

# SQL Analytics and Data Science

## Sumário

1	Versão.....	4
2	Introdução .....	5
3	Fundamentos da Linguagem SQL .....	6
3.1	Filtros e Operadores Lógicos .....	6
3.2	Funções de Agregação em SQL .....	8
4	Categorização, Codificação e Binarização de Variáveis.....	14
4.1	Objetivo .....	14
4.2	Tipos de Dados .....	14
4.3	Categorização .....	14
4.4	Codificação (Encoding) .....	15
4.5	Binarização .....	15
4.6	Explorando os Dados.....	16
4.7	Aplicando Binarização com SQL .....	17
4.8	Aplicando Categorização com SQL.....	23
4.9	Label Encoding com SQL .....	27
4.10	Aplicando One-Hot Encoding .....	33
4.11	Criando o Dataset Final Após as Transformações .....	35
4.12	Criando Uma Nova Tabela.....	37
4.13	Salvando os Dados em um Arquivo.....	39
5	Junção de Tabelas .....	41
6	Agregação para Análise de Dados .....	44
7	Window Functions e Subqueries.....	49
8	Análise Exploratória de Dados com SQL .....	54
8.1	Sumarização de Dados .....	54
8.2	Distribuição de Dados .....	55
8.3	Análise Multivariada.....	56
8.4	Identificação de Outliers .....	57
9	Limpeza e Processamento de Dados.....	59
9.1	Identificação de Dados Faltantes .....	59
9.2	Identificação de Ausência de Informação .....	59
9.3	Identificação de Dados Duplicados .....	60
9.4	Detecção de Outliers .....	61

9.5	Tratamento de Valores Ausentes.....	62
9.6	Tratamento de Outliers.....	69
9.7	Label Encoding com Linguagem SQL.....	71
9.8	Relatório de Resumo com Variáveis Quantitativas.....	78
9.9	Relatório de Resumo com Variáveis Quantitativas e Pivot da Tabela .....	78
10	Análise de Dados com SQL .....	79
11	Conclusão .....	92
12	Referências.....	93

## 1 Versão

Este documento foi criado por Cleber Zumba de Souza e pode ser distribuído livremente, desde que se faça menção à fonte.

Versão	Ação	Data
1.0	Criação do documento	12/07/2024

## 2 Introdução

Nos últimos anos, o volume de dados gerados e armazenados pelas organizações tem crescido exponencialmente. A capacidade de extrair insights significativos desses dados tornou-se crucial para a competitividade e o sucesso empresarial. Nesse contexto, o SQL (Structured Query Language) desempenha um papel fundamental, permitindo aos analistas e cientistas de dados acessar e manipular informações de bancos de dados relacionais de maneira eficiente e precisa.

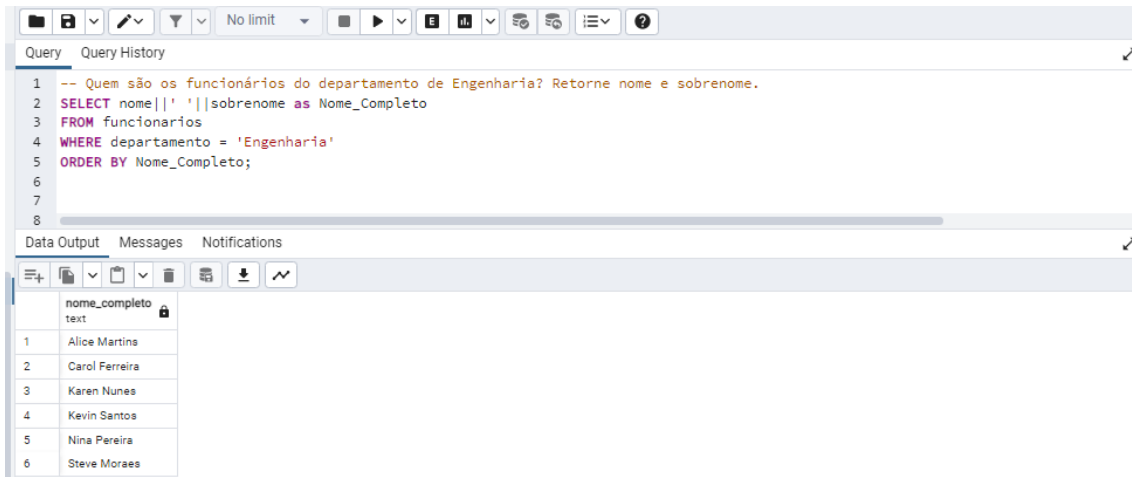
SQL é na sua essência uma ferramenta de análise de dados. Com SQL podemos extrair estatísticas, fazer cálculos e agregações, resumir os dados, gerar relatórios e realizar diversos tipos de análises.

Este artigo explora a linguagem SQL voltada para análise de dados. Ao longo do texto, mostrarei como utilizar SQL para extrair informações valiosas, realizar transformações de dados e gerar relatórios que apoiam decisões estratégicas.

## 3 Fundamentos da Linguagem SQL

### 3.1 Filtros e Operadores Lógicos

Selecione nome e sobrenome de todos os funcionários do departamento de Engenharia.



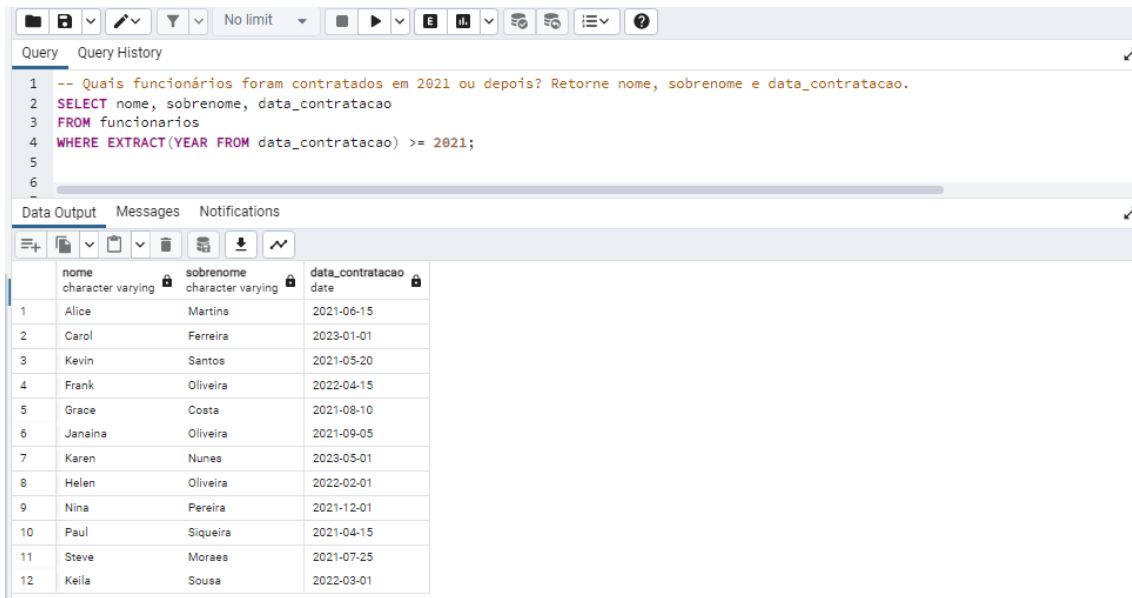
The screenshot shows a SQL IDE interface. The query editor contains the following SQL code:

```
1 -- Quem são os funcionários do departamento de Engenharia? Retorne nome e sobrenome.
2 SELECT nome||' '||sobrenome as Nome_Completo
3 FROM funcionarios
4 WHERE departamento = 'Engenharia'
5 ORDER BY Nome_Completo;
6
7
8
```

The Data Output tab shows the results of the query:

nome_completo
1 Alice Martins
2 Carol Ferreira
3 Karen Nunes
4 Kevin Santos
5 Nina Pereira
6 Steve Moraes

Selecione os funcionários foram contratados em 2021 ou depois. Retorne nome, sobrenome e data de contratação.



The screenshot shows a SQL IDE interface. The query editor contains the following SQL code:

```
1 -- Quais funcionários foram contratados em 2021 ou depois? Retorne nome, sobrenome e data_contratacao.
2 SELECT nome, sobrenome, data_contratacao
3 FROM funcionarios
4 WHERE EXTRACT(YEAR FROM data_contratacao) >= 2021;
5
6
```

The Data Output tab shows the results of the query:

nome	sobrenome	data_contratacao
1 Alice	Martins	2021-06-15
2 Carol	Ferreira	2023-01-01
3 Kevin	Santos	2021-05-20
4 Frank	Oliveira	2022-04-15
5 Grace	Costa	2021-08-10
6 Janaina	Oliveira	2021-09-05
7 Karen	Nunes	2023-05-01
8 Helen	Oliveira	2022-02-01
9 Nina	Pereira	2021-12-01
10 Paul	Siqueira	2021-04-15
11 Steve	Moraes	2021-07-25
12 Keila	Sousa	2022-03-01

Selecione os funcionários recebem salário entre 5000 e 6000? Retorne nome, sobrenome, salário e departamento.

Query Query History	
1	-- Quais funcionários recebem salário entre 5000 e 6000? Retorne nome, sobrenome, salario e departamento.
2	SELECT nome, sobrenome, salario, departamento
3	FROM funcionarios
4	WHERE salario between 5000 and 6000;
5	
6	

Data Output Messages Notifications			
	nome character varying	sobrenome character varying	salario numeric (10,2)
1	Alice	Martins	6000.00
2	Bob	Oliveira	5500.00
3	Josias	Silva	5000.00
4	Janaina	Oliveira	5100.00
5	Jack	Barbosa	5800.00
6	Mallory	Almeida	5200.00
7	Oscar	Oliveira	5700.00
8	Quincy	Teixeira	5300.00
9	Rita	Moreira	5600.00

Selecione funcionários têm nome começando com a letra 'J' ou com a letra 'B'? Retorne nome, sobrenome e departamento.

Query Query History	
1	-- Quais funcionários têm nome começando com a letra J ou com a letra B? Retorne nome, sobrenome e departamento.
2	SELECT nome, sobrenome, departamento
3	FROM funcionarios
4	WHERE nome like 'J%' or nome like 'B%';
5	
6	

Data Output Messages Notifications						
	id [PK] integer	nome character varying	sobrenome character varying	salario numeric (10,2)	departamento character varying	data_contratacao date
1	2	Bob	Oliveira	5500.00	Marketing	2020-03-21
2	4	Josias	Silva	5000.00	RH	2019-11-05
3	9	Janaina	Oliveira	5100.00	RH	2021-09-05
4	10	Jack	Barbosa	5800.00	Marketing	2019-01-10

Há algum funcionário cujo sobrenome tenha as letras 've', seja do departamento de Marketing e o salário seja maior do que 5500?

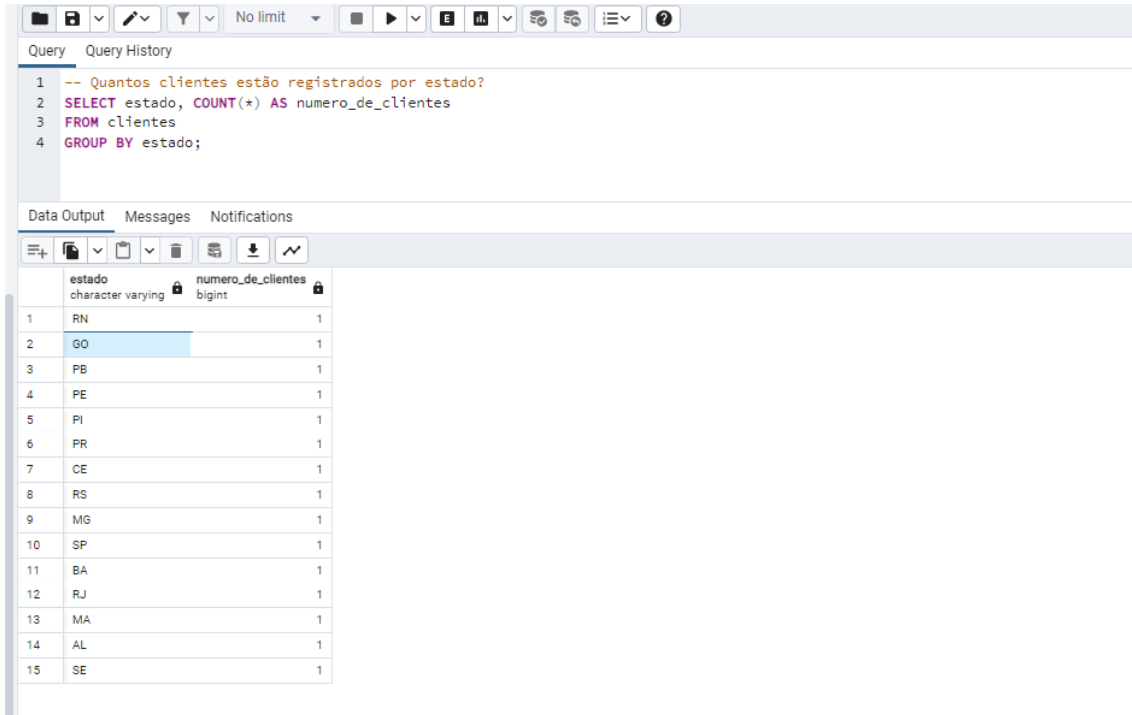
Query Query History	
1	-- Há algum funcionário cujo sobrenome tenha as letras 've', seja do departamento de Marketing e o
2	-- salário seja maior do que 5500?
3	SELECT *
4	FROM funcionarios
5	WHERE sobrenome like '%ve%'
6	AND departamento = 'Marketing'
7	AND salario > 5500;
8	
9	

Data Output Messages Notifications						
	id [PK] integer	nome character varying	sobrenome character varying	salario numeric (10,2)	departamento character varying	data_contratacao date
1	15	Oscar	Oliveira	5700.00	Marketing	2020-06-30

## 3.2 Funções de Agregação em SQL

Selecione clientes estão registrados por estado.



The screenshot shows a SQL IDE interface. The query editor contains the following SQL code:

```
1 -- Quantos clientes estão registrados por estado?
2 SELECT estado, COUNT(*) AS numero_de_clientes
3 FROM clientes
4 GROUP BY estado;
```

The 'Data Output' tab is active, displaying the results of the query. The table has two columns: 'estado' (character varying) and 'numero\_de\_clientes' (bigint). The results are as follows:

	estado	numero_de_clientes
1	RN	1
2	GO	1
3	PB	1
4	PE	1
5	PI	1
6	PR	1
7	CE	1
8	RS	1
9	MG	1
10	SP	1
11	BA	1
12	RJ	1
13	MA	1
14	AL	1
15	SE	1

Selecione idade média dos clientes



The screenshot shows a SQL IDE interface. The query editor contains the following SQL code:

```
1 -- Qual é a idade média dos clientes?
2 SELECT ROUND(AVG(EXTRACT(YEAR FROM AGE(data_nascimento)))) AS idade_media
3 FROM clientes;
4
```

The 'Data Output' tab is active, displaying the results of the query. The table has one column: 'idade\_media' (numeric). The result is as follows:

	idade_media
1	38



Selecione clientes com mais de 30 anos.

The screenshot shows a SQL IDE interface. The query editor contains the following SQL code:

```
1 -- Quantos clientes têm mais de 30 anos?
2 SELECT COUNT(*) AS clientes_mais_30
3 FROM clientes
4 WHERE EXTRACT(YEAR FROM AGE(current_date, data_nascimento)) > 30;
```

The Data Output tab is active, showing the results of the query:

	clientes_mais_30 bigint
1	13

Selecione as 3 cidades com o maior número de clientes

The screenshot shows a SQL IDE interface. The query editor contains the following SQL code:

```
1 -- Quais são as 3 cidades com o maior número de clientes?
2 SELECT cidade, COUNT(*) AS numero_de_clientes
3 FROM clientes
4 GROUP BY cidade
5 ORDER BY numero_de_clientes DESC
6 LIMIT 3;
```

The Data Output tab is active, showing the results of the query:

	cidade character varying	numero_de_clientes bigint
1	Porto Alegre	4
2	Rio de Janeiro	3
3	Recife	2

Selecione clientes que têm um endereço de e-mail registrado

The screenshot shows a SQL IDE interface. The query editor contains the following SQL code:

```
1 -- Quantos clientes têm um endereço de e-mail registrado?
2 SELECT COUNT(*)
3 FROM clientes
4 WHERE email <> '';
```

The Data Output tab is active, showing the results of the query:

	count bigint
1	12

Selecione o valor total de produtos em estoque por categoria.



The screenshot shows a SQL IDE interface. The query editor contains the following SQL code:

```
-- Qual é o valor total de produtos em estoque por categoria?  
SELECT categoria, SUM(preco * quantidade) AS valor_total  
FROM produtos  
GROUP BY categoria;
```

The Data Output tab shows the results of the query:

	categoria	valor_total
1	Categoria 5	7673.00
2	Categoria 4	10550.00
3	Categoria 1	8120.00
4	Categoria 3	7348.80
5	Categoria 2	9812.90

Selecione a quantidade média de produtos em estoque por categoria.



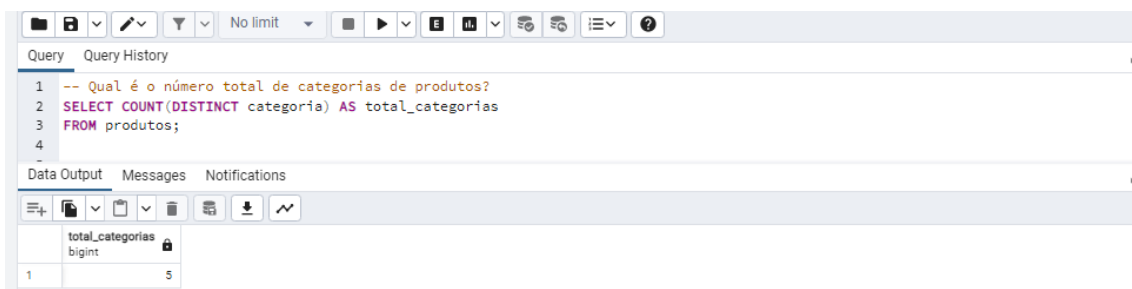
The screenshot shows a SQL IDE interface. The query editor contains the following SQL code:

```
-- Qual é a quantidade média de produtos em estoque por categoria?  
SELECT categoria, ROUND(AVG(quantidade), 0) AS quantidade_media_produtos  
FROM produtos  
GROUP BY categoria  
ORDER BY categoria;
```

The Data Output tab shows the results of the query:

	categoria	quantidade_media_produtos
1	Categoria 1	137
2	Categoria 2	163
3	Categoria 3	160
4	Categoria 4	67
5	Categoria 5	220

Selecione o número total de categorias de produtos.



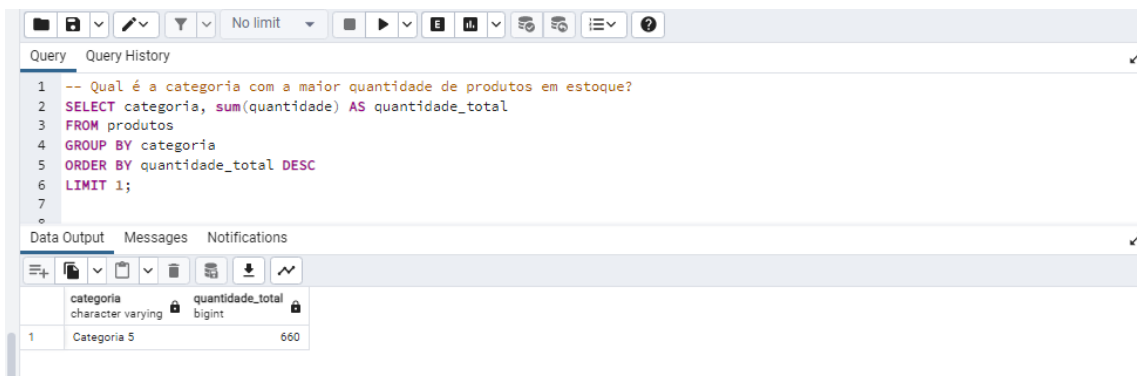
The screenshot shows a SQL IDE interface. The query editor contains the following SQL code:

```
-- Qual é o número total de categorias de produtos?  
SELECT COUNT(DISTINCT categoria) AS total_categorias  
FROM produtos;
```

The Data Output tab shows the results of the query:

	total_categorias
1	5

### Selecione a categoria com a maior quantidade de produtos em estoque



The screenshot shows a SQL query editor with a toolbar at the top. The query is as follows:

```
1 -- Qual é a categoria com a maior quantidade de produtos em estoque?
2 SELECT categoria, sum(quantidade) AS quantidade_total
3 FROM produtos
4 GROUP BY categoria
5 ORDER BY quantidade_total DESC
6 LIMIT 1;
```

Below the query editor, the 'Data Output' tab is active, showing the results of the query:

	categoria character varying	quantidade_total bigint
1	Categoria 5	660

### Selecione o estado com o maior número de fornecedores.



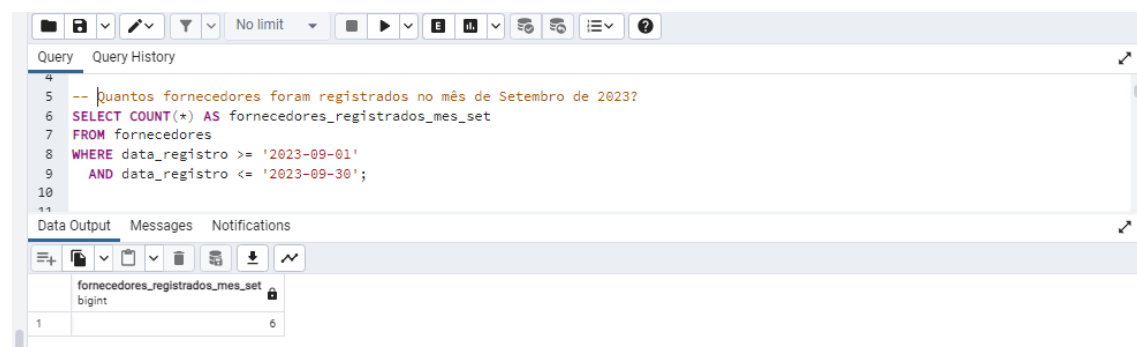
The screenshot shows a SQL query editor with a toolbar at the top. The query is as follows:

```
4
5 -- Qual é o estado com o maior número de fornecedores?
6 SELECT estado
7 FROM fornecedores
8 GROUP BY estado
9 ORDER BY COUNT(*) DESC
10 LIMIT 1;
```

Below the query editor, the 'Data Output' tab is active, showing the results of the query:

	estado character varying
1	GO

### Selecione quantos fornecedores foram registrados no mês de Setembro de 2023.



The screenshot shows a SQL query editor with a toolbar at the top. The query is as follows:

```
4
5 -- Quantos fornecedores foram registrados no mês de Setembro de 2023?
6 SELECT COUNT(*) AS fornecedores_registrados_mes_set
7 FROM fornecedores
8 WHERE data_registro >= '2023-09-01'
9 AND data_registro <= '2023-09-30';
```

Below the query editor, the 'Data Output' tab is active, showing the results of the query:

	fornecedores_registrados_mes_set bigint
1	6

Selecione a média de registros de fornecedores por mês.



The screenshot shows a SQL IDE with a query editor and a results pane. The query is as follows:

```

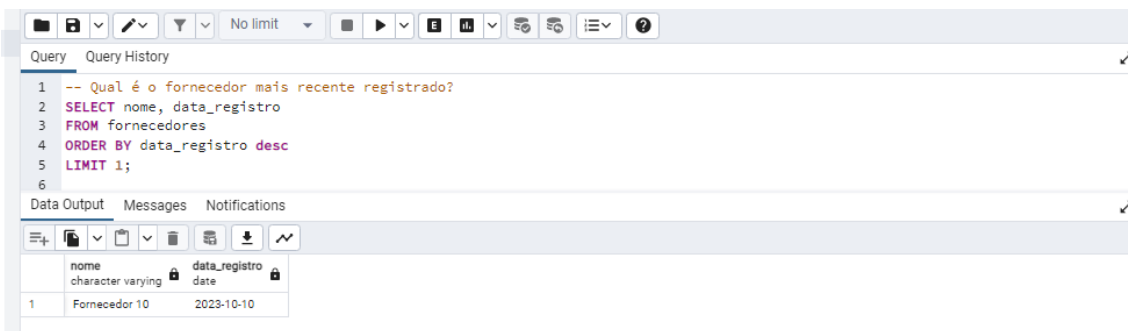
7 -- Qual é a média de registros de fornecedores por mês?
8 SELECT ROUND(AVG(qtd), 0) AS media_registros_por_mes
9 FROM (
10     SELECT EXTRACT(MONTH FROM data_registro) AS mes, COUNT(*) AS qtd
11     FROM fornecedores
12     GROUP BY EXTRACT(MONTH FROM data_registro)
13 ) AS subquery;

```

The results pane shows a table with two columns: 'media\_registros\_por\_mes' (numeric) and a single row with the value 5.

media_registros_por_mes
5

Selecione o fornecedor mais recente registrado. (Esta consulta com ORDER BY ... LIMIT 1 é performática, especialmente em tabelas grandes com índice adequado na coluna 'data\_registro'. Ela permite que o banco de dados use a ordenação e a limitação para encontrar rapidamente o registro desejado).



The screenshot shows a SQL IDE with a query editor and a results pane. The query is as follows:

```

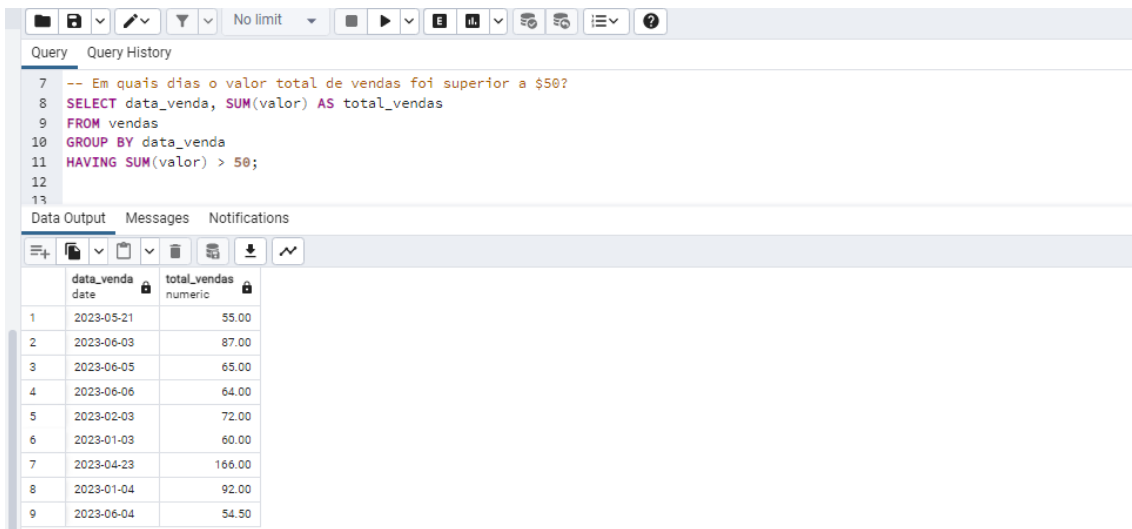
1 -- Qual é o fornecedor mais recente registrado?
2 SELECT nome, data_registro
3 FROM fornecedores
4 ORDER BY data_registro desc
5 LIMIT 1;

```

The results pane shows a table with two columns: 'nome' (character varying) and 'data\_registro' (date). The single row shows 'Fornecedor 10' and '2023-10-10'.

nome	data_registro
Fornecedor 10	2023-10-10

Selecione quais dias o valor total de vendas foi superior a \$50



The screenshot shows a SQL IDE with a query editor and a results pane. The query is as follows:

```

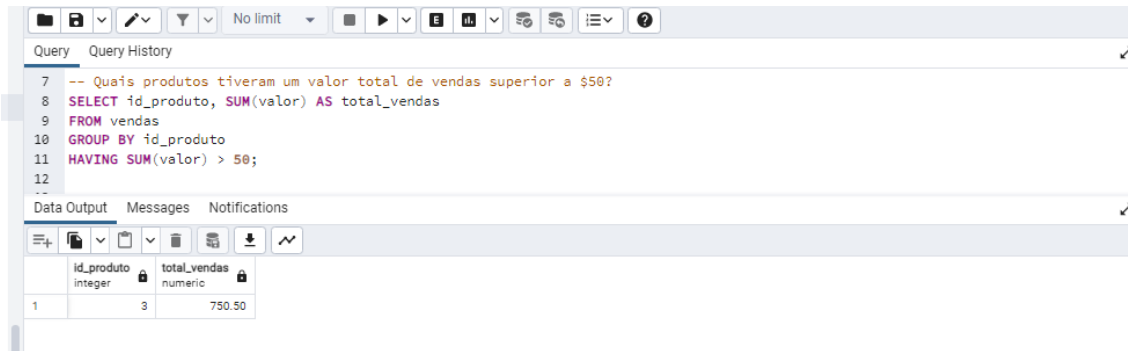
7 -- Em quais dias o valor total de vendas foi superior a $50?
8 SELECT data_venda, SUM(valor) AS total_vendas
9 FROM vendas
10 GROUP BY data_venda
11 HAVING SUM(valor) > 50;

```

The results pane shows a table with two columns: 'data\_venda' (date) and 'total\_vendas' (numeric). The results are as follows:

data_venda	total_vendas
2023-05-21	55.00
2023-06-03	87.00
2023-06-05	65.00
2023-06-06	64.00
2023-02-03	72.00
2023-01-03	60.00
2023-04-23	166.00
2023-01-04	92.00
2023-06-04	54.50

Selecione quais produtos tiveram um valor total de vendas superior a \$50



The screenshot shows a SQL query editor with a query window and a data output window. The query is as follows:

```
-- Quais produtos tiveram um valor total de vendas superior a $50?  
SELECT id_produto, SUM(valor) AS total_vendas  
FROM vendas  
GROUP BY id_produto  
HAVING SUM(valor) > 50;
```

The data output window shows the following results:

id_produto	total_vendas
1	750.50

## 4 Categorização, Codificação e Binarização de Variáveis

### 4.1 Objetivo

O objetivo nesse capítulo é aplicar uma série de transformações aos dados. Os dados de entrada terão diversas variáveis categóricas representadas através de texto e o trabalho é entregar os dados com representação numérica, cenário bastante usual em projetos de Machine Learning quando o Analista de Dados fica responsável pela preparação dos dados.

### 4.2 Tipos de Dados

Tipos de dados podem ser classificados em dois grupos: quantitativos e qualitativos.

**Dados Quantitativos:** São aqueles que expressão uma quantidade e podem ser mensurados em escala numérica.

**Dados Qualitativos:** Representam características que não podem ser medidas em uma escala numérica, mas podem ser categorizadas ou descritas.

### 4.3 Categorização

**Categorização:** No contexto da análise de dados e processamento de dados, refere-se ao processo de transformar dados numéricos contínuos ou discretos em categorias ou grupos discretos.

## 4.4 Codificação (Encoding)

**Codificação ou Encoding:** No contexto da análise de dados e processamento de dados, refere-se ao processo de converter dados categóricos ou textuais em um formato numérico que pode ser usado por algoritmos. Muitos algoritmos requerem que as entradas sejam numéricas, e, portanto os dados categóricos precisam ser transformados antes do treinamento ou análise.

➤ Técnicas de codificação:

- - One-Hot Encoding
- - Label Encoding
- - Frequency or Count Encoding
- - Target Encoding
- - Binary Encoding
- - Embedding Encoding (IA)

## 4.5 Binarização

**Binarização:** No contexto da análise de dados e processamento de dados, refere-se ao processo de converter dados numéricos ou categóricos em formato binário, ou seja, em valores 0 ou 1.

## 4.6 Explorando os Dados

Praticamente todos os dados são do tipo categórico, ou seja, representa alguma coisa nominal ou ordinal. São variáveis que não tem número, embora possa ter número e normalmente são do tipo texto. A variável 'deg\_malig' é a única variável que é do tipo "integer", mas o fato de ter apenas números não significa que a variável seja quantitativa. O dado pode estar como número, mas informação pode ser categórica, ex: categoria 1, categoria 2, categoria3...

Query Query History

1 SELECT \* FROM pacientes;

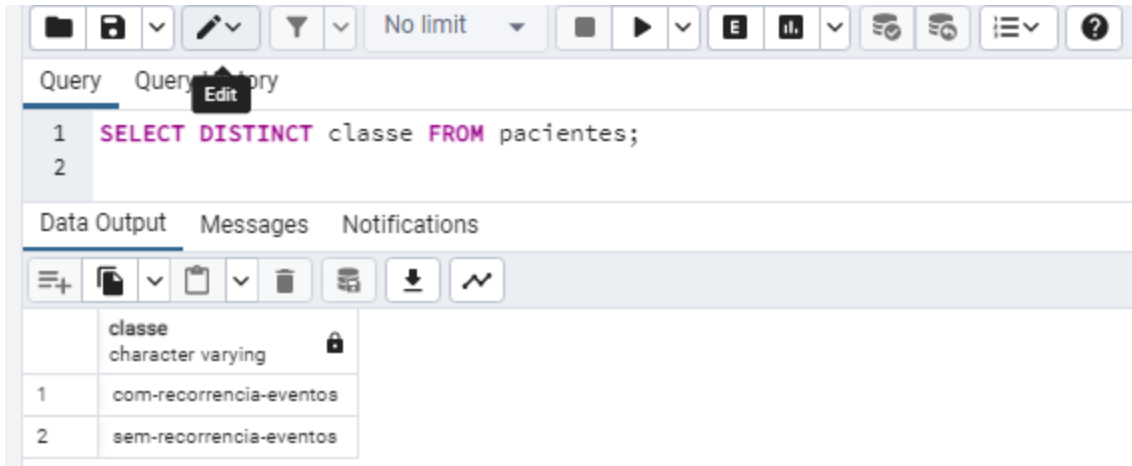
Data Output Messages Notifications

	classe character varying	idade character varying	menopausa character varying	tamanho_tumor character varying	inv_nodes character varying	node_caps character varying	deg_malig integer	seio character varying	quadrante character varying	irradiando character varying
1	sem-recorrencia-eventos	30-39	pré-menopausa	30-34	2-4	não		3	esquerdo	esquerdo_inferior
2	sem-recorrencia-eventos	40-49	pré-menopausa	20-24	0-2	sim		2	direito	direito_superior
3	com-recorrencia-eventos	40-49	pré-menopausa	20-24	0-2	não		2	esquerdo	esquerdo_inferior
4	sem-recorrencia-eventos	60-69	acima_de_40	15-19	0-2	sim		2	direito	esquerdo_superior
5	sem-recorrencia-eventos	40-49	pré-menopausa	0-4	0-2	não		2	direito	direito_inferior
6	sem-recorrencia-eventos	60-69	acima_de_40	15-19	0-2	não		2	esquerdo	esquerdo_inferior
7	sem-recorrencia-eventos	50-59	pré-menopausa	25-29	0-2	não		2	esquerdo	esquerdo_inferior
8	com-recorrencia-eventos	60-69	acima_de_40	20-24	0-2	sim		1	esquerdo	esquerdo_inferior
9	sem-recorrencia-eventos	40-49	pré-menopausa	50-54	0-2	não		2	esquerdo	esquerdo_inferior
10	sem-recorrencia-eventos	40-49	pré-menopausa	20-24	0-2	não		2	direito	esquerdo_superior
11	com-recorrencia-eventos	40-49	pré-menopausa	0-4	0-2	não		3	esquerdo	central
12	sem-recorrencia-eventos	50-59	acima_de_40	25-29	0-2	não		2	esquerdo	esquerdo_inferior
13	sem-recorrencia-eventos	60-69	abaixo_de_40	10-14	0-2	não		1	esquerdo	direito_superior
14	com-recorrencia-eventos	50-59	acima_de_40	25-29	0-2	não		3	esquerdo	direito_superior
15	com-recorrencia-eventos	60-69	pré-menopausa	0-4	2-4	não		3	esquerdo	central
16	sem-recorrencia-eventos	40-49	pré-menopausa	30-34	0-2	sim		3	esquerdo	esquerdo_superior
17	sem-recorrencia-eventos	60-69	abaixo_de_40	30-34	0-2	não		1	esquerdo	esquerdo_inferior
18	sem-recorrencia-eventos	40-49	pré-menopausa	15-19	0-2	não		2	esquerdo	esquerdo_inferior
19	sem-recorrencia-eventos	50-59	pré-menopausa	30-34	0-2	não		3	esquerdo	esquerdo_inferior
20	sem-recorrencia-eventos	60-69	acima_de_40	30-34	0-2	não		3	esquerdo	esquerdo_inferior



## 4.7 Aplicando Binarização com SQL

**Verificando as categorias da variável 'classe'.** Considerando que eu tenho que preparar esse "dataset", eu vou converter o dado sem modificar a informação.



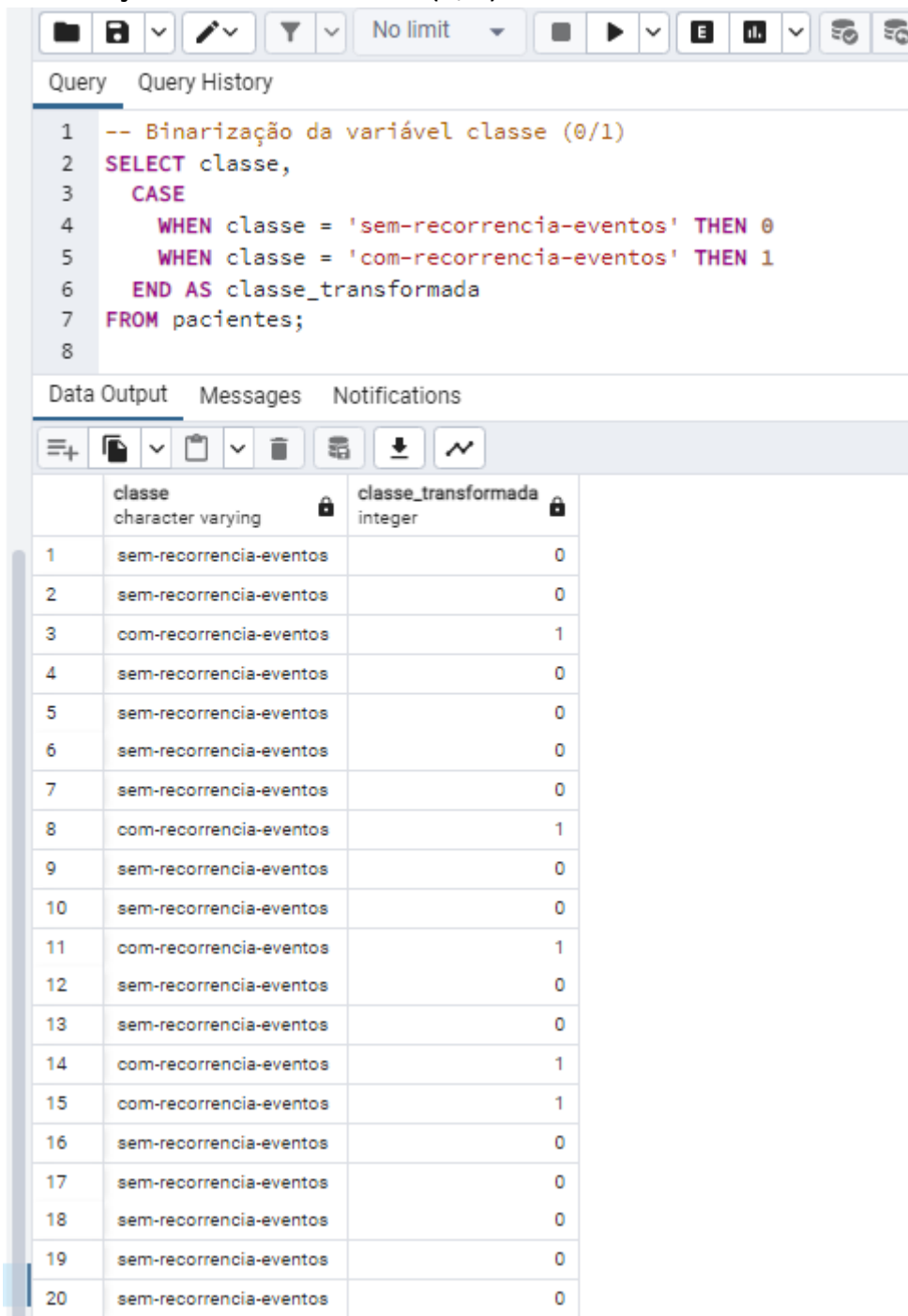
The screenshot shows a SQL query editor interface. The query editor has a toolbar at the top with icons for file operations, editing, and execution. Below the toolbar, the query editor shows the following SQL query:

```
1 SELECT DISTINCT classe FROM pacientes;  
2
```

The query is executed, and the results are displayed in the "Data Output" tab. The results are shown in a table with the following structure:

	classe character varying
1	com-recorrencia-eventos
2	sem-recorrencia-eventos

## Binarização da variável classe (0/1)



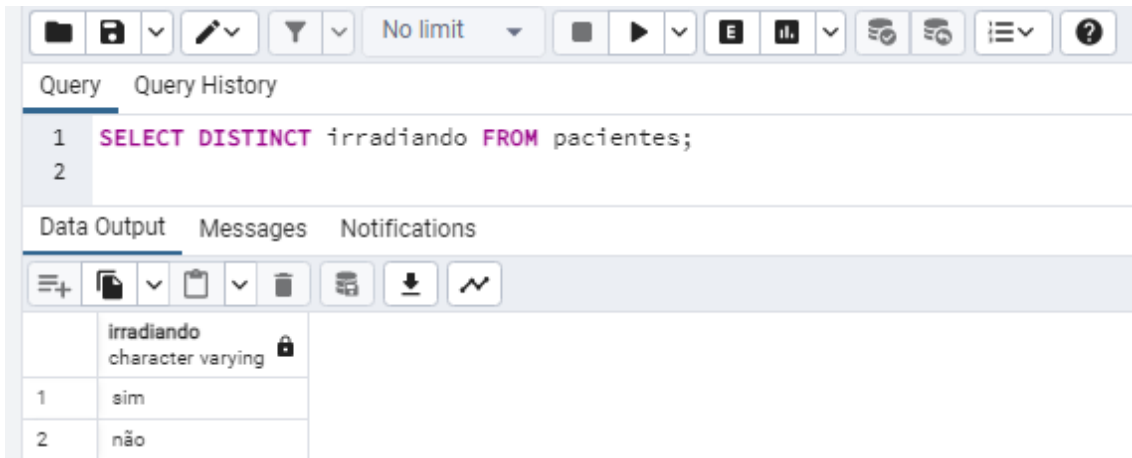
The screenshot shows a SQL IDE interface. At the top, there's a toolbar with icons for file operations, query execution, and settings. Below the toolbar, the 'Query' tab is active, displaying a SQL query. The query is a CASE statement that binarizes the 'classe' variable from the 'pacientes' table. The 'Data Output' tab is also visible, showing the results of the query in a table format. The table has two columns: 'classe' (character varying) and 'classe\_transformada' (integer). The results show 20 rows of data, where 'classe' values are either 'sem-recorrencia-eventos' or 'com-recorrencia-eventos', and 'classe\_transformada' values are 0 or 1 respectively.

```
1 -- Binarização da variável classe (0/1)
2 SELECT classe,
3     CASE
4         WHEN classe = 'sem-recorrencia-eventos' THEN 0
5         WHEN classe = 'com-recorrencia-eventos' THEN 1
6     END AS classe_transformada
7 FROM pacientes;
8
```

	classe character varying	classe_transformada integer
1	sem-recorrencia-eventos	0
2	sem-recorrencia-eventos	0
3	com-recorrencia-eventos	1
4	sem-recorrencia-eventos	0
5	sem-recorrencia-eventos	0
6	sem-recorrencia-eventos	0
7	sem-recorrencia-eventos	0
8	com-recorrencia-eventos	1
9	sem-recorrencia-eventos	0
10	sem-recorrencia-eventos	0
11	com-recorrencia-eventos	1
12	sem-recorrencia-eventos	0
13	sem-recorrencia-eventos	0
14	com-recorrencia-eventos	1
15	com-recorrencia-eventos	1
16	sem-recorrencia-eventos	0
17	sem-recorrencia-eventos	0
18	sem-recorrencia-eventos	0
19	sem-recorrencia-eventos	0
20	sem-recorrencia-eventos	0

Modifiquei o dado mas não perdi a informação. Binarização só funciona quando se tem apenas duas categorias, pois só posso ter valores zero (0) ou um (1) – ocorrência ou não ocorrência do evento.

**Verificando as categorias da variável 'irradiando'**. Considerando que eu tenho que preparar esse "dataset", eu vou converter o dado sem modificar a informação.



The screenshot shows a SQL query editor interface. At the top, there is a toolbar with various icons for file operations, editing, and execution. Below the toolbar, the 'Query' tab is active, displaying the following SQL query:

```
1 SELECT DISTINCT irradiando FROM pacientes;
2
```

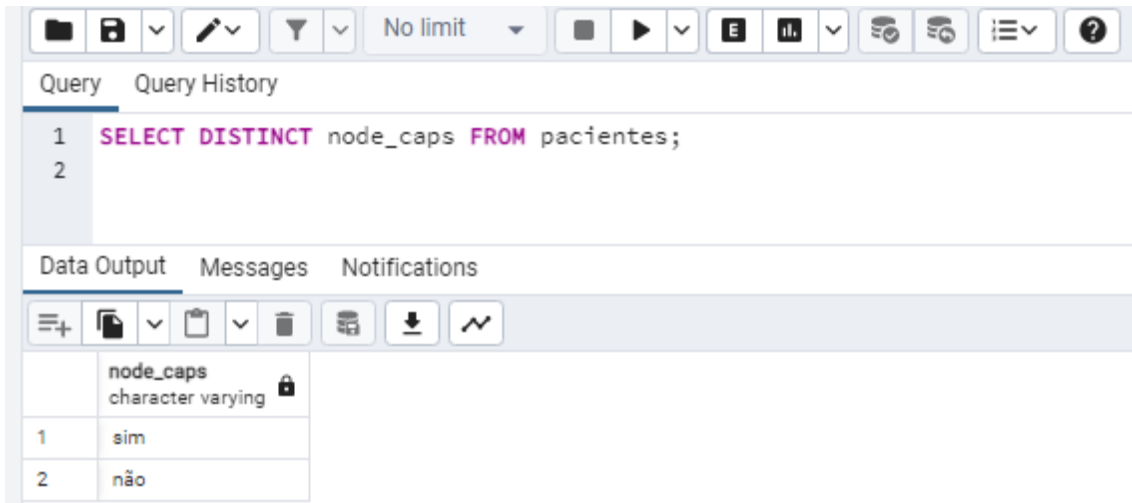
Below the query editor, the 'Data Output' tab is active, showing the results of the query. The output is a table with two columns: 'irradiando' (character varying) and a lock icon. The data is as follows:

	irradiando character varying
1	sim
2	não

## Binarização da variável irradiando (0/1)

Query		Query History	
1	SELECT irradiando,		
2	CASE		
3	WHEN irradiando = 'não' THEN 0		
4	WHEN irradiando = 'sim' THEN 1		
5	END AS irradiando_transformado		
6	FROM pacientes;		
7			
Data Output		Messages	
		Notifications	
	irradiando character varying	irradiando_transformado integer	
1	não	0	
2	não	0	
3	não	0	
4	não	0	
5	não	0	
6	sim	1	
7	não	0	
8	não	0	
9	sim	1	
10	não	0	
11	sim	1	
12	não	0	
13	não	0	
14	sim	1	
15	não	0	
16	não	0	
17	não	0	
18	sim	1	
19	não	0	
20	não	0	

**Verificando as categorias da variável 'node\_caps'.** Considerando que eu tenho que preparar esse "dataset", eu vou converter o dado sem modificar a informação.



The screenshot shows a SQL query editor interface. The top toolbar includes icons for file operations, query execution, and settings. The query editor displays the following SQL statement:

```
1 SELECT DISTINCT node_caps FROM pacientes;
2
```

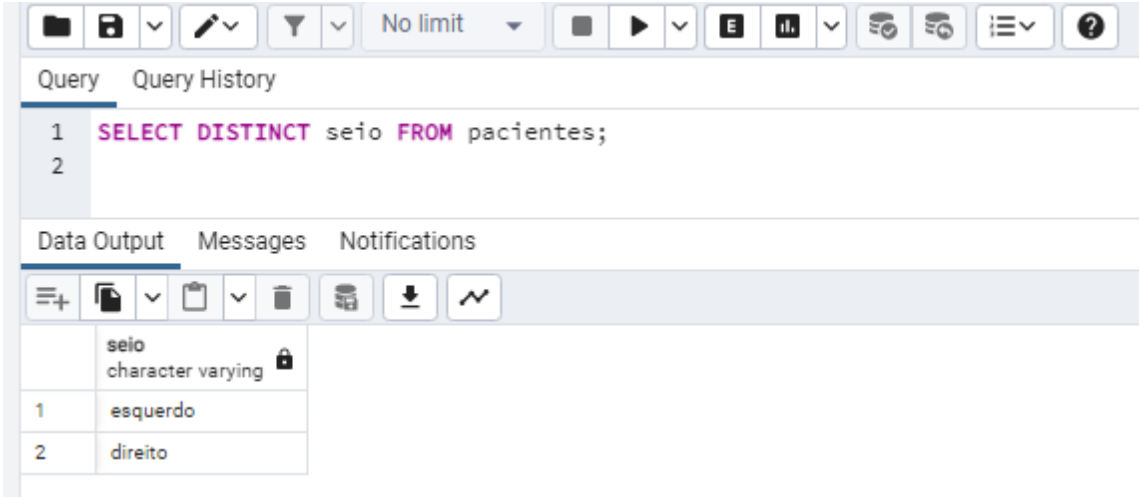
Below the query editor, the 'Data Output' tab is active, showing the results of the query. The results are displayed in a table with two columns: 'node\_caps' and 'character varying'. The table contains two rows of data:

	node_caps	character varying
1	sim	
2	não	

## Binarização da variável node\_caps (0/1)

Query		Query History	
1	SELECT node_caps,		
2	CASE		
3	WHEN node_caps = 'não' THEN 0		
4	WHEN node_caps = 'sim' THEN 1		
5	END AS node_caps_transformado		
6	FROM pacientes;		
7			
Data Output		Messages	
		Notifications	
	node_caps character varying	node_caps_transformado integer	
1	não	0	
2	sim	1	
3	não	0	
4	sim	1	
5	não	0	
6	não	0	
7	não	0	
8	sim	1	
9	não	0	
10	não	0	
11	não	0	
12	não	0	
13	não	0	
14	não	0	
15	não	0	
16	sim	1	
17	não	0	
18	não	0	
19	não	0	
20	não	0	

## 4.8 Aplicando Categorização com SQL



The screenshot shows a SQL IDE interface. At the top, there is a toolbar with various icons for file operations, editing, and execution. Below the toolbar, there are tabs for 'Query' and 'Query History'. The 'Query' tab is active, displaying the following SQL query:

```
1 SELECT DISTINCT seio FROM pacientes;
2
```

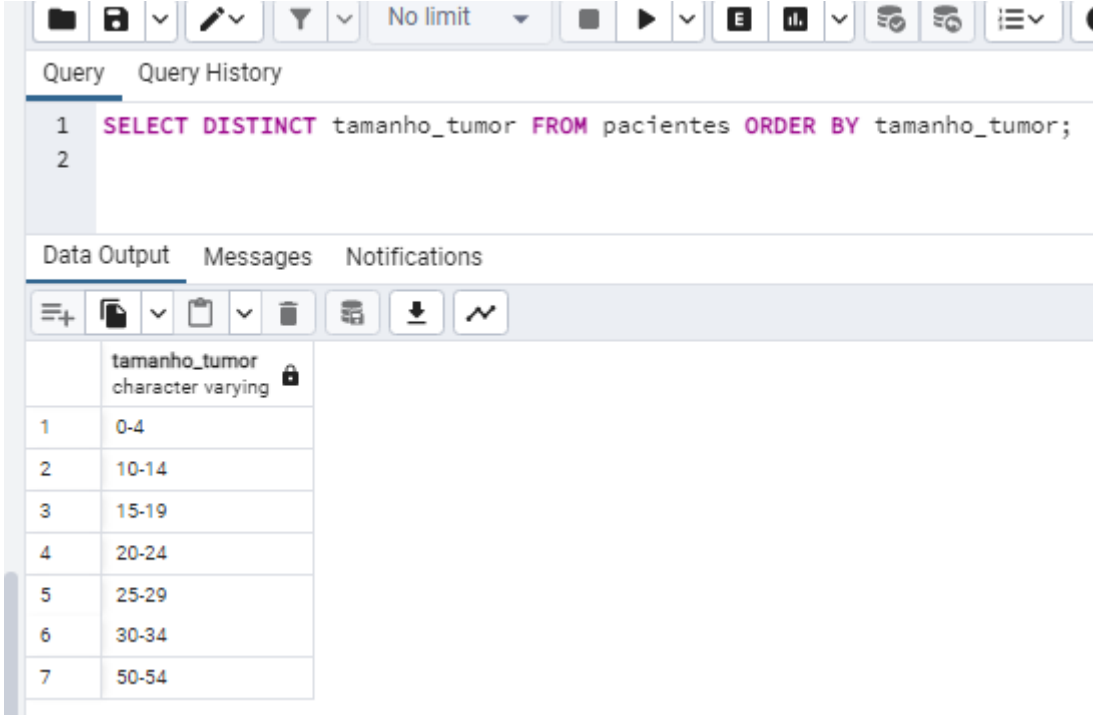
Below the query editor, there are tabs for 'Data Output', 'Messages', and 'Notifications'. The 'Data Output' tab is active, showing a table with the results of the query. The table has two columns: 'seio' and 'character varying'. The data is as follows:

	seio	character varying
1	esquerdo	
2	direito	

## Categorização da variável seio (E/D)

Query		Query History	
1	SELECT seio,		
2	CASE		
3	WHEN seio = 'esquerdo' THEN 'E'		
4	WHEN seio = 'direito' THEN 'D'		
5	END AS seio_categorico		
6	FROM pacientes;		
7			
Data Output		Messages	
		Notifications	
	seio character varying	seio_categorico text	
1	esquerdo	E	
2	direito	D	
3	esquerdo	E	
4	direito	D	
5	direito	D	
6	esquerdo	E	
7	esquerdo	E	
8	esquerdo	E	
9	esquerdo	E	
10	direito	D	
11	esquerdo	E	
12	esquerdo	E	
13	esquerdo	E	
14	esquerdo	E	
15	esquerdo	E	
16	esquerdo	E	
17	esquerdo	E	
18	esquerdo	E	
19	esquerdo	E	
20	esquerdo	E	





The screenshot shows a SQL query editor interface. At the top, there is a toolbar with various icons for file operations, query execution, and settings. Below the toolbar, the 'Query' tab is active, displaying the following SQL query:

```
1 SELECT DISTINCT tamanho_tumor FROM pacientes ORDER BY tamanho_tumor;
2
```

Below the query editor, the 'Data Output' tab is active, showing the results of the query in a table. The table has two columns: 'tamanho\_tumor' (character varying) and a lock icon. The results are as follows:

	tamanho_tumor
1	0-4
2	10-14
3	15-19
4	20-24
5	25-29
6	30-34
7	50-54

## Categorização da variável tamanho\_tumor (6 Categorias)

Query Query History

```

1 SELECT tamanho_tumor,
2 CASE
3 WHEN tamanho_tumor = '0-4' OR tamanho_tumor = '5-9' THEN 'Muito Pequeno'
4 WHEN tamanho_tumor = '10-14' OR tamanho_tumor = '15-19' THEN 'Pequeno'
5 WHEN tamanho_tumor = '20-24' OR tamanho_tumor = '25-29' THEN 'Medio'
6 WHEN tamanho_tumor = '30-34' OR tamanho_tumor = '35-39' THEN 'Grande'
7 WHEN tamanho_tumor = '40-44' OR tamanho_tumor = '45-49' THEN 'Muito Grande'
8 WHEN tamanho_tumor = '50-54' OR tamanho_tumor = '55-59' THEN 'Tratamento Urgente'
9 END AS tamanho_tumor_categorizado
10 FROM pacientes;
11

```

Data Output Messages Notifications

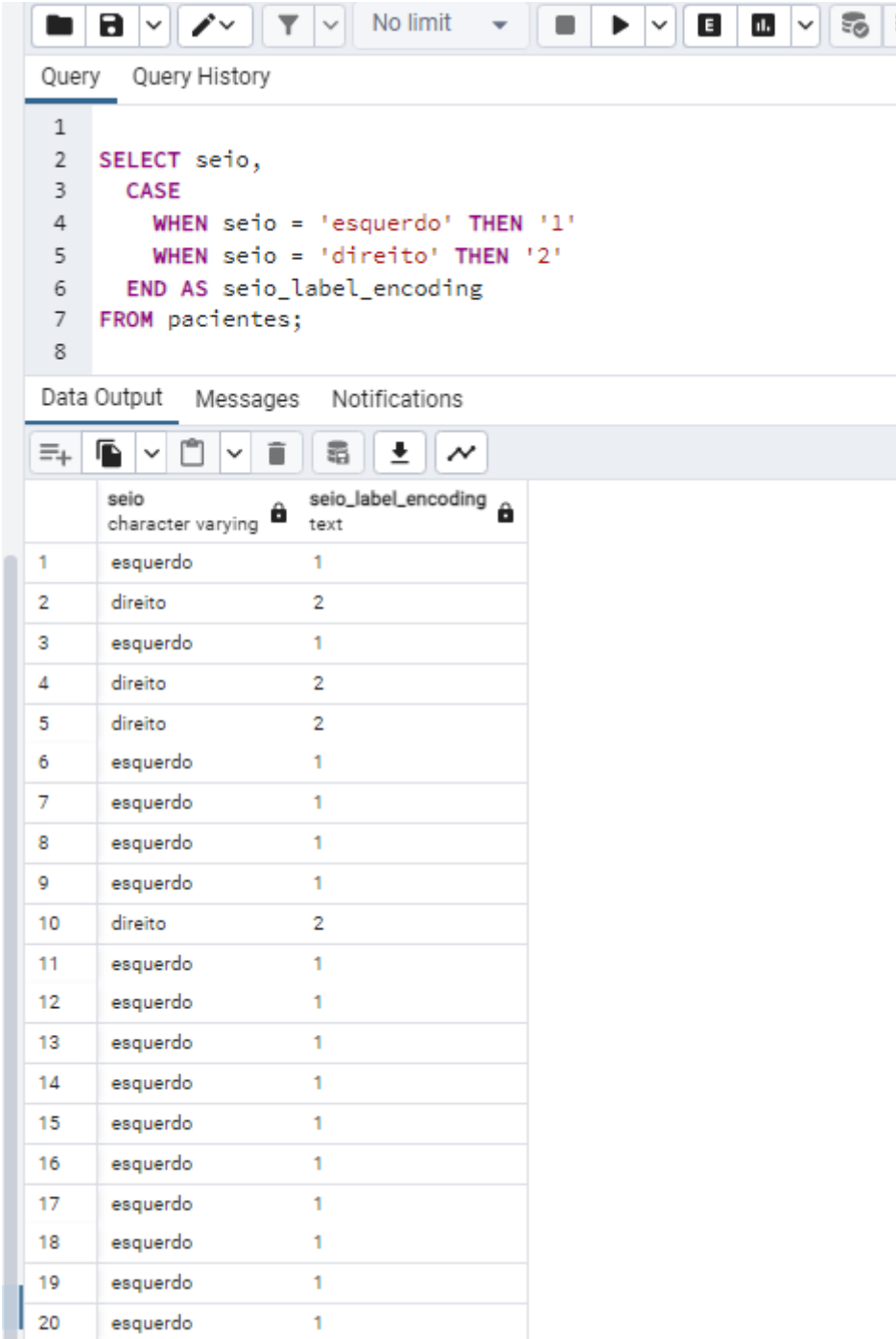
	tamanho_tumor character varying	tamanho_tumor_categorizado text
1	30-34	Grande
2	20-24	Medio
3	20-24	Medio
4	15-19	Pequeno
5	0-4	Muito Pequeno
6	15-19	Pequeno
7	25-29	Medio
8	20-24	Medio
9	50-54	Tratamento Urgente
10	20-24	Medio
11	0-4	Muito Pequeno
12	25-29	Medio
13	10-14	Pequeno
14	25-29	Medio
15	0-4	Muito Pequeno
16	30-34	Grande
17	30-34	Grande
18	15-19	Pequeno
19	30-34	Grande
20	30-34	Grande

## 4.9 Label Encoding com SQL

Label Encoding é uma técnica utilizada para converter variáveis categóricas em uma forma numérica que modelos de Machine Learning podem entender.

Em Label Encoding, a cada categoria é atribuído um valor único entre 1 e  $n-1$  (onde  $n$  é o número de categorias para a variável).

## Label Encoding da variável seio (1/2)



Query

```
1  
2 SELECT seio,  
3 CASE  
4 WHEN seio = 'esquerdo' THEN '1'  
5 WHEN seio = 'direito' THEN '2'  
6 END AS seio_label_encoding  
7 FROM pacientes;  
8
```

Data Output

	seio character varying	seio_label_encoding text
1	esquerdo	1
2	direito	2
3	esquerdo	1
4	direito	2
5	direito	2
6	esquerdo	1
7	esquerdo	1
8	esquerdo	1
9	esquerdo	1
10	direito	2
11	esquerdo	1
12	esquerdo	1
13	esquerdo	1
14	esquerdo	1
15	esquerdo	1
16	esquerdo	1
17	esquerdo	1
18	esquerdo	1
19	esquerdo	1
20	esquerdo	1

## Label Encoding da variável seio (1/2)

Query Query History

```

1  -- Label Encoding da variável seio (1/2)
2  SELECT
3      CASE
4          WHEN seio = 'esquerdo' THEN '1'
5          WHEN seio = 'direito' THEN '2'
6      END AS seio
7  FROM pacientes;

```

Data Output Messages Notifications

	seio text
1	1
2	2
3	1
4	2
5	2
6	1
7	1
8	1
9	1
10	2
11	1
12	1
13	1
14	1
15	1
16	1
17	1
18	1
19	1
20	1

## Label Encoding da variável tamanho\_tumor (6 Categorias)

Query Query History

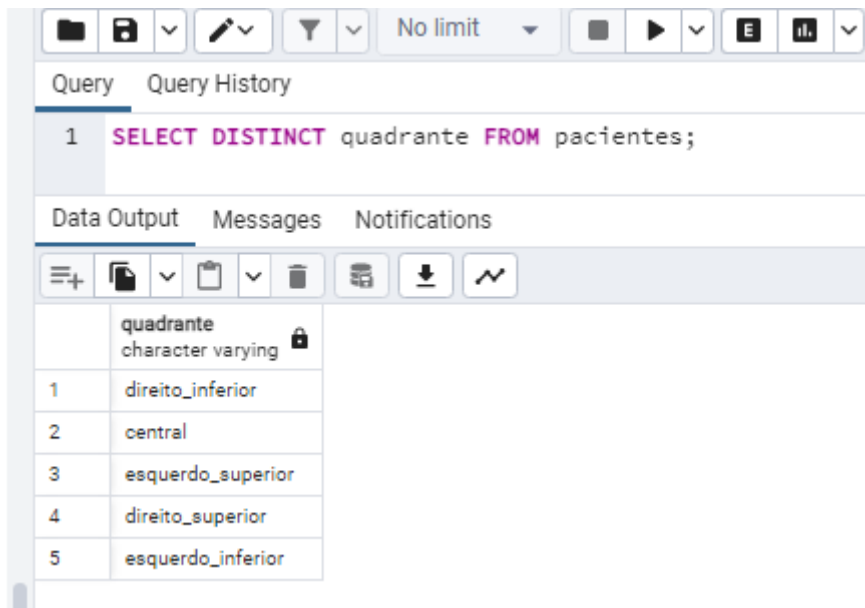
```

1  -- Label Encoding da variável tamanho_tumor (6 Categorias)
2  SELECT
3      CASE
4          WHEN tamanho_tumor = '0-4' OR tamanho_tumor = '5-9' THEN '1'
5          WHEN tamanho_tumor = '10-14' OR tamanho_tumor = '15-19' THEN '2'
6          WHEN tamanho_tumor = '20-24' OR tamanho_tumor = '25-29' THEN '3'
7          WHEN tamanho_tumor = '30-34' OR tamanho_tumor = '35-39' THEN '4'
8          WHEN tamanho_tumor = '40-44' OR tamanho_tumor = '45-49' THEN '5'
9          WHEN tamanho_tumor = '50-54' OR tamanho_tumor = '55-59' THEN '6'
10     END AS tamanho_tumor
11 FROM pacientes;

```

Data Output Messages Notifications

	tamanho_tumor text
1	4
2	3
3	3
4	2
5	1
6	2
7	3
8	3
9	6
10	3
11	1
12	3
13	2
14	3
15	1
16	4
17	4
18	2
19	4
20	4



The screenshot shows a SQL query editor interface. At the top, there is a toolbar with icons for file operations, query execution, and settings. Below the toolbar, the 'Query' tab is active, displaying a single SQL query: `1 SELECT DISTINCT quadrante FROM pacientes;`. The 'Data Output' tab is also visible, showing the results of the query in a table format. The table has two columns: 'quadrante' (character varying) and a lock icon. The results are as follows:

	quadrante character varying
1	direito_inferior
2	central
3	esquerdo_superior
4	direito_superior
5	esquerdo_inferior

## Label Encoding da variável quadrante (1,2,3,4,5)

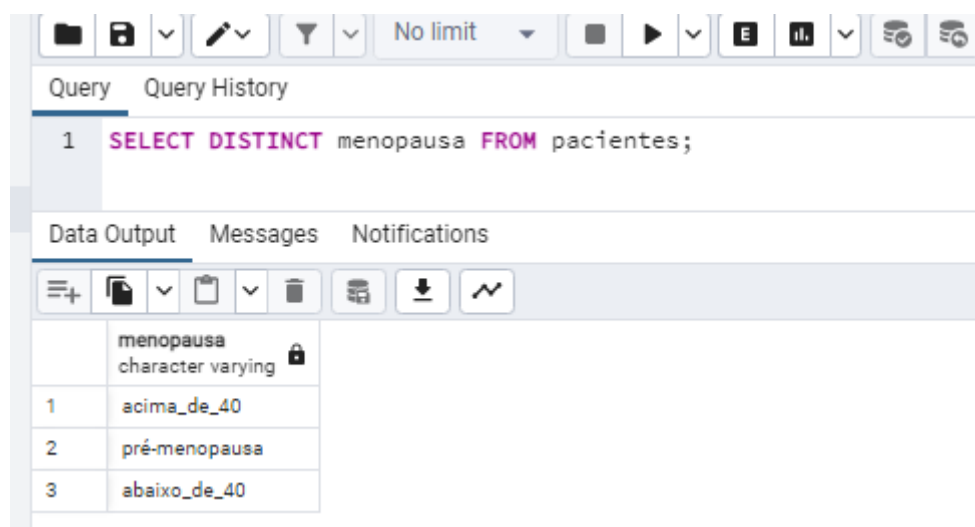
<



## 4.10 Aplicando One-Hot Encoding

One-Hot Encoding é uma técnica utilizada para converter variáveis categóricas em uma forma numérica que modelos de Machine Learning podem entender.

Em One-Hot Encoding para cada categoria da variável categórica, é criada uma nova coluna binária chamada de variável dummy. Para um dado registro, a coluna correspondente à sua categoria recebe o valor 1 e todas as outras recebem o valor 0.



The screenshot shows a SQL query editor interface. At the top, there is a toolbar with icons for file operations, query execution, and settings. Below the toolbar, the 'Query' tab is active, displaying the following SQL query:

```
1 SELECT DISTINCT menopausa FROM pacientes;
```

Below the query editor, the 'Data Output' tab is active, showing the results of the query. The results are displayed in a table with the following structure:

	menopausa character varying
1	acima_de_40
2	pré-menopausa
3	abaixo_de_40

## One-Hot Encoding (criação de variáveis dummy)

Query Query History

```

1  -- One-Hot Encoding (criação de variáveis dummy)
2  SELECT
3      menopausa,
4      CASE
5          WHEN menopausa = 'acima_de_40' THEN 1
6          ELSE 0
7      END AS acima_de_40,
8      CASE
9          WHEN menopausa = 'pré-menopausa' THEN 1
10         ELSE 0
11     END AS pre_menopausa,
12     CASE
13         WHEN menopausa = 'abaixo_de_40' THEN 1
14         ELSE 0
15     END AS abaixo_de_40
16 FROM pacientes;

```

Data Output Messages Notifications

	menopausa character varying	acima_de_40 integer	pre_menopausa integer	abaixo_de_40 integer
1	pré-menopausa	0	1	0
2	pré-menopausa	0	1	0
3	pré-menopausa	0	1	0
4	acima_de_40	1	0	0
5	pré-menopausa	0	1	0
6	acima_de_40	1	0	0
7	pré-menopausa	0	1	0
8	acima_de_40	1	0	0
9	pré-menopausa	0	1	0
10	pré-menopausa	0	1	0
11	pré-menopausa	0	1	0
12	acima_de_40	1	0	0
13	abaixo_de_40	0	0	1
14	acima_de_40	1	0	0
15	pré-menopausa	0	1	0
16	pré-menopausa	0	1	0
17	abaixo_de_40	0	0	1
18	pré-menopausa	0	1	0
19	pré-menopausa	0	1	0
20	acima_de_40	1	0	0

## 4.11 Criando o Dataset Final Após as Transformações

Query com todas as transformações SQL

```
1  -- Query com todas as transformações
2  SELECT
3      CASE
4          WHEN classe = 'sem-recorrencia-eventos' THEN 0
5          WHEN classe = 'com-recorrencia-eventos' THEN 1
6      END AS classe,
7      CASE
8          WHEN tamanho_tumor = '0-4' OR tamanho_tumor = '5-9' THEN '1'
9          WHEN tamanho_tumor = '10-14' OR tamanho_tumor = '15-19' THEN '2'
10         WHEN tamanho_tumor = '20-24' OR tamanho_tumor = '25-29' THEN '3'
11         WHEN tamanho_tumor = '30-34' OR tamanho_tumor = '35-39' THEN '4'
12         WHEN tamanho_tumor = '40-44' OR tamanho_tumor = '45-49' THEN '5'
13         WHEN tamanho_tumor = '50-54' OR tamanho_tumor = '55-59' THEN '6'
14     END AS tamanho_tumor,
15     CASE
16         WHEN node_caps = 'não' THEN 0
17         WHEN node_caps = 'sim' THEN 1
18     ELSE 2
19     END AS node_caps,
20     deg_malig,
21     CASE
22         WHEN seio = 'esquerdo' THEN '1'
23         WHEN seio = 'direito' THEN '2'
24     END AS seio,
25     CASE
26         WHEN quadrante = 'esquerdo_inferior' THEN 1
27         WHEN quadrante = 'direito_superior' THEN 2
28         WHEN quadrante = 'esquerdo_superior' THEN 3
29         WHEN quadrante = 'direito_inferior' THEN 4
30         WHEN quadrante = 'central' THEN 5
31     ELSE 6
32     END AS quadrante,
33     CASE
34         WHEN irradiando = 'não' THEN 0
35         WHEN irradiando = 'sim' THEN 1
36     END AS irradiando,
37     CASE
38         WHEN menopausa = 'acima_de_40' THEN 1
39     ELSE 0
40     END AS acima_de_40,
41     CASE
42         WHEN menopausa = 'pré-menopausa' THEN 1
43     ELSE 0
44     END AS pre_menopausa,
45     CASE
46         WHEN menopausa = 'abaixo_de_40' THEN 1
47     ELSE 0
48     END AS abaixo_de_40
49 FROM pacientes;
50
```

Total rows: 20 of 20    Query complete 00:00:00.127

Data Output Messages Notifications												
	classe integer	tamanho_tumor text	node_caps integer	deg_malign integer	seio text	quadrante integer	irradiando integer	acima_de_40 integer	pre_menopausa integer	abaixo_de_40 integer		
1	0	4	0	3	1	1	0	0	1	0		
2	0	3	1	2	2	2	0	0	1	0		
3	1	3	0	2	1	1	0	0	1	0		
4	0	2	1	2	2	3	0	1	0	0		
5	0	1	0	2	2	4	0	0	1	0		
6	0	2	0	2	1	1	1	1	0	0		
7	0	3	0	2	1	1	0	0	1	0		
8	1	3	1	1	1	1	0	1	0	0		
9	0	6	0	2	1	1	1	0	1	0		
10	0	3	0	2	2	3	0	0	1	0		
11	1	1	0	3	1	5	1	0	1	0		
12	0	3	0	2	1	1	0	1	0	0		
13	0	2	0	1	1	2	0	0	0	1		
14	1	3	0	3	1	2	1	1	0	0		
15	1	1	0	3	1	5	0	0	1	0		
16	0	4	1	3	1	3	0	0	1	0		
17	0	4	0	1	1	1	0	0	0	1		
18	0	2	0	2	1	1	1	0	1	0		
19	0	4	0	3	1	1	0	0	1	0		
20	0	4	0	3	1	1	0	1	0	0		

## 4.12 Criando Uma Nova Tabela

Criando uma nova tabela para não perder os dados do formato alterado.

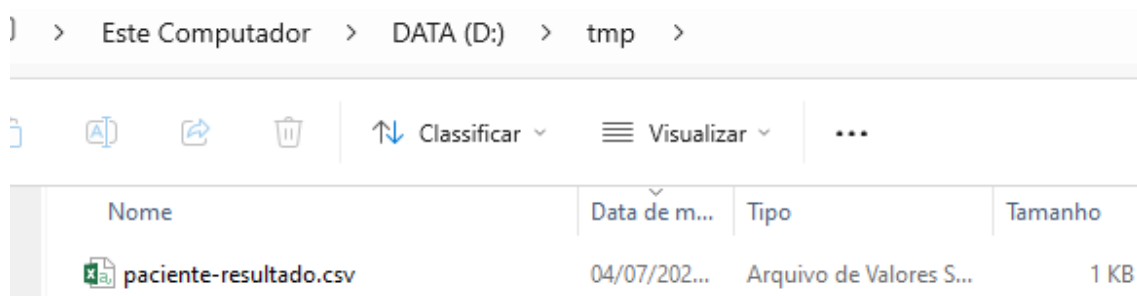
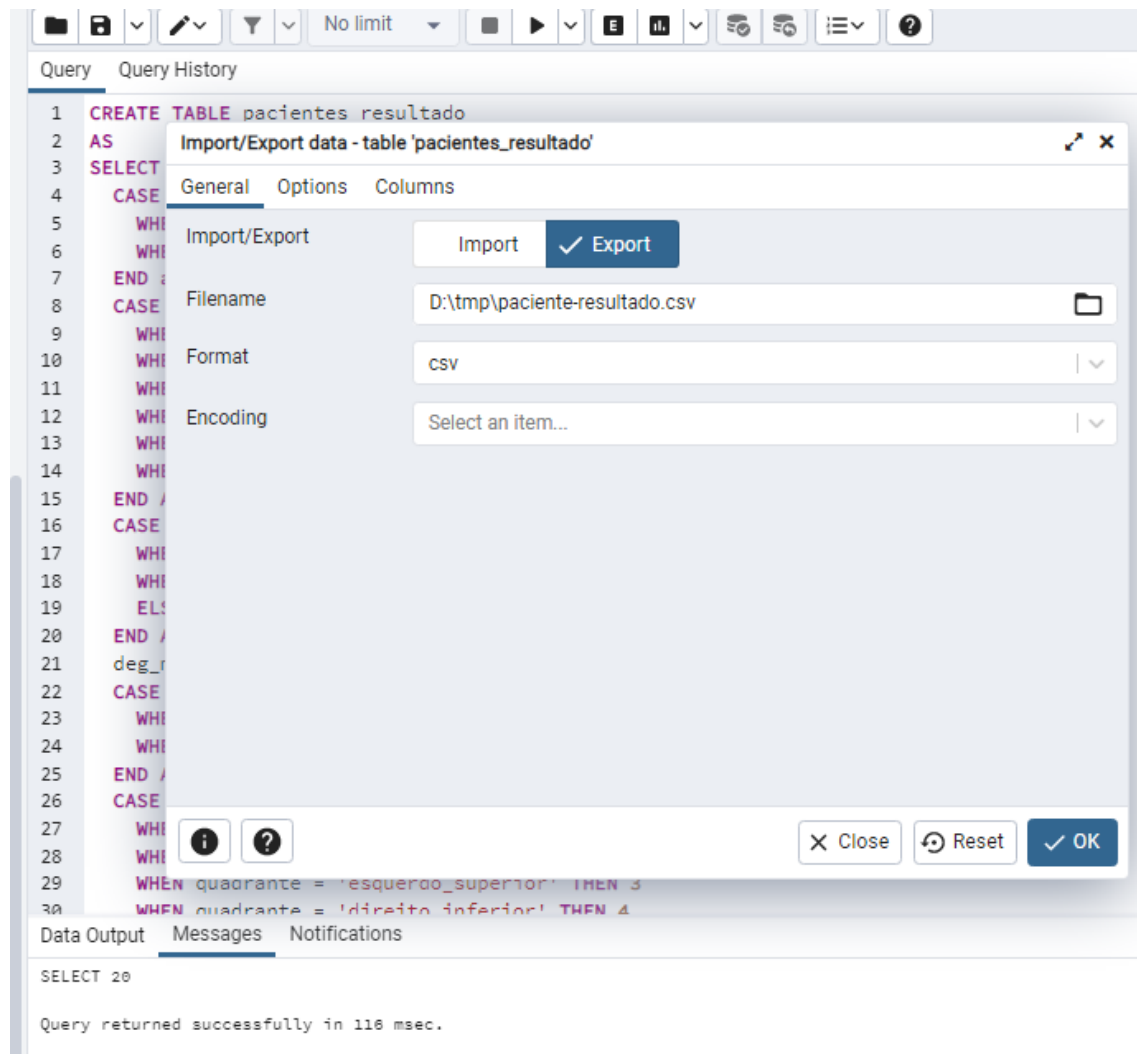
Query	Query History
1	CREATE TABLE pacientes_resultado
2	AS
3	SELECT
4	CASE
5	WHEN classe = 'sem-recorrencia-eventos' THEN 0
6	WHEN classe = 'com-recorrencia-eventos' THEN 1
7	END AS classe,
8	CASE
9	WHEN tamanho_tumor = '0-4' OR tamanho_tumor = '5-9' THEN '1'
10	WHEN tamanho_tumor = '10-14' OR tamanho_tumor = '15-19' THEN '2'
11	WHEN tamanho_tumor = '20-24' OR tamanho_tumor = '25-29' THEN '3'
12	WHEN tamanho_tumor = '30-34' OR tamanho_tumor = '35-39' THEN '4'
13	WHEN tamanho_tumor = '40-44' OR tamanho_tumor = '45-49' THEN '5'
14	WHEN tamanho_tumor = '50-54' OR tamanho_tumor = '55-59' THEN '6'
15	END AS tamanho_tumor,
16	CASE
17	WHEN node_caps = 'não' THEN 0
18	WHEN node_caps = 'sim' THEN 1
19	ELSE 2
20	END AS node_caps,
21	deg_malig,
22	CASE
23	WHEN seio = 'esquerdo' THEN '1'
24	WHEN seio = 'direito' THEN '2'
25	END AS seio,
26	CASE
27	WHEN quadrante = 'esquerdo_inferior' THEN 1
28	WHEN quadrante = 'direito_superior' THEN 2
29	WHEN quadrante = 'esquerdo_superior' THEN 3
30	WHEN quadrante = 'direito_inferior' THEN 4
31	WHEN quadrante = 'central' THEN 5
32	ELSE 6
33	END AS quadrante,
34	CASE
35	WHEN irradiando = 'não' THEN 0
36	WHEN irradiando = 'sim' THEN 1
37	END AS irradiando,
38	CASE
39	WHEN menopausa = 'acima_de_40' THEN 1
40	ELSE 0
41	END AS acima_de_40,
42	CASE
43	WHEN menopausa = 'pré-menopausa' THEN 1
44	ELSE 0
45	END AS pre_menopausa,
46	CASE
47	WHEN menopausa = 'abaixo_de_40' THEN 1
48	ELSE 0
49	END AS abaixo_de_40
50	FROM pacientes;

Consulta a nova tabela.

Query												
Query History												
1 SELECT * FROM pacientes_resultado;												
Data Output												
Messages												
Notifications												
	classe integer	tamanho_tumor text	node_caps integer	deg_malign integer	seio text	quadrante integer	irradiando integer	acima_de_40 integer	pre_menopausa integer	abaixo_de_40 integer		
1	0	4	0	3	1	1	0	0	1	0		
2	0	3	1	2	2	2	0	0	1	0		
3	1	3	0	2	1	1	0	0	1	0		
4	0	2	1	2	2	3	0	1	0	0		
5	0	1	0	2	2	4	0	0	1	0		
6	0	2	0	2	1	1	1	1	0	0		
7	0	3	0	2	1	1	0	0	1	0		
8	1	3	1	1	1	1	0	1	0	0		
9	0	6	0	2	1	1	1	0	1	0		
10	0	3	0	2	2	3	0	0	1	0		
11	1	1	0	3	1	5	1	0	1	0		
12	0	3	0	2	1	1	0	1	0	0		
13	0	2	0	1	1	2	0	0	0	1		
14	1	3	0	3	1	2	1	1	0	0		
15	1	1	0	3	1	5	0	0	1	0		
16	0	4	1	3	1	3	0	0	1	0		
17	0	4	0	1	1	1	0	0	0	1		
18	0	2	0	2	1	1	1	0	1	0		
19	0	4	0	3	1	1	0	0	1	0		
20	0	4	0	3	1	1	0	1	0	0		

Uma nova tabela, aplicando as transformações em SQL, convertendo todas as variáveis categóricas para a representação numérica. Na variável **menopausa** foram criadas 3 novas colunas dummy.

## 4.13 Salvando os Dados em um Arquivo



Dados salvo em disco em formato csv.



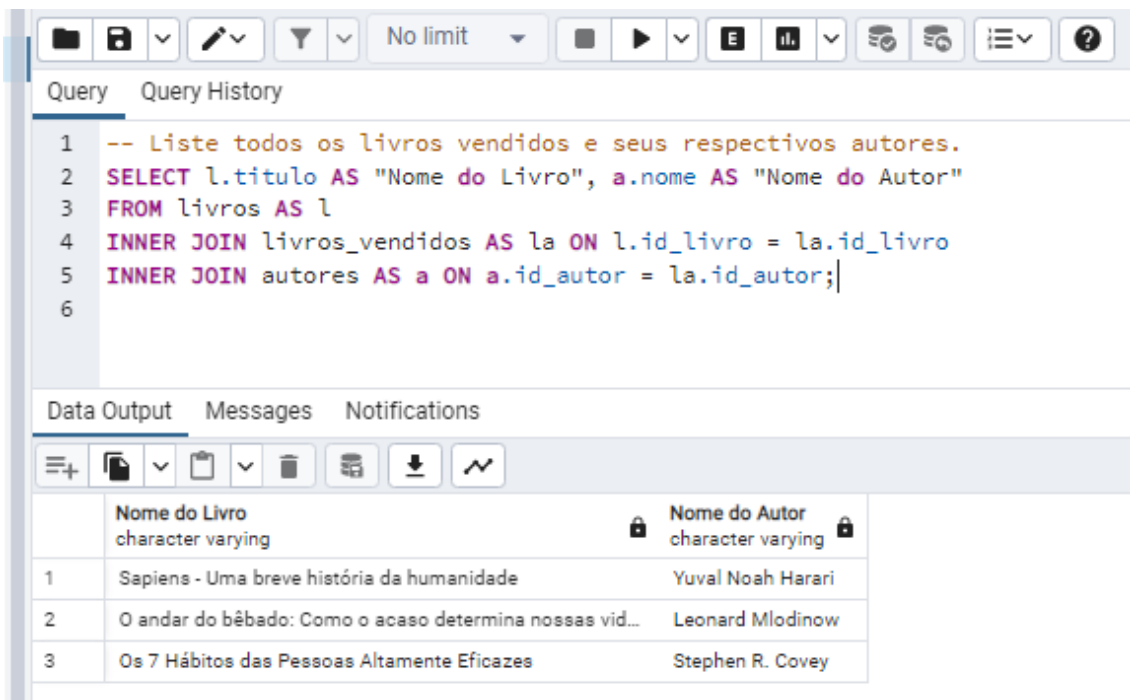
```
1 classe,tamanho_tumor,node_caps,deg_malig,seio,quadrante,irradiando,acima_de_40,pre_menopausa,abaixo_de_40
2 0,4,0,3,1,1,0,0,1,0
3 0,3,1,2,2,2,0,0,1,0
4 1,3,0,2,1,1,0,0,1,0
5 0,2,1,2,2,3,0,1,0,0
6 0,1,0,2,2,4,0,0,1,0
7 0,2,0,2,1,1,1,1,0,0
8 0,3,0,2,1,1,0,0,1,0
9 1,3,1,1,1,1,0,1,0,0
10 0,6,0,2,1,1,1,0,1,0
11 0,3,0,2,2,3,0,0,1,0
12 1,1,0,3,1,5,1,0,1,0
13 0,3,0,2,1,1,0,1,0,0
14 0,2,0,1,1,2,0,0,0,1
15 1,3,0,3,1,2,1,1,0,0
16 1,1,0,3,1,5,0,0,1,0
17 0,4,1,3,1,3,0,0,1,0
18 0,4,0,1,1,1,0,0,0,1
19 0,2,0,2,1,1,1,0,1,0
20 0,4,0,3,1,1,0,0,1,0
21 0,4,0,3,1,1,0,1,0,0
22
```

Esse é o resultado com valores numéricos, são esses valores que a máquina quer receber. Para treinar modelos de Machine Learning é isso que a máquina espera receber como entrada.



## 5 Junção de Tabelas

Liste todos os livros vendidos e seus respectivos autores



The screenshot shows a SQL query editor with a toolbar at the top. The query is as follows:

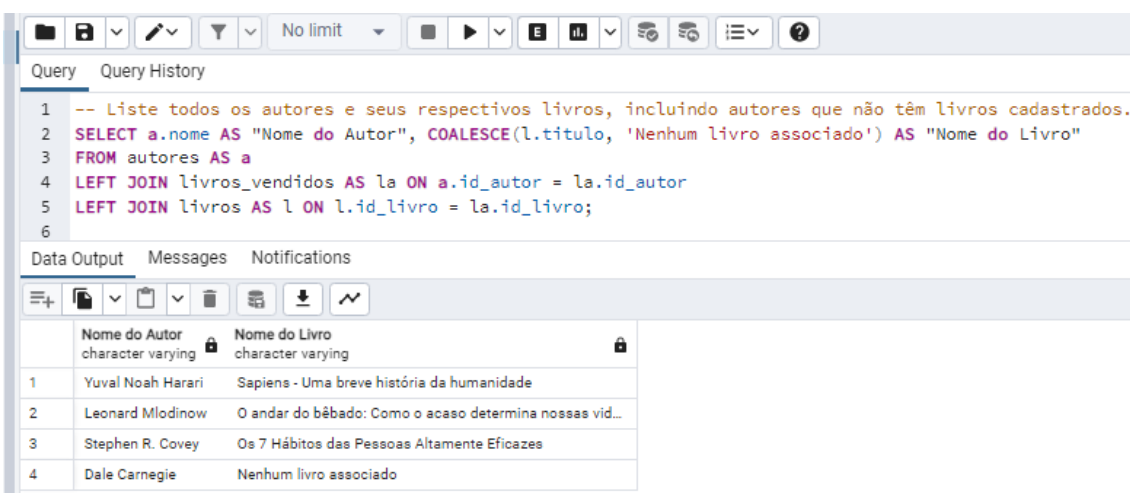
```
1 -- Liste todos os livros vendidos e seus respectivos autores.
2 SELECT l.titulo AS "Nome do Livro", a.nome AS "Nome do Autor"
3 FROM livros AS l
4 INNER JOIN livros_vendidos AS la ON l.id_livro = la.id_livro
5 INNER JOIN autores AS a ON a.id_autor = la.id_autor;
6
```

Below the query editor, the 'Data Output' tab is active, displaying the results of the query in a table:

	Nome do Livro character varying	Nome do Autor character varying
1	Sapiens - Uma breve história da humanidade	Yuval Noah Harari
2	O andar do bêbado: Como o acaso determina nossas vid...	Leonard Mlodinow
3	Os 7 Hábitos das Pessoas Altamente Eficazes	Stephen R. Covey

### INNER JOIN

Liste todos os autores e seus respectivos livros, incluindo autores que não têm livros cadastrados.



The screenshot shows a SQL query editor with a toolbar at the top. The query is as follows:

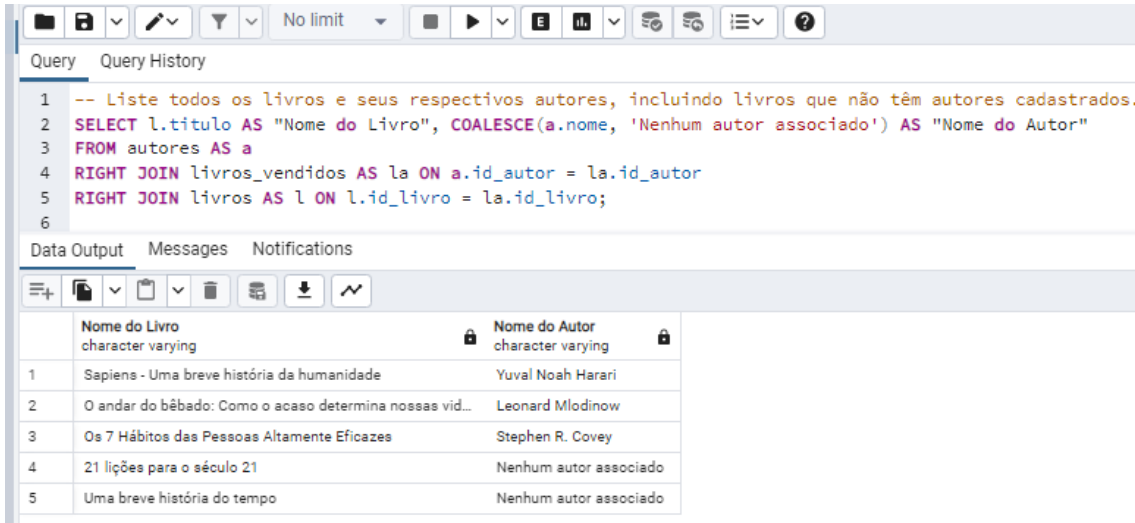
```
1 -- Liste todos os autores e seus respectivos livros, incluindo autores que não têm livros cadastrados.
2 SELECT a.nome AS "Nome do Autor", COALESCE(l.titulo, 'Nenhum livro associado') AS "Nome do Livro"
3 FROM autores AS a
4 LEFT JOIN livros_vendidos AS la ON a.id_autor = la.id_autor
5 LEFT JOIN livros AS l ON l.id_livro = la.id_livro;
6
```

Below the query editor, the 'Data Output' tab is active, displaying the results of the query in a table:

	Nome do Autor character varying	Nome do Livro character varying
1	Yuval Noah Harari	Sapiens - Uma breve história da humanidade
2	Leonard Mlodinow	O andar do bêbado: Como o acaso determina nossas vid...
3	Stephen R. Covey	Os 7 Hábitos das Pessoas Altamente Eficazes
4	Dale Carnegie	Nenhum livro associado

### LEFT JOIN

Liste todos os livros e seus respectivos autores, incluindo livros que não têm autores cadastrados.



The screenshot shows a SQL query editor with the following query:

```

1 -- Liste todos os livros e seus respectivos autores, incluindo livros que não têm autores cadastrados.
2 SELECT l.titulo AS "Nome do Livro", COALESCE(a.nome, 'Nenhum autor associado') AS "Nome do Autor"
3 FROM autores AS a
4 RIGHT JOIN livros_vendidos AS la ON a.id_autor = la.id_autor
5 RIGHT JOIN livros AS l ON l.id_livro = la.id_livro;
6

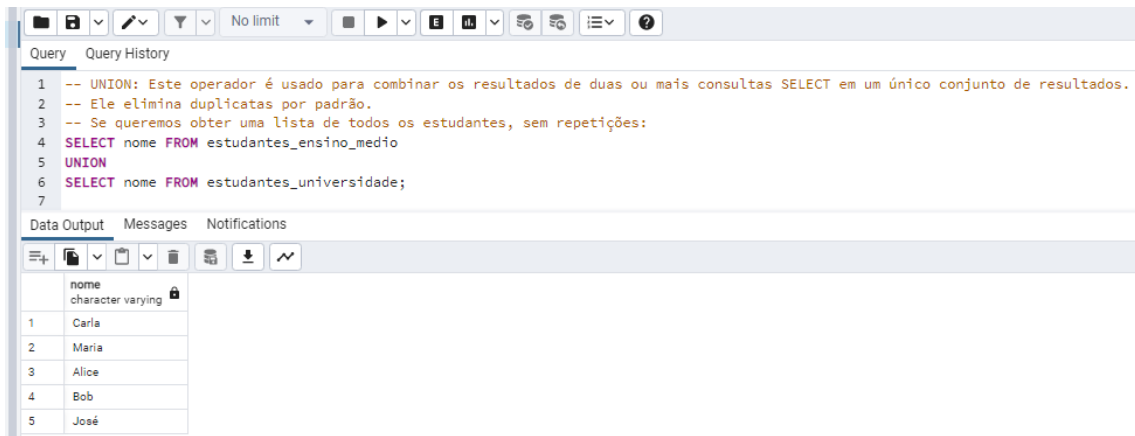
```

The results are displayed in a table with two columns: "Nome do Livro" and "Nome do Autor".

	Nome do Livro	Nome do Autor
1	Sapiens - Uma breve história da humanidade	Yuval Noah Harari
2	O andar do bêbado: Como o acaso determina nossas vid...	Leonard Mlodinow
3	Os 7 Hábitos das Pessoas Altamente Eficazes	Stephen R. Covey
4	21 lições para o século 21	Nenhum autor associado
5	Uma breve história do tempo	Nenhum autor associado

## RIGHT JOIN

UNION: Este operador é usado para combinar os resultados de duas ou mais consultas SELECT em um único conjunto de resultados. Ele elimina duplicatas por padrão. Selecione todos os estudantes, sem repetições:



The screenshot shows a SQL query editor with the following query:

```

1 -- UNION: Este operador é usado para combinar os resultados de duas ou mais consultas SELECT em um único conjunto de resultados.
2 -- Ele elimina duplicatas por padrão.
3 -- Se queremos obter uma lista de todos os estudantes, sem repetições:
4 SELECT nome FROM estudantes_ensino_medio
5 UNION
6 SELECT nome FROM estudantes_universidade;
7

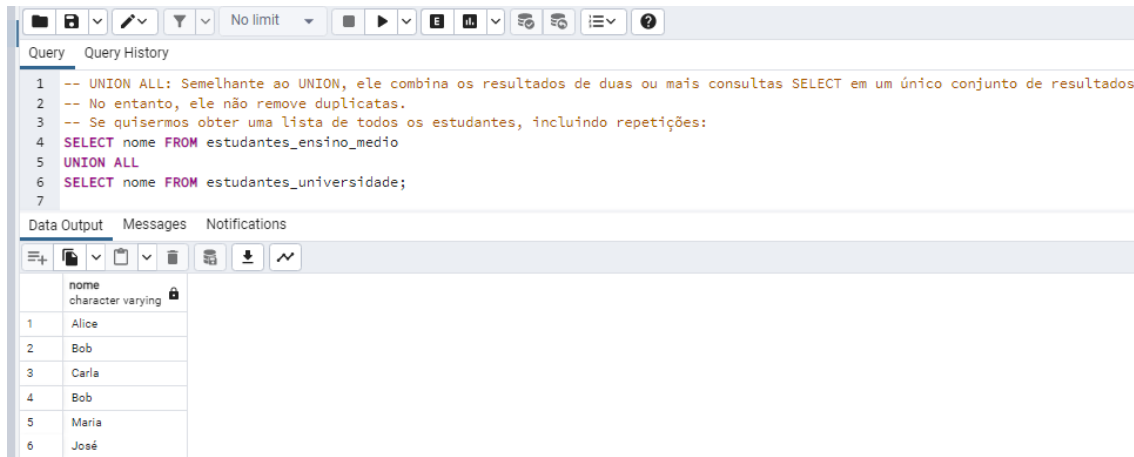
```

The results are displayed in a table with one column: "nome".

	nome
1	Carla
2	Maria
3	Alice
4	Bob
5	José

## UNION

UNION ALL: Semelhante ao UNION, ele combina os resultados de duas ou mais consultas SELECT em um único conjunto de resultados. No entanto, ele não remove duplicatas. Selecione todos os estudantes, incluindo repetições:



```

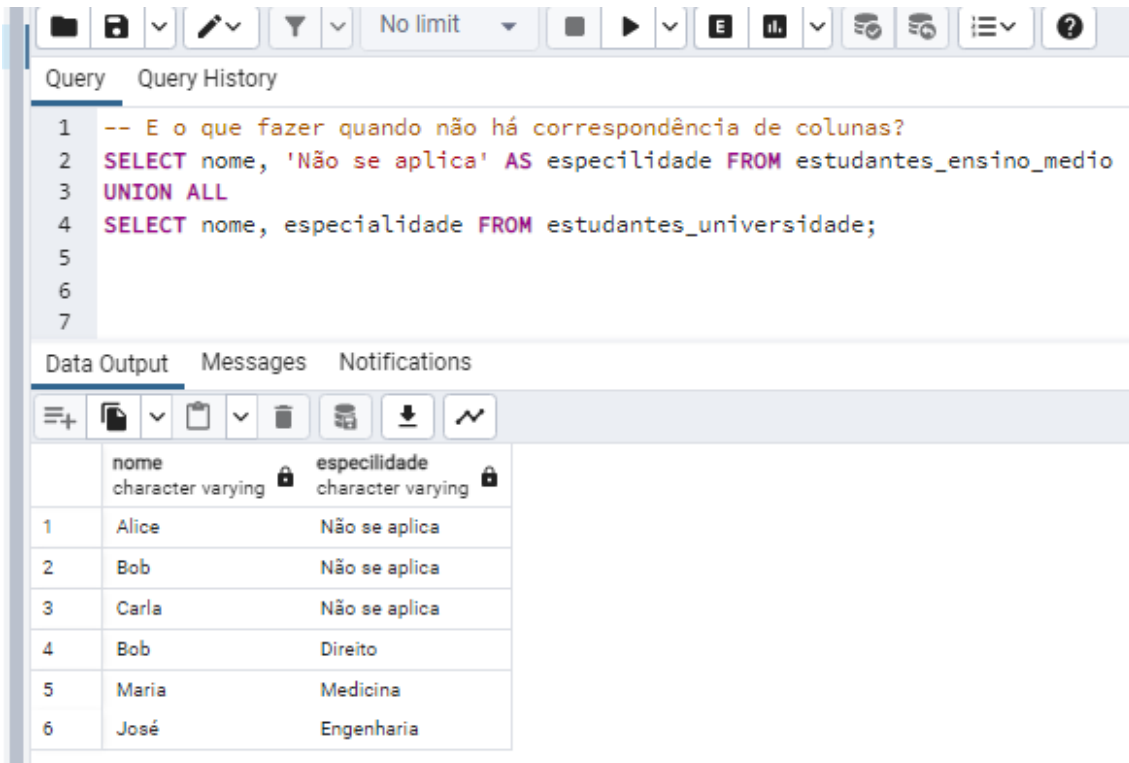
1  -- UNION ALL: Semelhante ao UNION, ele combina os resultados de duas ou mais consultas SELECT em um único conjunto de resultados.
2  -- No entanto, ele não remove duplicatas.
3  -- Se quisermos obter uma lista de todos os estudantes, incluindo repetições:
4  SELECT nome FROM estudantes_ensino_medio
5  UNION ALL
6  SELECT nome FROM estudantes_universidade;
7

```

	nome
1	Alice
2	Bob
3	Carla
4	Bob
5	Maria
6	José

## UNION ALL

E o que fazer quando não há correspondência de colunas?



```

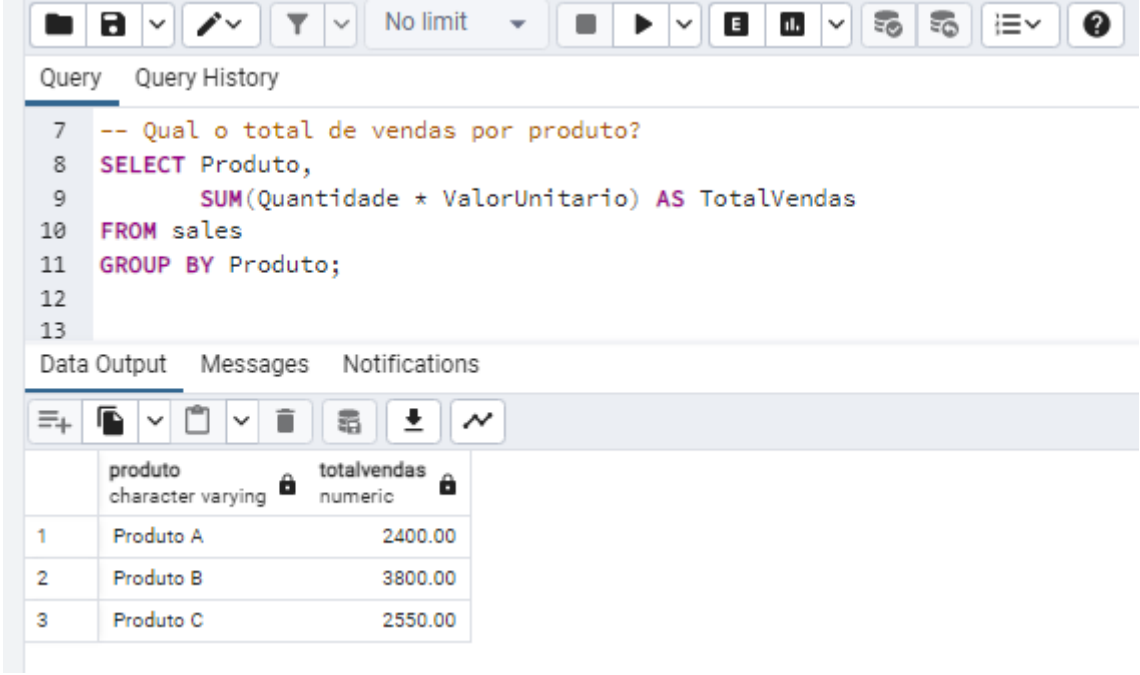
1  -- E o que fazer quando não há correspondência de colunas?
2  SELECT nome, 'Não se aplica' AS especialidade FROM estudantes_ensino_medio
3  UNION ALL
4  SELECT nome, especialidade FROM estudantes_universidade;
5
6
7

```

	nome	especialidade
1	Alice	Não se aplica
2	Bob	Não se aplica
3	Carla	Não se aplica
4	Bob	Direito
5	Maria	Medicina
6	José	Engenharia

## 6 Agregação para Análise de Dados

Qual o total de vendas por produto?



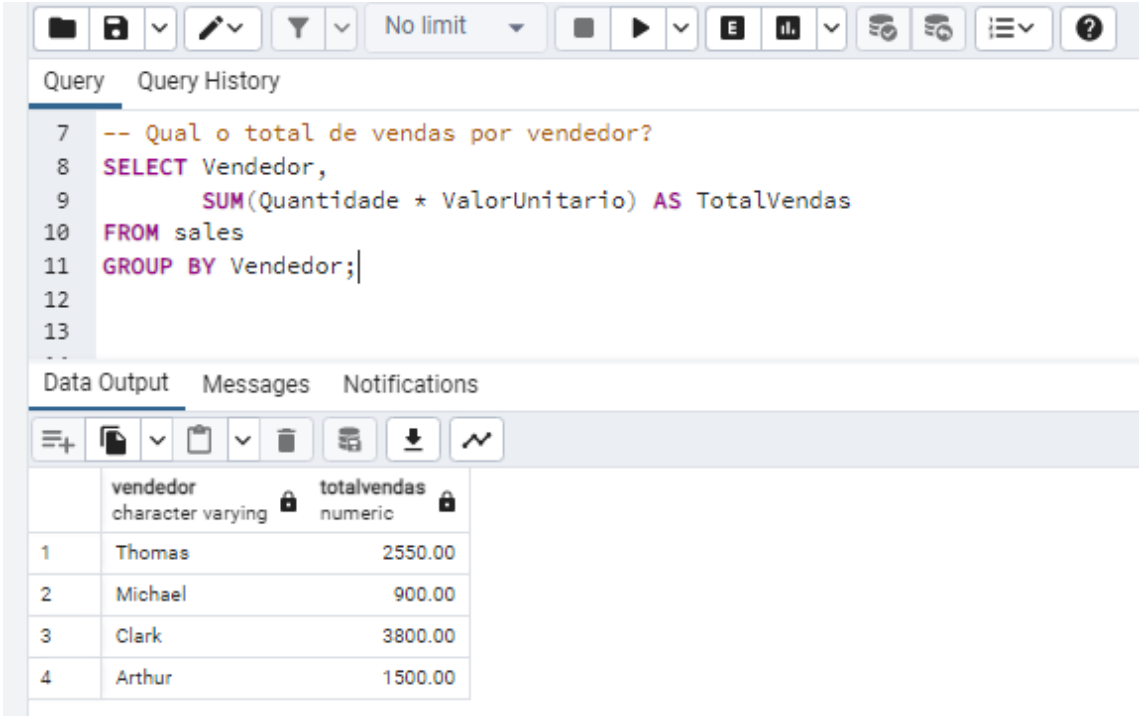
The screenshot shows a SQL IDE interface. The top toolbar includes icons for file operations, filters, and execution. Below the toolbar, there are tabs for 'Query' and 'Query History'. The 'Query' tab is active, displaying the following SQL code:

```
7 -- Qual o total de vendas por produto?
8 SELECT Produto,
9        SUM(Quantidade * ValorUnitario) AS TotalVendas
10 FROM sales
11 GROUP BY Produto;
```

Below the query editor, there are tabs for 'Data Output', 'Messages', and 'Notifications'. The 'Data Output' tab is active, showing a table with the results of the query:

	produto character varying	totalvendas numeric
1	Produto A	2400.00
2	Produto B	3800.00
3	Produto C	2550.00

Qual o total de vendas por vendedor?



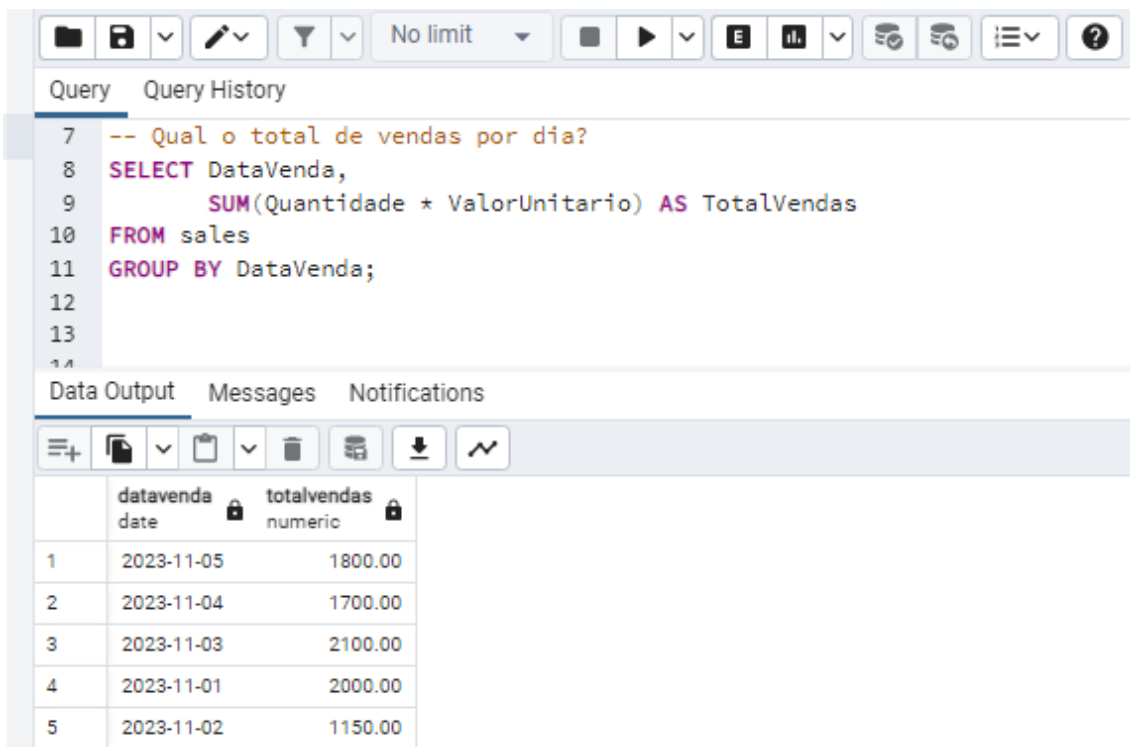
The screenshot shows a SQL IDE interface. The top toolbar includes icons for file operations, filters, and execution. Below the toolbar, there are tabs for 'Query' and 'Query History'. The 'Query' tab is active, displaying the following SQL code:

```
7 -- Qual o total de vendas por vendedor?
8 SELECT Vendedor,
9        SUM(Quantidade * ValorUnitario) AS TotalVendas
10 FROM sales
11 GROUP BY Vendedor;
```

Below the query editor, there are tabs for 'Data Output', 'Messages', and 'Notifications'. The 'Data Output' tab is active, showing a table with the results of the query:

	vendedor character varying	totalvendas numeric
1	Thomas	2550.00
2	Michael	900.00
3	Clark	3800.00
4	Arthur	1500.00

Qual o total de vendas por dia?



The screenshot shows a SQL query editor interface. The query is as follows:

```

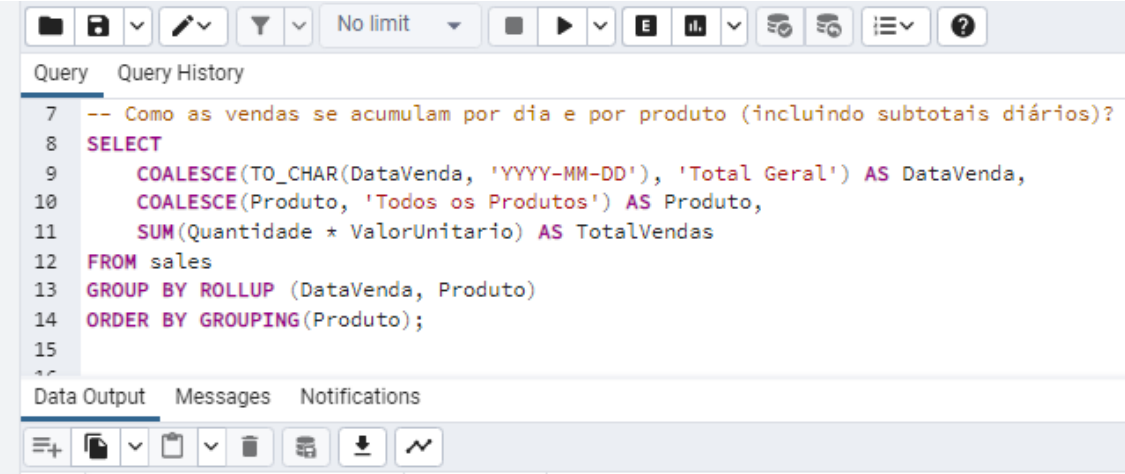
7  -- Qual o total de vendas por dia?
8  SELECT DataVenda,
9         SUM(Quantidade * ValorUnitario) AS TotalVendas
10 FROM sales
11 GROUP BY DataVenda;
12
13
14

```

Below the query editor, the 'Data Output' tab is active, displaying the results of the query in a table format:

	datavenda date	totalvendas numeric
1	2023-11-05	1800.00
2	2023-11-04	1700.00
3	2023-11-03	2100.00
4	2023-11-01	2000.00
5	2023-11-02	1150.00

Como as vendas se acumulam por dia e por produto (incluindo subtotais diários)?



Query

```

7  -- Como as vendas se acumulam por dia e por produto (incluindo subtotais diários)?
8  SELECT
9      COALESCE(TO_CHAR(DataVenda, 'YYYY-MM-DD'), 'Total Geral') AS DataVenda,
10     COALESCE(Produto, 'Todos os Produtos') AS Produto,
11     SUM(Quantidade * ValorUnitario) AS TotalVendas
12 FROM sales
13 GROUP BY ROLLUP (DataVenda, Produto)
14 ORDER BY GROUPING(Produto);
15

```

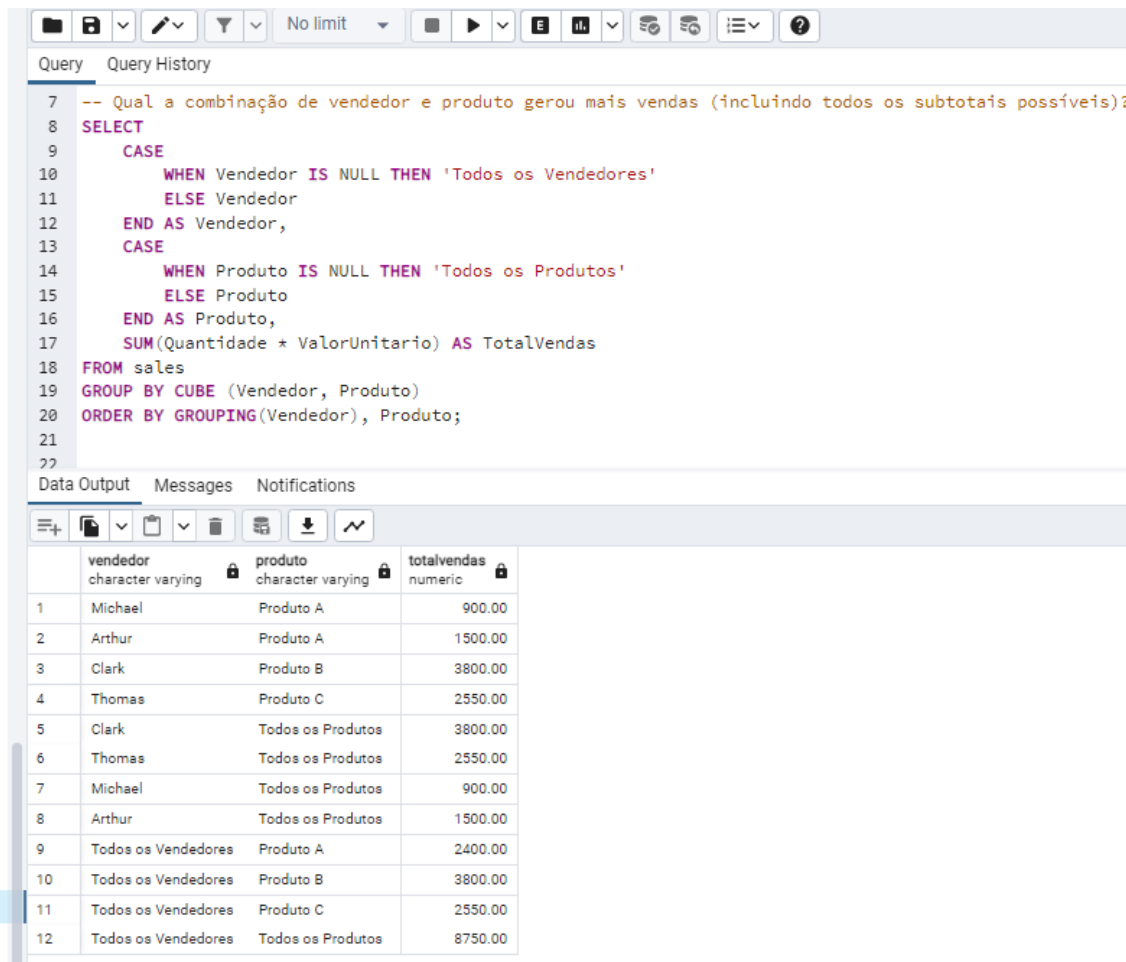
Data Output

	datavenda text	produto character varying	totalvendas numeric
1	2023-11-03	Produto B	1600.00
2	2023-11-04	Produto C	1500.00
3	2023-11-04	Produto A	200.00
4	2023-11-05	Produto B	1200.00
5	2023-11-01	Produto A	1000.00
6	2023-11-03	Produto A	500.00
7	2023-11-05	Produto C	600.00
8	2023-11-02	Produto A	700.00
9	2023-11-02	Produto C	450.00
10	2023-11-01	Produto B	1000.00
11	2023-11-02	Todos os Produtos	1150.00
12	2023-11-05	Todos os Produtos	1800.00
13	2023-11-04	Todos os Produtos	1700.00
14	2023-11-03	Todos os Produtos	2100.00
15	2023-11-01	Todos os Produtos	2000.00
16	Total Geral	Todos os Produtos	8750.00

A cláusula 'ROLLUP' é usada para adicionar subtotais e totais gerais ao conjunto de resultados de uma consulta 'GROUP BY'.

A função 'GROUPING' é usada para diferenciar entre linhas de subtotal e linhas de detalhe nos resultados das consultas que utilizam 'ROLLUP' ou 'CUBE'. A função 'GROUPING' retorna 1 para linhas de subtotal e 0 para linhas de detalhe.

Qual a combinação de vendedor e produto gerou mais vendas (incluindo todos os subtotais possíveis)?



```

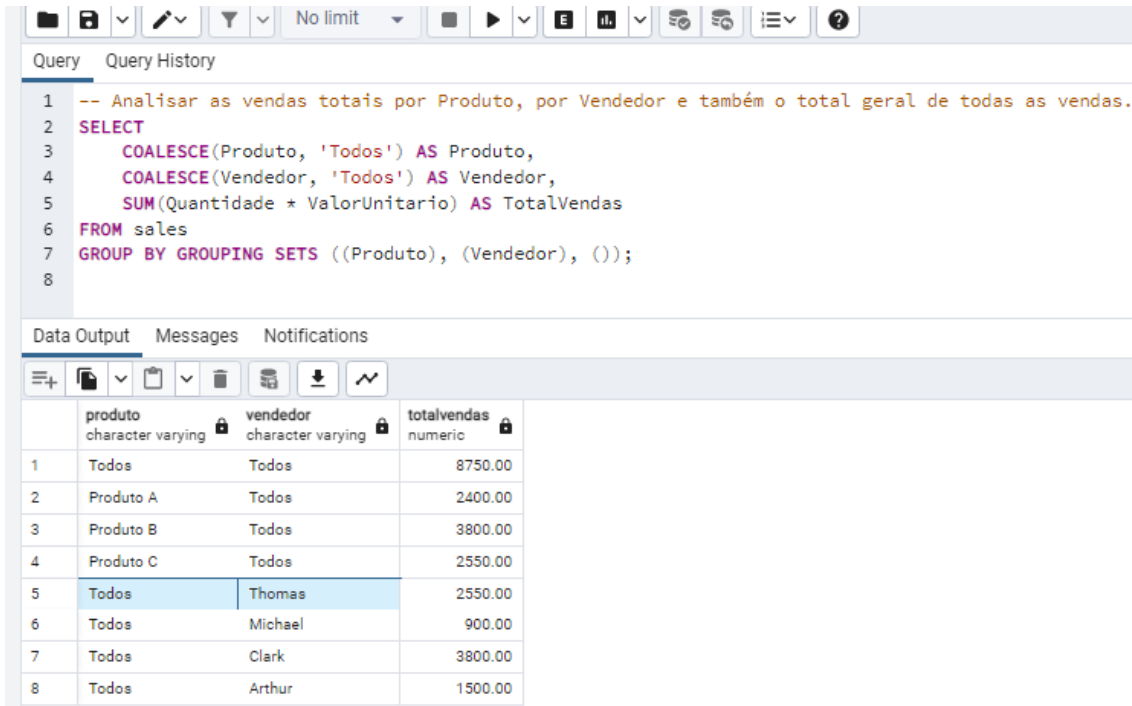
7  -- Qual a combinação de vendedor e produto gerou mais vendas (incluindo todos os subtotais possíveis)?
8  SELECT
9      CASE
10         WHEN Vendedor IS NULL THEN 'Todos os Vendedores'
11         ELSE Vendedor
12     END AS Vendedor,
13     CASE
14         WHEN Produto IS NULL THEN 'Todos os Produtos'
15         ELSE Produto
16     END AS Produto,
17     SUM(Quantidade * ValorUnitario) AS TotalVendas
18 FROM sales
19 GROUP BY CUBE (Vendedor, Produto)
20 ORDER BY GROUPING(Vendedor), Produto;
21
22

```

	vendedor character varying	produto character varying	totalvendas numeric
1	Michael	Produto A	900.00
2	Arthur	Produto A	1500.00
3	Clark	Produto B	3800.00
4	Thomas	Produto C	2550.00
5	Clark	Todos os Produtos	3800.00
6	Thomas	Todos os Produtos	2550.00
7	Michael	Todos os Produtos	900.00
8	Arthur	Todos os Produtos	1500.00
9	Todos os Vendedores	Produto A	2400.00
10	Todos os Vendedores	Produto B	3800.00
11	Todos os Vendedores	Produto C	2550.00
12	Todos os Vendedores	Todos os Produtos	8750.00

A cláusula 'CUBE' é usada para gerar subtotais para todas as combinações possíveis de um conjunto de colunas.

Analisar as vendas totais por Produto, por Vendedor e também o total geral de todas as vendas.



The screenshot shows a SQL query editor with a toolbar at the top. The query is as follows:

```
1 -- Analisar as vendas totais por Produto, por Vendedor e também o total geral de todas as vendas.
2 SELECT
3     COALESCE(Produto, 'Todos') AS Produto,
4     COALESCE(Vendedor, 'Todos') AS Vendedor,
5     SUM(Quantidade * ValorUnitario) AS TotalVendas
6 FROM sales
7 GROUP BY GROUPING SETS ((Produto), (Vendedor), ());
8
```

Below the query editor, the 'Data Output' tab is active, displaying the results in a table:

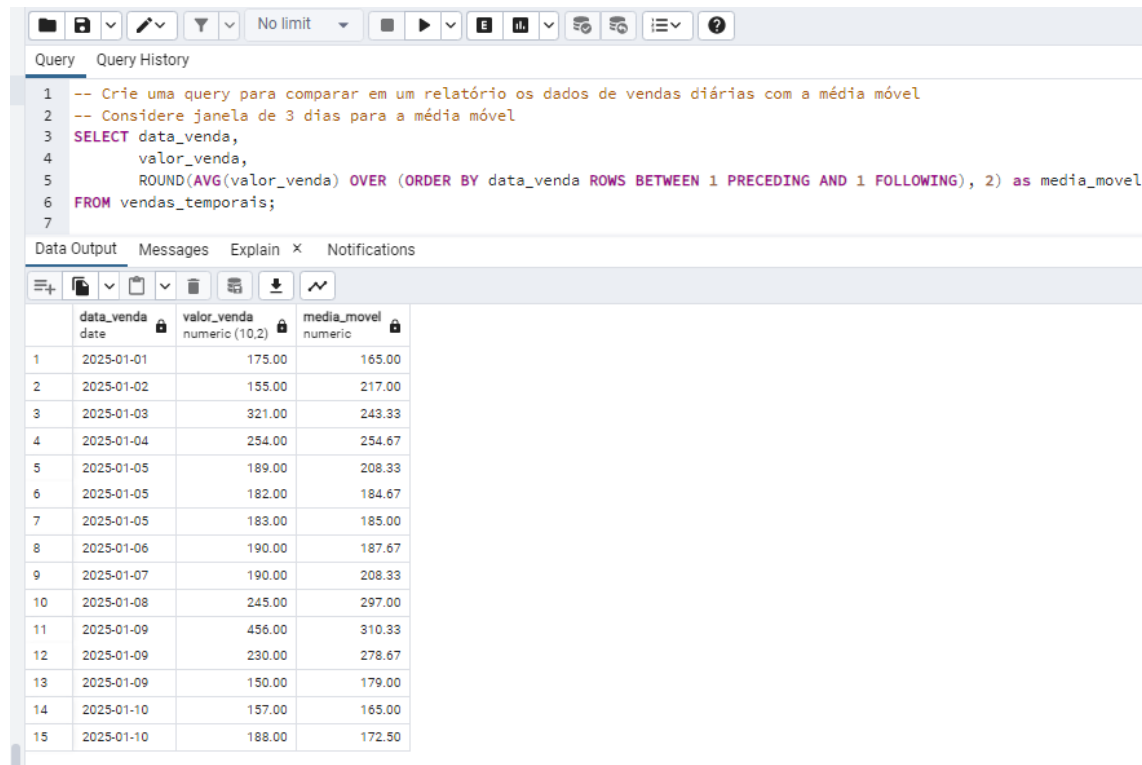
	produto character varying	vendedor character varying	totalvendas numeric
1	Todos	Todos	8750.00
2	Produto A	Todos	2400.00
3	Produto B	Todos	3800.00
4	Produto C	Todos	2550.00
5	Todos	Thomas	2550.00
6	Todos	Michael	900.00
7	Todos	Clark	3800.00
8	Todos	Arthur	1500.00

A cláusula 'GROUPING SETS' permite especificar várias combinações de colunas de agrupamento explicitamente.



## 7 Window Functions e Subqueries

Query para comparar em um relatório os dados de vendas diárias com a média móvel. Considere janela de 3 dias para a média móvel



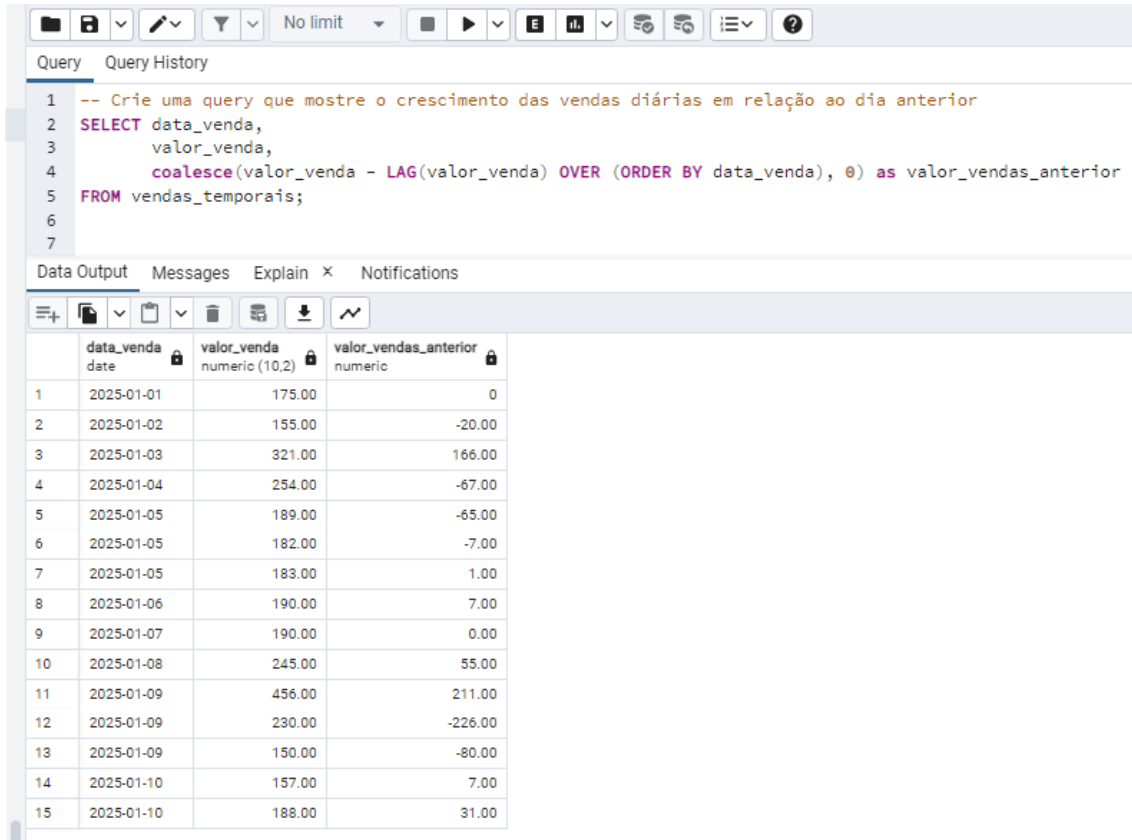
The screenshot shows a SQL IDE interface. The top toolbar includes icons for file operations, query execution, and settings. Below the toolbar, the 'Query' tab is active, displaying a SQL query. The 'Data Output' tab is also visible, showing the results of the query in a table format. The query calculates a moving average for sales data over a 3-day window.

```
1 -- Crie uma query para comparar em um relatório os dados de vendas diárias com a média móvel
2 -- Considere janela de 3 dias para a média móvel
3 SELECT data_venda,
4        valor_venda,
5        ROUND(AVG(valor_venda) OVER (ORDER BY data_venda ROWS BETWEEN 1 PRECEDING AND 1 FOLLOWING), 2) as media_movel
6 FROM vendas_temporais;
7
```

	data_venda date	valor_venda numeric (10,2)	media_movel numeric
1	2025-01-01	175.00	165.00
2	2025-01-02	155.00	217.00
3	2025-01-03	321.00	243.33
4	2025-01-04	254.00	254.67
5	2025-01-05	189.00	208.33
6	2025-01-05	182.00	184.67
7	2025-01-05	183.00	185.00
8	2025-01-06	190.00	187.67
9	2025-01-07	190.00	208.33
10	2025-01-08	245.00	297.00
11	2025-01-09	456.00	310.33
12	2025-01-09	230.00	278.67
13	2025-01-09	150.00	179.00
14	2025-01-10	157.00	165.00
15	2025-01-10	188.00	172.50



Crie uma query que mostre o crescimento das vendas diárias em relação ao dia anterior



The screenshot shows a SQL IDE interface. The top toolbar includes icons for file operations, query execution, and settings. Below the toolbar, the 'Query' tab is active, displaying the following SQL query:

```

1  -- Crie uma query que mostre o crescimento das vendas diárias em relação ao dia anterior
2  SELECT data_venda,
3         valor_venda,
4         coalesce(valor_venda - LAG(valor_venda) OVER (ORDER BY data_venda), 0) as valor_vendas_anterior
5  FROM vendas_temporais;
6
7

```

Below the query editor, the 'Data Output' tab is active, showing the results of the query in a table format. The table has three columns: 'data\_venda' (date), 'valor\_venda' (numeric), and 'valor\_vendas\_anterior' (numeric). The results are as follows:

	data_venda date	valor_venda numeric (10,2)	valor_vendas_anterior numeric
1	2025-01-01	175.00	0
2	2025-01-02	155.00	-20.00
3	2025-01-03	321.00	166.00
4	2025-01-04	254.00	-67.00
5	2025-01-05	189.00	-65.00
6	2025-01-05	182.00	-7.00
7	2025-01-05	183.00	1.00
8	2025-01-06	190.00	7.00
9	2025-01-07	190.00	0.00
10	2025-01-08	245.00	55.00
11	2025-01-09	456.00	211.00
12	2025-01-09	230.00	-226.00
13	2025-01-09	150.00	-80.00
14	2025-01-10	157.00	7.00
15	2025-01-10	188.00	31.00

## Query que mostre a soma acumulada de vendas dia a dia

Query			
<pre> 1  -- Crie uma query que mostre a soma acumulada de vendas dia a dia 2  SELECT data_venda, 3         valor_venda, 4         SUM(valor_venda) OVER (PARTITION BY EXTRACT(MONTH FROM data_venda) ORDER BY data_venda) as soma_acumulada 5  FROM vendas_temporais; 6 7 </pre>			
Data Output			
	data_venda date	valor_venda numeric (10,2)	soma_acumulada numeric
1	2025-01-01	175.00	175.00
2	2025-01-02	155.00	330.00
3	2025-01-03	321.00	651.00
4	2025-01-04	254.00	905.00
5	2025-01-05	189.00	1459.00
6	2025-01-05	182.00	1459.00
7	2025-01-05	183.00	1459.00
8	2025-01-06	190.00	1649.00
9	2025-01-07	190.00	1839.00
10	2025-01-08	245.00	2084.00
11	2025-01-09	456.00	2920.00
12	2025-01-09	230.00	2920.00
13	2025-01-09	150.00	2920.00
14	2025-01-10	157.00	3265.00
15	2025-01-10	188.00	3265.00

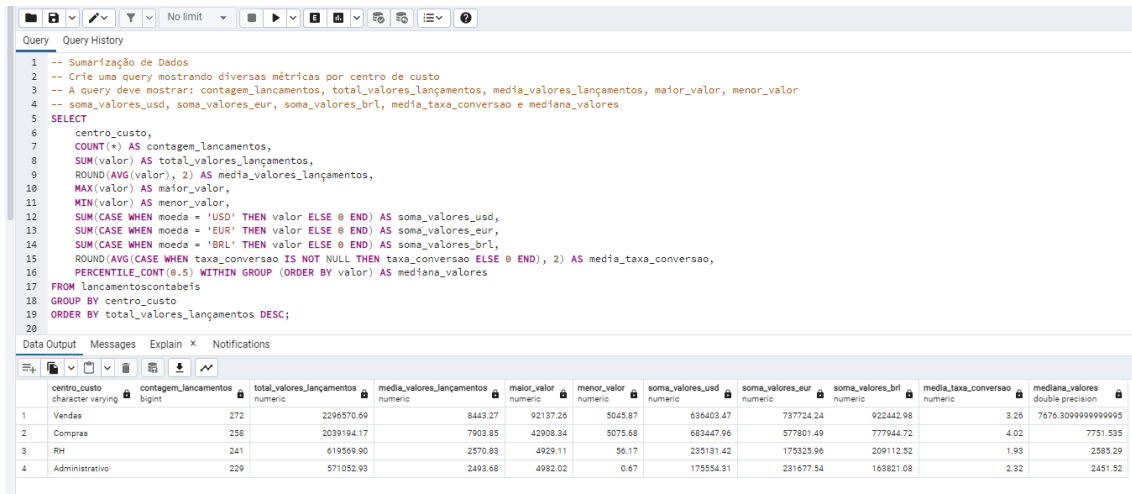
## Ranking de vendas por funcionário considerando o valor total de vendas por dia e de cada funcionário

Query Query History				
<pre> 1  -- Crie um ranking de vendas por funcionário considerando o valor total de vendas por dia e de cada funcionário 2  SELECT 3      *, 4      DENSE_RANK() OVER (PARTITION BY funcionario_id ORDER BY valor_venda DESC) as rank_vendas 5  FROM 6      (SELECT data_venda, funcionario_id, SUM(valor_venda) as valor_venda 7       FROM vendas_temporais 8       GROUP BY data_venda, funcionario_id) as subquery 9  ORDER BY 10     funcionario_id, rank_vendas; 11 </pre>				
Data Output Messages Explain × Notifications				
	data_venda	funcionario_id	valor_venda	rank_vendas
	date	integer	numeric	bigint
1	2025-01-04	1001	254.00	1
2	2025-01-10	1001	188.00	2
3	2025-01-01	1001	175.00	3
4	2025-01-02	1001	155.00	4
5	2025-01-05	1002	554.00	1
6	2025-01-03	1002	321.00	2
7	2025-01-10	1002	157.00	3
8	2025-01-06	1003	190.00	1
9	2025-01-07	1003	190.00	1
10	2025-01-09	1003	150.00	2
11	2025-01-08	1004	245.00	1
12	2025-01-09	1005	686.00	1

## 8 Análise Exploratória de Dados com SQL

### 8.1 Sumarização de Dados

Crie uma query mostrando diversas métricas por centro de custo. A query deve mostrar: `contagem_lancamentos`, `total_valores_lancamentos`, `media_valores_lancamentos`, `maior_valor`, `menor_valor`, `soma_valores_usd`, `soma_valores_eur`, `soma_valores_brl`, `media_taxa_conversao` e `mediana_valores`



The screenshot shows a SQL query editor with a query history pane and a data output pane. The query is as follows:

```

1 -- Sumarização de Dados
2 -- Crie uma query mostrando diversas métricas por centro de custo
3 -- A query deve mostrar: contagem_lancamentos, total_valores_lancamentos, media_valores_lancamentos, maior_valor, menor_valor
4 -- soma_valores_usd, soma_valores_eur, soma_valores_brl, media_taxa_conversao e mediana_valores
5 SELECT
6     centro_custo,
7     COUNT(*) AS contagem_lancamentos,
8     SUM(valor) AS total_valores_lancamentos,
9     ROUND(AVG(valor), 2) AS media_valores_lancamentos,
10    MAX(valor) AS maior_valor,
11    MIN(valor) AS menor_valor,
12    SUM(CASE WHEN moeda = 'USD' THEN valor ELSE 0 END) AS soma_valores_usd,
13    SUM(CASE WHEN moeda = 'EUR' THEN valor ELSE 0 END) AS soma_valores_eur,
14    SUM(CASE WHEN moeda = 'BRL' THEN valor ELSE 0 END) AS soma_valores_brl,
15    ROUND(AVG(CASE WHEN taxa_conversao IS NOT NULL THEN taxa_conversao ELSE 0 END), 2) AS media_taxa_conversao,
16    PERCENTILE_CONT(0.5) WITHIN GROUP (ORDER BY valor) AS mediana_valores
17 FROM lancamentoscontabelfs
18 GROUP BY centro_custo
19 ORDER BY total_valores_lancamentos DESC;
20

```

The data output pane shows the following results:

	centro_custo character varying	contagem_lancamentos bigint	total_valores_lancamentos numeric	media_valores_lancamentos numeric	maior_valor numeric	menor_valor numeric	soma_valores_usd numeric	soma_valores_eur numeric	soma_valores_brl numeric	media_taxa_conversao numeric	mediana_valores double precision
1	Vendas	272	2295570.69	8443.27	92137.26	5045.87	636403.47	737724.24	922442.98	3.26	7676.3099999999995
2	Compras	258	2039194.17	7903.85	42908.34	5075.68	688447.96	577801.49	777944.72	4.02	7751.535
3	RH	241	619569.90	2570.83	4929.11	56.17	235131.42	175325.96	209112.52	1.93	2385.29
4	Administrativo	229	571052.93	2493.68	4982.02	0.67	175554.31	231677.54	163821.08	2.32	2451.52

## 8.2 Distribuição de Dados

Crie uma query para mostrar a distribuição de dados na tabela. O relatório deve mostrar: quantidade\_lancamentos, media\_valor, desvio\_padrao\_valor, menor\_valor, maior\_valor e primeiro, segundo e terceiro quartil. Tudo isso por centro de custo e por moeda.

Query Query History

```

1 -- Distribuição de Dados
2 -- Crie uma query para mostrar a distribuição de dados na tabela.
3 -- o relatório deve mostrar: quantidade_lancamentos, media_valor, desvio_padrao_valor, menor_valor,
4 -- maior_valor e primeiro, segundo e terceiro quartil.
5 -- Tudo isso por centro de custo e por moeda.
6 SELECT
7     centro_custo,
8     moeda,
9     COUNT(*) AS quantidade_lancamentos,
10    ROUND(AVG(valor)::NUMERIC, 2) AS media_valor,
11    ROUND(STDDEV(valor)::NUMERIC, 2) AS desvio_padrao_valor,
12    MIN(valor) AS menor_valor,
13    MAX(valor) AS maior_valor,
14    ROUND((PERCENTILE_CONT(0.25) WITHIN GROUP (ORDER BY valor))::NUMERIC, 2) AS primeiro_quartil,
15    ROUND((PERCENTILE_CONT(0.5) WITHIN GROUP (ORDER BY valor))::NUMERIC, 2) AS mediana,
16    ROUND((PERCENTILE_CONT(0.75) WITHIN GROUP (ORDER BY valor))::NUMERIC, 2) AS terceiro_quartil
17 FROM lancamentoscontabeis
18 GROUP BY centro_custo, moeda
19 ORDER BY centro_custo, moeda;

```

Data Output Messages Explain X Notifications

	centro_custo	moeda	quantidade_lancamentos	media_valor	desvio_padrao_valor	menor_valor	maior_valor	primeiro_quartil	mediana	terceiro_quartil
	character varying	character varying	bigint	numeric	numeric	numeric	numeric	numeric	numeric	numeric
1	Administrativo	BRL	73	2244.12	1407.65	45.22	4967.86	1022.54	2093.50	3315.23
2	Administrativo	EUR	85	2725.62	1428.54	10.45	4982.02	1555.36	2898.47	3817.48
3	Administrativo	USD	71	2472.60	1440.58	0.67	4931.50	1388.79	2265.37	3571.28
4	Compras	BRL	97	8020.05	3846.76	5104.85	42908.34	6556.35	7749.94	9005.88
5	Compras	EUR	76	7602.65	1519.78	5075.68	9979.08	6437.52	7447.32	8936.76
6	Compras	USD	85	8040.56	3212.73	5169.48	28367.48	6567.41	7782.67	8757.58
7	RH	BRL	80	2613.91	1313.24	192.52	4781.33	1687.94	2497.03	3506.89
8	RH	EUR	78	2247.77	1454.77	56.17	4929.11	919.42	2066.81	3457.26
9	RH	USD	83	2832.91	1321.90	82.47	4897.08	1988.68	3014.76	3754.26
10	Vendas	BRL	100	9224.43	11000.37	5045.87	92137.26	6194.14	7613.55	9272.23
11	Vendas	EUR	94	7848.13	2962.11	5067.84	32821.78	6353.86	7692.31	8804.89
12	Vendas	USD	78	8159.02	5153.78	5074.35	51416.76	6540.46	7668.29	8753.43

## 8.3 Análise Multivariada

Análise Multivariada (mais de uma coluna no mesmo tempo)

Os requisitos do relatório:

- Calcule valor total dos lançamentos
- Calcule a média dos lançamentos
- Calcule a contagem dos lançamentos
- Calcule a média do valor de taxa de conversão somente se a moeda for diferente de BRL
- Crie ranking por valor total dos lançamentos, por média do valor dos lançamentos e por média da taxa de conversão
- Apenas quando o centro de custo for 'Compras' ou 'RH'

Query Query History

```

1 -- Análise Multivariada (mais de uma coluna no mesmo tempo)
2 -- Aqui estão os requisitos do relatório:
3 -- Calcule valor total dos lançamentos
4 -- Calcule a média dos lançamentos
5 -- Calcule a contagem dos lançamentos
6 -- Calcule a média do valor de taxa de conversão somente se a moeda for diferente de BRL
7 -- Crie ranking por valor total dos lançamentos, por média do valor dos lançamentos e por média da taxa de conversão
8 -- Apenas quando o centro de custo for 'Compras' ou 'RH'
9 SELECT
10     A.centro_custo,
11     A.moeda,
12     SUM(A.valor) AS total_valor_lancamento,
13     DENSE_RANK() OVER (ORDER BY SUM(A.valor) DESC) AS rank_total_valor,
14     ROUND(AVG(A.valor), 2) AS media_valor_lancamento,
15     DENSE_RANK() OVER (ORDER BY AVG(A.valor) DESC) AS rank_media_valor,
16     COUNT(A.*) AS numero_de_lancamentos,
17     COALESCE(ROUND(AVG(A.taxa_conversao) FILTER (WHERE A.moeda != 'BRL'), 2), 0) AS media_taxa_conversao,
18     DENSE_RANK() OVER (ORDER BY COALESCE(ROUND(AVG(A.taxa_conversao) FILTER (WHERE A.moeda != 'BRL'), 2), 0) DESC) AS rank_media_taxa
19 FROM lancamentoscontabeis A
20 WHERE A.centro_custo IN ('Compras', 'RH')
21 GROUP BY A.centro_custo, A.moeda
22 ORDER BY rank_total_valor, rank_media_valor, rank_media_taxa;
23
24

```

Data Output Messages Explain x Notifications

	centro_custo character varying	moeda character varying	total_valor_lancamento numeric	rank_total_valor bigint	media_valor_lancamento numeric	rank_media_valor bigint	numero_de_lancamentos bigint	media_taxa_conversao numeric	rank_media_taxa bigint
1	Compras	BRL	777944.72	1	8020.05	2	97	0	5
2	Compras	USD	683447.96	2	8040.56	1	85	7.90	1
3	Compras	EUR	577801.49	3	7602.65	3	76	5.97	2
4	RH	USD	235131.42	4	2832.91	4	83	4.37	4
5	RH	BRL	209112.52	5	2613.91	5	80	0	5
6	RH	EUR	175325.96	6	2247.77	6	78	4.61	3



## 8.4 Identificação de Outliers

Vamos analisar a coluna valor:

- Para melhorar a análise, vamos observar os outliers por centro de custo e moeda
- Para identificar outliers na tabela podemos usar uma abordagem baseada em Estatística,
- como calcular o intervalo interquartil (IQR) e identificar valores que estão significativamente acima ou abaixo desse intervalo.
- O IQR é a diferença entre o primeiro quartil (Q1, o 25º percentil) e o terceiro quartil (Q3, o 75º percentil).
- Os valores abaixo de  $Q1 - 0.5 * IQR$  ou acima de  $Q3 + 0.5 * IQR$  serão considerados outliers.
- Crie a query que identifique os outliers (se existirem), por centro de custo e moeda.

Query Query History

```

1 -- Identificação de Outliers
2 -- Vamos analisar a coluna valor
3 -- Para melhorar a análise, vamos observar os outliers por centro de custo e moeda
4 -- Para identificar outliers na tabela podemos usar uma abordagem baseada em Estatística,
5 -- como calcular o intervalo interquartil (IQR) e identificar valores que estão significativamente
6 -- acima ou abaixo desse intervalo.
7 -- O IQR é a diferença entre o primeiro quartil (Q1, o 25º percentil) e o terceiro quartil (Q3, o 75º percentil).
8 -- Os valores abaixo de  $Q1 - 0.5 * IQR$  ou acima de  $Q3 + 0.5 * IQR$  serão considerados outliers.
9 -- Crie a query que identifique os outliers (se existirem), por centro de custo e moeda.
10 SELECT
11     centro_custo,
12     moeda,
13     MIN(valor) as minimo_valor,
14     PERCENTILE_CONT(0.25) WITHIN GROUP (ORDER BY valor) - 0.5 * (PERCENTILE_CONT(0.75) WITHIN GROUP (ORDER BY valor) - PERCENTILE_CONT(0.25) WITHIN GROUP (ORDER BY valor)) AS limite_inferior,
15     PERCENTILE_CONT(0.25) WITHIN GROUP (ORDER BY valor) AS q1,
16     PERCENTILE_CONT(0.50) WITHIN GROUP (ORDER BY valor) AS q2,
17     PERCENTILE_CONT(0.75) WITHIN GROUP (ORDER BY valor) AS q3,
18     PERCENTILE_CONT(0.75) WITHIN GROUP (ORDER BY valor) + 0.5 * (PERCENTILE_CONT(0.75) WITHIN GROUP (ORDER BY valor) - PERCENTILE_CONT(0.25) WITHIN GROUP (ORDER BY valor)) AS limite_superior,
19     MAX(valor) as maximo_valor
20 FROM LancamentosContabeis
21 GROUP BY centro_custo, moeda;

```

Data Output Messages Explain x Notifications

	centro_custo	moeda	minimo_valor	limite_inferior	q1	q2	q3	limite_superior	maximo_valor
	character varying	character varying	numeric	double precision	double precision	double precision	double precision	double precision	numeric
1	Administrativo	BRL	45.22	-123.80500000000000	1022.54	2093.5	3315.23	4461.575	4967.86
2	Administrativo	EUR	10.45	424.29999999999995	1555.36	2898.47	3817.48	4948.54	4982.02
3	Administrativo	USD	0.67	297.54749999999999	1388.79	2265.37	3571.275	4662.5175	4931.50
4	Compras	BRL	5104.85	5331.5830000000001	6556.35	7749.94	9005.88	10230.649999999999	42908.34
5	Compras	EUR	5075.68	5187.9012500000001	6437.52	7447.32	8936.7575	10186.37625	9979.08
6	Compras	USD	5169.48	5472.325	6567.41	7782.67	8757.58	9852.665	26367.48
7	RH	BRL	192.52	778.46	1687.9375	2497.025	3506.8925	4416.37	4781.33
8	RH	EUR	56.17	-349.50500000000001	919.415	2066.8050000000003	3457.255	4726.175	4920.11
9	RH	USD	82.47	1105.8925	1988.68	3014.76	3754.255	4637.0425000000005	4897.08
10	Vendas	BRL	5045.87	4655.09375	6194.14	7613.545	9272.2325	10811.27875	92137.26
11	Vendas	EUR	5067.84	5128.3374999999999	6553.855	7692.3099999999995	8804.8900000000001	10030.407500000001	32821.78
12	Vendas	USD	5074.35	5433.9712499999999	6540.4574999999995	7668.285	8753.43	9859.91625	51416.76

No limit

Query

Query History

```

1  WITH Estatisticas AS (
2      SELECT
3          centro_custo,
4          moeda,
5          PERCENTILE_CONT(0.25) WITHIN GROUP (ORDER BY valor) AS q1,
6          PERCENTILE_CONT(0.75) WITHIN GROUP (ORDER BY valor) AS q3
7      FROM lancamentoscontabeis
8      GROUP BY centro_custo, moeda
9  ),
10  LimitesOutliers AS (
11      SELECT
12          centro_custo,
13          moeda,
14          q1,
15          q3,
16          q1 - 0.5 * (q3 - q1) AS limite_inferior,
17          q3 + 0.5 * (q3 - q1) AS limite_superior
18      FROM Estatisticas
19  )
20  SELECT
21      L.id,
22      L.data_lancamento,
23      L.centro_custo,
24      L.moeda,
25      L.valor
26  FROM lancamentoscontabeis L
27  INNER JOIN LimitesOutliers E
28      ON L.centro_custo = E.centro_custo AND L.moeda = E.moeda
29  WHERE L.valor < E.limite_inferior OR L.valor > E.limite_superior
30  ORDER BY L.valor, L.centro_custo, L.moeda;
31

```

Data Output

Messages

Notifications

	id [PK] integer	data_lancamento date	centro_custo character varying	moeda character varying	valor numeric (15,2)
1	377	2023-12-06	Administrativo	USD	0.67
2	176	2024-07-02	Administrativo	EUR	10.45
3	772	2023-11-27	Administrativo	EUR	79.91
4	568	2024-02-18	RH	USD	82.47
5	144	2023-11-11	Administrativo	USD	93.48
6	861	2024-01-02	Administrativo	USD	118.00
7	259	2023-07-18	RH	USD	154.37
8	384	2024-04-15	Administrativo	EUR	175.64
9	974	2023-12-31	Administrativo	USD	182.65
10	662	2023-10-11	RH	BRL	192.52
11	613	2024-05-10	Administrativo	USD	213.94

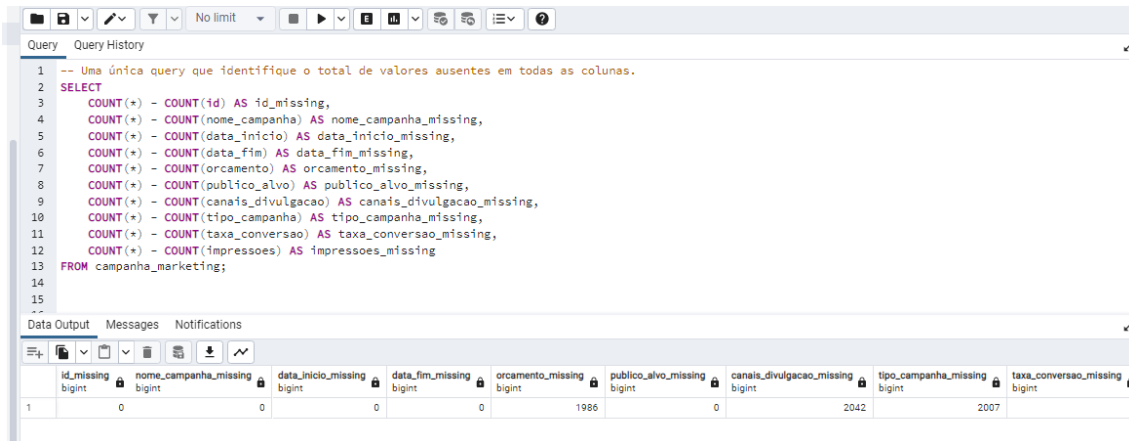
Total rows: 11 of 11

Query complete 00:00:00.150

## 9 Limpeza e Processamento de Dados

### 9.1 Identificação de Dados Faltantes

Uma única query que identifique o total de valores ausentes em todas as colunas.



```

1 -- Uma única query que identifique o total de valores ausentes em todas as colunas.
2 SELECT
3     COUNT(*) - COUNT(id) AS id_missing,
4     COUNT(*) - COUNT(nome_campanha) AS nome_campanha_missing,
5     COUNT(*) - COUNT(data_inicio) AS data_inicio_missing,
6     COUNT(*) - COUNT(data_fim) AS data_fim_missing,
7     COUNT(*) - COUNT(orcamento) AS orcamento_missing,
8     COUNT(*) - COUNT(publico_alvo) AS publico_alvo_missing,
9     COUNT(*) - COUNT(canais_divulgacao) AS canais_divulgacao_missing,
10    COUNT(*) - COUNT(tipo_campanha) AS tipo_campanha_missing,
11    COUNT(*) - COUNT(taxa_conversao) AS taxa_conversao_missing,
12    COUNT(*) - COUNT(impressoes) AS impressoes_missing
13 FROM campanha_marketing;

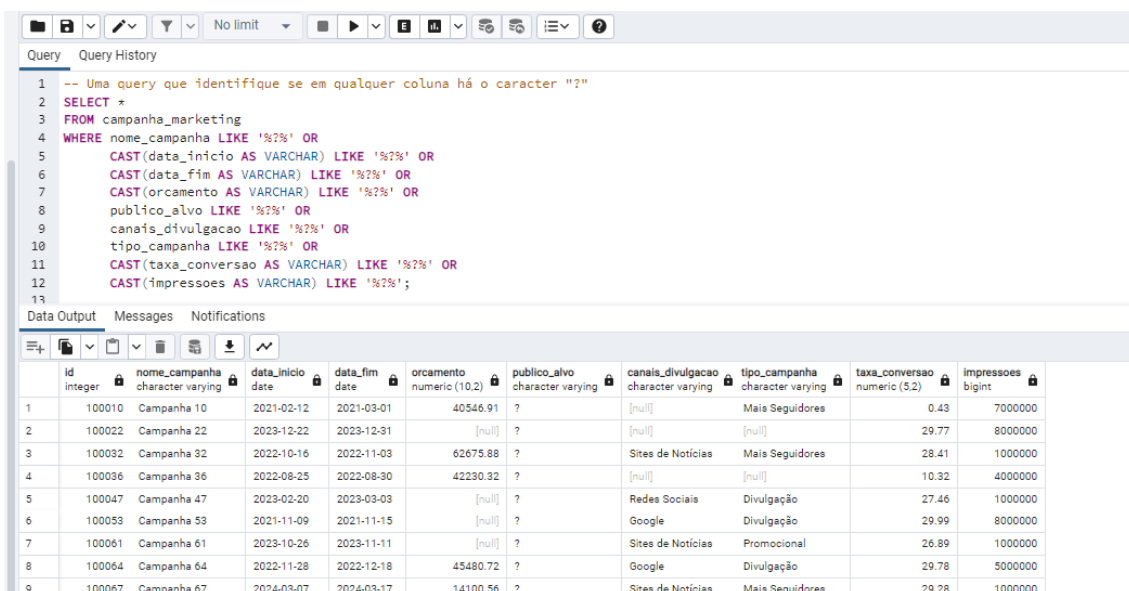
```

	id_missing bigint	nome_campanha_missing bigint	data_inicio_missing bigint	data_fim_missing bigint	orcamento_missing bigint	publico_alvo_missing bigint	canais_divulgacao_missing bigint	tipo_campanha_missing bigint	taxa_conversao_missing bigint
1	0	0	0	0	1986	0	2042	2007	0

Obs: Se não houver valor ausente o resultado deve ser zero "0".

### 9.2 Identificação de Ausência de Informação

Uma query que identifique se em qualquer coluna há o caracter "?"



```

1 -- Uma query que identifique se em qualquer coluna há o caracter "?"
2 SELECT *
3 FROM campanha_marketing
4 WHERE nome_campanha LIKE '%?%' OR
5        CAST(data_inicio AS VARCHAR) LIKE '%?%' OR
6        CAST(data_fim AS VARCHAR) LIKE '%?%' OR
7        CAST(orcamento AS VARCHAR) LIKE '%?%' OR
8        publico_alvo LIKE '%?%' OR
9        canais_divulgacao LIKE '%?%' OR
10       tipo_campanha LIKE '%?%' OR
11       CAST(taxa_conversao AS VARCHAR) LIKE '%?%' OR
12       CAST(impressoes AS VARCHAR) LIKE '%?%';

```

	id integer	nome_campanha character varying	data_inicio date	data_fim date	orcamento numeric (10,2)	publico_alvo character varying	canais_divulgacao character varying	tipo_campanha character varying	taxa_conversao numeric (5,2)	impressoes bigint
1	100010	Campanha 10	2021-02-12	2021-03-01	40546.91	?	[null]	Mais Seguidores	0.43	7000000
2	100022	Campanha 22	2023-12-22	2023-12-31	[null]	?	[null]	[null]	29.77	8000000
3	100032	Campanha 32	2022-10-16	2022-11-03	62675.88	?	Sites de Noticias	Mais Seguidores	28.41	1000000
4	100036	Campanha 36	2022-08-25	2022-08-30	42230.32	?	[null]	[null]	10.32	4000000
5	100047	Campanha 47	2023-02-20	2023-03-03	[null]	?	Redes Sociais	Divulgação	27.46	1000000
6	100053	Campanha 53	2021-11-09	2021-11-15	[null]	?	Google	Divulgação	29.99	8000000
7	100061	Campanha 61	2023-10-26	2023-11-11	[null]	?	Sites de Noticias	Promocional	26.89	1000000
8	100064	Campanha 64	2022-11-28	2022-12-18	45480.72	?	Google	Divulgação	29.78	5000000
9	100067	Campanha 67	2024-03-07	2024-03-17	14100.56	?	Sites de Noticias	Mais Seguidores	29.28	1000000

## 9.3 Identificação de Dados Duplicados

Uma query que identifique duplicatas (sem considerar a coluna id).

```
1 -- Uma query que identifique duplicatas (sem considerar a coluna id).
2 SELECT
3     nome_campanha,
4     data_inicio,
5     data_fim,
6     orcamento,
7     publico_alvo,
8     canais_divulgacao,
9     tipo_campanha,
10    taxa_conversao,
11    impressoes,
12    COUNT(*) as duplicatas
13 FROM campanha_marketing
14 GROUP BY nome_campanha, data_inicio, data_fim, orcamento, publico_alvo, canais_divulgacao, tipo_campanha, taxa_conversao, impressoes
15 HAVING COUNT(*) > 1;
```

The screenshot shows a SQL query editor with a toolbar at the top. Below the query text, there are tabs for 'Query', 'Query History', 'Data Output', 'Messages', and 'Notifications'. The 'Data Output' tab is active, showing a table with 11 columns: nome\_campanha (character varying), data\_inicio (date), data\_fim (date), orcamento (numeric (10,2)), publico\_alvo (character varying), canais\_divulgacao (character varying), tipo\_campanha (character varying), taxa\_conversao (numeric (5,2)), impressoes (bigint), and duplicatas (bigint). The table is currently empty.

Não há dados duplicados.

Uma query que identifique duplicatas considerando as colunas (nome\_campanha, data\_inicio, publico\_alvo, canais\_divulgacao).

```
1 -- Uma query que identifique duplicatas considerando as colunas (nome_campanha, data_inicio, publico_alvo, canais_divulgacao).
2
3 SELECT *
4 FROM campanha_marketing
5 WHERE (nome_campanha, data_inicio, publico_alvo, canais_divulgacao) IN (
6     SELECT nome_campanha, data_inicio, publico_alvo, canais_divulgacao
7     FROM campanha_marketing
8     GROUP BY nome_campanha, data_inicio, publico_alvo, canais_divulgacao
9     HAVING COUNT(*) > 1);
```

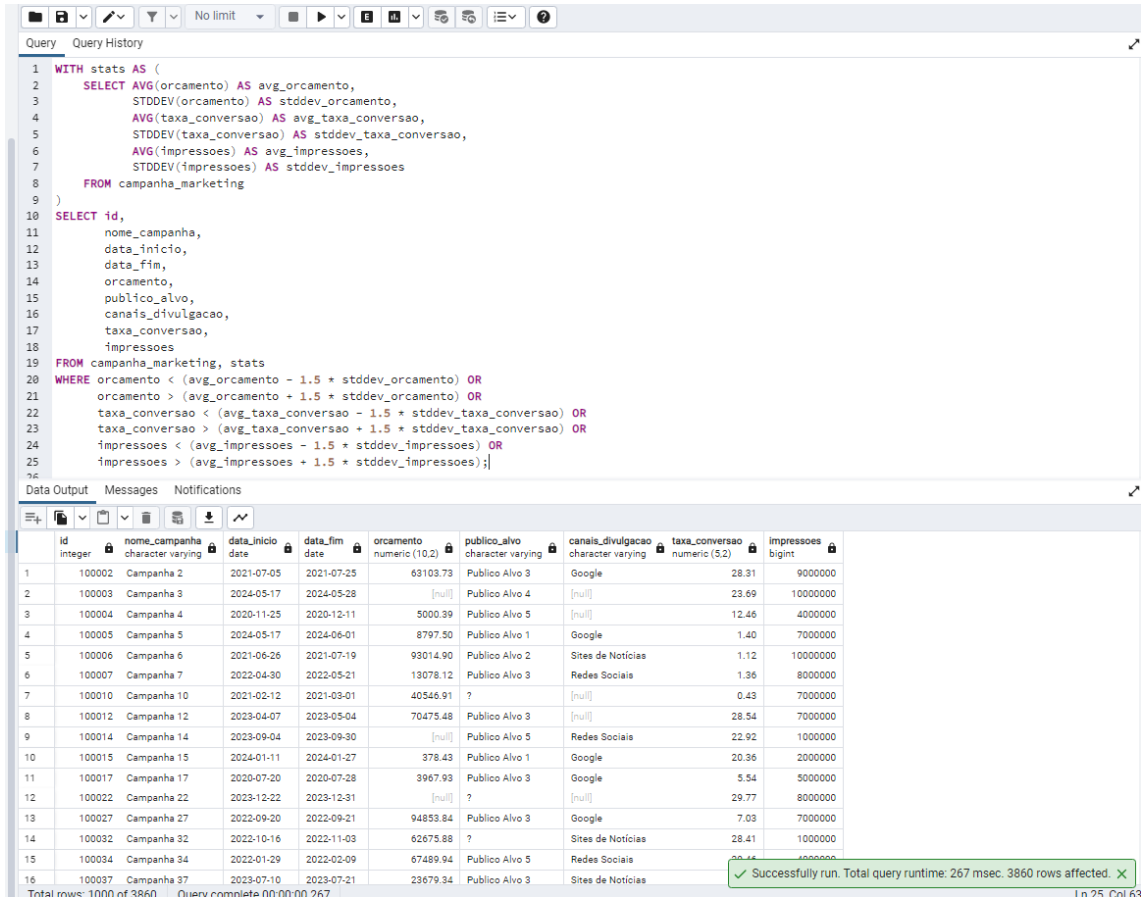
The screenshot shows a SQL query editor with a toolbar at the top. Below the query text, there are tabs for 'Query', 'Query History', 'Data Output', 'Messages', and 'Notifications'. The 'Data Output' tab is active, showing a table with 11 columns: id (integer), nome\_campanha (character varying), data\_inicio (date), data\_fim (date), orcamento (numeric (10,2)), publico\_alvo (character varying), canais\_divulgacao (character varying), tipo\_campanha (character varying), taxa\_conversao (numeric (5,2)), impressoes (bigint), and duplicatas (bigint). The table is currently empty.

## 9.4 Detecção de Outliers

Uma query que identifique outliers nas 3 colunas numéricas.

Considere como outliers valores que estejam acima ou abaixo das seguintes regras:

- $\text{media} + 1.5 * \text{desvio\_padrão}$
- $\text{media} - 1.5 * \text{desvio\_padrão}$



```

1 WITH stats AS (
2     SELECT AVG(orcamento) AS avg_orcamento,
3           STDDEV(orcamento) AS stddev_orcamento,
4           AVG(taxa_conversao) AS avg_taxa_conversao,
5           STDDEV(taxa_conversao) AS stddev_taxa_conversao,
6           AVG(impressoes) AS avg_impressoes,
7           STDDEV(impressoes) AS stddev_impressoes
8     FROM campanha_marketing
9 )
10 SELECT id,
11        nome_campanha,
12        data_inicio,
13        data_fim,
14        orcamento,
15        publico_alvo,
16        canal_divulgacao,
17        taxa_conversao,
18        impressoes
19 FROM campanha_marketing, stats
20 WHERE orcamento < (avg_orcamento - 1.5 * stddev_orcamento) OR
21        orcamento > (avg_orcamento + 1.5 * stddev_orcamento) OR
22        taxa_conversao < (avg_taxa_conversao - 1.5 * stddev_taxa_conversao) OR
23        taxa_conversao > (avg_taxa_conversao + 1.5 * stddev_taxa_conversao) OR
24        impressoes < (avg_impressoes - 1.5 * stddev_impressoes) OR
25        impressoes > (avg_impressoes + 1.5 * stddev_impressoes);

```

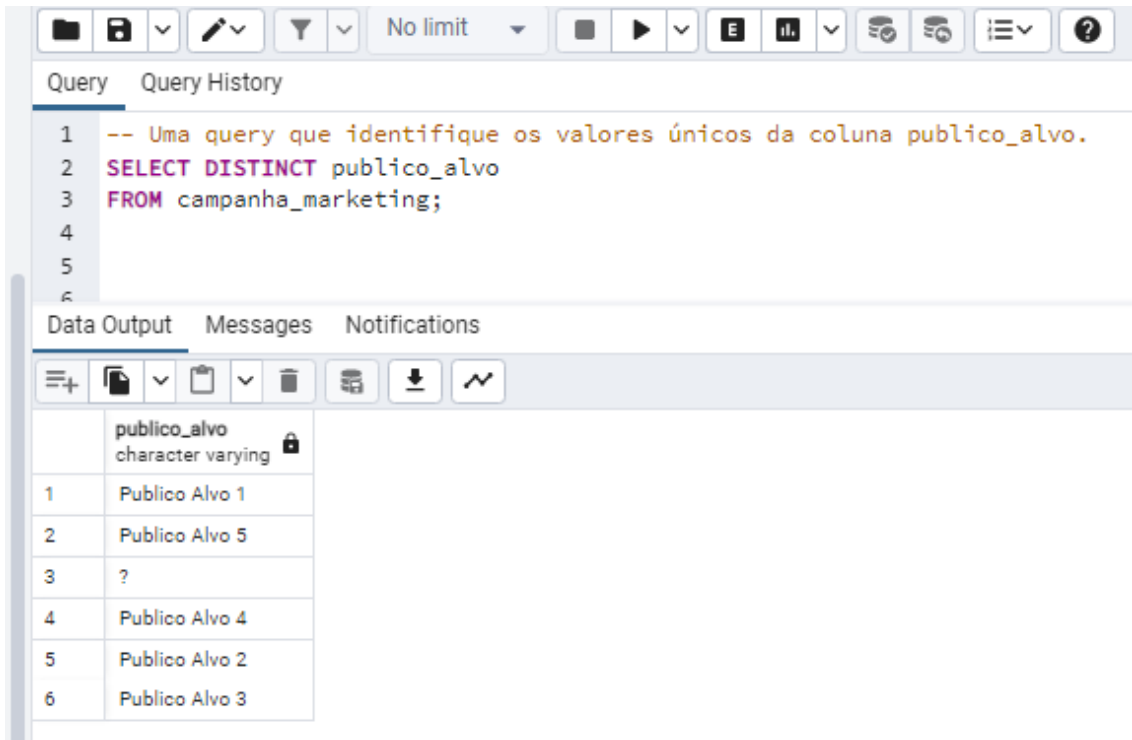
	id	nome_campanha	data_inicio	data_fim	orcamento	publico_alvo	canal_divulgacao	taxa_conversao	impressoes
1	100002	Campanha 2	2021-07-05	2021-07-25	63103.73	Publico Alvo 3	Google	28.31	9000000
2	100003	Campanha 3	2024-05-17	2024-05-28	[null]	Publico Alvo 4	[null]	23.69	10000000
3	100004	Campanha 4	2020-11-25	2020-12-11	5000.39	Publico Alvo 5	[null]	12.46	4000000
4	100005	Campanha 5	2024-05-17	2024-06-01	8797.50	Publico Alvo 1	Google	1.40	7000000
5	100006	Campanha 6	2021-06-26	2021-07-19	93014.90	Publico Alvo 2	Sites de Noticias	1.12	10000000
6	100007	Campanha 7	2022-04-30	2022-05-21	13078.12	Publico Alvo 3	Redes Sociais	1.36	8000000
7	100010	Campanha 10	2021-02-12	2021-03-01	40546.91	?	[null]	0.43	7000000
8	100012	Campanha 12	2023-04-07	2023-05-04	70475.48	Publico Alvo 3	[null]	28.54	7000000
9	100014	Campanha 14	2023-09-04	2023-09-30	[null]	Publico Alvo 5	Redes Sociais	22.92	1000000
10	100015	Campanha 15	2024-01-11	2024-01-27	378.43	Publico Alvo 1	Google	20.36	2000000
11	100017	Campanha 17	2020-07-20	2020-07-28	3967.93	Publico Alvo 3	Google	5.54	5000000
12	100022	Campanha 22	2023-12-22	2023-12-31	[null]	?	[null]	29.77	8000000
13	100027	Campanha 27	2022-09-20	2022-09-21	94853.84	Publico Alvo 3	Google	7.03	7000000
14	100032	Campanha 32	2022-10-16	2022-11-03	62675.88	?	Sites de Noticias	28.41	1000000
15	100034	Campanha 34	2022-01-29	2022-02-09	67489.94	Publico Alvo 5	Redes Sociais	28.16	1000000
16	100037	Campanha 37	2023-07-10	2023-07-21	23679.34	Publico Alvo 3	Sites de Noticias		

Successfully run. Total query runtime: 267 msec. 3860 rows affected.

3.860 registros, tem outliers em pelo menos uma das três colunas numéricas (orcamento, taxa\_conversao e impressoes).

## 9.5 Tratamento de Valores Ausentes

Uma query que identifique os valores únicos da coluna `publico_alvo`.



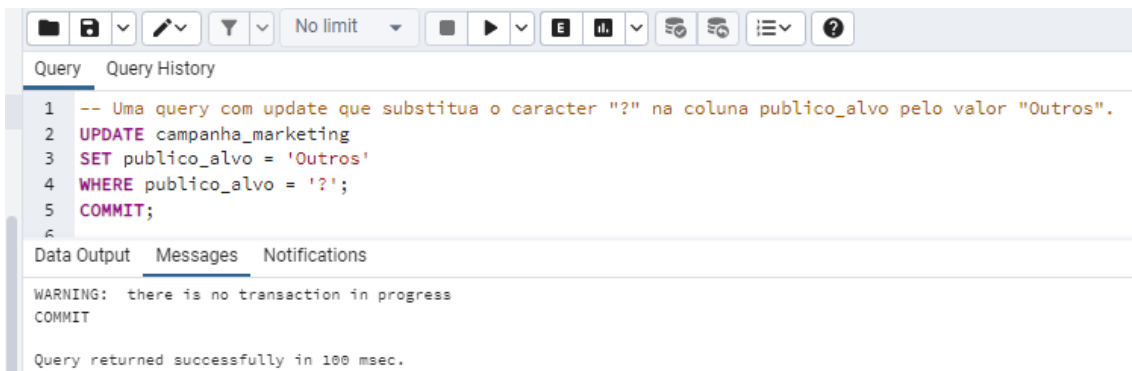
The screenshot shows a SQL IDE interface. The top toolbar includes icons for file operations, query execution, and settings. The 'Query' tab is active, displaying the following SQL code:

```
1 -- Uma query que identifique os valores únicos da coluna publico_alvo.
2 SELECT DISTINCT publico_alvo
3 FROM campanha_marketing;
4
5
6
```

Below the query editor, the 'Data Output' tab is active, showing the results of the query. The results are displayed in a table with two columns: 'publico\_alvo' (character varying) and a lock icon. The data rows are as follows:

	publico_alvo
1	Publico Alvo 1
2	Publico Alvo 5
3	?
4	Publico Alvo 4
5	Publico Alvo 2
6	Publico Alvo 3

Uma query com update que substitua o caracter "?" na coluna `publico_alvo` pelo valor "Outros".



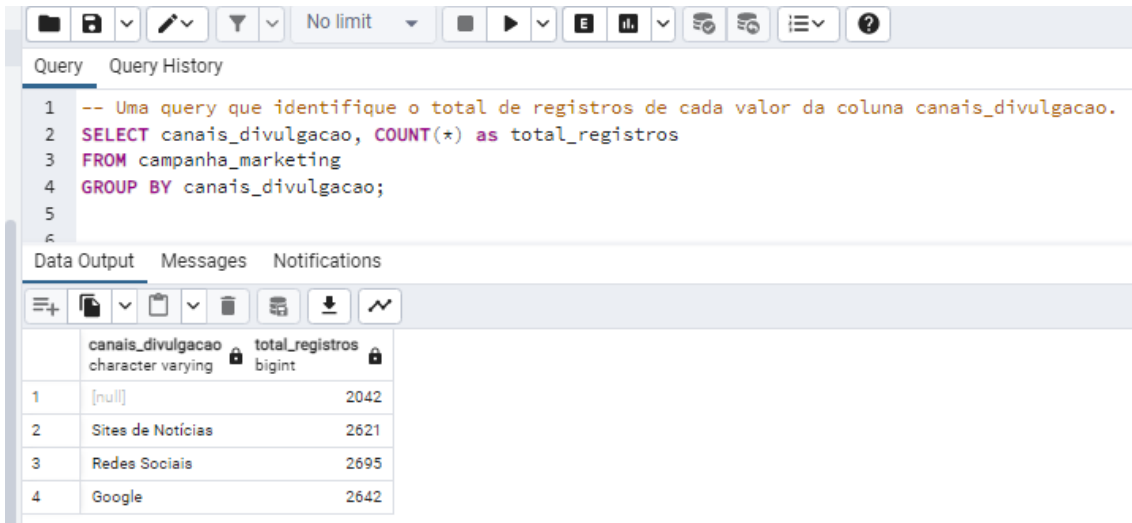
The screenshot shows the same SQL IDE interface. The 'Query' tab is active, displaying the following SQL code:

```
1 -- Uma query com update que substitua o caracter "?" na coluna publico_alvo pelo valor "Outros".
2 UPDATE campanha_marketing
3 SET publico_alvo = 'Outros'
4 WHERE publico_alvo = '?';
5 COMMIT;
6
```

Below the query editor, the 'Messages' tab is active, showing the execution status of the query:

```
WARNING: there is no transaction in progress
COMMIT
Query returned successfully in 100 msec.
```

Uma query que identifique o total de registros de cada valor da coluna `canais_divulgacao`.



The screenshot shows a SQL IDE interface. The top toolbar includes icons for file operations, query execution, and settings. Below the toolbar, there are tabs for 'Query' and 'Query History'. The 'Query' tab is active, displaying a SQL query. Below the query, there are tabs for 'Data Output', 'Messages', and 'Notifications'. The 'Data Output' tab is active, showing a table with the results of the query.

```

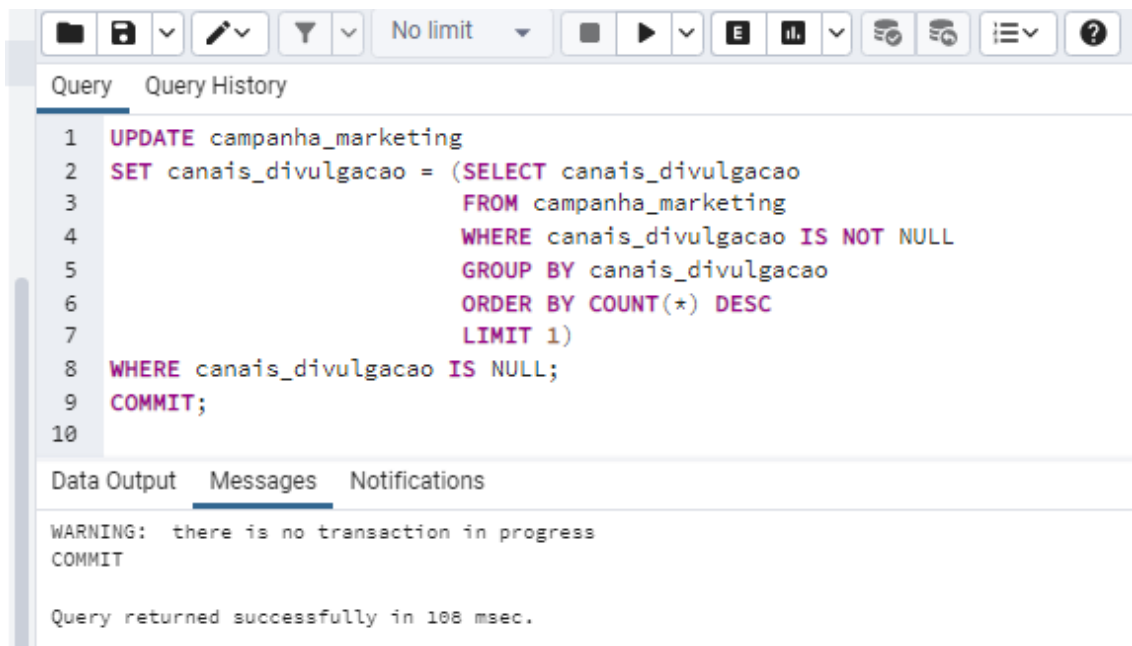
1  -- Uma query que identifique o total de registros de cada valor da coluna canais_divulgacao.
2  SELECT canais_divulgacao, COUNT(*) as total_registros
3  FROM campanha_marketing
4  GROUP BY canais_divulgacao;
5
6

```

	canais_divulgacao character varying	total_registros bigint
1	[null]	2042
2	Sites de Notícias	2621
3	Redes Sociais	2695
4	Google	2642

Uma query com update que substitua os valores ausentes pela moda (uma estatística que indica o valor que aparece com mais frequência) da coluna `canais_divulgacao`.

- Primeiro encontramos a moda da coluna `canais_divulgacao` e depois usamos para fazer o update:



The screenshot shows a SQL IDE interface. The top toolbar includes icons for file operations, query execution, and settings. Below the toolbar, there are tabs for 'Query' and 'Query History'. The 'Query' tab is active, displaying a SQL query. Below the query, there are tabs for 'Data Output', 'Messages', and 'Notifications'. The 'Messages' tab is active, showing a warning message and a success message.

```

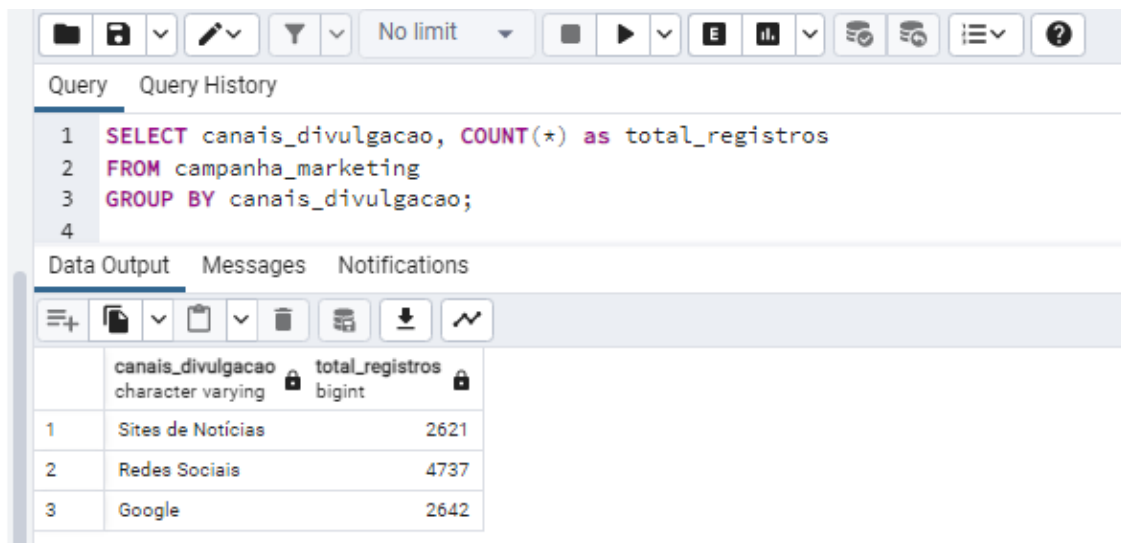
1  UPDATE campanha_marketing
2  SET canais_divulgacao = (SELECT canais_divulgacao
3                          FROM campanha_marketing
4                          WHERE canais_divulgacao IS NOT NULL
5                          GROUP BY canais_divulgacao
6                          ORDER BY COUNT(*) DESC
7                          LIMIT 1)
8  WHERE canais_divulgacao IS NULL;
9  COMMIT;
10

```

WARNING: there is no transaction in progress  
COMMIT

Query returned successfully in 108 msec.

Verificação:



The screenshot shows a SQL query editor with a toolbar at the top. The query is as follows:

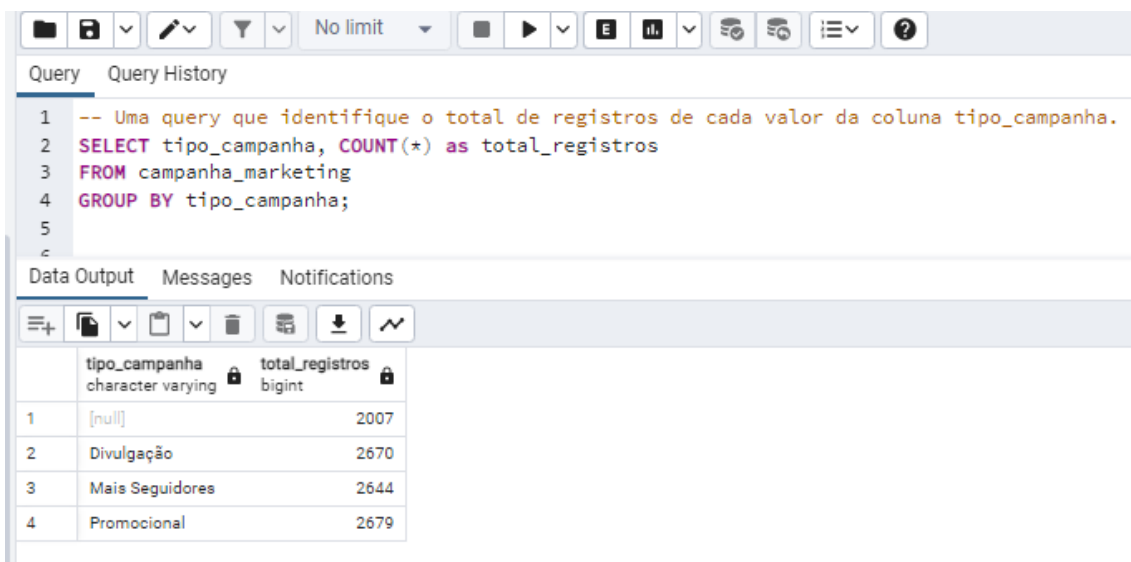
```
1 SELECT canais_divulgacao, COUNT(*) as total_registros
2 FROM campanha_marketing
3 GROUP BY canais_divulgacao;
4
```

Below the query, the 'Data Output' tab is active, displaying a table with the results of the query.

	canais_divulgacao character varying	total_registros bigint
1	Sites de Notícias	2621
2	Redes Sociais	4737
3	Google	2642

Problema resolvido.

Uma query que identifique o total de registros de cada valor da coluna tipo\_campanha.



The screenshot shows a SQL query editor with a toolbar at the top. The query is as follows:

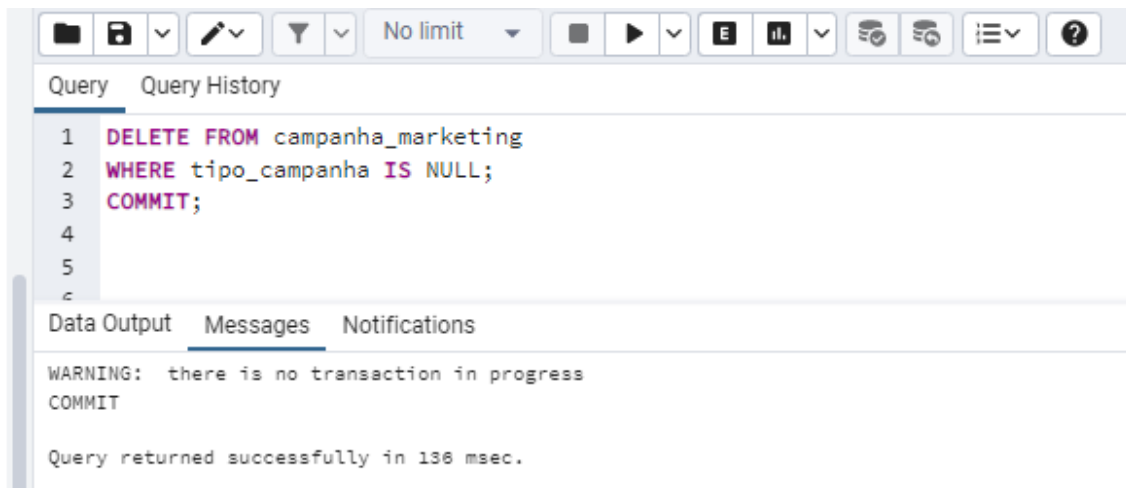
```
1 -- Uma query que identifique o total de registros de cada valor da coluna tipo_campanha.
2 SELECT tipo_campanha, COUNT(*) as total_registros
3 FROM campanha_marketing
4 GROUP BY tipo_campanha;
5
6
```

Below the query, the 'Data Output' tab is active, displaying a table with the results of the query.

	tipo_campanha character varying	total_registros bigint
1	[null]	2007
2	Divulgação	2670
3	Mais Seguidores	2644
4	Promocional	2679



Considere que os valores ausentes na coluna `tipo_campanha` sejam erros de coleta de dados. Crie uma query com delete que remova os registros onde `tipo_campanha` tiver valor nulo.



```

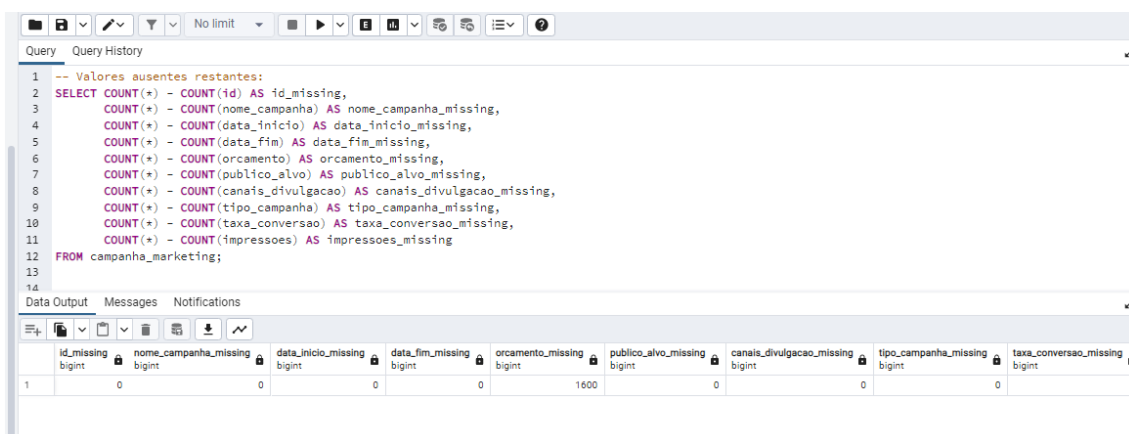
1 DELETE FROM campanha_marketing
2 WHERE tipo_campanha IS NULL;
3 COMMIT;

```

WARNING: there is no transaction in progress  
COMMIT

Query returned successfully in 136 msec.

Verificando se ainda existe valores ausentes.



```

1 -- Valores ausentes restantes:
2 SELECT COUNT(*) - COUNT(id) AS id_missing,
3        COUNT(*) - COUNT(nome_campanha) AS nome_campanha_missing,
4        COUNT(*) - COUNT(data_inicio) AS data_inicio_missing,
5        COUNT(*) - COUNT(data_fim) AS data_fim_missing,
6        COUNT(*) - COUNT(orcamento) AS orcamento_missing,
7        COUNT(*) - COUNT(publico_alvo) AS publico_alvo_missing,
8        COUNT(*) - COUNT(canal_divulgacao) AS canal_divulgacao_missing,
9        COUNT(*) - COUNT(tipo_campanha) AS tipo_campanha_missing,
10       COUNT(*) - COUNT(taxa_conversao) AS taxa_conversao_missing,
11       COUNT(*) - COUNT(impressoes) AS impressoes_missing
12 FROM campanha_marketing;

```

	id_missing bigint	nome_campanha_missing bigint	data_inicio_missing bigint	data_fim_missing bigint	orcamento_missing bigint	publico_alvo_missing bigint	canal_divulgacao_missing bigint	tipo_campanha_missing bigint	taxa_conversao_missing bigint
1	0	0	0	0	1600	0	0	0	0

Ainda existe valores ausentes na coluna 'orcamento'.

Uma query que identifique valores ausentes na coluna orçamento.

Query Query History

```

1 -- Uma query que identifique valores ausentes na coluna orçamento
2 SELECT *
3 FROM campanha_marketing
4 WHERE orcamento IS NULL;
5

```

Data Output Messages Notifications

	id integer	nome_campanha character varying	data_inicio date	data_fim date	orcamento numeric (10,2)	publico_alvo character varying	canais_divulgacao character varying	tipo_campanha character varying	taxa_conversao numeric (5,2)	impressoes bigint
1	100008	Campanha 8	2022-09-10	2022-09-28	[null]	Publico Alvo 4	Google	Mais Seguidores	5.94	3000000
2	100009	Campanha 9	2024-02-23	2024-03-09	[null]	Publico Alvo 5	Sites de Notícias	Divulgação	22.06	2000000
3	100014	Campanha 14	2023-09-04	2023-09-30	[null]	Publico Alvo 5	Redes Sociais	Mais Seguidores	22.92	1000000
4	100048	Campanha 48	2024-02-10	2024-02-15	[null]	Publico Alvo 4	Sites de Notícias	Promocional	15.36	4000000
5	100051	Campanha 51	2021-09-19	2021-10-17	[null]	Publico Alvo 2	Redes Sociais	Promocional	0.60	8000000
6	100059	Campanha 59	2023-12-30	2024-01-01	[null]	Publico Alvo 5	Google	Promocional	1.41	6000000
7	100062	Campanha 62	2023-10-12	2023-11-02	[null]	Publico Alvo 3	Sites de Notícias	Mais Seguidores	23.36	7000000
8	100003	Campanha 3	2024-05-17	2024-05-28	[null]	Publico Alvo 4	Redes Sociais	Divulgação	23.69	10000000
9	100019	Campanha 19	2020-11-11	2020-12-09	[null]	Publico Alvo 5	Redes Sociais	Mais Seguidores	17.26	7000000
10	100025	Campanha 25	2023-08-28	2023-09-27	[null]	Publico Alvo 1	Redes Sociais	Promocional	19.41	5000000
11	100080	Campanha 80	2022-09-07	2022-10-03	[null]	Publico Alvo 1	Google	Mais Seguidores	6.32	6000000
12	100084	Campanha 84	2024-04-11	2024-04-19	[null]	Publico Alvo 5	Sites de Notícias	Divulgação	12.90	3000000
13	100088	Campanha 88	2024-05-28	2024-06-16	[null]	Publico Alvo 4	Sites de Notícias	Promocional	6.55	1000000
14	100109	Campanha 109	2022-07-10	2022-07-21	[null]	Publico Alvo 5	Google	Divulgação	19.37	7000000

Considere que orçamento nulo para público alvo igual "Outros" não seja uma informação relevante.

Crie uma query com delete que remova registros se a coluna orcamento tiver valor ausente e a coluna publico\_alvo tiver o valor "Outros".

Query Query History

```

1 -- Considere que orçamento nulo para público alvo igual "Outros" não seja uma informação relevante.
2 -- Crie uma query com delete que remova registros se a coluna orcamento tiver valor ausente e a coluna
3 -- publico_alvo tiver o valor "Outros".
4 DELETE FROM campanha_marketing
5 WHERE orcamento IS NULL
6 AND publico_alvo = 'Outros';
7 COMMIT;
8

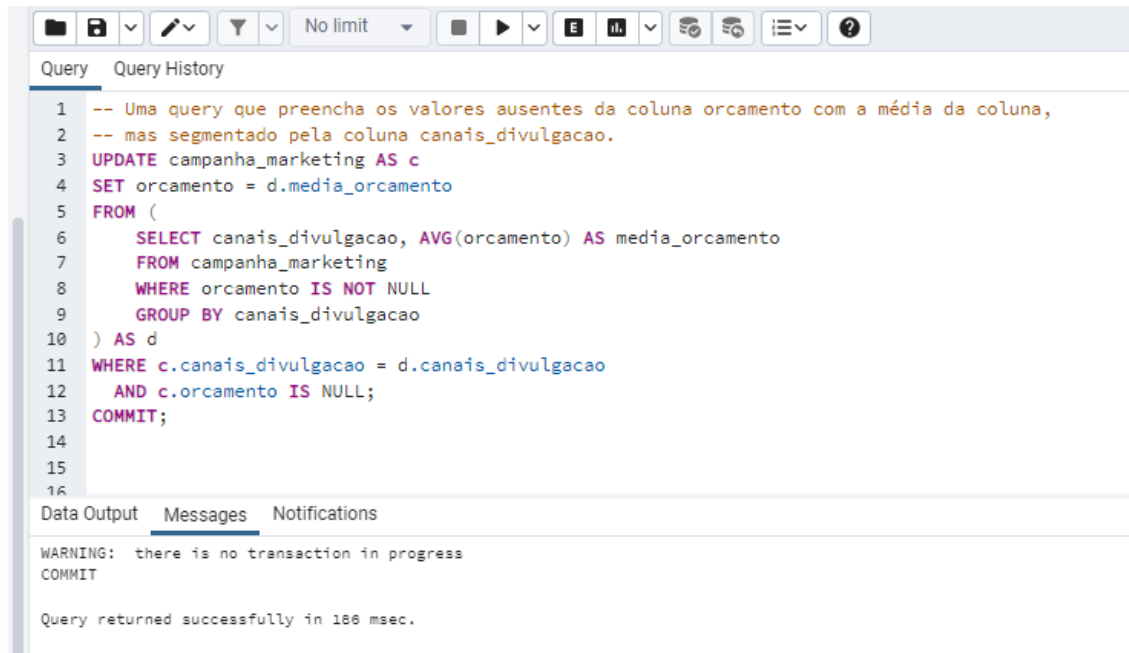
```

Data Output Messages Notifications

WARNING: there is no transaction in progress  
COMMIT

Query returned successfully in 189 msec.

Uma query que preencha os valores ausentes da coluna orcamento com a média da coluna, mas segmentado pela coluna canais\_divulgacao.



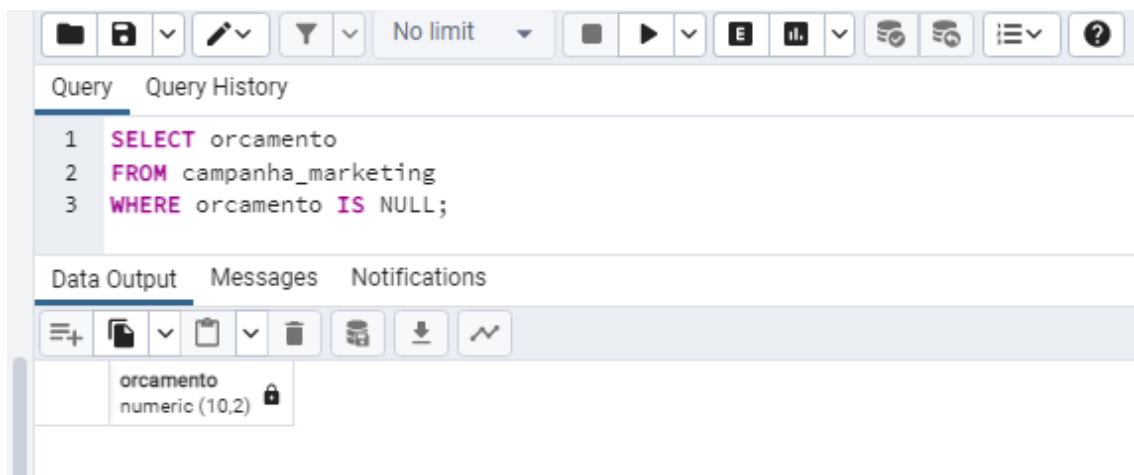
```

1  -- Uma query que preencha os valores ausentes da coluna orcamento com a média da coluna,
2  -- mas segmentado pela coluna canais_divulgacao.
3  UPDATE campanha_marketing AS c
4  SET orcamento = d.media_orcamento
5  FROM (
6  SELECT canais_divulgacao, AVG(orcamento) AS media_orcamento
7  FROM campanha_marketing
8  WHERE orcamento IS NOT NULL
9  GROUP BY canais_divulgacao
10 ) AS d
11 WHERE c.canais_divulgacao = d.canais_divulgacao
12 AND c.orcamento IS NULL;
13 COMMIT;
14
15
16

```

Query returned successfully in 186 msec.

Verificação de valores ausentes na coluna 'orcamento'



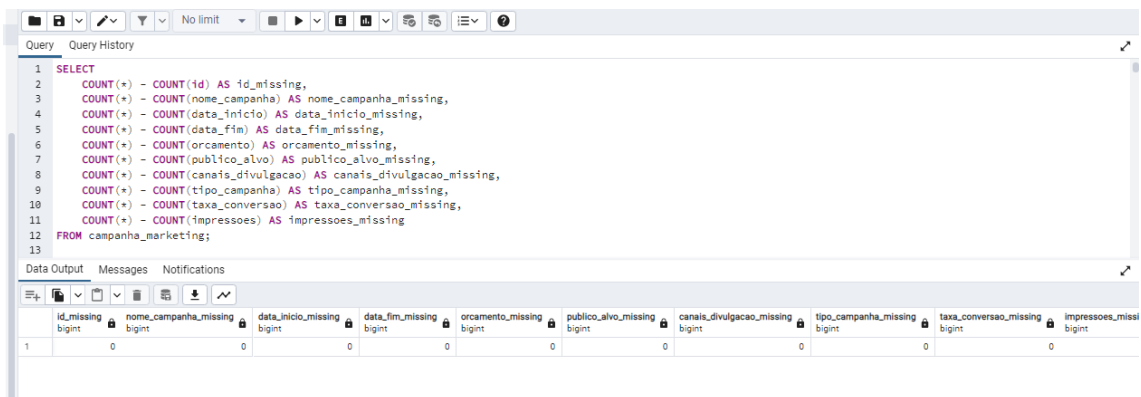
```

1  SELECT orcamento
2  FROM campanha_marketing
3  WHERE orcamento IS NULL;

```

orcamento  
numeric (10,2)

### Verificação de valores ausentes na tabela.



```
1 SELECT
2     COUNT(*) - COUNT(id) AS id_missing,
3     COUNT(*) - COUNT(nome_campanha) AS nome_campanha_missing,
4     COUNT(*) - COUNT(data_inicio) AS data_inicio_missing,
5     COUNT(*) - COUNT(data_fim) AS data_fim_missing,
6     COUNT(*) - COUNT(orcamento) AS orcamento_missing,
7     COUNT(*) - COUNT(publico_alvo) AS publico_alvo_missing,
8     COUNT(*) - COUNT(canais_divulgacao) AS canais_divulgacao_missing,
9     COUNT(*) - COUNT(tipo_campanha) AS tipo_campanha_missing,
10    COUNT(*) - COUNT(taxa_conversao) AS taxa_conversao_missing,
11    COUNT(*) - COUNT(impressoes) AS impressoes_missing
12 FROM campanha_marketing;
13
```

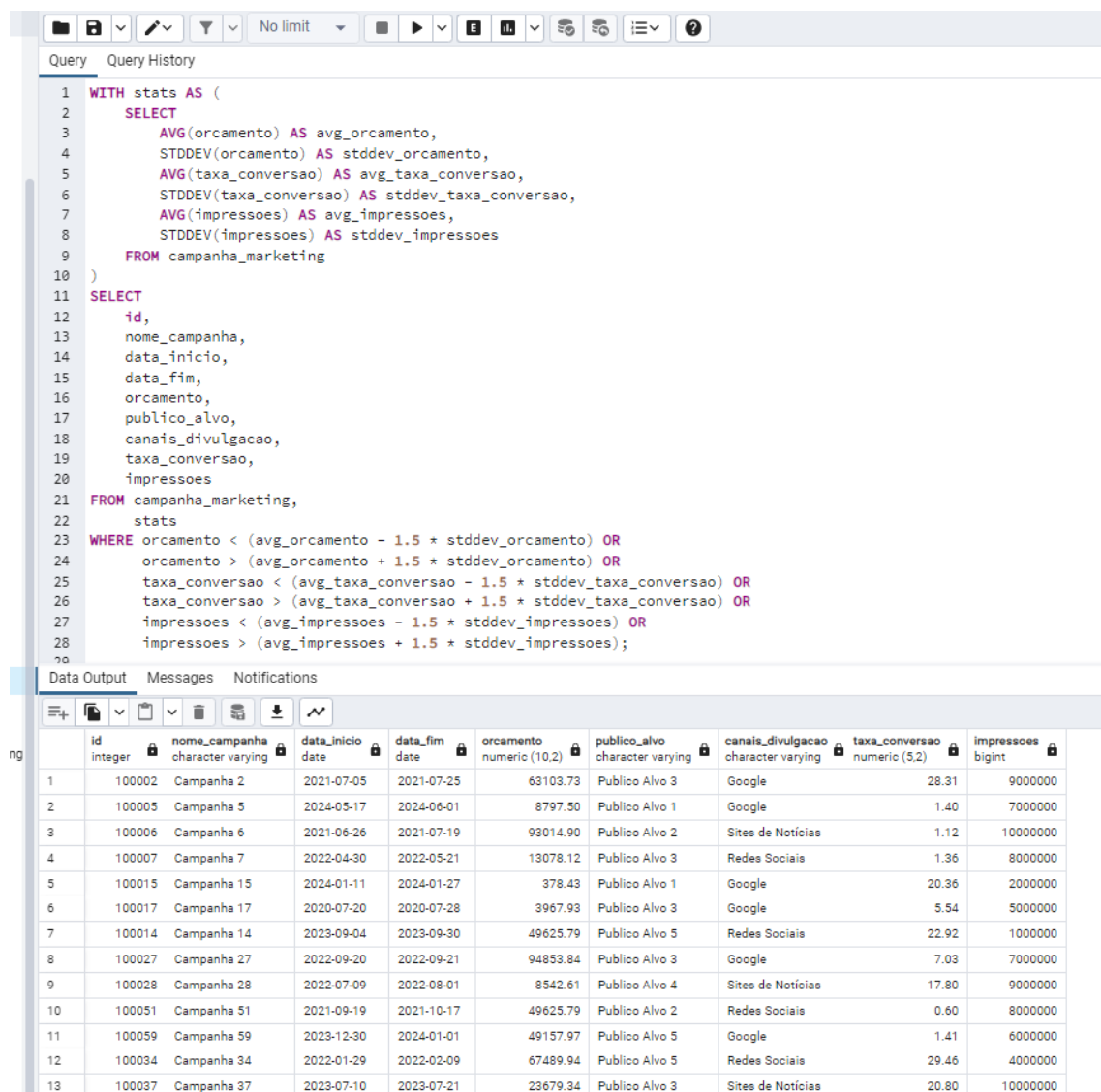
	id_missing	nome_campanha_missing	data_inicio_missing	data_fim_missing	orcamento_missing	publico_alvo_missing	canais_divulgacao_missing	tipo_campanha_missing	taxa_conversao_missing	impressoes_missing
1	0	0	0	0	0	0	0	0	0	0

Não há mais valores ausentes na tabela.

## 9.6 Tratamento de Outliers

Usando como estratégia de tratamento de outliers criando uma nova coluna e preenchê-la com True se houver outlier no registro e False caso contrário.

Identificando os outliers (na clausula 'where' são as regras de identificação de outliers):



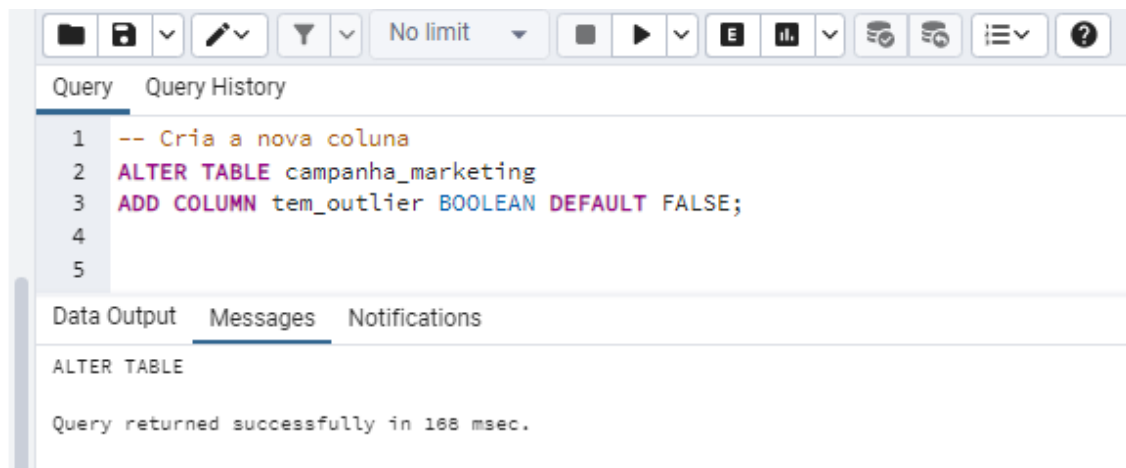
```

1 WITH stats AS (
2     SELECT
3         AVG(orcamento) AS avg_orcamento,
4         STDDEV(orcamento) AS stddev_orcamento,
5         AVG(taxa_conversao) AS avg_taxa_conversao,
6         STDDEV(taxa_conversao) AS stddev_taxa_conversao,
7         AVG(impressoes) AS avg_impressoes,
8         STDDEV(impressoes) AS stddev_impressoes
9     FROM campanha_marketing
10 )
11 SELECT
12     id,
13     nome_campanha,
14     data_inicio,
15     data_fim,
16     orcamento,
17     publico_alvo,
18     canais_divulgacao,
19     taxa_conversao,
20     impressoes
21 FROM campanha_marketing,
22 stats
23 WHERE orcamento < (avg_orcamento - 1.5 * stddev_orcamento) OR
24        orcamento > (avg_orcamento + 1.5 * stddev_orcamento) OR
25        taxa_conversao < (avg_taxa_conversao - 1.5 * stddev_taxa_conversao) OR
26        taxa_conversao > (avg_taxa_conversao + 1.5 * stddev_taxa_conversao) OR
27        impressoes < (avg_impressoes - 1.5 * stddev_impressoes) OR
28        impressoes > (avg_impressoes + 1.5 * stddev_impressoes);

```

	id	nome_campanha	data_inicio	data_fim	orcamento	publico_alvo	canais_divulgacao	taxa_conversao	impressoes
	integer	character varying	date	date	numeric (10,2)	character varying	character varying	numeric (5,2)	bigint
1	100002	Campanha 2	2021-07-05	2021-07-25	63103.73	Publico Alvo 3	Google	28.31	9000000
2	100005	Campanha 5	2024-05-17	2024-06-01	8797.50	Publico Alvo 1	Google	1.40	7000000
3	100006	Campanha 6	2021-06-26	2021-07-19	93014.90	Publico Alvo 2	Sites de Notícias	1.12	10000000
4	100007	Campanha 7	2022-04-30	2022-05-21	13078.12	Publico Alvo 3	Redes Sociais	1.36	8000000
5	100015	Campanha 15	2024-01-11	2024-01-27	378.43	Publico Alvo 1	Google	20.36	2000000
6	100017	Campanha 17	2020-07-20	2020-07-28	3967.93	Publico Alvo 3	Google	5.54	5000000
7	100014	Campanha 14	2023-09-04	2023-09-30	49625.79	Publico Alvo 5	Redes Sociais	22.92	1000000
8	100027	Campanha 27	2022-09-20	2022-09-21	94853.84	Publico Alvo 3	Google	7.03	7000000
9	100028	Campanha 28	2022-07-09	2022-08-01	8542.61	Publico Alvo 4	Sites de Notícias	17.80	9000000
10	100051	Campanha 51	2021-09-19	2021-10-17	49625.79	Publico Alvo 2	Redes Sociais	0.60	8000000
11	100059	Campanha 59	2023-12-30	2024-01-01	49157.97	Publico Alvo 5	Google	1.41	6000000
12	100034	Campanha 34	2022-01-29	2022-02-09	67489.94	Publico Alvo 5	Redes Sociais	29.46	4000000
13	100037	Campanha 37	2023-07-10	2023-07-21	23679.34	Publico Alvo 3	Sites de Notícias	20.80	10000000

## Cria a nova coluna



The screenshot shows a SQL IDE interface with a toolbar at the top containing icons for file operations, query execution, and settings. Below the toolbar, there are tabs for 'Query' and 'Query History'. The 'Query' tab is active, displaying the following SQL code:

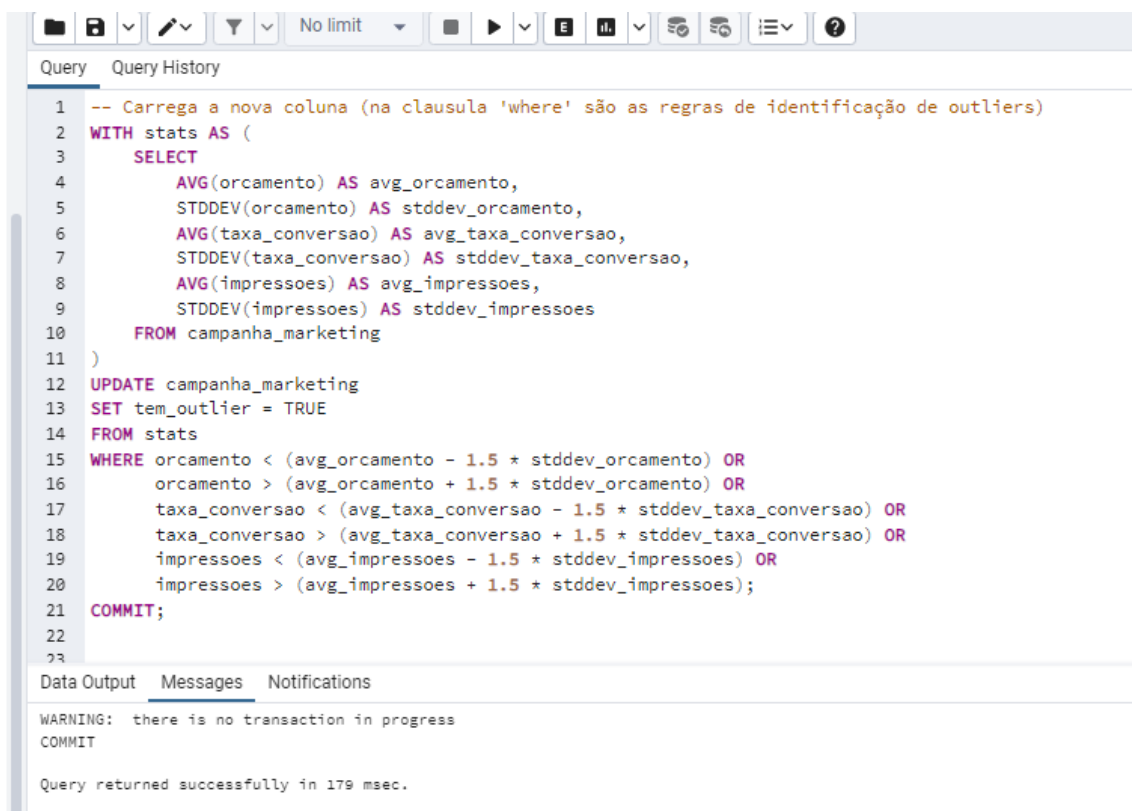
```
1 -- Cria a nova coluna
2 ALTER TABLE campanha_marketing
3 ADD COLUMN tem_outlier BOOLEAN DEFAULT FALSE;
4
5
```

Below the code editor, there are tabs for 'Data Output', 'Messages', and 'Notifications'. The 'Messages' tab is active, showing the following output:

```
ALTER TABLE

Query returned successfully in 168 msec.
```

## Carrega a nova coluna (na clausula 'where' são as regras de identificação de outliers)



The screenshot shows a SQL IDE interface with a toolbar at the top. Below the toolbar, there are tabs for 'Query' and 'Query History'. The 'Query' tab is active, displaying the following SQL code:

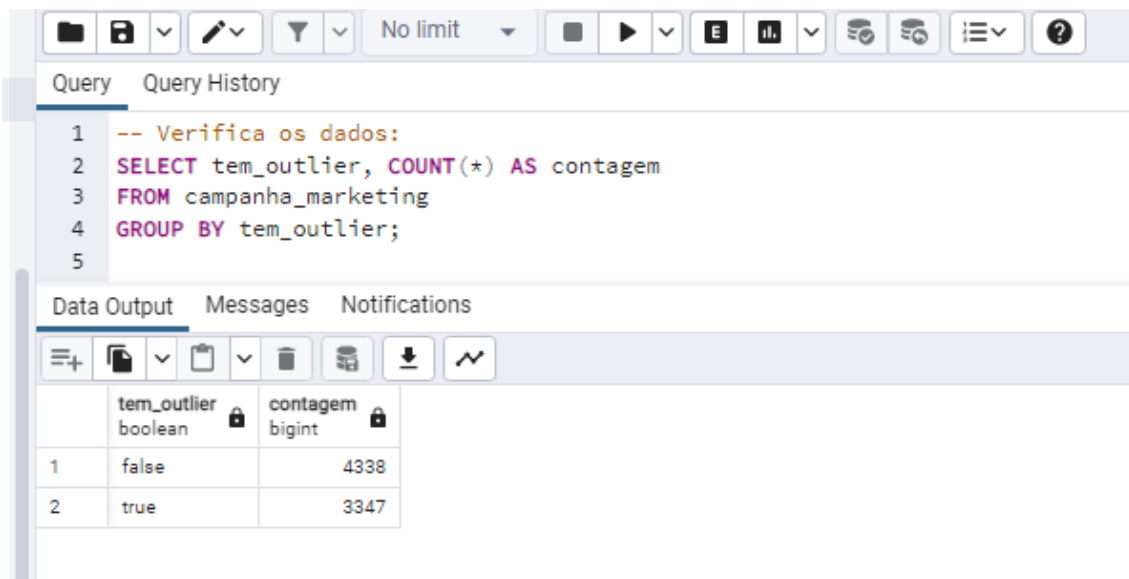
```
1 -- Carrega a nova coluna (na clausula 'where' são as regras de identificação de outliers)
2 WITH stats AS (
3     SELECT
4         AVG(orcamento) AS avg_orcamento,
5         STDDEV(orcamento) AS stddev_orcamento,
6         AVG(taxa_conversao) AS avg_taxa_conversao,
7         STDDEV(taxa_conversao) AS stddev_taxa_conversao,
8         AVG(impressoes) AS avg_impressoes,
9         STDDEV(impressoes) AS stddev_impressoes
10    FROM campanha_marketing
11 )
12 UPDATE campanha_marketing
13 SET tem_outlier = TRUE
14 FROM stats
15 WHERE orcamento < (avg_orcamento - 1.5 * stddev_orcamento) OR
16        orcamento > (avg_orcamento + 1.5 * stddev_orcamento) OR
17        taxa_conversao < (avg_taxa_conversao - 1.5 * stddev_taxa_conversao) OR
18        taxa_conversao > (avg_taxa_conversao + 1.5 * stddev_taxa_conversao) OR
19        impressoes < (avg_impressoes - 1.5 * stddev_impressoes) OR
20        impressoes > (avg_impressoes + 1.5 * stddev_impressoes);
21 COMMIT;
22
23
```

Below the code editor, there are tabs for 'Data Output', 'Messages', and 'Notifications'. The 'Messages' tab is active, showing the following output:

```
WARNING: there is no transaction in progress
COMMIT

Query returned successfully in 179 msec.
```

## Verifica os dados



The screenshot shows a SQL query editor with a toolbar at the top. The query is as follows:

```

1  -- Verifica os dados:
2  SELECT tem_outlier, COUNT(*) AS contagem
3  FROM campanha_marketing
4  GROUP BY tem_outlier;
5

```

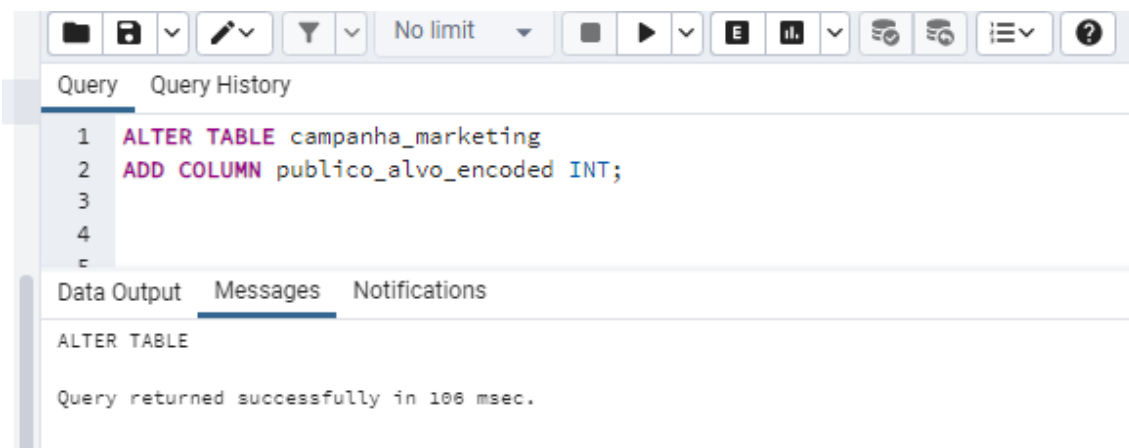
Below the query editor, the 'Data Output' tab is active, displaying the results of the query in a table format:

	tem_outlier boolean	contagem bigint
1	false	4338
2	true	3347

## 9.7 Label Encoding com Linguagem SQL

Aplicando label encoding na coluna publico\_alvo e salvando o resultado em uma nova coluna chamada publico\_alvo\_encoded.

### Criando uma nova coluna



The screenshot shows a SQL query editor with a toolbar at the top. The query is as follows:

```

1  ALTER TABLE campanha_marketing
2  ADD COLUMN publico_alvo_encoded INT;
3
4
5

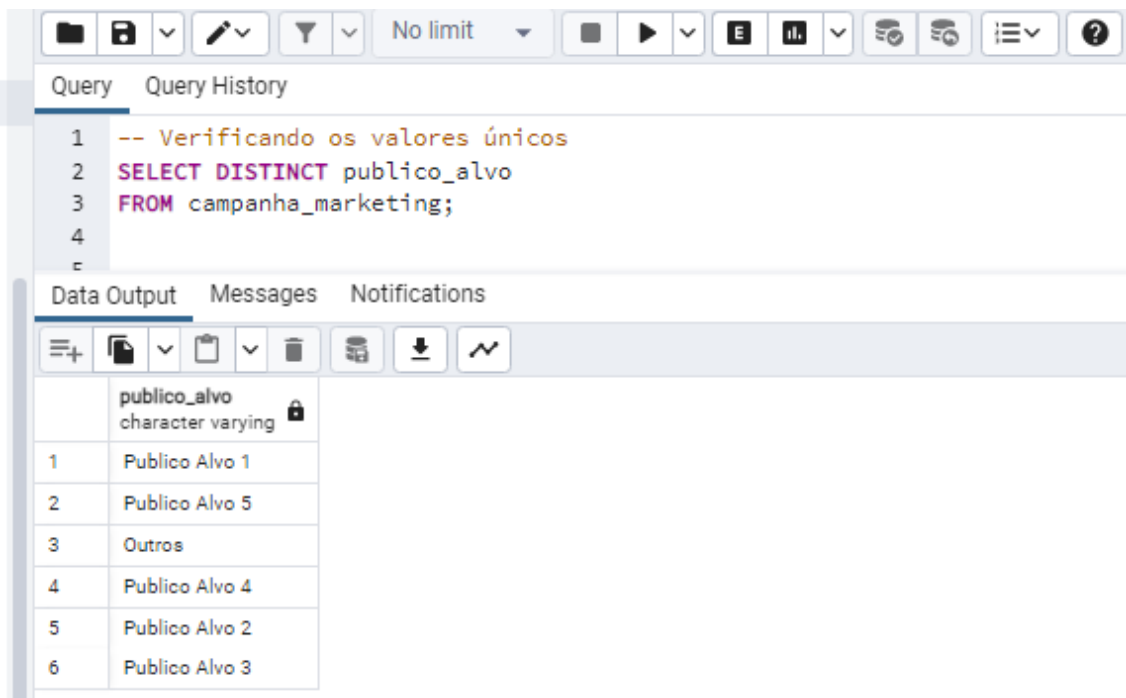
```

Below the query editor, the 'Data Output' tab is active, displaying the results of the query in a table format:

	tem_outlier boolean	contagem bigint
1	false	4338
2	true	3347

Query returned successfully in 106 msec.

## Verificando os valores únicos



Query Query History

```

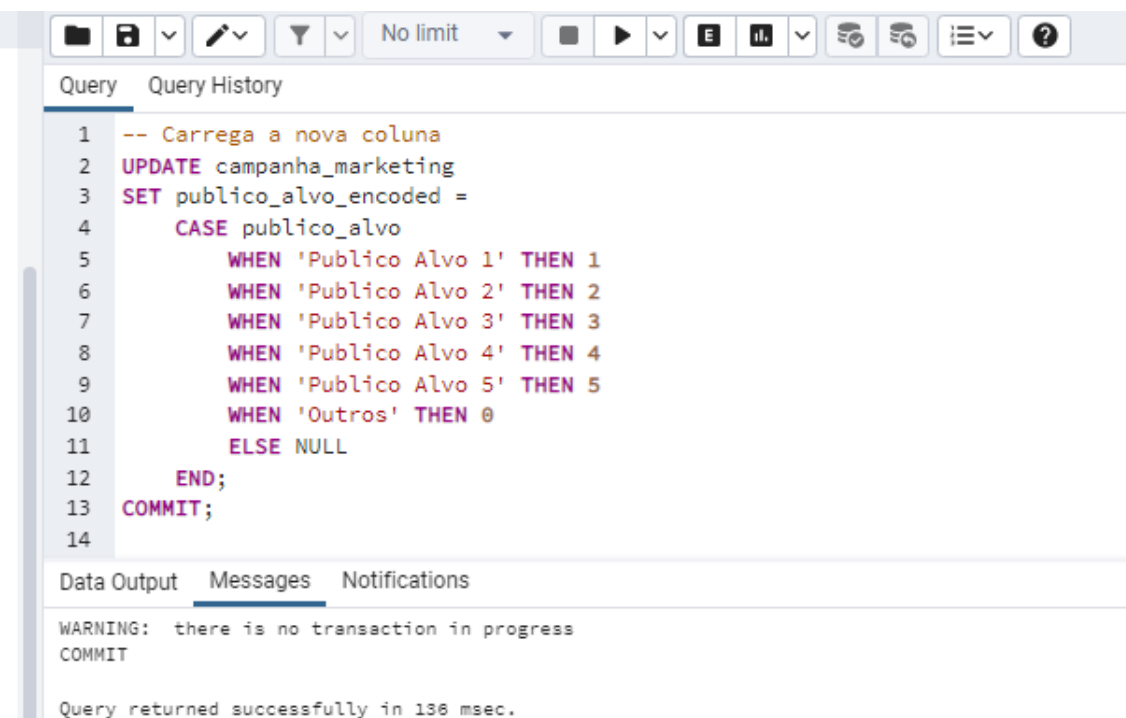
1 -- Verificando os valores únicos
2 SELECT DISTINCT publico_alvo
3 FROM campanha_marketing;
4
5

```

Data Output Messages Notifications

	publico_alvo character varying
1	Publico Alvo 1
2	Publico Alvo 5
3	Outros
4	Publico Alvo 4
5	Publico Alvo 2
6	Publico Alvo 3

## Carrega a nova coluna



Query Query History

```

1 -- Carrega a nova coluna
2 UPDATE campanha_marketing
3 SET publico_alvo_encoded =
4     CASE publico_alvo
5         WHEN 'Publico Alvo 1' THEN 1
6         WHEN 'Publico Alvo 2' THEN 2
7         WHEN 'Publico Alvo 3' THEN 3
8         WHEN 'Publico Alvo 4' THEN 4
9         WHEN 'Publico Alvo 5' THEN 5
10        WHEN 'Outros' THEN 0
11        ELSE NULL
12    END;
13 COMMIT;
14

```

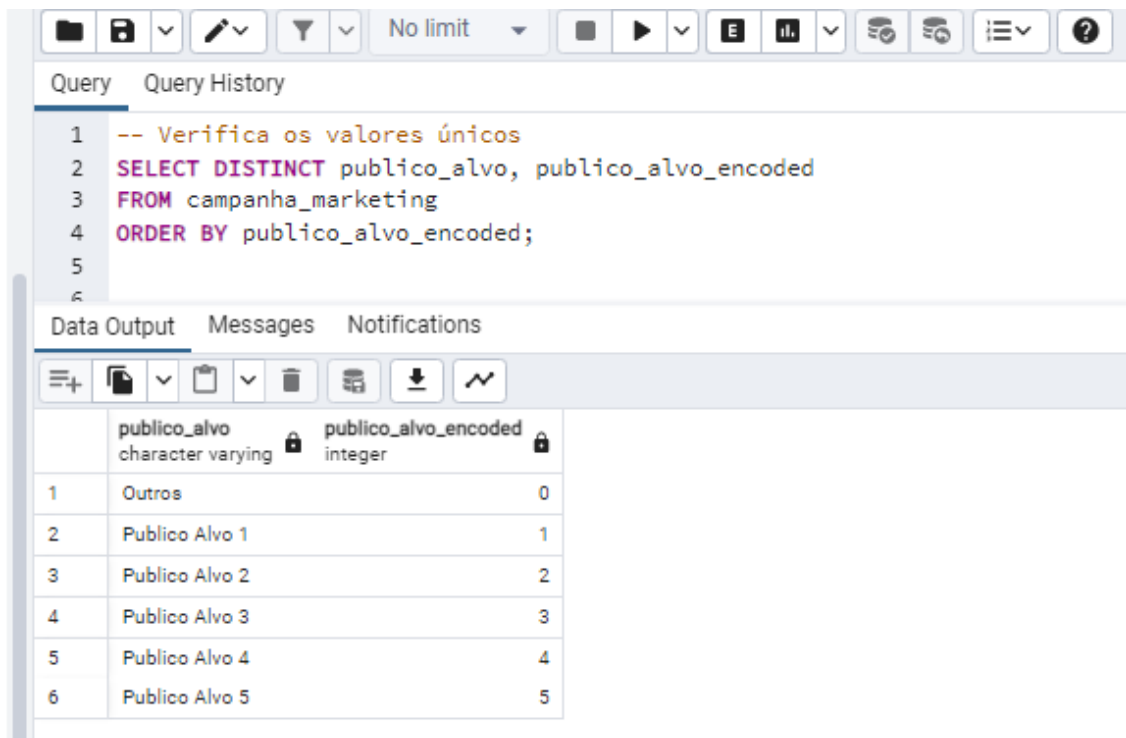
Data Output Messages Notifications

WARNING: there is no transaction in progress  
COMMIT

Query returned successfully in 136 msec.



## Verifica os valores únicos



The screenshot shows a SQL query editor with the following query:

```

1  -- Verifica os valores únicos
2  SELECT DISTINCT publico_alvo, publico_alvo_encoded
3  FROM campanha_marketing
4  ORDER BY publico_alvo_encoded;

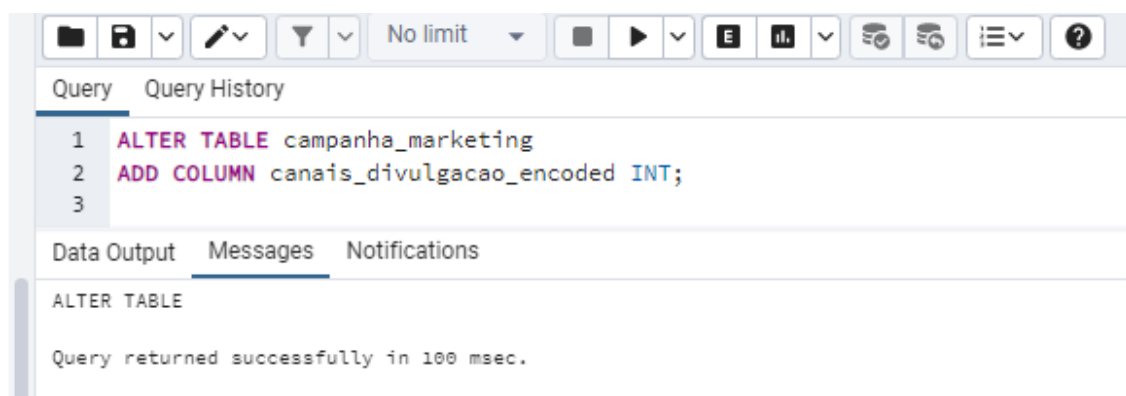
```

The query results are displayed in a table with two columns: `publico_alvo` (character varying) and `publico_alvo_encoded` (integer). The results are as follows:

	publico_alvo	publico_alvo_encoded
1	Outros	0
2	Publico Alvo 1	1
3	Publico Alvo 2	2
4	Publico Alvo 3	3
5	Publico Alvo 4	4
6	Publico Alvo 5	5

Aplicando label encoding na coluna `canais_divulgacao` e salvando o resultado em uma nova coluna chamada `canais_divulgacao_encoded`.

## Cria a nova coluna



The screenshot shows a SQL query editor with the following query:

```

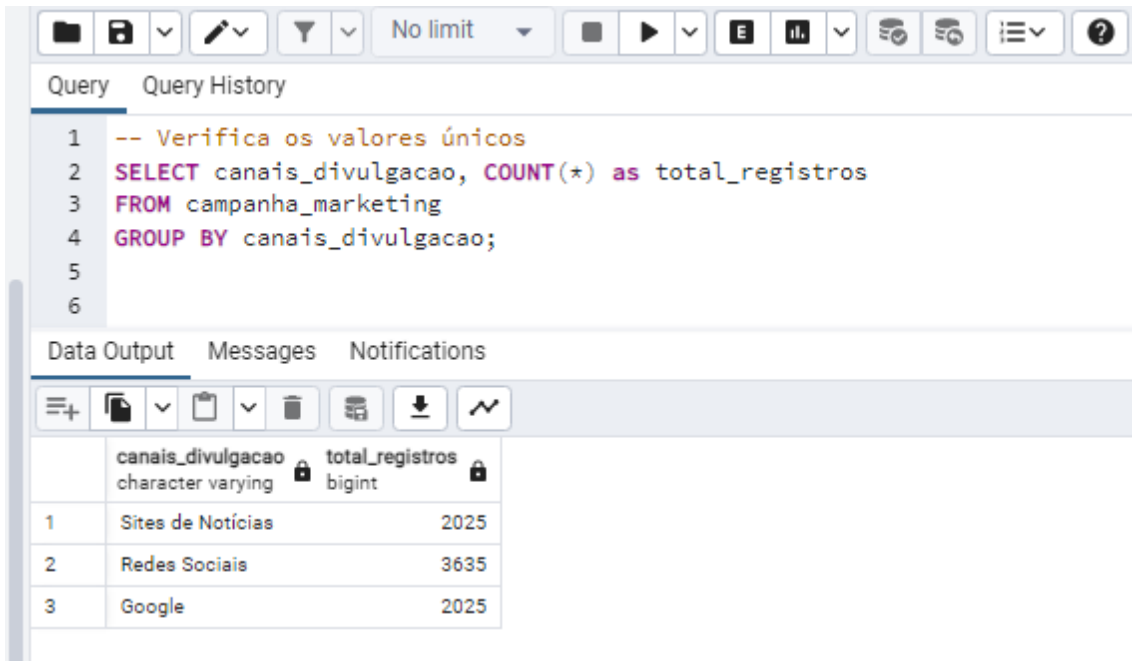
1  ALTER TABLE campanha_marketing
2  ADD COLUMN canais_divulgacao_encoded INT;
3

```

The query results are displayed in a table with the following message:

ALTER TABLE
Query returned successfully in 100 msec.

## Verifica os valores únicos



The screenshot shows a SQL IDE interface with a query editor and a data output table. The query is as follows:

```

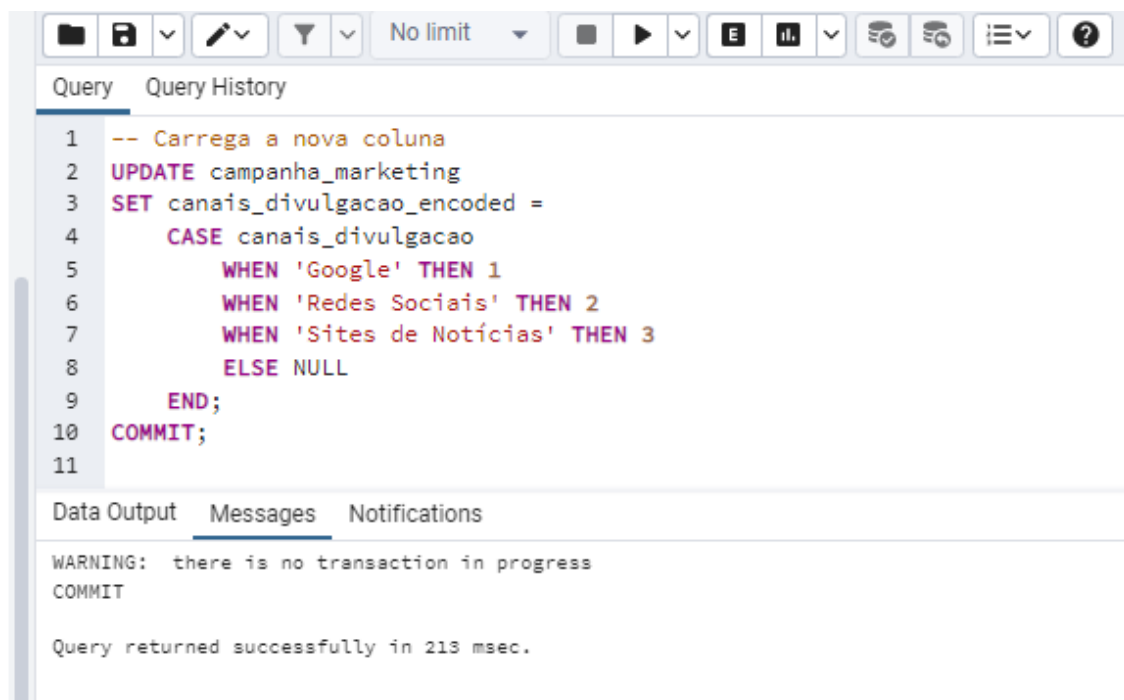
1  -- Verifica os valores únicos
2  SELECT canais_divulgacao, COUNT(*) as total_registros
3  FROM campanha_marketing
4  GROUP BY canais_divulgacao;
5
6

```

The Data Output tab shows the following results:

	canais_divulgacao character varying	total_registros bigint
1	Sites de Notícias	2025
2	Redes Sociais	3635
3	Google	2025

## Carrega a nova coluna



The screenshot shows a SQL IDE interface with a query editor and a messages panel. The query is as follows:

```

1  -- Carrega a nova coluna
2  UPDATE campanha_marketing
3  SET canais_divulgacao_encoded =
4      CASE canais_divulgacao
5          WHEN 'Google' THEN 1
6          WHEN 'Redes Sociais' THEN 2
7          WHEN 'Sites de Notícias' THEN 3
8          ELSE NULL
9      END;
10 COMMIT;
11

```

The Messages panel shows the following output:

```

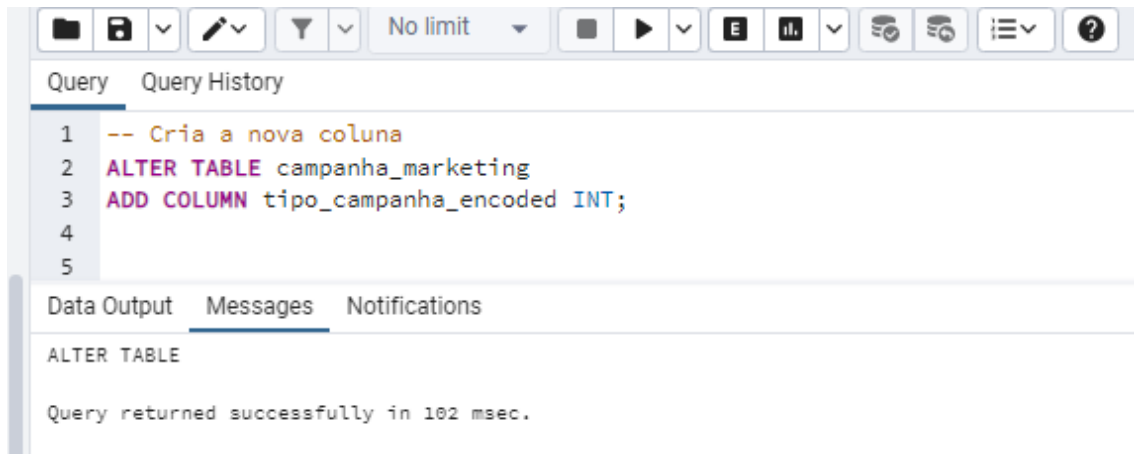
WARNING:  there is no transaction in progress
COMMIT

Query returned successfully in 213 msec.

```

Aplicando label encoding na coluna tipo\_campanha e salvando o resultado em uma nova coluna chamada tipo\_campanha\_encoded.

Cria a nova coluna



The screenshot shows a SQL IDE interface with a toolbar at the top. The 'Query' tab is active, displaying the following SQL code:

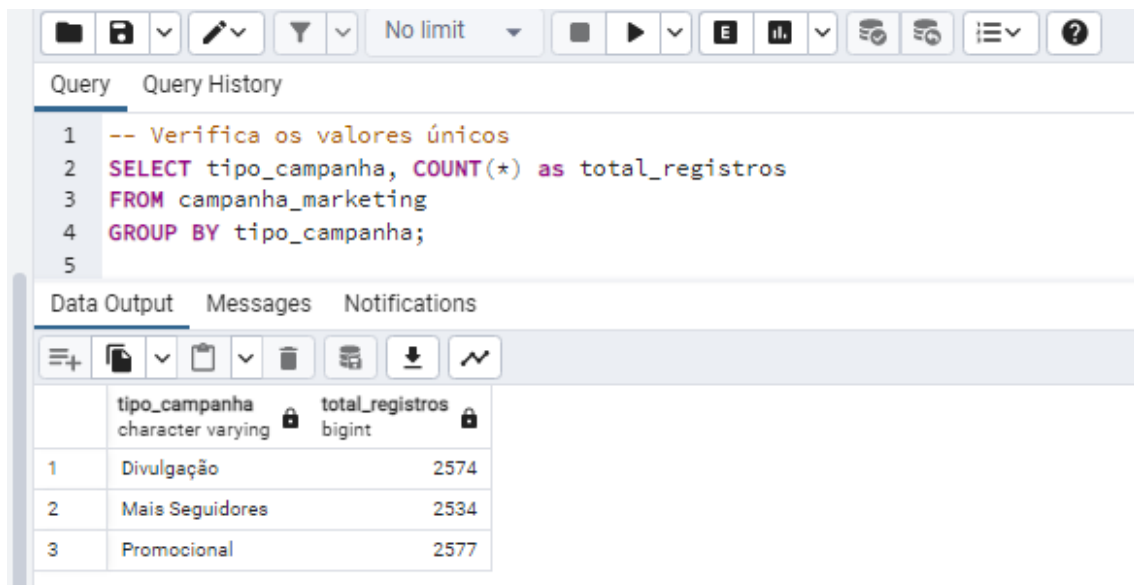
```

1  -- Cria a nova coluna
2  ALTER TABLE campanha_marketing
3  ADD COLUMN tipo_campanha_encoded INT;
4
5

```

Below the code editor, the 'Data Output' tab is selected, showing the message: 'ALTER TABLE' and 'Query returned successfully in 102 msec.'

Verifica os valores únicos



The screenshot shows a SQL IDE interface with a toolbar at the top. The 'Query' tab is active, displaying the following SQL code:

```

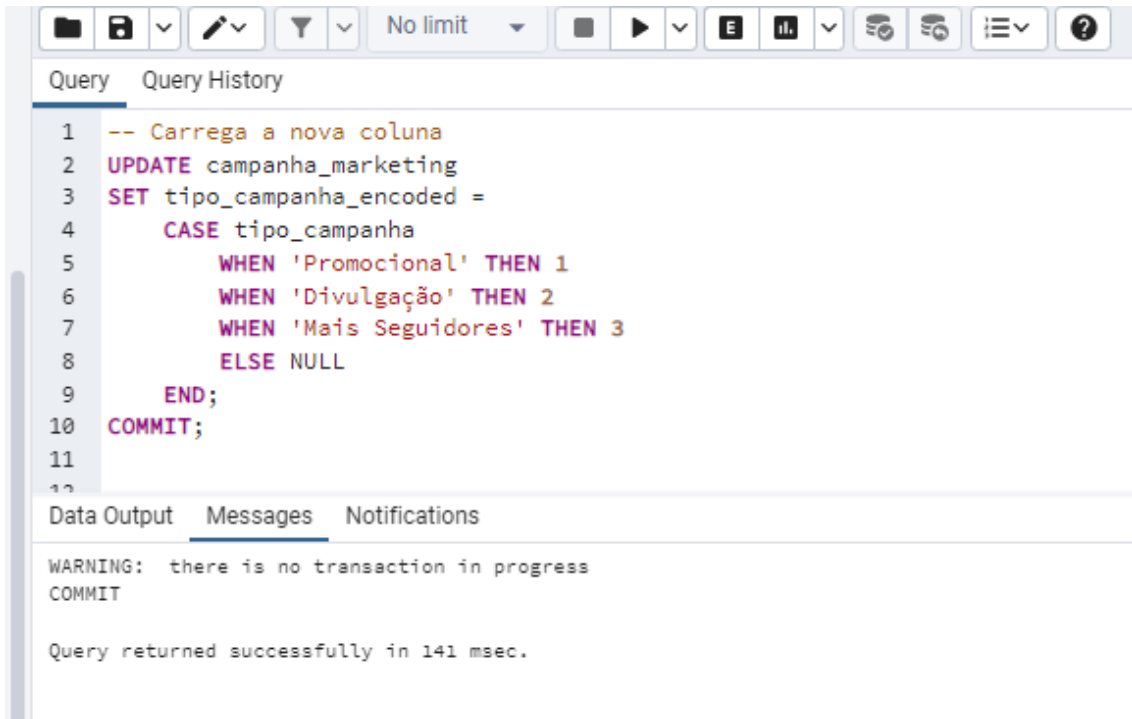
1  -- Verifica os valores únicos
2  SELECT tipo_campanha, COUNT(*) as total_registros
3  FROM campanha_marketing
4  GROUP BY tipo_campanha;
5

```

Below the code editor, the 'Data Output' tab is selected, showing a table with the following data:

	tipo_campanha character varying	total_registros bigint
1	Divulgação	2574
2	Mais Seguidores	2534
3	Promocional	2577

## Carrega a nova coluna



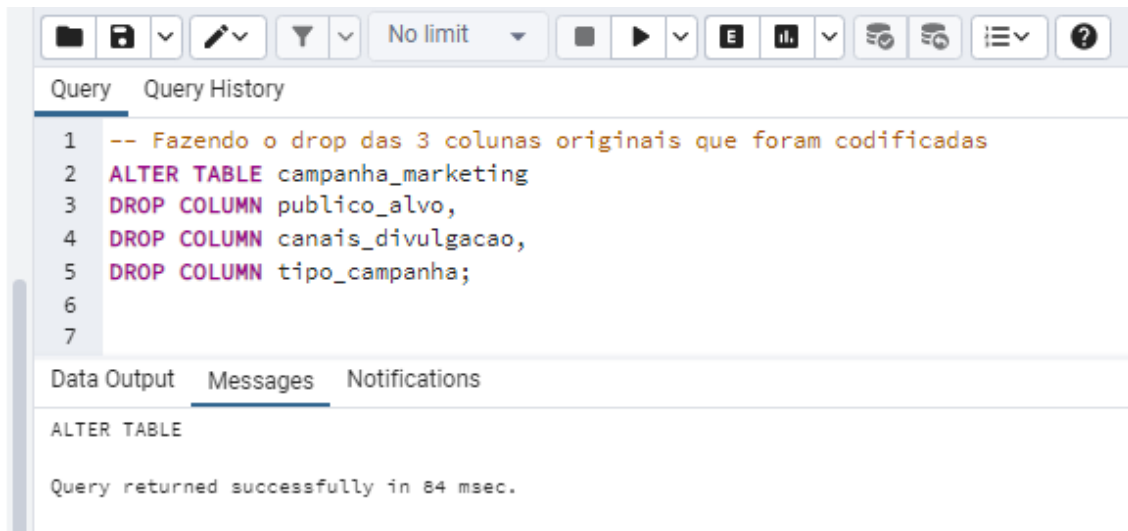
The screenshot shows a SQL IDE interface with a toolbar at the top containing icons for file operations, query execution, and settings. Below the toolbar, there are tabs for 'Query' and 'Query History'. The 'Query' tab is active, displaying a SQL script. The script is an UPDATE statement for a table named 'campanha\_marketing'. It sets a new column 'tipo\_campanha\_encoded' based on the values of the 'tipo\_campanha' column using a CASE statement. The values are: 'Promocional' maps to 1, 'Divulgação' maps to 2, 'Mais Seguidores' maps to 3, and any other value maps to NULL. The script ends with 'END;' and 'COMMIT;'. Below the query editor, there are tabs for 'Data Output', 'Messages', and 'Notifications'. The 'Messages' tab is active, showing a warning message: 'WARNING: there is no transaction in progress COMMIT'. Below the messages, it states 'Query returned successfully in 141 msec.'

```
1  -- Carrega a nova coluna
2  UPDATE campanha_marketing
3  SET tipo_campanha_encoded =
4      CASE tipo_campanha
5          WHEN 'Promocional' THEN 1
6          WHEN 'Divulgação' THEN 2
7          WHEN 'Mais Seguidores' THEN 3
8          ELSE NULL
9      END;
10 COMMIT;
```

WARNING: there is no transaction in progress  
COMMIT

Query returned successfully in 141 msec.

## Fazendo o drop das 3 colunas originais que foram codificadas



The screenshot shows the same SQL IDE interface as the previous one. The 'Query' tab is active, displaying a new SQL script. This script is an ALTER TABLE statement for the 'campanha\_marketing' table. It uses the 'DROP COLUMN' command to remove three columns: 'publico\_alvo', 'canais\_divulgacao', and 'tipo\_campanha'. The script consists of three lines for the DROP commands. Below the query editor, the 'Messages' tab is active, showing the message 'ALTER TABLE'. Below the messages, it states 'Query returned successfully in 64 msec.'

```
1  -- Fazendo o drop das 3 colunas originais que foram codificadas
2  ALTER TABLE campanha_marketing
3  DROP COLUMN publico_alvo,
4  DROP COLUMN canais_divulgacao,
5  DROP COLUMN tipo_campanha;
```

ALTER TABLE

Query returned successfully in 64 msec.

## Verifica os dados:

Query Query History

```
1 -- Verifica os dados:
2 SELECT * FROM campanha_marketing;
3
4
```

Data Output Messages Notifications

	id	nome_campanha	data_inicio	data_fim	orcamento	taxa_conversao	impressoes	tem_outlier	publico_alvo_encoded	canais_divulgacao_encoded	tipo_campanha_encoded
	integer	character varying	date	date	numeric (10,2)	numeric (5,2)	bigint	boolean	integer	integer	integer
1	100001	Campanha 1	2023-12-09	2024-01-02	65986.12	4.91	3000000	false	2	1	2
2	100008	Campanha 8	2022-09-10	2022-09-28	49157.97	5.94	3000000	false	4	1	3
3	100011	Campanha 11	2021-12-13	2021-12-31	11182.38	27.31	3000000	false	2	2	2
4	100013	Campanha 13	2021-10-15	2021-11-13	11025.74	27.52	9000000	false	4	1	1
5	100016	Campanha 16	2023-06-24	2023-06-28	33192.48	3.30	4000000	false	2	2	1
6	100009	Campanha 9	2024-02-23	2024-03-09	49817.09	22.06	2000000	false	5	3	2
7	100021	Campanha 21	2023-12-25	2023-12-27	73182.68	10.42	9000000	false	2	3	1
8	100048	Campanha 48	2024-02-10	2024-02-15	49817.09	15.36	4000000	false	4	3	1
9	100023	Campanha 23	2020-10-26	2020-11-16	35408.65	5.47	9000000	false	4	1	2
10	100024	Campanha 24	2021-12-21	2022-01-16	17662.54	11.08	7000000	false	5	1	1
11	100026	Campanha 26	2021-03-16	2021-04-03	46801.19	2.76	9000000	false	2	3	3
12	100029	Campanha 29	2023-01-10	2023-01-17	69336.11	20.41	5000000	false	5	3	1
13	100033	Campanha 33	2022-09-24	2022-10-15	22199.28	25.72	7000000	false	4	3	2
14	100035	Campanha 35	2020-08-05	2020-08-26	84502.44	15.46	7000000	false	1	1	3
15	100062	Campanha 62	2023-10-12	2023-11-02	49817.09	23.36	7000000	false	3	3	3
16	100038	Campanha 38	2021-07-30	2021-08-19	78488.73	4.58	7000000	false	4	1	2

## Verifica valores ausentes na tabela campanha\_marketing

Query Query History

```
1 -- Verifica valores ausentes na tabela campanha_marketing
2 SELECT
3   COUNT(*) - COUNT(id) AS id_missing,
4   COUNT(*) - COUNT(nome_campanha) AS nome_campanha_missing,
5   COUNT(*) - COUNT(data_inicio) AS data_inicio_missing,
6   COUNT(*) - COUNT(data_fim) AS data_fim_missing,
7   COUNT(*) - COUNT(orcamento) AS orcamento_missing,
8   COUNT(*) - COUNT(taxa_conversao) AS taxa_conversao_missing,
9   COUNT(*) - COUNT(impressoes) AS impressoes_missing,
10  COUNT(*) - COUNT(publico_alvo_encoded) AS publico_alvo_encoded_missing,
11  COUNT(*) - COUNT(canais_divulgacao_encoded) AS canais_divulgacao_encoded_missing,
12  COUNT(*) - COUNT(tipo_campanha_encoded) AS tipo_campanha_encoded_missing
13 FROM campanha_marketing;
14
```

Data Output Messages Notifications

	id_missing	nome_campanha_missing	data_inicio_missing	data_fim_missing	orcamento_missing	taxa_conversao_missing	impressoes_missing	publico_alvo_encoded_missing	canais_divulgacao_encoded_missing	tipo_campanha_encoded_missing
	bigint	bigint	bigint	bigint	bigint	bigint	bigint	bigint	bigint	bigint
1	0	0	0	0	0	0	0	0	0	0

## 9.8 Relatório de Resumo com Variáveis Quantitativas

Relatório de Resumo Com Variáveis Quantitativas. Totais dos anos 2022, 2023 e 2024 para as colunas orcamento, taxa\_conversao e impressões.

Query Query History

```

2 -- Totais dos anos 2022, 2023 e 2024 para as colunas orcamento, taxa_conversao e impressoes
3 SELECT
4     TO_CHAR(data_inicio, 'YYYY') AS ano,
5     SUM(orcamento) AS total_orcamento,
6     SUM(taxa_conversao) AS total_taxa_conversao,
7     SUM(impressoes) AS total_impressoes
8 FROM campanha_marketing
9 WHERE EXTRACT(YEAR FROM data_inicio) IN (2022, 2023, 2024)
10 GROUP BY TO_CHAR(data_inicio, 'YYYY')
11 ORDER BY TO_CHAR(data_inicio, 'YYYY') DESC;
12
13

```

Data Output Messages Notifications

	ano text	total_orcamento numeric	total_taxa_conversao numeric	total_impressoes numeric
1	2024	48421750.79	14414.66	5322000000
2	2023	92328316.28	28338.56	10571000000
3	2022	97828346.15	29391.44	10731000000

## 9.9 Relatório de Resumo com Variáveis Quantitativas e Pivot da Tabela

Relatório de Resumo Com Variáveis Quantitativas e Pivot da Tabela.

Query Query History

```

2 -- Relatório de Resumo Com Variáveis Quantitativas e Pivot da Tabela
3 SELECT
4     'Total' as Total,
5     SUM(CASE WHEN EXTRACT(YEAR FROM data_inicio) = 2022 THEN orcamento ELSE 0 END) AS "Orcamento_2022",
6     SUM(CASE WHEN EXTRACT(YEAR FROM data_inicio) = 2022 THEN taxa_conversao ELSE 0 END) AS "Taxa_Conversao_2022",
7     SUM(CASE WHEN EXTRACT(YEAR FROM data_inicio) = 2022 THEN impressoes ELSE 0 END) AS "Impressoes_2022",
8     SUM(CASE WHEN EXTRACT(YEAR FROM data_inicio) = 2023 THEN orcamento ELSE 0 END) AS "Orcamento_2023",
9     SUM(CASE WHEN EXTRACT(YEAR FROM data_inicio) = 2023 THEN taxa_conversao ELSE 0 END) AS "Taxa_Conversao_2023",
10    SUM(CASE WHEN EXTRACT(YEAR FROM data_inicio) = 2023 THEN impressoes ELSE 0 END) AS "Impressoes_2023",
11    SUM(CASE WHEN EXTRACT(YEAR FROM data_inicio) = 2024 THEN orcamento ELSE 0 END) AS "Orcamento_2024",
12    SUM(CASE WHEN EXTRACT(YEAR FROM data_inicio) = 2024 THEN taxa_conversao ELSE 0 END) AS "Taxa_Conversao_2024",
13    SUM(CASE WHEN EXTRACT(YEAR FROM data_inicio) = 2024 THEN impressoes ELSE 0 END) AS "Impressoes_2024"
14 FROM campanha_marketing;
15
16

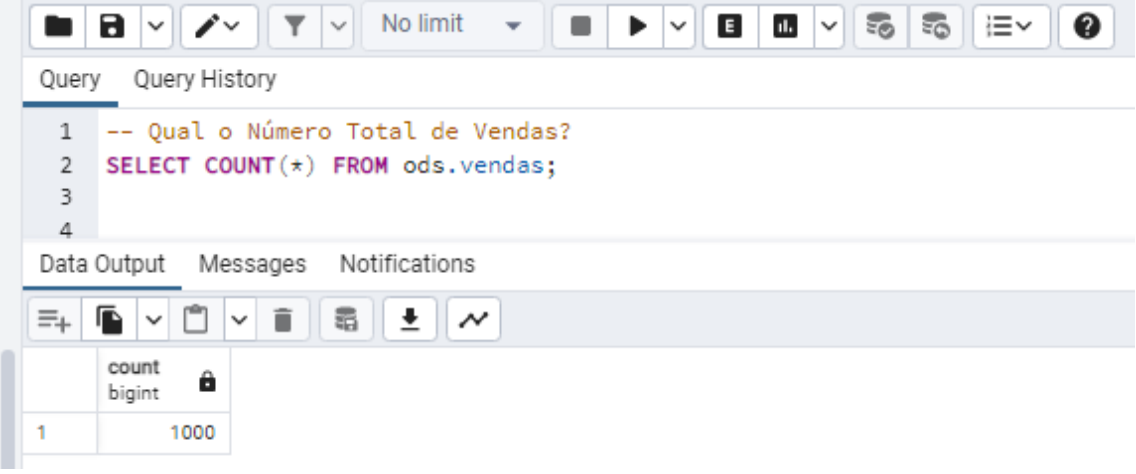
```

Data Output Messages Notifications

	total text	Orcamento_2022 numeric	Taxa_Conversao_2022 numeric	Impressoes_2022 numeric	Orcamento_2023 numeric	Taxa_Conversao_2023 numeric	Impressoes_2023 numeric	Orcamento_2024 numeric	Taxa_Conversao_2024 numeric	Impressoes_2024 numeric
1	Total	97828346.15	29391.44	10731000000	92328316.28	28338.56	10571000000	48421750.79	14414.66	5322000000

## 10 Análise de Dados com SQL

Qual o Número Total de Vendas?



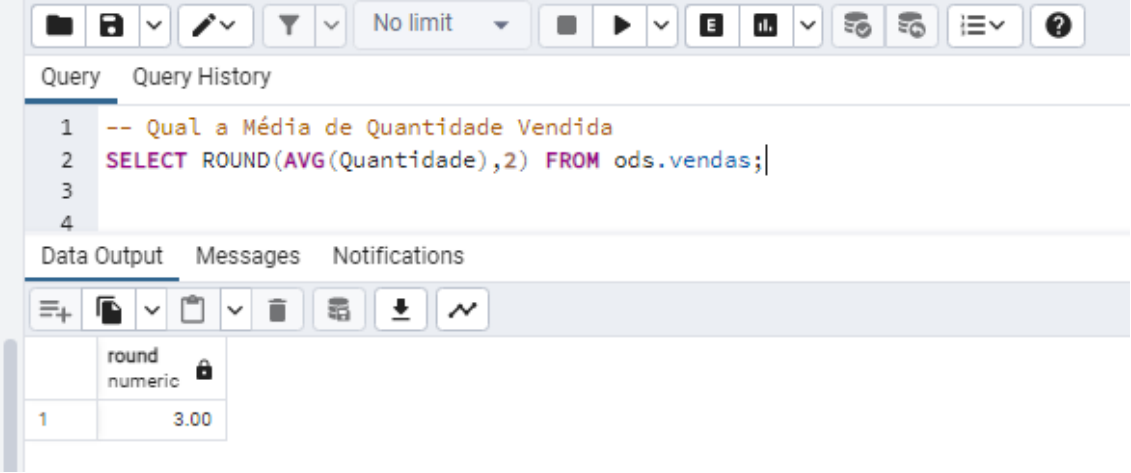
The screenshot shows a SQL query editor interface. The top toolbar includes icons for file operations, query execution, and settings. The 'Query' tab is active, displaying the following SQL code:

```
1 -- Qual o Número Total de Vendas?  
2 SELECT COUNT(*) FROM ods.vendas;  
3  
4
```

Below the query editor, the 'Data Output' tab is active, showing the results of the query in a table:

	count	bigint
1	1000	

Qual a Média de Quantidade Vendida?



The screenshot shows a SQL query editor interface. The top toolbar includes icons for file operations, query execution, and settings. The 'Query' tab is active, displaying the following SQL code:

```
1 -- Qual a Média de Quantidade Vendida  
2 SELECT ROUND(AVG(Quantidade),2) FROM ods.vendas;  
3  
4
```

Below the query editor, the 'Data Output' tab is active, showing the results of the query in a table:

	round	numeric
1	3.00	

## Qual o Número Total de Produtos Únicos Vendidos?

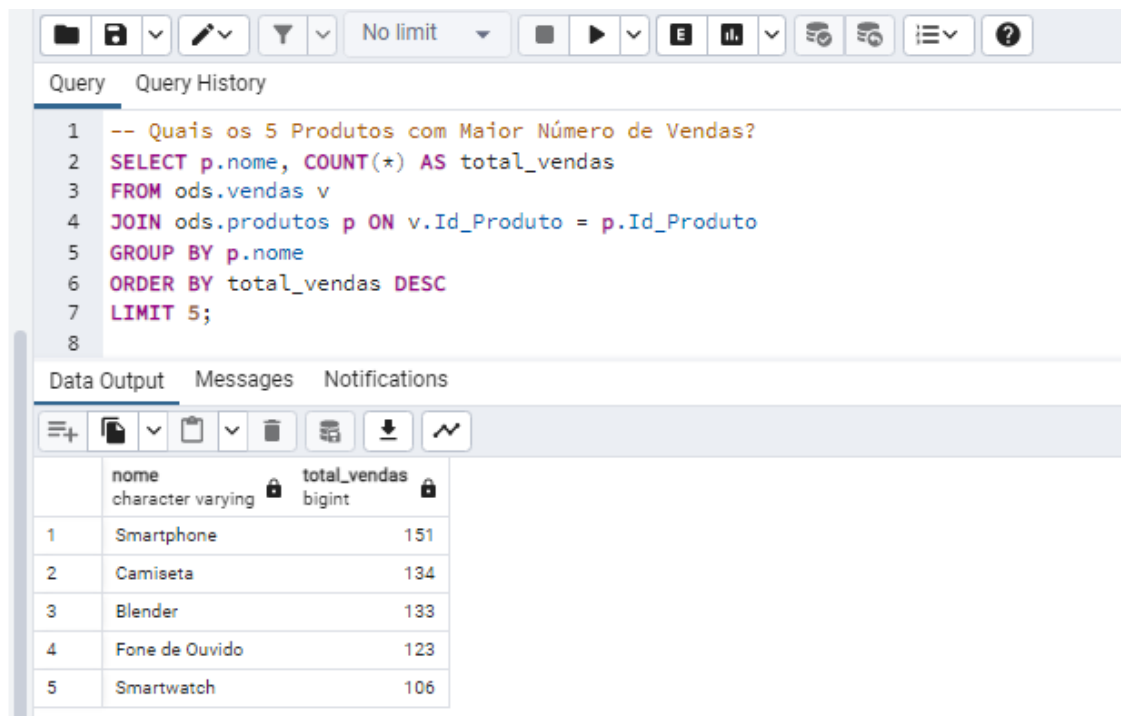
Query	Query History
1	-- Qual o Número Total de Produtos Únicos Vendidos?
2	SELECT COUNT(DISTINCT Id_Produto) FROM ods.vendas;
3	
4	
Data Output	Messages Notifications
count	bigint
1	273

## Quantas Vendas Ocorreram Por Produto? Mostre o Resultado em Ordem Decrescente.

Query	Query History
1	-- Quantas Vendas Ocorreram Por Produto? Mostre o Resultado em Ordem Decrescente.
2	SELECT p.nome, COUNT(v.Id_Produto) AS total_num_vendas
3	FROM ods.vendas v
4	INNER JOIN ods.produtos p ON v.Id_Produto = p.Id_Produto
5	GROUP BY p.nome
6	ORDER BY total_num_vendas DESC;
7	
Data Output	Messages Notifications
nome	total_num_vendas
character varying	bigint
1	Smartphone 151
2	Camiseta 134
3	Blender 133
4	Fone de Ouvido 123
5	Smartwatch 106
6	Perfume 105
7	Cafeteira 93
8	Mochila 88
9	Notebook 67



## Quais os 5 Produtos com Maior Número de Vendas?



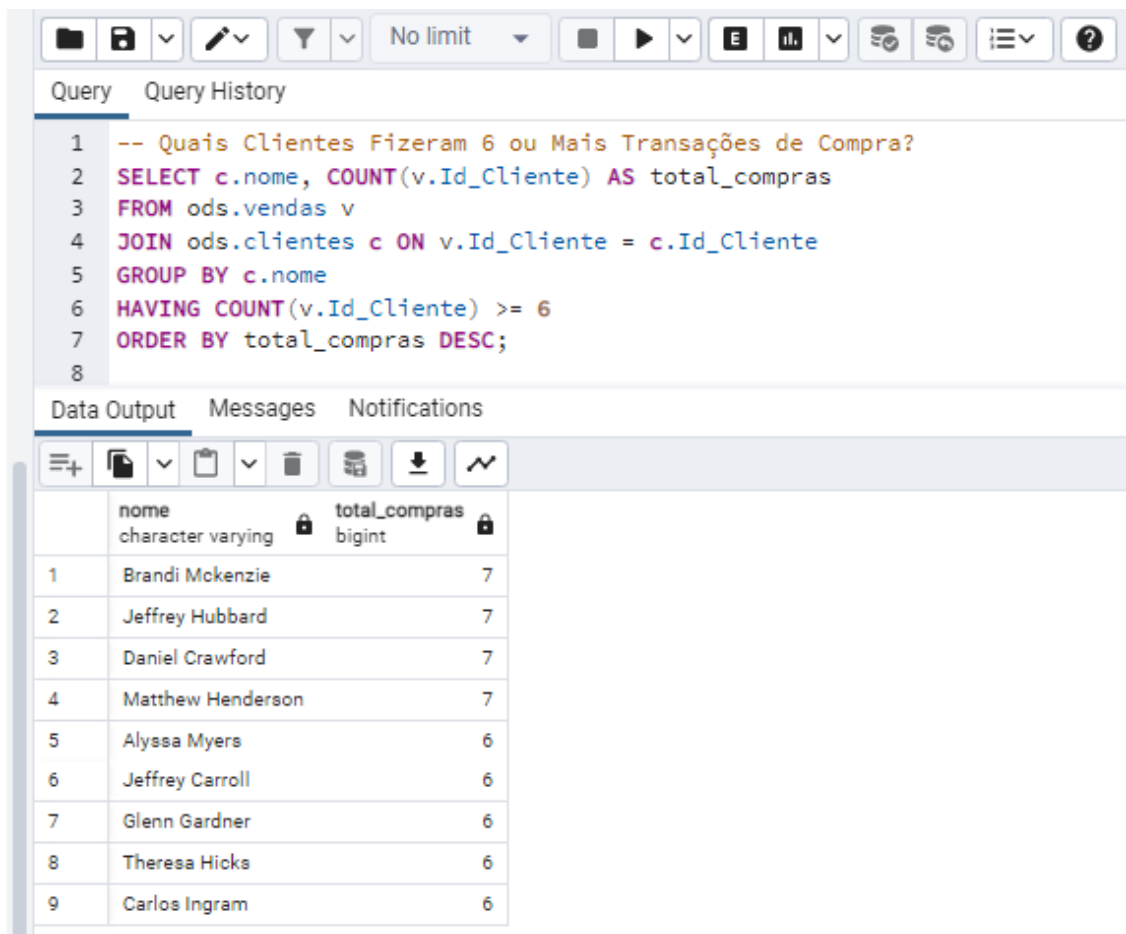
The screenshot shows a SQL query editor interface. The query is as follows:

```
1 -- Quais os 5 Produtos com Maior Número de Vendas?
2 SELECT p.nome, COUNT(*) AS total_vendas
3 FROM ods.vendas v
4 JOIN ods.produtos p ON v.Id_Produto = p.Id_Produto
5 GROUP BY p.nome
6 ORDER BY total_vendas DESC
7 LIMIT 5;
8
```

Below the query editor, the 'Data Output' tab is active, displaying the results of the query in a table format:

	nome character varying	total_vendas bigint
1	Smartphone	151
2	Camiseta	134
3	Blender	133
4	Fone de Ouvido	123
5	Smartwatch	106

## Quais Clientes Fizeram 6 ou Mais Transações de Compra?



The screenshot shows a SQL query editor interface. The query is as follows:

```

1  -- Quais Clientes Fizeram 6 ou Mais Transações de Compra?
2  SELECT c.nome, COUNT(v.Id_Cliente) AS total_compras
3  FROM ods.vendas v
4  JOIN ods.clientes c ON v.Id_Cliente = c.Id_Cliente
5  GROUP BY c.nome
6  HAVING COUNT(v.Id_Cliente) >= 6
7  ORDER BY total_compras DESC;
8

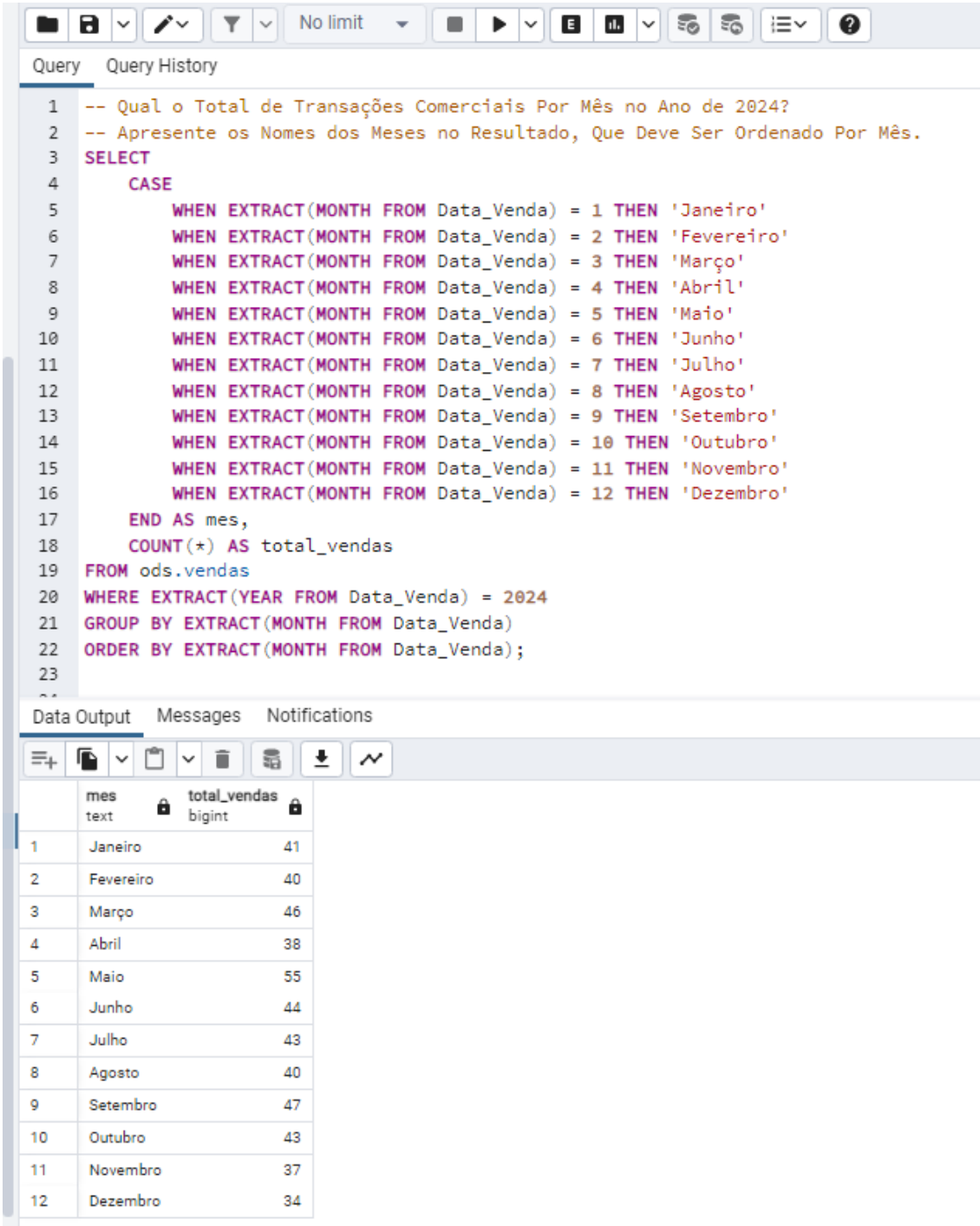
```

Below the query editor, the 'Data Output' tab is active, showing the results of the query in a table. The table has two columns: 'nome' (character varying) and 'total\_compras' (bigint). The results are as follows:

	nome	total_compras
1	Brandi Mckenzie	7
2	Jeffrey Hubbard	7
3	Daniel Crawford	7
4	Matthew Henderson	7
5	Alyssa Myers	6
6	Jeffrey Carroll	6
7	Glenn Gardner	6
8	Theresa Hicks	6
9	Carlos Ingram	6

Qual o Total de Transações Comerciais Por Mês no Ano de 2024?

Apresente os Nomes dos Meses no Resultado, Que Deve Ser Ordenado Por Mês.



The screenshot shows a SQL IDE interface. The top toolbar includes icons for file operations, query execution, and settings. The main editor displays a SQL query with line numbers 1 through 23. The query is as follows:

```

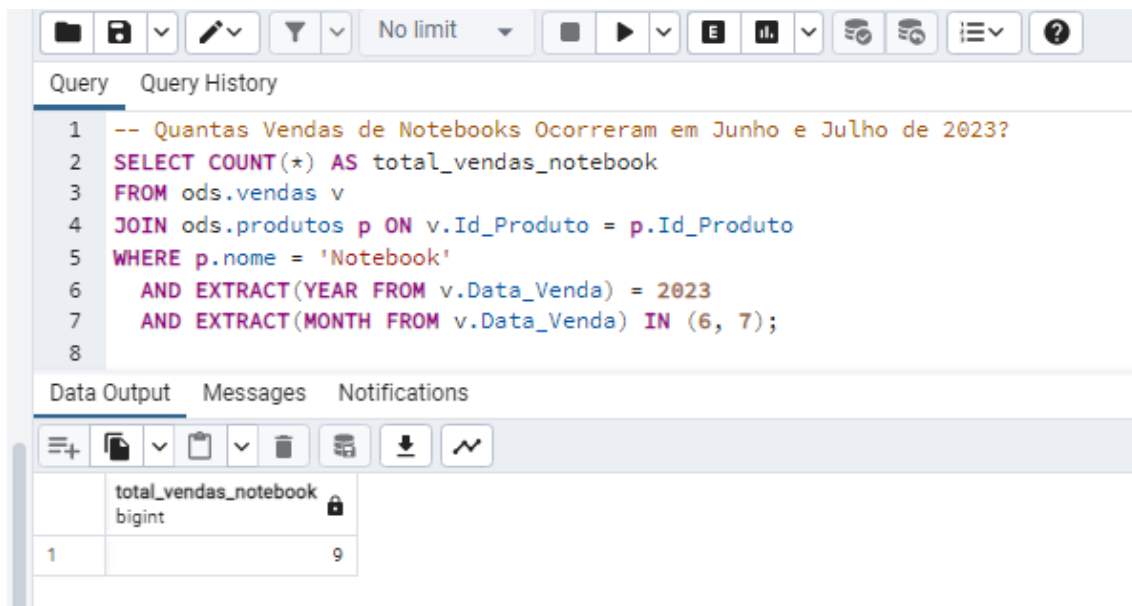
1  -- Qual o Total de Transações Comerciais Por Mês no Ano de 2024?
2  -- Apresente os Nomes dos Meses no Resultado, Que Deve Ser Ordenado Por Mês.
3  SELECT
4      CASE
5          WHEN EXTRACT(MONTH FROM Data_Venda) = 1 THEN 'Janeiro'
6          WHEN EXTRACT(MONTH FROM Data_Venda) = 2 THEN 'Fevereiro'
7          WHEN EXTRACT(MONTH FROM Data_Venda) = 3 THEN 'Março'
8          WHEN EXTRACT(MONTH FROM Data_Venda) = 4 THEN 'Abril'
9          WHEN EXTRACT(MONTH FROM Data_Venda) = 5 THEN 'Maio'
10         WHEN EXTRACT(MONTH FROM Data_Venda) = 6 THEN 'Junho'
11         WHEN EXTRACT(MONTH FROM Data_Venda) = 7 THEN 'Julho'
12         WHEN EXTRACT(MONTH FROM Data_Venda) = 8 THEN 'Agosto'
13         WHEN EXTRACT(MONTH FROM Data_Venda) = 9 THEN 'Setembro'
14         WHEN EXTRACT(MONTH FROM Data_Venda) = 10 THEN 'Outubro'
15         WHEN EXTRACT(MONTH FROM Data_Venda) = 11 THEN 'Novembro'
16         WHEN EXTRACT(MONTH FROM Data_Venda) = 12 THEN 'Dezembro'
17     END AS mes,
18     COUNT(*) AS total_vendas
19 FROM ods.vendas
20 WHERE EXTRACT(YEAR FROM Data_Venda) = 2024
21 GROUP BY EXTRACT(MONTH FROM Data_Venda)
22 ORDER BY EXTRACT(MONTH FROM Data_Venda);
23

```

Below the query editor, the 'Data Output' tab is active, showing a table with two columns: 'mes' (text) and 'total\_vendas' (bigint). The table contains 12 rows of data, representing the months of the year 2024.

	mes	total_vendas
1	Janeiro	41
2	Fevereiro	40
3	Março	46
4	Abril	38
5	Maio	55
6	Junho	44
7	Julho	43
8	Agosto	40
9	Setembro	47
10	Outubro	43
11	Novembro	37
12	Dezembro	34

## Quantas Vendas de Notebooks Ocorreram em Junho e Julho de 2023?



The screenshot displays a SQL query editor interface. The top toolbar includes icons for file operations, query execution, and settings. The query editor shows the following SQL code:

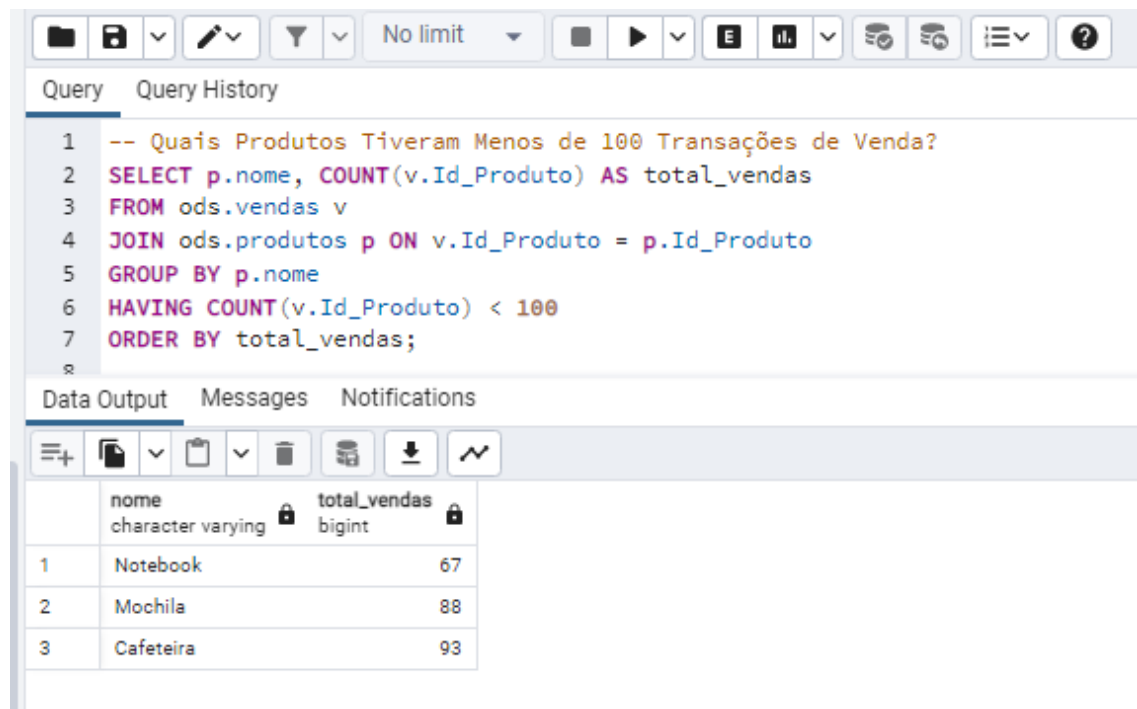
```
1 -- Quantas Vendas de Notebooks Ocorreram em Junho e Julho de 2023?
2 SELECT COUNT(*) AS total_vendas_notebook
3 FROM ods.vendas v
4 JOIN ods.produtos p ON v.Id_Produto = p.Id_Produto
5 WHERE p.nome = 'Notebook'
6       AND EXTRACT(YEAR FROM v.Data_Venda) = 2023
7       AND EXTRACT(MONTH FROM v.Data_Venda) IN (6, 7);
8
```

Below the query editor, the 'Data Output' tab is active, showing a table with the results of the query:

	total_vendas_notebook
1	9



## Quais Produtos Tiveram Menos de 100 Transações de Venda?



The screenshot shows a SQL query editor interface. The query is as follows:

```

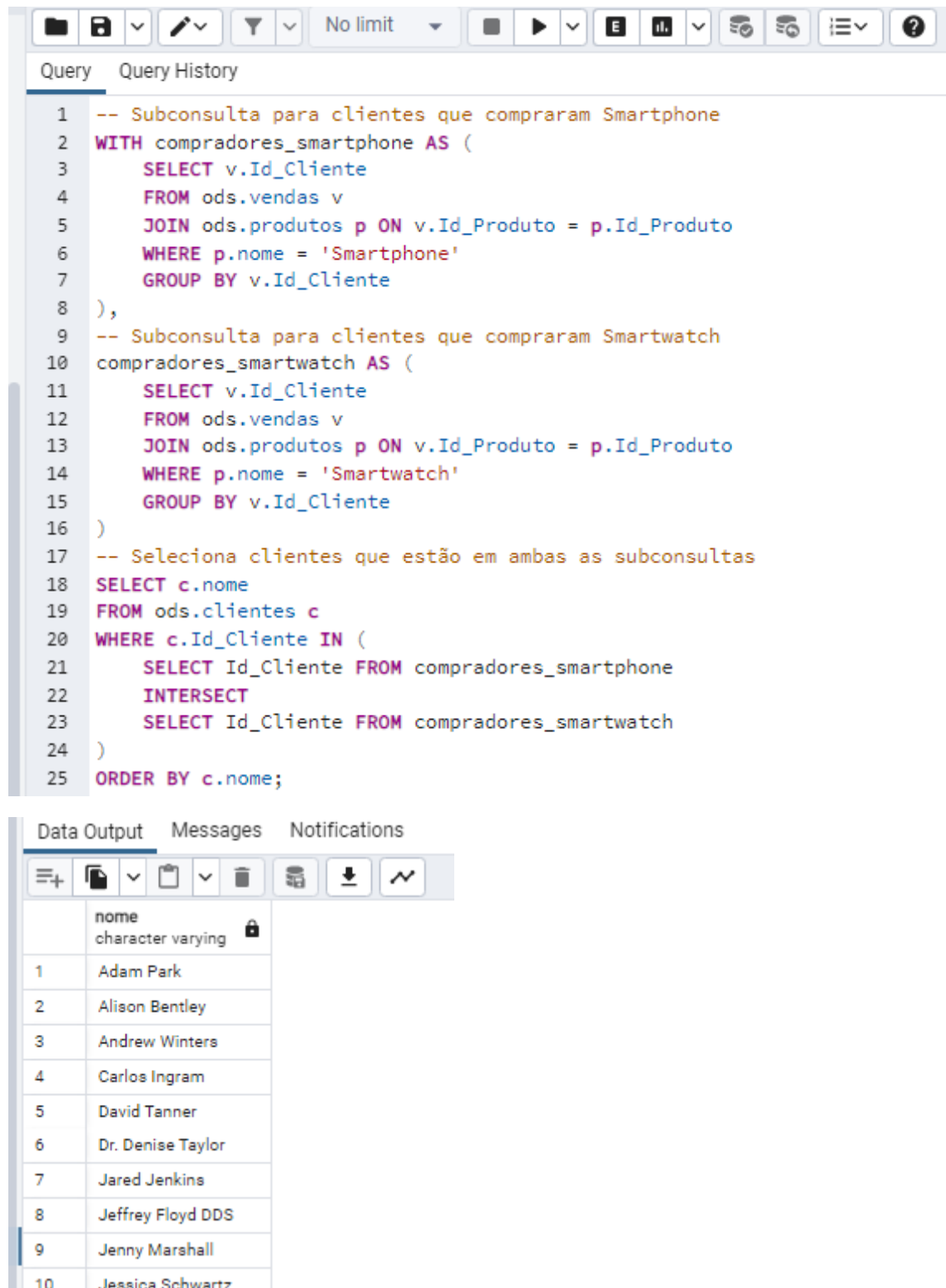
1  -- Quais Produtos Tiveram Menos de 100 Transações de Venda?
2  SELECT p.nome, COUNT(v.Id_Produto) AS total_vendas
3  FROM ods.vendas v
4  JOIN ods.produtos p ON v.Id_Produto = p.Id_Produto
5  GROUP BY p.nome
6  HAVING COUNT(v.Id_Produto) < 100
7  ORDER BY total_vendas;

```

The results are displayed in a table with the following data:

	nome	total_vendas
	character varying	bigint
1	Notebook	67
2	Mochila	88
3	Cafeteira	93

## Quais Clientes Compraram Smartphone e Também Compraram Smartwatch?



The screenshot shows a SQL query editor with a toolbar at the top containing icons for file operations, query execution, and settings. The query is written in SQL and aims to find clients who have purchased both a smartphone and a smartwatch. It uses two subqueries, `compradores_smartphone` and `compradores_smartwatch`, which select client IDs from the `ods.vendas` table joined with `ods.produtos` where the product name is 'Smartphone' and 'Smartwatch' respectively. These subqueries are then used in an `INTERSECT` clause to find common clients, which are then joined with `ods.clientes` to retrieve their names. The results are ordered by the client's name.

```
1  -- Subconsulta para clientes que compraram Smartphone
2  WITH compradores_smartphone AS (
3      SELECT v.Id_Cliente
4      FROM ods.vendas v
5      JOIN ods.produtos p ON v.Id_Produto = p.Id_Produto
6      WHERE p.nome = 'Smartphone'
7      GROUP BY v.Id_Cliente
8  ),
9  -- Subconsulta para clientes que compraram Smartwatch
10 compradores_smartwatch AS (
11     SELECT v.Id_Cliente
12     FROM ods.vendas v
13     JOIN ods.produtos p ON v.Id_Produto = p.Id_Produto
14     WHERE p.nome = 'Smartwatch'
15     GROUP BY v.Id_Cliente
16 )
17 -- Seleciona clientes que estão em ambas as subconsultas
18 SELECT c.nome
19 FROM ods.clientes c
20 WHERE c.Id_Cliente IN (
21     SELECT Id_Cliente FROM compradores_smartphone
22     INTERSECT
23     SELECT Id_Cliente FROM compradores_smartwatch
24 )
25 ORDER BY c.nome;
```

The results are displayed in a table with the following data:

	nome
1	Adam Park
2	Alison Bentley
3	Andrew Winters
4	Carlos Ingram
5	David Tanner
6	Dr. Denise Taylor
7	Jared Jenkins
8	Jeffrey Floyd DDS
9	Jenny Marshall
10	Jessica Schwartz

## Quais Clientes Compraram Smartphone e Também Compraram Smartwatch, Mas Não Compraram Notebook?

```

1  -- Clientes que compraram Smartphone
2  WITH clientes_smartphone AS (
3      SELECT Id_Cliente
4      FROM ods.vendas
5      JOIN ods.produtos ON vendas.Id_Produto = produtos.Id_Produto
6      WHERE produtos.nome = 'Smartphone'
7  ),
8  -- Clientes que compraram Smartwatch
9  clientes_smartwatch AS (
10     SELECT Id_Cliente
11     FROM ods.vendas
12     JOIN ods.produtos ON vendas.Id_Produto = produtos.Id_Produto
13     WHERE produtos.nome = 'Smartwatch'
14 ),
15 -- Clientes que compraram Notebook
16 clientes_notebook AS (
17     SELECT Id_Cliente
18     FROM ods.vendas
19     JOIN ods.produtos ON vendas.Id_Produto = produtos.Id_Produto
20     WHERE produtos.nome = 'Notebook'
21 )
22 -- Clientes que compraram Smartphone e Smartwatch, mas não compraram Notebook
23 SELECT clientes.nome
24 FROM ods.clientes
25 WHERE Id_Cliente IN (
26     SELECT Id_Cliente FROM clientes_smartphone
27     INTERSECT
28     SELECT Id_Cliente FROM clientes_smartwatch
29 )
30 AND Id_Cliente NOT IN (
31     SELECT Id_Cliente FROM clientes_notebook
32 );
33

```

Data Output		Messages	Notifications
	nome		
	character varying		
1	Adam Park		
2	Alison Bentley		
3	Andrew Winters		
4	Carlos Ingram		
5	David Tanner		
6	Dr. Denise Taylor		
7	Jared Jenkins		
8	Jeffrey Floyd DDS		
9	Jenny Marshall		



Qual a Média Móvel de Quantidade de Unidades Vendidas ao Longo do Tempo? Considere Janela de 7 Dias.

Qual a Média Móvel e Desvio Padrão Móvel de Quantidade de Unidades Vendidas ao Longo do Tempo? Considere Janela de 7 Dias.

## Quais Clientes Estão Cadastrados, Mas Ainda Não Fizeram Transação?

No limit

Query

Query History

```

13
14 -- Quais Clientes Estão Cadastrados, Mas Ainda Não Fizeram Transação?
15 SELECT c.Id_Cliente, c.nome
16 FROM ods.clientes c
17 LEFT JOIN ods.vendas v ON c.Id_Cliente = v.Id_Cliente
18 WHERE v.Id_Cliente IS NULL;
19
20
21
22
23
24

```

Data Output

Messages

Notifications

	id_cliente [PK] uuid	nome character varying
1	f2ccc3ff-9b2a-4403-ac21-9a5dc2057746	Mark Duran
2	8da2e866-7f7c-4694-bf0f-92beec87706c	Kyle Bridges
3	cb1db07a-8bc5-42ff-a4e6-be4f1f6b19c7	Steven Reid
4	c1145492-4238-430f-99b8-2e407121e17e	Cameron Vega
5	df3961d1-c67e-43bd-9d0c-f75eebde84be	Cody Ford
6	368542a1-6233-4b11-af45-09588cf24ebb	Angel Lewis
7	e57ce883-0f24-439e-ad7c-5cf1e001edbd	Danielle Johnson
8	2ca787ff-4f9b-41cc-83db-0587b597dab2	Brenda Marshall
9	bc1fd93c-0c6e-46e6-b700-c030a8a0ea3e	Beth Martin
10	3b788586-3043-4866-a115-6790a3ecb2...	David Jacobson
11	85c57a3b-006b-4796-9f79-fcd58df8ecee	Joshua Franco
12	ff242969-2cc6-4f8f-8172-55a235ce4815	Eric Norris

## 11 Conclusão

Em resumo, o SQL continua sendo uma ferramenta indispensável para análise de dados, capacitando profissionais a extrair, manipular e interpretar informações de maneira eficaz. Ao longo deste artigo, exploramos técnicas avançadas de SQL, demonstrando como suas capacidades podem ser aplicadas na prática para beneficiar decisões estratégicas e operacionais.

Dominar SQL não é apenas uma habilidade técnica, mas uma vantagem competitiva significativa em um mercado cada vez mais orientado por dados. À medida que avançamos para o futuro da análise de dados, novas tecnologias e abordagens continuarão a moldar o cenário. É essencial manter-se atualizado e adaptar-se às mudanças para aproveitar ao máximo o potencial do SQL e das ferramentas relacionadas.

Portanto, convido a todos a explorar mais profundamente as possibilidades oferecidas pelo SQL e a aplicar esses conhecimentos em suas próprias jornadas de análise de dados.

## 12 Referências

**TANIMURA, Cathy.** *SQL for Data Analysis: Advanced Techniques for Transforming Data Into Insights*. Sebastopol, CA, United States: O'Reilly Media, 2021.