Implementarea unei Baze de Date Relaționale pentru Gestionarea Eficientă a Produselor, Clienților și Comenzilor într-un Mediu Comercial

Costinean Sebastian in echipa cu Herman Darius Razvan

grupa:30126

1.Proiectul implică implementarea unei baze de date relaționale pentru gestionarea produselor, clienților și comenzilor într-un mediu

comercial.

2.Structura bazei de date cuprinde tabele precum Produs, Clienti și Comenzi, care sunt interconectate prin chei străine pentru a asigura

integritatea datelor.

3.Au fost create proceduri stocate și declanșatoare pentru a gestiona operațiuni precum adăugarea de clienți,

actualizarea stocului produselor și calculul valorilor de comandă.

4.Prin intermediul declanșatoarelor, stocul produselor este actualizat automat după fiecare comandă plasată,

asigurând astfel sincronizarea corectă a stocurilor disponibile.

5.De asemenea, au fost create funcții care facilitează calculul valorilor totale ale comenzilor

pentru un anumit client sau pentru o anumită comandă specificată.

```
CREATE TABLE Produs (
  ID int PRIMARY KEY,
```

```
  Nume varchar(50),
  Pret float,
  Stoc int
);

CREATE TABLE TipProdus (
  ID int PRIMARY KEY,
  Tip varchar(20)
);

CREATE TABLE Calculator (
  ID int PRIMARY KEY,
  Procesor varchar(50),
  RAM int,
  CapacitateHDD int,
  FOREIGN KEY (ID) REFERENCES Produs(ID)
);

CREATE TABLE Laptop (
  ID int PRIMARY KEY,
  Procesor varchar(50),
  RAM int,
  CapacitateHDD int,
  DimensiuneEcran float,
  FOREIGN KEY (ID) REFERENCES Produs(ID)
);
```

```
CREATE TABLE Imprimanta (
  ID int PRIMARY KEY,
  TipImprimanta varchar(50),
  Culoare bit,
  FOREIGN KEY (ID) REFERENCES Produs(ID)
);
CREATE TABLE Clienti (
  ID int PRIMARY KEY,
  Nume varchar(50),
  Prenume varchar(50),
  Email varchar(100),
  Telefon varchar(20)
);
-- Adăugare înregistrări în tabela "Produs"
INSERT INTO Produs (ID, Nume, Pret, Stoc) VALUES
(1, 'Calculator 1', 1500, 10),
(2, 'Laptop 1', 2000, 5),
(3, 'Imprimanta 1', 300, 20),
(4, 'Calculator 2', 1800, 8),
(5, 'Laptop 2', 2200, 3);

-- Adăugare înregistrări în tabela "TipProdus"
INSERT INTO TipProdus (ID, Tip) VALUES
(1, 'Calculator'),
(2, 'Laptop'),
```

```
(3, 'Imprimanta');

-- Adăugare înregistrări în tabela "Calculator"
INSERT INTO Calculator (ID, Procesor, RAM, CapacitateHDD) VALUES
(1, 'Intel Core i5', 8, 500),
(4, 'AMD Ryzen 7', 16, 1000);

-- Adăugare înregistrări în tabela "Laptop"
INSERT INTO Laptop (ID, Procesor, RAM, CapacitateHDD, DimensiuneEcran) VALUES
(2, 'Intel Core i7', 16, 1000, 15.6),
(5, 'AMD Ryzen 5', 8, 512, 14);

-- Adăugare înregistrări în tabela "Imprimanta"
INSERT INTO Imprimanta (ID, TipImprimanta, Culoare) VALUES
(3, 'Laser', 1);

CREATE PROCEDURE AdaugareClient (
  @ID int,
  @Nume varchar(50),
  @Prenume varchar(50),
  @Email varchar(100),
  @Telefon varchar(20)
)
AS
BEGIN
  INSERT INTO Clienti (ID,Nume, Prenume, Email, Telefon)
```

```
  VALUES (@ID,@Nume, @Prenume, @Email, @Telefon);
END

CREATE PROCEDURE ActualizareStoc (
  @ID int,
  @Stoc int
)
AS
BEGIN
  UPDATE Produs
  SET Stoc = @Stoc
  WHERE ID = @ID;
END

CREATE TRIGGER ActualizareStocDupaComanda
AFTER INSERT ON Comenzi
FOR EACH ROW
BEGIN
  DECLARE v_Stoc INT;
  SELECT Stoc INTO v_Stoc FROM Produs WHERE ID = NEW.ProdusID;
  UPDATE Produs SET Stoc = v_Stoc - NEW.Cantitate WHERE ID = NEW.ProdusID;
END

CREATE TRIGGER ActualizarePretMediu
ON Produs
AFTER UPDATE
```

```
AS
BEGIN
  UPDATE p
  SET PretMediu = (SELECT AVG(Pret) FROM Produs WHERE ID IN (SELECT ID FROM
inserted))
  FROM Produs p
  INNER JOIN inserted i ON p.ID = i.ID;
END


DROP TRIGGER IF EXISTS ActualizarePretMediu;



ALTER TABLE Produs ADD PretMediu float;







CREATE FUNCTION CalculValoareComanda (
  @ComandaID int
)
RETURNS float
AS
BEGIN
  DECLARE @Valoare float;
  SELECT @Valoare = SUM(Produs.Pret * Comenzi.Cantitate)
```

```
 FROM Comenzi
 INNER JOIN Produs ON Comenzi.ProdusID = Produs.ID
 WHERE Comenzi.ComandaID = @ComandaID;
 RETURN @Valoare;
END



CREATE FUNCTION CalculValoareTotalaComenzi (
 @ClientID int
)
RETURNS float
AS
BEGIN
 DECLARE @ValoareTotala float;
 SELECT @ValoareTotala = SUM(Comenzi.Cantitate * Produs.Pret)
 FROM Comenzi
 INNER JOIN Produs ON Comenzi.ProdusID = Produs.ID
 WHERE Comenzi.ClientID = @ClientID;
 RETURN @ValoareTotala;
END

ALTER TABLE Produs
ADD TipID int,
FOREIGN KEY (TipID) REFERENCES TipProdus(ID);
```

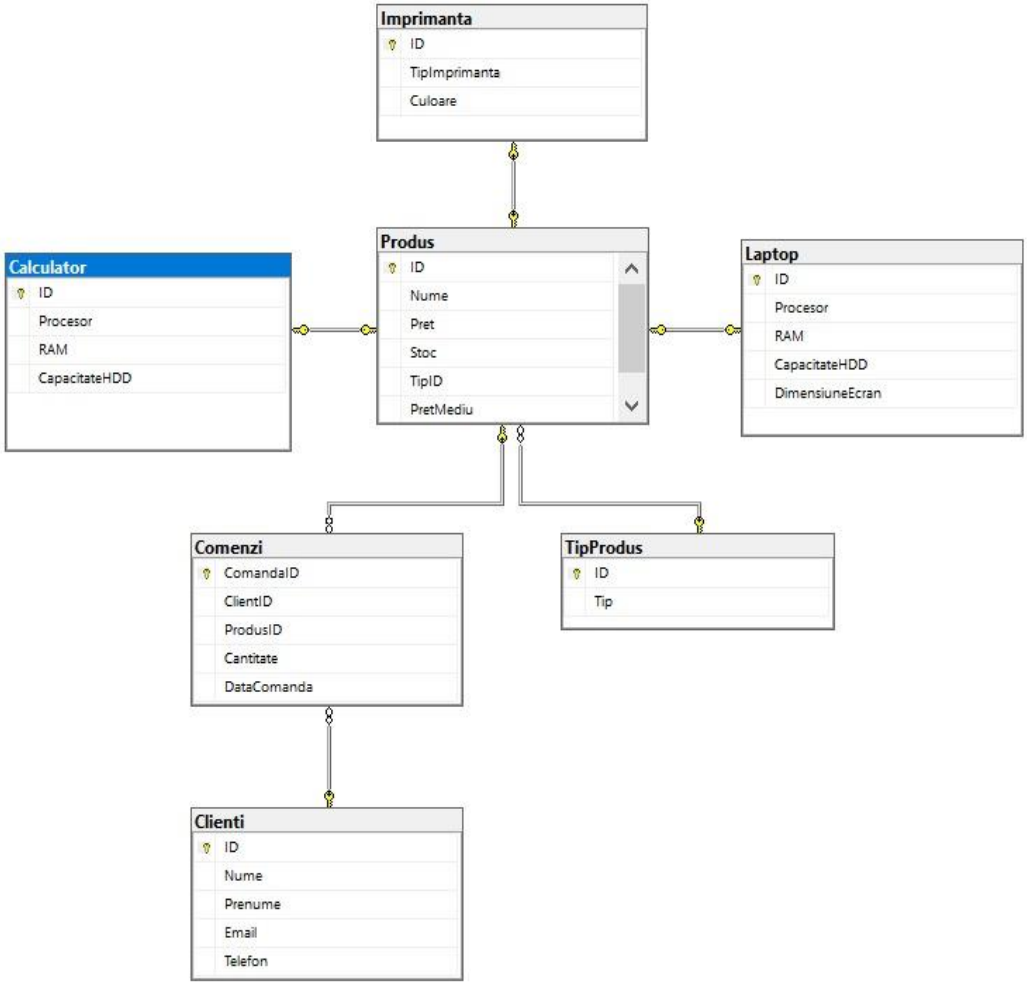
```
SELECT Produs.Nume, TipProdus.Tip
FROM Produs
JOIN TipProdus ON Produs.TipID = TipProdus.ID;
SELECT Produs.Nume, TipProdus.Tip
FROM Produs
LEFT JOIN TipProdus ON Produs.TipID = TipProdus.ID;


ALTER TABLE Comenzi
ADD FOREIGN KEY (ClientID) REFERENCES Clienti(ID);



CREATE TABLE Comenzi (
  ComandaID int PRIMARY KEY,
  ClientID int,
  ProdusID int,
  Cantitate int,
  DataComanda date,

  FOREIGN KEY (ProdusID) REFERENCES Produs(ID)
);



select * from Clienti
```

```
INSERT INTO Comenzi (ComandaID, ClientID, ProdusID, Cantitate, DataComanda)
VALUES (1, 1, 1, 2, GETDATE());

select * from Comenzi
SELECT * FROM Clienti;
INSERT INTO Clienti (ID, Nume, Prenume, Email, Telefon)
VALUES (1, 'NumeClient', 'PrenumeClient', 'email@example.com', '123456789');
SELECT dbo.CalculValoareTotalaComenzi(1) AS ValoareTotalaComenzi;


EXEC AdaugareClient 'John', 'Doe', 'john.doe@example.com', '1234567890'
EXEC ActualizareStoc @ID = 1, @Stoc = 8;
INSERT INTO Comenzi (ComandaID, ClientID, ProdusID, Cantitate, DataComanda)
VALUES (2, 1, 1, 2, GETDATE());


INSERT INTO Clienti (ID, Nume, Prenume, Email, Telefon)
VALUES (2, 'NumeClient2', 'PrenumeClient2', 'email2@example.com', '987654321');
INSERT INTO Comenzi (ComandaID, ClientID, ProdusID, Cantitate, DataComanda)
VALUES (3, 2, 1, 3, GETDATE());
EXEC ActualizareStoc @ID = 1, @Stoc = 8;
SELECT dbo.CalculValoareComanda(2) AS ValoareComanda;
SELECT TABLE_NAME
FROM INFORMATION_SCHEMA.TABLES
WHERE TABLE_TYPE = 'BASE TABLE' AND TABLE_SCHEMA='dbo';
SELECT * FROM Calculator;
```

```
SELECT * FROM Clienti;
SELECT * FROM Comenzi;
SELECT * FROM Laptop;
SELECT * FROM Imprimanta;
SELECT * FROM Produs;
SELECT * FROM TipProdus;

INSERT INTO Comenzi (ComandaID, ClientID, ProdusID, Cantitate, DataComanda)
VALUES (1, 1, 1, 2, GETDATE());



SELECT Stoc
FROM Produs
WHERE ID = 4;

-- Call the function and retrieve the result
SELECT dbo.CalculValoareComanda(1) AS ValoareComanda;

INSERT INTO Clienti (ID, Nume, Prenume, Email, Telefon)
VALUES (2, 'NumeClient2', 'PrenumeClient2', 'email2@example.com', '987654321');
EXEC AdaugareClient  3,'John', 'Doe', 'john.doe@example.com', '1234567890';



ALTER PROCEDURE AdaugareClient
  @ID int,
  @Nume varchar(50),
```

```
 @Prenume varchar(50),
 @Email varchar(100),
 @Telefon varchar(20)
AS
BEGIN
 INSERT INTO Clienti (ID, Nume, Prenume, Email, Telefon)
 VALUES (@ID, @Nume, @Prenume, @Email, @Telefon);
END




CREATE FUNCTION CalculateTotalSales
(
 @StartDate date,
 @EndDate date
)
RETURNS decimal(10,2)
AS
BEGIN
 DECLARE @TotalSales decimal(10,2);

 SELECT @TotalSales = SUM(Comenzi.Cantitate * Produs.Pret)
 FROM Comenzi
 INNER JOIN Produs ON Comenzi.ProdusID = Produs.ID
 WHERE Comenzi.DataComanda BETWEEN @StartDate AND @EndDate
```

```
  GROUP BY Produs.ID
  HAVING SUM(Comenzi.Cantitate * Produs.Pret) > 1000;

  RETURN @TotalSales;
END


CREATE TRIGGER InsteadOfDeleteClient
ON Clienti
INSTEAD OF DELETE
AS
BEGIN

  DELETE FROM Comenzi WHERE ClientID IN (SELECT ID FROM deleted);

  DELETE FROM Clienti WHERE ID IN (SELECT ID FROM deleted);

  RAISERROR('Deleting a client is not allowed.', 16, 1);
END
```


Imprimanta
ID
TipImprimanta
Culoare
Produs
ID
Nume
Pret
Stoc
TipID
PretMediu
Calculator
ID
Procesor
RAM
CapacitateHDD
Laptop
ID
Procesor
RAM
CapacitateHDD
DimensiuneEcran
Comenzi
ComandaID
ClientID
ProdusID
Cantitate
DataComanda
TipProdus
ID
Tip
Clienti
ID
Nume
Prenume
Email
Telefon