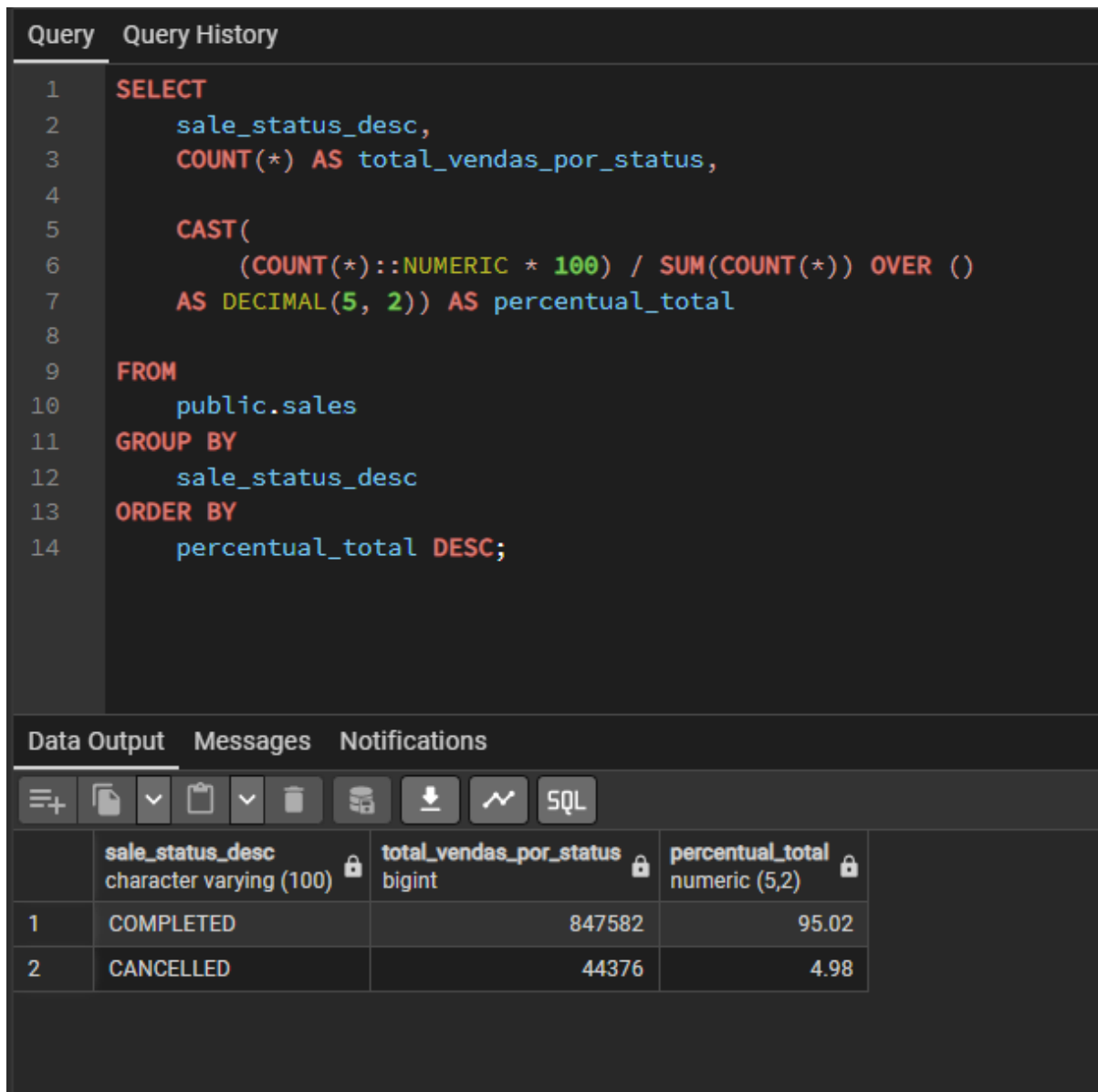


Análise inicial do conjunto de dados gerado pelo script.



The screenshot shows a SQL IDE interface. The top panel, titled 'Query', contains the following SQL query:

```
1 SELECT
2     sale_status_desc,
3     COUNT(*) AS total_vendas_por_status,
4
5     CAST(
6         (COUNT(*)::NUMERIC * 100) / SUM(COUNT(*)) OVER ()
7         AS DECIMAL(5, 2)) AS percentual_total
8
9 FROM
10    public.sales
11 GROUP BY
12     sale_status_desc
13 ORDER BY
14     percentual_total DESC;
```

The bottom panel, titled 'Data Output', shows the results of the query in a table. The table has three columns: 'sale_status_desc', 'total_vendas_por_status', and 'percentual_total'. The results are as follows:

	sale_status_desc character varying (100)	total_vendas_por_status bigint	percentual_total numeric (5,2)
1	COMPLETED	847582	95.02
2	CANCELLED	44376	4.98

Aproximadamente 5% dos pedidos são cancelados

Análise inicial do conjunto de dados gerado pelo script.

Query

Query History

1

SELECT

2

c.name AS canal,

3

COUNT(s.id) AS total_vendas,

4

SUM(s.total_amount) AS receita_total,

5

-- Calcula o Ticket Médio por Canal

6

ROUND(AVG(s.total_amount), 2) AS ticket_medio

7

FROM

8

sales s

9

JOIN

10

channels c ON s.channel_id = c.id

11

GROUP BY

12

c.name

13

ORDER BY

14

receita_total DESC;

Data Output

Messages

Notifications

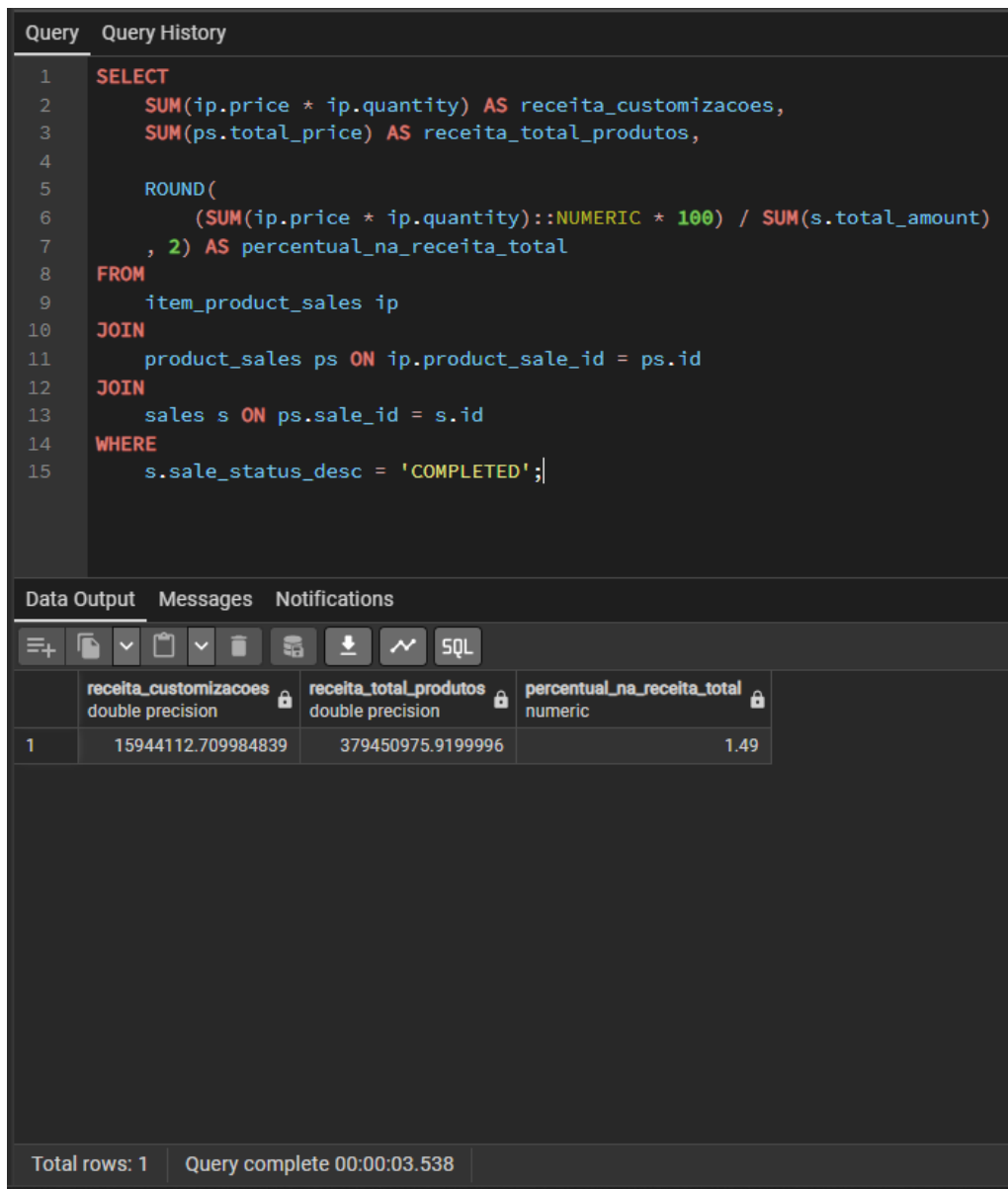
≡+

SQL

	canal character varying (100) 🔒	total_vendas bigint 🔒	receita_total numeric 🔒	ticket_medio numeric 🔒
1	Presencial	404198	143033931.13	353.87
2	iFood	303362	110328098.38	363.68
3	Rappi	152789	55483314.26	363.14
4	Uber Eats	80867	29356728.96	363.02
5	WhatsApp	51024	18474795.60	362.08
6	App Próprio	20281	7371972.56	363.49

Ticket Médio é aproximadamente igual para os diferentes canais de compra.

Análise inicial do conjunto de dados gerado pelo script.



The screenshot shows a SQL query editor with a query that calculates the percentage of revenue from customizations. The query is as follows:

```
1 SELECT
2     SUM(ip.price * ip.quantity) AS receita_customizacoes,
3     SUM(ps.total_price) AS receita_total_produtos,
4
5     ROUND(
6         (SUM(ip.price * ip.quantity)::NUMERIC * 100) / SUM(s.total_amount)
7         , 2) AS percentual_na_receita_total
8 FROM
9     item_product_sales ip
10 JOIN
11     product_sales ps ON ip.product_sale_id = ps.id
12 JOIN
13     sales s ON ps.sale_id = s.id
14 WHERE
15     s.sale_status_desc = 'COMPLETED';
```

The query is executed, and the results are displayed in a table with one row. The table has three columns: `receita_customizacoes` (double precision), `receita_total_produtos` (double precision), and `percentual_na_receita_total` (numeric). The values are 15944112.709984839, 379450975.9199996, and 1.49 respectively.

	receita_customizacoes double precision	receita_total_produtos double precision	percentual_na_receita_total numeric
1	15944112.709984839	379450975.9199996	1.49

Total rows: 1 Query complete 00:00:03.538

Porcentagem da receita vinda das customizações.

IMPORTANTE: Quase 4 segundos para essa query, indicação de gargalo.

Sugestões para melhorar :

Implementacao dos índices no script sql para as colunas mais utilizadas em filtros e joins

Análise inicial do conjunto de dados gerado pelo script.

Avaliar uso de materialized view futuramente

Query	Query History
1	-- Análise de Vendas Detalhada: Vendas e Ticket Médio por Loja, Mês e Categoria
2	SELECT
3	DATE_TRUNC('month', s.created_at) AS mes_venda,
4	st.name AS nome_loja,
5	c.name AS nome_categoria,
6	COUNT(s.id) AS total_vendas,
7	SUM(s.total_amount) AS receita_total,
8	ROUND(AVG(s.total_amount), 2) AS ticket_medio_categoria
9	FROM
10	public.sales s
11	JOIN
12	public.stores st ON s.store_id = st.id
13	JOIN
14	public.product_sales ps ON s.id = ps.sale_id
15	JOIN
16	public.products p ON ps.product_id = p.id
17	JOIN
18	public.categories c ON p.category_id = c.id
19	WHERE
20	s.sale_status_desc = 'COMPLETED'
21	GROUP BY
22	1, 2, 3
23	ORDER BY
24	mes_venda DESC, receita_total DESC;

Data Output	Messages	Notifications
Showing rows		
	mes_venda timestamp without time zone	nome_loja character varying (255)
		nome_categoria character varying (200)
		total_vendas bigint
		receita_total numeric
		ticket_medio_categoria numeric
1	2025-10-01 00:00:00	Porto - Dias de da Mota
2	2025-10-01 00:00:00	Porto - Dias de da Mota
3	2025-10-01 00:00:00	Cardoso - Araújo
4	2025-10-01 00:00:00	Cardoso - Araújo
5	2025-10-01 00:00:00	Porto - Dias de da Mota
6	2025-10-01 00:00:00	Cardoso - Araújo
Total rows: 3528 Query complete 00:00:08.268		

Uma análise de venda mais detalhada com vários joins e group by chega a demorar 8 segundos para retornar.

Considerações finais antes de começar o sketching:

É necessário implementar índices no sql. Acredito que não seja preciso usar materialized view a princípio dado o contexto da aplicação, que é mais com intuito de relatório , e não autenticação ou algo que deva ser muito rápido.