1. Proiectaţi o structură de bază de date relaţionlă pentru gestiunea vânzărilor şi a ratelor la un magazin.

2. Conectaţi-vă la SGBD Orcale

3. Creaţi structurile bazei de date.

CREATE TABLE Produse (

ID\_Produs INT PRIMARY KEY,

Nume VARCHAR(100),

Descriere VARCHAR(255),

Pret DECIMAL(10,2),

Stoc INT

);

CREATE TABLE Clienti (

ID\_Client INT PRIMARY KEY,

Nume VARCHAR(100),

Prenume VARCHAR(100),

Adresa VARCHAR(255),

Email VARCHAR(100)

);

CREATE TABLE Vanzari (

ID\_Vanzare INT PRIMARY KEY,

ID\_Client INT,

ID\_Transportator INT,

ID\_Produs INT,

Data\_Vanzare DATE,

Cantitate INT, -- Adăugăm coloana Cantitate

Total DECIMAL(10,2)

);

FOREIGN KEY (ID\_Client) REFERENCES Clienti(ID\_Client),

FOREIGN KEY (ID\_Transportator) REFERENCES Transportatori(ID\_Transportator) -- Relația străină pentru ID\_Transportator corectată

);

ALTER TABLE Vanzari

ADD CONSTRAINT fk\_vanzari\_produse FOREIGN KEY (ID\_Produs) REFERENCES Produse(ID\_Produs);

CREATE TABLE Rate (

ID\_Rata INT PRIMARY KEY,

ID\_Vanzare INT,

Data\_Rata DATE,

Suma DECIMAL(10,2),

Status VARCHAR(50),

FOREIGN KEY (ID\_Vanzare) REFERENCES Vanzari(ID\_Vanzare)

);

CREATE TABLE Transportator (

ID\_Transportator INT PRIMARY KEY,

Nume VARCHAR(100),

Adresa VARCHAR(255),

Email VARCHAR(100),

Numar\_Telefon VARCHAR(20)

);

4. Formulaţi restricţiile de integritate asociate bazei de date, pe care le consideraţi utile.

Am introdus restrictiile de integritate din momentul in care am creat tabelele, daca omiteam acest lucru s-ar fi putut crea in acest mod:

ALTER TABLE Vanzari ADD CONSTRAINT fk\_client FOREIGN KEY (ID\_Client) REFERENCES Clienti(ID\_Client);

ALTER TABLE Rate ADD CONSTRAINT fk\_vanzare FOREIGN KEY (ID\_Vanzare) REFERENCES Vanzari(ID\_Vanzare);

5. Definiţi indecşi asociaţi bazei de date, pe care îi consideraţi utili.

CREATE INDEX idx\_nume\_produs ON Produse(Nume);

CREATE INDEX idx\_nume\_client ON Clienti(Nume);

6. Definiţi secvenţe asociate bazei de date, pe care le consideraţi utile.

CREATE SEQUENCE seq\_id\_produs START WITH 1 INCREMENT BY 1;

CREATE SEQUENCE seq\_id\_client START WITH 1 INCREMENT BY 1;

CREATE SEQUENCE seq\_id\_vanzare START WITH 1 INCREMENT BY 1;

CREATE SEQUENCE seq\_id\_rata START WITH 1 INCREMENT BY 1;

CREATE SEQUENCE seq\_id\_transportator START WITH 1 INCREMENT BY 1;

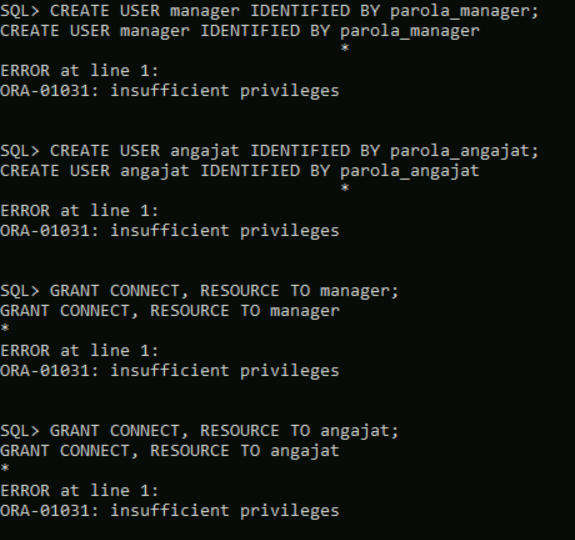
7. Definiţi tipuri de utilizatori (roluri) pentru această bază de date.

CREATE USER manager IDENTIFIED BY parola\_manager;

CREATE USER angajat IDENTIFIED BY parola\_angajat;

GRANT CONNECT, RESOURCE TO manager;

GRANT CONNECT, RESOURCE TO angajat;



8. Creaţi un trigger pentru recalcularea ratelor în momentul unei plăţi.

CREATE OR REPLACE TRIGGER recalculate\_rate

BEFORE UPDATE ON Rate

FOR EACH ROW

BEGIN

IF :NEW.Suma < 0 THEN -- Suma este negativă (probabil o rambursare)

:NEW.Status := 'Rambursare';

ELSIF :NEW.Suma = 0 THEN

:NEW.Status := 'Platit';

ELSE

:NEW.Status := 'Partial Platit';

END IF;

END;

/

9. Scrieţi o funcţie care să…

CREATE OR REPLACE FUNCTION total\_platit (id\_client\_in IN NUMBER)

RETURN NUMBER

IS

total\_plati NUMBER; -- Variabila pentru stocarea sumei totale plătite de client

BEGIN

-- Calculăm suma totală plătită de către client

SELECT NVL(SUM(Total), 0) INTO total\_plati

FROM Vanzari

WHERE ID\_Client = id\_client\_in;

-- Returnăm suma totală plătită de către client

RETURN total\_plati;

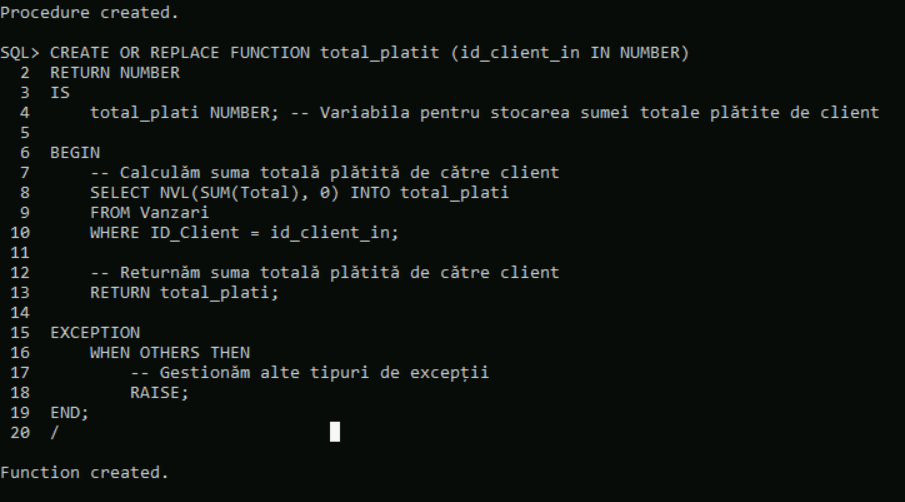
EXCEPTION

WHEN OTHERS THEN

-- Gestionăm alte tipuri de excepții

RAISE;

END;



10. Scrieţi o procedură care să …

Procedură, numită add\_sale, este concepută pentru a adăuga o nouă înregistrare de vânzare în tabela Vanzari.

CREATE OR REPLACE PROCEDURE add\_sale (

p\_id\_vanzare IN Vanzari.ID\_Vanzare%TYPE,

p\_id\_client IN Vanzari.ID\_Client%TYPE,

p\_id\_transportator IN Vanzari.ID\_Transportator%TYPE,

p\_data\_vanz IN Vanzari.Data\_Vanzare%TYPE,

p\_cantitate IN Vanzari.Cantitate%TYPE,

p\_total IN Vanzari.Total%TYPE

)

IS

BEGIN

-- Inserarea unei noi înregistrări de vânzare în tabelul Vanzari

INSERT INTO Vanzari (ID\_Vanzare, ID\_Client, ID\_Transportator, Data\_Vanzare, Cantitate, Total)

VALUES (p\_id\_vanzare, p\_id\_client, p\_id\_transportator, p\_data\_vanz, p\_cantitate, p\_total);

-- Logică suplimentară poate fi adăugată aici, dacă este necesară

-- Confirmarea tranzacției

COMMIT;

EXCEPTION

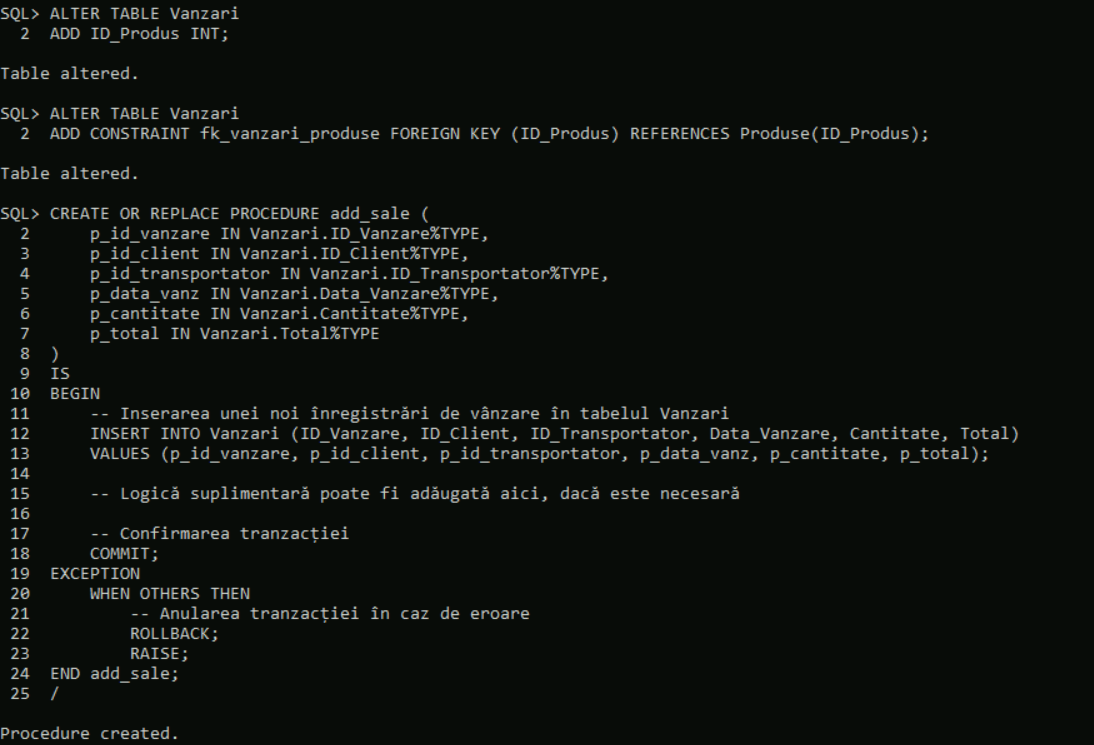
WHEN OTHERS THEN

-- Anularea tranzacției în caz de eroare

ROLLBACK;

RAISE;

END add\_sale;

/

11. Creaţi un pachet cu subprogramele de mai sus.

CREATE OR REPLACE PACKAGE pachet\_vanzari AS

FUNCTION total\_platit(id\_client\_in IN NUMBER) RETURN NUMBER;

PROCEDURE add\_sale(

p\_id\_vanzare IN Vanzari.ID\_Vanzare%TYPE,

p\_id\_client IN Vanzari.ID\_Client%TYPE,

p\_id\_transportator IN Vanzari.ID\_Transportator%TYPE,

p\_data\_vanz IN Vanzari.Data\_Vanzare%TYPE,

p\_cantitate IN Vanzari.Cantitate%TYPE,

p\_total IN Vanzari.Total%TYPE

);

END pachet\_vanzari;

/

CREATE OR REPLACE PACKAGE BODY pachet\_vanzari AS

FUNCTION total\_platit(id\_client\_in IN NUMBER) RETURN NUMBER IS

total\_plati NUMBER;

BEGIN

-- Calculăm suma totală plătită de către client

SELECT NVL(SUM(Total), 0) INTO total\_plati

FROM Vanzari

WHERE ID\_Client = id\_client\_in;

-- Returnăm suma totală plătită de către client

RETURN total\_plati;

EXCEPTION

WHEN OTHERS THEN

-- Gestionăm alte tipuri de excepții

RAISE;

END total\_platit;

PROCEDURE add\_sale(

p\_id\_vanzare IN Vanzari.ID\_Vanzare%TYPE,

p\_id\_client IN Vanzari.ID\_Client%TYPE,

p\_id\_transportator IN Vanzari.ID\_Transportator%TYPE,

p\_data\_vanz IN Vanzari.Data\_Vanzare%TYPE,

p\_cantitate IN Vanzari.Cantitate%TYPE,

p\_total IN Vanzari.Total%TYPE

) IS

BEGIN

-- Inserarea unei noi înregistrări de vânzare în tabelul Vanzari

INSERT INTO Vanzari (ID\_Vanzare, ID\_Client, ID\_Transportator, Data\_Vanzare, Cantitate, Total)

VALUES (p\_id\_vanzare, p\_id\_client, p\_id\_transportator, p\_data\_vanz, p\_cantitate, p\_total);

-- Confirmarea tranzacției

COMMIT;

EXCEPTION

WHEN OTHERS THEN

-- Anularea tranzacției în caz de eroare

ROLLBACK;

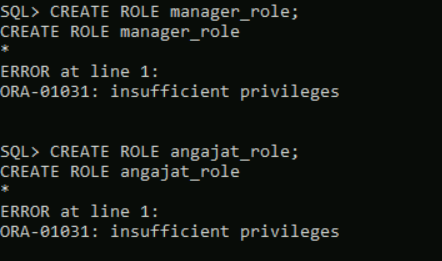
RAISE;

END add\_sale;

END pachet\_vanzari;

/

12. Creaţi roluri, precum şi utilizatori din fiecare categorie şi creaţi schema fiecăruia. (nu uitaţi să acordaţi drepturi, inclusiv pentru execuţia subprogramelor pachetului, utilizatorilor responsabili).



13. Conectaţi-vă pe rând drept fiecare din utilizatorii creaţi şi verificaţi buna funcţionare a obiectelor din schema lui şi a elementelor de aplicaţie de care dispune. Indicaţie: Folosiţi fişiere de comenzi SQL\*Plus pentru toate operaţiile de implementare a sistemului de baze de date.