CENTRO UNIVERSITÁRIO ESTÁCIO DE SANTA CATARINA

Desenvolvimento Full Stack

Nome do Aluno: Matheus Oliveira de Jesus

Matrícula: 2023.10.10087-8

Disciplina: Vamos Manter as Informações?

Turma: RPG0015

Este relatório tem como objetivo apresentar códigos e resultados referente aos exercícios solicitados em: Missão Prática | Nível 2 | Mundo 3.

Respostas: 1º Procedimento | Criando o Banco de Dados

1 – Título da Prática: Criando o Banco de Dados

2 – Objetivo da Prática:

- 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);
- No final do exercício, o aluno terá vivenciado a experiência de modelar a base de dados para um sistema simples, além de implementá-la, através da sintaxe SQL, na plataforma do SQL Server.

3 – Códigos solicitados

OBS: como fiz mais de uma tentativa para execução do exercício, estava esbarrando no problema de já ter tabelas criadas, por isso usei a lógica IF OBJECT_ID para verificação e remoção.

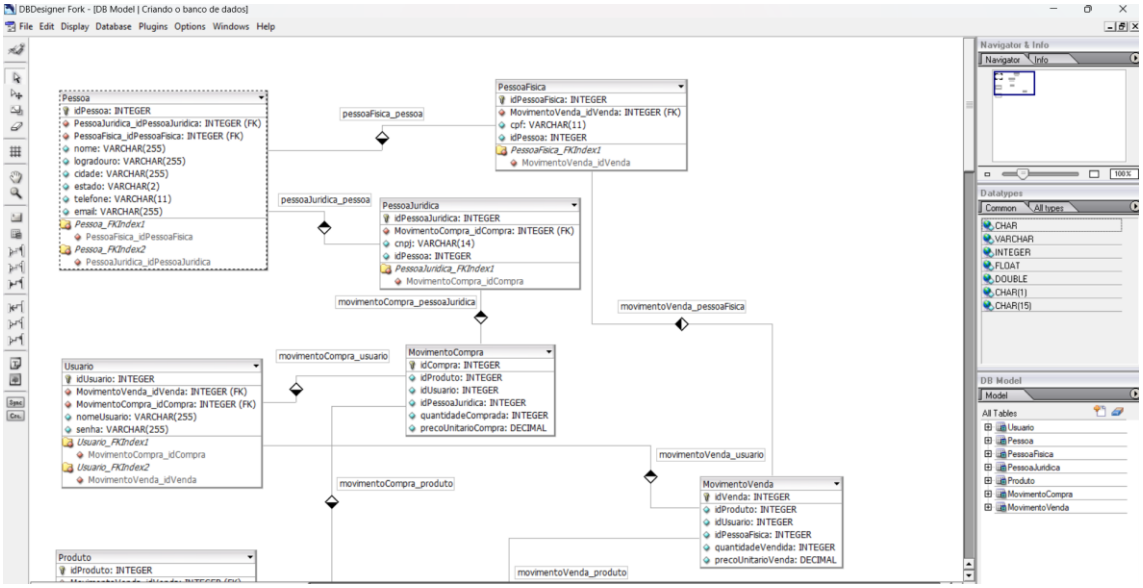
```
IF OBJECT_ID('dbo.MovimentoVenda', 'U') IS NOT NULL
   DROP TABLE dbo.MovimentoVenda;

IF OBJECT_ID('dbo.MovimentoCompra', 'U') IS NOT NULL
   DROP TABLE dbo.MovimentoCompra;

IF OBJECT_ID('dbo.PessoaJuridica', 'U') IS NOT NULL
   DROP TABLE dbo.PessoaJuridica;

IF OBJECT_ID('dbo.PessoaFisica', 'U') IS NOT NULL
   DROP TABLE dbo.PessoaFisica;

IF OBJECT_ID('dbo.Produto', 'U') IS NOT NULL
   DROP TABLE dbo.Produto;

IF OBJECT_ID('dbo.Pessoa', 'U') IS NOT NULL
   DROP TABLE dbo.Pessoa;

IF OBJECT_ID('dbo.Usuarios', 'U') IS NOT NULL
   DROP TABLE dbo.Usuarios;


CREATE TABLE Usuarios (
```

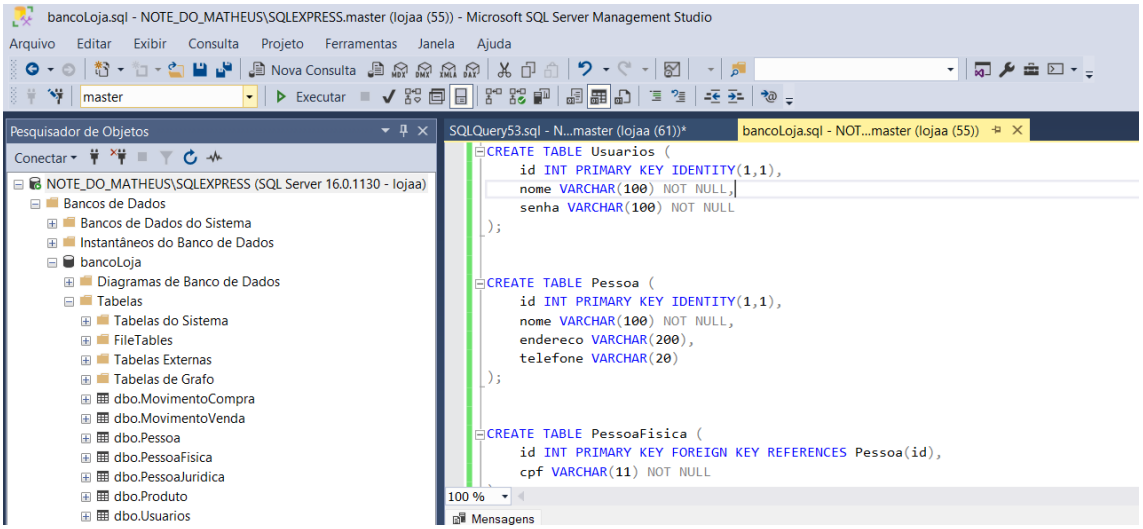
```
  id INT PRIMARY KEY IDENTITY(1,1),
  nome VARCHAR(100) NOT NULL,
  senha VARCHAR(100) NOT NULL
);

CREATE TABLE Pessoa (
  id INT PRIMARY KEY IDENTITY(1,1),
  nome VARCHAR(100) NOT NULL,
  endereco VARCHAR(200),
  telefone VARCHAR(20)
);

CREATE TABLE PessoaFisica (
  id INT PRIMARY KEY FOREIGN KEY REFERENCES Pessoa(id),
  cpf VARCHAR(11) NOT NULL
);

CREATE TABLE PessoaJuridica (
  id INT PRIMARY KEY FOREIGN KEY REFERENCES Pessoa(id),
  cnpj VARCHAR(14) NOT NULL
);

CREATE TABLE Produto (
  id INT PRIMARY KEY IDENTITY(1,1),
```

```
    nome VARCHAR(100) NOT NULL,
    quantidade INT NOT NULL,
    precoVenda DECIMAL(10, 2) NOT NULL
);


CREATE TABLE MovimentoCompra (
    id INT PRIMARY KEY IDENTITY(1,1),
    idProduto INT FOREIGN KEY REFERENCES Produto(id),
    idPessoaJuridica INT FOREIGN KEY REFERENCES PessoaJuridica(id),
    quantidade INT NOT NULL,
    precoUnitario DECIMAL(10, 2) NOT NULL,
    dataMovimento DATE NOT NULL,
    idUsuario INT FOREIGN KEY REFERENCES Usuarios(id)
);


CREATE TABLE MovimentoVenda (
    id INT PRIMARY KEY IDENTITY(1,1),
    idProduto INT FOREIGN KEY REFERENCES Produto(id),
    idPessoaFisica INT FOREIGN KEY REFERENCES PessoaFisica(id),
    quantidade INT NOT NULL,
    precoUnitario DECIMAL(10, 2) NOT NULL,
    dataMovimento DATE NOT NULL,
    idUsuario INT FOREIGN KEY REFERENCES Usuarios(id)
);
```

```
IF OBJECT_ID('dbo.PessoaSeq', 'SO') IS NOT NULL

DROP SEQUENCE dbo.PessoaSeq;


CREATE SEQUENCE dbo.PessoaSeq

AS INT

START WITH 1

INCREMENT BY 1;
```

4 – Resultado dos códigos executados:





5 – Análise e Conclusão:

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

- 1X1 (Um para Um): Tabelas relacionadas por chaves primárias e estrangeiras, onde cada registro em uma tabela corresponde a um único registro na outra.
- 1XN (Um para Muitos): Uma tabela tem uma chave primária e a outra tabela tem uma chave estrangeira que referência essa chave primária. Um registro na tabela "um" pode se relacionar com vários registros na tabela "muitos".
- NxN (Muitos para Muitos): É implementado usando uma tabela intermediária que contém chaves estrangeiras de ambas as tabelas que se relacionam, permitindo assim que múltiplos registros em uma tabela se relacionem com múltiplos registros na outra.

b. 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, existem três abordagens:

1. **Tabela por Classe:** Cada classe tem uma tabela separada. A superclasse contém atributos comuns, enquanto subclasses têm tabelas com atributos específicos e referências à superclasse.

2. **Tabela por Tipo:** Uma única tabela armazena todos os atributos da superclasse e subclasses, com um campo adicional para identificar o tipo de subclasse.

3. **Tabela por Instância:** A superclasse e subclasses têm tabelas separadas, com as subclasses usando chaves primárias que referenciam a superclasse.

   A escolha da abordagem depende das necessidades do sistema, como complexidade e operações de consulta

c. **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 melhora a produtividade no gerenciamento de bancos de dados de várias maneiras:

1. **Interface Gráfica:** Fornece uma interface intuitiva que facilita a navegação e o gerenciamento de objetos de banco de dados, como tabelas, procedimentos armazenados e visualizações.

2. **Autocompletar e Intellisense:** Oferece recursos de autocompletar e sugestões de código, acelerando a escrita de consultas SQL.

3. **Design Visual:** Permite o design visual de tabelas e esquemas, facilitando a criação e modificação de estruturas de banco de dados.

4. **Execução de Consultas:** Possibilita a execução e o teste de consultas de forma rápida e eficiente, com visualização de resultados em formato tabular.

5. **Scripts e Agendamentos:** Suporta a criação de scripts para automação de tarefas repetitivas e agendamentos de manutenção.

6. **Ferramentas de Monitoramento:** Inclui recursos de monitoramento de desempenho e análise de consultas para otimizar a eficiência do banco de dados.

Esses recursos ajudam a reduzir o tempo e o esforço necessários para gerenciar bancos de dados, aumentando a eficiência dos administradores.