Proiect Baze de Date

Lanțul de Cinematografe Flacăra

Sd. Cap. Stanciu Mihail Sebastian

1.Descriere:

În acest proiect m-am gândit să alcătueisc o bază de date pentru un lanț de cinematografe pe care l-am numit Flacăra, după numele unui vechi cinematograf de la mine din cartier. În aceast proiect am stocate date despre angajați, salariul acestora, diferitele sucursale ale companiei, precum și o parte din clienți și furnizorii de diferite produse, de la snacks-uri și sucuri la electricitate și încălzire.

2.Diagrama Bazei de Date:

Tabele:

-Employees: tabelul în care sunt stocate date despre angajați, cum ar fi numele, numărul de telefon, data nașterii și a angajării, funcția, salariul brut și impozitul, salariul net putând fi calculat din diferența celor două, precum și câte o cheie străină către sucursala la care lcrează și adresa de domiciliu și o costrângere pentru a verifica vârsta să fie peste 18 ani

-Customers: tabelul cu date despre clienți, cum ar fi numele, statutul(adult, pensionar, elev, student, copil) care determină discount-ul în momentul cumpărării biletului, precum si o coloană care ne informează dacă este clint fidel sau client nou(coloana Regular), având o cheie străină către biletul pe care l-a cumpărat

-Addresses: tabelul cu toate adresele utilizate în celelalte tabele, conținând date despre județul, orașul, strada și numărul la care se află angajații, sucursalele și furnizorii de produse, având și o cheie străină către tabelul cu teritorii

-Branches: tabelul cu cele 5 sucursale ale companiei, conținând date despre numele cinematografului, numărul de rânduri și coloane de scaune, precum și o cheie străina către adresa cinematografului

-Expenses: acest table leagă tabelul în care sunt stocate produsele de la furnizori de sucursala la care sunt trimise, oferind astfel date despre tipul cheltuielilor(mâncare, băutură sau utilitate) și prețul pe fiecare lună al produsului respectiv cheltuit de o sucursală

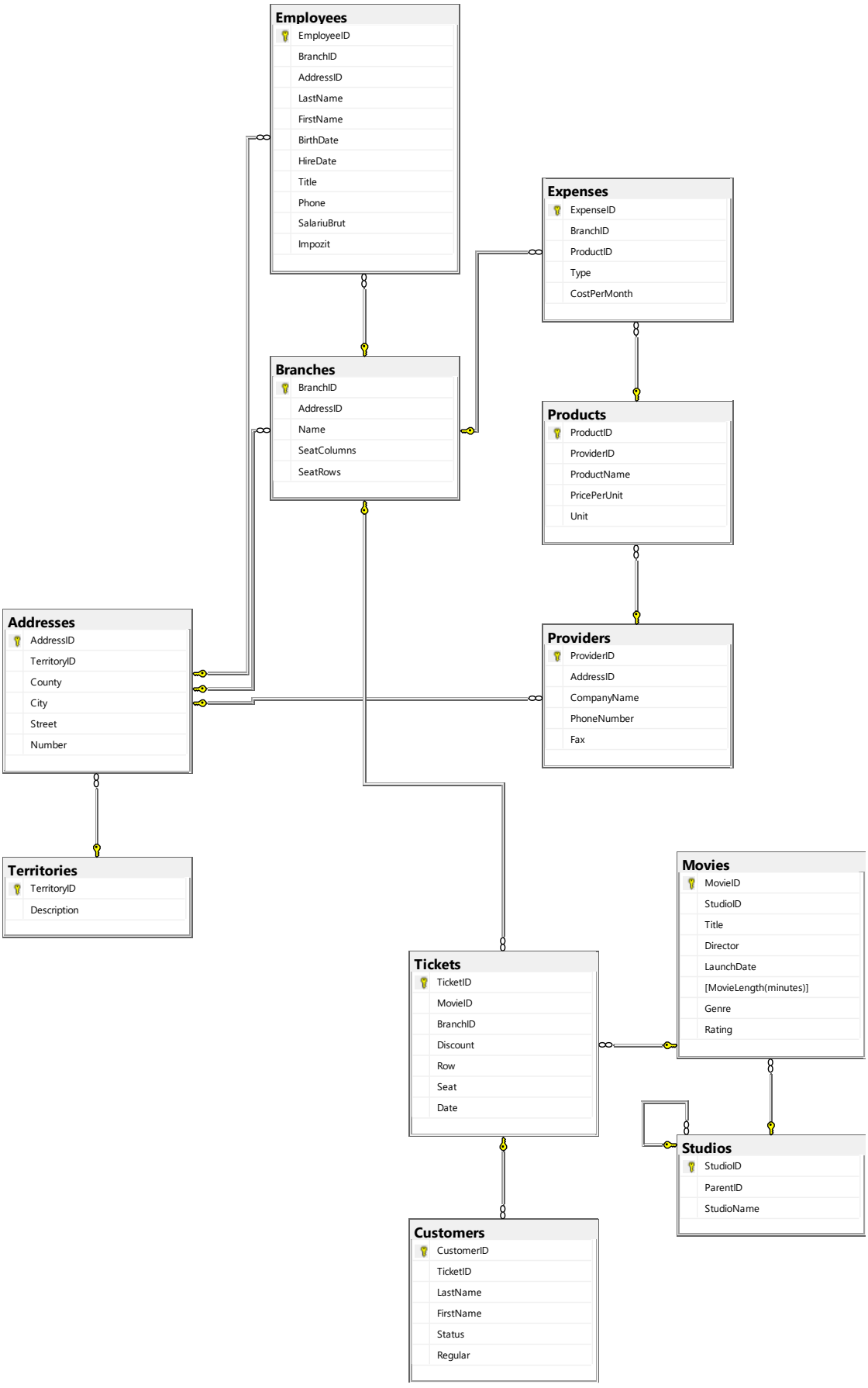
-Providers: tabelul care conține date despre furnizorii de diferite produse, spre exemplu suc, curent sau apă, având ca și coloane numele companiei, numărul de telefon și faxul, precum și o cheie străină către adresa la care se află sediul companiei

-Movies: tabelul cu filmele disponibile în arhiva cinematografelor, conținând numele filmului, directorul, data lansării, durata, genul și rating-ul acestuia, precum și o cheie străină câtre tabelul cu studouri

-Studios: tabelul cu diferitele studiouri de filme, precum Disney Studios sau Warner Bros. Pictures, inclusive studiourile cumpărate de acestea care conțin o cheie străină către studioul părinte

-Territories: tabelul cu diferetele teritorii ale țării, precum Oltenia sau Moldova, pentru o mai ușoară sortare a datelor în funcție de zona geografică la o scală mai mare

-Tickets: tabelul în care sunt stocate biletele vândute, conținând date despre discountul cu care a fost cumpărat biletul, rândul și numărul scaunul și data la care are loc filmul, precum și câte o cheie străină către sucursala de la care a fost cumpărat și filmul pentru are a fost cumpărat biletul


Employees
EmployeeID
BranchID
AddressID
LastName
FirstName
BirthDate
HireDate
Title
Phone
SalariuBrut
Impozit
Expenses
ExpenseID
BranchID
ProductID
Type
CostPerMonth
Branches
BranchID
AddressID
Name
SeatColumns
SeatRows
Products
ProductID
ProviderID
ProductName
PricePerUnit
Unit
Addresses
AddressID
TerritoryID
County
City
Street
Number
Providers
ProviderID
AddressID
CompanyName
PhoneNumber
Fax
Territories
TerritoryID
Description
Movies
MovieID
StudioID
Title
Director
LaunchDate
[MovieLength(minutes)]
Genre
Rating
Tickets
TicketID
MovieID
BranchID
Discount
Row
Seat
Date
Studios
StudioID
ParentID
StudioName
Customers
CustomerID
TicketID
LastName
FirstName
Status
Regular

Crearea bazei de date:

```
IF DB_ID('Cinematograf') IS NOT NULL
	DROP DATABASE Cinematograf
CREATE DATABASE Cinematograf
ON PRIMARY
(
	Name = MasterFile,
	FileName = 'E:\Cursuri\Anul 2 sem 2\Baze de Date\Proiect\MasterFile.mdf',
	size = 10MB, -- KB, Mb, GB, TB
	maxsize = unlimited,
	filegrowth = 1GB
),
(
	Name = DataFile1,
	FileName = 'E:\Cursuri\Anul 2 sem 2\Baze de Date\Proiect\DataFile1.ndf',
	size = 10MB, -- KB, Mb, GB, TB
	maxsize = unlimited,
	filegrowth = 1GB
),
(
	Name = DataFile2,
	FileName = 'E:\Cursuri\Anul 2 sem 2\Baze de Date\Proiect\DataFile2.ndf',
	size = 10MB, -- KB, Mb, GB, TB
	maxsize = unlimited,
	filegrowth = 1GB
)
LOG ON
(
	Name = LogFile1,
	FileName = 'E:\Cursuri\Anul 2 sem 2\Baze de Date\Proiect\LogFile1.ldf',
	size = 10MB, -- KB, Mb, GB, TB
	maxsize = unlimited,
	filegrowth = 1024MB
),
(
	Name = LogFile2,
	FileName = 'E:\Cursuri\Anul 2 sem 2\Baze de Date\Proiect\LogFile2.ldf',
	size = 10MB, -- KB, Mb, GB, TB
	maxsize = unlimited,
	filegrowth = 1024MB
)
```

Crearea tabelelor:

```
IF OBJECT_ID('Employees', 'U') IS NOT NULL
	DROP TABLE Employees;
GO
CREATE TABLE Employees
(
	EmployeeID int PRIMARY KEY IDENTITY(1,1),
	BranchID int NOT NULL,
	AddressID int NOT NULL,
	LastName varchar(30) NOT NULL,
	FirstName varchar(30) NOT NULL,
```

```
	BirthDate date CHECK (DATEDIFF(year, BirthDate, GETDATE()) >= 18),
	HireDate date,
	Title nvarchar(30),
	Phone nvarchar(15),
	SalariuNet int,
	SalariuBrut int

)

IF OBJECT_ID('Customers', 'U') IS NOT NULL
	DROP TABLE Customers;
GO
CREATE TABLE Customers
(
	CustomerID int PRIMARY KEY IDENTITY(1,1),
	TicketID int NOT NULL,
	LastName varchar(30),
	FirstName varchar(30),
	Status varchar(30),
	Regular bit
)

IF OBJECT_ID('Addresses', 'U') IS NOT NULL
	DROP TABLE Addresses;
GO
CREATE TABLE Addresses
(
	AddressID int PRIMARY KEY IDENTITY(1,1),
	TerritoryID int NOT NULL,
	County nvarchar(30),
	City nvarchar(30),
	Street nvarchar(30),
	Number int
)

IF OBJECT_ID('Branches', 'U') IS NOT NULL
	DROP TABLE Branches;
GO
CREATE TABLE Branches
(
	BranchID int PRIMARY KEY IDENTITY(1,1),
	AddressID int NOT NULL,
	Name varchar(30),
	SeatRows int,
	SeatColumns int
)

IF OBJECT_ID('Expenses', 'U') IS NOT NULL
	DROP TABLE Expenses;
GO
CREATE TABLE Expenses
(
	ExpenseID int PRIMARY KEY IDENTITY(1,1),
	BranchID int,
	ProductID int,
	Type varchar(30),
	CostPerMonth money
)
```

```sql
IF OBJECT_ID('Products', 'U') IS NOT NULL
	DROP TABLE Products;
GO
CREATE TABLE Products
(
	ProductID int PRIMARY KEY IDENTITY(1,1),
	ExpenseID int,
	ProviderID int,
	ProductName varchar(30),
	PricePerUnit money
)

IF OBJECT_ID('Movies', 'U') IS NOT NULL
	DROP TABLE Movies;
GO
CREATE TABLE Movies
(
	MovieID int PRIMARY KEY IDENTITY(1,1),
	StudioID int NOT NULL,
	Title varchar(30),
	Director varchar(30),
	LaunchDate date,
	MovieLength time(7),
	Genre varchar(30),
	Rating int
)

IF OBJECT_ID('Providers', 'U') IS NOT NULL
	DROP TABLE Providers;
GO
CREATE TABLE Providers
(
	ProviderID int PRIMARY KEY IDENTITY(1,1),
	AddressID int NOT NULL,
	CompanyName varchar(30),
	PhoneNumber varchar(15),
	Fax varchar(20)
)

IF OBJECT_ID('Studios', 'U') IS NOT NULL
	DROP TABLE Studios;
GO
CREATE TABLE Studios
(
	StudioID int PRIMARY KEY IDENTITY(1,1),
	ParentID int,
	StudioName varchar(30)
)

IF OBJECT_ID('Territories', 'U') IS NOT NULL
	DROP TABLE Territories;
GO
CREATE TABLE Territories
(
	TerritoryID int PRIMARY KEY IDENTITY(1,1),
	Description varchar(30)
)
```

```
IF OBJECT_ID('Tickets', 'U') IS NOT NULL
	DROP TABLE Tickets;
GO
CREATE TABLE Tickets
(
	TicketID int PRIMARY KEY IDENTITY(1,1),
	MovieID int,
	BranchID int,
	Discount int,
	TicketPrice int,
	Row int,
	Seat int,
	Date date
)
```

Adăugarea cheilor străine:

```
ALTER TABLE Addresses
	ADD CONSTRAINT FK_AddressTerritory
	FOREIGN KEY (TerritoryID) REFERENCES Territories(TerritoryID);

ALTER TABLE BranchID
	ADD CONSTRAINT FK_BranchAddress
	FOREIGN KEY (AddressID) REFERENCES Addresses(AddressID);

ALTER TABLE Customers
	ADD CONSTRAINT FK_CustomerTicket
	FOREIGN KEY (TicketID) REFERENCES Tickets(TicketID);

ALTER TABLE Employees
	ADD CONSTRAINT FK_EmployeeBranch
	FOREIGN KEY (BranchID) REFERENCES Branches(BranchID),
	CONSTRAINT FK_EmployeeAddress
	FOREIGN KEY (AddressID) REFERENCES Addresses(AddressID);

ALTER TABLE Expenses
	ADD CONSTRAINT FK_ExpenseBranch
	FOREIGN KEY (BranchID) REFERENCES Branches(BranchID),
	CONSTRAINT FK_ExpenseProducts
	FOREIGN KEY (ProductID) REFERENCES Products(ProductID);

ALTER TABLE Movies
	ADD CONSTRAINT FK_MovieStudio
	FOREIGN KEY (StudioID) REFERENCES Studios(StudioID);

ALTER TABLE Products
	ADD CONSTRAINT FK_ProductProvider
	FOREIGN KEY (ProviderID) REFERENCES Providers(ProviderID);

ALTER TABLE Providers
	ADD CONSTRAINT FK_ProviderAddress
	FOREIGN KEY (AddressID) REFERENCES Addresses(AddressID);

ALTER TABLE Studios
	ADD CONSTRAINT FK_StudioParent
	FOREIGN KEY (ParentID) REFERENCES Studios(StudioID);
```

```
ALTER TABLE Tickets
	ADD CONSTRAINT FK_TicketMovie
	FOREIGN KEY (MovieID) REFERENCES Movies(MovieID),
	CONSTRAINT FK_TicketBranch
	FOREIGN KEY (BranchID) REFERENCES Branches(BranchID);
```

View-uri:

1.Studiouri redundante

```
IF OBJECT_ID('Studiouri Redundante', 'V') IS NOT NULL
	DROP VIEW [Studiouri Redundante];
GO
CREATE VIEW [Studiouri Redundante] AS
SELECT S.StudioID, MAX(S.ParentID) AS ParentID, MAX(S.StudioName) AS StudioName
FROM Studios S
LEFT JOIN Movies M
ON M.StudioID = S.StudioID
GROUP BY S.StudioID
HAVING COUNT(M.MovieID) =0
```

Acest view conține studiourile care sunt introduse în tablul cu studouri, care nu au nici un film produse de acestea în baza de date și are pot fi arhivate sau șterse.

2.Angajați pensionabili

```
IF OBJECT_ID('Angajați Pensionabili', 'V') IS NOT NULL
	DROP VIEW [Angajați Pensionabili];
GO
CREATE VIEW [Angajați Pensionabili] AS
SELECT E.*
FROM Employees E
WHERE DATEDIFF(YEAR, GETDATE(), E.BirthDate) > 60
```

3.Clienți fideli

```
IF OBJECT_ID('Clienți Fideli', 'V') IS NOT NULL
	DROP VIEW [Clienți Fideli];
GO
CREATE VIEW [Clienți Fideli] AS
SELECT C.FirstName, C.LastName, COUNT(T.TicketID) AS [Bilete Cumpărate]
FROM Customers C
JOIN Tickets T
ON C.TicketID = T.TicketID
GROUP BY C.FirstName, C.LastName, C.Regular
HAVING C.Regular = 1
```

Acest view conține numele clienților marcați ca fideli și numărul de bilete cumpărate de aceștia.

4. Cost total

```
IF OBJECT_ID('Cost Total', 'V') IS NOT NULL
	DROP VIEW [Cost Total];
GO
CREATE VIEW [Cost Total] AS
SELECT E.Type, SUM(E.CostPerMonth) AS [Cost]
FROM Expenses E
JOIN Branches B
ON E.BranchID = B.BranchID
GROUP BY E.Type
```

Acest view conține costul total pentru fiecare tip de cheltuială (mâncare, băutură și utilitate) per total pe toate sucursalele.

5.Salarii

```
IF OBJECT_ID('Salarii', 'V') IS NOT NULL
	DROP VIEW [Salarii];
GO
CREATE VIEW [Salarii] AS
SELECT E.EmployeeID, E.FirstName, E.LastName, E.Title, E.SalariuBrut,
E.SalariuBrut /4 AS [Asigurare Socială]
, E.SalariuBrut / 10 AS [Asigurări Sociale de Sănătate],
(E.SalariuBrut - (E.SalariuBrut /4 +  E.SalariuBrut / 10 )) / 10 AS [Impozit pe Venit],
E.Impozit, E.SalariuBrut - E.Impozit AS [Salariu Net]
FROM Employees E
```

În acest view sunt stocate toate datele despre salariile și impozitele pe care le plătesc și primesc angajații

6.Profit per Sucursală

```
IF OBJECT_ID('Profit Per Sucusală', 'V') IS NOT NULL
	DROP VIEW [Profit Per Sucusală];
GO
CREATE VIEW [Profit Per Sucusală] AS
SELECT B.Name, (COUNT(T.TicketID) * 20 - COUNT(T.TicketID) * 20 * (AVG(T.Discount)/100)-
SUM(E.CostPerMonth)) AS [Profit]
FROM Branches B
JOIN Tickets T
ON T.BranchID = B.BranchID
JOIN Expenses E
ON E.BranchID = B.BranchID
GROUP BY B.Name
ORDER BY [Profit] DESC
```

TRIGGERS

```
--1.Trigger pentru ștergerea studiourilor redundante
IF OBJECT_ID('Delete Studios', 'TR') is NOT NULL

	DROP TRIGGER [Delete Studios]
GO
```

```
CREATE TRIGGER [Delete Studios]
ON Movies
AFTER INSERT, DELETE
AS
BEGIN
	DELETE Studios
	FROM Studios
	JOIN [Studiouri Redundante]
	ON Studios.StudioID = [Studiouri Redundante].StudioID
	WHERE Studios.StudioID = [Studiouri Redundante].StudioID AND
	Studios.ParentID is NOT NULL
END


-- 2. Crearea unui trigger care verifică dacă s-a introdus o adresă duplicat
IF OBJECT_ID('Address Duplicate', 'TR') IS NOT NULL
	DROP TRIGGER [Address Duplicate];
GO
CREATE TRIGGER [Address Duplicate]
ON Addresses
AFTER INSERT, UPDATE
AS
BEGIN
	IF @@ROWCOUNT = 0 RETURN;
	SET NOCOUNT ON;
	IF EXISTS
	(
		SELECT COUNT(*)
		FROM Inserted AS I
		JOIN Addresses AS A
		ON I.City = A.City
		GROUP BY A.City
		HAVING COUNT(*) > 1
	)
	BEGIN
		PRINT 'Warning for duplicate address';
	END
END;

--3. Crearea unui trigger care afișează coloanele introduse sau updatate în tabelul
Employees
IF OBJECT_ID('InsertOrUpdateTrigger', 'TR') IS NOT NULL
	DROP TRIGGER InsertOrUpdateTrigger;
GO
CREATE TRIGGER InsertOrUpdateTrigger
ON Employees
AFTER DELETE, INSERT, UPDATE
AS
BEGIN
	IF @@ROWCOUNT = 0 RETURN;
	SET NOCOUNT ON;
	SELECT COUNT(*) AS 'Number of Inserted Rows'
	FROM Inserted;
	SELECT COUNT(*) AS 'Number of Deleted Rows'
	FROM Deleted;
END;
```

```sql
--4.Calcularea automată a prețurilor biletelor
IF OBJECT_ID('TicketPriceCalculator', 'TR') IS NOT NULL
	DROP TRIGGER TicketPriceCalculator;
GO
CREATE TRIGGER TicketPriceCalculator
ON Tickets
AFTER INSERT, UPDATE
AS
BEGIN
	UPDATE Tickets
	SET TicketPrice = 20 * Discount / 100
END;
```

SELECT

```sql
--1.Arătați Numele angajaților care sunt din Transilvania
SELECT EmployeeID, LastName, FirstName, E.AddressID,A.County,A.City
FROM Employees E
JOIN Addresses A
ON E.AddressID = A.AddressID
JOIN Territories T
ON A.TerritoryID = T.TerritoryID
WHERE T.Description  = N'Transilvania'
ORDER BY EmployeeID
```

	EmployeeID	LastName	FirstName	AddressID	County	City
1	7	Ceaușescu	Florin	14	Sălaj	Zalău
2	9	Ungur	Valeria	8	Sibiu	Sibiu
3	21	Cojocaru	Iancu	1	Cluj	Cluj-Napoca
4	29	Roșu	Costin	15	Alba	Alba Iulia
5	30	Manea	Adelina	15	Alba	Alba Iulia

```sql
--2.Afișați toți clienții care au cumpărat bilete la filme produse de Universal Pictures
sau Studiourile afiliate
SELECT CustomerID, LastName, FirstName,M.Title, S.StudioName
FROM Customers C
INNER JOIN Tickets T
ON C.TicketID = T.TicketID
INNER JOIN Movies M
ON T.MovieID = M.MovieID
INNER JOIN Studios S
ON M.StudioID = S.StudioID
WHERE S.StudioID = 4 OR S.ParentID = 4
ORDER BY CustomerID
```

	CustomerID	LastName	FirstName	Title	StudioName
1	3	Crețu	Andrei	Shrek the Third	DreamWorks
2	4	Crețu	Virgil	Shrek the Third	DreamWorks
3	5	Crețu	Maria	Shrek the Third	DreamWorks
4	21	Vântu	Adrian	A Beatiful Mind	DreamWorks
5	22	Vadim	Marian	A Beatiful Mind	DreamWorks
6	23	Miruna	Adriana	A Beatiful Mind	DreamWorks
7	24	Rudaru	Iulian	A Beatiful Mind	DreamWorks
8	25	Stănciugel	Gheorghe	Catch Me If You Can	DreamWorks
9	26	Gelu	Cătălin	Catch Me If You Can	DreamWorks
10	27	Butoi	Raluca	Catch Me If You Can	DreamWorks
11	28	Iacobescu	Sebastian	Catch Me If You Can	DreamWorks
12	29	Popa	Narcisa	Jarhead	Universal Pictures
13	30	Dumitrescu	Dragoș	Jarhead	Universal Pictures
14	31	Moroșanu	Rebeca	Jarhead	Universal Pictures
15	32	Constantin	Simon	Jarhead	Universal Pictures

```sql
--3.Afișați sucursalele care primesc produse din alte localități
CREATE PROCEDURE ProduseNonLocale
AS
SELECT B.BranchID, B.Name, A1.County AS 'Branch County', P2.CompanyName, A2.County AS
'Provider County'
FROM Branches B
JOIN Expenses E
ON B.BranchID = E.BranchID
JOIN Products P1
ON E.ProductID = P1.ProductID
JOIN Providers P2
ON P1.ProviderID = P2.ProviderID
JOIN Addresses A1
ON A1.AddressID = B.AddressID
JOIN Addresses A2
ON A2.AddressID = P2.AddressID
WHERE A1.County != A2.County
GO

EXEC ProduseNonLocale
```

	BranchID	Name	Branch County	CompanyName	Provider Cour
1	1	Cinema Flacăra	Vâlcea	Enel	București
2	1	Cinema Flacăra	Vâlcea	Chio	București
3	1	Cinema Flacăra	Vâlcea	Chio	București
4	1	Cinema Flacăra	Vâlcea	Chio	București
5	1	Cinema Flacăra	Vâlcea	Chio	București
6	1	Cinema Flacăra	Vâlcea	PepsiCo Snacks	București
7	1	Cinema Flacăra	Vâlcea	PepsiCo Snacks	București
8	1	Cinema Flacăra	Vâlcea	PepsiCo Snacks	București
9	1	Cinema Flacăra	Vâlcea	PepsiCo Băuturi	București
10	1	Cinema Flacăra	Vâlcea	Coca-Cola România	București
11	1	Cinema Flacăra	Vâlcea	Borsec	București
12	1	Cinema Flacăra	Vâlcea	Engie	București
13	2	CineMagic	Vâlcea	Enel	București
14	2	CineMagic	Vâlcea	Chio	București
15	2	CineMagic	Vâlcea	Chio	București
16	2	CineMagic	Vâlcea	Chio	București
17	2	CineMagic	Vâlcea	Chio	București
18	2	CineMagic	Vâlcea	PepsiCo Snacks	București
19	2	CineMagic	Vâlcea	PepsiCo Snacks	București
20	2	CineMagic	Vâlcea	PepsiCo Snacks	București
21	2	CineMagic	Vâlcea	PepsiCo Băuturi	București
22	2	CineMagic	Vâlcea	Coca-Cola România	București
23	2	CineMagic	Vâlcea	Borsec	București
24	2	CineMagic	Vâlcea	Engie	București
25	3	Royal Cinema	Gorj	Enel	București
26	3	Royal Cinema	Gorj	Chio	București
27	3	Royal Cinema	Gorj	Chio	București
28	3	Royal Cinema	Gorj	Chio	București
29	3	Royal Cinema	Gorj	Chio	București
30	3	Royal Cinema	Gorj	PepsiCo Snacks	București
31	3	Royal Cinema	Gorj	PepsiCo Snacks	București
32	3	Royal Cinema	Gorj	PepsiCo Snacks	București
33	3	Royal Cinema	Gorj	PepsiCo Băuturi	București
34	3	Royal Cinema	Gorj	Coca-Cola România	București
35	3	Royal Cinema	Gorj	Apavil	Vâlcea
36	3	Royal Cinema	Gorj	Borsec	București
37	3	Royal Cinema	Gorj	Engie	București

```sql
--4.Ordonați regiunile după numărul de angajați din acestea
CREATE PROCEDURE AngajațiPerRegiune
AS
SELECT T.Description, COUNT(E.EmployeeID) AS [Număr de Angajați]
FROM Territories T
JOIN Addresses A
ON A.TerritoryID = T.TerritoryID
JOIN Employees E
```

```
ON E.AddressID = A.AddressID
GROUP BY T.Description
ORDER BY [Număr de Angajați] DESC
GO
```

	Description	Număr de Angajați
1	Muntenia	13
2	Moldova	5
3	Transilvania	5
4	Oltenia	4
5	București	3

```
--5. Arată adresele care nu sunt atribuite nicăieri
SELECT *
FROM Addresses A
LEFT JOIN Employees E
ON A.AddressID = E.AddressID
LEFT JOIN Branches B
ON A.AddressID = B.AddressID
LEFT JOIN Providers P
ON A.AddressID = P.AddressID
WHERE (E.EmployeeID is NULL) AND (B.BranchID is NULL) AND (P.ProviderID is NULL)
```

	AddressID	TerritoryID	County	City	Street	Number
1	16	1	Cluj	Cluj-Napoca	Mincă Dumitru	40
2	34	3	Vâlcea	Drăgășani	Rusidava	3
3	42	4	București	Sector 3	Idilei	10
4	43	4	București	Sector 3	Idilei	21
5	44	4	București	Sector 3	Idilei	35
6	46	4	București	Sector 5	Bvd. Tinuretului	3

```
--6. Ordonați studiourile de filme în funcție de numărul de filmele produse
SELECT S.StudioName, COUNT(M.title) AS [Movies]
FROM Movies M
LEFT JOIN Studios S
ON M.StudioID = S.StudioID
WHERE S.StudioID IS NOT NULL
GROUP BY S.StudioName
ORDER BY [Movies] DESC
```

	StudioName	Movies
1	Warner Bros. Studios	7
2	DreamWorks	6
3	Paramount Pictures	4
4	Marvel Studios	3
5	20th Century Fox	2
6	Columbia Pictures	2
7	Universal Pictures	1

```
--7.Afișați numărul de tichete cumpărate de studenți în fiecare an
SELECT YEAR(T.Date) AS [An], COUNT(T.TicketID) AS [Număr de Tichete]
FROM Tickets T
JOIN Customers C
```

```
ON C.TicketID = T.TicketID
WHERE C.Status = 'Student'
GROUP BY YEAR(T.Date)
```

	An	Număr de Tichete
1	2016	4
2	2017	9
3	2018	1
4	2019	5

```
--8.Afișați furnizorii în funcție de tipurile de produse pe care le vând
SELECT P2.CompanyName, MAX(E.Type) AS [Tip de Produs]
FROM Expenses E
JOIN Products P1
ON E.ProductID = P1.ProductID
JOIN Providers P2
ON P1.ProviderID = P2.ProviderID
GROUP BY P2.CompanyName
```

	CompanyName	Tip de Produs
1	Apa Nova	Utilitate
2	Apavil	Utilitate
3	Borsec	Băutură
4	Chio	Mâncare
5	Coca-Cola România	Băutură
6	Enel	Utilitate
7	Engie	Utilitate
8	PepsiCo Băuturi	Băutură
9	PepsiCo Snacks	Mâncare

```
--9.Ordonați furnizorii în funcție de costuri
SELECT P2.CompanyName, SUM(E.CostPerMonth)AS [Costuri]
FROM Expenses E
JOIN Products P1
ON E.ProductID = P1.ProductID
JOIN Providers P2
ON P1.ProviderID = P2.ProviderID
GROUP BY P2.CompanyName
ORDER BY [Costuri] DESC
```

	CompanyName	Costuri
1	Chio	9923.00
2	Enel	9051.00
3	PepsiCo Snacks	8189.00
4	Borsec	5493.00
5	PepsiCo Băuturi	5180.00
6	Coca-Cola România	3685.00
7	Apavil	3235.00
8	Engie	3022.00
9	Apa Nova	2705.00

```sql
--10.Ordonați funcțile angajaților după salariul net
SELECT E.Title, AVG(E.SalariuBrut - E.Impozit) AS [Media Salarilor Net]
FROM Employees E
GROUP BY E.Title
ORDER BY [Media Salarilor Net] DESC
```

	Title	Media Salarilor Net
1	Manager Șef	4849
2	Crew Leader	3711
3	Asistent Manager	3178
4	Vânzător Bilete	1929
5	Portar	1698
6	Vânzător Snacks	1687

```sql
--11.Calculați cantitatea de produse Consumată în fiecare lună
SELECT MAX(P2.CompanyName) as [Numele Companiei], P1.ProductName, (SUM(E.CostPerMonth) /
MAX(P1.PricePerUnit)) AS [Cantitate], MAX(P1.Unit) as [Unitate de Măsură]
FROM Products P1
JOIN Expenses E
ON E.ProductID = P1.ProductID
JOIN Providers P2
ON P1.ProviderID = P2.ProviderID
GROUP BY P1.ProductName
ORDER BY MAX(P2.CompanyName)
```

	Numele Companiei	ProductName	Cantitate	Unitate de Măsură
1	Apa Nova	Apă Curentă	1218.46846846847	m^3
2	Apavil	Apă Curentă Vâlcea	864.973262032085	m^3
3	Borsec	Apă Potabilă	1301.65876777251	m^3
4	Chio	Chips Chio	115.137931034483	Kg
5	Chio	Popcorn	56.74	Kg
6	Chio	Sos	65.2	l
7	Chio	Tortillas	63.9642857142857	Kg
8	Coca-Cola România	Băuturi Cola	1052.85714285714	l
9	Enel	Curent	67044.4444444444	KWh
10	Engie	Gaz	5925.49019607843	KWh
11	PepsiCo Băuturi	Băuturi Pepsi	1726.66666666667	l
12	PepsiCo Snacks	Doritos	108.653846153846	Kg
13	PepsiCo Snacks	Chips Lay's	87.2333333333333	Kg
14	PepsiCo Snacks	Snacks Star	98.1071428571429	Kg

```sql
--12.Afișați studiourile care au produs cel puțin 1 film prezente în baza de date
SELECT S.StudioName, COUNT(M.MovieID) AS [Filme]
FROM Movies M
JOIN Studios S
ON M.StudioID = S.StudioID
GROUP BY S.StudioName
HAVING COUNT(M.MovieID) > 0
ORDER BY [Filme]
```

	StudioName	Filme
1	Universal Pictures	1
2	20th Century Fox	2
3	Columbia Pictures	2
4	Marvel Studios	3
5	Paramount Pictures	4
6	DreamWorks	6
7	Warner Bros. Studios	7

```sql
--13.Afișați sucursalele care au vândut cel puțin 5 bilete
SELECT B.Name, COUNT(T.TicketID) as [Tichete Vândute]
FROM Branches B
JOIN Tickets T
ON T.BranchID = B.BranchID
GROUP BY B.Name
HAVING COUNT(T.TicketID)>4
ORDER BY [Tichete Vândute] DESC
```

	Name	Tichete Vândute
1	Royal Cinema	12
2	City Cinema	8
3	Cinema Flacăra	7

```sql
--14.Afișați angajații cu salariul peste medie
SELECT (MAX(E.FirstName) + ' ' + MAX(E.LastName)) AS [Nume], E.SalariuBrut
FROM Employees E
GROUP BY E.SalariuBrut
HAVING E.SalariuBrut > (SELECT AVG(Employees.SalariuBrut) FROM Employees)
```

	Nume	SalariuBrut
1	Ivan Apostolu	5031
2	Darius Găină	5903
3	Mihail Ștefănescu	6043
4	Valeria Ungur	6051
5	Mircea Ioan Iliescu	6437
6	Adelina Manea	6455
7	Claudiu Teodorescu	6530
8	Irina Georgescu	6790
9	Amalia Șerbănescu	7021
10	Adrian Ionuț Georgescu	8032
11	Aurora Crețu	8322
12	Adelin Marmură	8930
13	Marian Andrei Adam	9133

```sql
--15.Afișați angajații care lucrează de cel puțin 4 ani
SELECT MAX(E.FirstName) +' ' + MAX(E.LastName) AS [Nume], (YEAR(GETDATE()) -
YEAR(MAX(E.HireDate))) AS [Ani De Muncă]
FROM Employees E
GROUP BY E.FirstName
HAVING (YEAR(GETDATE()) - YEAR(MAX(E.HireDate))) >= 4
```

```
ORDER BY [Ani De Muncă]
```

	Nume	Ani De Muncă
1	Adelin Mamură	4
2	Amalia Șerbănescu	4
3	Claudiu Teodorescu	4
4	Darius Găină	4
5	Irina Georgescu	4
6	Marian Andrei Adam	4
7	Marian Iulian Cozma	5
8	Mariuța Cojoc	5
9	Marian Niculescu	5
10	Iancu Cojocaru	5
11	Andreea Cizmă	5
12	Aurora Crețu	5
13	Adelina Mirela Năstase	5
14	Răzvan Popa	6
15	Miron Ionescu	7

```
--16.Afișați angajații cu cel mai mare salariu comparat comparând pe cei cu aceeași
funcție
SELECT E.FirstName, E.Title, E.SalariuBrut
FROM Employees E
GROUP BY E.Title, E.FirstName, E.SalariuBrut
HAVING MAX(E.SalariuBrut) = (SELECT MAX(E1.SalariuBrut) FROM Employees E1 WHERE E1.Title
= E.Title)
ORDER BY E.SalariuBrut DESC
```

	FirstName	Title	SalariuBrut
1	Marian Andrei	Manager Șef	9133
2	Irina	Crew Leader	6790
3	Adelina	Asistent Manager	6455
4	Mariuța	Vânzător Bilete	4520
5	Miron	Portar	3590
6	Maria	Vânzător Snacks	3209

```
--17.Afișați cele mai costisitoare cheltuieli după tipul acestora
SELECT E1.Type, E1.CostPerMonth
FROM Expenses E1
JOIN Products P
ON E1.ProductID = E1.ProductID
WHERE E1.CostPerMonth = (SELECT MAX(CostPerMonth) FROM Expenses E2 WHERE E1.Type =
E2.Type )
Group By E1.Type, E1.CostPerMonth
```

```
ORDER BY E1.CostPerMonth DESC
```

	Type	CostPerMonth
1	Utilitate	2413.00
2	Băutură	1303.00
3	Mâncare	812.00

```
--18.Afișați studioul cu cele mai multe filme produse
SELECT TOP 1 S.StudioName, COUNT(M.MovieID) AS [Filme]
FROM Movies M
JOIN Studios S
ON M.StudioID = S.StudioID
GROUP BY S.StudioName
ORDER BY [Filme] DESC
```

	StudioName	Filme
1	Warner Bros. Studios	7

```
--19.Afișați filmele care durează cel mai mult grupate după rating
SELECT M.Title, M.[MovieLength(minutes)], M.Rating
FROM Movies M
GROUP BY M.Rating, M.Title,  M.[MovieLength(minutes)]
HAVING M.[MovieLength(minutes)] = (SELECT MAX(M1.[MovieLength(minutes)]) FROM Movies M1
WHERE M.Rating = M1.Rating)
```

	Title	MovieLength(minutes)	Rating
1	Mortal Kombat	110	6
2	Titanic	194	7
3	The Irishman	209	8
4	Pulp Fiction	154	9

```
--20.Afișați directorii de filme si numărul de filme produse de aceștia
SELECT M.Director, COUNT(M.MovieID) AS [Filme Produse]
FROM Movies M
GROUP BY M.Director
ORDER BY [Filme Produse] DESC
```

	Director	Filme Produse
1	Chad Stahelski	3
2	Quentin Tarantino	3
3	Martin Scorsese	3
4	Andrew Adamson	2
5	David Fincher	2
6	Edward Zwick	2
7	Ethan Coen	1
8	George Miller	1
9	James Cameron	1
10	Joon-ho Bong	1
11	Lana Wachowski	1
12	David Leitch	1
13	David Yates	1
14	Chris Miller	1
15	Christopher Nolan	1
16	Ron Howard	1
17	Ryan Coogler	1
18	Sam Mendes	1
19	Simon McQuoid	1
20	SJean-Marc Vell...	1
21	Steve McQueen	1
22	Steven Spilberg	1
23	Tin Miller	1
24	Mike Mitchell	1
25	Miroslav Slabos...	1

```sql
--21.Afișați suma totală de plată pe lună pentru fiecare sucursală
SELECT B.Name, SUM(E.CostPerMonth) AS [Costuri Lunare]
FROM Branches B
JOIN Expenses E
ON B.BranchID = E.BranchID
GROUP BY B.Name
```

	Name	Costuri Lunare
1	Cinema Flacăra	9768.00
2	CineMagic	10602.00
3	Cinematograf Flacăra București	11028.00
4	City Cinema	11840.00
5	Royal Cinema	7245.00

```
--22.Afișați numărul maxim de bilete care trebuie vândut de fiecare sucursală pentru a
face un profit de 1000 de lei/lună pentru fiecare sucursală,
--având în vedere că prețul unui bilet este de 20 lei
SELECT B.Name, ((SUM(E.CostPerMonth) + 1000) * 5) / (100 - AVG(t.Discount)) AS [Număr
Necesar de Bilete]
FROM Branches B
JOIN Expenses E
ON B.BranchID = E.BranchID
JOIN Tickets T
ON T.BranchID = T.TicketID
GROUP BY B.Name
ORDER BY [Număr Necesar de Bilete] DESC
```

	Name	Număr Necesar de Bilete
1	City Cinema	755.2941
2	Cinematograf Flacăra București	707.5294
3	CineMagic	682.4705
4	Cinema Flacăra	633.4117
5	Royal Cinema	485.00

```
--23.Ordonați teritoriile după numărul de clienți
SELECT T1.Description, COUNT(T2.TicketID) AS [Număr de Clienți]
FROM Territories T1
JOIN Addresses A
ON A.TerritoryID = T1.TerritoryID
JOIN Branches B
ON B.AddressID = A.AddressID
JOIN Tickets T2
ON T2.BranchID = B.BranchID
GROUP BY T1.Description
ORDER BY [Număr de Clienți] Desc
```

	Description	Număr de Clienți
1	Oltenia	15
2	București	10
3	Muntenia	7

```
--24.Afișați numele și numărul de telefon angajaților care își serbează ziua de naștere
SELECT (E.FirstName + ' ' + E.LastName) AS [Nume], E.Phone
FROM Employees E
```

```
WHERE MONTH(E.BirthDate) = MONTH(GETDATE()) AND DAY(E.BirthDate) = DAY(GETDATE())
```

	Nume	Phone
1	Andrei Mihai Rotaru	0747557172

```
--25.Afișați biletele care expiră azi și cine le-a cumpărat
SELECT T.TicketID, B.Name, C.FirstName, C.LastName, M.Title AS [Movie Title], T.Date
FROM Tickets T
JOIN Movies M
ON M.MovieID = T.MovieID
JOIN Customers C
ON C.TicketID = T.TicketID
JOIN Branches B
ON B.BranchID = T.BranchID
WHERE YEAR(T.Date) = YEAR(GETDATE()) AND MONTH(T.Date) = MONTH(GETDATE()) AND DAY(T.Date)
= DAY(GETDATE())
```

	TicketID	Name	FirstName	LastName	Movie Title	Date
1	18	City Cinema	Ionuț	Catană	Mortal Kombat	2021-05-12

INSERT

```
--1.
INSERT INTO Addresses( TerritoryID, County, City, Street, Number)
VALUES (3, N'Argeș', N'Pitești', N'Daciei',13)

--2.
INSERT INTO Movies(StudioID, Title, Director, LaunchDate, [MovieLength(minutes)], Genre,
Rating)
VALUES (2, 'Joker','Todd Phillips','2019-10-02',122,'Drama',8)

--3.
INSERT INTO Employees(BranchID, AddressID, LastName, FirstName, BirthDate, HireDate,
Title, Phone, SalariuBrut, Impozit)
VALUES(5,15,N'Roșu', N'Costin', '1990-06-09', '2019-06-22', N'Vânzător Snacks',
'0767368544', 3109,  1289)

--4.
INSERT INTO Addresses( TerritoryID, County, City, Street, Number)
VALUES (3, N'Argeș', N'Mioveni', N'Revoluției',22)

--5.
INSERT INTO Addresses( TerritoryID, County, City, Street, Number)
VALUES (3, N'Argeș', N'Topoloveni', N'Scândurii',13)

--6.
INSERT INTO Tickets(MovieID, BranchID, Discount, Row,Seat,Date)
VALUES(32,2,0,2,4,'2019-10-02')

--7.
INSERT INTO Tickets(MovieID, BranchID, Discount, Row,Seat,Date)
VALUES(32,3,5,14,8,'2019-10-02')

--8.
```

```
INSERT INTO Addresses( TerritoryID, County, City, Street, Number)
VALUES (3, N'Argeș', N'Mioveni', N'Revoluției',24)

 --9.
 INSERT INTO Addresses( TerritoryID, County, City, Street, Number)
VALUES (3, N'Argeș', N'Câmpulung', N'Mihai Țerbea',10)

--10.
INSERT INTO Customers(TicketID, LastName, FirstName,Status, Regular)
VALUES (13,N'Marinescu',N'Vasile','Student',0)

--11.
INSERT INTO Customers(TicketID, LastName, FirstName,Status, Regular)
VALUES (33,N'Marian',N'Tudor','Adult',1)
SELECT * FROM Tickets
--12.
INSERT INTO Customers(TicketID, LastName, FirstName,Status, Regular)
VALUES (16,N'Stănescu',N'Codrin','Adult',1)

--13.
INSERT INTO Customers(TicketID, LastName, FirstName,Status, Regular)
VALUES (17,N'Codrea',N'Mihai','Adult',1)

--14.
INSERT INTO Customers(TicketID, LastName, FirstName,Status, Regular)
VALUES (18,N'Catană',N'Ionuț','Adult',1), (19,N'Scorsese',N'Matei','Adult',1)

--15.
INSERT INTO Employees(BranchID, AddressID, LastName, FirstName, BirthDate, HireDate,
Title, Phone, SalariuBrut, Impozit)
VALUES(5,48,N'Calotă', N'Cătălin', '1993-02-15', '2019-06-22', N'Vânzător Snacks',
'0767354384', 3109,  1289)
```

UPDATE

```
--1.Promovați angajații care lucrează de cel puțin 3 ani sub funcția de Asistent Manager
la Manager
UPDATE Employees
SET Title = 'Manager'
WHERE Title = 'Asistent Manager' AND DATEDIFF(year, HireDate, GETDATE()) >=3

--2.Schimbați studiourile filmelor din studioul copil în părinte
UPDATE M
SET M.StudioID = S.ParentID
FROM Movies M
JOIN Studios S
ON M.StudioID = S.StudioID
WHERE S.ParentID IS NOT NULL AND M.StudioID IS NOT NULL

--3.Scadeți cu 10% salariul angajaților cu funcția de manager și creșteți cu 10%
UPDATE Employees
SET SalariuBrut = CASE
WHEN Title like '%Manager%' THEN SalariuBrut * 9 / 10
ELSE SalariuBrut * 11 / 10
END
```

```sql
--4.Treceți București ca localitate pentru adresele din teritoriul București
UPDATE A
SET A.County = N'București'
FROM Addresses A
JOIN Territories T
ON T.TerritoryID = A.TerritoryID
WHERE T.Description = N'București'

--5.Puneți studiourile fără părinte ca fiind propriul lor părinte
UPDATE Studios
SET ParentID = StudioID
WHERE ParentID is NULL

--6.Scade cu 50% costul pentru mâncare și 60% pentru băuturi
UPDATE Expenses
SET CostPerMonth = CASE
WHEN Type = N'Mâncare' THEN CostPerMonth / 2
WHEN Type = N'Băutură' THEN CostPerMonth * 3 / 5
ELSE CostPerMonth
END

--7.Creșteți cu 1 ratingul la filmele de tip dramă
UPDATE Movies
SET Rating = Rating + 1
WHERE Genre like '%Drama%' AND Rating < 10

--8.Schimbați teritoriul adreselor din București în Muntenia
UPDATE Addresses
SET TerritoryID = 3
WHERE TerritoryID = 4

--9.Schimbați unitatea de măsură a produselor din kg în grame
UPDATE Products
SET Unit = 'g', PricePerUnit = PricePerUnit / 1000
WHERE Unit = 'Kg'

--10.Schimbați numele de familie al unui angajat, datorită unei căsătorii
DECLARE @numeȘiPrenumeVechi NVARCHAR(30) = N'Șerbănescu Amalia';
DECLARE @numeNou NVARCHAR(30) = 'Iliescu';
UPDATE E
SET E.LastName = @numeNou
FROM Employees E
WHERE (E.LastName + ' ' + E.FirstName) = @numeȘiPrenumeVechi

--11.Adăugarea a 3 rânduri de locuri în cinematografele din București
UPDATE B
SET B.SeatRows = B.SeatRows + 3
FROM Branches B
JOIN Addresses A
ON A.AddressID = B.AddressID
WHERE A.TerritoryID = 4

--12.Scutirea de la plătirea impozitului a angajaților cu o vechime de peste 5 ani
UPDATE Employees
SET Impozit = 0
WHERE DATEDIFF(YEAR, HireDate,  GETDATE()) >=5
```

```sql
--13.Creșterea salariului cu 2% a angajaților din București
UPDATE E
SET SalariuBrut = SalariuBrut * 102 / 100
FROM Employees E
JOIN Addresses A
ON E.AddressID = A.AddressID
WHERE A.TerritoryID = 4

--14.Mutarea tuturor angajaților care lucrează la sucursalele din București
UPDATE A
SET A.County = N'București', A.City = N'București'
FROM Addresses A
JOIN Employees E
ON A.AddressID = E.AddressID
JOIN Branches B
ON E.BranchID = B.BranchID
JOIN Addresses AB
ON AB.AddressID = B.AddressID
WHERE AB.TerritoryID = 4

--15.Realculați discountul biletelor
UPDATE T
SET T.Discount = (CASE
       WHEN C.Regular = 1 THEN 5
       WHEN C.Status like 'Pensionar' THEN 100
       WHEN C.Status like 'Copil' THEN 100
       ELSE 0 END) + (CASE
       WHEN (C.Status like 'Student') THEN 15
       WHEN (C.Status like 'Elev') THEN 15
       ELSE 0
       END)
FROM Tickets T
JOIN Customers C
ON C.TicketID = T.TicketID
```

DELETE

```sql
--1.Ștergeți toți angajații pensionabil(care au vârsta de peste 60 de ani)
DELETE FROM Employees
WHERE DATEDIFF(YEAR,Employees.BirthDate,  GETDATE()) >= 60

--2.Ștergeți angajații din Vâlcea
DECLARE @județ NVARCHAR(30) = 'Vâlcea';
DELETE E
FROM Employees E
JOIN Addresses A
ON E.AddressID = A.AddressID
WHERE A.County = @județ

DELETE A
FROM Employees E
JOIN Addresses A
ON E.AddressID = A.AddressID
WHERE A.County = @județ
```

```sql
--3.Ștergeți biletele care au expirat acum mai bine de 1 ani
DECLARE @vechimeaBiletului int = 1;
UPDATE C
SET C.TicketID = NULL
FROM Customers C
JOIN Tickets T
ON T.TicketID = C.TicketID
WHERE DATEDIFF(YEAR,T.Date, GETDATE() ) >= @vechimeaBiletului

DELETE
FROM Tickets
WHERE DATEDIFF(YEAR,Tickets.Date, GETDATE() ) >= @vechimeaBiletului

--4.Ștergeți filmele prduse de studioul Warner Bros
DECLARE @studio NVARCHAR(30) = 'Warner Bros. Studios';

UPDATE T
SET T.MovieID = NULL
FROM Tickets T
JOIN Movies M
ON T.MovieID = M.MovieID
JOIN Studios S
ON M.StudioID = S.StudioID
WHERE S.StudioName = @studio

DELETE M
FROM Movies M
JOIN Studios S
ON M.StudioID = S.StudioID
WHERE S.StudioName = @studio

--5.Ștergeți providerul care consumă cel mai mult

DECLARE @numeProvider NVARCHAR(30) =
	(
	SELECT MAX(P.CompanyName)
	FROM Providers P
	JOIN Products P1
	ON P.ProviderID = P1.ProviderID
	JOIN Expenses E
	ON E.ProductID = P1.ProductID
	GROUP BY E.Type
	HAVING SUM(E.CostPerMonth) =
		(
		SELECT TOP 1 SUM(E1.CostPerMonth)
		FROM Expenses E1
		GROUP BY E1.Type
		ORDER BY SUM(E1.CostPerMonth) DESC
		)
	)

UPDATE P1
SET P1.ProviderID = NULL
FROM Products P1
JOIN Providers P2
ON P1.ProviderID = P2.ProviderID
WHERE P2.CompanyName = @numeProvider
```

```sql
DELETE P
FROM Providers P
WHERE P.CompanyName = @numeProvider

--6.Ștergeți anajații care plătesc cel mai mult impozit din fiecare funcție
DELETE E
FROM Employees E
WHERE E.Impozit = (
       SELECT MAX(E1.Impozit)
       FROM Employees E1
       WHERE E1.Title = E.Title
       GROUP BY E1.Title
       )

--7.Ștergeți angajații născuți în an bisect
DELETE FROM Employees
WHERE YEAR(Employees.BirthDate) % 4 = 0

--8.Ștergeți produsele din fiecare tip în afară de cele mai ieftine(excepție utilități)
DELETE E
FROM Expenses E
WHERE E.Type != 'Utilitate' AND E.ExpenseID != (
       SELECT MIN(E1.ExpenseID)
       FROM Expenses E1
       GROUP BY E1.Type
       HAVING E1.Type = E.Type
       )

--9.Ștergeți biletele vândute de sucursala City Cinema
DECLARE @numeSucursala NVARCHAR(30) = 'City Cinema'

UPDATE C
SET C.TicketID = NULL
FROM Customers C
JOIN Tickets T
ON T.TicketID = C.TicketID
JOIN Branches B
ON B.BranchID = T.BranchID
WHERE B.Name = @numeSucursala

DELETE T
FROM Tickets T
JOIN Branches B
ON T.BranchID = B.BranchID
WHERE B.Name = @numeSucursala

--10.Ștergeți clienții care nu sunt fideli
DELETE FROM Customers
WHERE Regular = 0
```