

Campus: Polo Ingleses

Curso: Desenvolvimento Full Stack

Disciplina: Nível 2 - Vamos Manter as Informações?

Turma: 9001

Semestre: 23.3

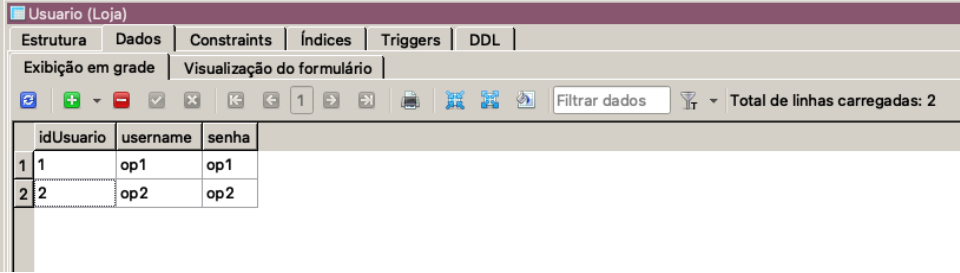
Aluna: Maria Carolina Knudsen Boabaid

2º Procedimento | Alimentando a base

O esperado é que o estudante apresente competência nas habilidades essenciais com a sintaxe SQL para a criação das estruturas necessárias, assim como para a realização de consultas com a DML.

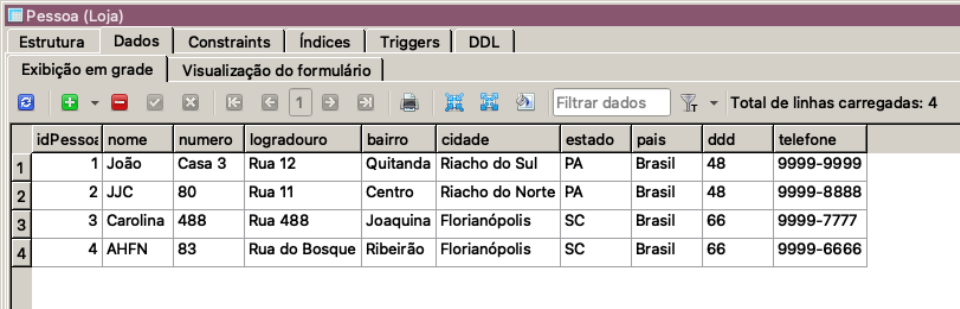
Inclusão de dados no banco:

1. Tabela Usuário



	idUsuario	username	senha
1	1	op1	op1
2	2	op2	op2

2. Tabela Pessoa



	idPessoa	nome	numero	logradouro	bairro	cidade	estado	pais	ddd	telefone
1	1	João	Casa 3	Rua 12	Quitanda	Riacho do Sul	PA	Brasil	48	9999-9999
2	2	JJC	80	Rua 11	Centro	Riacho do Norte	PA	Brasil	48	9999-8888
3	3	Carolina	488	Rua 488	Joaquina	Florianópolis	SC	Brasil	66	9999-7777
4	4	AHFN	83	Rua do Bosque	Ribeirão	Florianópolis	SC	Brasil	66	9999-6666

3. Tabela Pessoa Física

Pessoa_Fisica (Loja)			
Estrutura Dados Constraints Índices Triggers DDL			
Exibição em grade Visualização do formulário			
<div> <input type="text" value="Filtrar dados"/> </div> <div>Total de linhas carregadas: 2</div>			
	idPessoaFisica	idPessoa	cpf
1	1	1	22222222222
2	2	3	33344455566

4. Tabela Pessoa Jurídica

Pessoa_Juridica (Loja)			
Estrutura Dados Constraints Índices Triggers DDL			
Exibição em grade Visualização do formulário			
<div> <input type="text" value="Filtrar dados"/> </div> <div>Total de linhas carregadas: 2</div>			
	idPessoaJuridica	cnpj	idPessoa
1	1	11122233344455	2
2	2	44445555666677	4

5. Tabela Produto

Produto (Loja)				
Estrutura Dados Constraints Índices Triggers DDL				
Exibição em grade Visualização do formulário				
<div> <input type="text" value="Filtrar dados"/> </div> <div>Total de linhas carregadas: 4</div>				
	idProduto	nome	quantidade	valorVenda
1	1	Banana	100	5.4
2	2	Laranja	500	2.5
3	3	Manga	800	4.88
4	4	Maçã	500	1

6. Tabela Compra

Compra (Loja)						
Estrutura Dados Constraints Índices Triggers DDL						
Exibição em grade Visualização do formulário						
<div> <input type="text" value="Filtrar dados"/> </div> <div>Total de linhas carregadas: 5</div>						
	idCompra	quantidadeProduto	precoUnitario	idUsuario	idProduto	idPessoaJuridica
1	1	20	4	1	1	1
2	2	15	2	1	3	2
3	3	10	3	2	3	1
4	4	15	5	1	3	2
5	5	20	4	1	2	1

7. Tabela Venda

Venda (Loja)					
Estrutura Dados Constraints Índices Triggers DDL					
Exibição em grade Visualização do formulário					
<div> <input type="text" value="Filtrar dados"/> </div> <div>Total de linhas carregadas: 4</div>					
	idVenda	precoVenda	idPessoaFisica	idUsuario	idProduto
1	1	50	1	1	1
2	2	60	1	2	3
3	3	34.5	2	1	2
4	4	30	2	2	1

Consultas sobre os dados inseridos:

Dados completos de pessoas físicas

Editor SQL 1

Loja

Consulta | Histórico

1 SELECT Pessoa.idPessoa, Pessoa.nome, Pessoa.numero, Pessoa.logradouro, Pessoa.bairro,

2 Pessoa.cidade, Pessoa.estado, Pessoa.pais, Pessoa.ddd, Pessoa.telefone,

3 Pessoa_Fisica.cpf

4 FROM Pessoa

5 JOIN Pessoa_Fisica ON Pessoa.idPessoa = Pessoa_Fisica.idPessoa

Exibição em grade | Visualização do formulário

Total de linhas carregadas: 2

	idPessoa	nome	numero	logradouro	bairro	cidade	estado	pais	ddd	telefone	cpf
1	1	João	Casa 3	Rua 12	Quitanda	Riacho do Sul	PA	Brasil	48	9999-9999	22222222222
2	3	Carolina	488	Rua 488	Joaquina	Florianópolis	SC	Brasil	66	9999-7777	33344455566

Dados completos de pessoas jurídicas

Editor SQL 1

Loja

Consulta | Histórico

1 SELECT Pessoa.idPessoa, Pessoa.nome, Pessoa.numero, Pessoa.logradouro, Pessoa.bairro,

2 Pessoa.cidade, Pessoa.estado, Pessoa.pais, Pessoa.ddd, Pessoa.telefone,

3 Pessoa_Juridica.cnpj

4 FROM Pessoa

5 JOIN Pessoa_Juridica ON Pessoa.idPessoa = Pessoa_Juridica.idPessoa;

6

Exibição em grade | Visualização do formulário

Total de linhas carregadas: 2

	idPessoa	nome	numero	logradouro	bairro	cidade	estado	pais	ddd	telefone	cnpj
1	2	JJC	80	Rua 11	Centro	Riacho do Norte	PA	Brasil	48	9999-8888	11122233344455
2	4	AHFN	83	Rua do Bosque	Ribeirão	Florianópolis	SC	Brasil	66	9999-6666	44445555666677

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

Editor SQL 1

Loja

Consulta Histórico

```
1 SELECT
2     Compra.idCompra,
3     Produto.nome AS NomeProduto,
4     PessoaJuridica.cnpj AS CNPJFornecedor,
5     Compra.quantidadeProduto,
6     Compra.precoUnitario,
7     (Compra.quantidadeProduto * Compra.precoUnitario) AS ValorTotal
8 FROM
9     Compra
10 JOIN Produto ON Compra.idProduto = Produto.idProduto
11 JOIN PessoaJuridica ON Compra.idPessoaJuridica = PessoaJuridica.idPessoaJuridica;
```

Exibição em grade Visualização do formulário

Total de linhas carregadas: 5

	idCompra	NomePro	CNPJFornecedor	quantida	precoUnif	ValorTota
1	1	Banana	11122233344455	20	4	80
2	2	Manga	44445555666677	15	2	30
3	3	Manga	11122233344455	10	3	30
4	4	Manga	44445555666677	15	5	75
5	5	Laranja	11122233344455	20	4	80

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

Editor SQL 1

Loja

Consulta Histórico

```
1 SELECT
2     Venda.idVenda,
3     Produto.nome AS NomeProduto,
4     PessoaFisica.cpf AS CPFpessoaFisica,
5     Venda.precoVenda
6 FROM
7     Venda
8 JOIN Produto ON Venda.idProduto = Produto.idProduto
9 JOIN PessoaFisica ON Venda.idPessoaFisica = PessoaFisica.idPessoaFisica;
```

Exibição em grade Visualização do formulário

Total de linhas carregadas: 4

	idVenda	NomeProduto	CPFpessoaFisica	precoVenda
1	1	Banana	222222222222	50
2	2	Manga	222222222222	60
3	3	Laranja	333444555666	34.5
4	4	Banana	333444555666	30

Valor total das entradas agrupadas por produto

Editor SQL 1

Loja

Consulta | Histórico

```
1 SELECT Produto.idProduto, Produto.nome AS NomeProduto,
2       SUM(Compra.quantidadeProduto * Compra.precoUnitario) AS ValorTotal
3 FROM Compra
4 JOIN Produto ON Compra.idProduto = Produto.idProduto
5 GROUP BY Produto.idProduto, Produto.nome;
```

Exibição em grade | Visualização do formulário

Total de linhas carregadas: 3

	idProduto	NomeProduto	ValorTotal
1	1	Banana	80
2	2	Laranja	80
3	3	Manga	135

Valor total das saídas agrupadas por produto

Editor SQL 1

Loja

Consulta | Histórico

```
1 SELECT
2     Venda.idProduto,
3     Produto.nome AS NomeProduto,
4     SUM(Venda.precoVenda) AS ValorTotalSaidas
5 FROM
6     Venda
7 JOIN Produto ON Venda.idProduto = Produto.idProduto
8 GROUP BY
9     Venda.idProduto, Produto.nome;
```

Exibição em grade | Visualização do formulário

Total de linhas carregadas: 3

	idProduto	NomeProduto	ValorTotalSaidas
1	1	Banana	80
2	2	Laranja	34.5
3	3	Manga	60

Operadores que não efetuaram movimentações de entrada (compra)

Compra (Loja)						
Estrutura Dados Constraints Índices Triggers DDL						
Exibição em grade Visualização do formulário						
Filtrar dados Total de linhas carregadas: 5						
	idCompra	quantidac	precoUnit	idUsuario	idProduto	idPessoa
1	1	20	4	1	1	1
2	2	15	2	1	3	2
3	3	10	3	1	3	1
4	4	15	5	1	3	2
5	5	20	4	1	2	1

Editor SQL 1

Loja

Consulta

Histórico

```
1 SELECT Usuario.idUsuario, Usuario.username
2 FROM Usuario
3 LEFT JOIN Compra ON Usuario.idUsuario = Compra.idUsuario
4 WHERE Compra.idCompra IS NULL;
5
```

Exibição em grade

Visualização do formulário

✓

✕

⏪

⏩

1

⏴

⏵

Total de linhas carregadas: 1

	idUsuario	username
1	2	op2

Valor total de entrada, agrupado por operador

Editor SQL 1

Loja

Consulta Histórico

```
1 SELECT Usuario.idUsuario, Usuario.username, SUM(Compra.quantidadeProduto * Compra.precoUnitario) AS ValorTotalCompras
2 FROM Usuario
3 LEFT JOIN Compra ON Usuario.idUsuario = Compra.idUsuario
4 GROUP BY Usuario.idUsuario, Usuario.username;
5
```

Exibição em grade Visualização do formulário

Total de linhas carregadas: 2

	idUsuario	username	ValorTotalCompras
1	1	op1	295
2	2	op2	NULL

Valor total de saída, agrupado por operador

Editor SQL 1

Loja

Consulta Histórico

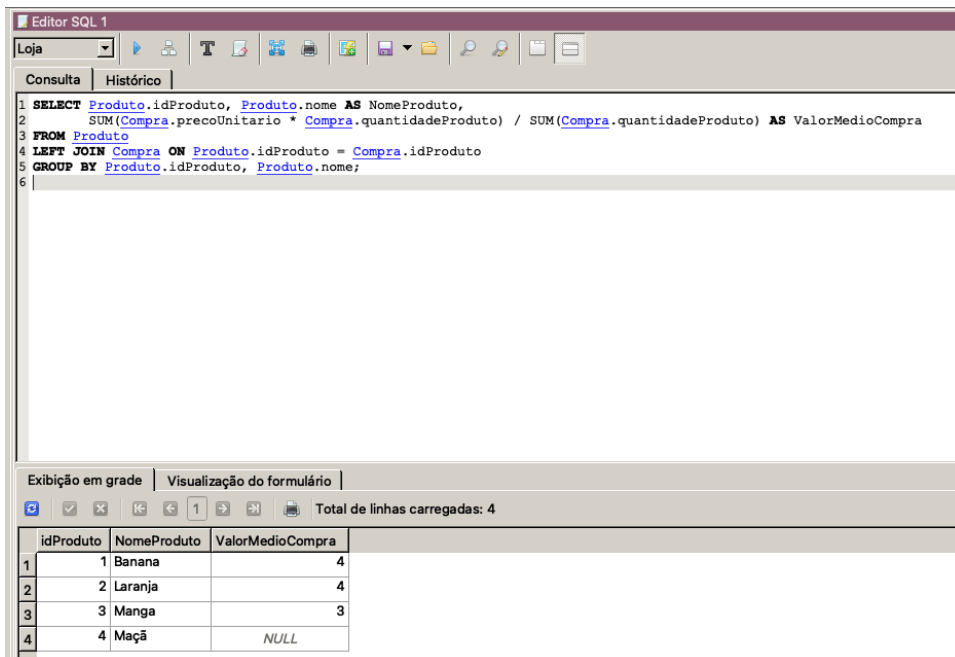
```
1 SELECT Usuario.idUsuario, Usuario.username, SUM(Venda.precoVenda) AS ValorTotalVendas
2 FROM Usuario
3 LEFT JOIN Venda ON Usuario.idUsuario = Venda.idUsuario
4 GROUP BY Usuario.idUsuario, Usuario.username;
5
```

Exibição em grade Visualização do formulário

Total de linhas carregadas: 2

	idUsuario	username	ValorTotalVendas
1	1	op1	84.5
2	2	op2	90

Valor médio de venda por produto, utilizando média ponderada



The screenshot shows a SQL Editor window titled 'Editor SQL 1'. The query editor contains the following SQL code:

```
1 SELECT Produto.idProduto, Produto.nome AS NomeProduto,
2 SUM(Compra.precoUnitario * Compra.quantidadeProduto) / SUM(Compra.quantidadeProduto) AS ValorMedioCompra
3 FROM Produto
4 LEFT JOIN Compra ON Produto.idProduto = Compra.idProduto
5 GROUP BY Produto.idProduto, Produto.nome;
```

Below the query editor, the 'Exibição em grade' (Grid View) tab is active, showing the results of the query in a table. The table has three columns: 'idProduto', 'NomeProduto', and 'ValorMedioCompra'. The results are as follows:

	idProduto	NomeProduto	ValorMedioCompra
1	1	Banana	4
2	2	Laranja	4
3	3	Manga	3
4	4	Maçã	NULL

Quais as diferenças no uso de sequence e identity?

SEQUENCE e IDENTITY são usados em bancos de dados relacionais para gerar valores únicos automaticamente em colunas de identificação (chave primária). A SEQUENCE é mais portátil entre diferentes sistemas (PostgreSQL, Oracle, IBM Db2), oferecendo controle manual e a capacidade de ser compartilhada entre tabelas. Requer referência explícita na inserção de dados. Já o IDENTITY é específico do SQL Server, MySQL, integrado diretamente ao sistema, com menos controle manual e autoincremento implícito. A escolha depende do sistema utilizado e das necessidades específicas de controle e portabilidade.

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

As chaves estrangeiras desempenham um papel crucial na consistência e integridade de um banco de dados relacional. Elas garantem a integridade referencial entre tabelas, evitam a criação de registros órfãos e simplificam a manutenção dos dados. Além disso, as chaves estrangeiras asseguram relacionamentos válidos, facilitam a análise de dados e mantêm a consistência transacional durante operações de modificação em várias tabelas. A presença dessas chaves promove uma estruturação lógica do banco de dados, refletindo os relacionamentos entre entidades no mundo do domínio. Em suma, as chaves estrangeiras são essenciais para preservar a consistência e a integridade dos dados em um ambiente de banco de dados relacional.

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

No universo do SQL, encontramos dois paradigmas fundamentais: a álgebra relacional e o cálculo relacional. Dentro da álgebra relacional, identificamos operadores essenciais como a seleção (SELECT), projeção (PROJECT), junção (JOIN), união (UNION), interseção (INTERSECT) e diferença (EXCEPT). Contudo, é importante destacar que o cálculo relacional não possui uma correspondência direta com operadores específicos no SQL. No SQL, a linguagem declarativa é empregada para recuperar dados, e a cláusula SELECT desempenha um papel central nesse processo. Embora a álgebra e o cálculo relacional tenham influenciado a concepção do SQL, a implementação específica pode variar, e o SQL incorpora características adicionais para atender às suas necessidades práticas.

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

No contexto das consultas em SQL, a execução de agrupamentos se dá por meio da cláusula GROUP BY. Esta se revela como um elemento fundamental ao permitir a categorização de linhas com base nos valores de uma ou mais colunas específicas. O requisito imperativo para empregar a cláusula GROUP BY reside na inclusão, na consulta, de uma função de agregação, tais como SUM(), AVG(), COUNT(), MAX() ou MIN(). Essas funções operam sobre as colunas não agrupadas, sendo essenciais para determinar como os valores agrupados devem ser resumidos ou manipulados. Em suma, a utilização da cláusula GROUP BY, aliada às funções de agregação, é crucial para realizar agrupamentos significativos em consultas SQL.