Campus: Conceição - São Paulo/SP

GRADUAÇÃO DESENVOLVIMENTO FULL STACK

Disciplina: Nível 2 - Vamos manter as informações!

Turma: 2022.03 - Mundo 3

Aluna: Fernanda G. Vargas

Matrícula: 202208836305

GITHUB

Objetivo 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.

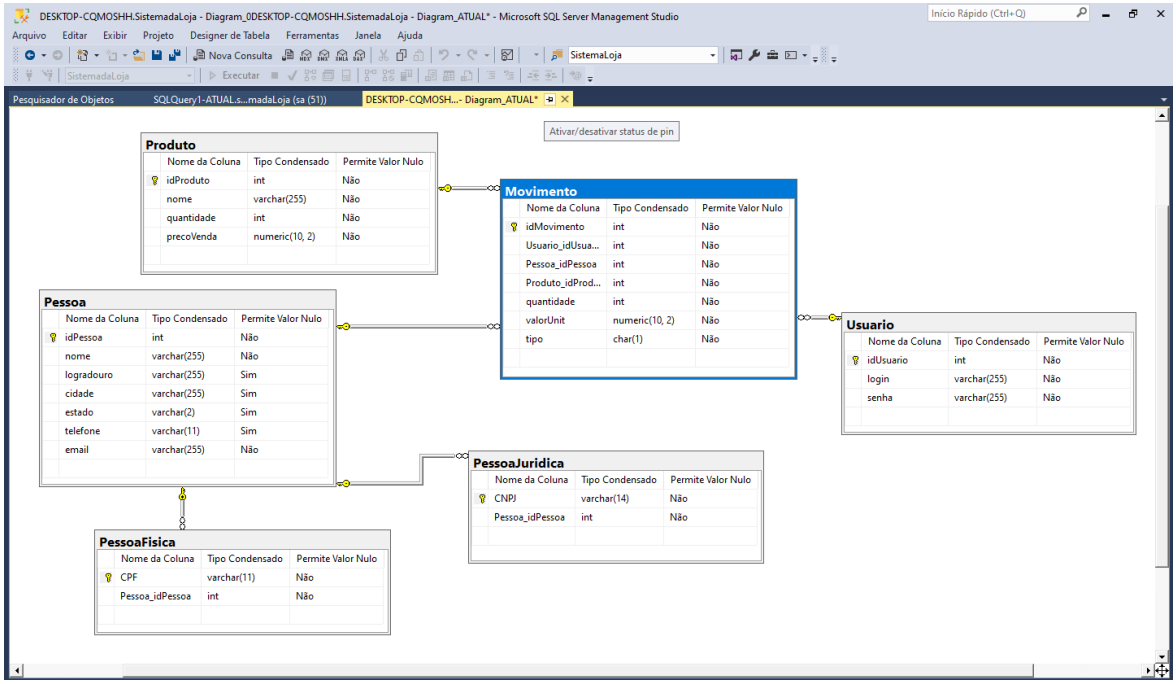
3. Explorar a sintaxe SQL na criação das estruturas do banco (DDL).

4. Explorar a sintaxe SQL na consulta e manipulação de dados (DML).

5.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.

1º Procedimento – Criando o Banco de Dados.

Modelagem Banco de Dados como DBDesingner:



Todos os códigos solicitados 1° Roteiro de aula:

```
CREATE TABLE Produto (
idProduto INTEGER NOT NULL IDENTITY,
nome VARCHAR(255) NOT NULL,
quantidade INTEGER NOT NULL,
precoVenda NUMERIC(10,2) NOT NULL,
PRIMARY KEY(idProduto)
);


CREATE TABLE Usuario (
idUsuario INTEGER NOT NULL,
login VARCHAR(255) NOT NULL,
senha VARCHAR(255) NOT NULL,
PRIMARY KEY(idUsuario)
);

CREATE TABLE Pessoa (
idPessoa INTEGER NOT NULL,
nome VARCHAR(255) NOT NULL,
logradouro VARCHAR(255),
cidade VARCHAR(255),
estado CHAR(2),
telefone VARCHAR(11),
email VARCHAR(255) NOT NULL,
PRIMARY KEY(idPessoa)
);


CREATE TABLE PessoaFisica (
CPF VARCHAR(11) NOT NULL,
Pessoa_idPessoa INTEGER NOT NULL,
PRIMARY KEY(CPF),
FOREIGN KEY(Pessoa_idPessoa)
REFERENCES Pessoa(idPessoa)
);

CREATE TABLE PessoaJuridica (
CNPJ VARCHAR(14) NOT NULL,
Pessoa_idPessoa INTEGER NOT NULL,
PRIMARY KEY(CNPJ),
FOREIGN KEY(Pessoa_idPessoa)
REFERENCES Pessoa(idPessoa)
);
```

```
CREATE TABLE Movimento (
idMovimento INTEGER NOT NULL   IDENTITY,
Usuario_idUsuario INTEGER NOT NULL,
Pessoa_idPessoa INTEGER NOT NULL,
Produto_idProduto INTEGER NOT NULL,
quantidade INTEGER NOT NULL,
tipo CHAR(1) NOT NULL,
valorUnit NUMERIC(10,2) NOT NULL,
PRIMARY KEY(idMovimento),

FOREIGN KEY(Produto_idProduto)
REFERENCES Produto(idProduto),

FOREIGN KEY(Pessoa_idPessoa)
REFERENCES Pessoa(idPessoa),

FOREIGN KEY(Usuario_idUsuario)
REFERENCES Usuario(idUsuario)
);


CREATE INDEX PessoaJuridica_FKIndex1 ON PessoaJuridica (Pessoa_idPessoa);

CREATE INDEX IFK_PessoaJuridica ON PessoaJuridica (Pessoa_idPessoa);

CREATE INDEX PessoaFisica_FKIndex1 ON PessoaFisica (Pessoa_idPessoa);

CREATE INDEX Movimento_FKIndex1 ON Movimento (Produto_idProduto);

CREATE INDEX Movimento_FKIndex2 ON Movimento (Pessoa_idPessoa);

CREATE INDEX Movimento_FKIndex3 ON Movimento (Usuario_idUsuario);

CREATE INDEX IFK_ItemMovimentado ON Movimento (Produto_idProduto);

CREATE INDEX IFK_Cliente ON Movimento (Pessoa_idPessoa);

CREATE INDEX IFK_Responsavel ON Movimento (Usuario_idUsuario);
```

Análise e Conclusão:

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

As cardinalidades em um banco de dados relacional referem-se à relação entre as tabelas e como as linhas de uma tabela estão relacionadas às linhas de outra tabela. As três cardinalidades principais são: 1X1 Um-para-um / 1XN Um-para-muitos / NxN Muitos-para-muitos.

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

O conceito de herança em bancos de dados relacionais, você pode usar duas abordagens principais: a herança por tabela única ou herança por tabela por classe.

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

O SSMS é uma ferramenta com interface gráfica amigável, facilita a administração de ambientes distribuídos, se conecta e gerenciar vários servidores e também logins, é uma ferramenta que se comunicação como o VSCode.E ssas são as principais identificadas por mim.

2º Procedimento – Alimentando a Base.

Todos os códigos solicitados neste 2° Roteiro de aula:

```sql
/* Usuario */

INSERT INTO Usuario (login, senha) values ('op1', 'op1');
INSERT INTO Usuario (login, senha) values ('op2', 'op2');

SELECT * FROM Usuario;

/* Produto */

INSERT INTO Produto (nome, quantidade, precoVenda)
VALUES
    ('Banana', '100', '5.00'),
    ('Laranja', '500', '2.00'),
    ('Manga', '800', '4.00');

SELECT * FROM Produto;

/* Pessoa */

INSERT INTO Pessoa ( idPessoa, nome, logradouro, cidade, estado, telefone, email)
VALUES
    (1, 'Josefa', 'Rua 1 Vilarejo', 'Riacho do Sul', 'PA', '1111-1111', 'jojo@riacho.com.br');
    (10, 'João', 'Rua 10, Quitandinha', 'Riacho do Sul', 'PA', '1111-1111', 'joao@riacho.com');
    (6, 'JJC', 'Rua A, Centro', 'Riacho do Sul', 'PA', '1212-1212', 'jjc@riacho.com');

SELECT * FROM Pessoa;

/* Pessoa Fisica*/

INSERT INTO PessoaFisica (cpf, Pessoa_idPessoa)
VALUES
    ('12345678910', 10);

SELECT * FROM PessoaFisica;

/* Pessoa Juridica */

INSERT INTO PessoaJuridica (Pessoa_idPessoa, cnpj)
VALUES
    (6, '12345678900010');

SELECT * FROM PessoaJuridica;

/* Movimentação */

INSERT INTO Movimento (Usuario_idUsuario, Pessoa_idPessoa, Produto_idProduto, quantidade, tipo, valorUnit)
VALUES
    (1, 10, 5, 30, 'S', 5.00),
    (1, 1, 6, 15, 'S', 2.00),
    (2, 6, 6, 500, 'E', 2.00),
    (1, 6, 5, 200, 'E', 5.00),
    (2, 6, 7, 50, 'E', 4.00);

SELECT * FROM  Movimento;

/* Dados Pessoa Fisica e Juridica */

SELECT * FROM Pessoa;
SELECT * FROM PessoaFisica;
SELECT * FROM PessoaJuridica;

/* Movimentação de Entrada  */

SELECT
    Movimento.tipo,
    Pessoa.nome AS Fornecedor,
    Produto.nome,
    Movimento.quantidade,
    Produto.precoVenda AS Preco_Unitario,
    (Movimento.quantidade * Produto.precoVenda) AS Valor_Total
FROM
    Movimento
JOIN
    Produto ON Movimento.Produto_idProduto = Produto.idProduto
JOIN
    Pessoa ON Movimento.Pessoa_idPessoa = Pessoa.idPessoa
WHERE
    Movimento.tipo = 'E';

/* Movimentação de Saida  */

SELECT
    Movimento.tipo,
    Pessoa.nome AS Comprador,
    Produto.nome,
    Movimento.quantidade,
    Produto.precoVenda AS Preco_Unitario,
    (Movimento.quantidade * Produto.precoVenda) AS Valor_Total
FROM
    Movimento
JOIN
    Produto ON Movimento.Produto_idProduto = Produto.idProduto
JOIN
    Pessoa ON Movimento.Pessoa_idPessoa = Pessoa.idPessoa
WHERE
    Movimento.tipo = 'S';
```

```sql
/* Entradas agrupadas */

SELECT
    Produto.nome AS Produto,
    SUM(Movimento.quantidade * Produto.precoVenda) AS Valor_Total_Entradas
FROM
    Movimento
JOIN
    Produto ON Movimento.Produto_idProduto = Produto.idProduto
WHERE
    Movimento.tipo = 'E'
GROUP BY
    Produto.nome;

/* Saídas agrupadas */

SELECT
    Produto.nome AS Produto,
    SUM(Movimento.quantidade * Produto.precoVenda) AS Valor_Total_Saidas
FROM
    Movimento
JOIN
    Produto ON Movimento.Produto_idProduto = Produto.idProduto
WHERE
    Movimento.tipo = 'S'
GROUP BY
    Produto.nome;


/* Sem Movimentação Entrada */

SELECT
    Pessoa.nome AS Sem_Movimentação_Entrada
FROM
    Pessoa
WHERE
    idPessoa NOT IN (
SELECT DISTINCT
    Movimento.Pessoa_idPessoa
FROM
    Movimento
WHERE
    Movimento.tipo = 'E');

/* Valor total de Entrada Op. */

SELECT
    u.login,
    SUM(quantidade * valorUnit) as Total_Entrada
FROM
    Movimento AS m
INNER JOIN
    Usuario AS u ON (m.Usuario_idUsuario = u.idUsuario)
WHERE
    m.tipo = 'E'
GROUP BY
    u.login


/* Valor total de Saída Op. */

SELECT
    Pessoa.nome AS Operador,
    SUM(Movimento.quantidade * Produto.precoVenda) AS Total_Saída
FROM
    Pessoa
LEFT JOIN
    Movimento ON Pessoa.idPessoa = Movimento.Pessoa_idPessoa
LEFT JOIN
    Produto ON Movimento.Produto_idProduto = Produto.idProduto
WHERE
    Movimento.tipo = 'S'
GROUP BY
    Pessoa.nome;

/* Média de Vendas por Produto (Ponderada) */

SELECT
    Produto.nome AS Produto,
    SUM(Movimento.quantidade * Movimento.valorUnit)/
    SUM(Movimento.quantidade) AS Medio_Venda
FROM
    Produto
LEFT JOIN
    Movimento ON Produto.idProduto = Movimento.Produto_idProduto
WHERE
    Movimento.tipo = 'S'
GROUP BY
    Produto.nome;

--Dev.Fer*
```

Análise e Conclusão:

a) Quais as diferenças no uso de sequence e identity?

Em bancos relacionais Sequencia e a Identidade são usadas gerar valores automáticos. Sequencia - Portátil, variados sistema de gerenciamento de BD suportam a especificação SQL padrão. Mais flexível na geração de valores únicos também em colunas e tabelas. Identidade - Mais fácil de usar por ser incorporada ao definir a coluna e o Sistema de gerenciamento de BD se encarrega de gerar automaticamente os valores. Amplamente suportada e utilizada porem não tão portátil quando comparada a sequencia, considerando erantdo uma mudança de sistema de gerenciamento de BD.

b) Qual a importância das chaves estrangerias para a consistência do banco?

São fundamentais para garantir a consistência e integridade dos dados em um BD relacional. Elas ajudam a manter relações confiáveis entre tabelas, evitam dados órfãos e contribuem para a robustez e segurança do SGBD.

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

Os principais operadores SQL são: WHERE: Especifica os critérios que os valores do campo devem cumprir. SELECT: Consulta e recupera dados de um banco de dados relacional. UNION: Combina os resultados de duas subconsultas em um único resultado INNER JOIN: Compara cada linha das tabelas para satisfazem a condição de junção. LEFT JOIN: Começa a selecionar dados da tabela a “direita”.