



**Estácio**

**Campus:** Polo Cohama

**Curso:** Desenvolvimento FullStack

**Turma:** 9001

**Disciplina:** Vamos Manter as Informações?

**Nome:** João Victor Sá de Araújo

## **Relatório de Prática: Alimentando a Base**

## 2. Objetivo da Prática

O objetivo desta prática é aplicar os conhecimentos sobre banco de dados relacionais utilizando o SQL Server Management Studio (SSMS). A atividade inclui a criação de tabelas, uso de sequences e identity para gerar identificadores, inserção de dados em tabelas, realização de consultas complexas e análises relacionadas à integridade dos dados e operações de consulta.

## 3. Códigos

ConsultarDados.sql:

```
SELECT
    p.idPessoa,
    p.nome,
    p.logradouro,
    p.cidade,
    p.estado,
    p.telefone,
    p.email,
    pf.cpf
FROM Pessoa p
INNER JOIN PessoaFisica pf ON p.idPessoa = pf.idPessoa;
```

```
SELECT
    p.idPessoa,
    p.nome,
    p.logradouro,
    p.cidade,
    p.estado,
    p.telefone,
    p.email,
    pj.cnpj
FROM Pessoa p
INNER JOIN PessoaJuridica pj ON p.idPessoa = pj.idPessoa;
```

```
SELECT
    m.idMovimento,
    pr.nome AS Produto,
    pe.nome AS Fornecedor,
    m.quantidade,
    m.valorUnitario,
    (m.quantidade * m.valorUnitario) AS ValorTotal
FROM Movimentacao m
INNER JOIN Produtos pr ON m.idProduto = pr.idProduto
INNER JOIN Pessoa pe ON m.idPessoa = pe.idPessoa
WHERE m.tipo = 'E';
```

```

SELECT
    m.idMovimento,
    pr.nome AS Produto,
    pe.nome AS Comprador,
    m.quantidade,
    m.valorUnitario,
    (m.quantidade * m.valorUnitario) AS ValorTotal
FROM Movimentacao m
INNER JOIN Produtos pr ON m.idProduto = pr.idProduto
INNER JOIN Pessoa pe ON m.idPessoa = pe.idPessoa
WHERE m.tipo = 'S';

```

```

SELECT
    pr.nome AS Produto,
    SUM(m.quantidade * m.valorUnitario) AS ValorTotalEntradas
FROM Movimentacao m
INNER JOIN Produtos pr ON m.idProduto = pr.idProduto
WHERE m.tipo = 'E'
GROUP BY pr.nome;

```

```

SELECT
    pr.nome AS Produto,
    SUM(m.quantidade * m.valorUnitario) AS ValorTotalSaidas
FROM Movimentacao m
INNER JOIN Produtos pr ON m.idProduto = pr.idProduto
WHERE m.tipo = 'S'
GROUP BY pr.nome;

```

```

SELECT u.UsuarioID, u.NomeUsuario
FROM Usuarios u
LEFT JOIN Movimentacao m ON u.UsuarioID = m.idUsuario
WHERE m.idMovimento IS NULL OR m.tipo != 'E';

```

```

SELECT
    u.NomeUsuario AS Operador,
    SUM(m.quantidade * m.valorUnitario) AS ValorTotalEntradas
FROM Movimentacao m
INNER JOIN Usuarios u ON m.idUsuario = u.UsuarioID
WHERE m.tipo = 'E'
GROUP BY u.NomeUsuario;

```

```

SELECT
    u.NomeUsuario AS Operador,
    SUM(m.quantidade * m.valorUnitario) AS ValorTotalSaidas
FROM Movimentacao m
INNER JOIN Usuarios u ON m.idUsuario = u.UsuarioID
WHERE m.tipo = 'S'

```

```
GROUP BY u.NomeUsuario;
```

```
SELECT
    pr.nome AS Produto,
    SUM(m.quantidade * m.valorUnitario) / SUM(m.quantidade) AS ValorMedioVenda
FROM Movimentacao m
INNER JOIN Produtos pr ON m.idProduto = pr.idProduto
WHERE m.tipo = 'S'
GROUP BY pr.nome;
```

CriaUsuarios.sql:

```
CREATE TABLE Usuarios (
    UsuarioID INT IDENTITY(1,1) PRIMARY KEY,
    NomeUsuario NVARCHAR(50) NOT NULL,
    Senha NVARCHAR(50) NOT NULL
);
```

```
INSERT INTO Usuarios (NomeUsuario, Senha)
VALUES
    ('op1', 'op1'),
    ('op2', 'op2');
```

```
SELECT * FROM Usuarios;
```

InserirMovimentacao.sql:

```
CREATE TABLE Movimentacao (
    idMovimento INT IDENTITY(1,1) PRIMARY KEY,
    idUsuario INT NOT NULL,
    idPessoa INT NOT NULL,
    idProduto INT NOT NULL,
    quantidade INT NOT NULL,
    tipo CHAR(1) NOT NULL, -- 'E' para Entrada, 'S' para Saída
    valorUnitario DECIMAL(10, 2) NOT NULL,
    FOREIGN KEY (idUsuario) REFERENCES Usuarios(UsuarioID),
    FOREIGN KEY (idPessoa) REFERENCES Pessoa(idPessoa),
    FOREIGN KEY (idProduto) REFERENCES Produtos(idProduto)
);
```

```
INSERT INTO Movimentacao (idUsuario, idPessoa, idProduto, quantidade, tipo,
valorUnitario)
VALUES
    (1, 7, 1, 50, 'E', 5.00),
    (1, 7, 1, 20, 'S', 5.00),
    (2, 15, 3, 100, 'E', 2.00),
```

(2, 15, 4, 200, 'S', 4.00);  
InserirFisicaEJuridica.sql:

```
DECLARE @NovoID INT;  
SET @NovoID = NEXT VALUE FOR PessoaSeq;
```

```
INSERT INTO Pessoa (PessoaID, Nome, TipoPessoa)  
VALUES (@NovoID, 'João da Silva', 'F');
```

```
INSERT INTO PessoaFisica (PessoaID, CPF)  
VALUES (@NovoID, '12345678901');
```

```
DECLARE @NovoID INT;  
SET @NovoID = NEXT VALUE FOR PessoaSeq;
```

```
INSERT INTO Pessoa (PessoaID, Nome, TipoPessoa)  
VALUES (@NovoID, 'Empresa X Ltda', 'J');
```

```
INSERT INTO PessoaJuridica (PessoaID, CNPJ)  
VALUES (@NovoID, '12345678000199');
```

EstruturaPessoas.sql:

```
CREATE SEQUENCE PessoaSeq  
START WITH 1  
INCREMENT BY 1;
```

```
CREATE TABLE Pessoa (  
    PessoaID INT PRIMARY KEY,  
    Nome NVARCHAR(100) NOT NULL,  
    TipoPessoa CHAR(1) NOT NULL -- 'F' para Física, 'J' para Jurídica  
);
```

```
CREATE TABLE PessoaFisica (  
    PessoaID INT PRIMARY KEY,  
    CPF CHAR(11) NOT NULL UNIQUE,  
    FOREIGN KEY (PessoaID) REFERENCES Pessoa(PessoaID)  
);
```

```
CREATE TABLE PessoaJuridica (  
    PessoaID INT PRIMARY KEY,  
    CNPJ CHAR(14) NOT NULL UNIQUE,  
    FOREIGN KEY (PessoaID) REFERENCES Pessoa(PessoaID)  
);
```

InsereProdutos.sql:

```
CREATE TABLE Produtos (  
    idProduto INT PRIMARY KEY,  
    nome NVARCHAR(50) NOT NULL,  
    quantidade INT NOT NULL,  
    precoVenda DECIMAL(10, 2) NOT NULL  
);
```

```
INSERT INTO Produtos (idProduto, nome, quantidade, precoVenda)  
VALUES  
    (1, 'Banana', 100, 5.00),  
    (3, 'Laranja', 500, 2.00),  
    (4, 'Manga', 800, 4.00);
```

```
SELECT * FROM Produtos;
```

#### 4. Resultados Obtidos

100 %

Results

Messages

	idPessoa	nome	logradouro	cidade	estado	telefone	email	cpf
1	7	Joao	Rua 12, casa 3, Quitanda	Riacho do Sul	PA	1111-1111	joao@riacho.com	11111111111

	idPessoa	nome	logradouro	cidade	estado	telefone	email	cnpj
1	15	JJC	Rua 11, Centro	Riacho do Norte	PA	1212-1212	jjc@riacho.com	22222222222222

	idMovimento	Produto	Fornecedor	quantidade	valorUnitario	ValorTotal
1	1	Banana	Joao	50	5.00	250.00
2	3	Laranja	JJC	100	2.00	200.00

	idMovimento	Produto	Comprador	quantidade	valorUnitario	ValorTotal
1	2	Banana	Joao	20	5.00	100.00
2	4	Manga	JJC	200	4.00	800.00

	Produto	ValorTotalEntradas
1	Banana	250.00
2	Laranja	200.00

	Produto	ValorTotalSaidas
1	Banana	100.00
2	Manga	800.00

	UsuarioID	NomeUsuario
1	1	op1

Imagem 1: Resultado da Query ConsultaDados.sql

## 5. Análise e Conclusão

### 5.1. Diferenças no Uso de Sequence e Identity:

- **Sequence:** Permite criar um gerador de números independente de tabelas, podendo ser reutilizado em diversas tabelas ou contextos.
- **Identity:** Restrito a uma tabela específica, é utilizado para criar valores automáticos em colunas incrementais.

### 5.2. Importância das Chaves Estrangeiras:

- Garantem a integridade referencial, evitando inconsistências entre tabelas relacionadas.
- Permitem que o banco rejeite operações que possam comprometer a integridade dos dados.

### 5.3. Operadores do SQL na Álgebra Relacional e Cálculo Relacional:

- **Álgebra Relacional:** Operadores como `SELECT`, `JOIN`, `UNION`, `INTERSECT`, `MINUS`.
- **Cálculo Relacional:** Baseia-se em expressões lógicas e é representado por consultas SQL usando `WHERE` e condições lógicas.

### 5.4. Agrupamento em Consultas:

- Feito com `GROUP BY`, utilizado para agregar resultados baseados em uma ou mais colunas.
- Requisito obrigatório: Qualquer coluna no `SELECT` que não for agregada (ex.: `SUM`, `AVG`) deve estar no `GROUP BY`.

Essa prática reforça a importância do uso de banco de dados relacionais para organização e manipulação eficiente de dados. O uso de chaves estrangeiras e operadores SQL é essencial para garantir a consistência e realizar análises robustas sobre os dados armazenados.