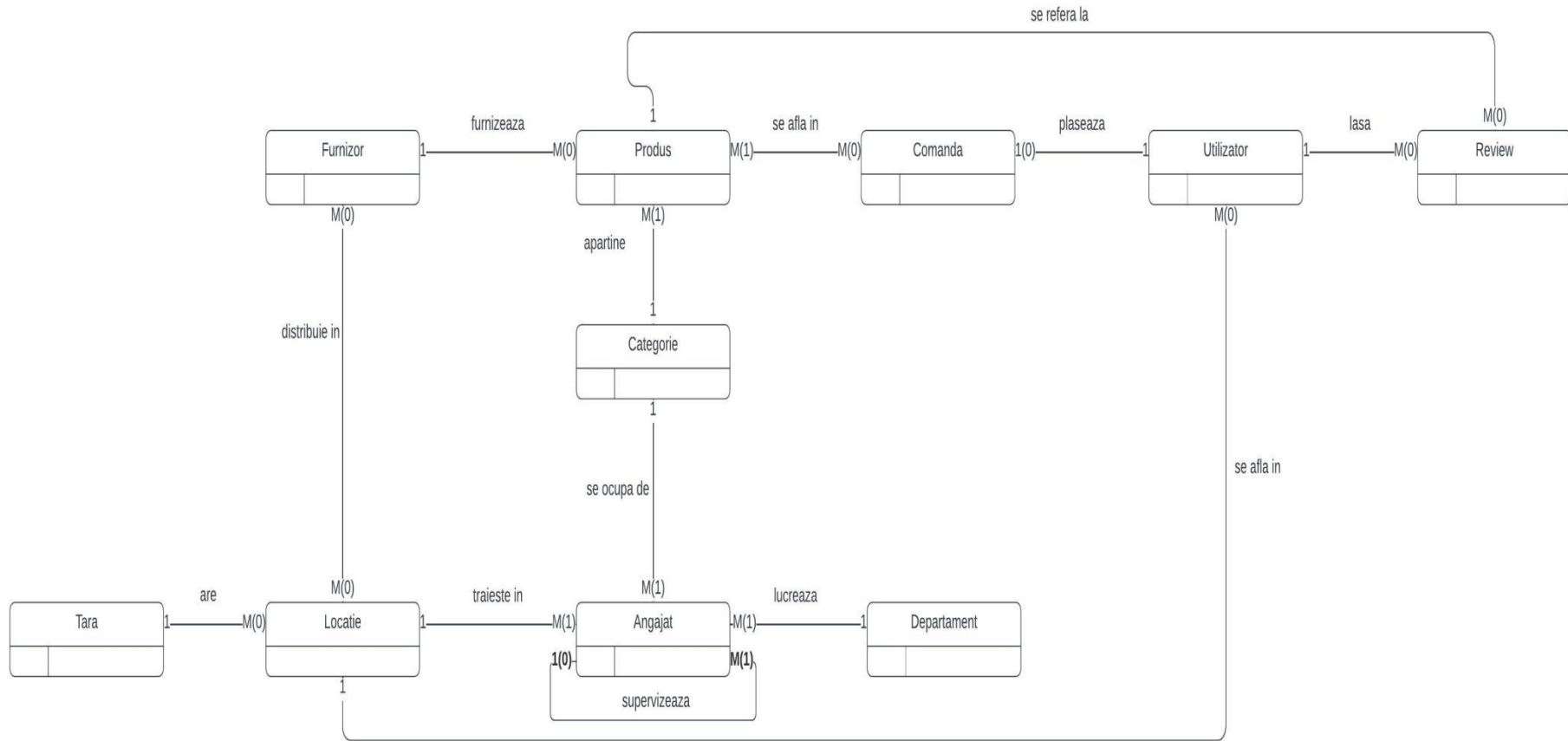
OCNARU MIHAI-OCTAVIAN

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

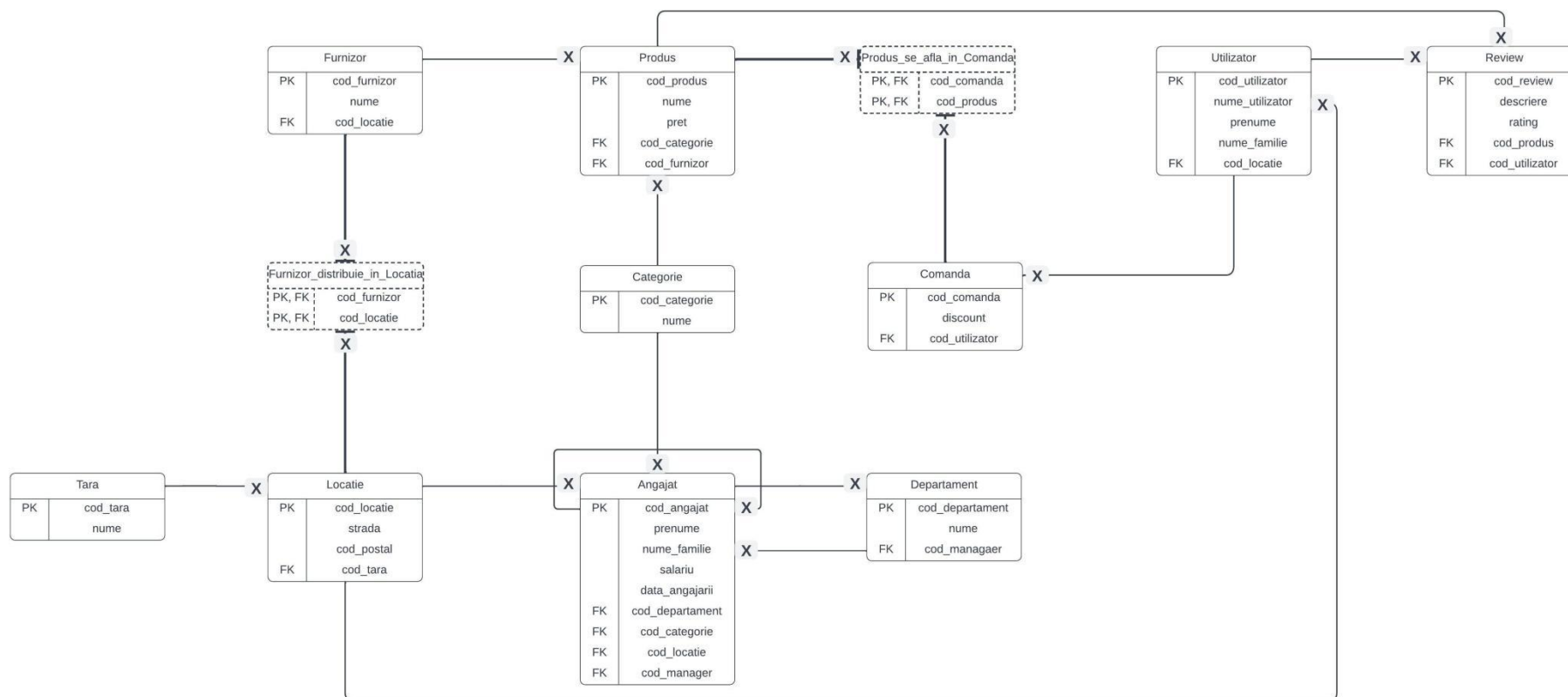
Modelul real al bazei de date pentru un magazin online este acela al unei afaceri care vinde produse online, primind comenzi de la clienți și livrând produsele la adresele specificate de aceștia. Magazinul online are o gamă largă de produse, inclusiv electronice, îmbrăcăminte, produse de îngrijire personală și multe altele.

Utilitatea bazei de date este de a gestiona și stoca informațiile despre produse, comenzi, clienți, plăți, recenzii, furnizori și angajați, permițând afacerii să-și gestioneze eficient operațiunile, să-și urmărească stocurile, să proceseze comenzi și să furnizeze un serviciu excelent clienților săi. Astfel pe baza acestor metrice putând să îmbunătățească atât recomandările de produse, dar și alegerea furnizorilor potriviți cererii pieței.

2. Realizați diagrama entitate-relație (ERD): entitățile, relațiile și atributele trebuie definite în limba română (vezi curs SGBD / model de diagrama ERD; nu se va accepta alt format).



3. Pornind de la diagrama entitate-relație realizați diagrama conceptuală a modelului propus, integrând toate attributele necesare: entitățile, relațiile și attributele trebuie definite în limba română.



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

– secvența pt o tabelă

-- Tabela **Tara**

```
CREATE TABLE Tara (  
    cod_tara INT PRIMARY KEY,  
    nume VARCHAR(50)  
);
```

-- Tabela **Categorie**

```
CREATE TABLE Categorie (  
    cod_categorie INT PRIMARY KEY,  
    nume VARCHAR(50)  
);
```

-- Tabela **Locatie**

```
CREATE TABLE Locatie (  
    cod_locatie INT PRIMARY KEY,  
    strada VARCHAR(50),  
    cod_postal VARCHAR(10),  
    cod_tara INT,  
    FOREIGN KEY (cod_tara) REFERENCES Tara(cod_tara)  
);
```

-- Tabela **Departament**

```
CREATE TABLE Departament (  
    cod_departament INT PRIMARY KEY,  
    nume VARCHAR(50),  
    cod_manager INT  
);
```

-- Tabela **Angajat**

```
CREATE TABLE Angajat (  
    cod_angajat INT PRIMARY KEY,  
    prenume VARCHAR(50),  
    nume_familie VARCHAR(50),  
    cod_departament INT,  
    cod_categorie INT,  
    cod_locatie INT,  
    cod_manager INT,
```

```

    FOREIGN KEY (cod_departament) REFERENCES
Departament(cod_departament),
    FOREIGN KEY (cod_categorie) REFERENCES Categorie(cod_categorie),
    FOREIGN KEY (cod_locatie) REFERENCES Locatie(cod_locatie),
    FOREIGN KEY (cod_manager) REFERENCES Angajat(cod_angajat)
);

-- Alterare tabel Departament pentru adăugarea restricției cheii externe
către tabela Angajat
ALTER TABLE Departament
ADD CONSTRAINT FK_Departament_Angajat FOREIGN KEY (cod_manager)
REFERENCES Angajat(cod_angajat);

-- Tabela Furnizor
CREATE TABLE Furnizor (
    cod_furnizor INT PRIMARY KEY,
    nume VARCHAR(50),
    cod_locatie INT,
    FOREIGN KEY (cod_locatie) REFERENCES Locatie(cod_locatie)
);

-- Tabela Produs
CREATE TABLE Produs (
    cod_produs INT PRIMARY KEY,
    nume VARCHAR(50),
    pret DECIMAL(10,2),
    cod_categorie INT,
    cod_furnizor INT,
    FOREIGN KEY (cod_categorie) REFERENCES Categorie(cod_categorie),
    FOREIGN KEY (cod_furnizor) REFERENCES Furnizor(cod_furnizor)
);

-- Tabela Utilizator
CREATE TABLE Utilizator (
    cod_utilizator INT PRIMARY KEY,
    nume_utilizator VARCHAR(50) NOT NULL,
    prenume VARCHAR(50),
    nume_familie VARCHAR(50),
    cod_locatie INT,
    FOREIGN KEY (cod_locatie) REFERENCES Locatie(cod_locatie)
);

```

-- Tabela Comanda

```
CREATE TABLE Comanda (  
    cod_comanda INT PRIMARY KEY,  
    suma DECIMAL(10,2),  
    cod_utilizator INT,  
    FOREIGN KEY (cod_utilizator) REFERENCES Utilizator(cod_utilizator)  
);
```

-- Tabela Review

```
CREATE TABLE Review (  
    cod_review INT PRIMARY KEY,  
    descriere VARCHAR(255),  
    rating INT,  
    cod_produs INT,  
    cod_utilizator INT,  
    FOREIGN KEY (cod_produs) REFERENCES Produs(cod_produs),  
    FOREIGN KEY (cod_utilizator) REFERENCES Utilizator(cod_utilizator)  
);
```

-- Tabela Produs_se_afila_in_Comanda

```
CREATE TABLE Produs_se_afila_in_Comanda (  
    cod_comanda INT,  
    cod_produs INT,  
    PRIMARY KEY (cod_comanda, cod_produs),  
    FOREIGN KEY (cod_comanda) REFERENCES Comanda(cod_comanda),  
    FOREIGN KEY (cod_produs) REFERENCES Produs(cod_produs)  
);
```

-- Tabela Furnizor_distribuie_in_Loc

```
CREATE TABLE Furnizor_distribuie_in_Locatia (  
    cod_furnizor INT,  
    cod_locatie INT,  
    PRIMARY KEY (cod_furnizor, cod_locatie),  
    FOREIGN KEY (cod_furnizor) REFERENCES Furnizor(cod_furnizor),  
    FOREIGN KEY (cod_locatie) REFERENCES Locatie(cod_locatie)  
);
```

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

Tabela CATEGORIE:

```
INSERT INTO Categorie (cod_categorie, nume) VALUES (1, 'Electronice');
INSERT INTO Categorie (cod_categorie, nume) VALUES (2, 'Îmbrăcăminte');
INSERT INTO Categorie (cod_categorie, nume) VALUES (3, 'Mobila');
INSERT INTO Categorie (cod_categorie, nume) VALUES (4, 'Cosmetice');
INSERT INTO Categorie (cod_categorie, nume) VALUES (5, 'Alimente');
INSERT INTO Categorie (cod_categorie, nume) VALUES (6, 'Jucării');
INSERT INTO Categorie (cod_categorie, nume) VALUES (7, 'Articole sportive');
INSERT INTO Categorie (cod_categorie, nume) VALUES (8, 'Bijuterii');
INSERT INTO Categorie (cod_categorie, nume) VALUES (9, 'Unelte');
INSERT INTO Categorie (cod_categorie, nume) VALUES (10, 'Rechizite');
```

```
1 -- INSERT INTO Categorie (cod_categorie, nume) VALUES (1, 'Electronice');
2 -- INSERT INTO Categorie (cod_categorie, nume) VALUES (2, 'Îmbrăcăminte');
3 -- INSERT INTO Categorie (cod_categorie, nume) VALUES (3, 'Mobila');
4 -- INSERT INTO Categorie (cod_categorie, nume) VALUES (4, 'Cosmetice');
5 -- INSERT INTO Categorie (cod_categorie, nume) VALUES (5, 'Alimente');
6 -- INSERT INTO Categorie (cod_categorie, nume) VALUES (6, 'Jucării');
7 -- INSERT INTO Categorie (cod_categorie, nume) VALUES (7, 'Articole sportive');
8 -- INSERT INTO Categorie (cod_categorie, nume) VALUES (8, 'Bijuterii');
9 -- INSERT INTO Categorie (cod_categorie, nume) VALUES (9, 'Unelte');
10 -- INSERT INTO Categorie (cod_categorie, nume) VALUES (10, 'Rechizite');
11
12 select * from categorie;
```

COD_CATEGORIE	NUME
1	Electronice
2	Îmbrăcăminte
3	Mobila
4	Cosmetice
5	Alimente
6	Jucării

Tabela TARA:

```
INSERT INTO Tara (cod_tara, nume) VALUES (1, 'România');
INSERT INTO Tara (cod_tara, nume) VALUES (2, 'Germania');
INSERT INTO Tara (cod_tara, nume) VALUES (3, 'Franța');
INSERT INTO Tara (cod_tara, nume) VALUES (4, 'Spania');
INSERT INTO Tara (cod_tara, nume) VALUES (5, 'Italia');
INSERT INTO Tara (cod_tara, nume) VALUES (6, 'Olanda');
INSERT INTO Tara (cod_tara, nume) VALUES (7, 'Marea Britanie');
INSERT INTO Tara (cod_tara, nume) VALUES (8, 'Suedia');
INSERT INTO Tara (cod_tara, nume) VALUES (9, 'Elveția');
INSERT INTO Tara (cod_tara, nume) VALUES (10, 'Grecia');
```

```
1  -- INSERT INTO Tara (cod_tara, nume) VALUES (1, 'România');
2  -- INSERT INTO Tara (cod_tara, nume) VALUES (2, 'Germania');
3  -- INSERT INTO Tara (cod_tara, nume) VALUES (3, 'Franța');
4  -- INSERT INTO Tara (cod_tara, nume) VALUES (4, 'Spania');
5  -- INSERT INTO Tara (cod_tara, nume) VALUES (5, 'Italia');
6  -- INSERT INTO Tara (cod_tara, nume) VALUES (6, 'Olanda');
7  -- INSERT INTO Tara (cod_tara, nume) VALUES (7, 'Marea Britanie');
8  -- INSERT INTO Tara (cod_tara, nume) VALUES (8, 'Suedia');
9  -- INSERT INTO Tara (cod_tara, nume) VALUES (9, 'Elveția');
10 -- INSERT INTO Tara (cod_tara, nume) VALUES (10, 'Grecia');
11
12 select * from tara;
```

COD_TARA	NUME
1	România
2	Germania
3	Franța
4	Spania
5	Italia
6	Olanda
7	Marea Britanie
8	Suedia

Tabela LOCATIE:

-- Romania

```
INSERT INTO Locatie (cod_locatie, strada, cod_postal, cod_tara) VALUES  
(1, 'Strada Victoriei', '010051', 1);
```

```
INSERT INTO Locatie (cod_locatie, strada, cod_postal, cod_tara) VALUES  
(2, 'Bulevardul Unirii', '030167', 1);
```

-- Germania

```
INSERT INTO Locatie (cod_locatie, strada, cod_postal, cod_tara) VALUES  
(3, 'Alexanderplatz', '10178', 2);
```

```
INSERT INTO Locatie (cod_locatie, strada, cod_postal, cod_tara) VALUES  
(4, 'Kurfürstendamm', '10719', 2);
```

-- Franta

```
INSERT INTO Locatie (cod_locatie, strada, cod_postal, cod_tara) VALUES  
(5, 'Avenue des Champs-Élysées', '75008', 3);
```

```
INSERT INTO Locatie (cod_locatie, strada, cod_postal, cod_tara) VALUES  
(6, 'Rue de Rivoli', '75001', 3);
```

-- Spania

```
INSERT INTO Locatie (cod_locatie, strada, cod_postal, cod_tara) VALUES  
(7, 'Paseo de la Castellana', '08046', 4);
```

```
INSERT INTO Locatie (cod_locatie, strada, cod_postal, cod_tara) VALUES  
(8, 'Rambla de Catalunya', '08007', 4);
```

-- Italia

```
INSERT INTO Locatie (cod_locatie, strada, cod_postal, cod_tara) VALUES  
(9, 'Via del Corso', '00187', 5);
```

```
INSERT INTO Locatie (cod_locatie, strada, cod_postal, cod_tara) VALUES  
(10, 'Piazza San Marco', '00124', 5);
```

-- Olanda

```
INSERT INTO Locatie (cod_locatie, strada, cod_postal, cod_tara) VALUES  
(11, 'Dam Square', '1012JS', 6);
```

```
INSERT INTO Locatie (cod_locatie, strada, cod_postal, cod_tara) VALUES  
(12, 'Leidseplein', '1017PT', 6);
```

-- Marea Britanie

```
INSERT INTO Locatie (cod_locatie, strada, cod_postal, cod_tara) VALUES  
(13, 'Baker Street', 'NW1 5LA', 7);
```

```
INSERT INTO Locatie (cod_locatie, strada, cod_postal, cod_tara) VALUES  
(14, 'Oxford Street', 'W1D 1BS', 7);
```

-- Suedia

```
INSERT INTO Locatie (cod_locatie, strada, cod_postal, cod_tara) VALUES  
(15, 'Drottninggatan', '111 21', 8);
```

```
INSERT INTO Locatie (cod_locatie, strada, cod_postal, cod_tara) VALUES  
(16, 'Kungsgatan', '111 43', 8);
```

-- Elvetia

```
INSERT INTO Locatie (cod_locatie, strada, cod_postal, cod_tara) VALUES  
(17, 'Bahnhofstrasse', '1001', 9);
```

```
INSERT INTO Locatie (cod_locatie, strada, cod_postal, cod_tara) VALUES  
(18, 'Rue du Rhône', '1204', 9);
```

-- Grecia

```
INSERT INTO Locatie (cod_locatie, strada, cod_postal, cod_tara) VALUES  
(19, 'Athinas Street', '105 51', 10);
```

```
INSERT INTO Locatie (cod_locatie, strada, cod_postal, cod_tara) VALUES  
(20, 'Ermou Street', '105 63', 10);
```

```
27 -- INSERT INTO Locatie (cod_locatie, strada, cod_postal, cod_tara) VALUES (14, 'Oxford Street', 'W1D 1BS', 7);  
28  
29 -- -- Suedia  
30 -- INSERT INTO Locatie (cod_locatie, strada, cod_postal, cod_tara) VALUES (15, 'Drottninggatan', '111 21', 8);  
31 -- INSERT INTO Locatie (cod_locatie, strada, cod_postal, cod_tara) VALUES (16, 'Kungsgatan', '111 43', 8);  
32  
33 -- -- Elvetia  
34 -- INSERT INTO Locatie (cod_locatie, strada, cod_postal, cod_tara) VALUES (17, 'Bahnhofstrasse', '8001', 9);  
35 -- INSERT INTO Locatie (cod_locatie, strada, cod_postal, cod_tara) VALUES (18, 'Rue du Rhône', '1204', 9);  
36  
37 -- -- Grecia  
38 -- INSERT INTO Locatie (cod_locatie, strada, cod_postal, cod_tara) VALUES (19, 'Athinas Street', '105 51', 10);  
39 -- INSERT INTO Locatie (cod_locatie, strada, cod_postal, cod_tara) VALUES (20, 'Ermou Street', '105 63', 10);  
40  
41 select * from locatie;
```

COD_LOCATIE	STRADA	COD_POSTAL	COD_TARA
1	Strada Victoriei	010051	1
2	Bulevardul Unirii	030167	1
3	Alexanderplatz	10178	2
4	Kurfürstendamm	10719	2
5	Avenue des Champs-Élysées	75008	3
6	Rue de Rivoli	75001	3

Tabela DEPARTAMENTE:

-- Department 1

```
INSERT INTO Departament (cod_departament, nume, cod_manager)
VALUES (1, 'IT', NULL);
```

-- Department 2

```
INSERT INTO Departament (cod_departament, nume, cod_manager)
VALUES (2, 'Vânzări', NULL);
```

-- Department 3

```
INSERT INTO Departament (cod_departament, nume, cod_manager)
VALUES (3, 'Resurse Umane', NULL);
```

-- Department 4

```
INSERT INTO Departament (cod_departament, nume, cod_manager)
VALUES (4, 'Marketing', NULL);
```

-- Department 5

```
INSERT INTO Departament (cod_departament, nume, cod_manager)
VALUES (5, 'Producție', NULL);
```

-- Department 6

```
INSERT INTO Departament (cod_departament, nume, cod_manager)
VALUES (6, 'Finanțe', NULL);
```

-- Department 7

```
INSERT INTO Departament (cod_departament, nume, cod_manager)
VALUES (7, 'Logistică', NULL);
```

```
8 -- INSERT INTO Departament (cod_departament, nume, cod_manager) VALUES (3, 'Resurse Umane', NULL);
9
10 -- -- Department 4
11 -- INSERT INTO Departament (cod_departament, nume, cod_manager) VALUES (4, 'Marketing', NULL);
12
13 -- -- Department 5
14 -- INSERT INTO Departament (cod_departament, nume, cod_manager) VALUES (5, 'Producție', NULL);
15
16 -- -- Department 6
17 -- INSERT INTO Departament (cod_departament, nume, cod_manager) VALUES (6, 'Finanțe', NULL);
18
19 -- -- Department 7
20 -- INSERT INTO Departament (cod_departament, nume, cod_manager) VALUES (7, 'Logistică', NULL);
21
22 select * from departament;
```

COD_DEPARTAMENT	NUME	COD_MANAGER
1	IT	-
2	Vânzări	-
3	Resurse Umane	-
4	Marketing	-

-- Manager for Department 1

```
INSERT INTO Angajat (cod_angajat, prenume, nume_familie, salariu,
data_angajarii, cod_departament, cod_categorie, cod_locatie,
cod_manager)
```

```
VALUES (1, 'John', 'Smith', 20000, TO_DATE('2002-05-21', 'YYYY-MM-DD'),
1, 5, 3, NULL);
```

-- Manager for Department 2

```
INSERT INTO Angajat (cod_angajat, prenume, nume_familie, salariu,
data_angajarii, cod_departament, cod_categorie, cod_locatie,
cod_manager)
```

```
VALUES (2, 'Maria', 'Johnson', 20500, TO_DATE('2000-04-02', 'YYYY-MM-
DD'), 2, 2, 4, NULL);
```

-- Manager for Department 3

```
INSERT INTO Angajat (cod_angajat, prenume, nume_familie, salariu,
data_angajarii, cod_departament, cod_categorie, cod_locatie,
cod_manager)
```

```
VALUES (3, 'Robert', 'Brown', 21000, TO_DATE('2001-10-23', 'YYYY-MM-
DD'), 3, 1, 6, NULL);
```

-- Manager for Department 4

```
INSERT INTO Angajat (cod_angajat, prenume, nume_familie, salariu,
data_angajarii, cod_departament, cod_categorie, cod_locatie,
cod_manager)
```

```
VALUES (4, 'Sophia', 'Wilson', 21500, TO_DATE('2003-12-29', 'YYYY-MM-
DD'), 4, 7, 8, NULL);
```

-- Manager for Department 5

```
INSERT INTO Angajat (cod_angajat, prenume, nume_familie, salariu,
data_angajarii, cod_departament, cod_categorie, cod_locatie,
cod_manager)
```

```
VALUES (5, 'Daniel', 'Taylor', 22000, TO_DATE('2000-09-05', 'YYYY-MM-
DD'), 5, 3, 10, NULL);
```

-- Manager for Department 6

```
INSERT INTO Angajat (cod_angajat, prenume, nume_familie, salariu,  
data_angajarii, cod_departament, cod_categorie, cod_locatie,  
cod_manager)
```

```
VALUES (6, 'Olivia', 'Miller', 22500, TO_DATE('2001-01-30', 'YYYY-MM-DD'),  
6, 6, 12, NULL);
```

-- Manager for Department 7

```
INSERT INTO Angajat (cod_angajat, prenume, nume_familie, salariu,  
data_angajarii, cod_departament, cod_categorie, cod_locatie,  
cod_manager)
```

```
VALUES (7, 'James', 'Anderson', 23000, TO_DATE('2002-03-17', 'YYYY-MM-DD'),  
7, 4, 14, NULL);
```

-- dupa inserare manageri

```
UPDATE Departament SET cod_manager = cod_departament;
```

Tabela ANGAJAT:

-- Department 1

```
INSERT INTO Angajat (cod_angajat, prenume, nume_familie, salariu,
data_angajarii, cod_departament, cod_categorie, cod_locatie,
cod_manager)
VALUES (36, 'Ion', 'Popescu', 5500, TO_DATE('2010-05-23', 'YYYY-MM-DD'),
1, 1, 5, 1);
INSERT INTO Angajat (cod_angajat, prenume, nume_familie, salariu,
data_angajarii, cod_departament, cod_categorie, cod_locatie,
cod_manager)
VALUES (37, 'Maria', 'Ionescu', 8000, TO_DATE('2007-09-15', 'YYYY-MM-
DD'), 1, 3, 3, 2);
INSERT INTO Angajat (cod_angajat, prenume, nume_familie, salariu,
data_angajarii, cod_departament, cod_categorie, cod_locatie,
cod_manager)
VALUES (38, 'Alexandru', 'Stoica', 5000, TO_DATE('2018-02-10', 'YYYY-
MM-DD'), 1, 9, 4, 3);
INSERT INTO Angajat (cod_angajat, prenume, nume_familie, salariu,
data_angajarii, cod_departament, cod_categorie, cod_locatie,
cod_manager)
VALUES (39, 'Andreea', 'Popa', 3500, TO_DATE('2015-11-07', 'YYYY-MM-
DD'), 1, 10, 1, 3);
INSERT INTO Angajat (cod_angajat, prenume, nume_familie, salariu,
data_angajarii, cod_departament, cod_categorie, cod_locatie,
cod_manager)
VALUES (40, 'Victor', 'Dumitru', 12000, TO_DATE('2012-04-18', 'YYYY-MM-
DD'), 1, 5, 2, 1);
```

-- Department 2

```
INSERT INTO Angajat (cod_angajat, prenume, nume_familie, salariu,
data_angajarii, cod_departament, cod_categorie, cod_locatie,
cod_manager)
VALUES (41, 'Ana', 'Baciu', 5000, TO_DATE('2009-12-05', 'YYYY-MM-DD'), 2,
3, 13, 6);
INSERT INTO Angajat (cod_angajat, prenume, nume_familie, salariu,
data_angajarii, cod_departament, cod_categorie, cod_locatie,
cod_manager)
VALUES (42, 'Mihai', 'Pavel', 9500, TO_DATE('2014-07-30', 'YYYY-MM-DD'),
2, 7, 11, 7);
```



```
INSERT INTO Angajat (cod_angajat, prenume, nume_familie, salariu,
data_angajarii, cod_departament, cod_categorie, cod_locatie,
cod_manager)
VALUES (8, 'Andrei', 'Stefan', 8000, TO_DATE('2006-11-19', 'YYYY-MM-DD'),
2, 2, 14, 6);
INSERT INTO Angajat (cod_angajat, prenume, nume_familie, salariu,
data_angajarii, cod_departament, cod_categorie, cod_locatie,
cod_manager)
VALUES (9, 'Mihaela', 'Coman', 3500, TO_DATE('2017-08-12', 'YYYY-MM-
DD'), 2, 8, 12, 4);
INSERT INTO Angajat (cod_angajat, prenume, nume_familie, salariu,
data_angajarii, cod_departament, cod_categorie, cod_locatie,
cod_manager)
VALUES (10, 'George', 'Nita', 11000, TO_DATE('2019-03-26', 'YYYY-MM-
DD'), 2, 5, 15, 7);
```

-- Department 3

```
INSERT INTO Angajat (cod_angajat, prenume, nume_familie, salariu,
data_angajarii, cod_departament, cod_categorie, cod_locatie,
cod_manager)
VALUES (11, 'Laura', 'Munteanu', 4000, TO_DATE('2008-09-17', 'YYYY-MM-
DD'), 3, 2, 1, 7);
INSERT INTO Angajat (cod_angajat, prenume, nume_familie, salariu,
data_angajarii, cod_departament, cod_categorie, cod_locatie,
cod_manager)
VALUES (12, 'Cristian', 'Dinu', 6000, TO_DATE('2010-06-23', 'YYYY-MM-
DD'), 3, 10, 5, 4);
INSERT INTO Angajat (cod_angajat, prenume, nume_familie, salariu,
data_angajarii, cod_departament, cod_categorie, cod_locatie,
cod_manager)
VALUES (13, 'Andreea', 'Popescu', 4500, TO_DATE('2005-11-09', 'YYYY-
MM-DD'), 3, 4, 2, 2);
INSERT INTO Angajat (cod_angajat, prenume, nume_familie, salariu,
data_angajarii, cod_departament, cod_categorie, cod_locatie,
cod_manager)
VALUES (14, 'Ionut', 'Popa', 9000, TO_DATE('2014-03-18', 'YYYY-MM-DD'),
3, 9, 14, 6);
INSERT INTO Angajat (cod_angajat, prenume, nume_familie, salariu,
data_angajarii, cod_departament, cod_categorie, cod_locatie,
cod_manager)
```

```
VALUES (15, 'Elena', 'Radu', 3500, TO_DATE('2019-08-29', 'YYYY-MM-DD'),  
3, 6, 13, 3);
```

```
-- Department 4
```

```
INSERT INTO Angajat (cod_angajat, prenume, nume_familie, salariu,  
data_angajarii, cod_departament, cod_categorie, cod_locatie,  
cod_manager)
```

```
VALUES (16, 'Daniel', 'Stan', 5500, TO_DATE('2006-07-12', 'YYYY-MM-DD'),  
4, 10, 2, 2);
```

```
INSERT INTO Angajat (cod_angajat, prenume, nume_familie, salariu,  
data_angajarii, cod_departament, cod_categorie, cod_locatie,  
cod_manager)
```

```
VALUES (17, 'Ana', 'Popescu', 7500, TO_DATE('2012-02-05', 'YYYY-MM-DD'),  
4, 9, 11, 3);
```

```
INSERT INTO Angajat (cod_angajat, prenume, nume_familie, salariu,  
data_angajarii, cod_departament, cod_categorie, cod_locatie,  
cod_manager)
```

```
VALUES (18, 'Mihai', 'Dumitru', 4000, TO_DATE('2007-09-21', 'YYYY-MM-DD'),  
4, 6, 9, 5);
```

```
INSERT INTO Angajat (cod_angajat, prenume, nume_familie, salariu,  
data_angajarii, cod_departament, cod_categorie, cod_locatie,  
cod_manager)
```

```
VALUES (19, 'Andreea', 'Munteanu', 6000, TO_DATE('2013-11-30', 'YYYY-MM-DD'),  
4, 2, 5, 6);
```

```
INSERT INTO Angajat (cod_angajat, prenume, nume_familie, salariu,  
data_angajarii, cod_departament, cod_categorie, cod_locatie,  
cod_manager)
```

```
VALUES (20, 'Marius', 'Ionescu', 9000, TO_DATE('2018-05-09', 'YYYY-MM-DD'),  
4, 8, 14, 1);
```

```
-- Department 5
```

```
INSERT INTO Angajat (cod_angajat, prenume, nume_familie, salariu,  
data_angajarii, cod_departament, cod_categorie, cod_locatie,  
cod_manager)
```

```
VALUES (21, 'Mihai', 'Popa', 4000, TO_DATE('2005-06-17', 'YYYY-MM-DD'),  
5, 4, 9, 5);
```

```
INSERT INTO Angajat (cod_angajat, prenume, nume_familie, salariu,  
data_angajarii, cod_departament, cod_categorie, cod_locatie,  
cod_manager)
```

```
VALUES (22, 'Andreea', 'Stoica', 5500, TO_DATE('2010-10-29', 'YYYY-MM-DD'), 5, 1, 15, 1);
INSERT INTO Angajat (cod_angajat, prenume, nume_familie, salariu, data_angajarii, cod_departament, cod_categorie, cod_locatie, cod_manager)
VALUES (23, 'Ionut', 'Munteanu', 7500, TO_DATE('2008-08-12', 'YYYY-MM-DD'), 5, 2, 7, 2);
INSERT INTO Angajat (cod_angajat, prenume, nume_familie, salariu, data_angajarii, cod_departament, cod_categorie, cod_locatie, cod_manager)
VALUES (24, 'Elena', 'Popescu', 6000, TO_DATE('2013-04-24', 'YYYY-MM-DD'), 5, 3, 12, 6);
INSERT INTO Angajat (cod_angajat, prenume, nume_familie, salariu, data_angajarii, cod_departament, cod_categorie, cod_locatie, cod_manager)
VALUES (25, 'Cristian', 'Dinu', 9000, TO_DATE('2019-09-07', 'YYYY-MM-DD'), 5, 5, 6, 4);
```

-- Department 6

```
INSERT INTO Angajat (cod_angajat, prenume, nume_familie, salariu, data_angajarii, cod_departament, cod_categorie, cod_locatie, cod_manager)
VALUES (26, 'Laura', 'Radu', 4000, TO_DATE('2006-07-21', 'YYYY-MM-DD'), 6, 7, 11, 2);
INSERT INTO Angajat (cod_angajat, prenume, nume_familie, salariu, data_angajarii, cod_departament, cod_categorie, cod_locatie, cod_manager)
VALUES (27, 'Cristina', 'Dumitru', 5500, TO_DATE('2011-12-15', 'YYYY-MM-DD'), 6, 1, 19, 1);
INSERT INTO Angajat (cod_angajat, prenume, nume_familie, salariu, data_angajarii, cod_departament, cod_categorie, cod_locatie, cod_manager)
VALUES (28, 'Andrei', 'Popa', 7500, TO_DATE('2008-09-03', 'YYYY-MM-DD'), 6, 3, 14, 4);
INSERT INTO Angajat (cod_angajat, prenume, nume_familie, salariu, data_angajarii, cod_departament, cod_categorie, cod_locatie, cod_manager)
VALUES (29, 'Mihaela', 'Ionescu', 6000, TO_DATE('2014-03-11', 'YYYY-MM-DD'), 6, 4, 18, 7);
```

```
INSERT INTO Angajat (cod_angajat, prenume, nume_familie, salariu,
data_angajarii, cod_departament, cod_categorie, cod_locatie,
cod_manager)
VALUES (30, 'George', 'Munteanu', 9000, TO_DATE('2020-11-19', 'YYYY-
MM-DD'), 6, 7, 8, 5);
```

-- Department 7

```
INSERT INTO Angajat (cod_angajat, prenume, nume_familie, salariu,
data_angajarii, cod_departament, cod_categorie, cod_locatie,
cod_manager)
VALUES (31, 'Alexandru', 'Popescu', 4000, TO_DATE('2011-08-22', 'yyyy-
mm-dd'), 7, 2, 13, 5);
INSERT INTO Angajat (cod_angajat, prenume, nume_familie, salariu,
data_angajarii, cod_departament, cod_categorie, cod_locatie,
cod_manager)
VALUES (32, 'Gabriela', 'Stoica', 5500, TO_DATE('2017-12-03', 'yyyy-mm-
dd'), 7, 10, 10, 4);
INSERT INTO Angajat (cod_angajat, prenume, nume_familie, salariu,
data_angajarii, cod_departament, cod_categorie, cod_locatie,
cod_manager)
VALUES (33, 'Mihai', 'Munteanu', 7500, TO_DATE('2013-10-18', 'yyyy-mm-
dd'), 7, 3, 17, 1);
INSERT INTO Angajat (cod_angajat, prenume, nume_familie, salariu,
data_angajarii, cod_departament, cod_categorie, cod_locatie,
cod_manager)
VALUES (34, 'Andreea', 'Popa', 6000, TO_DATE('2016-05-07', 'yyyy-mm-dd'),
7, 8, 6, 2);
INSERT INTO Angajat (cod_angajat, prenume, nume_familie, salariu,
data_angajarii, cod_departament, cod_categorie, cod_locatie,
cod_manager)
VALUES (35, 'Cristian', 'Dinu', 9000, TO_DATE('2019-09-21', 'yyyy-mm-dd'),
7, 5, 3, 3);
```

```

6 -- INSERT INTO Angajat (cod_angajat, prenume, nume_familie, salariu, cod_departament, cod_categorie, cod_locatie)
7 -- VALUES (38, 'Alexandru', 'Stoica', 5000, 1, 9, 4, 3);
8 -- INSERT INTO Angajat (cod_angajat, prenume, nume_familie, salariu, cod_departament, cod_categorie, cod_locatie)
9 -- VALUES (39, 'Andreea', 'Popa', 3500, 1, 10, 1, 3);
10 -- INSERT INTO Angajat (cod_angajat, prenume, nume_familie, salariu, cod_departament, cod_categorie, cod_locatie)
11 -- VALUES (40, 'Victor', 'Dumitru', 12000, 1, 5, 2, 1);
12 -- INSERT INTO Angajat (cod_angajat, prenume, nume_familie, salariu, cod_departament, cod_categorie, cod_locatie)
13 -- VALUES (41, 'Ana', 'Baciu', 5000, 2, 3, 13, 6);
14 -- INSERT INTO Angajat (cod_angajat, prenume, nume_familie, salariu, cod_departament, cod_categorie, cod_locatie)
15 -- VALUES (42, 'Mihai', 'Pavel', 9500, 2, 7, 11, 7);
16
17 select * from angajat;

```

COD_ANGAJAT	PRENUME	NUME_FAMILIE	SALARIU	COD_DEPARTAMENT	COD_CATEGORIE	COD_LOCATIE	COD_MANAGER
1	John	Smith	20000	1	5	3	-
2	Maria	Johnson	20500	2	2	4	-
3	Robert	Brown	21000	3	1	6	-
4	Sophia	Wilson	21500	4	7	8	-
5	Daniel	Taylor	22000	5	3	10	-
6	Olivia	Miller	22500	6	6	12	-
7	James	Anderson	23000	7	4	14	-
8	Andrei	Stefan	8000	2	2	14	6
9	Mihaela	Coman	3500	2	8	12	4

Tabela FURNIZOR:

-- Furnizor for Location 1

```
INSERT INTO Furnizor (cod_furnizor, nume, cod_locatie)
VALUES (1, 'Kaufland', 1);
```

-- Furnizor for Location 2

```
INSERT INTO Furnizor (cod_furnizor, nume, cod_locatie)
VALUES (2, 'Carrefour', 2);
```

-- Furnizor for Location 3

```
INSERT INTO Furnizor (cod_furnizor, nume, cod_locatie)
VALUES (3, 'Metro', 3);
```

-- Furnizor for Location 4

```
INSERT INTO Furnizor (cod_furnizor, nume, cod_locatie)
VALUES (4, 'Lidl', 4);
```

-- Furnizor for Location 5

```
INSERT INTO Furnizor (cod_furnizor, nume, cod_locatie)
VALUES (5, 'Auchan', 5);
```

-- Furnizor for Location 6

```
INSERT INTO Furnizor (cod_furnizor, nume, cod_locatie)
```

```
VALUES (6, 'Mega Image', 6);
```

```
-- Furnizor for Location 7
```

```
INSERT INTO Furnizor (cod_furnizor, nume, cod_locatie)
```

```
VALUES (7, 'Selgros', 7);
```

```
-- Furnizor for Location 8
```

```
INSERT INTO Furnizor (cod_furnizor, nume, cod_locatie)
```

```
VALUES (8, 'Mega Store', 8);
```

```
-- Furnizor for Location 9
```

```
INSERT INTO Furnizor (cod_furnizor, nume, cod_locatie)
```

```
VALUES (9, 'Eco Market', 9);
```

```
-- Furnizor for Location 10
```

```
INSERT INTO Furnizor (cod_furnizor, nume, cod_locatie)
```

```
VALUES (10, 'Super Mart', 10);
```

```
-- Furnizor for Location 11
```

```
INSERT INTO Furnizor (cod_furnizor, nume, cod_locatie)
```

```
VALUES (11, 'Fresh Foods', 11);
```

```
-- Furnizor for Location 12
```

```
INSERT INTO Furnizor (cod_furnizor, nume, cod_locatie)
VALUES (12, 'Discount Depot', 12);
```

-- Furnizor for Location 13

```
INSERT INTO Furnizor (cod_furnizor, nume, cod_locatie)
VALUES (13, 'Market Square', 13);
```

-- Furnizor for Location 14

```
INSERT INTO Furnizor (cod_furnizor, nume, cod_locatie)
VALUES (14, 'Value Mart', 14);
```

-- Furnizor for Location 15

```
INSERT INTO Furnizor (cod_furnizor, nume, cod_locatie)
VALUES (15, 'Fresh Mart', 15);
```

-- Furnizor for Location 16

```
INSERT INTO Furnizor (cod_furnizor, nume, cod_locatie)
VALUES (16, 'City Market', 16);
```

-- Furnizor for Location 17

```
INSERT INTO Furnizor (cod_furnizor, nume, cod_locatie)
VALUES (17, 'Gourmet Express', 17);
```


-- Furnizor for Location 18

INSERT INTO Furnizor (cod_furnizor, nume, cod_locatie)

VALUES (18, 'Healthy Choices', 18);

-- Furnizor for Location 19

INSERT INTO Furnizor (cod_furnizor, nume, cod_locatie)

VALUES (19, 'Farmers Market', 19);

-- Furnizor for Location 20

INSERT INTO Furnizor (cod_furnizor, nume, cod_locatie)

VALUES (20, 'Quality Foods', 20);

```
66 -- INSERT INTO Furnizor (cod_furnizor, nume, cod_locatie)
67 -- VALUES (17, 'Gourmet Express', 17);
68
69 -- -- Furnizor for Location 18
70 -- INSERT INTO Furnizor (cod_furnizor, nume, cod_locatie)
71 -- VALUES (18, 'Healthy Choices', 18);
72
73 -- -- Furnizor for Location 19
74 -- INSERT INTO Furnizor (cod_furnizor, nume, cod_locatie)
75 -- VALUES (19, 'Farmers Market', 19);
76
77 -- -- Furnizor for Location 20
78 -- INSERT INTO Furnizor (cod_furnizor, nume, cod_locatie)
79 -- VALUES (20, 'Quality Foods', 20);
80 select * from furnizor;
```

COD_FURNIZOR	NUME	COD_LOCATIE
1	Kaufland	1
2	Carrefour	2
3	Metro	3
4	Lidl	4
5	Auchan	5
6	Mega Image	6
7	Selgros	7
8	Mega Store	8

Tabela PRODUS:

-- Produse pentru Categoria "Electronice"

```
INSERT INTO Produs (cod_produs, nume, pret, cod_categorie,  
cod_furnizor)
```

```
VALUES (1, 'Laptop', 2499.99, 1, 1);
```

```
INSERT INTO Produs (cod_produs, nume, pret, cod_categorie,  
cod_furnizor)
```

```
VALUES (2, 'Smartphone', 1299.99, 1, 6);
```

```
INSERT INTO Produs (cod_produs, nume, pret, cod_categorie,  
cod_furnizor)
```

```
VALUES (3, 'Tabletă', 899.99, 1, 7);
```

```
INSERT INTO Produs (cod_produs, nume, pret, cod_categorie,  
cod_furnizor)
```

```
VALUES (4, 'Televizor', 1999.99, 1, 2);
```

```
INSERT INTO Produs (cod_produs, nume, pret, cod_categorie,  
cod_furnizor)
```

```
VALUES (5, 'Cameră foto', 799.99, 1, 3);
```

-- Produse pentru Categoria "Îmbrăcăminte"

```
INSERT INTO Produs (cod_produs, nume, pret, cod_categorie,  
cod_furnizor)
```

```
VALUES (6, 'Cămașă', 99.99, 2, 8);
```

```
INSERT INTO Produs (cod_produs, nume, pret, cod_categorie,  
cod_furnizor)
```

```
VALUES (7, 'Pantaloni', 79.99, 2, 7);
```

```
INSERT INTO Produs (cod_produs, nume, pret, cod_categorie,  
cod_furnizor)
```

```
VALUES (8, 'Rochie', 149.99, 2, 19);
```

```
INSERT INTO Produs (cod_produs, nume, pret, cod_categorie,  
cod_furnizor)
```

```
VALUES (9, 'Geacă', 199.99, 2, 2);
```

```
INSERT INTO Produs (cod_produs, nume, pret, cod_categorie,  
cod_furnizor)
```

```
VALUES (10, 'Papuci', 49.99, 2, 5);
```

```
-- Produse pentru Categoria "Mobilă"
```

```
INSERT INTO Produs (cod_produs, nume, pret, cod_categorie,  
cod_furnizor)
```

```
VALUES (11, 'Canapea', 999.99, 3, 3);
```

```
INSERT INTO Produs (cod_produs, nume, pret, cod_categorie,  
cod_furnizor)
```

```
VALUES (12, 'Masă de dining', 799.99, 3, 9);
```

```
INSERT INTO Produs (cod_produs, nume, pret, cod_categorie,  
cod_furnizor)
```

```
VALUES (13, 'Scaun de birou', 199.99, 3, 4);
```

```
INSERT INTO Produs (cod_produs, nume, pret, cod_categorie,  
cod_furnizor)
```

```
VALUES (14, 'Dulap', 1499.99, 3, 2);
```

```
INSERT INTO Produs (cod_produs, nume, pret, cod_categorie,  
cod_furnizor)
```

```
VALUES (15, 'Pat', 699.99, 3, 1);
```

-- Produse pentru Categoria "Cosmetice"

INSERT INTO Produs (cod_produs, nume, pret, cod_categorie, cod_furnizor)

VALUES (16, 'Fond de ten', 59.99, 4, 12);

INSERT INTO Produs (cod_produs, nume, pret, cod_categorie, cod_furnizor)

VALUES (17, 'Ruj', 19.99, 4, 14);

INSERT INTO Produs (cod_produs, nume, pret, cod_categorie, cod_furnizor)

VALUES (18, 'Mascara', 29.99, 4, 13);

INSERT INTO Produs (cod_produs, nume, pret, cod_categorie, cod_furnizor)

VALUES (19, 'Ser facial', 99.99, 4, 9);

INSERT INTO Produs (cod_produs, nume, pret, cod_categorie, cod_furnizor)

VALUES (20, 'Parfum', 149.99, 4, 16);

-- Produse pentru Categoria "Alimente"

INSERT INTO Produs (cod_produs, nume, pret, cod_categorie, cod_furnizor)

VALUES (21, 'Cafea', 19.99, 5, 8);

INSERT INTO Produs (cod_produs, nume, pret, cod_categorie, cod_furnizor)

VALUES (22, 'Ciocolată', 4.99, 5, 11);

INSERT INTO Produs (cod_produs, nume, pret, cod_categorie, cod_furnizor)

```
VALUES (23, 'Ulei de măsline', 9.99, 5, 19);
```

```
INSERT INTO Produs (cod_produs, nume, pret, cod_categorie,  
cod_furnizor)
```

```
VALUES (24, 'Biscuiți', 2.99, 5, 16);
```

```
INSERT INTO Produs (cod_produs, nume, pret, cod_categorie,  
cod_furnizor)
```

```
VALUES (25, 'Suc', 3.99, 5, 14);
```

```
-- Produse pentru Categoria "Jucării"
```

```
INSERT INTO Produs (cod_produs, nume, pret, cod_categorie,  
cod_furnizor)
```

```
VALUES (26, 'Păpușă', 29.99, 6, 20);
```

```
INSERT INTO Produs (cod_produs, nume, pret, cod_categorie,  
cod_furnizor)
```

```
VALUES (27, 'Mașinuță', 24.99, 6, 9);
```

```
INSERT INTO Produs (cod_produs, nume, pret, cod_categorie,  
cod_furnizor)
```

```
VALUES (28, 'Puzzle', 14.99, 6, 15);
```

```
INSERT INTO Produs (cod_produs, nume, pret, cod_categorie,  
cod_furnizor)
```

```
VALUES (29, 'Lego', 49.99, 6, 10);
```

```
INSERT INTO Produs (cod_produs, nume, pret, cod_categorie,  
cod_furnizor)
```

```
VALUES (30, 'Joc de societate', 39.99, 6, 17);
```

```
-- Produse pentru Categoria "Articole"
```

```
INSERT INTO Produs (cod_produs, nume, pret, cod_categorie,  
cod_furnizor)
```

```
VALUES (31, 'Caiet', 4.99, 7, 3);
```

```
INSERT INTO Produs (cod_produs, nume, pret, cod_categorie,  
cod_furnizor)
```

```
VALUES (32, 'Pix', 1.99, 7, 8);
```

```
INSERT INTO Produs (cod_produs, nume, pret, cod_categorie,  
cod_furnizor)
```

```
VALUES (33, 'Adeziv', 2.99, 7, 5);
```

```
INSERT INTO Produs (cod_produs, nume, pret, cod_categorie,  
cod_furnizor)
```

```
VALUES (34, 'Creion colorat', 0.99, 7, 7);
```

```
INSERT INTO Produs (cod_produs, nume, pret, cod_categorie,  
cod_furnizor)
```

```
VALUES (35, 'Foarfecă', 3.99, 7, 16);
```

```
-- Produse pentru Categoria "Bijuterii"
```

```
INSERT INTO Produs (cod_produs, nume, pret, cod_categorie,  
cod_furnizor)
```

```
VALUES (36, 'Inel', 49.99, 8, 8);
```

```
INSERT INTO Produs (cod_produs, nume, pret, cod_categorie,  
cod_furnizor)
```

```
VALUES (37, 'Cercei', 39.99, 8, 2);
```

```
INSERT INTO Produs (cod_produs, nume, pret, cod_categorie,  
cod_furnizor)
```

```
VALUES (38, 'Colier', 59.99, 8, 19);
```

```
INSERT INTO Produs (cod_produs, nume, pret, cod_categorie,  
cod_furnizor)
```

```
VALUES (39, 'Brățară', 29.99, 8, 1);
```

```
INSERT INTO Produs (cod_produs, nume, pret, cod_categorie,  
cod_furnizor)
```

```
VALUES (40, 'Ceas', 99.99, 8, 14);
```

```
-- Produse pentru Categoria "Unelte"
```

```
INSERT INTO Produs (cod_produs, nume, pret, cod_categorie,  
cod_furnizor)
```

```
VALUES (41, 'Șurubelniță', 9.99, 9, 20);
```

```
INSERT INTO Produs (cod_produs, nume, pret, cod_categorie,  
cod_furnizor)
```

```
VALUES (42, 'Foarfecă de grădină', 14.99, 9, 9);
```

```
INSERT INTO Produs (cod_produs, nume, pret, cod_categorie,  
cod_furnizor)
```

```
VALUES (43, 'Ciocan', 19.99, 9, 15);
```

```
INSERT INTO Produs (cod_produs, nume, pret, cod_categorie,  
cod_furnizor)
```

```
VALUES (44, 'Ruletă', 7.99, 9, 10);
```

```
INSERT INTO Produs (cod_produs, nume, pret, cod_categorie,  
cod_furnizor)
```

```
VALUES (45, 'Mașină de găurit', 99.99, 9, 17);
```

```
-- Produse pentru Categoria "Rechizite"
```

INSERT INTO Produs (cod_produs, nume, pret, cod_categorie, cod_furnizor)

VALUES (46, 'Penar', 9.99, 10, 8);

INSERT INTO Produs (cod_produs, nume, pret, cod_categorie, cod_furnizor)

VALUES (47, 'Creion HB', 0.99, 10, 11);

INSERT INTO Produs (cod_produs, nume, pret, cod_categorie, cod_furnizor)

VALUES (48, 'Radieră', 1.99, 10, 19);

INSERT INTO Produs (cod_produs, nume, pret, cod_categorie, cod_furnizor)

VALUES (49, 'Caiet dictando', 2.99, 10, 16);

INSERT INTO Produs (cod_produs, nume, pret, cod_categorie, cod_furnizor)

VALUES (50, 'Stilou', 4.99, 10, 14);

```
108
109 -- -- Produse pentru Categoria "Rechizite"
110 -- INSERT INTO Produs (cod_produs, nume, pret, cod_categorie, cod_furnizor)
111 -- VALUES (46, 'Penar', 9.99, 10, 8);
112 -- INSERT INTO Produs (cod_produs, nume, pret, cod_categorie, cod_furnizor)
113 -- VALUES (47, 'Creion HB', 0.99, 10, 11);
114 -- INSERT INTO Produs (cod_produs, nume, pret, cod_categorie, cod_furnizor)
115 -- VALUES (48, 'Radieră', 1.99, 10, 19);
116 -- INSERT INTO Produs (cod_produs, nume, pret, cod_categorie, cod_furnizor)
117 -- VALUES (49, 'Caiet dictando', 2.99, 10, 16);
118 -- INSERT INTO Produs (cod_produs, nume, pret, cod_categorie, cod_furnizor)
119 -- VALUES (50, 'Stilou', 4.99, 10, 14);
120
121 select * from produs;
```

COD_PRODUS	NUME	PRET	COD_CATEGORIE	COD_FURNIZOR
1	Laptop	2499.99	1	1
2	Smartphone	1299.99	1	6
3	Tabletă	899.99	1	7
4	Televizor	1999.99	1	2
5	Cameră foto	799.99	1	3

Tabela UTILIZATOR:

-- Utilizatori

```
INSERT INTO Utilizator (cod_utilizator, nume_utilizator, prenume,  
nume_familie, cod_locatie)
```

```
VALUES (1, 'popescu_1', 'Alexandra', 'Popescu', 1);
```

```
INSERT INTO Utilizator (cod_utilizator, nume_utilizator, prenume,  
nume_familie, cod_locatie)
```

```
VALUES (2, 'ionescu_2', 'Andrei', 'Ionescu', 2);
```

```
INSERT INTO Utilizator (cod_utilizator, nume_utilizator, prenume,  
nume_familie, cod_locatie)
```

```
VALUES (3, 'vasilescu_3', 'Maria', 'Vasilescu', 3);
```

```
INSERT INTO Utilizator (cod_utilizator, nume_utilizator, prenume,  
nume_familie, cod_locatie)
```

```
VALUES (4, 'georgescu_4', 'Mihai', 'Georgescu', 4);
```

```
INSERT INTO Utilizator (cod_utilizator, nume_utilizator, prenume,  
nume_familie, cod_locatie)
```

```
VALUES (5, 'popa_5', 'Elena', 'Popa', 5);
```

```
INSERT INTO Utilizator (cod_utilizator, nume_utilizator, prenume,  
nume_familie, cod_locatie)
```

```
VALUES (6, 'dumitru_6', 'Andreea', 'Dumitru', 6);
```

```
INSERT INTO Utilizator (cod_utilizator, nume_utilizator, prenume,  
nume_familie, cod_locatie)
```

```
VALUES (7, 'stoica_7', 'Adrian', 'Stoica', 7);
```

```
INSERT INTO Utilizator (cod_utilizator, nume_utilizator, prenume,  
nume_familie, cod_locatie)
```

```
VALUES (8, 'constantin_8', 'Cristina', 'Constantin', 8);
```

```
INSERT INTO Utilizator (cod_utilizator, nume_utilizator, prenume,  
nume_familie, cod_locatie)
```

```
VALUES (9, 'ilie_9', 'Gabriel', 'Ilie', 9);
```

```
INSERT INTO Utilizator (cod_utilizator, nume_utilizator, prenume,  
nume_familie, cod_locatie)
```

```
VALUES (10, 'dobre_10', 'Andrei', 'Dobre', 10);
```

```
INSERT INTO Utilizator (cod_utilizator, nume_utilizator, prenume,  
nume_familie, cod_locatie)
```

```
VALUES (11, 'popovici_11', 'Diana', 'Popovici', 11);
```

```
INSERT INTO Utilizator (cod_utilizator, nume_utilizator, prenume,  
nume_familie, cod_locatie)
```

```
VALUES (12, 'marinescu_12', 'Ionut', 'Marinescu', 12);
```

```
INSERT INTO Utilizator (cod_utilizator, nume_utilizator, prenume,  
nume_familie, cod_locatie)
```

```
VALUES (13, 'mihai_13', 'Andreea', 'Mihai', 13);
```

```
INSERT INTO Utilizator (cod_utilizator, nume_utilizator, prenume,  
nume_familie, cod_locatie)
```

```
VALUES (14, 'sandu_14', 'Ana', 'Sandu', 14);
```

```
INSERT INTO Utilizator (cod_utilizator, nume_utilizator, prenume,  
nume_familie, cod_locatie)
```

```
VALUES (15, 'radu_15', 'Marius', 'Radu', 15);
```

```
INSERT INTO Utilizator (cod_utilizator, nume_utilizator, prenume,  
nume_familie, cod_locatie)
```

```
VALUES (16, 'petrescu_16', 'Elena', 'Petrescu', 16);
```

```
INSERT INTO Utilizator (cod_utilizator, nume_utilizator, prenume,  
nume_familie, cod_locatie)
```

```
VALUES (17, 'stan_17', 'Gabriel', 'Stan', 17);
```

INSERT INTO Utilizator (cod_utilizator, nume_utilizator, prenume, nume_familie, cod_locatie)

VALUES (18, 'andrei_18', 'Ioana', 'Andrei', 18);

INSERT INTO Utilizator (cod_utilizator, nume_utilizator, prenume, nume_familie, cod_locatie)

VALUES (19, 'matei_19', 'Razvan', 'Matei', 19);

INSERT INTO Utilizator (cod_utilizator, nume_utilizator, prenume, nume_familie, cod_locatie)

VALUES (20, 'gheorghe_20', 'Alexandru', 'Gheorghe', 20);

```
29 -- VALUES (14, 'sandu_14', 'Ana', 'Sandu', 14);
30 -- INSERT INTO Utilizator (cod_utilizator, nume_utilizator, prenume, nume_familie, cod_locatie)
31 -- VALUES (15, 'radu_15', 'Marius', 'Radu', 15);
32 -- INSERT INTO Utilizator (cod_utilizator, nume_utilizator, prenume, nume_familie, cod_locatie)
33 -- VALUES (16, 'petrescu_16', 'Elena', 'Petrescu', 16);
34 -- INSERT INTO Utilizator (cod_utilizator, nume_utilizator, prenume, nume_familie, cod_locatie)
35 -- VALUES (17, 'stan_17', 'Gabriel', 'Stan', 17);
36 -- INSERT INTO Utilizator (cod_utilizator, nume_utilizator, prenume, nume_familie, cod_locatie)
37 -- VALUES (18, 'andrei_18', 'Ioana', 'Andrei', 18);
38 -- INSERT INTO Utilizator (cod_utilizator, nume_utilizator, prenume, nume_familie, cod_locatie)
39 -- VALUES (19, 'matei_19', 'Razvan', 'Matei', 19);
40 -- INSERT INTO Utilizator (cod_utilizator, nume_utilizator, prenume, nume_familie, cod_locatie)
41 -- VALUES (20, 'gheorghe_20', 'Alexandru', 'Gheorghe', 20);
42
43 select * from utilizator;
```

COD_UTILIZATOR	NUME_UTILIZATOR	PRENUME	NUME_FAMILIE	COD_LOCATIE
1	popescu_1	Alexandra	Popescu	1
2	ionescu_2	Andrei	Ionescu	2
3	vasilescu_3	Maria	Vasilescu	3
4	georgescu_4	Mihai	Georgescu	4
5	popa_5	Elena	Popa	5

Tabela COMANDA:

-- Comenzi

```
INSERT INTO Comanda (cod_comanda, discount, cod_utilizator)
VALUES (1, 0.25, 5);
INSERT INTO Comanda (cod_comanda, discount, cod_utilizator)
VALUES (2, 0.12, 17);
INSERT INTO Comanda (cod_comanda, discount, cod_utilizator)
VALUES (3, 0.45, 9);
INSERT INTO Comanda (cod_comanda, discount, cod_utilizator)
VALUES (4, 0.33, 12);
INSERT INTO Comanda (cod_comanda, discount, cod_utilizator)
VALUES (5, 0.58, 4);
INSERT INTO Comanda (cod_comanda, discount, cod_utilizator)
VALUES (6, 0.27, 19);
INSERT INTO Comanda (cod_comanda, discount, cod_utilizator)
VALUES (7, 0.61, 3);
INSERT INTO Comanda (cod_comanda, discount, cod_utilizator)
VALUES (8, 0.15, 10);
INSERT INTO Comanda (cod_comanda, discount, cod_utilizator)
VALUES (9, 0.50, 7);
INSERT INTO Comanda (cod_comanda, discount, cod_utilizator)
VALUES (10, 0.40, 15);
```

```
11 -- VALUES (5, 0.58, 4);
12 -- INSERT INTO Comanda (cod_comanda, discount, cod_utilizator)
13 -- VALUES (6, 0.27, 19);
14 -- INSERT INTO Comanda (cod_comanda, discount, cod_utilizator)
15 -- VALUES (7, 0.61, 3);
16 -- INSERT INTO Comanda (cod_comanda, discount, cod_utilizator)
17 -- VALUES (8, 0.15, 10);
18 -- INSERT INTO Comanda (cod_comanda, discount, cod_utilizator)
19 -- VALUES (9, 0.50, 7);
20 -- INSERT INTO Comanda (cod_comanda, discount, cod_utilizator)
21 -- VALUES (10, 0.40, 15);
22
23 select * from comanda;
```

COD_COMANDA	DISCOUNT	COD_UTILIZATOR
1	.25	5
2	.12	17
3	.45	9
4	.33	12
5	.58	4

Tabela REVIEW:

-- Reviews

```
INSERT INTO Review (cod_review, descriere, rating, cod_produș,
cod_utilizator)
```

```
VALUES (1, 'Bun produș!', 4, 15, 8);
```

```
INSERT INTO Review (cod_review, descriere, rating, cod_produș,
cod_utilizator)
```

```
VALUES (2, 'Slab produș...', 2, 27, 12);
```

```
INSERT INTO Review (cod_review, descriere, rating, cod_produș,
cod_utilizator)
```

```
VALUES (3, 'Produș excelent!', 5, 6, 3);
```

```
INSERT INTO Review (cod_review, descriere, rating, cod_produș,
cod_utilizator)
```

```
VALUES (4, 'Nu recomand...', 1, 42, 19);
```

```
INSERT INTO Review (cod_review, descriere, rating, cod_produș,
cod_utilizator)
```

```
VALUES (5, 'Super calitate!', 5, 10, 5);
```

```
INSERT INTO Review (cod_review, descriere, rating, cod_produș,
cod_utilizator)
```

```
VALUES (6, 'Mai mult decăt mă așteptam.', 4, 35, 11);
```

```
INSERT INTO Review (cod_review, descriere, rating, cod_produș,
cod_utilizator)
```

```
VALUES (7, 'Dezamăgitor...', 2, 20, 2);
```

```
INSERT INTO Review (cod_review, descriere, rating, cod_produș,
cod_utilizator)
```

```
VALUES (8, 'Recomand cu încredere!', 5, 17, 14);
```

```
INSERT INTO Review (cod_review, descriere, rating, cod_produș,
cod_utilizator)
```

VALUES (9, 'Calitate medie.', 3, 8, 9);

INSERT INTO Review (cod_review, descriere, rating, cod_produș, cod_utilizator)

VALUES (10, 'Foarte rau...', 1, 49, 16);

INSERT INTO Review (cod_review, descriere, rating, cod_produș, cod_utilizator)

VALUES (11, 'Produș ok.', 3, 12, 7);

INSERT INTO Review (cod_review, descriere, rating, cod_produș, cod_utilizator)

VALUES (12, 'Foarte bun raport calitate-pret.', 4, 33, 13);

INSERT INTO Review (cod_review, descriere, rating, cod_produș, cod_utilizator)

VALUES (13, 'Nu a meritat investiția...', 2, 24, 18);

INSERT INTO Review (cod_review, descriere, rating, cod_produș, cod_utilizator)

VALUES (14, 'Satisfăcut de produș.', 3, 9, 4);

INSERT INTO Review (cod_review, descriere, rating, cod_produș, cod_utilizator)

VALUES (15, 'Slab calitate.', 2, 38, 15);

INSERT INTO Review (cod_review, descriere, rating, cod_produș, cod_utilizator)

VALUES (16, 'Excelent produș!', 5, 14, 10);

INSERT INTO Review (cod_review, descriere, rating, cod_produș, cod_utilizator)

VALUES (17, 'Nu recomand acest produș...', 1, 45, 17);

INSERT INTO Review (cod_review, descriere, rating, cod_produș, cod_utilizator)

VALUES (18, 'M-a dezamăgit...', 2, 19, 6);

INSERT INTO Review (cod_review, descriere, rating, cod_produs, cod_utilizator)

VALUES (19, 'Produs de calitate superioara!', 5, 4, 20);

INSERT INTO Review (cod_review, descriere, rating, cod_produs, cod_utilizator)

VALUES (20, 'Nu a corespuns asteptarilor...', 2, 31, 1);

```
29 -- VALUES (14, 'Satisfacut de produs.', 3, 9, 4);
30 -- INSERT INTO Review (cod_review, descriere, rating, cod_produs, cod_utilizator)
31 -- VALUES (15, 'Slab calitate.', 2, 38, 15);
32 -- INSERT INTO Review (cod_review, descriere, rating, cod_produs, cod_utilizator)
33 -- VALUES (16, 'Excelent produs!', 5, 14, 10);
34 -- INSERT INTO Review (cod_review, descriere, rating, cod_produs, cod_utilizator)
35 -- VALUES (17, 'Nu recomand acest produs...', 1, 45, 17);
36 -- INSERT INTO Review (cod_review, descriere, rating, cod_produs, cod_utilizator)
37 -- VALUES (18, 'M-a dezamagit...', 2, 19, 6);
38 -- INSERT INTO Review (cod_review, descriere, rating, cod_produs, cod_utilizator)
39 -- VALUES (19, 'Produs de calitate superioara!', 5, 23, 20);
40 -- INSERT INTO Review (cod_review, descriere, rating, cod_produs, cod_utilizator)
41 -- VALUES (20, 'Nu a corespuns asteptarilor...', 2, 31, 1);
42
43 select * from review;
```

COD_REVIEW	DESCRIERE	RATING	COD_PRODUS	COD_UTILIZATOR
1	Bun produs!	4	15	8
2	Slab produs...	2	27	12
3	Produs excelent!	5	6	3
4	Nu recomand...	1	42	19
5	Super calitate!	5	10	5

Tabela PRODUS_se_afla_in_COMANDA:

-- Produs_se_afla_in_Comanda

--1

```
INSERT INTO Produs_se_afla_in_Comanda (cod_comanda, cod_produs)
VALUES (1, 4);
```

```
INSERT INTO Produs_se_afla_in_Comanda (cod_comanda, cod_produs)
VALUES (1, 9);
```

```
INSERT INTO Produs_se_afla_in_Comanda (cod_comanda, cod_produs)
VALUES (1, 14);
```

--2

```
INSERT INTO Produs_se_afla_in_Comanda (cod_comanda, cod_produs)
VALUES (2, 21);
```

```
INSERT INTO Produs_se_afla_in_Comanda (cod_comanda, cod_produs)
VALUES (2, 27);
```

```
INSERT INTO Produs_se_afla_in_Comanda (cod_comanda, cod_produs)
VALUES (2, 4);
```

--3

```
INSERT INTO Produs_se_afla_in_Comanda (cod_comanda, cod_produs)
VALUES (3, 4);
```

```
INSERT INTO Produs_se_afla_in_Comanda (cod_comanda, cod_produs)
VALUES (3, 15);
```

```
INSERT INTO Produs_se_afla_in_Comanda (cod_comanda, cod_produs)
VALUES (3, 25);
```

--4

```
INSERT INTO Produs_se_afla_in_Comanda (cod_comanda, cod_produs)
VALUES (4, 1);
```



```
INSERT INTO Produs_se_afla_in_Comanda (cod_comanda, cod_produs)
VALUES (4, 7);
```

```
INSERT INTO Produs_se_afla_in_Comanda (cod_comanda, cod_produs)
VALUES (4, 4);
```

--5

```
INSERT INTO Produs_se_afla_in_Comanda (cod_comanda, cod_produs)
VALUES (5, 18);
```

```
INSERT INTO Produs_se_afla_in_Comanda (cod_comanda, cod_produs)
VALUES (5, 4);
```

```
INSERT INTO Produs_se_afla_in_Comanda (cod_comanda, cod_produs)
VALUES (5, 33);
```

--6

```
INSERT INTO Produs_se_afla_in_Comanda (cod_comanda, cod_produs)
VALUES (6, 2);
```

```
INSERT INTO Produs_se_afla_in_Comanda (cod_comanda, cod_produs)
VALUES (6, 11);
```

```
INSERT INTO Produs_se_afla_in_Comanda (cod_comanda, cod_produs)
VALUES (6, 4);
```

--7

```
INSERT INTO Produs_se_afla_in_Comanda (cod_comanda, cod_produs)
VALUES (7, 4);
```

```
INSERT INTO Produs_se_afla_in_Comanda (cod_comanda, cod_produs)
VALUES (7, 13);
```

```
INSERT INTO Produs_se_afla_in_Comanda (cod_comanda, cod_produs)
VALUES (7, 24);
```

--8

```
INSERT INTO Produs_se_afla_in_Comanda (cod_comanda, cod_produs)
VALUES (8, 10);
```

```
INSERT INTO Produs_se_afla_in_Comanda (cod_comanda, cod_produs)
VALUES (8, 4);
```

```
INSERT INTO Produs_se_afila_in_Comanda (cod_comanda, cod_produs)
VALUES (8, 28);
```

--9

```
INSERT INTO Produs_se_afila_in_Comanda (cod_comanda, cod_produs)
VALUES (9, 9);
```

```
INSERT INTO Produs_se_afila_in_Comanda (cod_comanda, cod_produs)
VALUES (9, 4);
```

```
INSERT INTO Produs_se_afila_in_Comanda (cod_comanda, cod_produs)
VALUES (9, 23);
```

--10

```
INSERT INTO Produs_se_afila_in_Comanda (cod_comanda, cod_produs)
VALUES (10, 6);
```

```
INSERT INTO Produs_se_afila_in_Comanda (cod_comanda, cod_produs)
VALUES (10, 4);
```

```
INSERT INTO Produs_se_afila_in_Comanda (cod_comanda, cod_produs)
VALUES (10, 20);
```

```
29 -- INSERT INTO Produs_se_afila_in_Comanda (cod_comanda, cod_produs) VALUES (7, 24);
30 -- --8
31 -- INSERT INTO Produs_se_afila_in_Comanda (cod_comanda, cod_produs) VALUES (8, 10);
32 -- INSERT INTO Produs_se_afila_in_Comanda (cod_comanda, cod_produs) VALUES (8, 17);
33 -- INSERT INTO Produs_se_afila_in_Comanda (cod_comanda, cod_produs) VALUES (8, 28);
34 -- --9
35 -- INSERT INTO Produs_se_afila_in_Comanda (cod_comanda, cod_produs) VALUES (9, 9);
36 -- INSERT INTO Produs_se_afila_in_Comanda (cod_comanda, cod_produs) VALUES (9, 15);
37 -- INSERT INTO Produs_se_afila_in_Comanda (cod_comanda, cod_produs) VALUES (9, 23);
38 -- --10
39 -- INSERT INTO Produs_se_afila_in_Comanda (cod_comanda, cod_produs) VALUES (10, 6);
40 -- INSERT INTO Produs_se_afila_in_Comanda (cod_comanda, cod_produs) VALUES (10, 14);
41 -- INSERT INTO Produs_se_afila_in_Comanda (cod_comanda, cod_produs) VALUES (10, 20);
42
43 select * from Produs_se_afila_in_Comanda;
```

COD_COMANDA	COD_PRODUS
1	4
1	9
1	14
2	21
2	27

Tabela FURNIZOR_distribue_in_LOCATIA:

```
-- Fornecedor_distribue_in_Locatia
INSERT INTO Fornecedor_distribue_in_Locatia (cod_fornecedor, cod_locatie)
VALUES (1, 1);
INSERT INTO Fornecedor_distribue_in_Locatia (cod_fornecedor, cod_locatie)
VALUES (2, 2);
INSERT INTO Fornecedor_distribue_in_Locatia (cod_fornecedor, cod_locatie)
VALUES (3, 3);
INSERT INTO Fornecedor_distribue_in_Locatia (cod_fornecedor, cod_locatie)
VALUES (4, 4);
INSERT INTO Fornecedor_distribue_in_Locatia (cod_fornecedor, cod_locatie)
VALUES (5, 5);
INSERT INTO Fornecedor_distribue_in_Locatia (cod_fornecedor, cod_locatie)
VALUES (6, 6);
INSERT INTO Fornecedor_distribue_in_Locatia (cod_fornecedor, cod_locatie)
VALUES (7, 7);
INSERT INTO Fornecedor_distribue_in_Locatia (cod_fornecedor, cod_locatie)
VALUES (8, 8);
INSERT INTO Fornecedor_distribue_in_Locatia (cod_fornecedor, cod_locatie)
VALUES (9, 9);
INSERT INTO Fornecedor_distribue_in_Locatia (cod_fornecedor, cod_locatie)
VALUES (10, 10);
```

```
1 -- -- Fornecedor_distribue_in_Locatia
2 -- INSERT INTO Fornecedor_distribue_in_Locatia (cod_fornecedor, cod_locatie) VALUES (1, 1);
3 -- INSERT INTO Fornecedor_distribue_in_Locatia (cod_fornecedor, cod_locatie) VALUES (2, 2);
4 -- INSERT INTO Fornecedor_distribue_in_Locatia (cod_fornecedor, cod_locatie) VALUES (3, 3);
5 -- INSERT INTO Fornecedor_distribue_in_Locatia (cod_fornecedor, cod_locatie) VALUES (4, 4);
6 -- INSERT INTO Fornecedor_distribue_in_Locatia (cod_fornecedor, cod_locatie) VALUES (5, 5);
7 -- INSERT INTO Fornecedor_distribue_in_Locatia (cod_fornecedor, cod_locatie) VALUES (6, 6);
8 -- INSERT INTO Fornecedor_distribue_in_Locatia (cod_fornecedor, cod_locatie) VALUES (7, 7);
9 -- INSERT INTO Fornecedor_distribue_in_Locatia (cod_fornecedor, cod_locatie) VALUES (8, 8);
10 -- INSERT INTO Fornecedor_distribue_in_Locatia (cod_fornecedor, cod_locatie) VALUES (9, 9);
11 -- INSERT INTO Fornecedor_distribue_in_Locatia (cod_fornecedor, cod_locatie) VALUES (10, 10);
12
13 select * from Fornecedor_distribue_in_Locatia;
```

COD_FURNIZOR	COD_LOCATIE
1	1
2	2
3	3
4	4
5	5
6	6

6. 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 utilizator comenzile sale, respectiv pentru fiecare comanda, numele, codul si pretul produsului din comanda respectiva.

```
CREATE OR REPLACE PROCEDURE display_user_orders
```

```
IS
```

```
    TYPE produs_record IS RECORD (
```

```
        cod_produs produs.cod_produs%TYPE,
```

```
        pret produs.pret%TYPE,
```

```
        nume produs.nume%TYPE
```

```
    );
```

```
    TYPE vector_produse IS VARRAY(200) OF produs_record;
```

```
    TYPE comanda_cu_produse_record IS RECORD (
```

```
        cod_comanda comanda.cod_comanda%TYPE,
```

```
        produse vector_produse
```

```
    );
```

```
    TYPE comanda_cu_produse_nested_table IS TABLE OF  
comanda_cu_produse_record;
```

```
TYPE utilizator_comanda_indexed_table IS TABLE OF  
comanda_cu_produce_nested_table INDEX BY PLS_INTEGER;
```

```
    utilizatori_cu_comenzi utilizator_comanda_indexed_table;
```

```
BEGIN
```

```
    FOR utilizator IN (SELECT DISTINCT cod_utilizator FROM comanda)
```

```
    LOOP
```

```
        utilizatori_cu_comenzi(utilizator.cod_utilizator) :=  
comanda_cu_produce_nested_table();
```

```
        DBMS_OUTPUT.PUT_LINE('Utilizator: ' || utilizator.cod_utilizator);
```

```
        FOR comanda IN (SELECT cod_comanda FROM comanda WHERE  
cod_utilizator = utilizator.cod_utilizator)
```

```
        LOOP
```

```
            DECLARE
```

```
                produse vector_produce := vector_produce();
```

```
            BEGIN
```

```
                FOR produs IN (SELECT p.cod_produs, p.pret, p.numa
```

```
                                FROM produs p,
```

```
produs_se_afla_in_comanda pc
```

```
                                WHERE p.cod_produs = pc.cod_produs
```

```
                                AND pc.cod_comanda =  
comanda.cod_comanda)
```

```
                LOOP
```

```
                    produse.extend();
```

```
                    produse(produse.last) := produs_record(
```

```

        cod_produs => produs.cod_produs,
        pret => produs.pret,
        nume => produs.nume
    );
END LOOP;

utilizatori_cu_comenzi(utilizator.cod_utilizator).EXTEND;

utilizatori_cu_comenzi(utilizator.cod_utilizator)(utilizatori_cu_comenzi(
utilizator.cod_utilizator).LAST) :=

    comanda_cu_produce_record(
        cod_comanda => comanda.cod_comanda,
        produce => produce
    );

    DBMS_OUTPUT.PUT_LINE('    Comanda: ' ||
comanda.cod_comanda);

    FOR i IN 1..produce.count
    LOOP
        DBMS_OUTPUT.PUT_LINE('        Produs: ' ||
produce(i).cod_produs ||
                                ', Nume: ' || produce(i).nume ||
                                ', Pret: ' || produce(i).pret);

    END LOOP;

END;

END LOOP;
```

```
DBMS_OUTPUT.PUT_LINE('-----');
```

```
DBMS_OUTPUT.NEW_LINE;
```

```
END LOOP;
```

```
END display_user_orders;
```

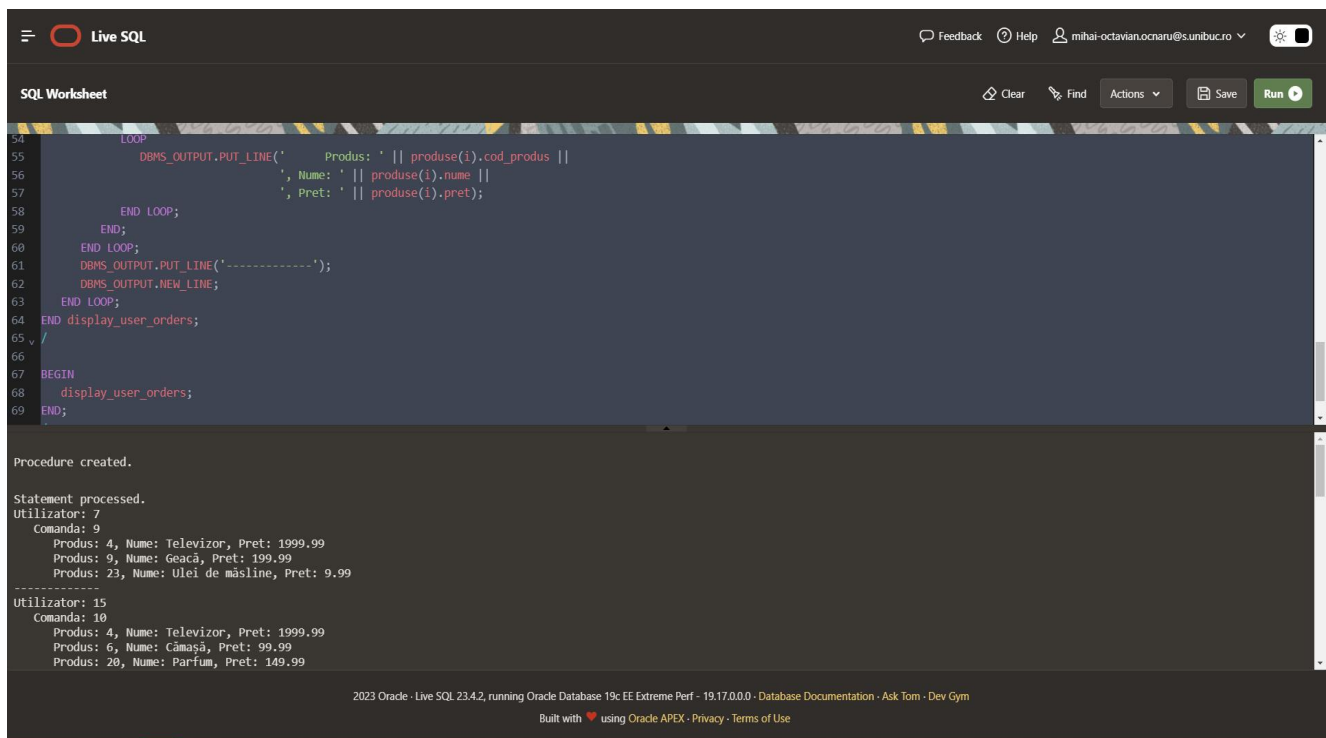
```
/
```

```
BEGIN
```

```
display_user_orders;
```

```
END;
```

```
/
```



The screenshot shows the Live SQL interface with a dark theme. The top bar includes a hamburger menu, the 'Live SQL' logo, and user information. The main area is titled 'SQL Worksheet' and contains a code editor with the following SQL code:

```
54 LOOP
55     DBMS_OUTPUT.PUT_LINE('      Produs: ' || produse(i).cod_produs ||
56                           ', Nume: ' || produse(i).nume ||
57                           ', Pret: ' || produse(i).pret);
58 END LOOP;
59 END;
60 END LOOP;
61 DBMS_OUTPUT.PUT_LINE('-----');
62 DBMS_OUTPUT.NEW_LINE;
63 END LOOP;
64 END display_user_orders;
65 /
66
67 BEGIN
68     display_user_orders;
69 END;
```

Below the code editor, the output area shows the results of the execution:

```
Procedure created.

Statement processed.
Utilizator: 7
Comanda: 9
      Produs: 4, Nume: Televizor, Pret: 1999.99
      Produs: 9, Nume: Geacă, Pret: 199.99
      Produs: 23, Nume: Ulei de măsline, Pret: 9.99
-----
Utilizator: 15
Comanda: 10
      Produs: 4, Nume: Televizor, Pret: 1999.99
      Produs: 6, Nume: Cămașă, Pret: 99.99
      Produs: 20, Nume: Parfum, Pret: 149.99
```

The footer of the interface displays the version information: '2023 Oracle - Live SQL 23.4.2, running Oracle Database 19c EE Extreme Perf - 19.17.0.0.0 - Database Documentation - Ask Tom - Dev Gym' and mentions it is built with Oracle APEX.

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, dependent de celălalt cursor. Apelați subprogramul.

Sa afiseze numele, pretul, reviuirile si recenzii produselor care se afla intr-o comanda plasata de un utilizator dintr-o tara latina.

CREATE OR REPLACE PROCEDURE raport_produce_recenzii IS

CURSOR curson_produce IS

SELECT distinct p.cod_produș, p.nume, p.pret

FROM produș p, produș_se_afla_in_comanda pc, comanda c

where pc.cod_produș = p.cod_produș

and pc.cod_comanda = c.cod_comanda

and c.cod_utilizator in (select distinct u.cod_utilizator

from locatie l, utilizator u, tara t

where l.cod_locatie = u.cod_locatie

and l.cod_tara = t.cod_tara

and initcap(t.nume) in ('România',

'Franța', 'Spania', 'Italia'));

CURSOR curson_recenzii (v_cod_produș INT) IS

SELECT descriere, rating

FROM Review

WHERE cod_produș = v_cod_produș;

v_cod_produș Produș.cod_produș%TYPE;


```

v_ume_produs Produs.ume%TYPE;
v_pret Produs.pret%TYPE;
v_descriere Review.descriere%TYPE;
v_rating Review.rating%TYPE;

BEGIN

    OPEN curson_produce;

    LOOP

        FETCH curson_produce INTO v_cod_produs, v_ume_produs,
v_pret;

        EXIT WHEN curson_produce%NOTFOUND;

        OPEN curson_recenzii(v_cod_produs);

        LOOP

            FETCH curson_recenzii INTO v_descriere, v_rating;

            EXIT WHEN curson_recenzii%NOTFOUND;

            DBMS_OUTPUT.PUT_LINE('Produs: ' || v_ume_produs || ',
Pret: ' || v_pret);

            DBMS_OUTPUT.PUT_LINE('  Recenzie: ' || v_descriere || ',
Rating: ' || v_rating);

            DBMS_OUTPUT.PUT_LINE('-----');

            DBMS_OUTPUT.NEW_LINE;

        END LOOP;

        CLOSE curson_recenzii;

    END LOOP;

```

CLOSE curson_produce;

END raport_produce_recenzii;

/

BEGIN

raport_produce_recenzii;

END;

/

The screenshot shows the Oracle Live SQL interface. The top bar includes the 'Live SQL' logo, a 'Feedback' link, a 'Help' link, a user profile 'mihai-octavian.ocnaru@s.unibuc.ro', and a settings icon. Below the top bar is the 'SQL Worksheet' section with a 'Clear' button, a 'Find' search bar, and 'Actions', 'Save', and 'Run' buttons. The main area contains a PL/SQL script with line numbers 1 through 17. The script defines a procedure 'raport_produce_recenzii' that uses two cursors: 'curson_produce' to select product details and 'curson_recenzii' to select reviews for a given product code. The output shows the procedure was created successfully, followed by the execution of the procedure for product codes 1, 2, and 3. The output for each product shows the product name, price, and a review with a rating. The footer of the interface displays '2023 Oracle - Live SQL 23.4.2, running Oracle Database 19c EE Extreme Perf - 19.17.0.0.0 - Database Documentation - Ask Tom - Dev Gym' and a note about being built with Oracle APEX.

```
1
2 CREATE OR REPLACE PROCEDURE raport_produce_recenzii IS
3
4     CURSOR curson_produce IS
5         SELECT distinct p.cod_produc, p.nume, p.pret
6         FROM produs p, produs_se afla_in comanda pc, comanda c
7         where pc.cod_produc = p.cod_produc
8         and pc.cod_comanda = c.cod_comanda
9         and c.cod_utilizator in (select distinct u.cod_utilizator
10                                from locatie l, utilizator u, tara t
11                                where l.cod_locatie = u.cod_locatie
12                                and l.cod_tara = t.cod_tara
13                                and initcap(t.nume) in ('România', 'Franța', 'Spania', 'Italia'));
14
15 CURSOR curson_recenzii (v_cod_produc INT) IS
16     SELECT descriere, rating
17     FROM Review
```

Procedure created.

Statement processed.

Produs: Televizor, Pret: 1999.99
Recenzie: Produs de calitate superioara!, Rating: 5

Produs: Dulap, Pret: 1499.99
Recenzie: Excelent produs!, Rating: 5

Produs: Pat, Pret: 699.99
Recenzie: Bun produs!, Rating: 4

Produs: Scaun, Pret: 49.99

2023 Oracle - Live SQL 23.4.2, running Oracle Database 19c EE Extreme Perf - 19.17.0.0.0 - Database Documentation - Ask Tom - Dev Gym
Built with using Oracle APEX - Privacy - Terms of Use

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

Sa se afiseze un raport care sa afiseze numele si pretul produselor livrate de un furnizor dat, intr-o anumita tara si sub o anumita limita de pret. In cazul in care tara nu exista sau pretul este prea mic se vor afisa erori adecvate.

```
CREATE OR REPLACE FUNCTION raport_produce_tara_categorie(
```

```
    v_nume_tara IN VARCHAR2,
```

```
    v_nume_categorie IN VARCHAR2,
```

```
    v_max_pret IN NUMBER
```

```
) RETURN VARCHAR2
```

```
IS
```

```
    v_raport VARCHAR2(4000);
```

```
    invalid_country_name EXCEPTION;
```

```
    PRAGMA EXCEPTION_INIT(invalid_country_name, -1333);
```

```
    too_small_price EXCEPTION;
```

```
    PRAGMA EXCEPTION_INIT(too_small_price, -1332);
```

```
BEGIN
```

```
    v_raport := 'Products in ' || v_nume_tara || ', Category: ' ||  
v_nume_categorie || ' with price <= ' || v_max_pret || ':';
```

DECLARE

min_price produs.pret%TYPE;

BEGIN

select min(p.pret)

into min_price

FROM tara t, locatie l, furnizor f, furnizor_distributie_in_locatia fl,
produs p

WHERE t.cod_tara = l.cod_tara

AND f.cod_furnizor = fl.cod_furnizor

AND fl.cod_locatie = l.cod_locatie

AND f.cod_furnizor = p.cod_furnizor;

if min_price > v_max_pret then

RAISE too_small_price;

end if;

END;

DECLARE

flag Boolean := false;

BEGIN

FOR country IN (

SELECT 1

FROM dual

WHERE EXISTS (

SELECT 1

```
FROM tara t, locatie l, furnizor f,  
furnizor_distributie_in_locatia fl  
  
WHERE t.cod_tara = l.cod_tara  
  
AND f.cod_furnizor = fl.cod_furnizor  
  
AND fl.cod_locatie = l.cod_locatie  
  
AND t.nume = v_nume_tara
```

```
)
```

```
) LOOP
```

```
flag := true;
```

```
EXIT;
```

```
END LOOP;
```

```
IF NOT flag THEN
```

```
RAISE invalid_country_name;
```

```
END IF;
```

```
END;
```

```
FOR prod_rec IN (
```

```
SELECT distinct p.nume, p.pret
```

```
FROM produs p, produs_se_afla_in_comanda pc, comanda c,  
locatie l, tara t, categorie cat, furnizor f, furnizor_distributie_in_locatia fl
```

```
where p.cod_produs = pc.cod_produs
```

```
and pc.cod_comanda = c.cod_comanda
```

```
and l.cod_tara = t.cod_tara
```

```
and p.cod_categorie = cat.cod_categorie
```

```
and f.cod_furnizor = fl.cod_furnizor
```

and fl.cod_locatie = l.cod_locatie

and f.cod_furnizor = p.cod_furnizor

and initcap(t.num) = initcap(v_num_tara)

AND initcap(cat.num)= initcap(v_num_categorie)

AND p.pret <= v_max_pret

) LOOP

v_raport := v_raport || CHR(10) || 'Product: ' || prod_rec.num || '
Price: ' || prod_rec.pret;

END LOOP;

IF v_raport = 'Products in ' || v_num_tara || ', Category: ' ||
v_num_categorie || ' with price <= ' || v_max_pret || ':' THEN

RAISE NO_DATA_FOUND;

END IF;

RETURN v_raport;

EXCEPTION

WHEN invalid_country_name THEN

DBMS_OUTPUT.PUT_LINE('Error: The country name given is not
in the database.');

RAISE_APPLICATION_ERROR(-20001, 'Invalid country name');

RETURN NULL;

WHEN too_small_price THEN

DBMS_OUTPUT.PUT_LINE('Error: The max price given is too small compared to the database prices.');

RAISE_APPLICATION_ERROR(-20001, 'Price too small');

RETURN NULL;

WHEN NO_DATA_FOUND THEN

DBMS_OUTPUT.PUT_LINE('Error: No products found for the given criteria.');

RAISE_APPLICATION_ERROR(-20002, 'No products found for the given criteria');

RETURN NULL;

WHEN OTHERS THEN

DBMS_OUTPUT.PUT_LINE('An error occurred: ' || SQLERRM);

RETURN NULL;

END raport_produse_tara_categorie;

/

Declaram codul pentru un apel reusit al functiei:

DECLARE

 v_result VARCHAR2(4000);

BEGIN

 v_result := raport_produce_tara_categorie('România', 'Electronice',
3000);

 IF v_result IS NOT NULL THEN

 DBMS_OUTPUT.PUT_LINE(v_result);

 END IF;

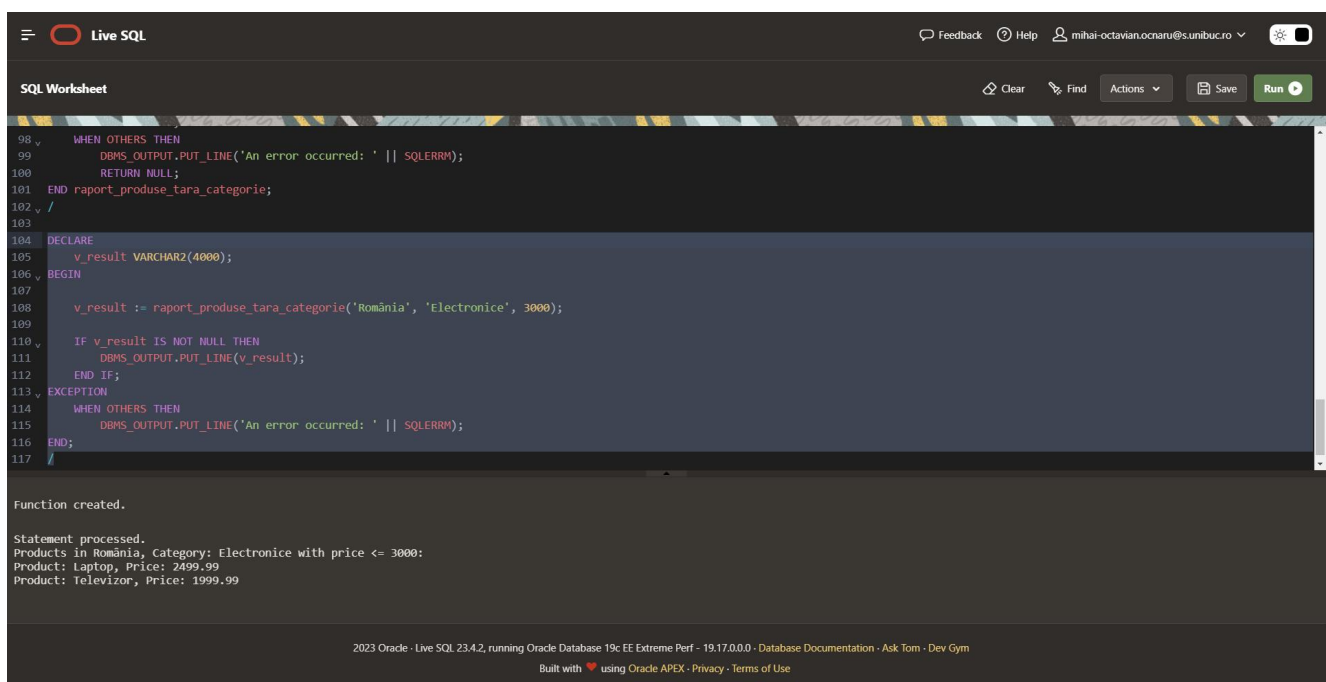
EXCEPTION

 WHEN OTHERS THEN

 DBMS_OUTPUT.PUT_LINE('An error occurred: ' || SQLERRM);

END;

/



The screenshot shows the 'Live SQL' web interface. At the top, there's a navigation bar with 'Live SQL', 'Feedback', 'Help', and a user profile. Below that is the 'SQL Worksheet' header with 'Clear', 'Find', 'Actions', 'Save', and 'Run' buttons. The main area contains the PL/SQL code from the previous blocks, with line numbers 98 to 117. The code is syntax-highlighted. Below the code editor, the output is displayed: 'Function created.', 'Statement processed.', and the results of the query: 'Products in România, Category: Electronics with price <= 3000: Product: Laptop, Price: 2499.99 Product: Televizor, Price: 1999.99'. At the bottom, there's a footer with version information and links to documentation and terms of use.

```
98 WHEN OTHERS THEN
99     DBMS_OUTPUT.PUT_LINE('An error occurred: ' || SQLERRM);
100     RETURN NULL;
101 END raport_produce_tara_categorie;
102 /
103
104 DECLARE
105     v_result VARCHAR2(4000);
106 BEGIN
107
108     v_result := raport_produce_tara_categorie('România', 'Electronice', 3000);
109
110     IF v_result IS NOT NULL THEN
111         DBMS_OUTPUT.PUT_LINE(v_result);
112     END IF;
113 EXCEPTION
114     WHEN OTHERS THEN
115         DBMS_OUTPUT.PUT_LINE('An error occurred: ' || SQLERRM);
116 END;
117 /
```

Function created.

Statement processed.

Products in România, Category: Electronics with price <= 3000:
Product: Laptop, Price: 2499.99
Product: Televizor, Price: 1999.99

2023 Oracle - Live SQL 23.4.2, running Oracle Database 19c EE Extreme Perf - 19.17.0.0.0 - Database Documentation - Ask Tom - Dev Gym
Built with using Oracle APEX - Privacy - Terms of Use

Declaram codul pentru o tara inexistentă în baza de date:

DECLARE

 v_result VARCHAR2(4000);

BEGIN

 v_result := raport_produce_tara_categorie('China', 'Electronice',
3000);

 IF v_result IS NOT NULL THEN

 DBMS_OUTPUT.PUT_LINE(v_result);

 END IF;

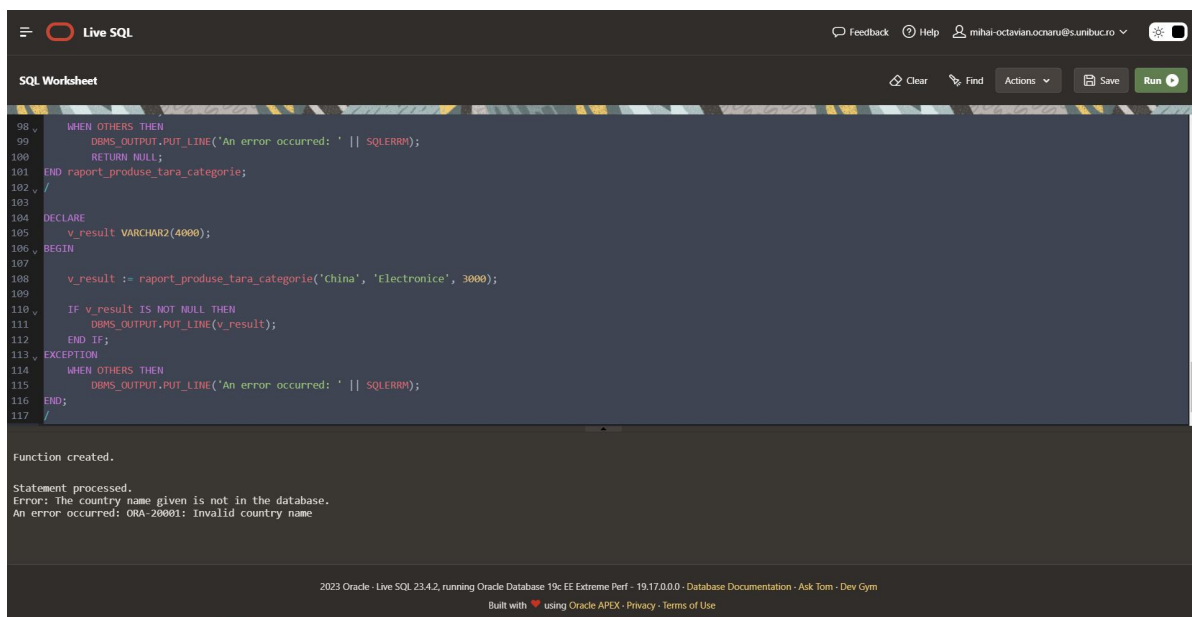
EXCEPTION

 WHEN OTHERS THEN

 DBMS_OUTPUT.PUT_LINE('An error occurred: ' || SQLERRM);

END;

/



Live SQL

SQL Worksheet

```
98 WHEN OTHERS THEN
99     DBMS_OUTPUT.PUT_LINE('An error occurred: ' || SQLERRM);
100     RETURN NULL;
101 END raport_produce_tara_categorie;
102 /
103
104 DECLARE
105     v_result VARCHAR2(4000);
106 BEGIN
107
108     v_result := raport_produce_tara_categorie('china', 'Electronice', 3000);
109
110     IF v_result IS NOT NULL THEN
111         DBMS_OUTPUT.PUT_LINE(v_result);
112     END IF;
113 EXCEPTION
114     WHEN OTHERS THEN
115         DBMS_OUTPUT.PUT_LINE('An error occurred: ' || SQLERRM);
116 END;
117 /
```

Function created.

Statement processed.
Error: The country name given is not in the database.
An error occurred: ORA-20001: Invalid country name

2023 Oracle - Live SQL 23.4.2, running Oracle Database 19c EE Extreme Perf - 19.17.0.0.0 - Database Documentation - Ask Tom - Dev Gym
Built with ♥ using Oracle APEX - Privacy - Terms of Use

Declaram codul pentru un pret maxim prea mic:

DECLARE

 v_result VARCHAR2(4000);

BEGIN

 v_result := raport_produce_tara_categorie('România', 'Electronice',
0.5);

IF v_result IS NOT NULL THEN

 DBMS_OUTPUT.PUT_LINE(v_result);

END IF;

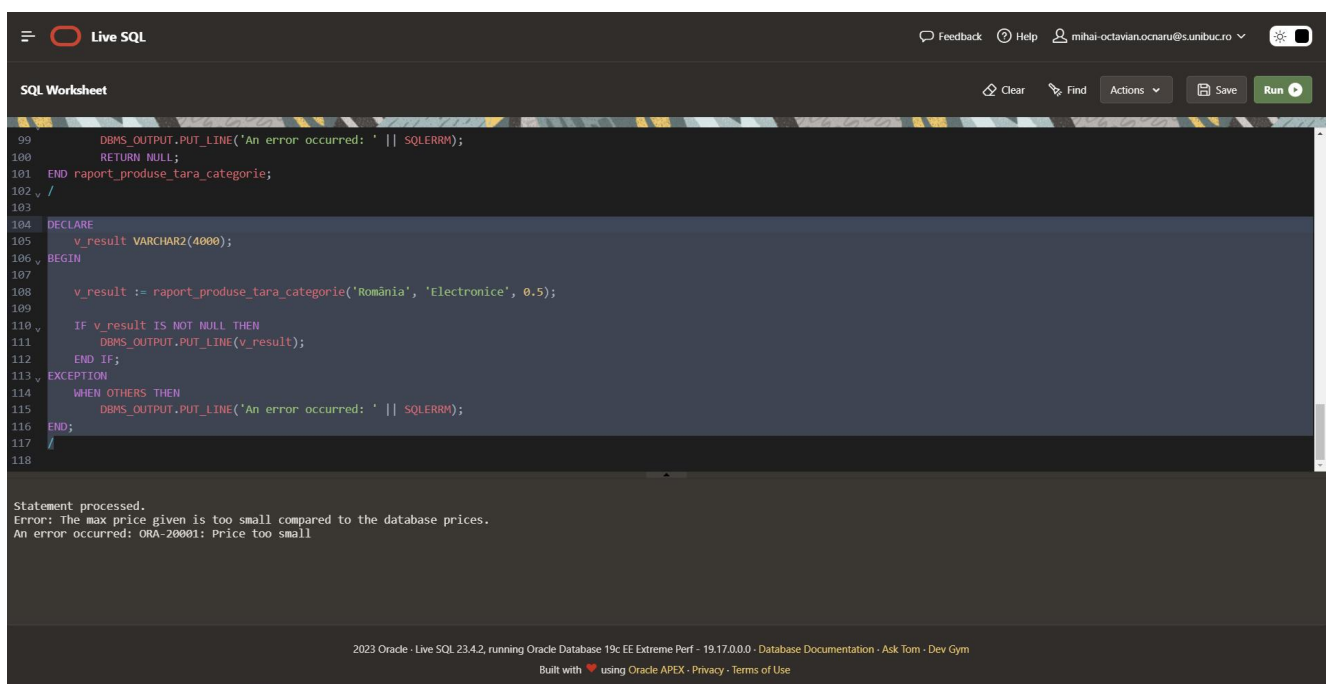
EXCEPTION

WHEN OTHERS THEN

 DBMS_OUTPUT.PUT_LINE('An error occurred: ' || SQLERRM);

END;

/



The screenshot shows the Live SQL interface with the following content:

SQL Worksheet

```
99      DBMS_OUTPUT.PUT_LINE('An error occurred: ' || SQLERRM);
100     RETURN NULL;
101 END raport_produce_tara_categorie;
102 /
103
104 DECLARE
105     v_result VARCHAR2(4000);
106 BEGIN
107
108     v_result := raport_produce_tara_categorie('România', 'Electronice', 0.5);
109
110     IF v_result IS NOT NULL THEN
111         DBMS_OUTPUT.PUT_LINE(v_result);
112     END IF;
113 EXCEPTION
114     WHEN OTHERS THEN
115         DBMS_OUTPUT.PUT_LINE('An error occurred: ' || SQLERRM);
116 END;
117 /
118
```

Statement processed.
Error: The max price given is too small compared to the database prices.
An error occurred: ORA-20001: Price too small

2023 Oracle - Live SQL 23.4.2, running Oracle Database 19c EE Extreme Perf - 19.17.0.0.0 - Database Documentation - Ask Tom - Dev Gym
Built with ❤️ using Oracle APEX - Privacy - Terms of Use

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 numele tarii si numarul angajatilor in care traiesc cei mai multi lucratori care presteaza servicii in domeniul categoriei cu pretul mediu al produselor cel mai ridicat. Sa se afiseze de asemenea si numele acestei categorii.

```
CREATE OR REPLACE PROCEDURE
afisare_tara_max_angajati_categorie_max_pret AS

    v_nume_tara Tara.nume%TYPE;

    v_numar_angajati Number;

    v_nume_categorie categorie.nume%TYPE;

BEGIN

    WITH tari as (

        SELECT DISTINCT t.nume, COUNT(COD_ANGAJAT) AS nr_angajati,
        c.nume as nume_categorie

            FROM tara t

            JOIN locatie l ON t.cod_tara = l.cod_tara

            JOIN angajat a ON l.cod_locatie = a.cod_locatie

            JOIN categorie c ON a.cod_categorie = c.cod_categorie

        WHERE a.cod_categorie IN (

            SELECT COD_CATEGORIE

            FROM produs

            GROUP BY COD_CATEGORIE
```

```

        HAVING AVG(PRET) = (
            SELECT MAX(AVG(PRET))
            FROM PRODUS
            GROUP BY COD_CATEGORIE
        )
    )
    GROUP BY t.cod_tara, t.numa, c.numa
)

SELECT tari.numa, tari.nr_angajati, tari.numa_categorie
INTO v_numa_tara, v_numar_angajati, v_numa_categorie
FROM tari
GROUP BY tari.numa, tari.nr_angajati, tari.numa_categorie
HAVING tari.nr_angajati = (SELECT MAX(nr_angajati)
                           FROM tari
                           );

```

```

DBMS_OUTPUT.PUT_LINE('Tara cu cei mai multi angajati care
lucreaza la categoria cu numele: ' || v_numa_categorie || ' si cu pretul
mediu cel mai mare: ' || v_numa_tara || ' - numar angajati: ' ||
v_numar_angajati);

```

```

EXCEPTION

```

```

    WHEN NO_DATA_FOUND THEN

```

```

        DBMS_OUTPUT.PUT_LINE('Nu există date pentru cerință.');
```

```

    WHEN TOO_MANY_ROWS THEN

```

```

        DBMS_OUTPUT.PUT_LINE('Eroare: Prea multe rânduri
returnate.');
```

WHEN OTHERS THEN

```
DBMS_OUTPUT.PUT_LINE('A apărut o eroare: ' || SQLERRM);
```

```
END afisare_tara_max_angajati_categorie_max_pret;
```

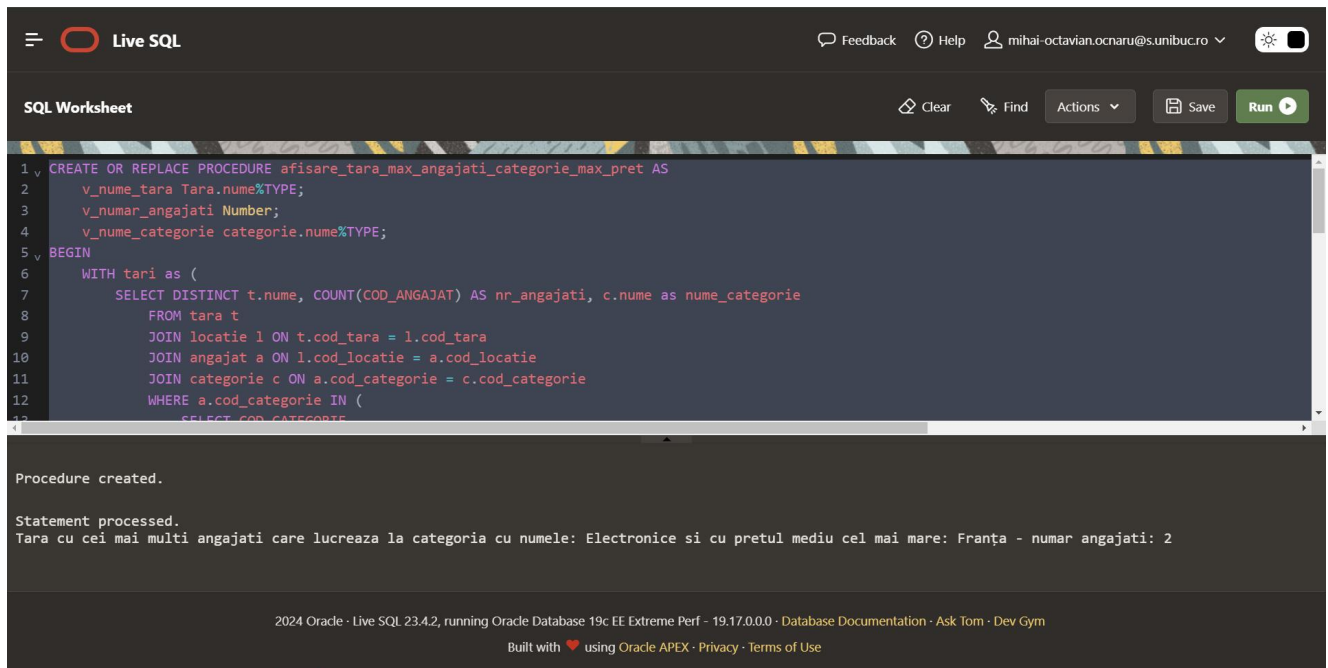
```
/
```

BEGIN

```
afisare_tara_max_angajati_categorie_max_pret;
```

```
END;
```

```
/
```



The screenshot shows the 'Live SQL' web interface. The top navigation bar includes a hamburger menu, the 'Live SQL' logo, and user information for 'mihai-octavian.ocnaru@s.unibuc.ro'. Below the navigation bar is a toolbar with 'Clear', 'Find', 'Actions', 'Save', and a 'Run' button. The main area is titled 'SQL Worksheet' and contains the following SQL code:

```
1 CREATE OR REPLACE PROCEDURE afisare_tara_max_angajati_categorie_max_pret AS
2   v_nume_tara Tara.nume%TYPE;
3   v_numar_angajati Number;
4   v_nume_categorie categorie.nume%TYPE;
5 BEGIN
6   WITH tari as (
7     SELECT DISTINCT t.nume, COUNT(COD_ANGAJAT) AS nr_angajati, c.nume as nume_categorie
8     FROM tara t
9     JOIN locatie l ON t.cod_tara = l.cod_tara
10    JOIN angajat a ON l.cod_locatie = a.cod_locatie
11    JOIN categorie c ON a.cod_categorie = c.cod_categorie
12    WHERE a.cod_categorie IN (
13      SELECT COD_CATEGORIE
```

Below the code editor, the execution results are displayed:


```
Procedure created.

Statement processed.
Tara cu cei mai multi angajati care lucreaza la categoria cu numele: Electronice si cu pretul mediu cel mai mare: Franța - numar angajati: 2
```

The footer of the interface contains the following text:

2024 Oracle - Live SQL 23.4.2, running Oracle Database 19c EE Extreme Perf - 19.17.0.0.0 - [Database Documentation](#) - [Ask Tom](#) - [Dev Gym](#)
Built with ❤️ using Oracle APEX - [Privacy](#) - [Terms of Use](#)

Pentru exemplificarea cazului NO_DATA_FOUND nu am mai rulat scriptul de inserare in baza de date:

 Live SQL

16 database objects have been dropped. Session history has been removed. NLS settings reset.

SQL Worksheet


```
33
34 EXCEPTION
35     WHEN NO_DATA_FOUND THEN
36         DBMS_OUTPUT.PUT_LINE('Nu există date pentru cerință.');
```

```
37     WHEN TOO_MANY_ROWS THEN
38         DBMS_OUTPUT.PUT_LINE('Eroare: Prea multe rânduri returnate.');
```

```
39     WHEN OTHERS THEN
40         DBMS_OUTPUT.PUT_LINE('A apărut o eroare: ' || SQLERRM);
41 END afisare_tara_max_angajati_categorie_max_pret;
42 /
43
44 BEGIN
45     afisare_tara_max_angajati_categorie_max_pret;
46 END;
47 /
```

Procedure created.

Statement processed.
Nu există date pentru cerință.

2023 Oracle - Live SQL 23.4.2, running Oracle Database 19c EE Extreme Perf - 19.17.0.0.0 - Database Documentation - Ask Tom - Dev Gym
Built with  using Oracle APEX - Privacy - Terms of Use

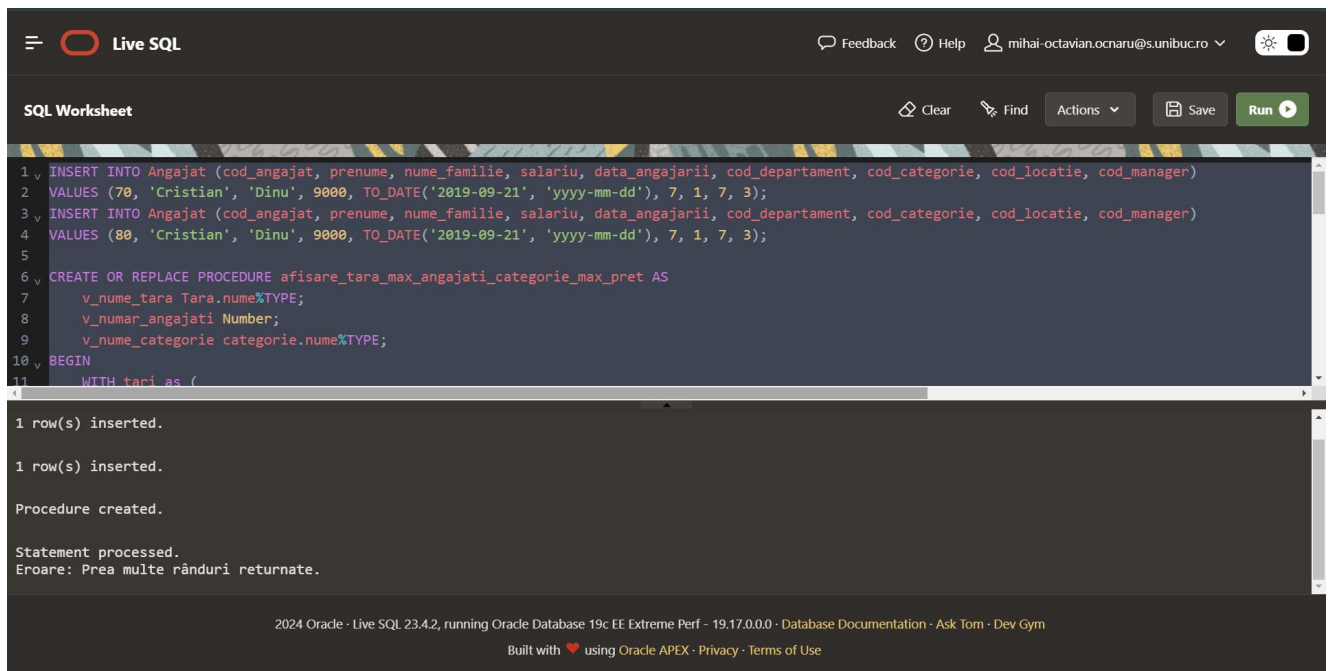
Pentru exemplificarea cazului TOO_MANY_ROWS am inserat doi angajati ce lucreaza in categoria 1 si sunt situati in Spania:

```
INSERT INTO Angajat (cod_angajat, prenume, nume_familie, salariu, data_angajarii, cod_departament, cod_categorie, cod_locatie, cod_manager)
```

```
VALUES (70, 'Cristian', 'Dinu', 9000, TO_DATE('2019-09-21', 'yyyy-mm-dd'), 7, 1, 7, 3);
```

```
INSERT INTO Angajat (cod_angajat, prenume, nume_familie, salariu, data_angajarii, cod_departament, cod_categorie, cod_locatie, cod_manager)
```

```
VALUES (80, 'Cristian', 'Dinu', 9000, TO_DATE('2019-09-21', 'yyyy-mm-dd'), 7, 1, 7, 3);
```



The screenshot shows the Live SQL interface with the following content:

SQL Worksheet

```
1 INSERT INTO Angajat (cod_angajat, prenume, nume_familie, salariu, data_angajarii, cod_departament, cod_categorie, cod_locatie, cod_manager)
2 VALUES (70, 'Cristian', 'Dinu', 9000, TO_DATE('2019-09-21', 'yyyy-mm-dd'), 7, 1, 7, 3);
3 INSERT INTO Angajat (cod_angajat, prenume, nume_familie, salariu, data_angajarii, cod_departament, cod_categorie, cod_locatie, cod_manager)
4 VALUES (80, 'Cristian', 'Dinu', 9000, TO_DATE('2019-09-21', 'yyyy-mm-dd'), 7, 1, 7, 3);
5
6 CREATE OR REPLACE PROCEDURE ofisare_tara_max_angajati_categorie_max_pret AS
7   v_nume_tara Tara.numetype;
8   v_numar_angajati Number;
9   v_nume_categorie categorie.numetype;
10 BEGIN
11   WITH tari AS (
```

Execution Results:

```
1 row(s) inserted.
1 row(s) inserted.
Procedure created.
Statement processed.
Eroare: Prea multe rânduri returnate.
```

2024 Oracle · Live SQL 23.4.2, running Oracle Database 19c EE Extreme Perf - 19.17.0.0.0 · Database Documentation · Ask Tom · Dev Gym
Built with ❤️ using Oracle APEX · Privacy · Terms of Use

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

Sa se creeze un declansator astfel incat adaugarea unui produs intr-o comanda sa fie interzisa daca valoarea cumulata a reducerilor oferite prin discount comenzilor depaseste suma de 10.000.

```
CREATE OR REPLACE TRIGGER discount_comenzi
    BEFORE INSERT OR UPDATE ON PRODUS_SE_AFLA_IN_COMANDA
DECLARE
    red_total produs.pret%TYPE;
BEGIN
    WITH tabela_comenzi AS (
        SELECT c.cod_comanda, SUM(p.pret) AS pret_initial, SUM(p.pret
* (1 - c.discount)) AS pret_redus, c.discount * SUM(p.pret) AS reducere
        FROM comanda c, produs p, produs_se_afla_in_comanda pc
        WHERE c.cod_comanda = pc.cod_comanda
        AND pc.cod_produs = p.cod_produs
        GROUP BY c.cod_comanda, c.discount
    )
    SELECT SUM(t.reducere)
    INTO red_total
    FROM tabela_comenzi t;

    IF red_total > 10000 THEN
        RAISE_APPLICATION_ERROR(-20101, 'Nu puteti sa mai
introduceti o noua comanda cu discount nenul. Suma reducerilor
depasita.');
```


END IF;

END;

/

WITH tabela_comenzi AS (

SELECT c.cod_comanda, SUM(p.pret) AS pret_initial, SUM(p.pret
* (1 - c.discount)) AS pret_redus, c.discount * SUM(p.pret) AS reducere

FROM comanda c, produs p, produs_se_afla_in_comanda pc

WHERE c.cod_comanda = pc.cod_comanda

AND pc.cod_produs = p.cod_produs

GROUP BY c.cod_comanda, c.discount

)

SELECT SUM(t.reducere) suma_reduceri

FROM tabela_comenzi t;

Pentru evidentiere am creat o noua comanda si am inserat produse intr-o comanda creata recent, astfel suma totala fiind initial 9897.2102, dupa prima inserare ar depasi suma target de 10.000 ceea ce rezulta intr-o eroare.

BEGIN

INSERT INTO comanda VALUES (13, 0.5, 1);

INSERT INTO produs_se_afla_in_comanda VALUES (13, 15); --
199.99

INSERT INTO produs_se_afla_in_comanda VALUES (13, 14); --
1499.99

END;

/

Live SQL

Feedback Help mihai-octavian.ocnaru@s.unibuc.ro

SQL Worksheet Clear Find Actions Save Run

```
30      )
31      GROUP BY c.cod_comanda, c.discount
32  )
33  SELECT SUM(t.reduceri) suma_reduceri
34  FROM tabela_comenzi t;
35  BEGIN
36      INSERT INTO comanda VALUES (13, 0.5, 1);
37      INSERT INTO produs_se_afla_in_comanda VALUES (13, 15); -- 199.99
38      INSERT INTO produs_se_afla_in_comanda VALUES (13, 14); -- 1499.99
39  END;
40  /
41
```

Trigger created.

SUMA_REduceri
9897.2102

Download CSV

ORA-20101: Nu puteti sa mai introduceti o noua comanda cu discount nenul. Suma reducerilor depasita. ORA-06512: at "SQL_CZQAPEEEXPZYBINTSW5JHDPWK.DISCOUNT_COMENZI", line 16
ORA-06512: at line 4
ORA-06512: at "SYS.DBMS_SQL", line 1721

More Details: <https://docs.oracle.com/error-help/db/ora-20101>

2023 Oracle - Live SQL 23.4.2, running Oracle Database 19c EE Extreme Perf - 19.17.0.0.0 - Database Documentation - Ask Tom - Dev Gym
Built with using Oracle APEX - Privacy - Terms of Use

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

Sa se creeze un declansator care sa interzica inserarea unui angajat daca acesta are salariu mai mic decat media departamentului sau, de asemenea se interzice cresterea salariala daca acest raport este mai mare decat cel mai mic salariu maxim pe departamente, iar procentul micsorarii salariului nu poate fi mai mic decat raportul dintre cele mai mici doua salarii.

Exemplificare micsorare: 3500 si 4000 sunt cele mai mici salarii, avand raportul $3500/4000 = 0.875$. Un salariat cu salariul de 4000 nu va putea primi un salariu mai mic de 3500, astfel raportul va fi mai mic decat 0.875.

```
CREATE OR REPLACE TRIGGER salariu_angajat
    BEFORE INSERT OR UPDATE ON ANGAJAT
    FOR EACH ROW
DECLARE
    PRAGMA AUTONOMOUS_TRANSACTION;
    v_dep_sal_mediu Angajat.salariu%TYPE;
    v_sal_aux Angajat.salariu%TYPE;
    TYPE sal_array IS VARRAY(2) OF Angajat.salariu%TYPE;
    v_sal_array sal_array;
    v_proc Number;
BEGIN
    IF INSERTING THEN
        SELECT AVG(salariu)
        INTO v_dep_sal_mediu
        FROM angajat
        WHERE cod_departament = :NEW.cod_departament;
```

```

        IF v_dep_sal_mediu > :NEW.salariu THEN

            RAISE_APPLICATION_ERROR(-20001, 'Noul angajat trebuie
            să aibă cel puțin un salariu egal cu ' || v_dep_sal_mediu || '.');

        END IF;

    ELSIF UPDATING THEN

        -- crestere salariala

        IF :NEW.salariu > :OLD.salariu THEN

            WITH sal_max AS (

                SELECT cod_departament, MAX(salariu) maxim

                FROM angajat

                GROUP BY cod_departament

            )

            SELECT MIN(s.maxim)

            INTO v_sal_aux

            FROM sal_max s;

            IF :NEW.salariu > v_sal_aux THEN

                RAISE_APPLICATION_ERROR(-20002, 'Majorarea
                salarială nu poate depăși minimul salariilor maxime pe departament!');

            END IF;

        -- scadere salariala

    ELSE

        WITH salarii AS (

            SELECT DISTINCT salariu

            FROM angajat

            ORDER BY salariu

```

)

SELECT salariu

BULK COLLECT INTO v_sal_array

FROM salarii

WHERE ROWNUM <= 2;

v_proc := v_sal_array(1) / v_sal_array(2);

IF :NEW.salariu / :OLD.salariu < v_proc THEN

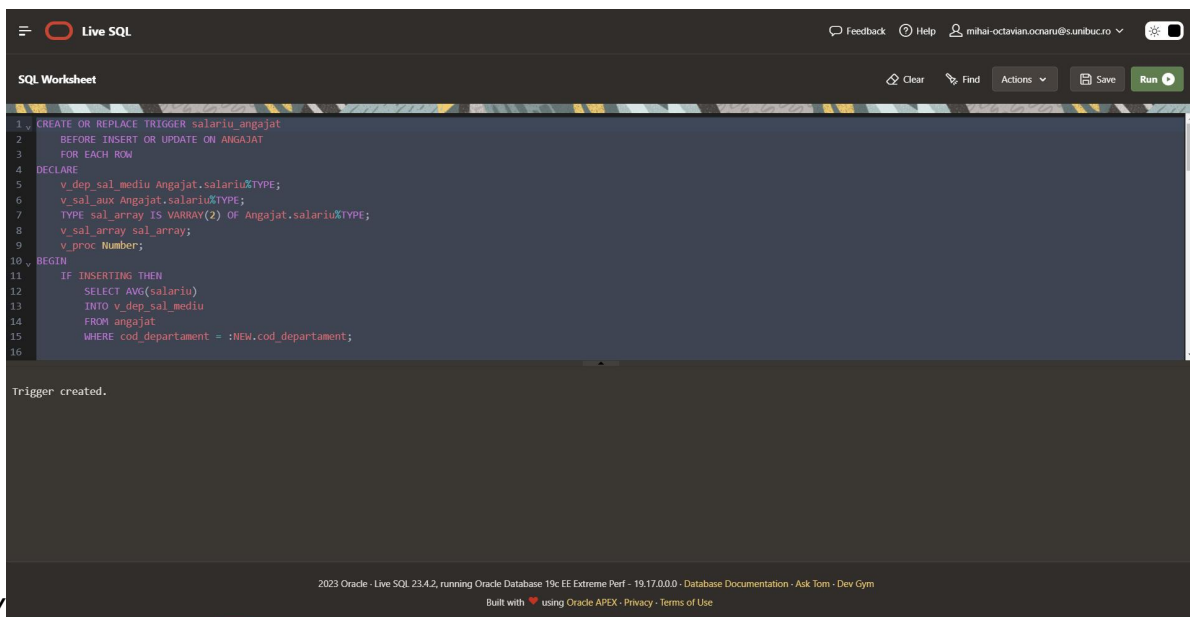
RAISE_APPLICATION_ERROR(-20003, 'Micșorarea salariului nu poate fi mai mică decât diferența procentuală a celor mai mici 2 salarii');

END IF;

END IF;

END IF;

END;/



The screenshot shows the 'Live SQL' web interface. At the top, there's a navigation bar with 'Live SQL', 'Feedback', 'Help', a user profile 'mihai-octavian.ocnaru@sunibuc.ro', and a settings icon. Below this is the 'SQL Worksheet' section, which includes a toolbar with 'Clear', 'Find', 'Actions', 'Save', and a 'Run' button. The main area contains an SQL script for creating a trigger named 'salariu_angajat'. The script is as follows:

```
1 CREATE OR REPLACE TRIGGER salariu_angajat
2 BEFORE INSERT OR UPDATE ON ANGAJAT
3 FOR EACH ROW
4 DECLARE
5     v_dep_sal_mediu Angajat.salariu%TYPE;
6     v_sal_aux Angajat.salariu%TYPE;
7     TYPE sal_array IS VARRAY(2) OF Angajat.salariu%TYPE;
8     v_sal_array sal_array;
9     v_proc Number;
10 BEGIN
11     IF INSERTING THEN
12         SELECT AVG(salariu)
13         INTO v_dep_sal_mediu
14         FROM angajat
15         WHERE cod_departament = :NEW.cod_departament;
16
```

Below the script, a message states 'Trigger created.' At the very bottom of the interface, there is a footer with the text: '2023 Oracle - Live SQL 23.4.2, running Oracle Database 19c EE Extreme Perf - 19.17.0.0.0 - Database Documentation - Ask Tom - Dev Gym' and 'Built with using Oracle APEX - Privacy - Terms of Use'.

Pentru cazul de inserare am afisat media pe departamentul 1 si am adaugat un nou angajat in departamentul respectiv cu salariul mai mic decat media.

DECLARE

 v_avg_sal Angajat.salariu%TYPE;

 v_cod_dep Angajat.cod_departament%TYPE;

BEGIN

 v_cod_dep := 1;

 select avg(salariu)

 into v_avg_sal

 from angajat

 where cod_departament = v_cod_dep

 group by cod_departament;

 v_avg_sal := v_avg_sal - 10;

 INSERT INTO Angajat (cod_angajat, prenume, nume_familie, salariu,
data_angajarii, cod_departament, cod_categorie, cod_locatie,
cod_manager)

 VALUES (100, 'John', 'Doe', v_avg_sal, TO_DATE('2022-01-01', 'YYYY-MM-DD'), v_cod_dep, 1, 1, NULL);

END;

/

SQL Worksheet

Clear

Find

Actions

Save

Run

```
59 v_avg_sal Angajat.salariu%TYPE;  
60 v_cod_dep Angajat.cod_departament%TYPE;  
61 BEGIN  
62  
63     v_cod_dep := 1;  
64  
65     select avg(salariu)  
66     into v_avg_sal  
67     from angajat  
68     where cod_departament = v_cod_dep  
69     group by cod_departament;  
70  
71     v_avg_sal := v_avg_sal - 10;  
72  
73     INSERT INTO Angajat (cod_angajat, prenume, nume_familie, salariu, data_angajarii, cod_departament, cod_categorie, cod_locatie, cod_manager)  
74     VALUES (100, 'John', 'Doe', v_avg_sal, TO_DATE('2022-01-01', 'YYYY-MM-DD'), v_cod_dep, 1, 1, NULL);  
75  
76 END;  
77 /  
78
```

Trigger created.

ORA-20001: noul angajat trebuie să aibă cel puțin un salariu egal cu 9000. ORA-06512: at "SQL_CZQAPEEEXPZYBINTSMSJHIMWK.SALARIU_ANGAJAT", line 15
ORA-06512: at line 16
ORA-06512: at "SYS.DBMS_SQL", line 1721

More Details: <https://docs.oracle.com/error-help/db/ora-20001>

Pentru cazul micșorării am luat angajatul din exemplu cu salariul 4000 și am încercat să îl updatez salariul la 3200:

DECLARE

 v_cod_angajat Angajat.cod_angajat%TYPE;

 v_sal Angajat.salariu%TYPE;

BEGIN

 v_cod_angajat := 11;

 v_sal := 3200;

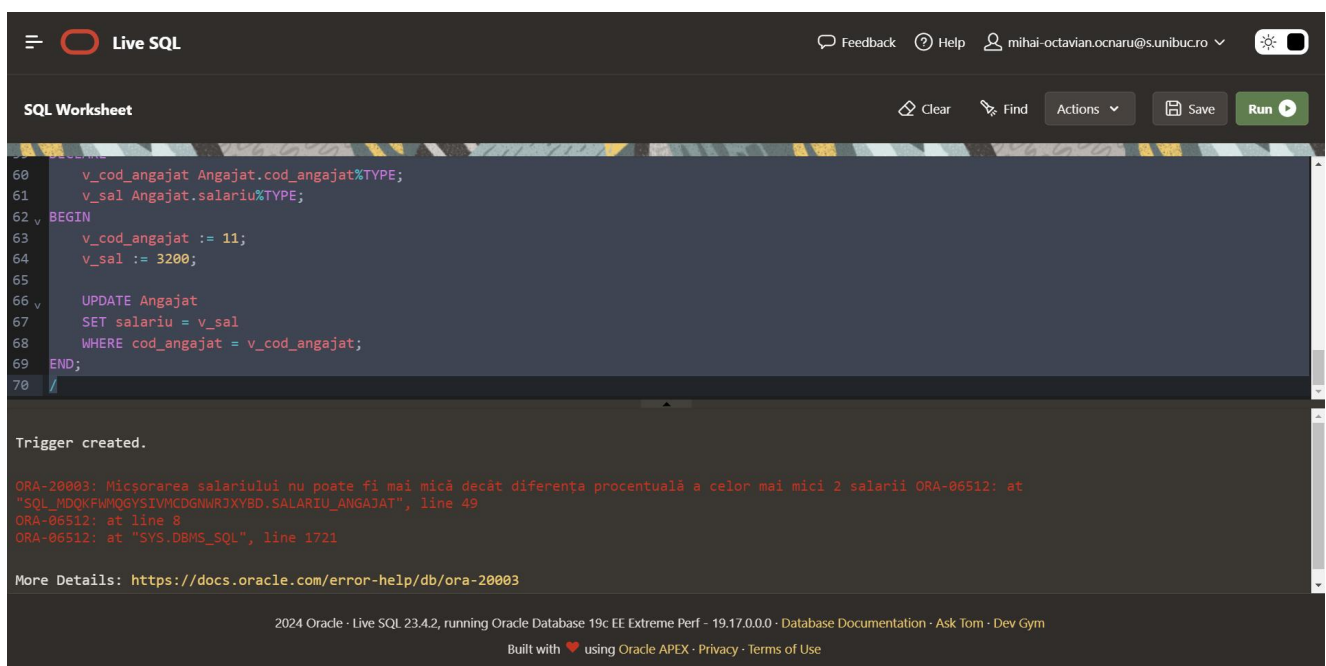
 UPDATE Angajat

 SET salariu = v_sal

 WHERE cod_angajat = v_cod_angajat;

END;

/



The screenshot displays the Oracle Live SQL web interface. At the top, there's a navigation bar with 'Live SQL', 'Feedback', 'Help', a user profile 'mihai-octavian.ocnaru@s.unibuc.ro', and a dark mode toggle. Below this is the 'SQL Worksheet' section with buttons for 'Clear', 'Find', 'Actions', 'Save', and a 'Run' button. The main area contains a PL/SQL block with line numbers 60 to 70. The code declares two variables, assigns values to them, and then updates the 'salariu' field in the 'Angajat' table for the employee with 'cod_angajat' 11. Below the code, the execution results are shown: 'Trigger created.' followed by three Oracle error messages (ORA-20003, ORA-06512, and ORA-06512) related to a salary reduction constraint. A link for 'More Details' points to the Oracle documentation. The footer indicates the version is '2024 Oracle - Live SQL 23.4.2' and provides links for 'Database Documentation', 'Ask Tom', and 'Dev Gym'.

```
60     v_cod_angajat Angajat.cod_angajat%TYPE;
61     v_sal Angajat.salariu%TYPE;
62 BEGIN
63     v_cod_angajat := 11;
64     v_sal := 3200;
65
66     UPDATE Angajat
67     SET salariu = v_sal
68     WHERE cod_angajat = v_cod_angajat;
69 END;
70 /
```

Trigger created.

ORA-20003: Micșorarea salariului nu poate fi mai mică decât diferența procentuală a celor mai mici 2 salarii ORA-06512: at "SQL_MDQKFWMQGYSIVKCDGWRJXYBD.SALARIU_ANGAJAT", line 49
ORA-06512: at line 8
ORA-06512: at "SYS.DBMS_SQL", line 1721

More Details: <https://docs.oracle.com/error-help/db/ora-20003>

2024 Oracle - Live SQL 23.4.2, running Oracle Database 19c EE Extreme Perf - 19.17.0.0.0 - Database Documentation - Ask Tom - Dev Gym
Built with ♥ using Oracle APEX · Privacy · Terms of Use

Iar pentru cazul cresterii salariale am ales un salariu de 150000 ce nu se regaseste in inserarile bazei de date:

DECLARE

 v_cod_angajat Angajat.cod_angajat%TYPE;

 v_sal Angajat.salariu%TYPE;

BEGIN

 v_cod_angajat := 11;

 v_sal := 150000;

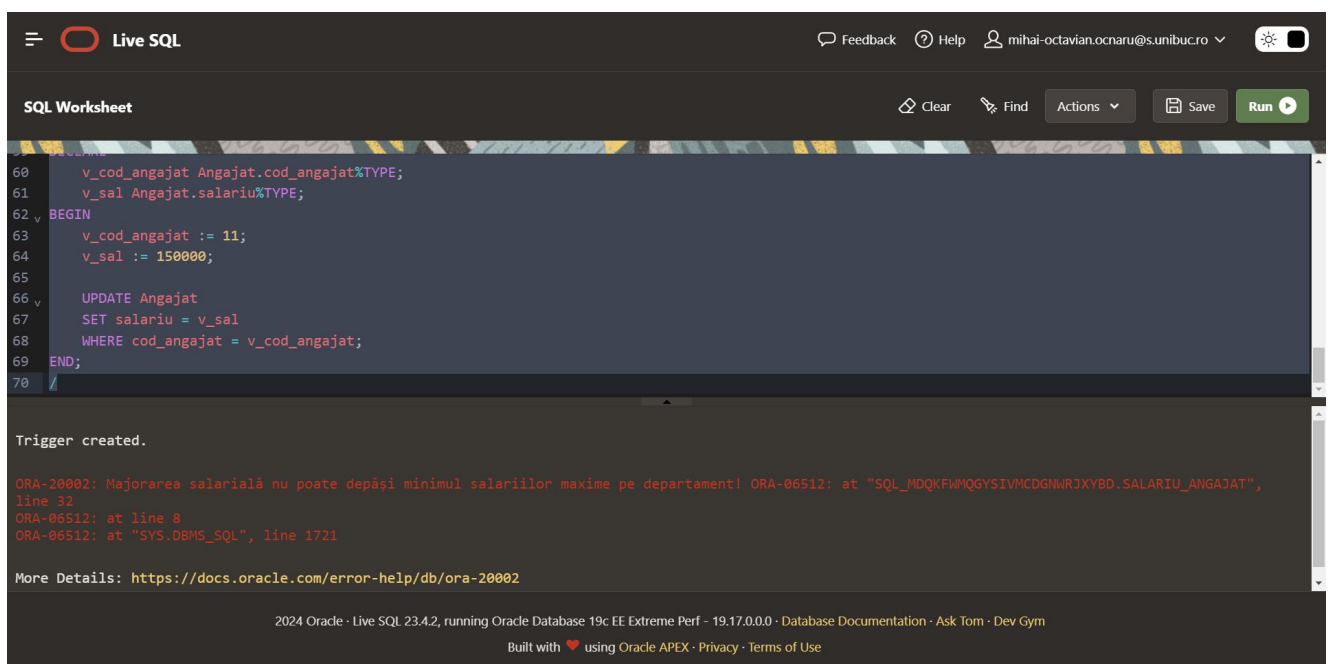
 UPDATE Angajat

 SET salariu = v_sal

 WHERE cod_angajat = v_cod_angajat;

END;

/



The screenshot shows the 'Live SQL' web interface. At the top, there's a navigation bar with 'Live SQL', 'Feedback', 'Help', a user profile 'mihai-octavian.ocnaru@s.unibuc.ro', and a dark mode toggle. Below this is the 'SQL Worksheet' section with buttons for 'Clear', 'Find', 'Actions', 'Save', and a 'Run' button. The main area contains a PL/SQL script with line numbers 60 to 70. The script declares variables, sets values, and updates the 'Angajat' table. Below the script, the execution results are shown, including a success message 'Trigger created.' and several Oracle error messages (ORA-20002, ORA-06512) related to salary constraints and line numbers. At the bottom, there's a footer with version information and a link to Oracle documentation.

```
60     v_cod_angajat Angajat.cod_angajat%TYPE;
61     v_sal Angajat.salariu%TYPE;
62 BEGIN
63     v_cod_angajat := 11;
64     v_sal := 150000;
65
66     UPDATE Angajat
67     SET salariu = v_sal
68     WHERE cod_angajat = v_cod_angajat;
69 END;
70 /
```

Trigger created.

ORA-20002: Majorarea salarială nu poate depăși minimul salariilor maxime pe departament! ORA-06512: at "SQL_MDQKFWMQGYSIVMCDGNWRJXYBD.SALARIU_ANGAJAT", line 32
ORA-06512: at line 8
ORA-06512: at "SYS.DBMS_SQL", line 1721

More Details: <https://docs.oracle.com/error-help/db/ora-20002>

2024 Oracle - Live SQL 23.4.2, running Oracle Database 19c EE Extreme Perf - 19.17.0.0.0 - Database Documentation - Ask Tom - Dev Gym
Built with ♥ using Oracle APEX · Privacy · Terms of Use

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

Sa se creeze un declansator care sa nu permita creerea/editarea/stergera tabelelor decat in timpul programului de munca. Prin program de munca se inteleg zilele de Luni pana Vineri de la ora 8:00 - 16:00. Sa se creeze un tabel in care sa se pastreze arhive ale utilizatorului, data, actiunea si statusul comenzii efectuate.

```
CREATE TABLE logs (
```

```
    username VARCHAR2(30),
```

```
    log_date DATE,
```

```
    sysevent VARCHAR2(20),
```

```
    status VARCHAR2(10)
```

```
);
```

```
CREATE OR REPLACE TRIGGER time_control_trigger
```

```
BEFORE CREATE OR ALTER OR DROP ON SCHEMA
```

```
DECLARE
```

```
    valid_time BOOLEAN;
```

```
    v_status VARCHAR2(10);
```

```
BEGIN
```

```
    v_status := 'Completed';
```

```
    valid_time := TO_CHAR(SYSDATE, 'Dy',  
'NLS_DATE_LANGUAGE=American') IN ('Mon', 'Tue', 'Wed', 'Thu', 'Fri') AND  
        TO_NUMBER(TO_CHAR(SYSDATE, 'HH24')) BETWEEN 8  
AND 16;
```

```
    IF NOT valid_time THEN
```

```
        RAISE_APPLICATION_ERROR(-20001, 'Action not allowed outside  
working hours.');
```

```
        v_status := 'Failed';
```

```
    END IF;
```

```
    INSERT INTO logs (username, log_date, sysevent, status)
```

```
    VALUES (USER, SYSDATE, SYS.SYSEVENT, v_status);
```

```
END;
```

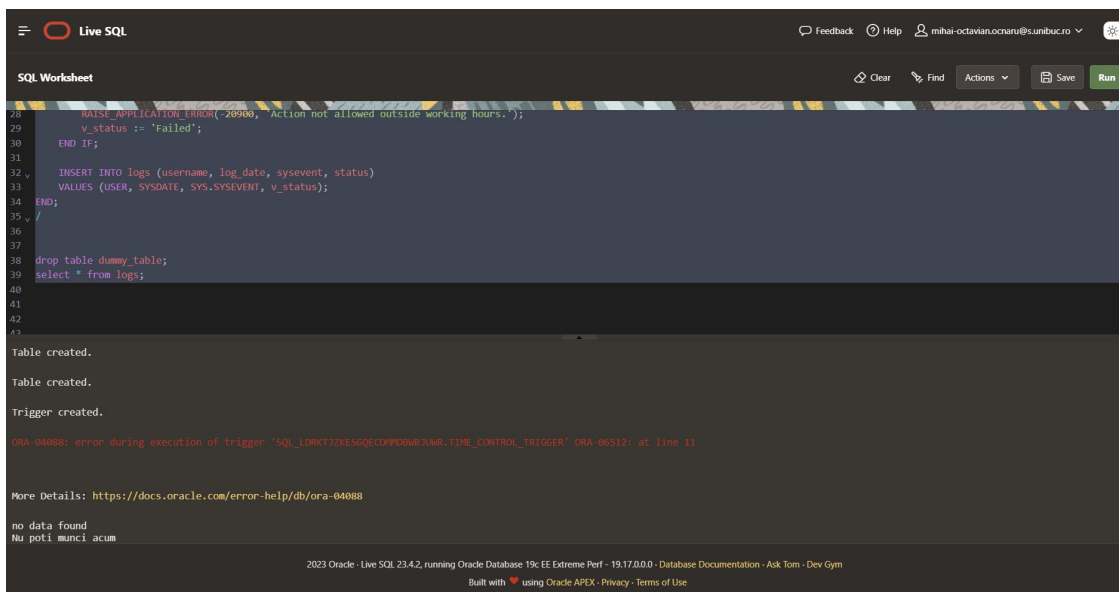
```
/
```

Am creat o tabela dummy inainte de initializarea scriptului:

```
CREATE TABLE dummy_table (  
  
    id NUMBER,  
  
    name VARCHAR2(50),  
  
    created_at TIMESTAMP  
  
);
```

Si am incercat sa o sterg dupa initializarea scriptului:

```
DROP TABLE dummy_table;
```



The screenshot shows the Live SQL interface with a dark theme. The top bar includes the 'Live SQL' logo, a user profile, and navigation links. The main area is divided into a code editor and a results pane. The code editor contains the following SQL script:

```
28 RAISE_APPLICATION_ERROR(-20988, 'Action not allowed outside working hours. ');  
29 v_status := 'Failed';  
30 END IF;  
31  
32  
33 INSERT INTO logs (username, log_date, sysevent, status)  
34 VALUES (USER, SYSDATE, SYS.SYSEVENT, v_status);  
35  
36  
37  
38 drop table dummy_table;  
39 select * from logs;  
40  
41  
42  
43
```

The results pane shows the following output:

```
Table created.  
Table created.  
Trigger created.  
  
ORA-04088: error during execution of trigger 'SQL_LDRKT3ZKESGQECPPKPBWJLWR.TIME_CONTROL_TRIGGER' ORA-06512: at line 11  
  
More Details: https://docs.oracle.com/error-help/db/ora-04088  
  
no data found  
Nu poti munci acum
```

The footer of the interface indicates it is running Oracle Database 19c EE Extreme Perf on 19.17.0.0.0, built with Oracle APEX.

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

```
CREATE OR REPLACE PACKAGE proiect_ocnaru_mihai AS

    PROCEDURE display_user_orders;

    PROCEDURE raport_produce_recenzii;

    FUNCTION raport_produce_tara_categorie(

        v_nume_tara IN VARCHAR2,

        v_nume_categorie IN VARCHAR2,

        v_max_pret IN NUMBER

    ) RETURN VARCHAR2;

    PROCEDURE afisare_tara_max_angajati_categorie_max_pret;

END;

/
```

```
CREATE OR REPLACE PACKAGE BODY proiect_ocnaru_mihai AS

    --6

    PROCEDURE display_user_orders

    AS

        TYPE produs_record IS RECORD (

            cod_produș produs.cod_produș%TYPE,

            pret produs.pret%TYPE,

            nume produs.nume%TYPE

        );

        TYPE vector_produce IS VARRAY(200) OF produs_record;
```

```
TYPE comanda_cu_produse_record IS RECORD (  
    cod_comanda comanda.cod_comanda%TYPE,  
    produse vector_produse  
);
```

```
TYPE comanda_cu_produse_nested_table IS TABLE OF  
comanda_cu_produse_record;
```

```
TYPE utilizator_comanda_indexed_table IS TABLE OF  
comanda_cu_produse_nested_table INDEX BY PLS_INTEGER;
```

```
utilizzatori_cu_comenzi utilizator_comanda_indexed_table;
```

```
BEGIN
```

```
FOR utilizator IN (SELECT DISTINCT cod_utilizator FROM comanda)
```

```
LOOP
```

```
    utilizatori_cu_comenzi(utilizator.cod_utilizator) :=  
comanda_cu_produse_nested_table();
```

```
    DBMS_OUTPUT.PUT_LINE('Utilizator: ' ||  
utilizator.cod_utilizator);
```

```
    FOR comanda IN (SELECT cod_comanda FROM comanda  
WHERE cod_utilizator = utilizator.cod_utilizator)
```

```
    LOOP
```

```
        DECLARE
```

```
            produse vector_produse := vector_produse();
```

```

BEGIN

    FOR produs IN (SELECT p.cod_produs, p.pret, p.num
                    FROM produs p,
produs_se_afla_in_comanda pc
                    WHERE p.cod_produs = pc.cod_produs
                    AND pc.cod_comanda =
comanda.cod_comanda)
    LOOP
        produse.extend();
        produse(produse.last) := produs_record(
            cod_produs => produs.cod_produs,
            pret => produs.pret,
            nume => produs.nume
        );
    END LOOP;

    utilizatori_cu_comenzi(utilizator.cod_utilizator).EXTEND;

    utilizatori_cu_comenzi(utilizator.cod_utilizator)(utilizatori_cu_comenzi(
    utilizator.cod_utilizator).LAST) :=
        comanda_cu_produse_record(
            cod_comanda => comanda.cod_comanda,
            produse => produse
        );

```

```

        DBMS_OUTPUT.PUT_LINE('    Comanda: ' ||
comanda.cod_comanda);

        FOR i IN 1..produse.count
        LOOP

            DBMS_OUTPUT.PUT_LINE('        Produs: ' ||
produse(i).cod_produs ||

                                ', Nume: ' || produse(i).nume ||

                                ', Pret: ' || produse(i).pret);

        END LOOP;

    END;

END LOOP;

DBMS_OUTPUT.PUT_LINE('-----');

DBMS_OUTPUT.NEW_LINE;

END LOOP;

END display_user_orders;

```

--7

PROCEDURE raport_produse_recenzii AS

CURSOR curson_produse IS

```

    SELECT distinct p.cod_produs, p.nume, p.pret
    FROM produs p, produs_se_afla_in_comanda pc, comanda c
    where pc.cod_produs = p.cod_produs
    and pc.cod_comanda = c.cod_comanda
    and c.cod_utilizator in (select distinct u.cod_utilizator

```



```
from locatie l, utilizator u, tara t
where l.cod_locatie = u.cod_locatie
and l.cod_tara = t.cod_tara
and initcap(t.num) in ('România',
'Franța', 'Spania', 'Italia'));
```

```
CURSOR curson_recenzii (v_cod_produ INT) IS
```

```
SELECT descriere, rating
```

```
FROM Review
```

```
WHERE cod_produ = v_cod_produ;
```

```
v_cod_produ Produ.cod_produ%TYPE;
```

```
v_num_produ Produ.num%TYPE;
```

```
v_pret Produ.pret%TYPE;
```

```
v_descriere Review.descriere%TYPE;
```

```
v_rating Review.rating%TYPE;
```

```
BEGIN
```

```
OPEN curson_produce;
```

```
LOOP
```

```
FETCH curson_produce INTO v_cod_produ,
v_num_produ, v_pret;
```

```
EXIT WHEN curson_produce%NOTFOUND;
```

```
OPEN curson_recenzii(v_cod_produ);
```

LOOP

FETCH curson_recenzii INTO v_descriere, v_rating;

EXIT WHEN curson_recenzii%NOTFOUND;

DBMS_OUTPUT.PUT_LINE('Produs: ' || v_num_produs ||
, Pret: ' || v_pret);

DBMS_OUTPUT.PUT_LINE(' Recenzie: ' || v_descriere || ',
Rating: ' || v_rating);

DBMS_OUTPUT.PUT_LINE('-----');

DBMS_OUTPUT.NEW_LINE;

END LOOP;

CLOSE curson_recenzii;

END LOOP;

CLOSE curson_produce;

END raport_produce_recenzii;

--8

FUNCTION raport_produce_tara_categorie(
v_num_tara IN VARCHAR2,
v_num_categorie IN VARCHAR2,
v_max_pret IN NUMBER
) RETURN VARCHAR2
AS
v_raport VARCHAR2(4000);

invalid_country_name EXCEPTION;

PRAGMA EXCEPTION_INIT(invalid_country_name, -1333);

too_small_price EXCEPTION;

PRAGMA EXCEPTION_INIT(too_small_price, -1332);

BEGIN

v_raport := 'Products in ' || v_num_tara || ', Category: ' ||
v_num_categorie || ' with price <= ' || v_max_pret || ':';

DECLARE

min_price produs.pret%TYPE;

BEGIN

select min(p.pret)

into min_price

FROM tara t, locatie l, furnizor f,
furnizor_distributie_in_locatia fl, produs p

WHERE t.cod_tara = l.cod_tara

AND f.cod_furnizor = fl.cod_furnizor

AND fl.cod_locatie = l.cod_locatie

AND f.cod_furnizor = p.cod_furnizor;

if min_price > v_max_pret then

```
        RAISE too_small_price;
    end if;
END;
```

```
DECLARE
```

```
    flag Boolean := false;
```

```
BEGIN
```

```
    FOR country IN (
```

```
        SELECT 1
```

```
        FROM dual
```

```
        WHERE EXISTS (
```

```
            SELECT 1
```

```
            FROM tara t, locatie l, furnizor f,  
furnizor_distribuite_in_locatia fl
```

```
            WHERE t.cod_tara = l.cod_tara
```

```
            AND f.cod_furnizor = fl.cod_furnizor
```

```
            AND fl.cod_locatie = l.cod_locatie
```

```
            AND t.nume = v_nume_tara
```

```
        )
```

```
    ) LOOP
```

```
        flag := true;
```

```
        EXIT;
```

```
    END LOOP;
```

```
    IF NOT flag THEN
```

```
        RAISE invalid_country_name;
```

END IF;

END;

FOR prod_rec IN (

SELECT distinct p.nume, p.pret

FROM produs p, produs_se_afla_in_comanda pc, comanda c,
locatie l, tara t, categorii cat, furnizor f, furnizor_distributie_in_locatia fl

where p.cod_produs = pc.cod_produs

and pc.cod_comanda = c.cod_comanda

and l.cod_tara = t.cod_tara

and p.cod_categorie = cat.cod_categorie

and f.cod_furnizor = fl.cod_furnizor

and fl.cod_locatie = l.cod_locatie

and f.cod_furnizor = p.cod_furnizor

and initcap(t.nume) = initcap(v_nume_tara)

AND initcap(cat.nume)= initcap(v_nume_categorie)

AND p.pret <= v_max_pret

) LOOP

v_raport := v_raport || CHR(10) || 'Product: ' || prod_rec.nume
|| ', Price: ' || prod_rec.pret;

END LOOP;

IF v_raport = 'Products in ' || v_nume_tara || ', Category: ' ||
v_nume_categorie || ' with price <= ' || v_max_pret || ':' THEN

```

        RAISE NO_DATA_FOUND;

    END IF;

    RETURN v_raport;

EXCEPTION

    WHEN invalid_country_name THEN

        DBMS_OUTPUT.PUT_LINE('Error: The country name given is
not in the database.');
```

RAISE_APPLICATION_ERROR(-20001, 'Invalid country name');

```

        RETURN NULL;

    WHEN too_small_price THEN

        DBMS_OUTPUT.PUT_LINE('Error: The max price given is too
small compared to the database prices.');
```

RAISE_APPLICATION_ERROR(-20001, 'Price too small');

```

        RETURN NULL;

    WHEN NO_DATA_FOUND THEN

        DBMS_OUTPUT.PUT_LINE('Error: No products found for the
given criteria.');
```

RAISE_APPLICATION_ERROR(-20002, 'No products found for
the given criteria');

```

        RETURN NULL;

    WHEN OTHERS THEN

        DBMS_OUTPUT.PUT_LINE('An error occurred: ' || SQLERRM);
```

RETURN NULL;

END raport_produse_tara_categorie;

--9

PROCEDURE afisare_tara_max_angajati_categorie_max_pret AS

v_nume_tara Tara.nume%TYPE;

v_numar_angajati Number;

BEGIN

WITH tari as (

nr_angajati
SELECT DISTINCT t.nume, COUNT(COD_ANGAJAT) AS

FROM tara t

JOIN locatie l ON t.cod_tara = l.cod_tara

JOIN angajat a ON l.cod_locatie = a.cod_locatie

WHERE a.cod_categorie IN (

SELECT COD_CATEGORIE

FROM produs

GROUP BY COD_CATEGORIE

HAVING AVG(PRET) = (

SELECT MAX(AVG(PRET))

FROM PRODUS

GROUP BY COD_CATEGORIE

)

)

GROUP BY t.cod_tara, t.nume

```

)

SELECT tari.numa, tari.nr_angajati
INTO v_nume_tara, v_numar_angajati
FROM tari
GROUP BY tari.numa, tari.nr_angajati
HAVING tari.nr_angajati = (SELECT MAX(nr_angajati)
                           FROM tari
                           );

```

```

DBMS_OUTPUT.PUT_LINE('Tara cu cei mai multi angajati care
lucreaza la categoria cu pretul mediu cel mai mare: ' || v_nume_tara || '
numar angajati: ' || v_numar_angajati);

```

```

EXCEPTION

```

```

    WHEN NO_DATA_FOUND THEN

```

```

        DBMS_OUTPUT.PUT_LINE('Nu există date pentru cerință.');
```

```

    WHEN TOO_MANY_ROWS THEN

```

```

        DBMS_OUTPUT.PUT_LINE('Eroare: Prea multe rânduri
returnate.');
```

```

    WHEN OTHERS THEN

```

```

        DBMS_OUTPUT.PUT_LINE('A apărut o eroare: ' || SQLERRM);

```

```

END afisare_tara_max_angajati_categorie_max_pret;

```

```

END proiect_ocnaru_mihai;

```

```

/

```


Apelam ex. 6 din pachet:

EXECUTE proiect_ocnaru_mihai.display_user_orders();

Live SQL

Feedback Help mihai-octavian.ocnaru@s.unibuc.ro

SQL Worksheet

Clear Find Actions Save Run

```
252 SELECT tari.numa, tari.nr_angajati
253 INTO v_numa_tara, v_numar_angajati
254 FROM tari
255 GROUP BY tari.numa, tari.nr_angajati
256 HAVING tari.nr_angajati = (SELECT MAX(nr_angajati)
257 FROM tari
258 );
259
260 DBMS_OUTPUT.PUT_LINE('Tara cu cei mai multi angajati care lucreaza la categoria cu pretul mediu cel mai mare: ' || v_numa_tara || ' numar angajati: ' || v_numar_angajati);
261
262 EXCEPTION
263 WHEN NO_DATA_FOUND THEN
264 DBMS_OUTPUT.PUT_LINE('Nu există date pentru cerință.');
```

Statement processed.
Utilizator: 7
Comanda: 9
Produs: 4, Nume: Televizor, Pret: 1999.99
Produs: 9, Nume: Geacă, Pret: 199.99
Produs: 23, Nume: Ulei de măsline, Pret: 9.99
Utilizator: 15

2024 Oracle - Live SQL 23.4.2, running Oracle Database 19c EE Extreme Perf - 19.17.0.0.0 - Database Documentation - Ask Tom - Dev Gym
Built with using Oracle APEX - Privacy - Terms of Use

Apelam ex. 7 din pachet:

EXECUTE proiect_ocnaru_mihai.raport_produce_recenzii();

Live SQL

Feedback Help mihai-octavian.ocnaru@s.unibuc.ro

SQL Worksheet

Clear Find Actions Save Run

```
252 SELECT tari.numa, tari.nr_angajati
253 INTO v_numa_tara, v_numar_angajati
254 FROM tari
255 GROUP BY tari.numa, tari.nr_angajati
256 HAVING tari.nr_angajati = (SELECT MAX(nr_angajati)
257 FROM tari
258 );
259
260 DBMS_OUTPUT.PUT_LINE('Tara cu cei mai multi angajati care lucreaza la categoria cu pretul mediu cel mai mare: ' || v_numa_tara || ' numar angajati: ' || v_numar_angajati);
261
262 EXCEPTION
263 WHEN NO_DATA_FOUND THEN
264 DBMS_OUTPUT.PUT_LINE('Nu există date pentru cerință.');
```

Statement processed.
Produs: Televizor, Pret: 1999.99

Recenzie: Produs de calitate superioara, Rating: 5

Produs: Dulap, Pret: 1499.99
Recenzie: Excelent produs!, Rating: 5

Produs: Pat, Pret: 699.99

2024 Oracle - Live SQL 23.4.2, running Oracle Database 19c EE Extreme Perf - 19.17.0.0.0 - Database Documentation - Ask Tom - Dev Gym
Built with using Oracle APEX - Privacy - Terms of Use

Apelam ex. 8 din pachet:

Apel reusit al functiei

DECLARE

v_result VARCHAR2(4000);

BEGIN

v_result :=
proiect_ocnaru_mihai.raport_produce_tara_categorie('România',
'Electronice', 3000);

IF v_result IS NOT NULL THEN

DBMS_OUTPUT.PUT_LINE(v_result);

END IF;

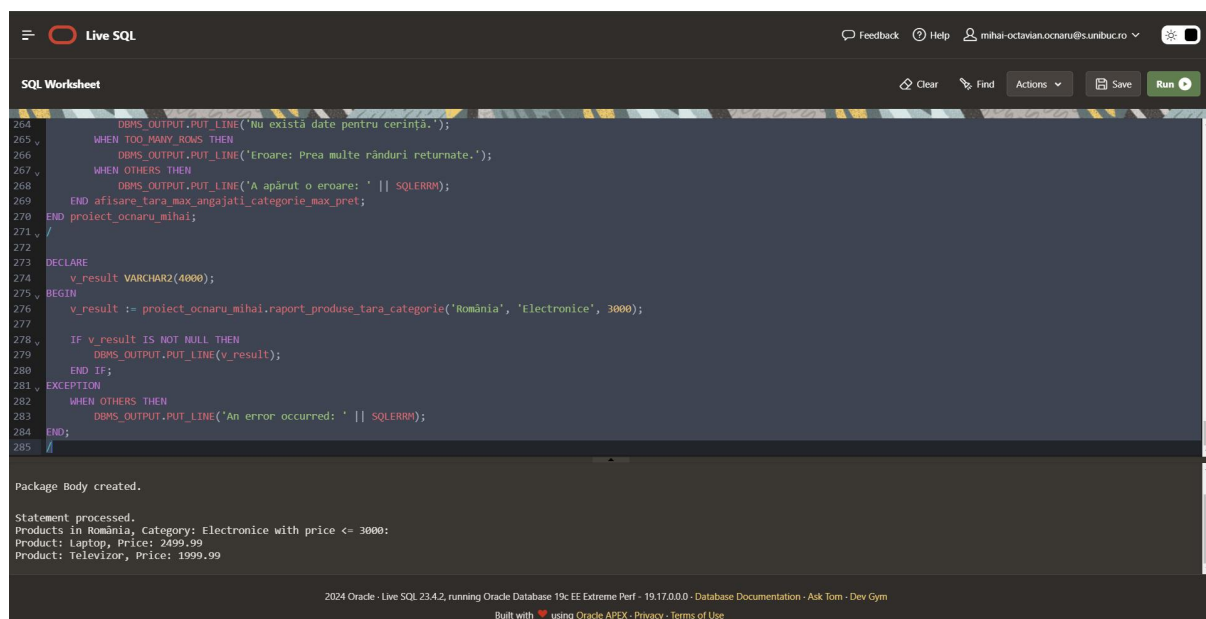
EXCEPTION

WHEN OTHERS THEN

DBMS_OUTPUT.PUT_LINE('An error occurred: ' || SQLERRM);

END;

/



```
264 DBMS_OUTPUT.PUT_LINE('Nu există date pentru cerința.');
```

```
265 WHEN TOO_MANY_ROWS THEN
```

```
266 DBMS_OUTPUT.PUT_LINE('Eroare: Prea multe rânduri returnate.');
```

```
267 WHEN OTHERS THEN
```

```
268 DBMS_OUTPUT.PUT_LINE('A apărut o eroare: ' || SQLERRM);
```

```
269 END afisare_tara_max_angajati_categorie_max_pret;
```

```
270 END proiect_ocnaru_mihai;
```

```
271 /
```

```
272
```

```
273 DECLARE
```

```
274 v_result VARCHAR2(4000);
```

```
275 BEGIN
```

```
276 v_result := proiect_ocnaru_mihai.raport_produce_tara_categorie('România', 'Electronice', 3000);
```

```
277
```

```
278 IF v_result IS NOT NULL THEN
```

```
279 DBMS_OUTPUT.PUT_LINE(v_result);
```

```
280 END IF;
```

```
281 EXCEPTION
```

```
282 WHEN OTHERS THEN
```

```
283 DBMS_OUTPUT.PUT_LINE('An error occurred: ' || SQLERRM);
```

```
284 END;
```

```
285 /
```

Package Body created.

Statement processed.

Products in România, Category: Electronice with price <= 3000:

Product: Laptop, Price: 2499.99

Product: Televizor, Price: 1999.99

2024 Oracle - Live SQL 23.4.2, running Oracle Database 19c EE Extreme Perf - 19.17.0.0.0 - Database Documentation - Ask Tom - Dev Gym

Built with using Oracle APEX - Privacy - Terms of Use

Apel pentru o tara inexistentă

Declaram codul pentru o tara inexistentă in baza de date:

DECLARE

 v_result VARCHAR2(4000);

BEGIN

 v_result :=
proiect_ocnaru_mihai.raport_produce_tara_categorie('China',
'Electronice', 3000);

 IF v_result IS NOT NULL THEN

 DBMS_OUTPUT.PUT_LINE(v_result);

 END IF;

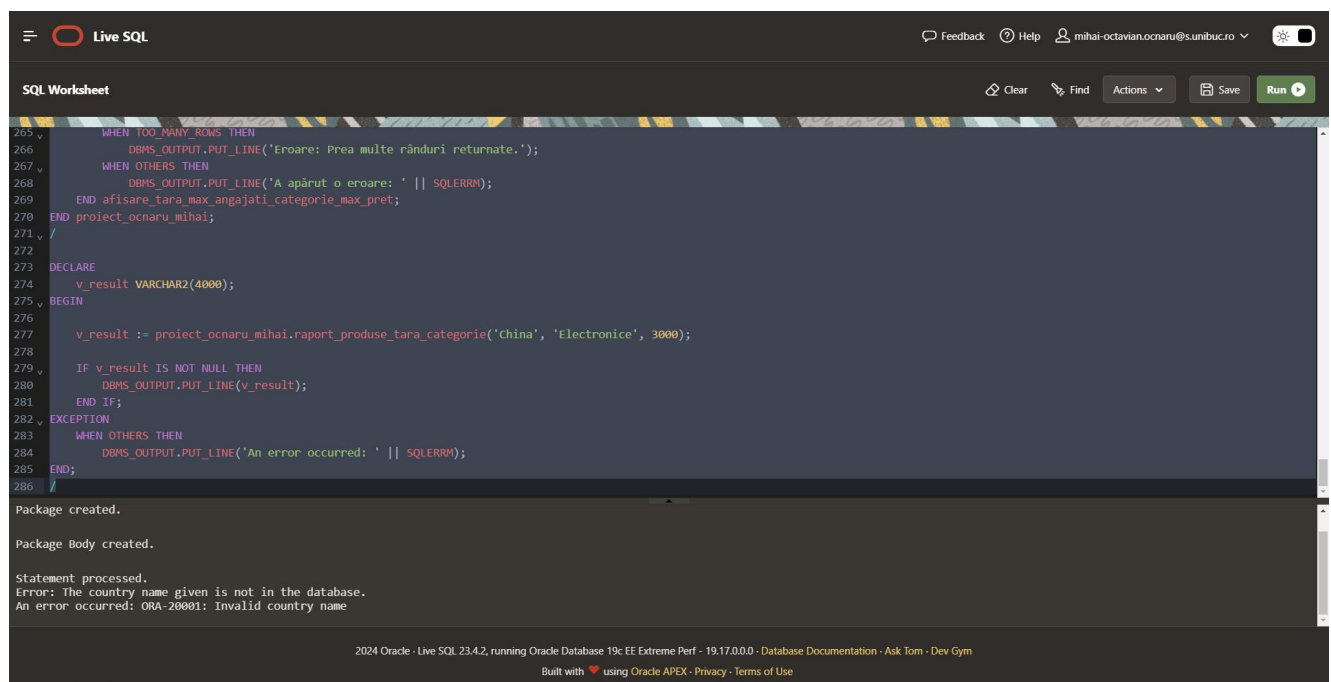
EXCEPTION

 WHEN OTHERS THEN

 DBMS_OUTPUT.PUT_LINE('An error occurred: ' || SQLERRM);

END;

/



Live SQL

Feedback Help mihai-octavian.ocnaru@s.unibuc.ro

SQL Worksheet Clear Find Actions Save Run

```
265 WHEN TOO_MANY_ROWS THEN
266     DBMS_OUTPUT.PUT_LINE('Eroare: Prea multe rânduri returnate. ');
267 WHEN OTHERS THEN
268     DBMS_OUTPUT.PUT_LINE('A apărut o eroare: ' || SQLERRM);
269 END afisare_tara_max_angajati_categorie_max_pret;
270 END proiect_ocnaru_mihai;
271 /
272
273 DECLARE
274     v_result VARCHAR2(4000);
275 BEGIN
276
277     v_result := proiect_ocnaru_mihai.raport_produce_tara_categorie('China', 'Electronice', 3000);
278
279     IF v_result IS NOT NULL THEN
280         DBMS_OUTPUT.PUT_LINE(v_result);
281     END IF;
282 EXCEPTION
283     WHEN OTHERS THEN
284         DBMS_OUTPUT.PUT_LINE('An error occurred: ' || SQLERRM);
285 END;
```

Package created.

Package Body created.

Statement processed.

Error: The country name given is not in the database.

An error occurred: ORA-20801: Invalid country name

2024 Oracle - Live SQL 23.4.2, running Oracle Database 19c EE Extreme Perf - 19.17.0.0.0 - Database Documentation - Ask Tom - Dev Gym

Built with using Oracle APEX - Privacy - Terms of Use

Apel pentru un pret maxim inexistent

Declaram codul pentru un pret maxim prea mic:

DECLARE

v_result VARCHAR2(4000);

BEGIN

v_result :=
proiect_ocnaru_mihai.raport_produce_tara_categorie('România',
'Electronice', 0.5);

IF v_result IS NOT NULL THEN

DBMS_OUTPUT.PUT_LINE(v_result);

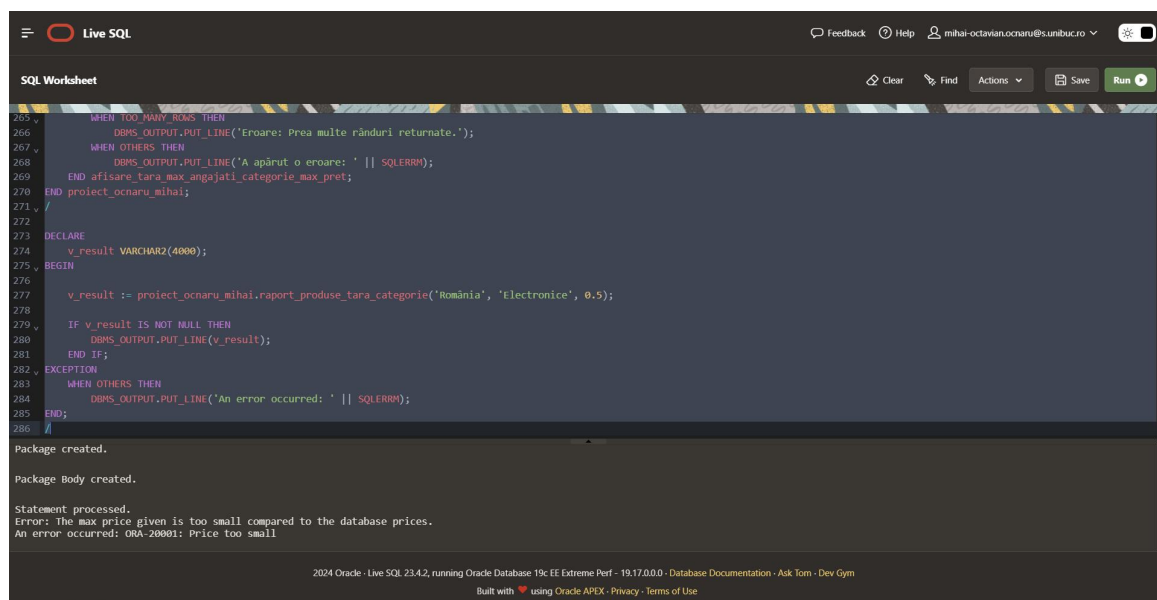
END IF;

EXCEPTION

WHEN OTHERS THEN

DBMS_OUTPUT.PUT_LINE('An error occurred: ' || SQLERRM);

END;/



The screenshot shows a web-based SQL editor interface. At the top, there's a header with 'Live SQL' and user information. Below that, a toolbar contains 'Clear', 'Find', 'Actions', 'Save', and 'Run' buttons. The main area is a code editor with a dark theme, displaying a PL/SQL script. The script includes a package declaration, a variable declaration, a BEGIN block with a function call, an IF statement, and an EXCEPTION block. The script is executed, and the output pane at the bottom shows the results: 'Package created.', 'Package Body created.', 'Statement processed.', and an error message: 'Error: The max price given is too small compared to the database prices. An error occurred: ORA-20001: Price too small'.

```
265 WHEN TOO_MANY_ROWS THEN
266     DBMS_OUTPUT.PUT_LINE('Eroare: Prea multe rânduri returnate.');
```

```
267 WHEN OTHERS THEN
268     DBMS_OUTPUT.PUT_LINE('A apărut o eroare: ' || SQLERRM);
269 END afisare_tara_max_angajati_categorie_max_pret;
270 END proiect_ocnaru_mihai;
271 /
272
273 DECLARE
274     v_result VARCHAR2(4000);
275 BEGIN
276     v_result := proiect_ocnaru_mihai.raport_produce_tara_categorie('România', 'Electronice', 0.5);
277
278     IF v_result IS NOT NULL THEN
279         DBMS_OUTPUT.PUT_LINE(v_result);
280     END IF;
281
282 EXCEPTION
283     WHEN OTHERS THEN
284         DBMS_OUTPUT.PUT_LINE('An error occurred: ' || SQLERRM);
285 END;
```

Package created.

Package Body created.

Statement processed.

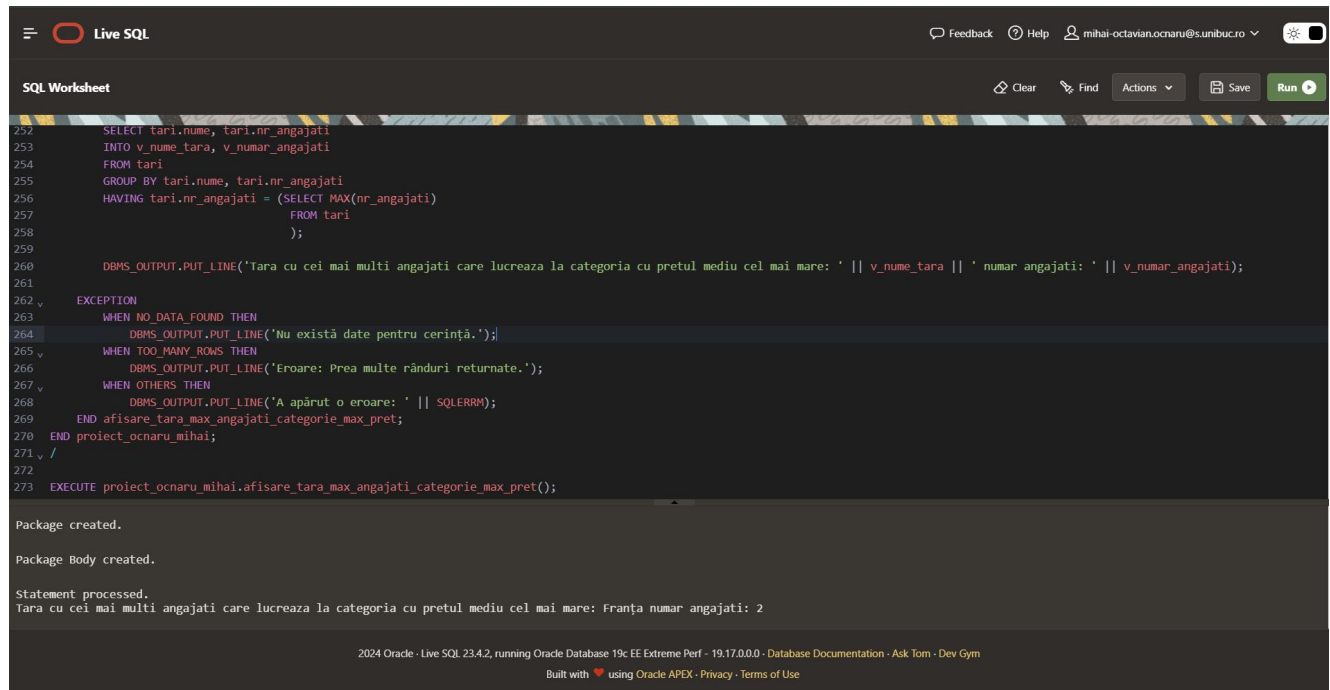
Error: The max price given is too small compared to the database prices.
An error occurred: ORA-20001: Price too small

2024 Oracle - Live SQL 23.4.2, running Oracle Database 19c EE Extreme Perf - 19.17.0.0.0 - Database Documentation - Ask Tom - Dev Gym
Built with ❤️ using Oracle APEX - Privacy - Terms of Use

Apelam ex. 9 din pachet:

EXECUTE

proiect_ocnaru_mihai.afisare_tara_max_angajati_categorie_max_pret
());



The screenshot shows the Live SQL interface with a dark theme. At the top, there's a header with 'Live SQL', a user profile 'mihai-octavian.ocnaru@sunibuc.ro', and a dark mode toggle. Below the header is a toolbar with 'Clear', 'Find', 'Actions', 'Save', and 'Run' buttons. The main area is an 'SQL Worksheet' containing a PL/SQL script. The script defines a procedure to find the country with the highest number of employees in a category with the highest average salary. It includes a SELECT statement with a subquery, a DBMS_OUTPUT.PUT_LINE statement, and an EXCEPTION block for NO_DATA_FOUND, TOO_MANY_ROWS, and OTHERS. The script is executed, and the output shows the package creation, package body creation, and the statement processing result: 'Tara cu cei mai multi angajati care lucreaza la categoria cu pretul mediu cel mai mare: Franța numar angajati: 2'. At the bottom, there's a footer with Oracle version information and a note about using Oracle APEX.

```
252 SELECT tari.ume, tari.nr_angajati
253 INTO v_ume_tara, v_numar_angajati
254 FROM tari
255 GROUP BY tari.ume, tari.nr_angajati
256 HAVING tari.nr_angajati = (SELECT MAX(nr_angajati)
257 FROM tari
258 );
259
260 DBMS_OUTPUT.PUT_LINE('Tara cu cei mai multi angajati care lucreaza la categoria cu pretul mediu cel mai mare: ' || v_ume_tara || ' numar angajati: ' || v_numar_angajati);
261
262 EXCEPTION
263 WHEN NO_DATA_FOUND THEN
264 DBMS_OUTPUT.PUT_LINE('Nu există date pentru cerință.');
```

Apel pentru TOO_MANY_ROWS

BEGIN

--Pentru exemplificarea cazului TOO_MANY_ROWS am inserat doi
-- angajati ce lucreaza in categoria 1 si sunt situati in Spania:

INSERT INTO Angajat (cod_angajat, prenume, nume_familie, salariu,
data_angajarii, cod_departament, cod_categorie, cod_locatie,
cod_manager)

VALUES (70, 'Cristian', 'Dinu', 9000, TO_DATE('2019-09-21', 'yyyy-mm-dd'), 7, 1, 7, 3);

INSERT INTO Angajat (cod_angajat, prenume, nume_familie, salariu,
data_angajarii, cod_departament, cod_categorie, cod_locatie,
cod_manager)

VALUES (80, 'Cristian', 'Dinu', 9000, TO_DATE('2019-09-21', 'yyyy-mm-dd'), 7, 1, 7, 3);

```
proiect_ocnaru_mihai.afisare_tara_max_angajati_categorie_max_pret  
(  
);  
END;  
/  

```

```
261  
262 EXCEPTION  
263 WHEN NO_DATA_FOUND THEN  
264 DBMS_OUTPUT.PUT_LINE('Nu există date pentru cerință.');
```

```
265 WHEN TOO_MANY_ROWS THEN  
266 DBMS_OUTPUT.PUT_LINE('Eroare: Prea multe rânduri returnate.');
```

```
267 WHEN OTHERS THEN  
268 DBMS_OUTPUT.PUT_LINE('A apărut o eroare: ' || SQLERRM);  
269 END afisare_tara_max_angajati_categorie_max_pret;  
270 END proiect_ocnaru_mihai;  
271 /  
272  
273 BEGIN  
274 --Pentru exemplificarea cazului TOO_MANY_ROWS am inserat doi  
275 -- angajati ce lucreaza in categoria 1 si sunt situati in Spania:  
276 INSERT INTO Angajat (cod_angajat, prenume, nume_familie, salariu, data_angajarii, cod_departament, cod_categorie, cod_locatie, cod_manager)  
277 VALUES (70, 'Cristian', 'Dinu', 9000, TO_DATE('2019-09-21', 'yyyy-mm-dd'), 7, 1, 7, 3);  
278 INSERT INTO Angajat (cod_angajat, prenume, nume_familie, salariu, data_angajarii, cod_departament, cod_categorie, cod_locatie, cod_manager)  
279 VALUES (80, 'Cristian', 'Dinu', 9000, TO_DATE('2019-09-21', 'yyyy-mm-dd'), 7, 1, 7, 3);  
280 proiect_ocnaru_mihai.afisare_tara_max_angajati_categorie_max_pret();  
281 END;  
282 /
```

Package created.

Package Body created.

Statement processed.
Eroare: Prea multe rânduri returnate.

2024 Oracle - Live SQL 23.4.2, running Oracle Database 19c EE Extreme Perf - 19.17.0.0.0 - Database Documentation - Ask Tom - Dev Gym
Built with ❤️ using Oracle APEX - Privacy - Terms of Use

Apel pentru NO_DATA_FOUND

Am rulat fara a face inserarea in baza de date

```
259 DBMS_OUTPUT.PUT_LINE('Tara cu cei mai multi angajati care lucreaza la categoria cu pretul mediu cel mai mare: ' || v_nume_tara || ' numar angajati: ' || v_numar_angajati);  
260  
261  
262 EXCEPTION  
263 WHEN NO_DATA_FOUND THEN  
264 DBMS_OUTPUT.PUT_LINE('Nu există date pentru cerință.');
```

```
265 WHEN TOO_MANY_ROWS THEN  
266 DBMS_OUTPUT.PUT_LINE('Eroare: Prea multe rânduri returnate.');
```

```
267 WHEN OTHERS THEN  
268 DBMS_OUTPUT.PUT_LINE('A apărut o eroare: ' || SQLERRM);  
269 END afisare_tara_max_angajati_categorie_max_pret;  
270 END proiect_ocnaru_mihai;  
271 /  
272  
273 EXECUTE proiect_ocnaru_mihai.afisare_tara_max_angajati_categorie_max_pret();
```

Package created.

Package Body created.

Statement processed.
Nu există date pentru cerință.

2024 Oracle - Live SQL 23.4.2, running Oracle Database 19c EE Extreme Perf - 19.17.0.0.0 - Database Documentation - Ask Tom - Dev Gym
Built with ❤️ using Oracle APEX - Privacy - Terms of Use

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

Sa se defineasca un pachet care sa cuprinda:

- **O procedura pentru afisarea codului comenzilor, username-ul si numele produselor pentru un utilizator dat**
- **O procedura pentru afisarea codului de review, al descrierii, al ratingului si numele produselor care au fost evaluate de un utilizator dat**
- **O functie care va returna un array ce cuprinde suma produselor dintr-o comanda, impreuna cu codul comenzii**
- **O functie ce returneaza ratingul mediu al produselor evaluate de utilizator**

CREATE OR REPLACE PACKAGE EX14_Package AS

TYPE ProductArray is VARRAY(200) of produs.ume%TYPE;

TYPE OrderRecord IS RECORD (

cod_comanda Comanda.cod_comanda%TYPE,

username Utilizator.ume_utilizator%TYPE,

product_names ProductArray

);

TYPE TotalCostRecord IS RECORD (

cod_comanda Comanda.cod_comanda%TYPE,

total_cost produs.pret%TYPE

);

TYPE TotalOrderCostArray is VARRAY(500) OF TotalCostRecord;

TYPE AverageRatingRecord IS RECORD (

product_id Produs.cod_produs%TYPE,

product_name Produs.numename%TYPE,

average_rating NUMBER

);

TYPE AverageRatingArray is VARRAY(500) OF AverageRatingRecord;

PROCEDURE DisplayOrderDetails(userId IN
Utilizator.cod_utilizator%TYPE);

PROCEDURE DisplayReviewData(userId IN
Utilizator.cod_utilizator%TYPE);

FUNCTION CalculateTotalCost(userId IN
Utilizator.cod_utilizator%TYPE)

RETURN TotalOrderCostArray;

FUNCTION AverageRatingForUser(userId IN
Utilizator.cod_utilizator%TYPE)

RETURN AverageRatingArray;

END EX14_Package;

/

CREATE OR REPLACE PACKAGE BODY Ex14_Package as

 PROCEDURE DisplayOrderDetails(userId IN
Utilizator.cod_utilizator%TYPE) IS

 CURSOR order_cursor IS

 SELECT c.cod_comanda, u.ume_utilizator, p.ume

 FROM Comanda c

 JOIN Utilizator u ON c.cod_utilizator = u.cod_utilizator

 JOIN Produs_se_afla_in_Comanda pc ON c.cod_comanda =
pc.cod_comanda

 JOIN Produs p ON pc.cod_produs = p.cod_produs

 WHERE u.cod_utilizator = userId;

 order_rec order_cursor%ROWTYPE;

 BEGIN

 OPEN order_cursor;

 LOOP

 FETCH order_cursor INTO order_rec;

 EXIT WHEN order_cursor%NOTFOUND;

 DBMS_OUTPUT.PUT_LINE('ID comanda: ' ||
order_rec.cod_comanda || ' | username: ' || order_rec.ume_utilizator || ' |
ume produs: ' || order_rec.ume);

 END LOOP;

 CLOSE order_cursor;

END DisplayOrderDetails;

PROCEDURE DisplayReviewData(userId IN
Utilizator.cod_utilizator%TYPE) IS

CURSOR review_cursor IS

SELECT r.cod_review, r.descriere, r.rating, p.num

FROM Review r

JOIN Produs p ON r.cod_produs = p.cod_produs

JOIN Utilizator u ON r.cod_utilizator = u.cod_utilizator

WHERE u.cod_utilizator = userId;

review_rec review_cursor%ROWTYPE;

BEGIN

OPEN review_cursor;

LOOP

FETCH review_cursor INTO review_rec;

EXIT WHEN review_cursor%NOTFOUND;

DBMS_OUTPUT.PUT_LINE('ID review: ' ||
review_rec.cod_review || ' | descriere: ' || review_rec.descriere || ' | rating:
' || review_rec.rating || ' | nume produs: ' || review_rec.num);

END LOOP;

CLOSE review_cursor;

END DisplayReviewData;

FUNCTION CalculateTotalCost(userId IN
Utilizator.cod_utilizator%TYPE)

```

RETURN TotalOrderCostArray IS

total_cost_array TotalOrderCostArray := TotalOrderCostArray();

total_cost_record TotalCostRecord;

total_cost produs.pret%TYPE;

BEGIN

    FOR order_rec IN (

        SELECT c.cod_comanda, SUM(p.pret) AS total_cost

        FROM Comanda c

        JOIN Utilizator u ON c.cod_utilizator = u.cod_utilizator

        JOIN Produs_se_afla_in_Comanda pc ON c.cod_comanda =

pc.cod_comanda

        JOIN Produs p ON pc.cod_produs = p.cod_produs

        WHERE u.cod_utilizator = userId

        GROUP BY c.cod_comanda

    ) LOOP

        total_cost_record.cod_comanda :=

order_rec.cod_comanda;

        total_cost_record.total_cost := order_rec.total_cost;

        total_cost_array.extend;

        total_cost_array(total_cost_array.last) :=

total_cost_record;

    END LOOP;

    RETURN total_cost_array;

END CalculateTotalCost;

```

```

FUNCTION AverageRatingForUser(userId IN
Utilizator.cod_utilizator%TYPE)

    RETURN AverageRatingArray IS

    average_rating_array AverageRatingArray :=
AverageRatingArray();

    average_rating_record AverageRatingRecord;

    average_rating NUMBER;

BEGIN

    FOR product_rec IN (

        SELECT p.cod_produș, p.nume

        FROM Produș p

        JOIN Review r ON p.cod_produș = r.cod_produș

        JOIN Utilizator u ON r.cod_utilizator = u.cod_utilizator

        WHERE u.cod_utilizator = userId

        GROUP BY p.cod_produș, p.nume

    ) LOOP

        SELECT AVG(r.rating) INTO average_rating

        FROM Review r

        JOIN Produș p ON r.cod_produș = p.cod_produș

        JOIN Utilizator u ON r.cod_utilizator = u.cod_utilizator

        WHERE u.cod_utilizator = userId AND p.cod_produș =
product_rec.cod_produș;

        average_rating_record.product_id :=
product_rec.cod_produș;

        average_rating_record.product_name := product_rec.nume;

        average_rating_record.average_rating := average_rating;

```

```

        average_rating_array.extend;

        average_rating_array(average_rating_array.last) :=
average_rating_record;

    END LOOP;


    RETURN average_rating_array;

END AverageRatingForUser;

END Ex14_Package;

/

```


Live SQL
Feedback
Help
mihai-octavian.ocnaru@s.unibuc.ro
🌙

SQL Worksheet
Clear
Find
Actions
Save
Run

```

120      JOIN Utilizator u ON r.cod_utilizator = u.cod_utilizator
121      WHERE u.cod_utilizator = userId AND p.cod_produus = product_rec.cod_produus;
122      average_rating_record.product_id := product_rec.cod_produus;
123      average_rating_record.product_name := product_rec.numc;
124      average_rating_record.average_rating := average_rating;
125      average_rating_array.extend;
126      average_rating_array(average_rating_array.last) := average_rating_record;
127  END LOOP;
128  RETURN average_rating_array;
129  END AverageRatingForUser;
130
131  END Ex14_Package;
132  /

```

Package created.

 Package Body created.

2024 Oracle · Live SQL 23.4.2, running Oracle Database 19c EE Extreme Perf - 19.17.0.0.0 · Database Documentation · Ask Tom · Dev Gym

 Built with ❤️ using Oracle APEX · Privacy · Terms of Use

In continuare vom testa functionalitatea pentru un utilizator dat:

DECLARE

userId Utilizator.cod_utilizator%TYPE := 3;

total_cost_array Ex14_Package.TotalOrderCostArray;

average_rating_array Ex14_Package.AverageRatingArray;

BEGIN

DBMS_OUTPUT.PUT_LINE('Afisare detalii comenzi si review-uri
pentru utilizatorul cu ID-ul ' || userId);

Ex14_Package.DisplayOrderDetails(userId);

DBMS_OUTPUT.PUT_LINE('-----');

DBMS_OUTPUT.PUT_LINE('Afisare rating pentru fiecare produs
evaluat de utilizatorul cu ID-ul ' || userId);

Ex14_Package.DisplayReviewData(userId);

DBMS_OUTPUT.PUT_LINE('----- ');

DBMS_OUTPUT.PUT_LINE('Afisare cost total pentru fiecare
comanda a utilizatorului cu ID-ul ' || userId);

total_cost_array := Ex14_Package.CalculateTotalCost(userId);

FOR i IN 1..total_cost_array.count LOOP

DBMS_OUTPUT.PUT_LINE('ID comanda: ' ||
total_cost_array(i).cod_comanda || ' | cost total: ' ||
total_cost_array(i).total_cost);

END LOOP;

DBMS_OUTPUT.PUT_LINE('----- ');

```
DBMS_OUTPUT.PUT_LINE('Afisare rating mediu pentru fiecare  
produs evaluat de utilizatorul cu ID-ul ' || userId);
```

```
average_rating_array :=  
Ex14_Package.AverageRatingForUser(userId);
```

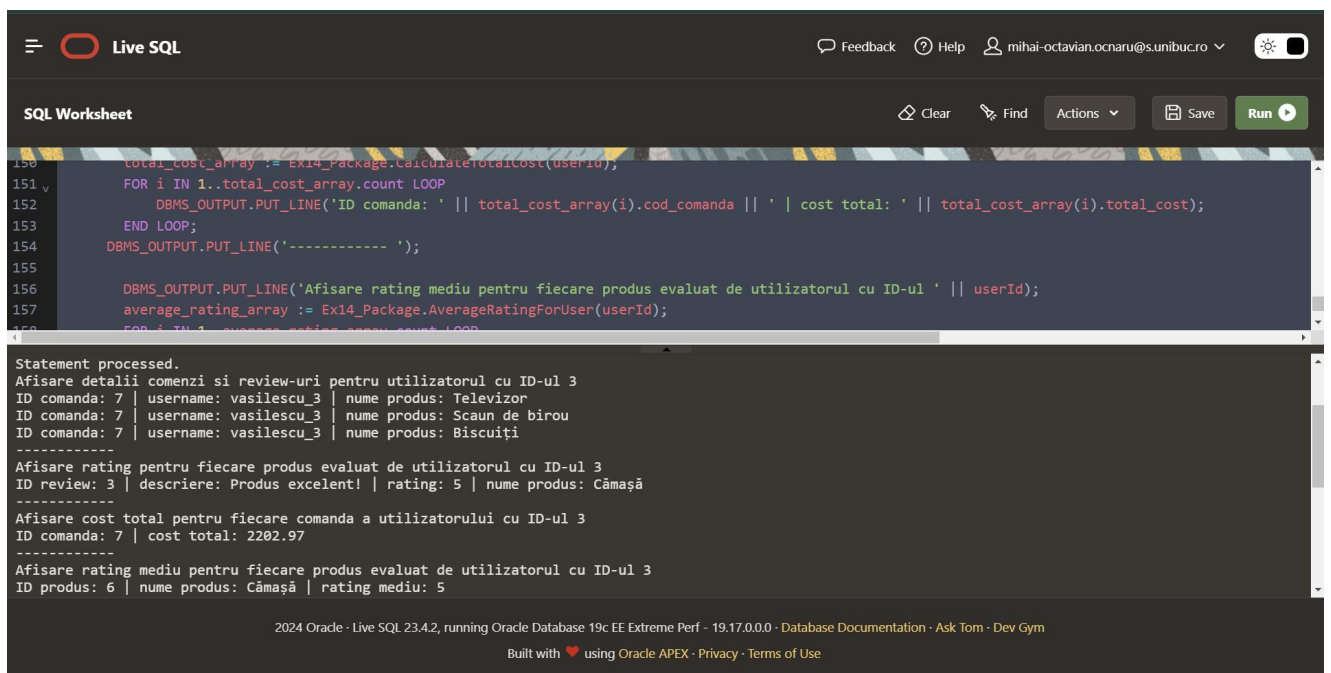
```
FOR i IN 1..average_rating_array.count LOOP
```

```
DBMS_OUTPUT.PUT_LINE('ID produs: ' ||  
average_rating_array(i).product_id || ' | nume produs: ' ||  
average_rating_array(i).product_name || ' | rating mediu: ' ||  
average_rating_array(i).average_rating);
```

```
END LOOP;
```

```
END;
```

```
/
```



The screenshot shows the Live SQL interface with a SQL worksheet containing the following code:

```
150 total_cost_array := Ex14_Package.CalculateTotalCost(userId);  
151 FOR i IN 1..total_cost_array.count LOOP  
152     DBMS_OUTPUT.PUT_LINE('ID comanda: ' || total_cost_array(i).cod_comanda || ' | cost total: ' || total_cost_array(i).total_cost);  
153 END LOOP;  
154 DBMS_OUTPUT.PUT_LINE('----- ');  
155  
156 DBMS_OUTPUT.PUT_LINE('Afisare rating mediu pentru fiecare produs evaluat de utilizatorul cu ID-ul ' || userId);  
157 average_rating_array := Ex14_Package.AverageRatingForUser(userId);  
158 FOR i IN 1..average_rating_array.count LOOP
```

The output of the statement is as follows:

```
Statement processed.  
Afisare detalii comenzi si review-uri pentru utilizatorul cu ID-ul 3  
ID comanda: 7 | username: vasilescu_3 | nume produs: Televizor  
ID comanda: 7 | username: vasilescu_3 | nume produs: Scaun de birou  
ID comanda: 7 | username: vasilescu_3 | nume produs: Biscuiti  
-----  
Afisare rating pentru fiecare produs evaluat de utilizatorul cu ID-ul 3  
ID review: 3 | descriere: Produs excelent! | rating: 5 | nume produs: Cămașă  
-----  
Afisare cost total pentru fiecare comanda a utilizatorului cu ID-ul 3  
ID comanda: 7 | cost total: 2282.97  
-----  
Afisare rating mediu pentru fiecare produs evaluat de utilizatorul cu ID-ul 3  
ID produs: 6 | nume produs: Cămașă | rating mediu: 5
```

At the bottom of the interface, it states: "2024 Oracle - Live SQL 23.4.2, running Oracle Database 19c EE Extreme Perf - 19.17.0.0.0 - Database Documentation - Ask Tom - Dev Gym" and "Built with ❤️ using Oracle APEX - Privacy - Terms of Use".