



# **Estácio**

**FACULDADE ESTÁCIO**

**CÂMPUS MANAUS – AM**

**DESENVOLVIMENTO FULL STACK**

**DISCIPLINA – VAMOS MANTER AS INFORMAÇÕES?**

**TURMA – 2023.2**

**SEMESTRE – 3**

**MANAUS, JUNHO 2024.**

**DESENVOLVIMENTO FULL STACK**

**DISCIPLINA – VAMOS MANTER AS INFORMAÇÕES?**

**TURMA – 2023.2**

**SEMESTRE – 3**

**ALUNO – ALEX BARROSO PAZ**

**TUTOR – JOSYANE SOUZA**

***GITHUB* - <https://github.com/finntroll89/Vamos-Manter-as-Informa-es-parte-1>**

**MANAUS, JUNHO 2024.**

## RESUMO

O código apresentado implementa um sistema de modelagem de dados utilizando o DBDesigner Fork e o SQL Server Management Studio. O sistema permite a criação de tabelas para cadastro de usuários, pessoas físicas e jurídicas, produtos, e movimentos de compra e venda.

Para começar, o DBDesigner Fork é baixado do endereço <https://sourceforge.net/projects/dbdesigner-fork/>, descompactado e executado conforme as instruções. No sistema, a tabela de usuários permite registrar operadores que terão acesso ao sistema e gerenciarão as operações de compra e venda de produtos. A tabela de pessoas físicas e jurídicas inclui dados básicos de identificação, localização e contato, diferenciando-se pelo uso de CPF ou CNPJ. A tabela de produtos registra identificador, nome, quantidade e preço de venda dos produtos.

Além disso, os operadores podem registrar movimentos de compra para adquirir produtos de pessoas jurídicas, especificando a quantidade e o preço unitário. Os movimentos de venda são registrados quando os operadores vendem produtos para pessoas físicas, utilizando o preço de venda atual registrado no sistema.

Utilizando o SQL Server Management Studio, o modelo de dados é implementado com a criação de tabelas, definição de chaves primárias e estrangeiras, e a configuração de uma sequence para geração dos identificadores de pessoa. O processo inclui o login com um usuário administrador, a criação de um novo usuário com permissões adequadas, e a execução de scripts SQL para criar a base de dados conforme modelada.

Palavras-chave: Ferramenta de Modelagem, DBDesigner Fork, SQL Server Management Studio, Cadastro de Usuários, Cadastro de Pessoas, Pessoas Físicas, Pessoas Jurídicas, Cadastro de Produtos, Movimentos de Compra, Movimentos de Venda, Identificação e Contato, Operadores do Sistema, Quantidade e Preço de Venda, Relacionamentos de Tabelas, Banco de Dados, Criação de Tabelas, Sistema de Cadastro, Gerenciamento de Dados.

## SUMÁRIO

<b>1 INTRODUÇÃO .....</b>	<b>5</b>
<b>2 DBDESIGNER FORK.....</b>	<b>6</b>
2.1 MODELAGEM.....	6
<b>3 SQL SERVER MANAGEMENT STUDIO.....</b>	<b>7</b>
3.1 estruturas do modelo.....	7
<b>4 códigos.....</b>	<b>8</b>
5.1 MOVIMENTOS VENDA.....	9
5.2 PESSOAS.....	10
5.3 PESSOAS FISICAS.....	11
5.4 PESSOAS JURIDICAS.....	12
5.5 PRODUTOS.....	12
5.6 USUÁRIOS.....	13
<b>6 ALIMENTANDO A BASE.....</b>	<b>14</b>
<b>12 ANALISE DOCUMENTO 1.....</b>	<b>18</b>
12.1 O Como são implementadas as diferentes cardinalidades, basicamente 1X1, 1XN ou NxN, em um banco de dados relacional?.....	18
12.2 Que tipo de relacionamento deve ser utilizado para representar o uso de herança em bancos de dados relacionais?.....	18
12.3 Como o SQL Server Management Studio permite a melhoria da produtividade nas tarefas relacionadas ao gerenciamento do banco de dados?.....	19
<b>13 ANALISE DOCUMENTO 2.....</b>	<b>19</b>
14 Quais as diferenças no uso de sequence e identity?.....	20
15 Qual a importância das chaves estrangeiras para a consistência do banco?.....	20
16 Quais operadores do SQL pertencem à álgebra relacional e quais são definidos no cálculo relacional?.....	21
17 Como é feito o agrupamento em consultas, e qual requisito é obrigatório?.....	21
<b>18 CONCLUSÃO.....</b>	<b>23</b>
<b>19 REFERÊNCIAS.....</b>	<b>24</b>

## ***1 INTRODUÇÃO***

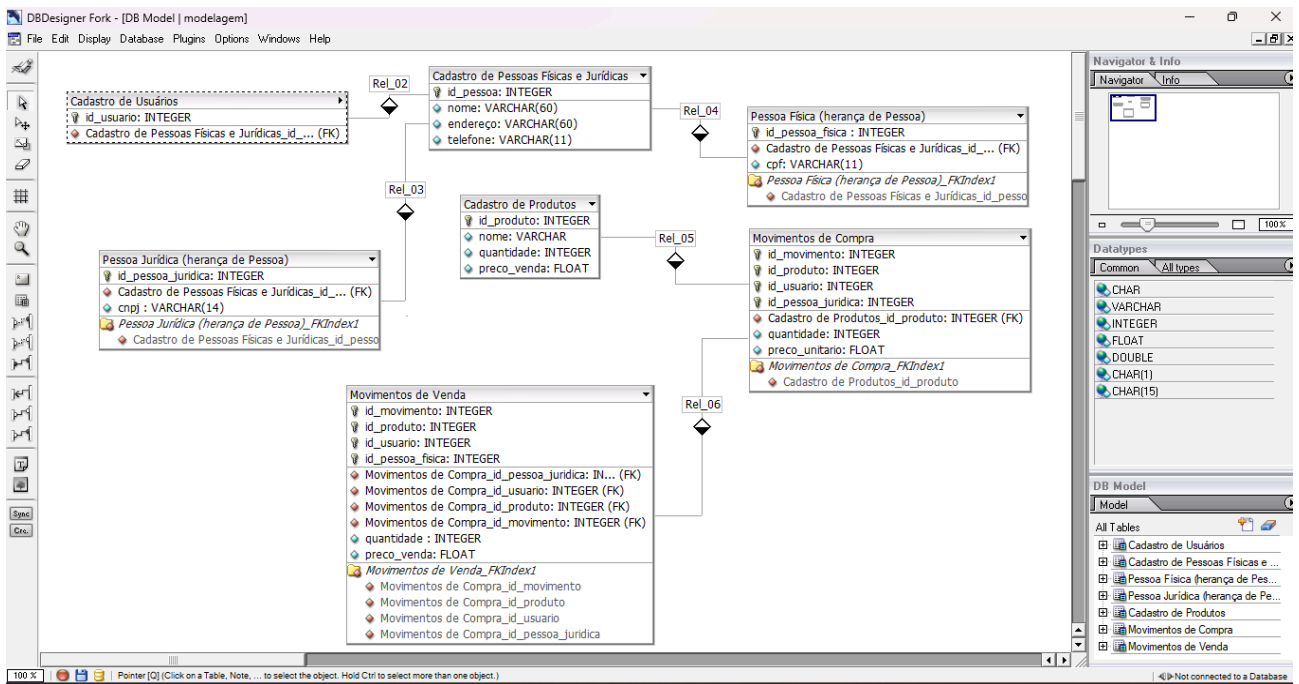
Este trabalho apresenta a criação de um modelo de dados para um sistema de cadastro utilizando a ferramenta de modelagem DBDesigner Fork e o SQL Server Management Studio. O sistema é projetado para gerenciar eficientemente usuários, pessoas físicas e jurídicas, produtos, e movimentos de compra e venda. A implementação foca em criar uma estrutura clara e organizada de tabelas e relações, facilitando o armazenamento e a manipulação de dados. A definição do modelo de dados abrange a criação de tabelas para cadastro de usuários, que atuam como operadores no sistema, e tabelas separadas para pessoas físicas e jurídicas, diferenciadas pelo uso de CPF e CNPJ. Adicionalmente, são criadas tabelas para produtos, registrando identificador, nome, quantidade e preço de venda.

Os movimentos de compra e venda são modelados para registrar transações efetuadas pelos operadores, garantindo que compras sejam associadas a pessoas jurídicas e vendas a pessoas físicas, mantendo a integridade dos dados e a clareza nas operações.

A utilização do DBDesigner Fork proporciona uma interface visual que facilita a definição e o gerenciamento dessas tabelas e suas relações, promovendo uma abordagem prática e intuitiva para a criação de sistemas de banco de dados. O SQL Server Management Studio é utilizado para implementar a base de dados modelada, incluindo a definição de uma sequence para geração dos identificadores de pessoa, e a criação das estruturas do modelo no editor de SQL. Este documento discutirá o processo de criação e configuração das tabelas, a definição das chaves primárias e estrangeiras, e a importância de um modelo de dados bem estruturado para a eficiência e manutenção do sistema.

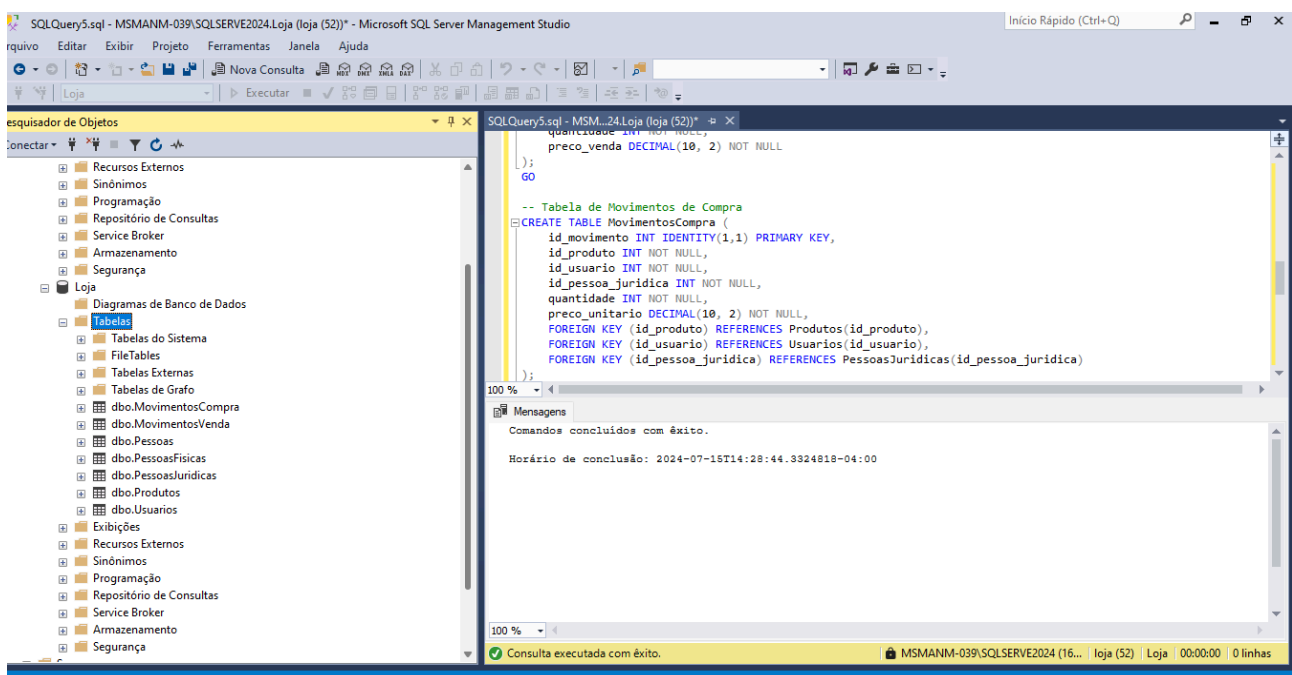
## 2 DBDESIGNER FORK

### 2.1 MODELAGEM



## 3 SQL SERVER MANAGEMENT STUDIO

### 3.1 ESTRUTURAS DO MODELO



## 4 CÓDIGOS

### 5 MOVIMENTOS COMPRA

USE [Loja]  
GO

```
/****** Object: Table [dbo].[MovimentosCompra]    Script Date: 23/07/2024 07:42:36 *****/
SET ANSI_NULLS ON
GO
```

```
SET QUOTED_IDENTIFIER ON
```

GO

```
CREATE TABLE [dbo].[MovimentosCompra](
    [id_movimento] [int] IDENTITY(1,1) NOT NULL,
    [id_produto] [int] NOT NULL,
    [id_fornecedor] [int] NOT NULL,
    [quantidade] [int] NOT NULL,
    [preco_unitario] [decimal](10, 2) NOT NULL,
    [data_movimento] [datetime] NOT NULL,
PRIMARY KEY CLUSTERED
(
    [id_movimento] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF, ALLOW_ROW_LOCKS
= ON, ALLOW_PAGE_LOCKS = ON, OPTIMIZE_FOR_SEQUENTIAL_KEY = OFF) ON [PRIMARY]
) ON [PRIMARY]
GO
```

```
ALTER TABLE [dbo].[MovimentosCompra] WITH CHECK ADD FOREIGN KEY([id_fornecedor])
REFERENCES [dbo].[PessoasJuridicas] ([id_pessoa_juridica])
GO
```

```
ALTER TABLE [dbo].[MovimentosCompra] WITH CHECK ADD FOREIGN KEY([id_produto])
REFERENCES [dbo].[Produtos] ([id_produto])
GO
```

## 5.1 MOVIMENTOS VENDA

```
USE [Loja]
GO
```

```
/****** Object: Table [dbo].[MovimentosVenda]    Script Date: 23/07/2024 07:43:31 *****/
SET ANSI_NULLS ON
GO
```

```
SET QUOTED_IDENTIFIER ON
GO
```

```
CREATE TABLE [dbo].[MovimentosVenda](
    [id_movimento] [int] IDENTITY(1,1) NOT NULL,
    [id_produto] [int] NOT NULL,
    [id_cliente] [int] NOT NULL,
    [quantidade] [int] NOT NULL,
    [preco_unitario] [decimal](10, 2) NOT NULL,
```



```

        [data_movimento] [datetime] NOT NULL,
PRIMARY KEY CLUSTERED
(
    [id_movimento] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF, ALLOW_ROW_LOCKS
= ON, ALLOW_PAGE_LOCKS = ON, OPTIMIZE_FOR_SEQUENTIAL_KEY = OFF) ON [PRIMARY]
) ON [PRIMARY]
GO

ALTER TABLE [dbo].[MovimentosVenda] WITH CHECK ADD FOREIGN KEY([id_cliente])
REFERENCES [dbo].[PessoasFisicas] ([id_pessoa_fisica])
GO

ALTER TABLE [dbo].[MovimentosVenda] WITH CHECK ADD FOREIGN KEY([id_produto])
REFERENCES [dbo].[Produtos] ([id_produto])
GO

```

## 5.2 PESSOAS

```

USE [Loja]
GO

/***** Object: Table [dbo].[Pessoas]    Script Date: 23/07/2024 07:44:07 *****/
SET ANSI_NULLS ON
GO

SET QUOTED_IDENTIFIER ON
GO

CREATE TABLE [dbo].[Pessoas](
    [id_pessoa] [int] IDENTITY(1,1) NOT NULL,
    [nome] [nvarchar](255) NOT NULL,
    [endereco] [nvarchar](255) NULL,
    [telefone] [nvarchar](20) NULL,
PRIMARY KEY CLUSTERED
(
    [id_pessoa] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF, ALLOW_ROW_LOCKS
= ON, ALLOW_PAGE_LOCKS = ON, OPTIMIZE_FOR_SEQUENTIAL_KEY = OFF) ON [PRIMARY]
) ON [PRIMARY]
GO

```

## 5.3 PESSOAS FISICAS

```

USE [Loja]

```

GO

/\*\*\*\*\* Object: Table [dbo].[PessoasFisicas] Script Date: 23/07/2024 07:44:51 \*\*\*\*\*/

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

CREATE TABLE [dbo].[PessoasFisicas](  
 [id\_pessoa\_fisica] [int] NOT NULL,  
 [cpf] [char](11) NOT NULL,  
 PRIMARY KEY CLUSTERED

(

[id\_pessoa\_fisica] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON, OPTIMIZE\_FOR\_SEQUENTIAL\_KEY = OFF) ON [PRIMARY],  
 UNIQUE NONCLUSTERED

(

[cpf] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON, OPTIMIZE\_FOR\_SEQUENTIAL\_KEY = OFF) ON [PRIMARY]  
) ON [PRIMARY]

GO

ALTER TABLE [dbo].[PessoasFisicas] WITH CHECK ADD FOREIGN KEY([id\_pessoa\_fisica])  
REFERENCES [dbo].[Pessoas] ([id\_pessoa])

GO

## 5.4 PESSOAS JURIDICAS

USE [Loja]

GO

/\*\*\*\*\* Object: Table [dbo].[PessoasJuridicas] Script Date: 23/07/2024 07:45:56 \*\*\*\*\*/

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

```
CREATE TABLE [dbo].[PessoasJuridicas](
    [id_pessoa_juridica] [int] NOT NULL,
    [cnpj] [char](14) NOT NULL,
PRIMARY KEY CLUSTERED
(
    [id_pessoa_juridica] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF, ALLOW_ROW_LOCKS
= ON, ALLOW_PAGE_LOCKS = ON, OPTIMIZE_FOR_SEQUENTIAL_KEY = OFF) ON [PRIMARY],
UNIQUE NONCLUSTERED
(
    [cnpj] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF, ALLOW_ROW_LOCKS
= ON, ALLOW_PAGE_LOCKS = ON, OPTIMIZE_FOR_SEQUENTIAL_KEY = OFF) ON [PRIMARY]
) ON [PRIMARY]
GO
```

```
ALTER TABLE [dbo].[PessoasJuridicas] WITH CHECK ADD FOREIGN KEY([id_pessoa_juridica])
REFERENCES [dbo].[Pessoas] ([id_pessoa])
GO
```

## 5.5 PRODUTOS

```
USE [Loja]
GO
```

```
/****** Object: Table [dbo].[Produtos]    Script Date: 23/07/2024 07:46:32 *****/
SET ANSI_NULLS ON
GO
```

```
SET QUOTED_IDENTIFIER ON
GO
```

```
CREATE TABLE [dbo].[Produtos](
    [id_produto] [int] IDENTITY(1,1) NOT NULL,
    [nome] [nvarchar](100) NOT NULL,
    [preco] [decimal](10, 2) NOT NULL,
PRIMARY KEY CLUSTERED
(
    [id_produto] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF, ALLOW_ROW_LOCKS
= ON, ALLOW_PAGE_LOCKS = ON, OPTIMIZE_FOR_SEQUENTIAL_KEY = OFF) ON [PRIMARY]
) ON [PRIMARY]
```

GO

## 5.6 USUÁRIOS

```
USE [Loja]
```

GO

```
/****** Object: Table [dbo].[Usuarios]    Script Date: 23/07/2024 07:47:09 *****/
```

```
SET ANSI_NULLS ON
```

GO

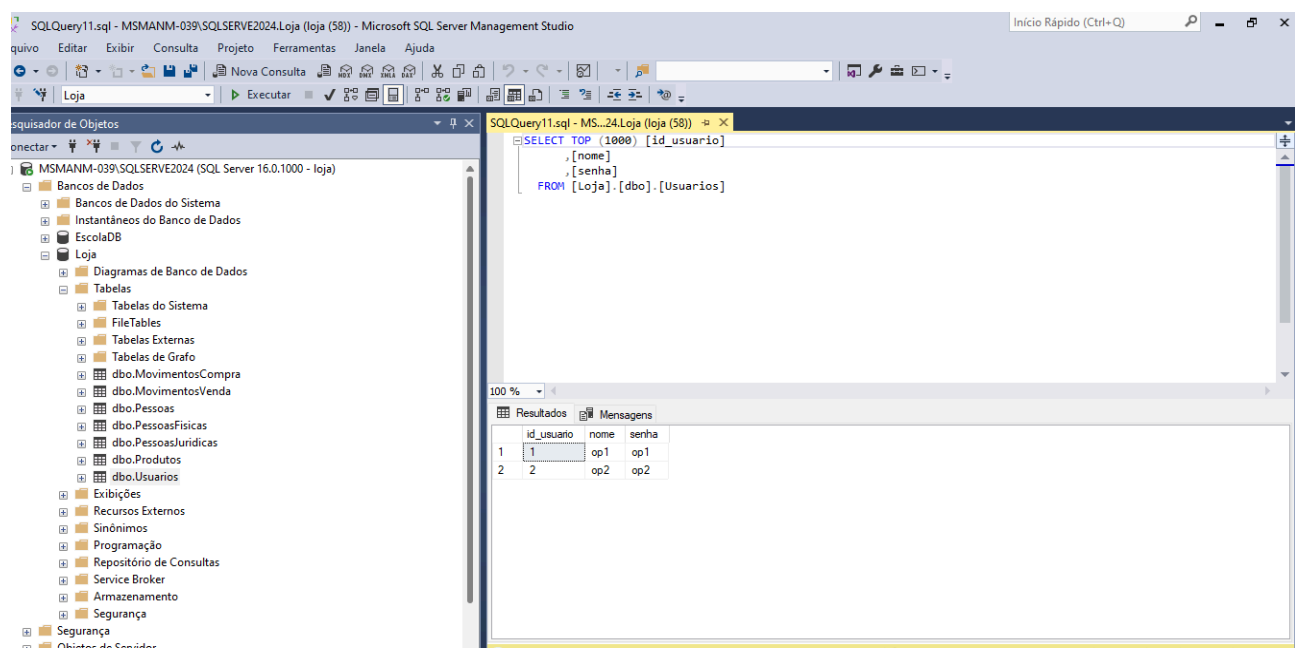
```
SET QUOTED_IDENTIFIER ON
```

GO

```
CREATE TABLE [dbo].[Usuarios](
    [id_usuario] [int] IDENTITY(1,1) NOT NULL,
    [nome] [nvarchar](100) NOT NULL,
    [senha] [nvarchar](100) NOT NULL,
    PRIMARY KEY CLUSTERED
(
    [id_usuario] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF, ALLOW_ROW_LOCKS
= ON, ALLOW_PAGE_LOCKS = ON, OPTIMIZE_FOR_SEQUENTIAL_KEY = OFF) ON [PRIMARY]
) ON [PRIMARY]
GO
```

## 6 ALIMENTANDO A BASE

### 7 OPI, OP2



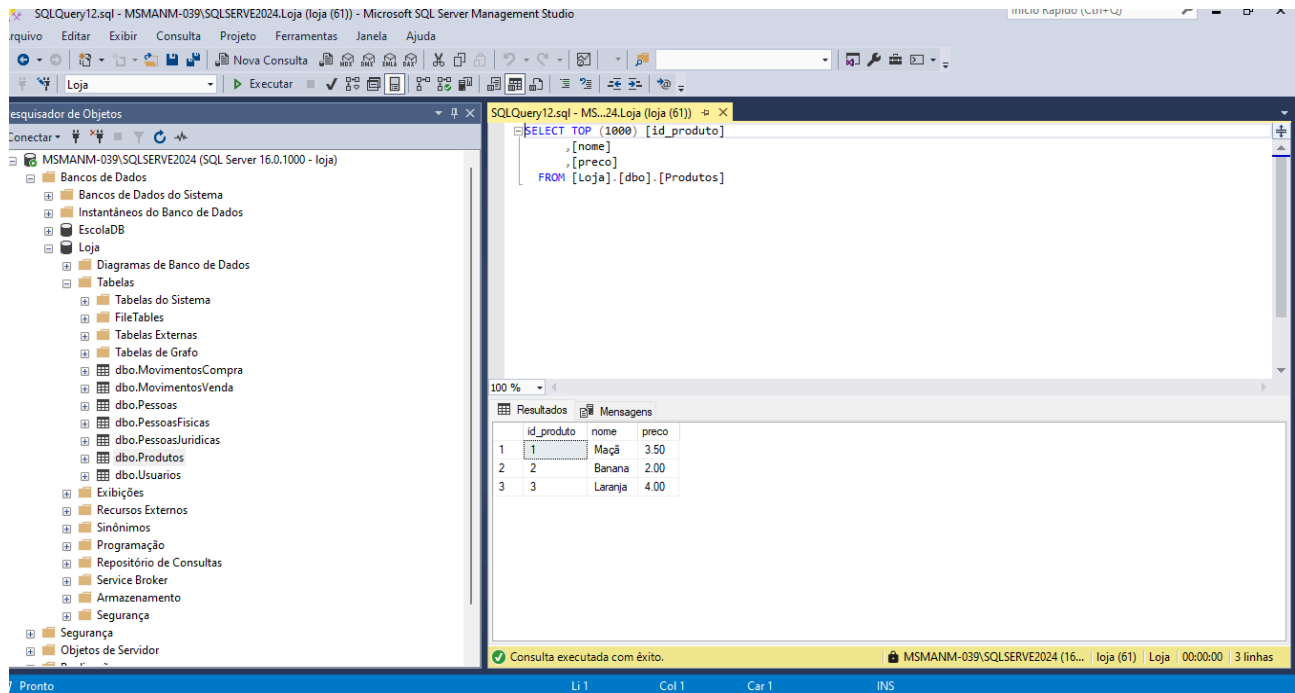
The screenshot shows the Microsoft SQL Server Management Studio interface. The left pane displays the 'Explorador de Objetos' (Object Explorer) with the 'Loja' database selected. The right pane shows a SQL query window with the following query:

```
SELECT TOP (1000) [id_usuario]
, [nome]
, [senha]
FROM [Loja].[dbo].[Usuarios]
```

Below the query window, the 'Resultados' (Results) pane displays the following data:

	id_usuario	nome	senha
1	1	op1	op1
2	2	op2	op2

## 8 PRODUTOS



The screenshot displays the Microsoft SQL Server Management Studio interface. The left pane shows the 'Esquema de Objetos' (Object Explorer) for the 'Loja' database. The central pane contains a SQL query: `SELECT TOP (1000) [id_produto], [nome], [preco] FROM [Loja].[dbo].[Produtos]`. The bottom pane shows the 'Resultados' (Results) tab with a table of 3 rows and 3 columns: `id_produto`, `nome`, and `preco`. The status bar at the bottom indicates 'Consulta executada com êxito.' (Query executed successfully).

	id_produto	nome	preco
1	1	Maça	3.50
2	2	Banana	2.00
3	3	Laranja	4.00

## 9 PESSOA FISICA E JURIDICA

SQLQuery13.sql - MSMANM-039\SQLSERVE2024.Loja (loja (61)) - Microsoft SQL Server Management Studio

Arquivo Editar Exibir Consultar Projeto Ferramentas Janela Ajuda

Loja

SQLQuery13.sql - MS...24.Loja (loja (61))

```
SELECT TOP (1000) [id_pessoa_fisica]
, [cpf]
FROM [Loja].[dbo].[PessoasFisicas]
```

100 %

	id_pessoa_fisica	cpf
1	1	12345678901
2	2	98765432101

Consultas executadas com êxito. MSMANM-039\SQLSERVE2024 (16... | loja (61) | Loja | 00:00:00 | 2 linhas

SQLQuery14.sql - MSMANM-039\SQLSERVE2024.Loja (loja (66)) - Microsoft SQL Server Management Studio

Arquivo Editar Exibir Consultar Projeto Ferramentas Janela Ajuda

Loja

SQLQuery14.sql - MS...24.Loja (loja (66))

```
SELECT TOP (1000) [id_pessoa_juridica]
, [cnpj]
FROM [Loja].[dbo].[PessoasJuridicas]
```

100 %

	id_pessoa_juridica	cnpj
1	3	12345678000123

Consultas executadas com êxito. MSMANM-039\SQLSERVE2024 (16... | loja (66) | Loja | 00:00:00 | 1 linhas

SQLQuery16.sql - MSMANM-039\SQLSERVE2024.Loja (loja (62)) - Microsoft SQL Server Management Studio

Arquivo Editar Exibir Consulta Projeto Ferramentas Janela Ajuda

Loja

Esquema de Objetos

MSMANM-039\SQLSERVE2024 (SQL Server 16.0.1000 - loja)

Bancos de Dados

Bancos de Dados do Sistema

Instantâneos do Banco de Dados

EscolaDB

Loja

Diagramas de Banco de Dados

Tabelas

Tabelas do Sistema

FileTables

Tabelas Externas

Tabelas de Grafo

dbo.MovimentosCompra

dbo.MovimentosVenda

dbo.Pessoas

dbo.PessoasFisicas

dbo.PessoasJuridicas

dboProdutos

dbo.Usuarios

Exibições

Recursos Externos

Sinônimos

Programação

Repositório de Consultas

Service Broker

Armazenamento

Segurança

Objetos de Servidor

SQLQuery16.sql - MS...24.Loja (loja (62))

```
SELECT TOP (1000) [id_movimento]
,[id_produto]
,[id_fornecedor]
,[quantidade]
,[preco_unitario]
,[data_movimento]
FROM [Loja].[dbo].[MovimentosCompra]
```

100 %

Resultados Mensagens

	id_movimento	id_produto	id_fornecedor	quantidade	preco_unitario	data_movimento
1	1	1	3	100	3.00	2024-07-22 21:26:12.547
2	2	2	3	200	1.50	2024-07-22 21:26:12.547
3	3	3	3	150	3.50	2024-07-22 21:26:12.547

Consulta executada com êxito. MSMANM-039\SQLSERVE2024 (16... loja (62)) Loja 00:00:00 3 linhas

Pronto Li 1 Col 1 Car 1 INS

## 11 MOVIMENTOS VENDA

SQLQuery15.sql - MSMANM-039\SQLSERVE2024.Loja (loja (60)) - Microsoft SQL Server Management Studio

Arquivo Editar Exibir Consulta Projeto Ferramentas Janela Ajuda

Loja

Esquema de Objetos

MSMANM-039\SQLSERVE2024 (SQL Server 16.0.1000 - loja)

Bancos de Dados

Bancos de Dados do Sistema

Instantâneos do Banco de Dados

EscolaDB

Loja

Diagramas de Banco de Dados

Tabelas

Tabelas do Sistema

FileTables

Tabelas Externas

Tabelas de Grafo

dbo.MovimentosCompra

dbo.MovimentosVenda

dbo.Pessoas

dbo.PessoasFisicas

dbo.PessoasJuridicas

dboProdutos

dbo.Usuarios

Exibições

Recursos Externos

Sinônimos

Programação

Repositório de Consultas

Service Broker

Armazenamento

Segurança

Objetos de Servidor

SQLQuery15.sql - MS...24.Loja (loja (60))

```
SELECT TOP (1000) [id_movimento]
,[id_produto]
,[id_cliente]
,[quantidade]
,[preco_unitario]
,[data_movimento]
FROM [Loja].[dbo].[MovimentosVenda]
```

100 %

Resultados Mensagens

	id_movimento	id_produto	id_cliente	quantidade	preco_unitario	data_movimento
1	1	1	1	10	3.50	2024-07-22 21:26:12.557
2	2	2	2	20	2.00	2024-07-22 21:26:12.557
3	3	3	1	15	4.00	2024-07-22 21:26:12.557

Consulta executada com êxito. MSMANM-039\SQLSERVE2024 (16... loja (60)) Loja 00:00:00 3 linhas

Pronto Li 1 Col 1 Car 1 INS

## 12 *ANALISE DOCUMENTO 1*

### 12.1 *O COMO SÃO IMPLEMENTADAS AS DIFERENTES CARDINALIDADES, BASICAMENTE IXI, IXN OU NxN, EM UM BANCO DE DADOS RELACIONAL?*

Em um banco de dados relacional, as diferentes cardinalidades são implementadas utilizando chaves primárias e estrangeiras, assim como tabelas de junção para relacionamentos muitos-para-muitos.

- **Relacionamento 1x1:** Este tipo de relacionamento é implementado utilizando chaves primárias que também funcionam como chaves estrangeiras em ambas as tabelas relacionadas. Por exemplo, a tabela *PessoasFisicas* e *PessoasJuridicas* ambas possuem uma chave estrangeira que é ao mesmo tempo uma chave primária e refere-se à chave primária da tabela *Pessoas*.
- **Relacionamento 1xN:** Este relacionamento é implementado utilizando uma chave estrangeira na tabela que representa o lado "muitos" do relacionamento. Por exemplo, um produto pode ter muitos movimentos de compra ou venda, então as tabelas *MovimentosCompra* e *MovimentosVenda* têm chaves estrangeiras que referenciam *Produtos*.
- **Relacionamento NxN:** Para implementar este tipo de relacionamento, utiliza-se uma tabela de junção que contém chaves estrangeiras que referenciam as tabelas envolvidas no relacionamento. Por exemplo, se tivéssemos um relacionamento entre *Produtos* e *Categorias*, uma tabela de junção *ProdutoCategoria* poderia ser usada para associar produtos a categorias.

### 12.2 *QUE TIPO DE RELACIONAMENTO DEVE SER UTILIZADO PARA REPRESENTAR O USO DE HERANÇA EM BANCOS DE DADOS RELACIONAIS?*

Para representar herança em bancos de dados relacionais, o tipo de relacionamento mais adequado é a especialização/generalização. Isso é geralmente implementado utilizando uma combinação de tabelas:



- **Tabela Única (Single Table Inheritance):** Uma única tabela contém todos os campos das classes base e derivadas, e um campo adicional é utilizado para discriminar o tipo da entidade (por exemplo, um campo tipo\_pessoa para diferenciar entre pessoas físicas e jurídicas).
- **Tabelas Separadas (Class Table Inheritance):** Cada classe tem sua própria tabela, e as tabelas derivadas têm uma chave estrangeira que é a chave primária da tabela base. Este método é utilizado no sistema descrito, onde PessoasFisicas e PessoasJuridicas têm chaves estrangeiras que referenciam a chave primária de Pessoas.

### *12.3 COMO O SQL SERVER MANAGEMENT STUDIO PERMITE A MELHORIA DA PRODUTIVIDADE NAS TAREFAS RELACIONADAS AO GERENCIAMENTO DO BANCO DE DADOS?*

O SQL Server Management Studio (SSMS) oferece várias funcionalidades que melhoram a produtividade no gerenciamento de bancos de dados:

- **Interface Gráfica:** A interface gráfica do SSMS facilita a visualização e manipulação de tabelas, índices e outros objetos do banco de dados sem a necessidade de escrever código SQL manualmente.
- **Editor de SQL:** O editor de SQL do SSMS permite a escrita, execução e depuração de scripts SQL de forma eficiente, com funcionalidades como destaque de sintaxe, auto-completar e ferramentas de análise de performance.
- **Ferramentas de Administração:** SSMS inclui ferramentas para backup, restauração, monitoramento de performance e tuning de queries, o que ajuda os administradores a manter o banco de dados otimizado e disponível.
- **Gerenciamento de Segurança:** A interface para gerenciamento de usuários e permissões facilita a configuração de segurança, assegurando que apenas usuários autorizados possam acessar e modificar os dados.

### 13 *ANALISE DOCUMENTO 2*

#### 14 *QUAIS AS DIFERENÇAS NO USO DE SEQUENCE E IDENTITY?*

- ***Sequence:*** Utilizada para gerar números sequenciais independentes de tabelas específicas. É configurável e pode ser usada por várias tabelas ao mesmo tempo. Requer chamada explícita para obter o próximo valor.
- ***Identity:*** Propriedade de uma coluna de tabela que gera automaticamente um número sequencial para novas linhas inseridas. Mais simples de usar, pois não requer chamadas explícitas.

#### 15 *QUAL A IMPORTÂNCIA DAS CHAVES ESTRANGEIRAS PARA A CONSISTÊNCIA DO BANCO?*

*As chaves estrangeiras são cruciais para a integridade referencial e a consistência dos dados em um banco de dados.*

- ***Integridade Referencial:***
  - *Garantem que os valores em uma coluna (chave estrangeira) correspondam a valores em outra tabela (chave primária).*
  - *Previnem a criação de registros órfãos, mantendo relacionamentos válidos entre tabelas.*
- ***Consistência dos Dados:***
  - ***Manutenção de Relacionamentos:*** *Asseguram que as atualizações e exclusões mantenham os dados relacionados corretamente.*
  - ***Prevenção de Erros:*** *Impedem inserções ou atualizações que poderiam levar a inconsistências nos dados.*

16      *QUAIS OPERADORES DO SQL PERTENCEM À ÁLGEBRA RELACIONAL E QUAIS SÃO DEFINIDOS NO CÁLCULO RELACIONAL?*

*Os operadores SQL derivam tanto da álgebra relacional quanto do cálculo relacional, cada um com funções distintas.*

- **Álgebra Relacional:**

- **SELECT:** Para projeção de colunas.
- **FROM:** Especifica as tabelas.
- **WHERE:** Aplica condições de seleção.
- **JOIN:** Combina linhas de tabelas com base em condições.
- **UNION:** Combina resultados de duas consultas.
- **INTERSECT:** Retorna linhas comuns a duas consultas.
- **EXCEPT:** Retorna linhas da primeira consulta que não estão na segunda.
- **GROUP BY:** Agrupa linhas por valores.
- **HAVING:** Aplica condições aos grupos.

- **Cálculo Relacional:**

- **Subconsultas:** Consultas aninhadas dentro de outras.
- **EXISTS:** Verifica a existência de linhas que atendem a uma condição.
- **ALL:** Verifica se uma condição é verdadeira para todos os valores de um subconjunto.

17      *COMO É FEITO O AGRUPAMENTO EM CONSULTAS, E QUAL REQUISITO É OBRIGATÓRIO?*

*O agrupamento em consultas SQL é realizado utilizando a cláusula **GROUP BY**, que permite a agregação de dados com base em valores comuns em colunas especificadas.*

- **Como é feito:**

- Utiliza-se **GROUP BY** após a cláusula **FROM** para agrupar linhas com valores iguais.
- Funciona em conjunto com funções de agregação como **SUM**, **COUNT**, **AVG**, **MIN** e **MAX**.

**Requisito Obrigatório:**

- Todas as colunas selecionadas que não são usadas em funções de agregação devem ser incluídas na cláusula **GROUP BY**.
- Isso garante que cada coluna na seleção seja ou uma coluna agregada ou uma coluna de agrupamento.

## **18 CONCLUSÃO**

Este trabalho detalha a criação de um modelo de dados para um sistema de cadastro utilizando o DBDesigner Fork e o SQL Server Management Studio. A definição cuidadosa das tabelas e seus relacionamentos, a implementação de chaves primárias e estrangeiras, e a utilização de herança para modelar pessoas físicas e jurídicas demonstram uma abordagem robusta e estruturada para o design de bancos de dados relacionais. A análise das cardinalidades e a aplicação de conceitos de herança garantem a integridade e a eficiência do sistema. A utilização do SQL Server Management Studio melhora significativamente a produtividade no gerenciamento do banco de dados, oferecendo uma interface amigável, ferramentas de administração poderosas e funcionalidades de segurança avançadas. Este documento não só discute a implementação técnica do modelo de dados, mas também realça a importância de ferramentas eficazes para o sucesso do desenvolvimento e manutenção de sistemas de banco de dados.

## 19 REFERÊNCIAS

**MICROSOFT DOCS-SQL SERVER MANAGEMENT STUDIO (SSMS)**, Descrição: Documentação oficial da Microsoft para o SQL Server Management Studio, fornecendo instruções detalhadas sobre a instalação, configuração e uso da ferramenta.

Acessado em 2024.

Site: <https://docs.microsoft.com/en-us/sql/ssms/sql-server-management-studio-ssms>

**DBDesigner Fork-SourceForge**, Descrição: Página do projeto DBDesigner Fork no SourceForge, oferecendo download, documentação e detalhes sobre a ferramenta de modelagem de banco de dados.

Acessado em 2024. Site: <https://sourceforge.net/projects/dbdesigner-fork/>

**W3Schools-SQL Tutorial**, Descrição: Um tutorial abrangente sobre SQL, cobrindo desde conceitos básicos até tópicos avançados, incluindo a criação de tabelas, chaves primárias e estrangeiras, e modelagem de dados relacionais.

Acessado em 2024, Site: <https://www.w3schools.com/sql/>

**Oracle - Database Design and Data Modeling**, Descrição: Recursos e guias sobre design de banco de dados e modelagem de dados, com foco em boas práticas e estratégias de design eficiente.

Acessado em 2024,

Site: <https://www.oracle.com/database/technologies/database-design-data-modeling.html>

**Stack Overflow - Implementing Inheritance in Relational Databases**, Descrição: Discussão detalhada sobre diferentes abordagens para implementar herança em bancos de dados relacionais, com exemplos práticos e recomendações.

Database System Concepts - Abraham Silberschatz, Henry F. Korth, S. Sudarshan:

Livro: "Database System Concepts" Descrição: Um livro abrangente sobre conceitos de sistemas de banco de dados, cobrindo design, implementação e administração de bancos de dados relacionais.

Acessado em 2024,

*Site:*

*<https://stackoverflow.com/questions/144615/what-is-the-best-way-to-implement-inheritance-in-a-database>*

