



Desenvolvimento Full Stack

Aluno: Victor de A. Costa

Missão Prática | Nível 2 | Mundo 3 - 2024.3
VAMOS MANTER AS INFORMAÇÕES!

Modelagem e implementação de um banco de dados simples, utilizando como base o SQL Server.

2º Procedimento | Alimentando a Base

---> CÓDIGOS UTILIZADOS

```
INSERT INTO Pessoa (nome, logradouro, cidade, estado, telefone, email) VALUES ('Victor', 'Rua 01', 'SJM', 'RJ', '21994921277', 'victor@estacio.com');  
INSERT INTO Pessoa (nome, logradouro, cidade, estado, telefone, email) VALUES ('Yumi', 'Rua 12', 'Caragua', 'SP', '1111-1111', 'yumi@gmail.com');  
INSERT INTO Pessoa (nome, logradouro, cidade, estado, telefone, email) VALUES ('Marise', 'Rua 14', 'Riacho do Sul', 'PA', '1112-1111', 'marise@gmail.com');  
INSERT INTO Pessoa (nome, logradouro, cidade, estado, telefone, email) VALUES ('Joao', 'Rua 13', 'RJ', 'PA', '1123-1111', 'JOAOaaa@gmail.com');  
INSERT INTO Pessoa (nome, logradouro, cidade, estado, telefone, email) VALUES ('JJC', 'Rua 12', 'RJ', 'PA', '1233-1111', 'JJC@gmail.com');  
INSERT INTO Pessoa (nome, logradouro, cidade, estado, telefone, email) VALUES ('Enterprise', 'logradouro', 'RJ', 'PA', '1523-1111', '1234566@gmail.com');  
  
INSERT INTO Produto (nome, quantidade, precoVenda) VALUES ('Caneca Createe', 50, 5000);  
INSERT INTO Produto (nome, quantidade, precoVenda) VALUES ('Camiseta Createe', 564, 7000);  
INSERT INTO Produto (nome, quantidade, precoVenda) VALUES ('Pôster Createe', 15, 400);  
INSERT INTO Produto (nome, quantidade, precoVenda) VALUES ('Placa Createe', 55, 4000);  
  
INSERT INTO Usuario (login, senha) VALUES ('loja', 'loja');  
INSERT INTO Usuario (login, senha) VALUES ('op1', 'op1');  
INSERT INTO Usuario (login, senha) VALUES ('op2', 'op2');  
INSERT INTO Usuario (login, senha) VALUES ('op3', 'op3');  
  
INSERT INTO Pessoa_Fisica (cpf, idPessoa) VALUES ('45451556', 1);  
INSERT INTO Pessoa_Fisica (cpf, idPessoa) VALUES ('11111111111', 2);  
INSERT INTO Pessoa_Fisica (cpf, idPessoa) VALUES ('11111111112', 3);  
INSERT INTO Pessoa_Fisica (cpf, idPessoa) VALUES ('11111111113', 4);  
INSERT INTO Pessoa_Fisica (cpf, idPessoa) VALUES ('11111111453', 5);  
  
INSERT INTO Pessoa_Juridica (cnpj, idPessoa) VALUES ('454162533456', 1);  
INSERT INTO Pessoa_Juridica (cnpj, idPessoa) VALUES ('2222222222222222', 5);  
INSERT INTO Pessoa_Juridica (cnpj, idPessoa) VALUES ('3333333333333333', 6);  
  
INSERT INTO Mover(idUsuario, idPessoa, idProduto, quantidade, tipo, valorUnitario) VALUES (1, 1, 2, 'E', 5000);  
INSERT INTO Mover (idUsuario, idPessoa, idProduto, quantidade, tipo, valorUnitario) VALUES (2, 2, 2, 1, 'E', 1000);  
INSERT INTO Mover (idUsuario, idPessoa, idProduto, quantidade, tipo, valorUnitario) VALUES (2, 2, 2, 1, 'E', 1000);  
INSERT INTO Mover (idUsuario, idPessoa, idProduto, quantidade, tipo, valorUnitario) VALUES (2, 2, 2, 1, 'S', 1000);  
INSERT INTO Mover (idUsuario, idPessoa, idProduto, quantidade, tipo, valorUnitario) VALUES (2, 2, 2, 1, 'S', 1000);  
INSERT INTO Mover (idUsuario, idPessoa, idProduto, quantidade, tipo, valorUnitario) VALUES (2, 2, 2, 1, 'S', 1000);  
INSERT INTO Mover (idUsuario, idPessoa, idProduto, quantidade, tipo, valorUnitario) VALUES (3, 3, 2, 1, 'E', 10000);  
INSERT INTO Mover (idUsuario, idPessoa, idProduto, quantidade, tipo, valorUnitario) VALUES (3, 3, 3, 10, 'E', 50000);  
INSERT INTO Mover (idUsuario, idPessoa, idProduto, quantidade, tipo, valorUnitario) VALUES (3, 3, 4, 10, 'E', 40000);  
INSERT INTO Mover (idUsuario, idPessoa, idProduto, quantidade, tipo, valorUnitario) VALUES (2, 2, 3, 2, 'S', 4000);  
INSERT INTO Mover (idUsuario, idPessoa, idProduto, quantidade, tipo, valorUnitario) VALUES (2, 2, 3, 2, 'S', 4000);
```

```
INSERT INTO Mover (idUsuario, idPessoa, idProduto, quantidade, tipo, valorUnitario) VALUES (2, 2, 4, 2, 'S', 4000);
INSERT INTO Mover (idUsuario, idPessoa, idProduto, quantidade, tipo, valorUnitario) VALUES (3, 3, 4, 2, 'S', 4000);
```

```
SELECT * FROM Pessoa_Fisica pf
JOIN Pessoa p ON pf.idPessoa = p.idPessoa;
```

```
SELECT * FROM Pessoa_Juridica pj
JOIN Pessoa p ON pj.idPessoa = p.idPessoa;
```

```
SELECT m.idMovimento, p.nome AS Produto, u.login AS Fornecedor, m.quantidade, m.valorUnitario,
(m.quantidade * m.valorUnitario) AS ValorTotal
FROM Mover m
JOIN Produto p ON m.idProduto = p.idProduto
JOIN Usuario u ON m.idUsuario = u.idUsuario
WHERE m.tipo = 'E';
```

```
SELECT m.idMovimento, p.nome AS Produto, c.nome AS Comprador, m.quantidade, m.valorUnitario,
(m.quantidade * m.valorUnitario) AS ValorTotal
FROM Mover m
JOIN Produto p ON m.idProduto = p.idProduto
JOIN Pessoa c ON m.idPessoa = c.idPessoa
WHERE m.tipo = 'S';
```

```
SELECT p.nome AS Produto, SUM((m.quantidade * m.valorUnitario)) AS ValorTotalEntrada
FROM Mover m
JOIN Produto p ON m.idProduto = p.idProduto
WHERE m.tipo = 'E'
GROUP BY p.nome;
```

```
SELECT p.nome AS Produto, SUM((m.quantidade * m.valorUnitario)) AS ValorTotalSaida
FROM Mover m
JOIN Produto p ON m.idProduto = p.idProduto
WHERE m.tipo = 'S'
GROUP BY p.nome;
```

```
SELECT u.login
FROM Usuario u
LEFT JOIN Mover m ON u.idUsuario = m.idUsuario AND m.tipo = 'E'
WHERE m.idMovimento IS NULL;
```

```
SELECT u.login, SUM((m.quantidade * m.valorUnitario)) AS ValorTotalEntrada
FROM Mover m
JOIN Usuario u ON m.idUsuario = u.idUsuario
WHERE m.tipo = 'E'
GROUP BY u.login;
```

```
SELECT u.login, SUM((m.quantidade * m.valorUnitario)) AS ValorTotalSaida
FROM Mover m
JOIN Usuario u ON m.idUsuario = u.idUsuario
WHERE m.tipo = 'S'
GROUP BY u.login;
```

```
SELECT p.nome AS Produto, AVG(m.quantidade * p.precoVenda) AS MediaPonderada
FROM Mover m
JOIN Produto p ON m.idProduto = p.idProduto
WHERE m.tipo = 'S'
GROUP BY p.nome;
```

---> RESULTADO

Resultados		Mensagens									
	cpf	idPessoa	idPessoa	nome	logradouro	cidade	estado	telefone	email		
1	111111111111	2	2	Victor	Rua 01	SJM	RJ	21994921277	victor@estacio.com		
2	111111111112	3	3	Yumi	Rua 12	Caragua	SP	1111-1111	yumi@gmail.com		
3	111111111113	4	4	Marise	Rua 14	Riacho do Sul	PA	1112-1111	marise@gmail.com		
4	1111111111453	5	5	Joao	Rua 13	RJ	PA	1123-1111	JOAOaaa@gmail.com		
5	45451556	1	1	Victor	Av. N. S. das Graças	SJM	RJ	21994921277	victorcosta@estacio.com		
6	454516456	1	1	Victor	Av. N. S. das Graças	SJM	RJ	21994921277	victorcosta@estacio.com		
cnPJ		idPessoa	idPessoa	nome	logradouro	cidade	estado	telefone	email		
1	22222222222222	5	5	Joao	Rua 13	RJ	PA	1123-1111	JOAOaaa@gmail.com		
2	33333333333333	6	6	JJC	Rua 12	RJ	PA	1233-1111	JJC@gmail.com		
3	454162533456	1	1	Victor	Av. N. S. das Graças	SJM	RJ	21994921277	victorcosta@estacio.com		
4	454516213456	1	1	Victor	Av. N. S. das Graças	SJM	RJ	21994921277	victorcosta@estacio.com		
idMovimento	Produto	Fornecedor	quantidade	valorUnitario	ValorTotal						
1	1	Camiseta CREATEEE	loja	2	5000	10000					
2	2	Caneca Createe	loja	1	1000	1000					
3	3	Caneca Createe	loja	1	1000	1000					
4	7	Caneca Createe	op1	1	10000	10000					
5	8	Camiseta Createe	op1	10	50000	500000					
6	9	Pôster Createe	op1	10	40000	400000					
idMovimento	Produto	Comprador	quantidade	valorUnitario	ValorTotal						
1	4	Caneca Createe	Victor	1	1000	1000					
2	5	Caneca Createe	Victor	1	1000	1000					
3	6	Caneca Createe	Victor	1	1000	1000					

	Produto	ValorTotalEntrada
1	Camiseta CREATEE	510000
2	Caneca Createe	12000
3	Pôster Createe	400000

	Produto	ValorTotalSaída
1	Camiseta Createe	16000
2	Caneca Createe	3000
3	Pôster Createe	16000

	login
1	op2
2	op3
3	loja
4	op1
5	op2
6	op3

	login	ValorTotalEntrada
1	loja	12000
2	op1	910000

	login	ValorTotalSaída
1	loja	27000
2	op1	8000

---> ANÁLISE E CONCLUSÃO

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

Sequence é usado para criar uma sequência de números em série, utilizado quando é preciso, uma lista ordenada de números, como IDs automáticos.

Identity é uma propriedade de colunas em tabelas que também gera números automáticos, mas a cada vez que uma nova linha é inserida.

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

As chaves estrangeiras são fundamentais para garantir a integridade referencial no banco de dados. Elas asseguram que os relacionamentos entre tabelas sejam mantidos de forma consistente,

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

Pertencem à álgebra relacional, operadores como seleção, projeção, união, interseção, diferença, produto cartesiano e junção.

Já no cálculo relacional, utilizamos operadores de quantificação e predicados lógicos.

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

Para agrupar dados em uma consulta SQL, usa-se a cláusula GROUP BY. Isso organiza os dados em grupos com base em uma ou mais colunas. Um requisito obrigatório ao usar GROUP BY é que todas as colunas na cláusula SELECT que não estejam sendo agregadas devem ser incluídas na cláusula GROUP BY.