

# Bancos de Dados

## Roteiro de Exercícios 2

Versão: 0.3

Data: nov/2024

---

Autor: Oclair Prado

Oclair Prado

oclairprado@gmail.br

---

**Objetivos:**

- O objetivo fundamental deste material de apoio complementar é fornecer uma oportunidade adicional para o aluno praticar os conceitos estudados em sala de aula.
- Os exercícios estão divididos em ordem crescente de dificuldade e foram distribuídos em seções.
- Para melhor aproveitamento do material fornecido, o aluno deve resolver os exercícios propostos em casa e trazer os resultados obtidos para serem discutidos em sala de aula.

**Orientações iniciais:**

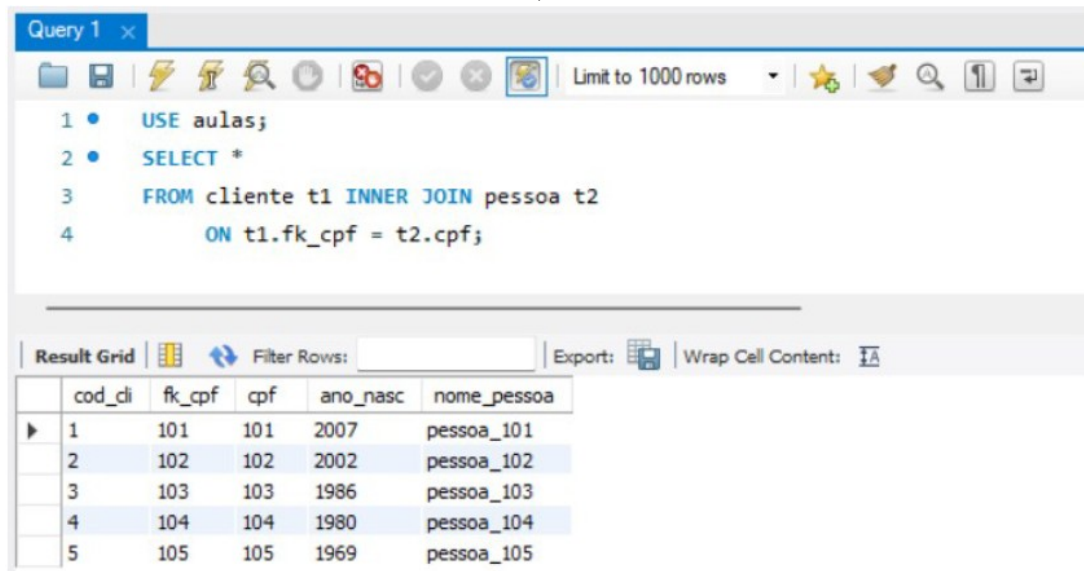
- Este material é continuação do GADS2022\_BD\_roteiro\_1.pdf, que foi criado considerando as vendas em uma loja de alguns produtos para alguns clientes realizadas por alguns vendedores.
- Consulte o documento anterior para orientações sobre a inicialização do SGBD MySQL (Maria DB) e também para o preparo das tabelas com os dados necessários para a realização dos exercícios deste roteiro.

**b continuação) Formas de JOIN:****JOIN** (intersecção de conjuntos)

Traz apenas as linhas que possuem correspondentes em todas as tabelas envolvidas.

Exemplo: `SELECT * FROM t1 INNER JOIN t2 ON t1.a = t2.b;`

Forma alternativa: `SELECT * FROM t1, t2 WHERE t1.a = t2.b;`



The screenshot shows a query editor window titled 'Query 1'. The query text is as follows:

```
1 • USE aulas;
2 • SELECT *
3 FROM cliente t1 INNER JOIN pessoa t2
4 ON t1.fk_cpf = t2.cpf;
```

Below the query editor is a 'Result Grid' showing the results of the query. The grid has 5 columns: 'cod\_cli', 'fk\_cpf', 'cpf', 'ano\_nasc', and 'nome\_pessoa'. There are 5 rows of data:

	cod_cli	fk_cpf	cpf	ano_nasc	nome_pessoa
1	101	101	2007	101	pessoa_101
2	102	102	2002	102	pessoa_102
3	103	103	1986	103	pessoa_103
4	104	104	1980	104	pessoa_104
5	105	105	1969	105	pessoa_105

**CROSS JOIN** (produto cartesiano de conjuntos)

Relaciona cada linha de uma tabela com todas as linhas da outra tabela. Todo **JOIN** sem a cláusula **ON** é um **CROSS JOIN**.

Exemplo: `SELECT * FROM t1 CROSS JOIN t2;`

Variações com o mesmo resultado:

`SELECT * FROM t1 INNER JOIN t2; ou SELECT * FROM t1, t2;`

Query 1 x

Limit to 1000 rows

```

1 • USE aulas;
2 • SELECT *
3 • FROM cliente t1 CROSS JOIN pessoa t2;

```

Result Grid | Filter Rows: | Export: | Wrap Cell Content: |

	cod_cli	fk_cpf	cpf	ano_nasc	nome_pessoa
1	101	101	101	2007	pessoa_101
2	102	101	101	2007	pessoa_101
3	103	101	101	2007	pessoa_101
4	104	101	101	2007	pessoa_101
5	105	101	101	2007	pessoa_101
6	101	101	102	2002	pessoa_102
7	102	102	102	2002	pessoa_102
8	103	102	102	2002	pessoa_102

**LEFT JOIN** é um tipo de **OUTER JOIN**.

Traz todas as linhas da tabela da esquerda (left) e somente os valores da outra tabela que estiverem de acordo com a cláusula de seleção e será usado NULL para as lacunas.

Exemplo: `SELECT * FROM t1 LEFT JOIN t2 ON t1.a = t2.b;`

Query 1 x

Limit to 1000 rows

```

1 • USE aulas;
2 • SELECT * FROM pessoa t1 LEFT JOIN cliente t2 ON t1.cpf = t2.fk_cpf;

```

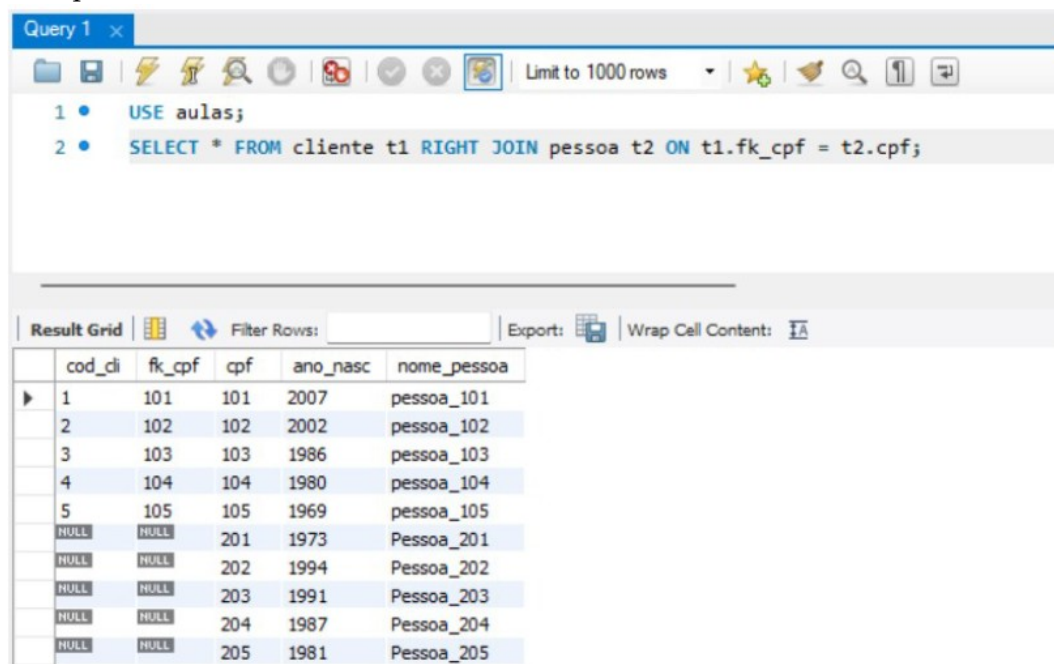
Result Grid | Filter Rows: | Export: | Wrap Cell Content: |

	cpf	ano_nasc	nome_pessoa	cod_cli	fk_cpf
1	101	2007	pessoa_101	1	101
2	102	2002	pessoa_102	2	102
3	103	1986	pessoa_103	3	103
4	104	1980	pessoa_104	4	104
5	105	1969	pessoa_105	5	105
6	201	1973	Pessoa_201	NULL	NULL
7	202	1994	Pessoa_202	NULL	NULL
8	203	1991	Pessoa_203	NULL	NULL
9	204	1987	Pessoa_204	NULL	NULL
10	205	1981	Pessoa_205	NULL	NULL

**RIGHT JOIN** é um tipo de **OUTER JOIN**.

Traz todas as linhas da tabela da direita (right) e somente os valores da outra tabela que estiverem de acordo com a cláusula de seleção e será usado NULL para as lacunas.

Exemplo: `SELECT * FROM t1 RIGHT JOIN t2 ON t1.a = t2.b;`



The screenshot shows a database query tool interface. The query editor contains two steps: 1. `USE aulas;` and 2. `SELECT * FROM cliente t1 RIGHT JOIN pessoa t2 ON t1.fk_cpf = t2.cpf;`. The results are displayed in a table with the following data:

	cod_cli	fk_cpf	cpf	ano_nasc	nome_pessoa
1	101	101	2007	101	101
2	102	102	2002	102	102
3	103	103	1986	103	103
4	104	104	1980	104	104
5	105	105	1969	105	105
NULL	NULL	201	1973	201	201
NULL	NULL	202	1994	202	202
NULL	NULL	203	1991	203	203
NULL	NULL	204	1987	204	204
NULL	NULL	205	1981	205	205

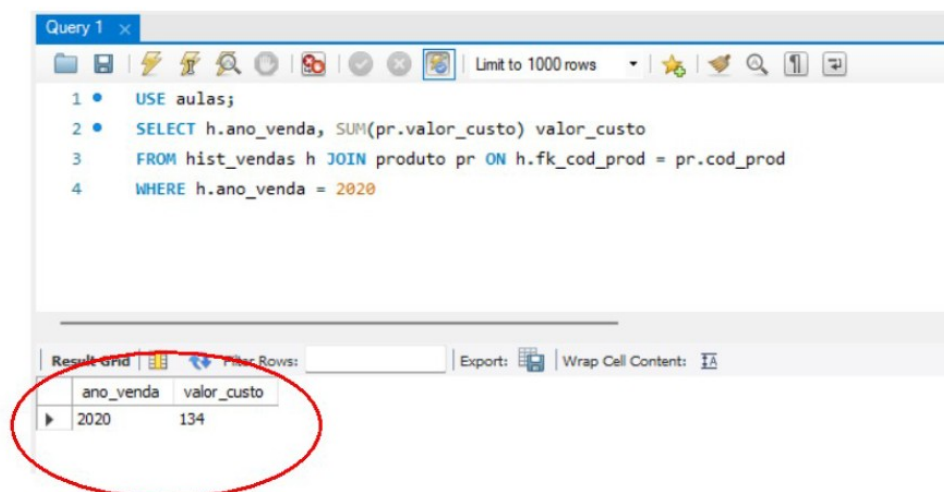
#### e) Select com funções de agregação (sum, max, min, avg etc.):

Funções de agregação são funções SQL que permitem executar uma operação aritmética nos valores de uma coluna em todos os registros de uma tabela. Retornam um valor simples baseado em um conjunto de valores de entrada.

e.1) SUM = Total (Soma) de um conjunto de valores.

Mostre a soma total dos valores de custo dos produtos vendidos em 2020.

**Resultado esperado e exemplo de resolução:**



The screenshot shows a database query tool interface. The query editor contains four steps: 1. `USE aulas;`, 2. `SELECT h.ano_venda, SUM(pr.valor_custo) valor_custo`, 3. `FROM hist_vendas h JOIN produto pr ON h.fk_cod_prod = pr.cod_prod`, and 4. `WHERE h.ano_venda = 2020`. The results are displayed in a table with the following data:

ano_venda	valor_custo
2020	134

e.2) Mostre a soma total dos valores de custo dos produtos vendidos em 2020 pelo vendedor com código 3.

**Resultado esperado:**

Result Grid			
	cod_vendedor	ano_venda	valor_custo
▶	3	2020	24

e.3) Mostre a soma total dos valores de custo dos produtos vendidos em 2020 pelo vendedor com código 3. Mostre também o **CPF** deste vendedor.

**Resultado esperado e exemplo de resolução:**

Query 1 SQL File 3*			
1	USE aulas;		
2	SELECT h.ano_venda, h.fk_cod_vendedor cod_vendedor, v.fk_cpf CPF, SUM(pr.valor_custo) valor_custo		
3	FROM hist_vendas h JOIN produto pr ON h.fk_cod_prod = pr.cod_prod		
4	JOIN vendedor v ON h.fk_cod_vendedor = v.cod_vendedor		
5	WHERE h.ano_venda = 2020 AND h.fk_cod_vendedor = 3		

Result Grid			
	ano_venda	cod_vendedor	CPF
▶	2020	3	203
			24

e.4) Mostre a soma total dos valores de custo dos produtos vendidos em 2020 pelo vendedor com código 3. Mostre também o **nome** deste vendedor.

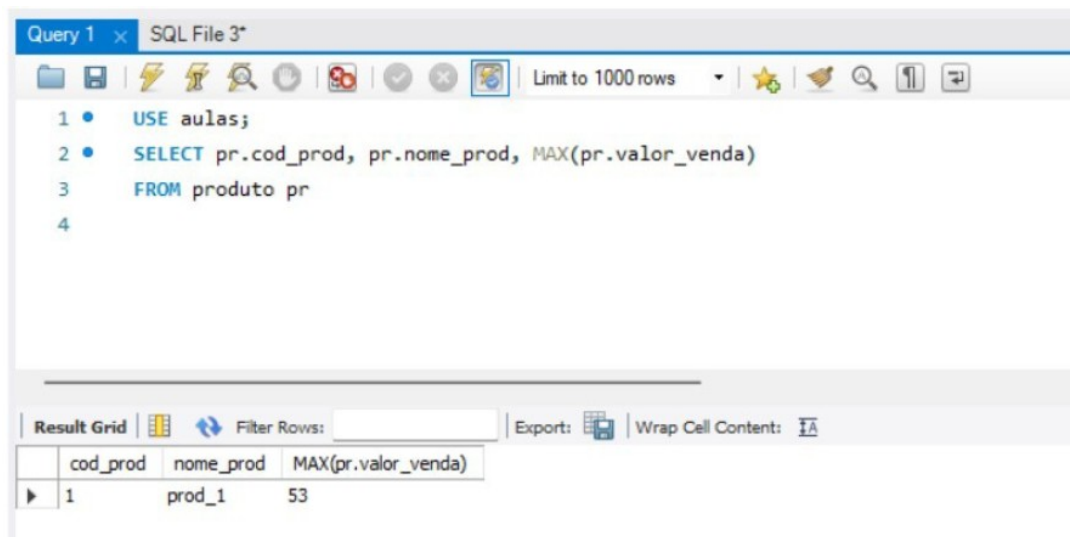
**Resultado esperado:**

Resultado #1 (1r x 5c)					
ano_venda	cod_vendedor	CPF	nome_vendedor	valor_custo	
2.020	3	203	Pessoa_203	24	

e.5) MAX = Valor Máximo de um conjunto de valores.

Localize o produto com maior valor de venda e mostre seu nome e seu valor.

**Resolução e resultado esperado:**



e.6) Mostre o código, nome e valor de venda do produto com maior valor de venda, vendido em 2019.

**Resultado esperado:**

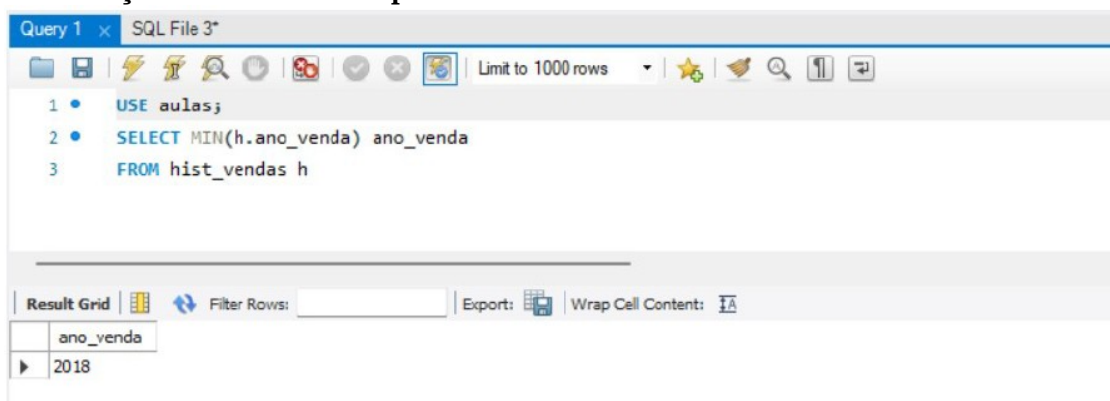
The screenshot shows the expected result for query e.6, which is a table with the following data:

	cod_prod	nome_prod	valor_venda
▶	1	prod_1	46

e.7) MIN = Valor Mínimo de um conjunto de valores.

Mostre o ano do registro de vendas mais antigo em nosso banco de dados.

**Resolução e resultado esperado:**



e.8) Mostre o valor total (use valor\_venda) de vendas em 2020 do vendedor com menor CPF.

**Resultado esperado:**

Resultado #1 (1r x 5c)					
ano_venda	cod_prod	valor_venda	fk_cod_vendedor	CPF	
2.020	1	46	1	201	

e.9) AVG = Média Aritmética de um conjunto de valores.

Mostre o valor médio de venda dos produtos vendidos em 2020.

**Resolução e resultado esperado:**

Query 1 x SQL File 3*					
<pre> 1 • USE aulas; 2 • SELECT h.ano_venda, AVG(pr.valor_venda) média_vendas 3   FROM hist_vendas h JOIN produto pr ON h.fk_cod_prod = pr.cod_prod 4  WHERE h.ano_venda = 2020 </pre>					
<div>Result Grid   Filter Rows:   Export:   Wrap Cell Content: <a href="#">IA</a></div> <table> <tr> <th>ano_venda</th><th>média_vendas</th></tr> <tr> <td>2020</td><td>47</td></tr> </table>		ano_venda	média_vendas	2020	47
ano_venda	média_vendas				
2020	47				

e.10) Mostre o valor médio de venda dos produtos vendidos em 2022 pelo vendedor com código 1.

**Resultado esperado:**

Result Grid   Filter Rows:   Export:   Wrap Cell Content: <a href="#">IA</a>			
ano_venda	cod_vendedor	média_vendas	
2022	1	52.33333333333336	

**f) Select com funções especiais (distinct, count, group by, having, case):**

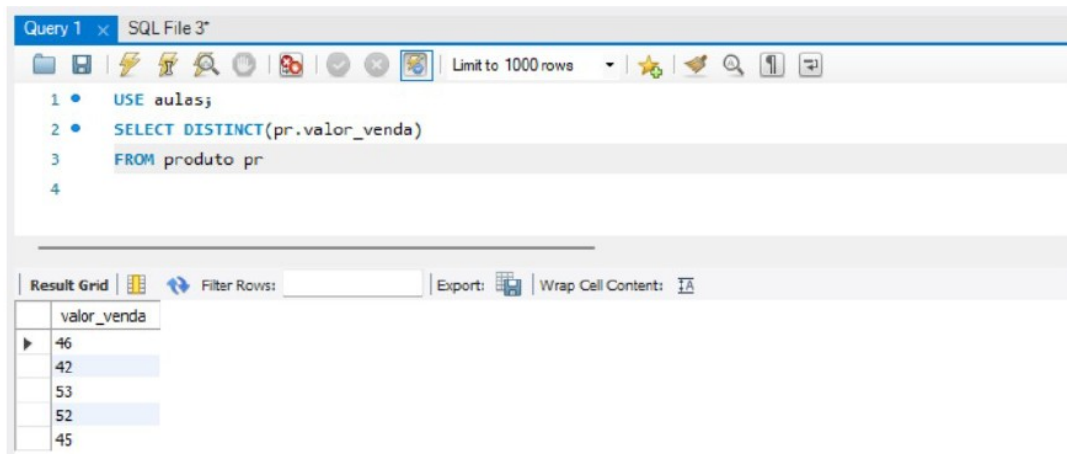
**Resultado esperado:**

f.1) DISTINCT – Usa apenas valores distintos (sem repetição) ao avaliar a função.

Mostre os valores dos preços de venda sem repetições.

**Resolução e resultado esperado:**





Query 1 x SQL File 3\*

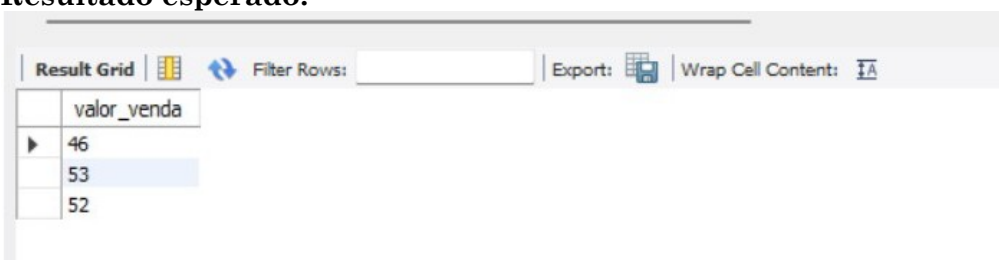
1 • USE aulas;  
2 • SELECT DISTINCT(pr.valor\_venda)  
3 FROM produto pr  
4

Result Grid | Filter Rows: | Export: | Wrap Cell Content: |

valor_venda
46
42
53
52
45

f.2) Mostre os valores de venda sem repetição dos produtos vendidos em 2022 pelo vendedor com código 1.

**Resultado esperado:**



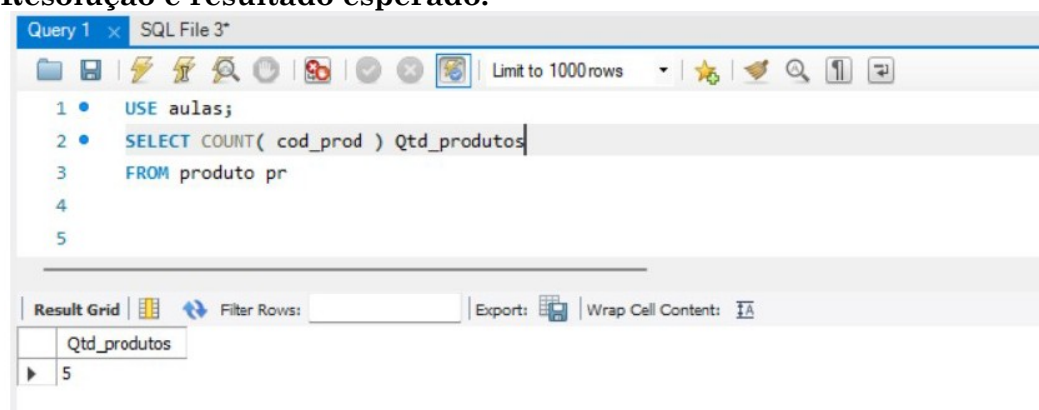
Result Grid | Filter Rows: | Export: | Wrap Cell Content: |

valor_venda
46
53
52

f.3) COUNT = Contar quantidade total de itens.

Mostre a quantidade de produtos cadastrados em nosso banco de dados.

**Resolução e resultado esperado:**



Query 1 x SQL File 3\*

1 • USE aulas;  
2 • SELECT COUNT( cod\_prod ) Qtd\_produtos  
3 FROM produto pr  
4  
5

Result Grid | Filter Rows: | Export: | Wrap Cell Content: |

Qtd_produtos
5

f.4) Mostre a quantidade de valores sem repetição dos produtos vendidos em 2022 pelo vendedor com código 1.

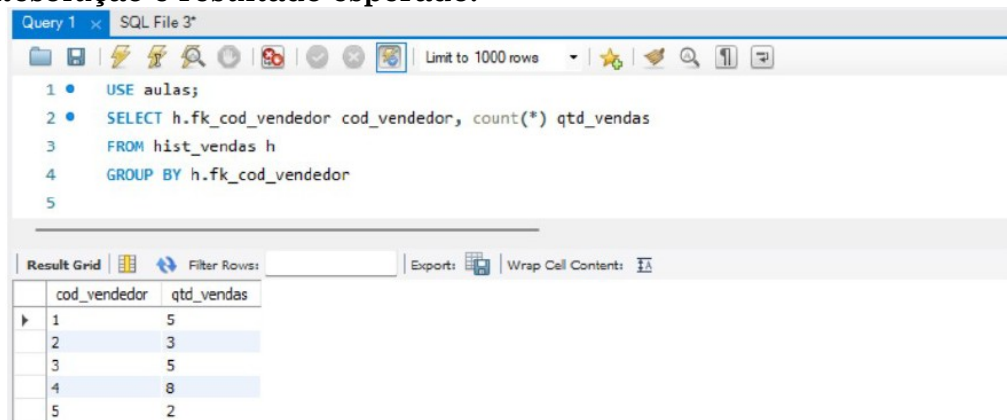
**Resultado esperado:**



hist_vendas (1r x 1c)
qtd
2

## f.5) GROUP BY

Mostre a quantidade de vendas de cada vendedor de nosso banco de dados.

**Resolução e resultado esperado:**


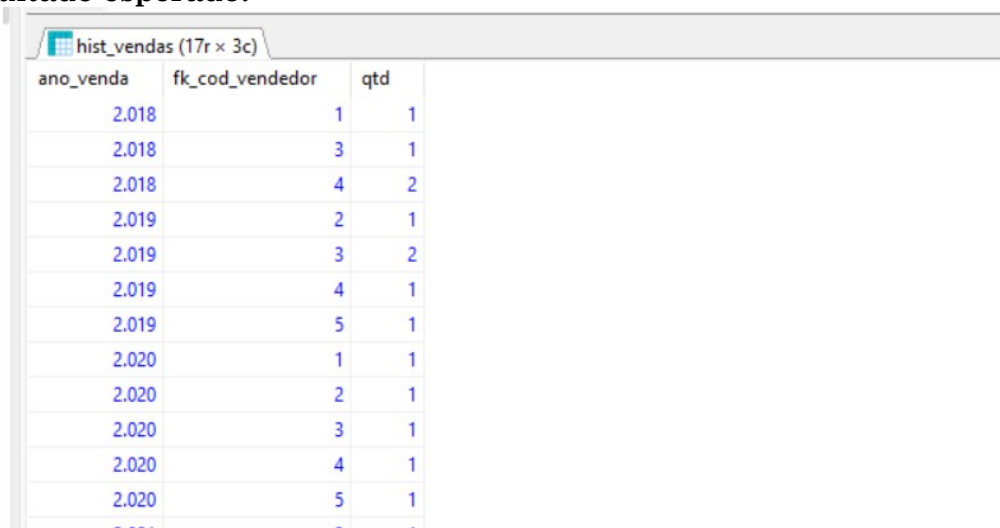
```

1 • USE aulas;
2 • SELECT h.fk_cod_vendedor cod_vendedor, count(*) qtd_vendas
3 FROM hist_vendas h
4 GROUP BY h.fk_cod_vendedor
5

```

cod_vendedor	qtd_vendas
1	5
2	3
3	5
4	8
5	2

## f.6) Mostre a quantidade de vendas de cada vendedor por ano.

**Resultado esperado:**


ano_venda	fk_cod_vendedor	qtd
2.018	1	1
2.018	3	1
2.018	4	2
2.019	2	1
2.019	3	2
2.019	4	1
2.019	5	1
2.020	1	1
2.020	2	1
2.020	3	1
2.020	4	1
2.020	5	1

## f.7) HAVING

Mostre a quantidade de vendas de cada vendedor por ano que seja maior do que 1.

**Resolução e resultado esperado:**

Query 1 x SQL File 3\*

```

1 • USE aulas;
2 • SELECT h.fk_cod_vendedor cod_vendedor, h.ano_venda, count(*) qtd_vendas
3 FROM hist_vendas h
4 GROUP BY h.fk_cod_vendedor, h.ano_venda
5 HAVING qtd_vendas > 1
6

```

Result Grid | Filter Rows: | Export: | Wrap Cell Content: |

	cod_vendedor	ano_venda	qtd_vendas
▶	1	2022	3
	3	2019	2
	4	2018	2
	4	2021	2
	4	2022	2

f.8) Mostre a quantidade de vendas de cada vendedor no ano de 2022.

**Resultado esperado:**

Result Grid | Filter Rows: | Export: | Wrap Cell Content: |

	cod_vendedor	ano_venda	qtd_vendas
▶	1	2022	3
	4	2022	2

f.9) CASE

Mostre nossos clientes classificando seus códigos em menor que 3, igual a 3 e acima de 3.

**Resolução e resultado esperado:**

Query 1 x SQL File 3\*

```

1 • USE aulas;
2 • SELECT cod_cli, fk_cpf, CASE
3     WHEN cod_cli > 3 THEN 'Maior que 3'
4     WHEN cod_cli = 3 THEN 'Igual a 3'
5     ELSE 'Menor que 3'
6     END 'Descrição'
7 FROM cliente c





```

Result Grid | Filter Rows: | Export: | Wrap Cell Content: |

	cod_cli	fk_cpf	Descrição
▶	1	101	Menor que 3
	2	102	Menor que 3
	3	103	Igual a 3
	4	104	Maior que 3
	5	105	Maior que 3

f.10) Mostre nossos vendedores classificando seus códigos em menor que 4, igual a 4 e acima de 4.

**Resultado esperado:**

Result Grid |   Filter Rows:  | Export:  | Wrap Cell Content: 

	cod_vendedor	fk_cpf	Descrição
▶	1	201	Menor que 4
	2	202	Menor que 4
	3	203	Menor que 4
	4	204	Igual a 4
	5	205	Maior que 4

f.11) Select no lugar de coluna

Mostre os produtos com preços de custo acima da média dos preços de custo em nossa base de dados.

**Resolução e resultado esperado:**

Query 1

SQL File 3\*

Limit to 1000 rows

```
1 • SELECT pr.nome_prod, pr.valor_custo
2 FROM produto pr
3 WHERE pr.valor_custo > (SELECT AVG(pr.valor_custo) media
4                        FROM produto pr) ;
```

Result Grid

Filter Rows:

Export:

Wrap Cell Content:

	nome_prod	valor_custo
▶	prod_1	24
	prod_3	26
	prod_5	30

f.12) Select no lugar de coluna

Mostre a soma dos produtos com preços de custo acima da média dos preços de custo em nossa base de dados.

**Resultado esperado:**

	Soma_acima_media
1	80

**Agradecimentos:**

Agradecemos a colaboração e atenção dos amigos que contribuíram para a elaboração deste material de apoio.

Toda e qualquer contribuição será sempre muito bem-vinda.