Missao Pratica | Nivel 2 | Mundo 3



Universidade: Estácio

Campus: 125 POLO R9 - TAQUARA - RJ

Curso: Desenvolvimento Full Stack

Disciplina: Nível 2: Vamos Manter as Informações?

Turma: 9001

Semestre Letivo: Terceiro

Nome dos integrantes: Marlon Santos Corrêa

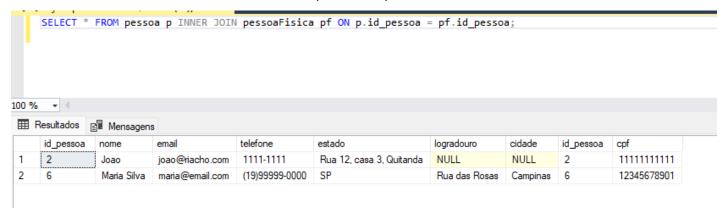
https://github.com/marlonsc15/Missao-Pratica-Nivel-2-Mundo-3.git

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).
- 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.

Resultados da Execucao

Dados completos de pessoas físicas



Dados completos de pessoas jurídicas



Movimentações de entrada, com produto, fornecedor, quantidade, preço unitário e valor total

```
□ SELECT
         c.id compra,
         pr.nome AS produto,
         pj.cnpj AS fornecedor_cnpj,
         p.nome AS fornecedor_nome,
         c.quantidade,
         c.preco unitario,
         (c.quantidade * c.preco_unitario) AS valor_total,
         c.data compra
     FROM
         compra c
     INNER JOIN
         produto pr ON c.id_produto = pr.id_produto
     INNER JOIN
         pessoaJuridica pj ON c.id_pessoa_jur = pj.id_pessoa
     INNER JOIN
         pessoa p ON pj.id_pessoa = p.id_pessoa;
100 % 🕶 🖪
id compra
               produto
                       fomecedor_cnpj
                                      fomecedor_nome
                                                     quantidade
                                                               preco_unitario
                                                                           valor_total
                                                                                     data_compra
 1
      4
                Banana
                       2222222222222
                                                     50
                                                               2.50
                                                                            125.00
                                                                                      2025-04-22
```

Movimentações de saída, com produto, comprador, quantidade, preço unitário e valor total

```
SELECT
         v.id_venda,
         pr.nome AS produto,
         pf.cpf AS comprador_cpf,
         p.nome AS comprador_nome,
         v.quantidade,
         v.preco_unitario,
         (v.quantidade * v.preco_unitario) AS valor_total,
         v.data_venda
     FROM
         venda v
     INNER JOIN
         produto pr ON v.id_produto = pr.id_produto
     INNER JOIN
         pessoaFisica pf ON v.id_pessoa_fis = pf.id_pessoa
     INNER JOIN
        pessoa p ON pf.id_pessoa = p.id_pessoa;
100 % + <

    ⊞ Resultados

              Mensagens
     id_venda
               produto comprador_cpf comprador_nome
                                                   quantidade
                                                              preco_unitario valor_total
                                                                                     data_venda
               Banana 11111111111
                                                                           45.00
                                                                                     2025-04-22 16:07:36.073
                                     Joao
                                                              4.50
```

Valor total das entradas agrupadas por produto

```
□ SELECT

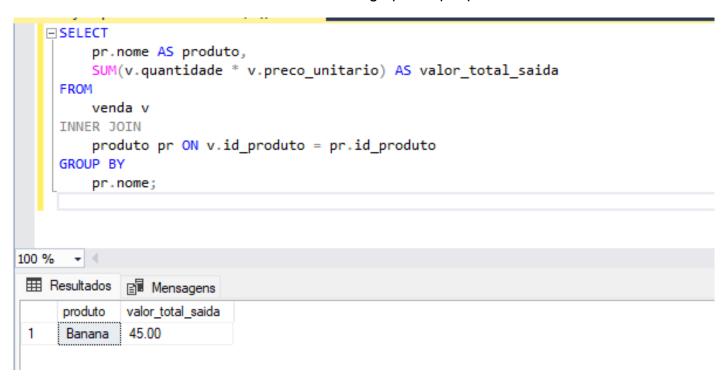
         pr.nome AS produto,
         SUM(c.quantidade * c.preco_unitario) AS valor_total_entrada
     FROM
         compra c
     INNER JOIN
         produto pr ON c.id_produto = pr.id_produto
     GROUP BY
         pr.nome;
100 % ▼ ◀

    ⊞ Resultados

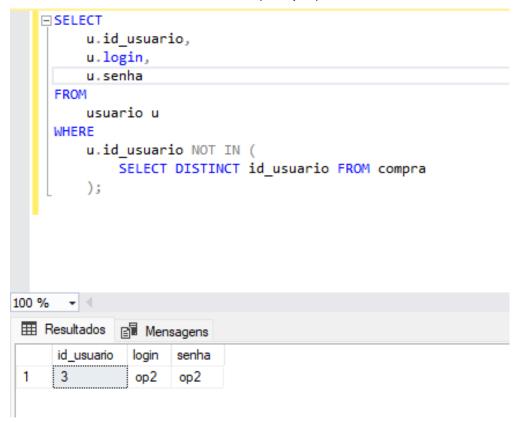
    Mensagens

     produto
              valor_total_entrada
              125.00
      Banana
```

Valor total das saídas agrupadas por produto



Operadores que não efetuaram movimentações de entrada (compra)



Valor total de entrada, agrupado por operador

```
□ SELECT

         u.id_usuario AS operador,
         SUM(c.quantidade * c.preco_unitario) AS valor_total_entrada
     FROM
         compra c
     INNER JOIN
         usuario u ON c.id_usuario = u.id_usuario
     GROUP BY
         u.id_usuario;
100 % → <
 Resultados Mensagens
     operador
             valor_total_entrada
     2
 1
              125.00
```

Valor total de saída, agrupado por operador

```
□ SELECT
         u.id_usuario AS operador,
         SUM(v.quantidade * v.preco unitario) AS valor total saida
     FROM
         venda v
     INNER JOIN
         usuario u ON v.id_usuario = u.id_usuario
     GROUP BY
         u.id usuario;
100 %
       + 4
 Resultados
              Mensagens
               valor_total_saida
      operador
      2
 1
               45.00
```

Valor médio de venda por produto, utilizando média ponderada

```
| P. nome AS produto, | CAST(SUM(v.quantidade * v.preco_unitario) AS FLOAT) / NULLIF(SUM(v.quantidade), 0) AS media_ponderada | venda v | R JOIN | produto p ON v.id_produto = p.id_produto | JP BY | p.nome; | Resultados | Mensagens | produto | media_ponderada | 1 | Banana | 4.5
```

Analise e Conclusao

Quais as diferenças no uso de sequence e identity?

R: IDENTITY é usado diretamente em uma coluna de tabela e SEQUENCE é um objeto separado do banco, que pode ser usado em várias tabelas.

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

R: São essenciais para manter os relacionamentos corretos entre tabelas e evitar erros lógicos nos dados.

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

R: SELECT, WHERE, GROUP BY, ORDER BY, UNION, INTERSECT, EXCEPT, JOIN, LEFT JOIN, RIGHT JOIN, FULL JOIN e CROSS JOIN são Álgebra Relacional.

Ja =, >, <, !=, AND, OR, NOT são Calculo Relacional.

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

R: O agrupamento em consultas é feito com a cláusula GROUP BY no SQL, e o requisito obrigatório é que se houver funções de agregação (SUM, AVG, COUNT, MAX, MIN) na consulta, todas as colunas não agregadas que estão sendo exibidas na seleção (SELECT) também precisam estar na cláusula GROUP BY.