UNIVERSIDADE ESTÁCIO DE SÁ

CAMPUS POLO CENTRO - MARICÁ - RJ

ALUNO: LUIZ CARLOS MARINHO JUNIOR

MATRÍCULA: 2023.11.17557-1

CURSO: DESENVOLVIMENTO FULL STACK

SEMESTRE LETIVO: 2024.4

DISCIPLINA: NÍVEL 2: VAMOS MANTER AS INFORMAÇÕES

TÍTULO: 1º PROCEDIMENTO | CRIANDO O BANCO DE DADOS

RIO DE JANEIRO

2024

LUIZ CARLOS MARINHO JUNIOR

1º PROCEDIMENTO | CRIANDO O BANCO DE DADOS

Trabalho prático para aprovação na disciplina de Nível 2: Vamos manter as informações.

Tutora: Prof. Maria B.

RIO DE JANEIRO

2024

SUMÁRIO

1. OBJETIVOS DA PRÁTICA

1. Identificar os requisitos de um sistema e transformá-los no modelo adequado.
2. Utilizar ferramentas de modelagem para bases de dados relacionais.
3. Explorar a sintaxe SQL na criação das estruturas do banco (DDL).

2. CÓDIGOS SOLICITADOS

2.1. APRESENTAÇÃO

Todos os códigos que serão aqui apresentados, estão disponíveis de forma completa no repositório do Github.

O **link para o repositório do Github** se encontra logo abaixo: https://github.com/luizmarinhojr/loja-database

2.2. PRINCIPAIS INSTRUÇÕES DO 1ª PROCEDIMENTO

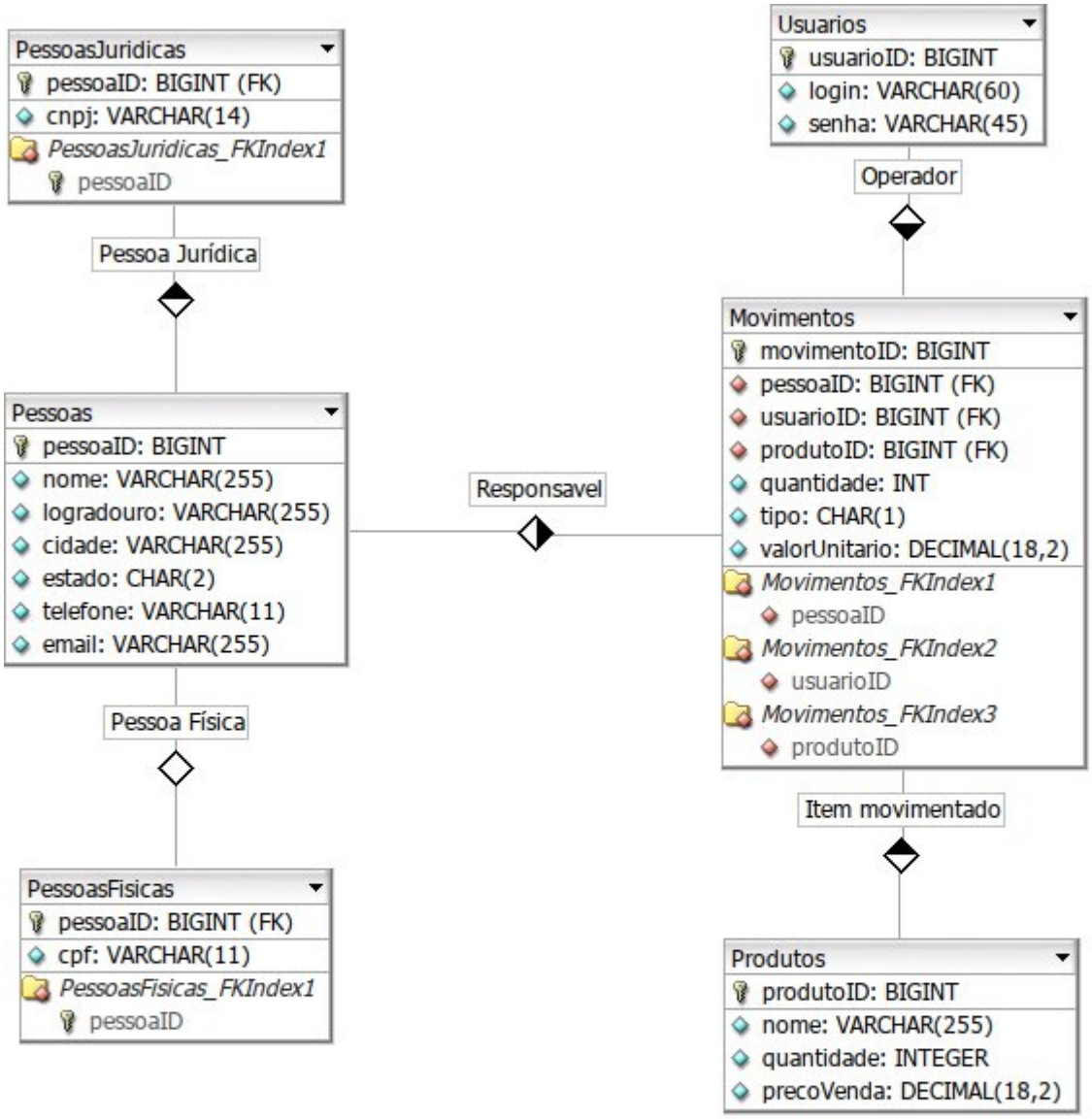
1. Baixar e executar a ferramenta de modelagem:
   a. Acessar o endereço https://sourceforge.net/projects/dbdesigner-fork/ ;
   b. Efetuar o download do DBDesigner Fork no formato Zip;
   c. Descompactar e executar o aplicativo.

2. Definir o modelo de dados para um sistema com as características apresentadas nos tópicos seguintes:
   a. Deve haver um cadastro de usuários para acesso ao sistema, os quais irão atuar como operadores para a compra e venda de produtos.
   b. Deve haver um cadastro de pessoas físicas e pessoas jurídicas, com os dados básicos de identificação, localização e contato, diferenciando-se apenas pelo uso de CPF ou CNPJ.
   c. Deve haver um cadastro de produtos, contendo identificador, nome, quantidade e preço de venda.
   d. Os operadores (usuários) poderão efetuar movimentos de compra para um determinado produto, sempre de uma pessoa jurídica, indicando a quantidade de produtos e preço unitário.

e. Os operadores (usuários) poderão efetuar movimentos de venda para um determinado produto, sempre para uma pessoa física, utilizando o preço de venda atualmente na base.

Observação! No futuro sistema, criado na plataforma Java, será utilizada a herança na definição de pessoas físicas e jurídicas.

Seguindo as instruções acima, foi elaborado o modelo de dados, que se encontra logo abaixo:



3. Utilizar o SQL Server Management Studio para criar a base de dados modelada no tópico anterior:
   a. Logar como usuário sa (System Administrator) e adicionar o logon loja, com senha loja.

```
-- Cria o usuario loja com a senha loja e remove a verificacao de politica de senha
CREATE LOGIN loja WITH PASSWORD = 'loja', CHECK_POLICY = OFF;

-- Garante permissão de administrador para o usuario loja
EXEC sp_addsrvrolemember @loginame='loja', @rolename='sysadmin';
```

   b. Logar novamente com o usuário loja, que deve ter permissão para criação de tabelas e demais estruturas do banco de dados
   c. Utilizar o editor de SQL para criar as estruturas do modelo.
   d. Definir uma sequence para geração dos identificadores de pessoa, dado o relacionamento 1x1 com pessoa física ou jurídica.

O Script completo segue na próxima página e também está disponível no repositório deste projeto, localizado no diretório sql.

```
CREATE SEQUENCE Seq_PessoaID
START WITH 7
INCREMENT BY 8;

CREATE TABLE Usuarios (
    usuarioID BIGINT IDENTITY(1,1) PRIMARY KEY,
    login VARCHAR(60) NOT NULL UNIQUE,
    senha VARCHAR(45) NOT NULL
);

CREATE TABLE Pessoas (
    pessoaID BIGINT PRIMARY KEY DEFAULT NEXT VALUE FOR Seq_PessoaID,
    nome VARCHAR(255) NOT NULL,
    logradouro VARCHAR(255) NOT NULL,
    cidade VARCHAR(255) NOT NULL,
    estado CHAR(2) NOT NULL,
    telefone VARCHAR(11) NOT NULL,
    email VARCHAR(255) NOT NULL
);

CREATE TABLE PessoasFisicas (
    pessoaID BIGINT PRIMARY KEY,
    cpf VARCHAR(11) UNIQUE NOT NULL,
    FOREIGN KEY (pessoaID) REFERENCES Pessoas (pessoaID)
);

CREATE TABLE PessoasJuridicas (
    pessoaID BIGINT PRIMARY KEY,
    cnpj VARCHAR(14) UNIQUE NOT NULL,
    FOREIGN KEY (pessoaID) REFERENCES Pessoas (pessoaID)
);

CREATE TABLE Produtos (
    produtoID BIGINT IDENTITY(1,1) PRIMARY KEY,
    nome VARCHAR(255) NOT NULL,
    quantidade INT NOT NULL DEFAULT 0,
    precoVenda DECIMAL(18, 2) NOT NULL
);

CREATE TABLE Movimentos (
    movimentoID BIGINT PRIMARY KEY IDENTITY(1,1),
    usuarioID BIGINT NOT NULL,
    pessoaID BIGINT NOT NULL,
    produtoID BIGINT NOT NULL,
    quantidade INT NOT NULL CHECK (quantidade > 0),
    tipo CHAR(1) NOT NULL CHECK (tipo IN ('E', 'S')), -- 'E' para entrada, 'S' para saída
    valorUnitario DECIMAL(18,2) NOT NULL CHECK (valorUnitario >= 0),
    FOREIGN KEY (usuarioID) REFERENCES Usuarios(usuarioID),
    FOREIGN KEY (pessoaID) REFERENCES Pessoas(pessoaID),
    FOREIGN KEY (produtoID) REFERENCES Produtos(produtoID)
);
```

3. RESULTADO DA EXECUÇÃO DOS CÓDIGOS

O resultado da execução dos códigos se refere somente às mensagens de sucesso do SGBD, como quando confirma a criação de uma coluna ou inserção de um valor em uma tabela, portanto não vem ao caso coloca-las para exposição nesse procedimento.

4. ANÁLISE E CONCLUSÃO

4.1. COMO SÃO IMPLEMENTADAS AS DIFERENTES CARDINALIDADES, BASICAMENTE 1X1, 1XN OU NXN, EM UM BANCO DE DADOS RELACIONAL?

Em bancos de dados relacionais, as diferentes cardinalidades são implementadas por meio de chaves primárias (PK) e chaves estrangeiras (FK). Cada tipo de relacionamento é modelado de maneira específica.

A **cardinalidade 1x1** ou **um para um**, significa que “um só tem um”. Exemplo: Uma pessoa só pode ter um CPF, e um CPF só pode ter uma pessoa.

A **cardinalidade 1xN** ou **um para muitos**, significa que “um tem muitos e esses muitos só tem um”. Exemplo: Um país tem muitas cidades, porém uma cidade tem apenas um país.

A **cardinalidade NxN** ou **muitos para muitos**, significa que ambos podem ter muitos/vários. Exemplo: Um aluno pode cursar muitas disciplinas e essas muitas disciplinas tem muitos alunos.

4.2. QUE TIPO DE RELACIONAMENTO DEVE SER UTILIZADO PARA REPRESENTAR O USO DE HERANÇA EM BANCOS DE DADOS RELACIONAIS?

Deve ser utilizado, preferencialmente, o relacionamento 1x1 entre tabelas.

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

O SSMS (SQL Server Management Studio) oferece um interface gráfica de usuário (GUI) que facilita diversas tarefas que antes somente poderiam ser realizadas através de linha de comando (CLI), além de possuir, também, integrações com outras ferramentas, funcionalidades de automação e customização, monitoramento e diagnóstico e diversas ferramentas de desenvolvimento e visualização de dados. Englobando tudo isso, o SSMS pode trazer uma melhoria significativa da produtividade em diversas tarefas relacionadas ao gerenciamento do banco de dados.

REFERÊNCIAS

MICROSOFT, SQL Server technical documentation. Disponível em: <https://learn.microsoft.com/en-us/sql/sql-server/?view=sql-server-ver16>. Acesso em 03 de janeiro de 2025.