

Universidade Estácio

Campus: POLO FSP-RO

Curso: Desenvolvimento Full Stack

Disciplina: Missão prática do 2º nível de conhecimento

Turma: 2025

Semestre Letivo: 1

Integrantes: Eliton Rodriguês de Oliveira

2º Procedimento Alimentando a Base de Dados

Título da Prática:

Alimentando a Base de Dados no SQL Server Management Studio

Objetivo da Prática:

Popular as tabelas criadas no banco de dados com dados básicos, incluindo usuários, produtos, pessoas físicas e jurídicas, e movimentações. Realizar consultas SQL para verificar a consistência dos dados.

1. Inserção de Usuários

Os seguintes comandos SQL foram utilizados para inserir os usuários operadores no banco de dados:

INSERT INTO Usuario (nome, email, senha) VALUES ('Operador1', 'op1@empresa.com', 'op1');

INSERT INTO Usuario (nome, email, senha) VALUES ('Operador2', 'op2@empresa.com', 'op2');

Observação: Para sistemas reais, as senhas devem ser armazenadas em formato hash, codificado em Base64, para garantir segurança.

2. Inserção de Produtos

Os seguintes produtos foram adicionados à base de dados:

INSERT INTO Produto (nome, quantidade, precoVenda) VALUES ('Produto A', 100, 50.00);

INSERT INTO Produto (nome, quantidade, precoVenda) VALUES ('Produto B', 200, 30.00);

3. Inserção de Pessoas

Obter o próximo ID da sequence:

SELECT NEXT VALUE FOR PessoaSeq;

Inserir Pessoa Comum:

INSERT INTO Pessoa (idPessoa, nome, endereco, telefone) VALUES (1, 'Empresa XYZ', 'Rua 1, 100', '(11) 99999-9999');

INSERT INTO Pessoa (idPessoa, nome, endereco, telefone) VALUES (2, 'João Silva', 'Avenida 2, 200', '(11) 98888-8888');

Inserir Pessoa Jurídica:

INSERT INTO PessoaJuridica (idPessoa, CNPJ) VALUES (1, '12.345.678/0001-99');

Inserir Pessoa Física:

INSERT INTO PessoaFisica (idPessoa, CPF) VALUES (2, '123.456.789-00');

4. Criação de Movimentações

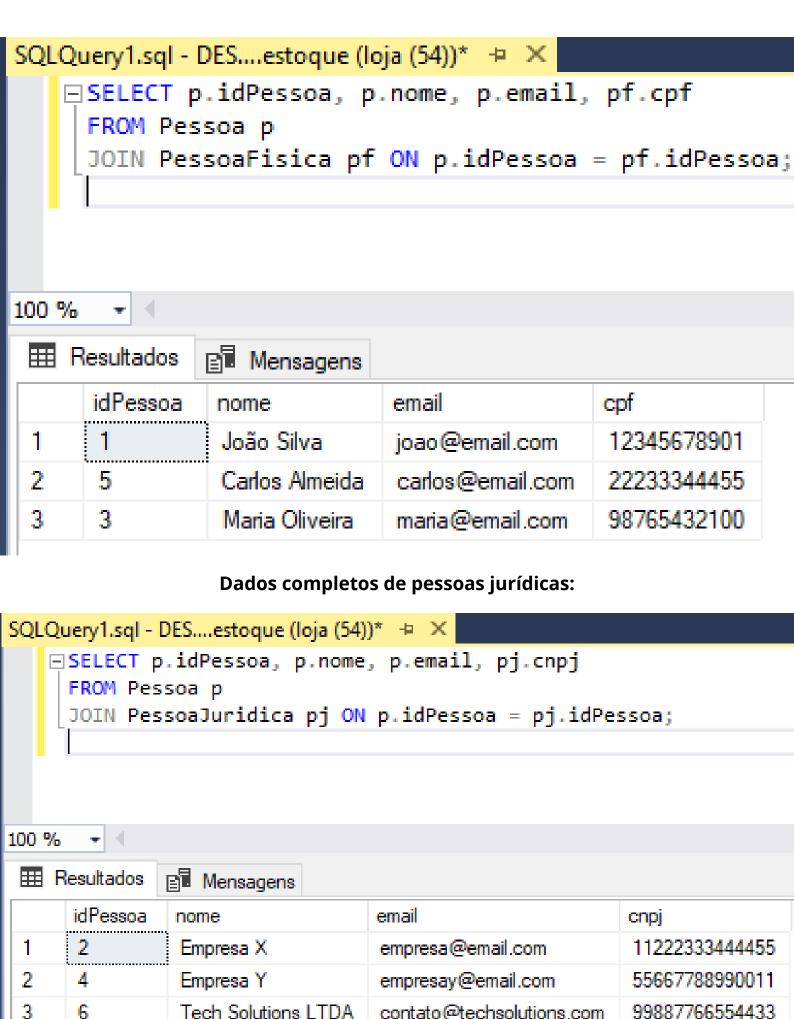
Inserir Movimentações de Entrada e Saída:

INSERT INTO Movimento (tipoMovimento, idProduto, idPessoa, quantidade, precoUnitario) VALUES ('E', 1, 1, 50, 45.00);

INSERT INTO Movimento (tipoMovimento, idProduto, idPessoa, quantidade, precoUnitario) VALUES ('S', 1, 2, 10, 50.00);

5. Consultas Realizadas

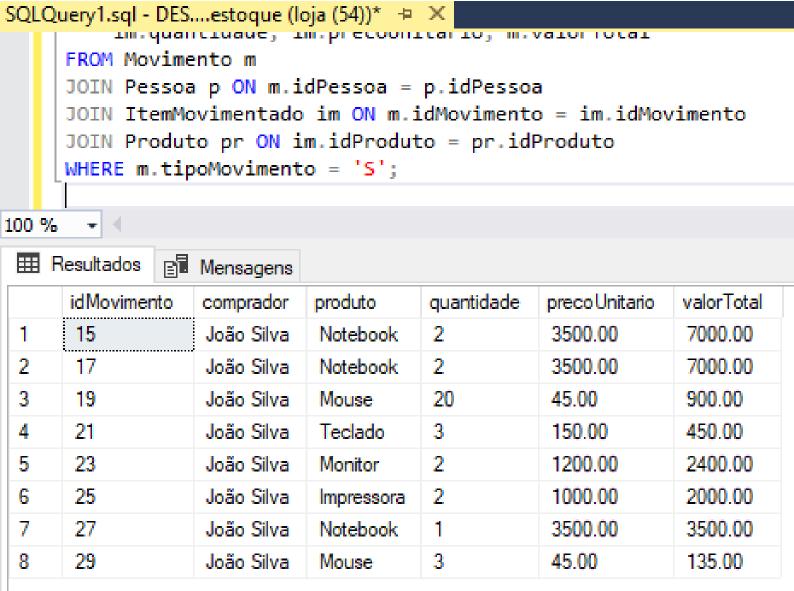
Dados completos de pessoas físicas:



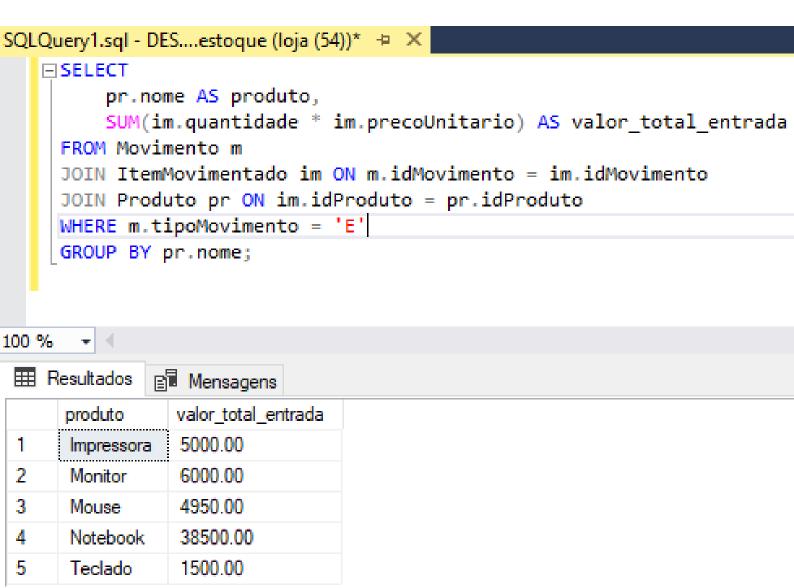
Movimentações de Entrada:

```
SQLQuery1.sql - DES....estoque (loja (54))* → ×
          im.quanciuade, im.precoonicario, m.vaiorrocai
      FROM Movimento m
      JOIN Pessoa p ON m.idPessoa = p.idPessoa
      JOIN ItemMovimentado im ON m.idMovimento = im.idMovimento
      JOIN Produto pr ON im.idProduto = pr.idProduto
     WHERE m.tipoMovimento = 'E';
        + | ∢
100 %
 Mensagens
                              produto
                                                     preco Unitario
                   fomecedor
                                          quantidade
                                                                   valorTotal
      idMovimento.
 1
                   Empresa X
                               Notebook
                                          4
                                                      3500 00
                                                                   14000.00
      14
                    Empresa X
 2
       16
                               Mouse
                                          50
                                                      45 00
                                                                   2250.00
 3
       18
                    Empresa X
                               Notebook
                                          5
                                                      3500.00
                                                                   17500.00
                    Empresa X
                               Teclado.
 4
       20
                                          10
                                                      150.00
                                                                   1500.00
                    Empresa X
                                          5
 5
       22
                               Monitor
                                                      1200.00
                                                                   6000.00
                                                      1000.00
       24
                    Empresa X
                               Impressora
                                          5
 6
                                                                   5000.00
 7
       26
                    Empresa X
                               Notebook
                                          2
                                                      3500.00
                                                                   7000.00
                    Empresa X
       28
                               Mouse
                                          10
                                                      45.00
                                                                   450.00
 8
                    Empresa X
       30
                               Mouse
                                          50
                                                      45.00
                                                                   250.00
 9
```

Movimentações de Saída:



Valor total das entradas agrupadas por produto:



Valor total das saídas agrupadas por produto:

```
SQLQuery1.sql - DES....estoque (loja (54))* 垣 🗶
   FISELECT
          pr.nome AS produto,
          SUM(im.quantidade * im.precoUnitario) AS valor_total_saida
     FROM Movimento m
     JOIN ItemMovimentado im ON m.idMovimento = im.idMovimento
     JOIN Produto pr ON im.idProduto = pr.idProduto
     WHERE m.tipoMovimento = 'S'
     GROUP BY pr.nome;
100 %
       - | ◀

    ⊞ Resultados

    Mensagens

                 valor total saida
      produto
      Impressora 2000.00
 1
 2
      Monitor
                 2400.00
 3
                 1035.00
      Mouse
 4
                 17500.00
      Notebook
```

Operadores que não efetuaram movimentações de entrada (compra)

5

Teclado

450.00

```
SQLQuery2.sql - DES....estoque (loja (56))* 🖈 🔀

□ SELECT u.idUsuario, u.nome

     FROM Usuario u
     WHERE u.idUsuario NOT IN (
          SELECT DISTINCT m.idUsuario
          FROM Movimento m
          WHERE m.tipoMovimento = 'E'
100 %

    ⊞ Resultados

              Mensagens
      idUsuario
               nome
```

Valor total de entrada, agrupado por operador

Operador1

1

```
SQLQuery2.sql - DES....estoque (loja (56))* → ×

□ SELECT

          u.nome AS operador,
          SUM(m.valorTotal) AS total entrada
     FROM Movimento m
     JOIN Usuario u ON m.idUsuario = u.idUsuario
     WHERE m.tipoMovimento = 'E'
     GROUP BY u.nome;
100 %

    ⊞ Resultados

    Mensagens

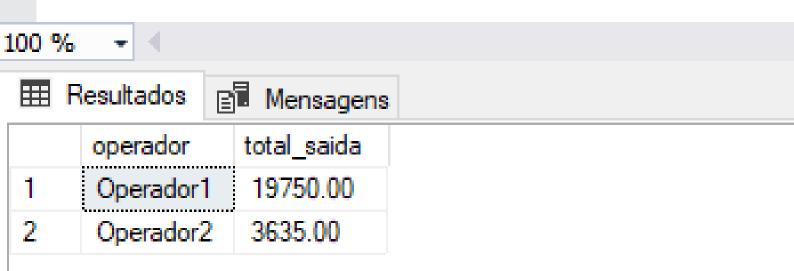
      operador
                   total_entrada
      Administrador 46250.00
 1
 2
      Operador2
                   7700.00
```

Valor total de saída, agrupado por operador

SQLQuery2.sql - DES....estoque (loja (56))* SELECT u.nome AS operador, SUM(m.valorTotal) AS total_saida FROM Movimento m JOIN Usuario u ON m.idUsuario = u.idUsuario

WHERE m.tipoMovimento = 'S'

GROUP BY u.nome;



Valor médio de venda por produto, utilizando média ponderada.

```
SQLQuery2.sql - DES....estoque (loja (56))* 

SELECT

pr.nome AS produto,
SUM(im.quantidade * im.precoUnitario) / SUM(im.quantidade) AS media_ponderada
FROM Movimento m
JOIN ItemMovimentado im ON m.idMovimento = im.idMovimento

JOIN Produto pr ON im.idProduto = pr.idProduto
WHERE m.tipoMovimento = 'S'
GROUP BY pr.nome;
```

	Resultados	Mensagens
	produto	media_ponderada
1	Impressora	1000.000000
2	Monitor	1200.000000
3	Mouse	45.000000
4	Notebook	3500.000000
5	Teclado	150.000000

Análise e Conclusão

1. Quais as diferenças no uso de sequence e identity?

- Sequence permite reutilização de valores e pode ser usada em múltiplas tabelas.
- Identity é específica de uma tabela e gera valores automáticos para uma coluna.

2. Qual a importância das chaves estrangeiras para a consistência do banco?

 As chaves estrangeiras garantem a integridade referencial, impedindo registros "soltos" e assegurando que relações entre tabelas sejam mantidas corretamente.

3. Quais operadores do SQL pertencem à álgebra relacional e quais são definidos no cálculo relacional?

- Operadores da álgebra relacional: SELECT, PROJECT, JOIN, UNION, INTERSECT, DIFFERENCE.
- Operadores do cálculo relacional: EXISTS, FOR ALL, NOT EXISTS, QUANTIFIED PREDICATES.

4. Como é feito o agrupamento em consultas, e qual requisito é obrigatório?

 O agrupamento é realizado com GROUP BY, e qualquer coluna selecionada fora de funções agregadas deve estar no GROUP BY.

Conclusão:

A prática permitiu a alimentação adequada do banco de dados e a realização de consultas SQL, garantindo a consistência dos dados inseridos. O uso de sequence, chaves estrangeiras e operadores SQL foi essencial para estruturar corretamente as informações, garantindo integridade e facilidade de consulta.

Repositório Git

O projeto está armazenado no seguinte repositório Git: ## GitHub - Elitonr65/banco_estoque