



Estácio

Polo Tijuca

Aluno	Luiz Felipe Sarmiento Bonet
Matrícula	2023.09.34276-6
Curso	Desenvolvimento Full Stack
Disciplina	RPG0015 - Vamos Manter as Informações?
Turma	9001
Período	2024 – 2º Semestre

Missão Prática: Mundo 3 - Nível 2

Título da Prática 1: Criando o Banco de Dados

Título da Prática 2: Alimentando a Base

Objetivos:

- Identificar os requisitos de um sistema e transformá-los no modelo adequado;
- Utilizar ferramentas de modelagem para bases de dados relacionais;
- Explorar a sintaxe SQL na criação das estruturas do banco (DDL);
- Explorar a sintaxe SQL na consulta e manipulação de dados (DML);
- Modelar a base de dados para um sistema simples, além de implementá-la, através da sintaxe SQL, na plataforma do SQL Server.

Resultados de Execução – Prática 1

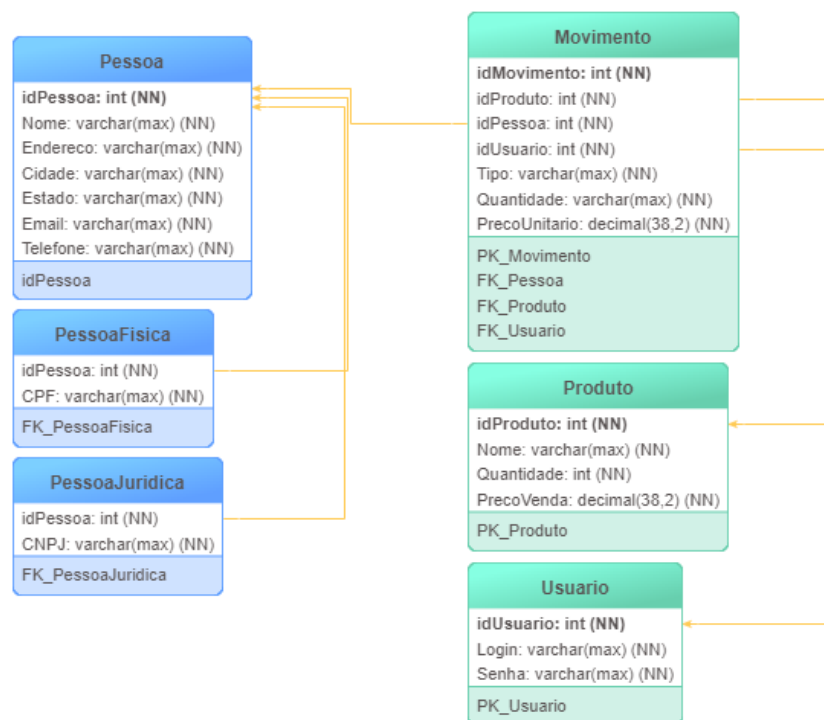


Figura 1. Modelo de dados do sistema a ser criado. Após a criação no DBDesigner, o modelo foi editado no dbDiff.

Resultados de Execução – Prática 2

Felps-Notebook.Loja - dbo.Pessoa		SQLQuery1.sql - fe...OTEBOOK(fel-f (52))*					
	idPessoa	Nome	Endereco	Cidade	Estado	Email	Telefone
▶	1	Joao Fernandes Correia	Rua Jose Aristides	Divinopolis	Minas Gerais	JoaoFernandesCorreia@rhyta.com	37 4647-9323
	2	Kaua Azevedo Silva	Rua Faustino Siqueira Franco	Guarulhos	Sao Paulo	KauaAzevedoSilva@armyspy.com	11 4681-5575
	3	Heilig-Meyers	Estrada do Recanto 161	Timoteo	Minas Gerais	Heilig-Meyers@teleworm.us	31 5667-6663
	4	Grupo Técnico Jardins	Praça Presidente Gamal Abdel Nasser 1403	Sao Paulo	Sao Paulo	Henceall@dayrep.com	11 8783-6448
•	NULL	NULL	NULL	NULL	NULL	NULL	NULL

Figura 2. Preenchimento da tabela de Pessoas.

Felps-Notebook.Loja - dbo.PessoaFisica			Felps-Notebook.Loja...dbo.PessoaJuridica		
	FK_Pessoa_id	CPF		FK_Pessoa_id	CNPJ
▶	1	365.410.330-80	▶	3	95.823.031/0001-59
	2	78349632011		4	67652009000158
•	NULL	NULL	•	NULL	NULL

Figura 3. Preenchimento das tabelas de Pessoa Física e Jurídica.

Felps-Notebook.Loja - dbo.Usuario			
	idUsuario	Logim	Senha
▶	1	CorreiaJF	hDqHej8
	2	KakuaA	Gjyakyjav
	3	HM.Estoque	hgd6au28
	4	LojaGTJ	senha123
•	NULL	NULL	NULL

Figura 4. Preenchimento da tabela de Usuários.

Felps-Notebook.Loja - dbo.Produto		SQLQuery1.sql - fe...OTEBOOK(fel-f (52))*		
	idProduto	Nome	Quantidade	PrecoVenda
▶	1	Placa Mae	250	790,00
	2	Processador	150	850,00
	3	Placa de Video	100	1150,00
	4	Fonte de Alimentacao	300	650,00
	5	Memoria RAM	350	350,00
	6	Monitor	200	590,00
	7	Teclado	500	150,00
	8	Mouse	500	90,00
•	NULL	NULL	NULL	NULL

Figura 5. Preenchimento da tabela de Produtos.

Felps-Notebook.Loja - dbo.Movimento - SQLQuery1.sql - fe...OTEBOOK\fel-f (52))*							
	idMovimento	FK_Usuario_id	FK_Pessoa_id	FK_Produto_id	Tipo	Quantidade	PrecoUnitario
▶	1	1	1	4	E	2	650,00
	2	1	1	4	S	1	700,00
	3	2	2	5	S	2	450,00
	4	3	3	1	E	50	790,00
	5	3	3	2	E	50	850,00
	6	3	3	1	S	3	840,00
	7	3	3	3	S	2	1300,00
	8	4	4	6	E	25	590,00
	9	4	4	7	S	7	169,90
	10	4	4	8	S	9	109,90
*	NULL	NULL	NULL	NULL	NULL	NULL	NULL

Figura 6. Preenchimento da tabela de Movimentos.

SQLQuery1.sql - fe...OTEBOOK\fel-f (52))*								
<pre> SELECT Pe.*, PF.CPF FROM Pessoa Pe JOIN PessoaFisica PF ON Pe.idPessoa = PF.FK_Pessoa_id; </pre>								
100 %								
Results	Messages							
	idPessoa	Nome	Endereco	Cidade	Estado	Email	Telefone	CPF
1	1	Joao Fernandes Correia	Rua Jose Aristides	Divinopolis	Minas Gerais	JoaoFernandesCorreia@rhyta.com	37 4647-9323	365.410.330-80
2	2	Kaua Azevedo Silva	Rua Faustino Siqueira Franco	Guarulhos	Sao Paulo	KauaAzevedoSilva@amyspy.com	11 4681-5575	78349632011

Figura 7. Resultados da consulta de dados letra A.

SQLQuery1.sql - fe...OTEBOOK\fel-f (52))*								
<pre> SELECT Pe.*, PJ.CNPJ FROM Pessoa Pe JOIN PessoaJuridica PJ ON Pe.idPessoa = PJ.FK_Pessoa_id; </pre>								
100 %								
Results	Messages							
	idPessoa	Nome	Endereco	Cidade	Estado	Email	Telefone	CNPJ
1	3	Heilig-Meyers	Estrada do Recanto 161	Timoteo	Minas Gerais	Heilig-Meyers@telewom.us	31 5667-6663	95.823.031/0001-59
2	4	Grupo Técnico Jardins	Praça Presidente Gamal Abdel Nasser 1403	Sao Paulo	Sao Paulo	Henceall@dayrep.com	11 8783-6448	67652009000158

Figura 8. Resultados da consulta de dados letra B.

SQLQuery2.sql - fe...OTEBOOK\fel-f (54))*							
<pre> SELECT idMovimento, Tipo, pe.Nome AS Fornecedor, pr.Nome AS Produto, Mov.Quantidade, Mov.PrecoUnitario, (Mov.Quantidade * Mov.PrecoUnitario) AS Tot FROM Movimento Mov INNER JOIN Pessoa pe ON pe.idPessoa = Mov.FK_Pessoa_id INNER JOIN Produto pr ON pr.idProduto = Mov.FK_Produto_id WHERE Mov.Tipo = 'E'; </pre>							
100 %							
Results	Messages						
	idMovimento	Tipo	Fornecedor	Produto	Quantidade	PrecoUnitario	Total
1	1	E	Joao Fernandes Correia	Fonte de Alimentacao	2	650.00	1300.00
2	4	E	Heilig-Meyers	Placa Mae	50	790.00	39500.00
3	5	E	Heilig-Meyers	Processador	50	850.00	42500.00
4	8	E	Grupo Técnico Jardins	Monitor	25	590.00	14750.00

Figura 9. Resultados da consulta de dados letra C.

SQLQuery2.sql - fe...OTEBOOK\fel-f (54))* ✕

```

SELECT Mov.idMovimento, Tipo, pe.Nome AS Fornecedor, pr.Nome AS Produto, Mov.Quantidade, Mov.PrecoUnitario, (Mov.Quantidade * Mov.PrecoUnitario) AS
FROM Movimento Mov
INNER JOIN Pessoa pe ON pe.idPessoa = Mov.FK_Pessoa_id
INNER JOIN Produto pr ON pr.idProduto = Mov.FK_Produto_id
WHERE Mov.Tipo = 'S';

```

100 %

Results Messages

	idMovimento	Tipo	Fornecedor	Produto	Quantidade	PrecoUnitario	Total
1	2	S	Joao Fernandes Correia	Fonte de Alimentacao	1	700.00	700.00
2	3	S	Kaua Azevedo Silva	Memoria RAM	2	450.00	900.00
3	6	S	Heilig-Meyers	Placa Mae	3	840.00	2520.00
4	7	S	Heilig-Meyers	Placa de Video	2	1300.00	2600.00
5	9	S	Grupo Técnico Jardins	Teclado	7	169.90	1189.30
6	10	S	Grupo Técnico Jardins	Mouse	9	109.90	989.10

Figura 10. Resultados da consulta de dados letra D.

SQLQuery2.sql - fe...OTEBOOK\fel-f (54))* ✕

```

SELECT pr.Nome, SUM(Mov.Quantidade * Mov.PrecoUnitario) AS Compras
FROM Movimento Mov
INNER JOIN Produto pr ON Mov.FK_Produto_id = pr.idProduto
WHERE Mov.Tipo = 'E'
GROUP BY pr.Nome;

```

100 %

Results Messages

	Nome	Compras
1	Fonte de Alimentacao	1300.00
2	Monitor	14750.00
3	Placa Mae	39500.00
4	Processador	42500.00

Figura 11. Resultados da consulta de dados letra E.

SQLQuery2.sql - fe...OTEBOOK\fel-f (54))* ✕

```

SELECT pr.Nome, SUM(Mov.Quantidade * Mov.PrecoUnitario) AS Vendas
FROM Movimento Mov
INNER JOIN Produto pr ON Mov.FK_Produto_id = pr.idProduto
WHERE Mov.Tipo = 'S'
GROUP BY pr.Nome;

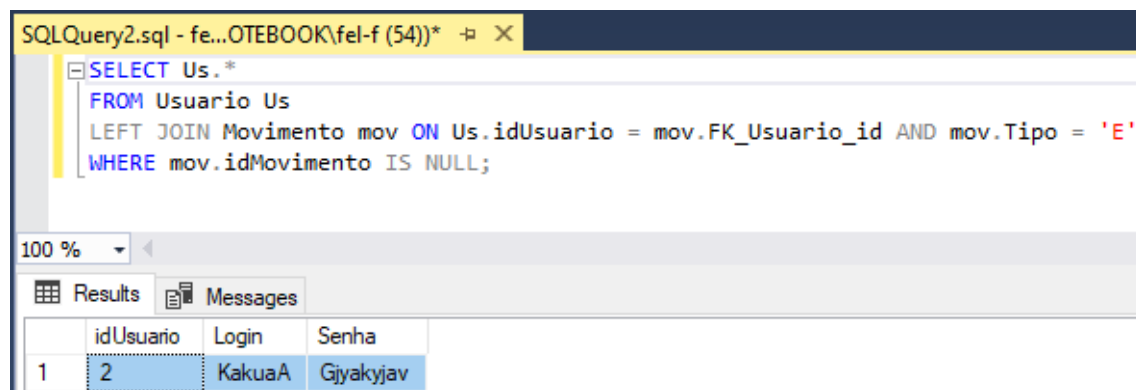
```

100 %

Results Messages

	Nome	Vendas
1	Fonte de Alimentacao	700.00
2	Memoria RAM	900.00
3	Mouse	989.10
4	Placa de Video	2600.00
5	Placa Mae	2520.00
6	Teclado	1189.30

Figura 12. Resultados da consulta de dados letra F.



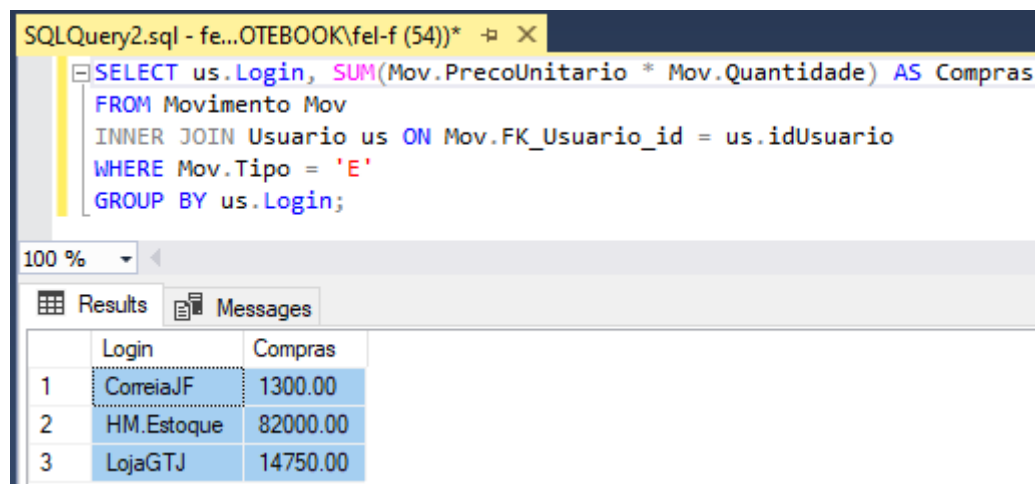
The screenshot shows a SQL query window titled 'SQLQuery2.sql - fe...OTEBOOK\fel-f (54))*'. The query is as follows:

```
SELECT Us.*
FROM Usuario Us
LEFT JOIN Movimento mov ON Us.idUsuario = mov.FK_Usuario_id AND mov.Tipo = 'E'
WHERE mov.idMovimento IS NULL;
```

Below the query, the 'Results' tab is active, displaying a table with 4 columns: idUsuario, Login, Senha, and an unlabeled column. The data is as follows:

	idUsuario	Login	Senha	
1	2	KakuaA	Gjyakyjav	

Figura 13. Resultados da consulta de dados letra G.



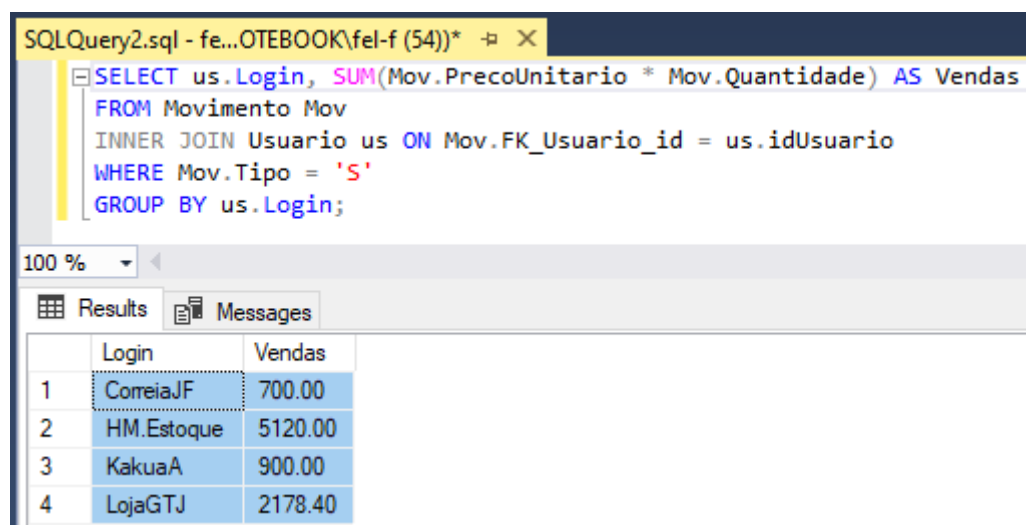
The screenshot shows a SQL query window titled 'SQLQuery2.sql - fe...OTEBOOK\fel-f (54))*'. The query is as follows:

```
SELECT us.Login, SUM(Mov.PrecoUnitario * Mov.Quantidade) AS Compras
FROM Movimento Mov
INNER JOIN Usuario us ON Mov.FK_Usuario_id = us.idUsuario
WHERE Mov.Tipo = 'E'
GROUP BY us.Login;
```

Below the query, the 'Results' tab is active, displaying a table with 3 columns: Login, Compras, and an unlabeled column. The data is as follows:

	Login	Compras	
1	CorreiaJF	1300.00	
2	HM.Estoque	82000.00	
3	LojaGTJ	14750.00	

Figura 14. Resultados da consulta de dados letra H.



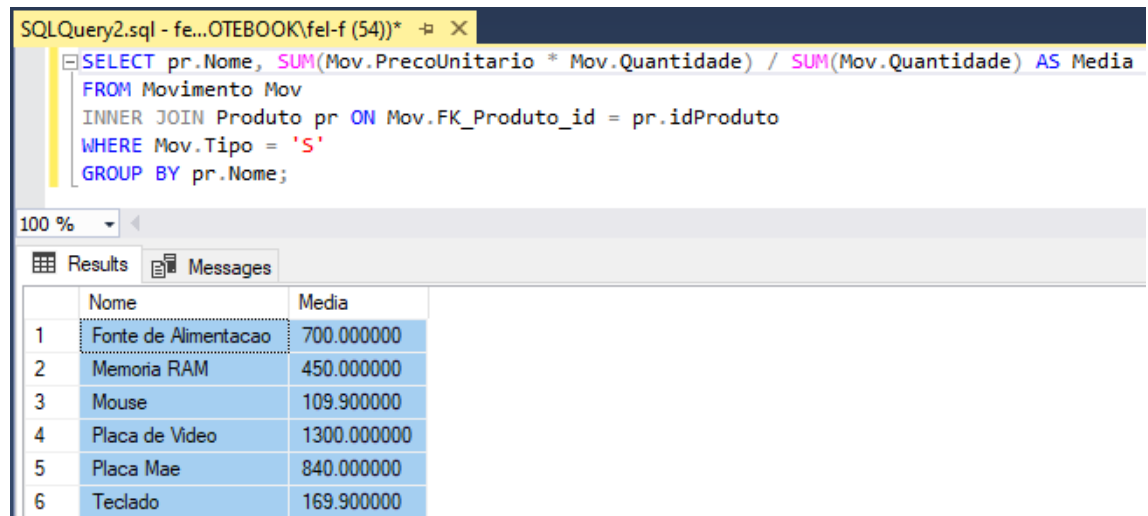
The screenshot shows a SQL query window titled 'SQLQuery2.sql - fe...OTEBOOK\fel-f (54))*'. The query is as follows:

```
SELECT us.Login, SUM(Mov.PrecoUnitario * Mov.Quantidade) AS Vendas
FROM Movimento Mov
INNER JOIN Usuario us ON Mov.FK_Usuario_id = us.idUsuario
WHERE Mov.Tipo = 'S'
GROUP BY us.Login;
```

Below the query, the 'Results' tab is active, displaying a table with 3 columns: Login, Vendas, and an unlabeled column. The data is as follows:

	Login	Vendas	
1	CorreiaJF	700.00	
2	HM.Estoque	5120.00	
3	KakuaA	900.00	
4	LojaGTJ	2178.40	

Figura 15. Resultados da consulta de dados letra I.



The screenshot shows a SQL query window with the following text:

```
SQLQuery2.sql - fe...OTEBOOK\fel-f (54))* -p X
SELECT pr.Nome, SUM(Mov.PrecoUnitario * Mov.Quantidade) / SUM(Mov.Quantidade) AS Media
FROM Movimento Mov
INNER JOIN Produto pr ON Mov.FK_Produto_id = pr.idProduto
WHERE Mov.Tipo = 'S'
GROUP BY pr.Nome;
```

Below the query, the 'Results' tab is active, displaying a table with 2 columns: 'Nome' and 'Media'. The table contains 6 rows of data:

	Nome	Media
1	Fonte de Alimentacao	700.000000
2	Memoria RAM	450.000000
3	Mouse	109.900000
4	Placa de Video	1300.000000
5	Placa Mae	840.000000
6	Teclado	169.900000

Figura 16. Resultados da consulta de dados letra J.

Análise e Conclusão – Prática 1

A) Como são implementadas as diferentes cardinalidades, basicamente 1X1, 1XN ou NxN, em um banco de dados relacional?

1x1: É implementado entre tabelas através de uma *foreign key*, que é associada a uma (E somente uma, em 1x1) *primary key* de outra tabela.

1xN: É implementado de maneira similar a 1x1, envolvendo *primary keys* e *foreign keys*. Entretanto, nesse caso, uma *primary key* teria associação com mais de uma *foreign key*.

NxN: Diferentemente das demais cardinalidades, essa implementação faz uso de tabelas de associação. As *foreign keys* nas tabelas de associação estão associadas a elementos de outras tabelas, dessa forma, a relação é indireta.

B) Que tipo de relacionamento deve ser utilizado para representar o uso de herança em bancos de dados relacionais?

Em bancos de dados relacionais usa-se a hierarquia para organizar as tabelas, especificamente uma relação de generalização e especialização. Com isso, tabelas podem ser organizadas mais facilmente. Utilizando como exemplo as classes utilizadas neste trabalho, uma Pessoa (superclasse, mais genérica) pode ser uma Pessoa Física ou Jurídica (subclasses, mais específicas). Em termos técnicos, herança é implementada em bancos de dados relacionais através do uso de *primary* e *foreign keys*.

C) Como o SQL Server Management Studio permite a melhoria da produtividade nas tarefas relacionadas ao gerenciamento do banco de dados?

Uma das principais vantagens do software é o fato de ser gratuito e, ainda assim, de alta qualidade. O SSMS possui ainda diversas ferramentas que facilitam a automação, monitoramento, segurança dos dados, entre outros. Deve-se destacar também a correção em tempo real do que é digitado no *query* do programa, que juntamente com a presença de uma interface gráfica, facilita incrivelmente a vida do programador, aumentando sua produtividade.

Análise e Conclusão – Prática 2

A) Quais as diferenças no uso de *sequence* e *identity*?

Ambas podem ser usadas para gerar numeração automática que pode ser incrementada de acordo com a necessidade. Entretanto, *identity* é uma propriedade que fica associada a uma tabela, enquanto *sequence* é uma propriedade independente, que pode ser utilizado em várias tabelas. Cada uma possui, portanto, seu caso de uso.

B) Qual a importância das chaves estrangeiras para a consistência do banco?

Funcionalmente, *foreign keys* são conexões entre tabelas de um banco de dados. Essa conexão facilita o gerenciamento de bancos de dados pois automaticamente mantém a consistência dos dados mesmo quando são feitas operações sobre estes, sem a necessidade de o programador perder muito tempo checando e recheando os dados.

C) Quais operadores do SQL pertencem à álgebra relacional e quais são definidos no cálculo relacional?

Os operadores de álgebra relacional utilizados no SQL incluem: *selection*, *projection*, *union*, *intersection*, *difference*, *cartesian product*, *join*, *renaming*. SQL tem maior alinhamento com álgebra relacional do que cálculo relacional, mas ainda assim existem algumas aplicações deste como cálculo de tuplas e de domínios.

D) Como é feito o agrupamento em consultas, e qual requisito é obrigatório?

O agrupamento em consultas se faz através do uso de GROUP BY, que agrupa linhas que tem valores iguais em colunas específicas. Alguns requisitos são obrigatórios para o uso dessa técnica. Primeiramente, o uso de funções de agregação (como COUNT, SUM, AVG, MAX, MIN) que serão usadas para calcular o valor de cada grupo. Também é essencial que todas as colunas declaradas em SELECT que não estão em funções de agregação devem estar dentro da clausula GROUP BY.

Anexo – Códigos do Projeto

Arquivo – New model.sql

```
CREATE DATABASE Loja
```

```
CREATE LOGIN loja WITH PASSWORD = 'loja', CHECK_POLICY = OFF
```

```
CREATE USER loja FOR LOGIN loja
```

```
USE Loja
```

```
CREATE SEQUENCE listaPessoa AS INTEGER START WITH 1  
INCREMENTED BY 1;
```

```
CREATE TABLE [Pessoa] (  
    idPessoa int NOT NULL,  
    Nome varchar(100) NOT NULL,  
    Endereco varchar(200) NOT NULL,  
    Cidade varchar(50) NOT NULL,  
    Estado varchar(50) NOT NULL,  
    Email varchar(255) NOT NULL,  
    Telefone varchar(15) NOT NULL  
    CONSTRAINT PK_Pessoa PRIMARY KEY CLUSTERED (idPessoa)  
)
```

```
CREATE TABLE [PessoaFisica] (  
    FK_Pessoa_id int NOT NULL,  
    CPF varchar(14) NOT NULL,  
    CONSTRAINT PK_PessoaFisica PRIMARY KEY CLUSTERED  
(FK_Pessoa_id),  
    CONSTRAINT FK_Pessoa_Fisica FOREIGN KEY (FK_Pessoa_id)  
REFERENCES Pessoa(idPessoa)  
    ON UPDATE CASCADE ON DELETE CASCADE,  
)
```

```
CREATE TABLE [PessoaJuridica] (  
    FK_Pessoa_id int NOT NULL,  
    CNPJ varchar(18) NOT NULL,  
    CONSTRAINT PK_PessoaJuridica PRIMARY KEY CLUSTERED  
(FK_Pessoa_id),  
    CONSTRAINT FK_Pessoa_Juridica FOREIGN KEY (FK_Pessoa_id)  
REFERENCES Pessoa(idPessoa)
```

```

        ON UPDATE CASCADE ON DELETE CASCADE,
    )

CREATE TABLE [Produto] (
    idProduto int NOT NULL,
    Nome varchar(255) NOT NULL,
    Quantidade int NOT NULL,
    PrecoVenda decimal(25,2) NOT NULL,
    CONSTRAINT PK_Produto PRIMARY KEY CLUSTERED (idProduto)
)

CREATE TABLE [Usuario] (
    idUsuario int NOT NULL,
    Login varchar(25) NOT NULL,
    Senha varchar(25) NOT NULL,
    CONSTRAINT PK_Usuario PRIMARY KEY CLUSTERED (idUsuario)
)

CREATE TABLE [Movimento] (
    idMovimento int NOT NULL UNIQUE,
    FK_Usuario_id int NOT NULL,
    FK_Pessoa_id int NOT NULL,
    FK_Produto_id int NOT NULL,
    Tipo varchar(255) NOT NULL,
    Quantidade int NOT NULL,
    PrecoUnitario decimal(25,2) NOT NULL,
    CONSTRAINT PK_Movimento PRIMARY KEY CLUSTERED (idMovimento),
    CONSTRAINT FK_Usuario FOREIGN KEY (FK_Usuario_id) REFERENCES
Usuario(idUsuario)
        ON UPDATE CASCADE ON DELETE CASCADE,
    CONSTRAINT FK_Pessoa FOREIGN KEY (FK_Pessoa_id) REFERENCES
Pessoa(idPessoa)
        ON UPDATE CASCADE ON DELETE CASCADE,
    CONSTRAINT FK_Produto FOREIGN KEY (FK_Produto_id) REFERENCES
Produto(idProduto)
        ON UPDATE CASCADE ON DELETE CASCADE
)

```

Arquivo – New content.sql

USE Loja;

```
INSERT INTO [Pessoa] (idPessoa, Nome, Endereco, Cidade, Estado, Email,
Telefone)
```

```
VALUES
```

```
    (NEXT VALUE FOR listaPESSOA, 'Joao Fernandes Correia', 'Rua Jose
Aristides', 'Divinopolis', 'Minas Gerais', 'JoaoFernandesCorreia@rhyta.com', '37
4647-9323'),
```

```
    (NEXT VALUE FOR listaPESSOA, 'Kaua Azevedo Silva', 'Rua Faustino
Siqueira Franco', 'Guarulhos', 'Sao Paulo', 'KauaAzevedoSilva@armyspy.com',
'11 4681-5575'),
```

```
    (NEXT VALUE FOR listaPESSOA, 'Heilig-Meyers', 'Estrada do Recanto
161', 'Timoteo', 'Minas Gerais', 'Heilig-Meyers@teleworm.us', '31 5667-6663'),
```

```
    (NEXT VALUE FOR listaPESSOA, 'Grupo Técnico Jardins', 'Praça
Presidente Gamal Abdel Nasser 1403', 'Sao Paulo', 'Sao Paulo',
'Henceall@dayrep.com', '11 8783-6448');
```

```
INSERT INTO [PessoaFisica] (FK_Pessoa_id, CPF)
```

```
VALUES
```

```
    (1, '365.410.330-80'),
```

```
    (2, '78349632011');
```

```
INSERT INTO [PessoaJuridica] (FK_Pessoa_id, CNPJ)
```

```
VALUES
```

```
    (3, '95.823.031/0001-59'),
```

```
    (4, '67652009000158');
```

```
INSERT INTO [Usuario] (idUsuario, Login, Senha)
```

```
VALUES
```

```
    (1, 'CorreiaJF', 'hDq$@j8'),
```

```
    (2, 'KakuaA', 'Gjyakyjav'),
```

```
    (3, 'HM.Estoque', 'hgd6au28'),
```

```
    (4, 'LojaGTJ', 'senha123');
```

```
INSERT INTO [Produto] (idProduto, Nome, Quantidade, PrecoVenda)
VALUES
```

```
(1, 'Placa Mae', 250, 790.00),
(2, 'Processador', 150, 850.00),
(3, 'Placa de Video', 100, 1150.00),
(4, 'Fonte de Alimentacao', 300, 650.00),
(5, 'Memoria RAM', 350, 350.00),
(6, 'Monitor', 200, 590.00),
(7, 'Teclado', 500, 150.00),
(8, 'Mouse', 500, 90.00);
```

```
INSERT INTO [Movimento] (idMovimento, FK_Usuario_id, FK_Pessoa_id,
FK_Produto_id, Tipo, Quantidade, PrecoUnitario)
```

```
VALUES
```

```
(1, 1, 1, 4, 'E', 2, 650.00),
(2, 1, 1, 4, 'S', 1, 700.00),
(3, 2, 2, 5, 'S', 2, 450.00),
(4, 3, 3, 1, 'E', 50, 790.00),
(5, 3, 3, 2, 'E', 50, 850.00),
(6, 3, 3, 1, 'S', 3, 840.00),
(7, 3, 3, 3, 'S', 2, 1300.00),
(8, 4, 4, 6, 'E', 25, 590.00),
(9, 4, 4, 7, 'S', 7, 169.90),
(10, 4, 4, 8, 'S', 9, 109.90);
```

Arquivo – New search.sql

```
SELECT Pe.*, PF.CPF
```

```
FROM Pessoa Pe
```

```
JOIN PessoaFisica PF ON Pe.idPessoa = PF.FK_Pessoa_id;
```

```
SELECT Pe.*, PJ.CNPJ
```

```
FROM Pessoa Pe
```

```
JOIN PessoaJuridica PJ ON Pe.idPessoa = PJ.FK_Pessoa_id;
```

```
SELECT Mov.idMovimento, Tipo, pe.Nome AS Fornecedor, pr.Nome AS  
Produto, Mov.Quantidade, Mov.PrecoUnitario, (Mov.Quantidade *  
Mov.PrecoUnitario) AS Total
```

```
FROM Movimento Mov
```

```
INNER JOIN Pessoa pe ON pe.idPessoa = Mov.FK_Pessoa_id
```

```
INNER JOIN Produto pr ON pr.idProduto = Mov.FK_Produto_id
```

```
WHERE Mov.Tipo = 'E';
```

```
SELECT Mov.idMovimento, Tipo, pe.Nome AS Fornecedor, pr.Nome AS  
Produto, Mov.Quantidade, Mov.PrecoUnitario, (Mov.Quantidade *  
Mov.PrecoUnitario) AS Total
```

```
FROM Movimento Mov
```

```
INNER JOIN Pessoa pe ON pe.idPessoa = Mov.FK_Pessoa_id
```

```
INNER JOIN Produto pr ON pr.idProduto = Mov.FK_Produto_id
```

```
WHERE Mov.Tipo = 'S';
```

```
SELECT pr.Nome, SUM(Mov.Quantidade * Mov.PrecoUnitario) AS Compras
```

```
FROM Movimento Mov
```

```
INNER JOIN Produto pr ON Mov.FK_Produto_id = pr.idProduto
```

```
WHERE Mov.Tipo = 'E'
```

```
GROUP BY pr.Nome;
```

```
SELECT pr.Nome, SUM(Mov.Quantidade * Mov.PrecoUnitario) AS Vendas
```

```
FROM Movimento Mov
INNER JOIN Produto pr ON Mov.FK_Produto_id = pr.idProduto
WHERE Mov.Tipo = 'S'
GROUP BY pr.Nome;
```

```
SELECT Us.*
FROM Usuario Us
LEFT JOIN Movimento mov ON Us.idUsuario = mov.FK_Usuario_id AND
mov.Tipo = 'E'
WHERE mov.Movimento IS NULL;
```

```
SELECT us.Login, SUM(Mov.PrecoUnitario * Mov.Quantidade) AS Compras
FROM Movimento Mov
INNER JOIN Usuario us ON Mov.FK_Usuario_id = us.idUsuario
WHERE Mov.Tipo = 'E'
GROUP BY us.Login;
```

```
SELECT us.Login, SUM(Mov.PrecoUnitario * Mov.Quantidade) AS Vendas
FROM Movimento Mov
INNER JOIN Usuario us ON Mov.FK_Usuario_id = us.idUsuario
WHERE Mov.Tipo = 'S'
GROUP BY us.Login;
```

```
SELECT pr.Nome, SUM(Mov.PrecoUnitario * Mov.Quantidade) /
SUM(Mov.Quantidade) AS Media)
FROM Movimento Mov
INNER JOIN Produto pr ON Mov.FK_Produto_id = pr.idProduto
WHERE Mov.Tipo = 'S'
GROUP BY pr.Nome;
```