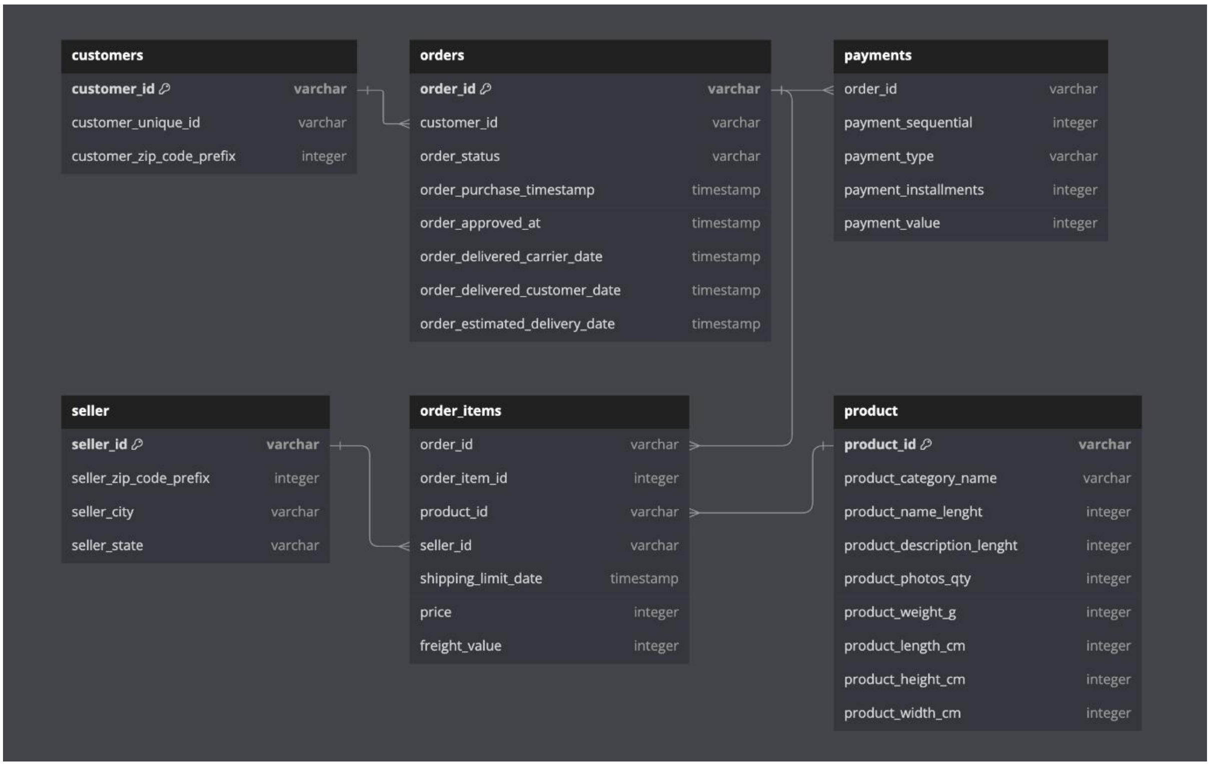
Title: Amazon Brazil Sales & Customer Insights using SQL

Subtitle: A Data-Driven Analysis of Payments, Products, and Customer Trends

Prepared by: Pratik Garud

Date: 03/10/2025

Table And Schema -



customers	
customer_id	varchar
customer_unique_id	varchar
customer_zip_code_prefix	integer

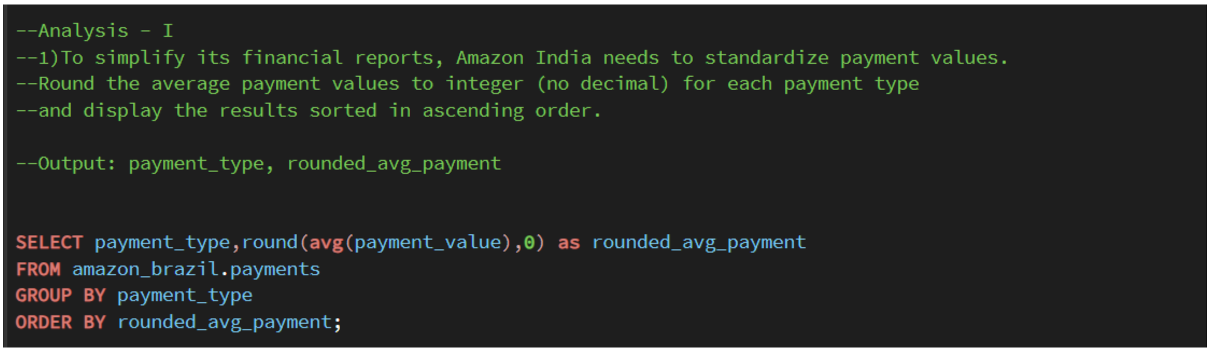
orders	
order_id	varchar
customer_id	varchar
order_status	varchar
order_purchase_timestamp	timestamp
order_approved_at	timestamp
order_delivered_carrier_date	timestamp
order_delivered_customer_date	timestamp
order_estimated_delivery_date	timestamp

payments	
order_id	varchar
payment_sequential	integer
payment_type	varchar
payment_installments	integer
payment_value	integer

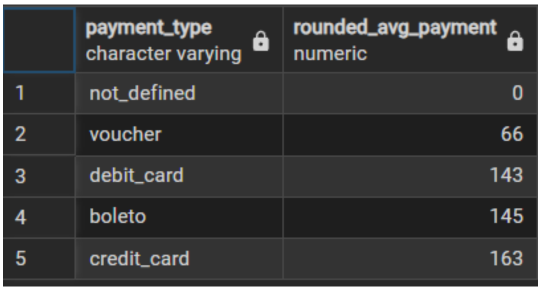
seller	
seller_id	varchar
seller_zip_code_prefix	integer
seller_city	varchar
seller_state	varchar

order_items	
order_id	varchar
order_item_id	integer
product_id	varchar
seller_id	varchar
shipping_limit_date	timestamp
price	integer
freight_value	integer

product	
product_id	varchar
product_category_name	varchar
product_name_lenght	integer
product_description_lenght	integer
product_photos_qty	integer
product_weight_g	integer
product_length_cm	integer
product_height_cm	integer
product_width_cm	integer



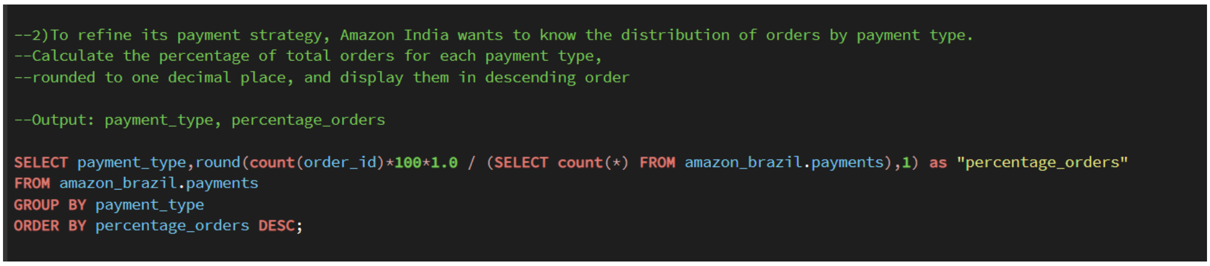
```sql
--Analysis - I
--1)To simplify its financial reports, Amazon India needs to standardize payment values.
--Round the average payment values to integer (no decimal) for each payment type
--and display the results sorted in ascending order.

--Output: payment_type, rounded_avg_payment


SELECT payment_type,round(avg(payment_value),0) as rounded_avg_payment
FROM amazon_brazil.payments
GROUP BY payment_type
ORDER BY rounded_avg_payment;
```



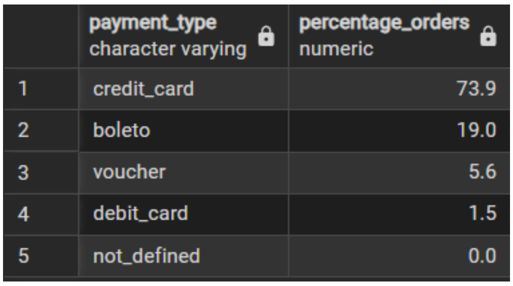
	payment_type character varying	rounded_avg_payment numeric
1	not_defined	0
2	voucher	66
3	debit_card	143
4	boleto	145
5	credit_card	163

1) Rounded Average Payment Values

- **Analysis:** Payment values differ across methods, but rounding highlights general spending ranges per payment type. This simplifies reporting and comparisons.
- **Recommendation:** Use these standardized averages in dashboards and reporting to quickly benchmark transaction behavior by payment method.



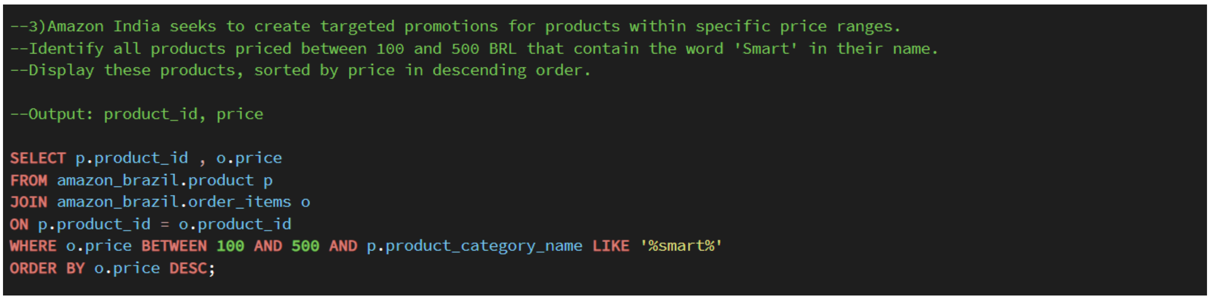
```sql
--2)To refine its payment strategy, Amazon India wants to know the distribution of orders by payment type.
--Calculate the percentage of total orders for each payment type,
--rounded to one decimal place, and display them in descending order

--Output: payment_type, percentage_orders

SELECT payment_type,round(count(order_id)*100*1.0 / (SELECT count(*) FROM amazon_brazil.payments),1) as "percentage_orders"
FROM amazon_brazil.payments
GROUP BY payment_type
ORDER BY percentage_orders DESC;
```



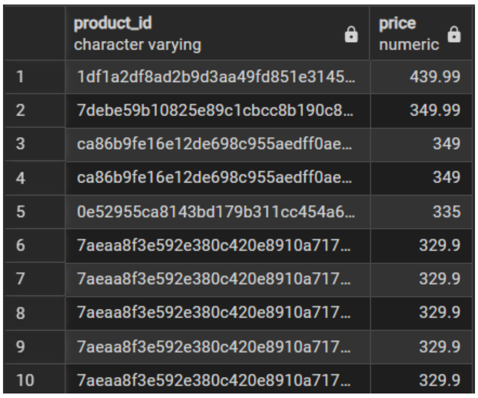
	payment_type character varying	percentage_orders numeric
1	credit_card	73.9
2	boleto	19.0
3	voucher	5.6
4	debit_card	1.5
5	not_defined	0.0

2) Distribution of Orders by Payment Type

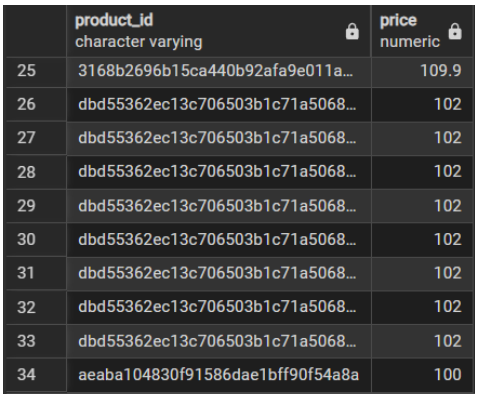
- **Analysis:** Some payment methods dominate order volumes, while others contribute minimally. This shows customer preference trends.
- **Recommendation:** Prioritize support, offers, and reliability for high-usage payment methods while promoting adoption of less-used ones via discounts or incentives.



```
--3)Amazon India seeks to create targeted promotions for products within specific price ranges.
--Identify all products priced between 100 and 500 BRL that contain the word 'Smart' in their name.
--Display these products, sorted by price in descending order.

--Output: product_id, price

SELECT p.product_id , o.price
FROM amazon_brazil.product p
JOIN amazon_brazil.order_items o
ON p.product_id = o.product_id
WHERE o.price BETWEEN 100 AND 500 AND p.product_category_name LIKE '%smart%'
ORDER BY o.price DESC;
```



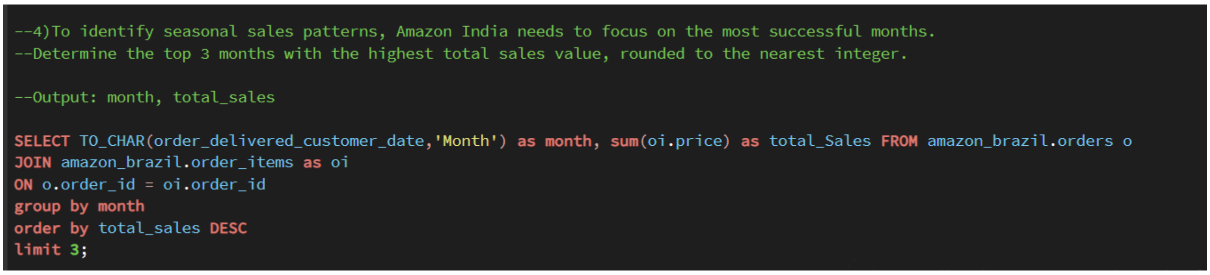
	product_id character varying	price numeric
1	1df1a2df8ad2b9d3aa49fd851e3145...	439.99
2	7debe59b10825e89c1cbcc8b190c8...	349.99
3	ca86b9fe16e12de698c955aedff0ae...	349
4	ca86b9fe16e12de698c955aedff0ae...	349
5	0e52955ca8143bd179b311cc454a6...	335
6	7aeaa8f3e592e380c420e8910a717...	329.9
7	7aeaa8f3e592e380c420e8910a717...	329.9
8	7aeaa8f3e592e380c420e8910a717...	329.9
9	7aeaa8f3e592e380c420e8910a717...	329.9
10	7aeaa8f3e592e380c420e8910a717...	329.9



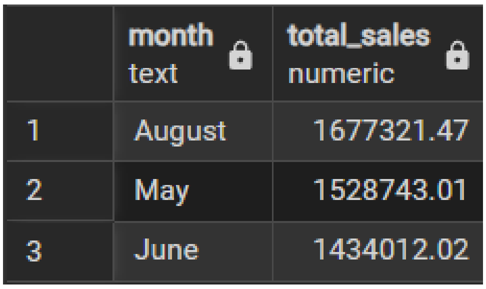
	product_id character varying	price numeric
25	3168b2696b15ca440b92afa9e011a...	109.9
26	dbd55362ec13c706503b1c71a5068...	102
27	dbd55362ec13c706503b1c71a5068...	102
28	dbd55362ec13c706503b1c71a5068...	102
29	dbd55362ec13c706503b1c71a5068...	102
30	dbd55362ec13c706503b1c71a5068...	102
31	dbd55362ec13c706503b1c71a5068...	102
32	dbd55362ec13c706503b1c71a5068...	102
33	dbd55362ec13c706503b1c71a5068...	102
34	aeaba104830f91586dae1bff90f54a8a	100

3) Products Priced 100–500 BRL Containing 'Smart'

- **Analysis:** Mid-range “Smart” products represent a competitive category where customer interest is strong.
- **Recommendation:** Position these products for targeted campaigns (bundles, promotions) since they are affordable yet high-demand.



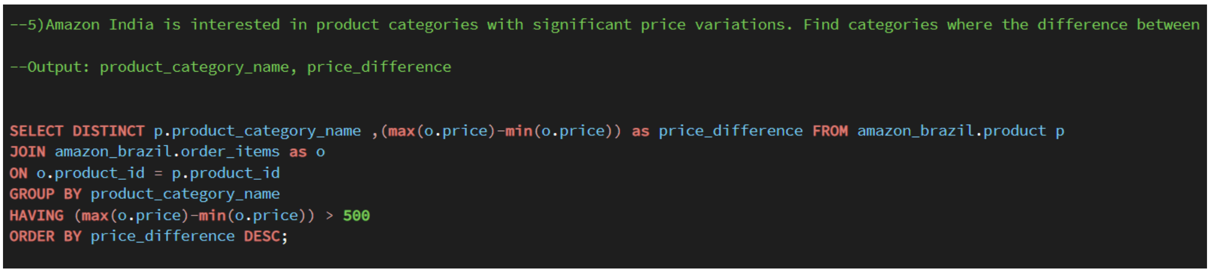
```
--4)To identify seasonal sales patterns, Amazon India needs to focus on the most successful months.
--Determine the top 3 months with the highest total sales value, rounded to the nearest integer.

--Output: month, total_sales

SELECT TO_CHAR(order_delivered_customer_date,'Month') as month, sum(oi.price) as total_Sales FROM amazon_brazil.orders o
JOIN amazon_brazil.order_items as oi
ON o.order_id = oi.order_id
group by month
order by total_sales DESC
limit 3;
```



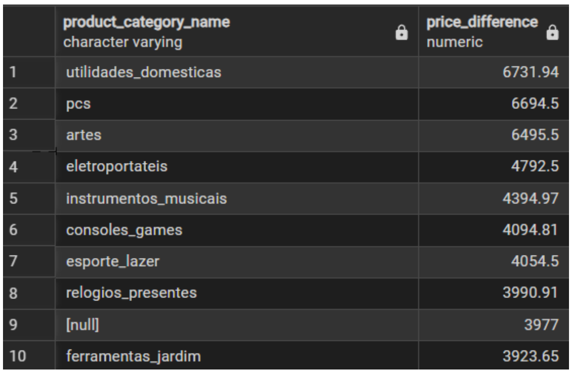
	month text	total_sales numeric
1	August	1677321.47
2	May	1528743.01
3	June	1434012.02

4) Top 3 Months with Highest Sales

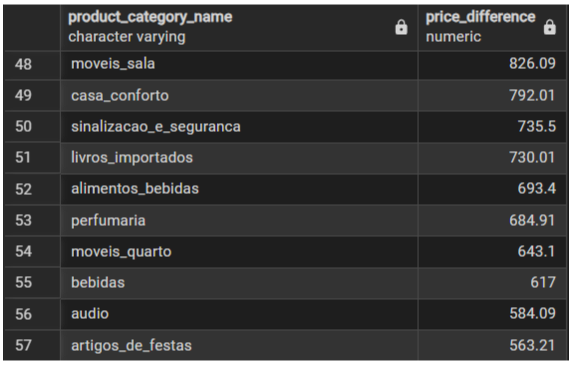
- **Analysis:** Certain months show clear seasonal peaks, likely influenced by holidays, festivals, or sales events.
- **Recommendation:** Align inventory planning, ad spend, and promotions with these months to maximize returns during high-demand periods.



```
--5)Amazon India is interested in product categories with significant price variations. Find categories where the difference between

--Output: product_category_name, price_difference


SELECT DISTINCT p.product_category_name ,(max(o.price)-min(o.price)) as price_difference FROM amazon_brazil.product p
JOIN amazon_brazil.order_items as o
ON o.product_id = p.product_id
GROUP BY product_category_name
HAVING (max(o.price)-min(o.price)) > 500
ORDER BY price_difference DESC;
```



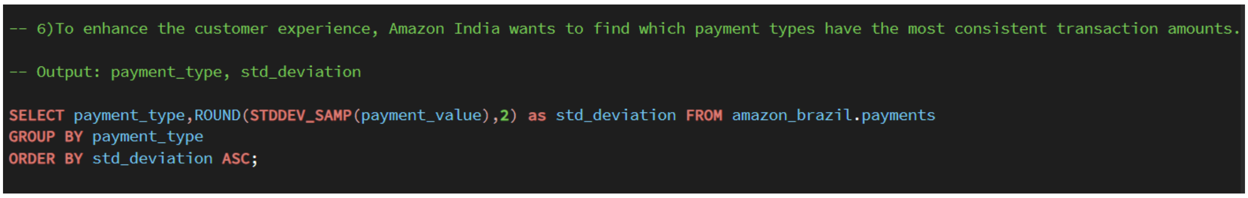
	product_category_name character varying	price_difference numeric
1	utilidades_domesticas	6731.94
2	pcs	6694.5
3	artes	6495.5
4	eletroportateis	4792.5
5	instrumentos_musicais	4394.97
6	consoles_games	4094.81
7	esporte_lazer	4054.5
8	relogios_presentes	3990.91
9	[null]	3977
10	ferramentas_jardim	3923.65



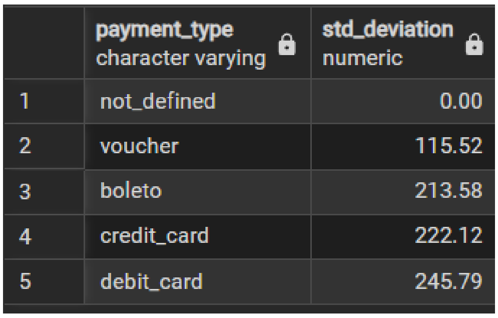
	product_category_name character varying	price_difference numeric
48	moveis_sala	826.09
49	casa_conforto	792.01
50	sinalizacao_e_seguranca	735.5
51	livros_importados	730.01
52	alimentos_bebidas	693.4
53	perfumaria	684.91
54	moveis_quarto	643.1
55	bebidas	617
56	audio	584.09
57	artigos_de_festas	563.21

5) Categories with High Price Variations

- **Analysis:** Large price gaps within categories indicate diverse product ranges or inconsistent pricing strategies.
- **Recommendation:** Review pricing structures and consider tiered marketing (premium vs. budget segments) to cater to different customer groups.



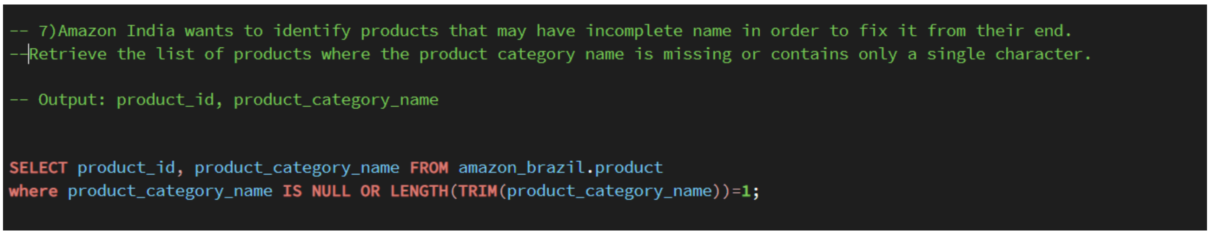
```
-- 6)To enhance the customer experience, Amazon India wants to find which payment types have the most consistent transaction amounts.

-- Output: payment_type, std_deviation

SELECT payment_type,ROUND(STDDEV_SAMP(payment_value),2) as std_deviation FROM amazon_brazil.payments
GROUP BY payment_type
ORDER BY std_deviation ASC;
```



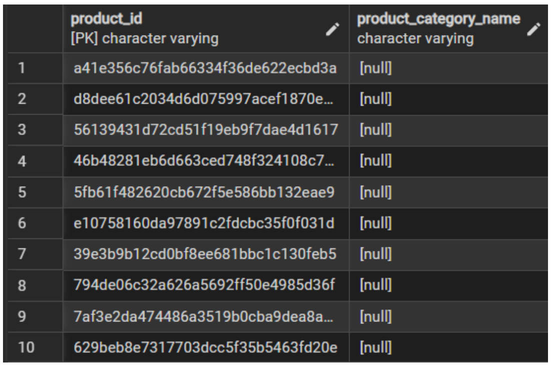
	payment_type character varying	std_deviation numeric
1	not_defined	0.00
2	voucher	115.52
3	boleto	213.58
4	credit_card	222.12
5	debit_card	245.79

6) Payment Types with Least Variance in Amounts

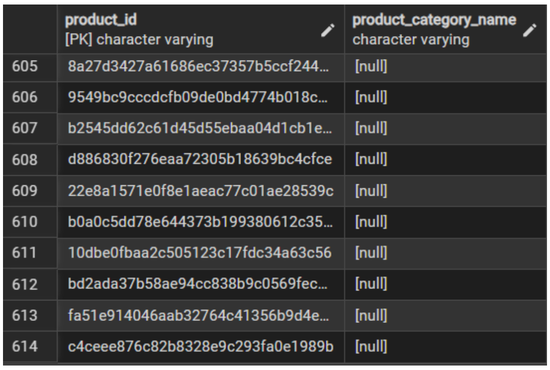
- **Analysis:** Some payment methods have very stable transaction amounts, suggesting consistent usage patterns.
- **Recommendation:** Leverage these stable payment methods for subscriptions, EMI plans, or recurring payments where predictability is key.



```
-- 7)Amazon India wants to identify products that may have incomplete name in order to fix it from their end.
--Retrieve the list of products where the product category name is missing or contains only a single character.

-- Output: product_id, product_category_name


SELECT product_id, product_category_name FROM amazon_brazil.product
where product_category_name IS NULL OR LENGTH(TRIM(product_category_name))=1;
```



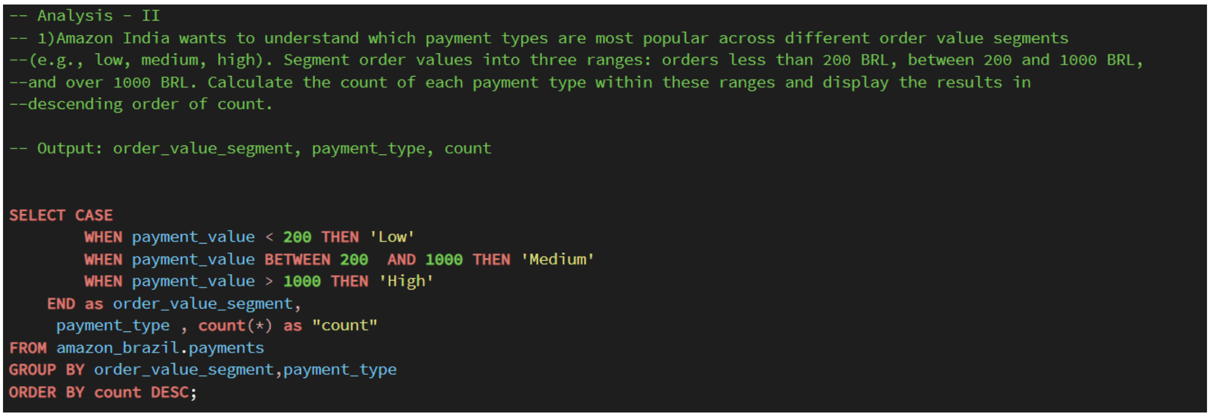
	product_id [PK] character varying	product_category_name character varying
1	a41e356c76fab66334f36de622ecbd3a	[null]
2	d8dee61c2034d6d075997acef1870e...	[null]
3	56139431d72cd51f19eb9f7dae4d1617	[null]
4	46b48281eb6d663ced748f324108c7...	[null]
5	5fb61f482620cb672f5e586bb132eae9	[null]
6	e10758160da97891c2fdcbc35f0f031d	[null]
7	39e3b9b12cd0bf8ee681bbc1c130feb5	[null]
8	794de06c32a626a5692ff50e4985d36f	[null]
9	7af3e2da474486a3519b0cba9dea8a...	[null]
10	629beb8e7317703dcc5f35b5463fd20e	[null]



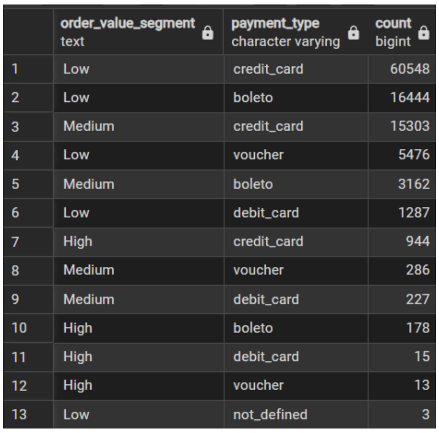
	product_id [PK] character varying	product_category_name character varying
605	8a27d3427a61686ec37357b5ccf244...	[null]
606	9549bc9cccdcfb09de0bd4774b018c...	[null]
607	b2545dd62c61d45d55ebaa04d1cb1e...	[null]
608	d886830f276eaa72305b18639bc4cfce	[null]
609	22e8a1571e0f8e1aeac77c01ae28539c	[null]
610	b0a0c5dd78e644373b199380612c35...	[null]
611	10dbe0fbaa2c505123c17fdc34a63c56	[null]
612	bd2ada37b58ae94cc838b9c0569fec...	[null]
613	fa51e914046aab32764c41356b9d4e...	[null]
614	c4ceee876c82b8328e9c293fa0e1989b	[null]

7) Products with Missing or Incomplete Names

- **Analysis:** Missing or incomplete product categories suggest data quality issues that can affect reporting and customer trust.
- **Recommendation:** Standardize and clean product data regularly to ensure accurate listings, better search visibility, and improved customer experience.



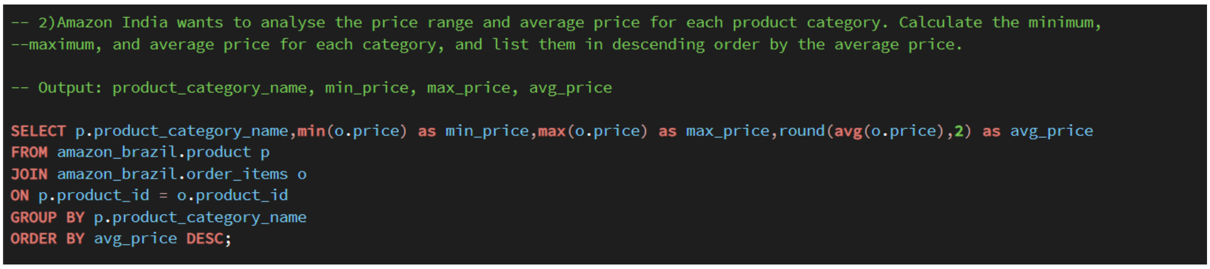
```
-- Analysis - II
-- 1)Amazon India wants to understand which payment types are most popular across different order value segments
--(e.g., low, medium, high). Segment order values into three ranges: orders less than 200 BRL, between 200 and 1000 BRL,
--and over 1000 BRL. Calculate the count of each payment type within these ranges and display the results in
--descending order of count.

-- Output: order_value_segment, payment_type, count


SELECT CASE
        WHEN payment_value < 200 THEN 'Low'
        WHEN payment_value BETWEEN 200  AND 1000 THEN 'Medium'
        WHEN payment_value > 1000 THEN 'High'
    END as order_value_segment,
     payment_type , count(*) as "count"
FROM amazon_brazil.payments
GROUP BY order_value_segment,payment_type
ORDER BY count DESC;
```



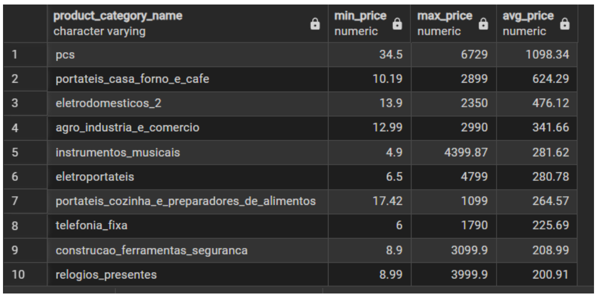
	order_value_segment text	payment_type character varying	count bigint
1	Low	credit_card	60548
2	Low	boleto	16444
3	Medium	credit_card	15303
4	Low	voucher	5476
5	Medium	boleto	3162
6	Low	debit_card	1287
7	High	credit_card	944
8	Medium	voucher	286
9	Medium	debit_card	227
10	High	boleto	178
11	High	debit_card	15
12	High	voucher	13
13	Low	not_defined	3

1) Payment Types by Order Value Segment

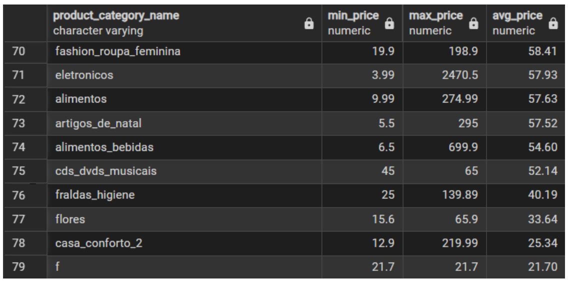
- **Analysis:** Payment preferences differ across low, medium, and high order values, with some methods dominating specific ranges.
- **Recommendation:** Align promotions and payment offers with segments—for example, push EMI or credit for high-value orders, and UPI/wallets for low-value ones.



```
-- 2)Amazon India wants to analyse the price range and average price for each product category. Calculate the minimum,
--maximum, and average price for each category, and list them in descending order by the average price.

-- Output: product_category_name, min_price, max_price, avg_price

SELECT p.product_category_name,min(o.price) as min_price,max(o.price) as max_price,round(avg(o.price),2) as avg_price
FROM amazon_brazil.product p
JOIN amazon_brazil.order_items o
ON p.product_id = o.product_id
GROUP BY p.product_category_name
ORDER BY avg_price DESC;
```



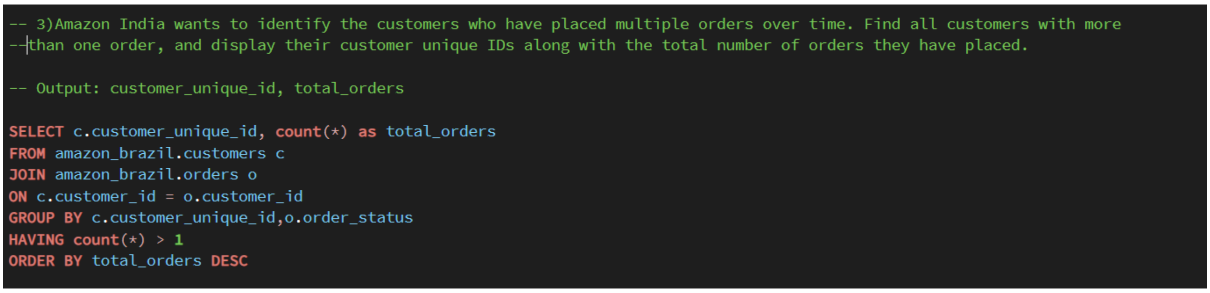
	product_category_name character varying	min_price numeric	max_price numeric	avg_price numeric
1	pcs	34.5	6729	1098.34
2	portateis_casa_forno_e_cafe	10.19	2899	624.29
3	eletrodomesticos_2	13.9	2350	476.12
4	agro_industria_e_comercio	12.99	2990	341.66
5	instrumentos_musicais	4.9	4399.87	281.62
6	eletroportateis	6.5	4799	280.78
7	portateis_cozinha_e_preparadores_de_alimentos	17.42	1099	264.57
8	telefonia_fixa	6	1790	225.69
9	construcao_ferramentas_seguranca	8.9	3099.9	208.99
10	relogios_presentes	8.99	3999.9	200.91



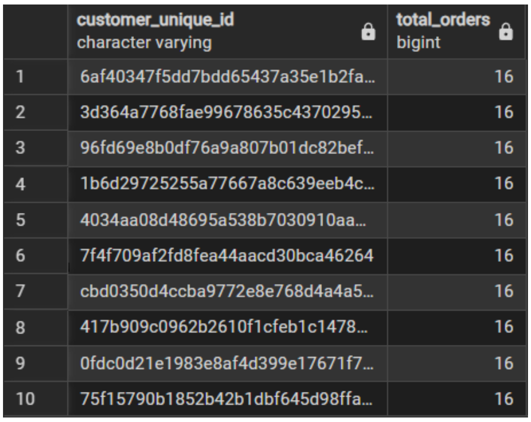
	product_category_name character varying	min_price numeric	max_price numeric	avg_price numeric
70	fashion_roupa_feminina	19.9	198.9	58.41
71	eletronicos	3.99	2470.5	57.93
72	alimentos	9.99	274.99	57.63
73	artigos_de_natal	5.5	295	57.52
74	alimentos_bebidas	6.5	699.9	54.60
75	cds_dvds_musicais	45	65	52.14
76	fraldas_higiene	25	139.89	40.19
77	flores	15.6	65.9	33.64
78	casa_conforto_2	12.9	219.99	25.34
79	f	21.7	21.7	21.70

2) Price Range & Averages per Category

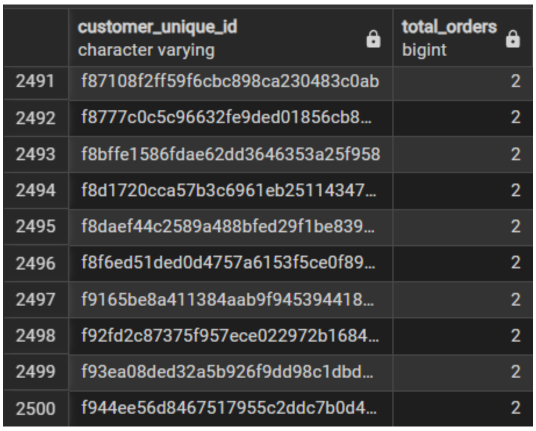
- **Analysis:** Categories vary widely in minimum, maximum, and average prices, showing distinct affordability bands.
- **Recommendation:** Position premium categories with higher marketing spend, while offering discounts on budget categories to drive volume.



```sql
-- 3)Amazon India wants to identify the customers who have placed multiple orders over time. Find all customers with more
--than one order, and display their customer unique IDs along with the total number of orders they have placed.

-- Output: customer_unique_id, total_orders

SELECT c.customer_unique_id, count(*) as total_orders
FROM amazon_brazil.customers c
JOIN amazon_brazil.orders o
ON c.customer_id = o.customer_id
GROUP BY c.customer_unique_id,o.order_status
HAVING count(*) > 1
ORDER BY total_orders DESC
```



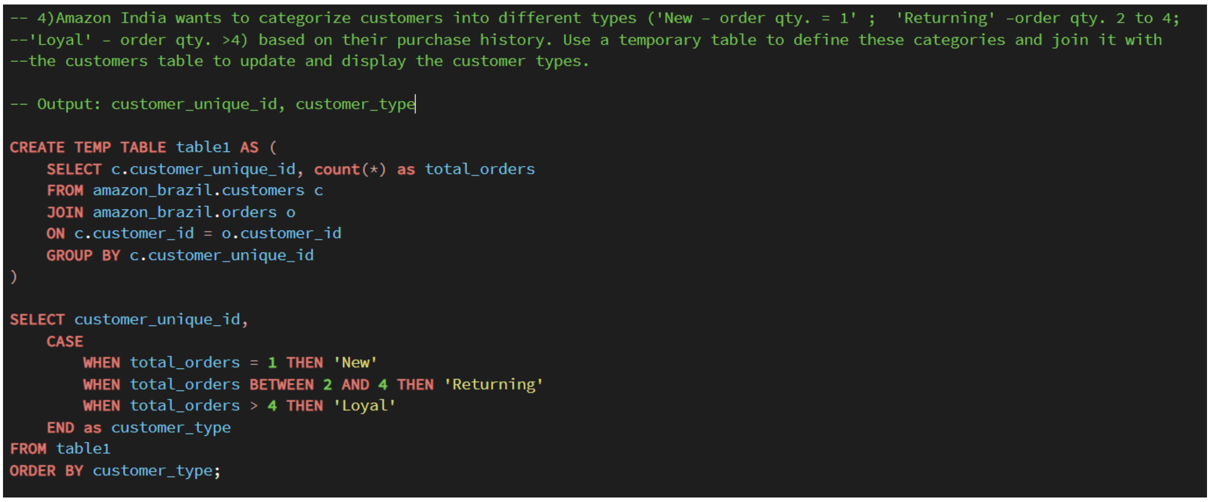
	customer_unique_id character varying	total_orders bigint
1	6af40347f5dd7bdd65437a35e1b2fa...	16
2	3d364a7768fae99678635c4370295...	16
3	96fd69e8b0df76a9a807b01dc82bef...	16
4	1b6d29725255a77667a8c639eeb4c...	16
5	4034aa08d48695a538b7030910aa...	16
6	7f4f709af2fd8fea44aacd30bca46264	16
7	cbd0350d4ccba9772e8e768d4a4a5...	16
8	417b909c0962b2610f1cfeb1c1478...	16
9	0fdc0d21e1983e8af4d399e17671f7...	16
10	75f15790b1852b42b1dbf645d98ffa...	16



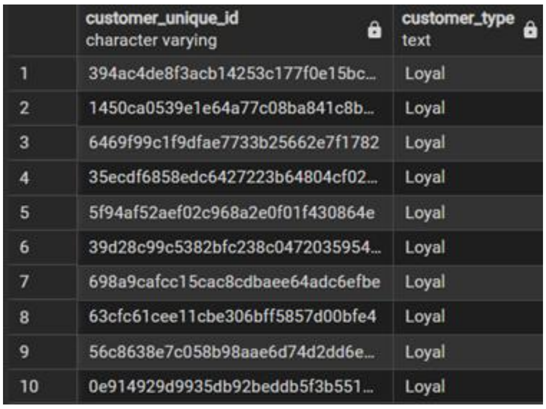
	customer_unique_id character varying	total_orders bigint
2491	f87108f2ff59f6cbc898ca230483c0ab	2
2492	f8777c0c5c96632fe9ded01856cb8...	2
2493	f8bffe1586fdae62dd3646353a25f958	2
2494	f8d1720cca57b3c6961eb25114347...	2
2495	f8daef44c2589a488bfed29f1be839...	2
2496	f8f6ed51ded0d4757a6153f5ce0f89...	2
2497	f9165be8a411384aab9f945394418...	2
2498	f92fd2c87375f957ece022972b1684...	2
2499	f93ea08ded32a5b926f9dd98c1dbd...	2
2500	f944ee56d8467517955c2ddc7b0d4...	2

3) Customers with Multiple Orders

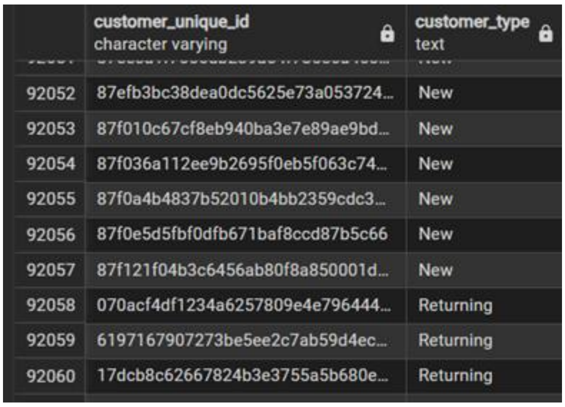
- **Analysis:** Identifies a segment of repeat buyers who contribute significantly to long-term sales.
- **Recommendation:** Launch loyalty perks and personalized campaigns for these customers to maximize retention.



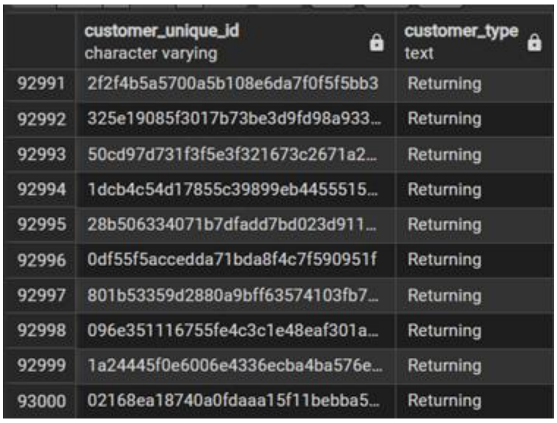
```sql
-- 4)Amazon India wants to categorize customers into different types ('New – order qty. = 1' ;  'Returning' -order qty. 2 to 4;
--'Loyal' – order qty. >4) based on their purchase history. Use a temporary table to define these categories and join it with
--the customers table to update and display the customer types.

-- Output: customer_unique_id, customer_type

CREATE TEMP TABLE table1 AS (
    SELECT c.customer_unique_id, count(*) as total_orders
    FROM amazon_brazil.customers c
    JOIN amazon_brazil.orders o
    ON c.customer_id = o.customer_id
    GROUP BY c.customer_unique_id
)

SELECT customer_unique_id,
    CASE
        WHEN total_orders = 1 THEN 'New'
        WHEN total_orders BETWEEN 2 AND 4 THEN 'Returning'
        WHEN total_orders > 4 THEN 'Loyal'
    END as customer_type
FROM table1
ORDER BY customer_type;
```



	customer_unique_id character varying	customer_type text
1	394ac4de8f3acb14253c177f0e15bc...	Loyal
2	1450ca0539e1e64a77c08ba841c8b...	Loyal
3	6469f99c1f9dfae7733b25662e7f1782	Loyal
4	35ecdf6858edc6427223b64804cf02...	Loyal
5	5f94af52aef02c968a2e0f01f430864e	Loyal
6	39d28c99c5382bfc238c0472035954...	Loyal
7	698a9cafcc15cac8cdbaee64adc6efbe	Loyal
8	63cfc61cee11cbe306bff5857d00bfe4	Loyal
9	56c8638e7c058b98aae6d74d2dd6e...	Loyal
10	0e914929d9935db92beddb5f3b551...	Loyal



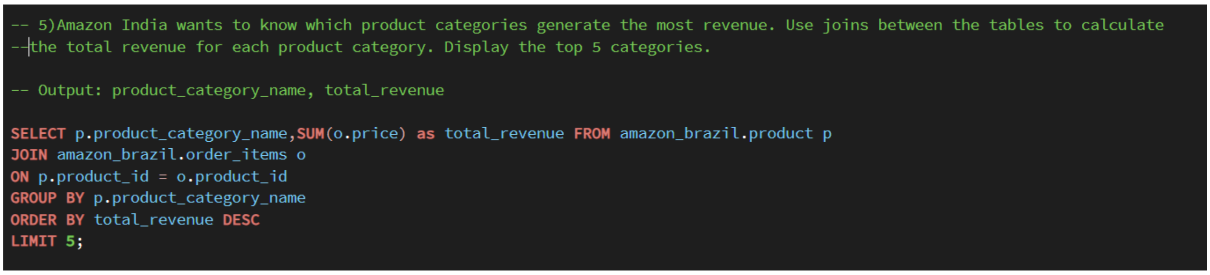
	customer_unique_id character varying	customer_type text
92052	87efb3bc38dea0dc5625e73a053724...	New
92053	87f010c67cf8eb940ba3e7e89ae9bd...	New
92054	87f036a112ee9b2695f0eb5f063c74...	New
92055	87f0a4b4837b52010b4bb2359cdc3...	New
92056	87f0e5d5fbf0dfb671baf8ccd87b5c66	New
92057	87f121f04b3c6456ab80f8a850001d...	New
92058	070acf4df1234a6257809e4e796444...	Returning
92059	6197167907273be5ee2c7ab59d4ec...	Returning
92060	17dcb8c62667824b3e3755a5b680e...	Returning



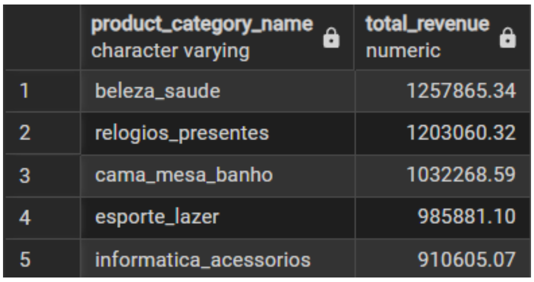
	customer_unique_id character varying	customer_type text
92991	2f2f4b5a5700a5b108e6da7f0f5f5bb3	Returning
92992	325e19085f3017b73be3d9fd98a933...	Returning
92993	50cd97d731f3f5e3f321673c2671a2...	Returning
92994	1dcb4c54d17855c39899eb4455515...	Returning
92995	28b506334071b7dfadd7bd023d911...	Returning
92996	0df55f5accedda71bda8f4c7f590951f	Returning
92997	801b53359d2880a9bff63574103fb7...	Returning
92998	096e351116755fe4c3c1e48eaf301a...	Returning
92999	1a24445f0e6006e4336ecba4ba576e...	Returning
93000	02168ea18740a0fdaaa15f11bebba5...	Returning

4) Customer Segmentation by Orders

- **Analysis:** Customers fall into clear groups—new, returning, and loyal—based on purchase history.
- **Recommendation:** Tailor engagement: welcome discounts for new, upselling for returning, and exclusive rewards for loyal customers.



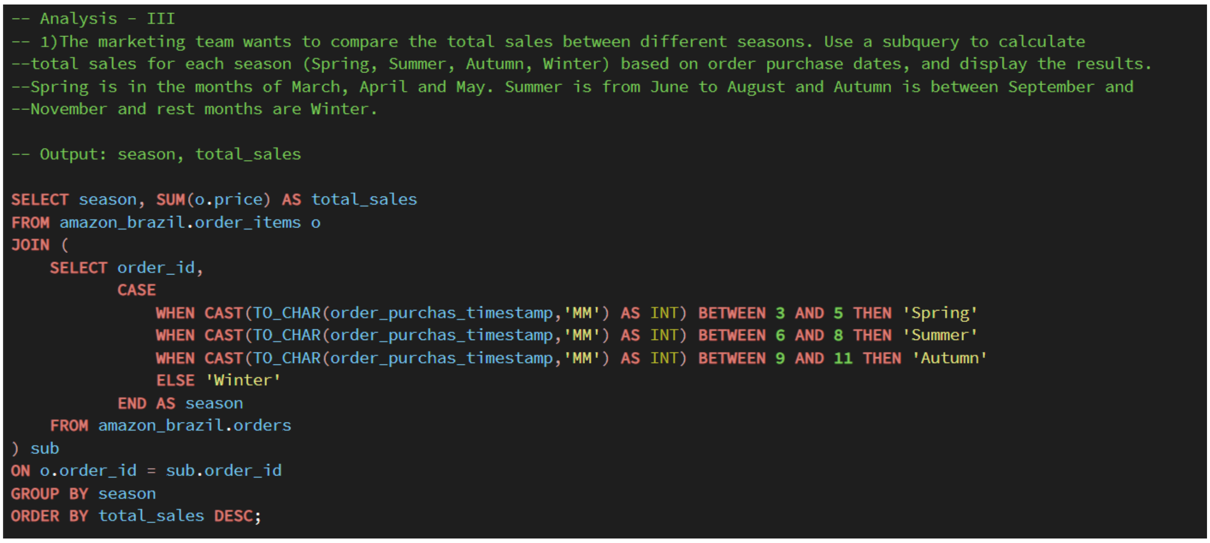
```sql
-- 5)Amazon India wants to know which product categories generate the most revenue. Use joins between the tables to calculate
--the total revenue for each product category. Display the top 5 categories.

-- Output: product_category_name, total_revenue

SELECT p.product_category_name,SUM(o.price) as total_revenue FROM amazon_brazil.product p
JOIN amazon_brazil.order_items o
ON p.product_id = o.product_id
GROUP BY p.product_category_name
ORDER BY total_revenue DESC
LIMIT 5;
```



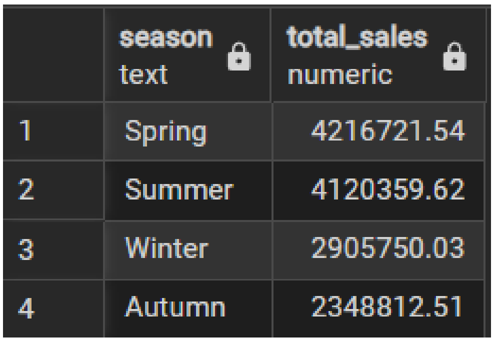
	product_category_name character varying	total_revenue numeric
1	beleza_saude	1257865.34
2	relogios_presentes	1203060.32
3	cama_mesa_banho	1032268.59
4	esporte_lazer	985881.10
5	informatica_acessorios	910605.07

5) Top 5 Revenue-Generating Categories

- **Analysis:** A small set of product categories contributes disproportionately to revenue.
- **Recommendation:** Focus inventory, promotions, and partnerships on these top-performing categories to sustain growth.



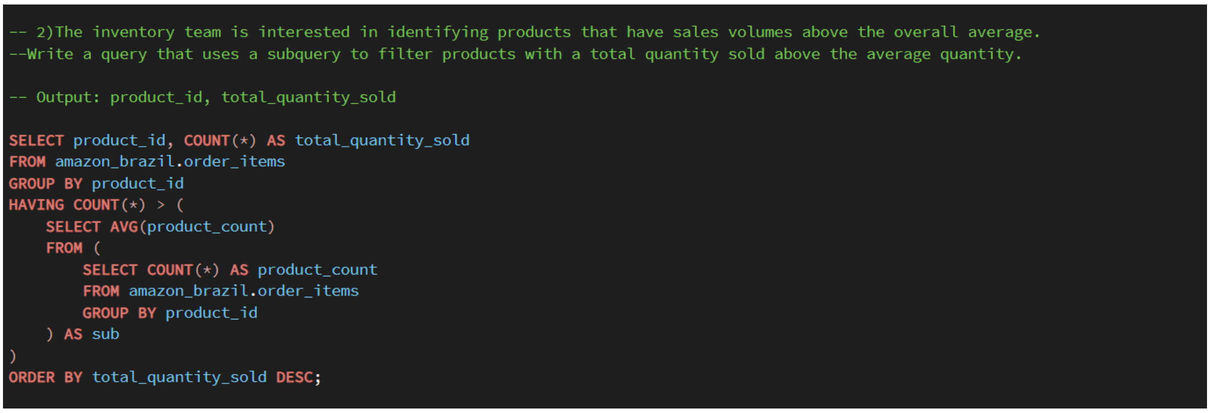
```sql
-- Analysis - III
-- 1)The marketing team wants to compare the total sales between different seasons. Use a subquery to calculate
--total sales for each season (Spring, Summer, Autumn, Winter) based on order purchase dates, and display the results.
--Spring is in the months of March, April and May. Summer is from June to August and Autumn is between September and
--November and rest months are Winter.

-- Output: season, total_sales

SELECT season, SUM(o.price) AS total_sales
FROM amazon_brazil.order_items o
JOIN (
    SELECT order_id,
           CASE
               WHEN CAST(TO_CHAR(order_purchas_timestamp,'MM') AS INT) BETWEEN 3 AND 5 THEN 'Spring'
               WHEN CAST(TO_CHAR(order_purchas_timestamp,'MM') AS INT) BETWEEN 6 AND 8 THEN 'Summer'
               WHEN CAST(TO_CHAR(order_purchas_timestamp,'MM') AS INT) BETWEEN 9 AND 11 THEN 'Autumn'
               ELSE 'Winter'
           END AS season
    FROM amazon_brazil.orders
) sub
ON o.order_id = sub.order_id
GROUP BY season
ORDER BY total_sales DESC;
```



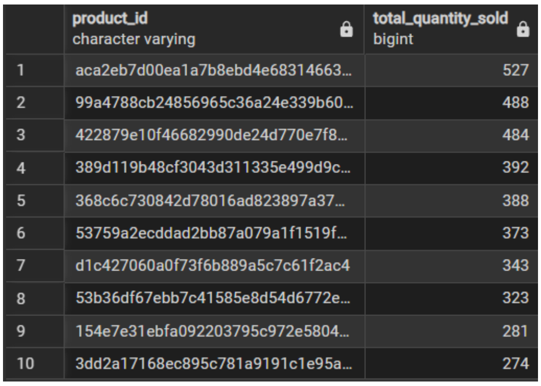
	season text	total_sales numeric
1	Spring	4216721.54
2	Summer	4120359.62
3	Winter	2905750.03
4	Autumn	2348812.51

1) Seasonal Sales Comparison

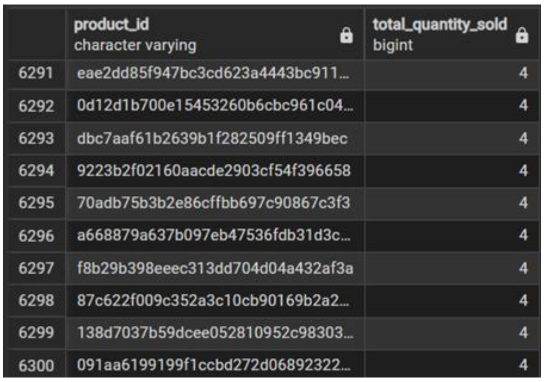
- **Analysis:** Sales vary across seasons, with clear peaks in specific quarters tied to shopping events and festivals.
- **Recommendation:** Boost marketing budgets and stock during peak seasons, while using off-peak months for clearance sales.



```
-- 2)The inventory team is interested in identifying products that have sales volumes above the overall average.
--Write a query that uses a subquery to filter products with a total quantity sold above the average quantity.

-- Output: product_id, total_quantity_sold

SELECT product_id, COUNT(*) AS total_quantity_sold
FROM amazon_brazil.order_items
GROUP BY product_id
HAVING COUNT(*) > (
    SELECT AVG(product_count)
    FROM (
        SELECT COUNT(*) AS product_count
        FROM amazon_brazil.order_items
        GROUP BY product_id
    ) AS sub
)
ORDER BY total_quantity_sold DESC;
```



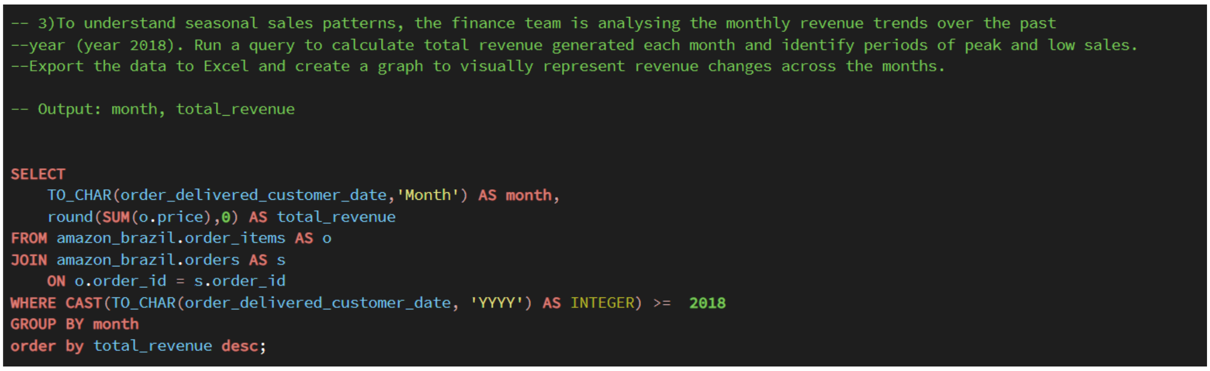
	product_id character varying	total_quantity_sold bigint
1	aca2eb7d00ea1a7b8ebd4e68314663…	527
2	99a4788cb24856965c36a24e339b60…	488
3	422879e10f46682990de24d770e7f8…	484
4	389d119b48cf3043d311335e499d9c…	392
5	368c6c730842d78016ad823897a37…	388
6	53759a2ecddad2bb87a079a1f1519f…	373
7	d1c427060a0f73f6b889a5c7c61f2ac4	343
8	53b36df67ebb7c41585e8d54d6772e…	323
9	154e7e31ebfa092203795c972e5804…	281
10	3dd2a17168ec895c781a9191c1e95a…	274



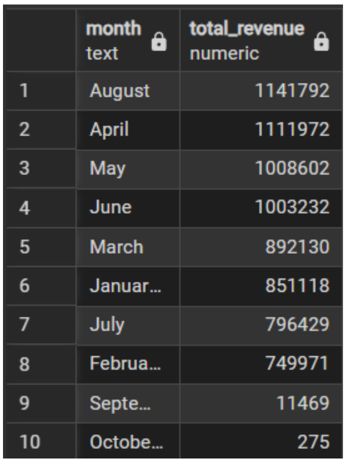
	product_id character varying	total_quantity_sold bigint
6291	eae2dd85f947bc3cd623a4443bc911…	4
6292	0d12d1b700e15453260b6cbc961c04…	4
6293	dbc7aaf61b2639b1f282509ff1349bec	4
6294	9223b2f02160aacde2903cf54f396658	4
6295	70adb75b3b2e86cffbb697c90867c3f3	4
6296	a668879a637b097eb47536fdb31d3c…	4
6297	f8b29b398eeec313dd704d04a432af3a	4
6298	87c622f009c352a3c10cb90169b2a2…	4
6299	138d7037b59dcee052810952c98303…	4
6300	091aa6199199f1ccbd272d06892322…	4

2) Products with Sales Above Average

- **Analysis:** Certain products consistently outperform the overall average in sales volume.
- **Recommendation:** Prioritize these high-demand products for promotions, bundle offers, and fast replenishment.



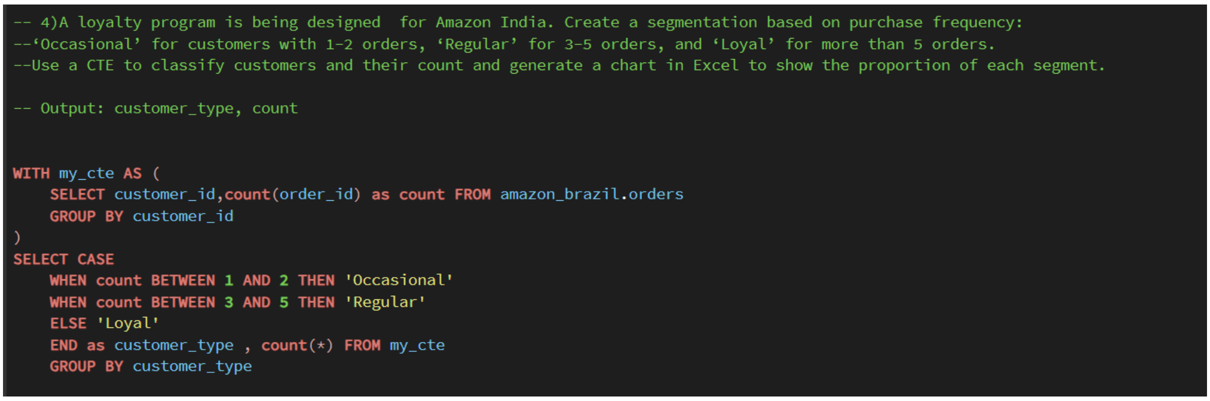
```
-- 3)To understand seasonal sales patterns, the finance team is analysing the monthly revenue trends over the past
--year (year 2018). Run a query to calculate total revenue generated each month and identify periods of peak and low sales.
--Export the data to Excel and create a graph to visually represent revenue changes across the months.

-- Output: month, total_revenue


SELECT
    TO_CHAR(order_delivered_customer_date,'Month') AS month,
    round(SUM(o.price),0) AS total_revenue
FROM amazon_brazil.order_items AS o
JOIN amazon_brazil.orders AS s
    ON o.order_id = s.order_id
WHERE CAST(TO_CHAR(order_delivered_customer_date, 'YYYY') AS INTEGER) >=  2018
GROUP BY month
order by total_revenue desc;
```



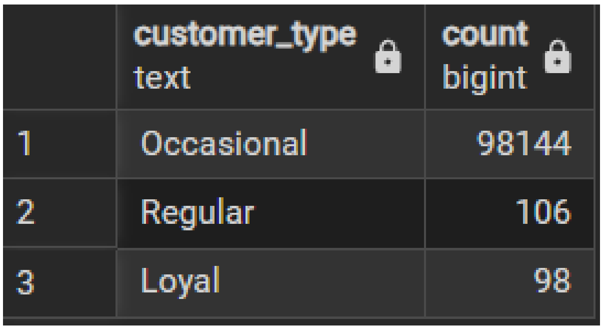
	month text	total_revenue numeric
1	August	1141792
2	April	1111972
3	May	1008602
4	June	1003232
5	March	892130
6	Januar...	851118
7	July	796429
8	Februa...	749971
9	Septe...	11469
10	Octobe...	275

3) Monthly Revenue Trends (2018)

- **Analysis:** Revenue fluctuates month to month, with visible high and low periods indicating customer buying cycles.
- **Recommendation:** Use insights for budget planning and launch seasonal offers during low-revenue months to stabilize sales.



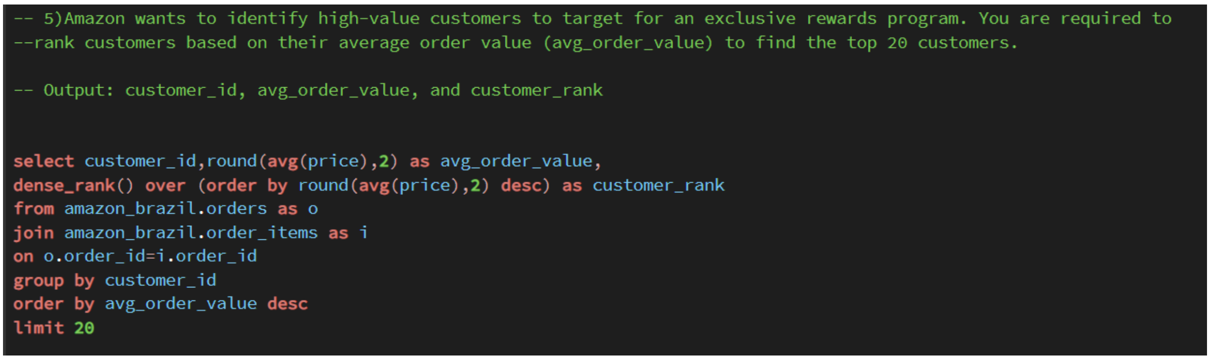
```
-- 4)A loyalty program is being designed  for Amazon India. Create a segmentation based on purchase frequency:
--‘Occasional’ for customers with 1-2 orders, ‘Regular’ for 3-5 orders, and ‘Loyal’ for more than 5 orders.
--Use a CTE to classify customers and their count and generate a chart in Excel to show the proportion of each segment.

-- Output: customer_type, count


WITH my_cte AS (
    SELECT customer_id,count(order_id) as count FROM amazon_brazil.orders
    GROUP BY customer_id
)
SELECT CASE
    WHEN count BETWEEN 1 AND 2 THEN 'Occasional'
    WHEN count BETWEEN 3 AND 5 THEN 'Regular'
    ELSE 'Loyal'
    END as customer_type , count(*) FROM my_cte
    GROUP BY customer_type
```



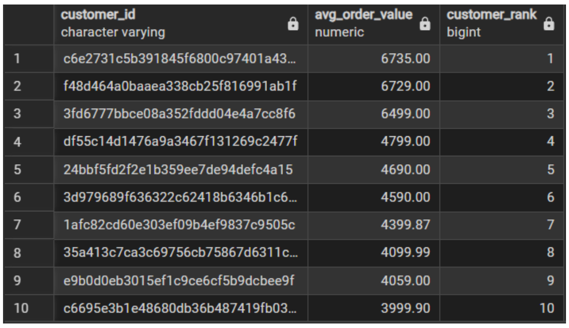
	customer_type text	count bigint
1	Occasional	98144
2	Regular	106
3	Loyal	98

4) Customer Segmentation by Purchase Frequency (CTE)

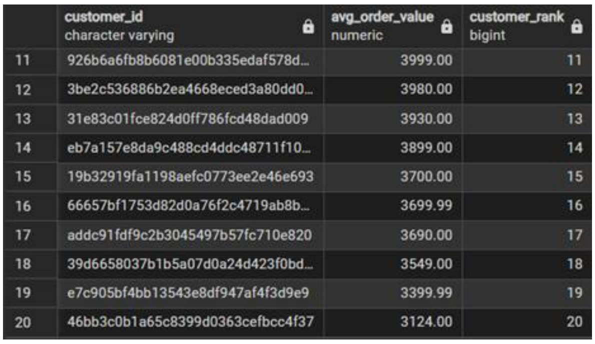
- **Analysis:** Customers cluster into occasional, regular, and loyal groups, each with different value potential.
- **Recommendation:** Nurture occasional customers with reactivation offers, encourage regulars with bundles, and reward loyal customers with premium benefits.



```
-- 5)Amazon wants to identify high-value customers to target for an exclusive rewards program. You are required to
--rank customers based on their average order value (avg_order_value) to find the top 20 customers.

-- Output: customer_id, avg_order_value, and customer_rank


select customer_id,round(avg(price),2) as avg_order_value,
dense_rank() over (order by round(avg(price),2) desc) as customer_rank
from amazon_brazil.orders as o
join amazon_brazil.order_items as i
on o.order_id=i.order_id
group by customer_id
order by avg_order_value desc
limit 20
```



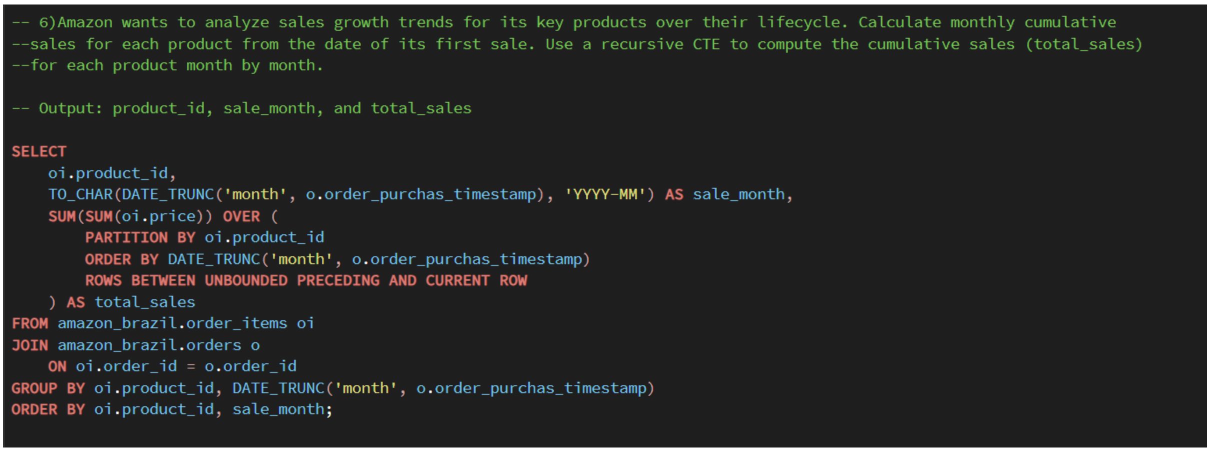
	customer_id character varying	avg_order_value numeric	customer_rank bigint
1	c6e2731c5b391845f6800c97401a43...	6735.00	1
2	f48d464a0baaea338cb25f816991ab1f	6729.00	2
3	3fd6777bbce08a352fddd04e4a7cc8f6	6499.00	3
4	df55c14d1476a9a3467f131269c2477f	4799.00	4
5	24bbf5fd2f2e1b359ee7de94defc4a15	4690.00	5
6	3d979689f636322c62418b6346b1c6...	4590.00	6
7	1afc82cd60e303ef09b4ef9837c9505c	4399.87	7
8	35a413c7ca3c69756cb75867d6311c...	4099.99	8
9	e9b0d0eb3015ef1c9ce6cf5b9dcbee9f	4059.00	9
10	c6695e3b1e48680db36b487419fb03...	3999.90	10



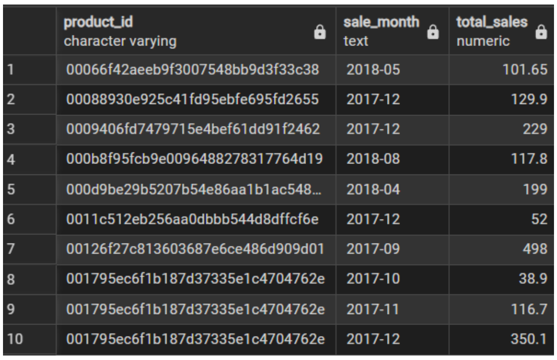
	customer_id character varying	avg_order_value numeric	customer_rank bigint
11	926b6a6fb8b6081e00b335edaf578d...	3999.00	11
12	3be2c536886b2ea4668eced3a80dd0...	3980.00	12
13	31e83c01fce824d0ff786fcd48dad009	3930.00	13
14	eb7a157e8da9c488cd4ddc48711f10...	3899.00	14
15	19b32919fa1198aefc0773ee2e46e693	3700.00	15
16	66657bf1753d82d0a76f2c4719ab8b...	3699.99	16
17	addc91fdf9c2b3045497b57fc710e820	3690.00	17
18	39d6658037b1b5a07d0a24d423f0bd...	3549.00	18
19	e7c905bf4bb13543e8df947af4f3d9e9	3399.99	19
20	46bb3c0b1a65c8399d0363cefbcc4f37	3124.00	20

5) High-Value Customers (Top 20 Ranking)

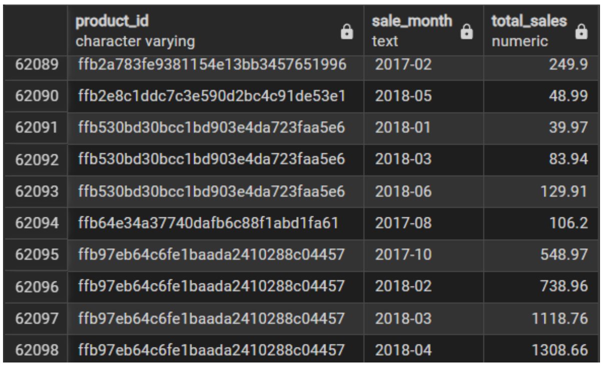
- **Analysis:** A small set of customers drives high order values, making them key contributors to revenue.
- **Recommendation:** Build exclusive VIP programs for these customers with personalized deals to increase lifetime value.



```sql
-- 6)Amazon wants to analyze sales growth trends for its key products over their lifecycle. Calculate monthly cumulative
--sales for each product from the date of its first sale. Use a recursive CTE to compute the cumulative sales (total_sales)
--for each product month by month.

-- Output: product_id, sale_month, and total_sales

SELECT
    oi.product_id,
    TO_CHAR(DATE_TRUNC('month', o.order_purchas_timestamp), 'YYYY-MM') AS sale_month,
    SUM(SUM(oi.price)) OVER (
        PARTITION BY oi.product_id
        ORDER BY DATE_TRUNC('month', o.order_purchas_timestamp)
        ROWS BETWEEN UNBOUNDED PRECEDING AND CURRENT ROW
    ) AS total_sales
FROM amazon_brazil.order_items oi
JOIN amazon_brazil.orders o
    ON oi.order_id = o.order_id
GROUP BY oi.product_id, DATE_TRUNC('month', o.order_purchas_timestamp)
ORDER BY oi.product_id, sale_month;
```



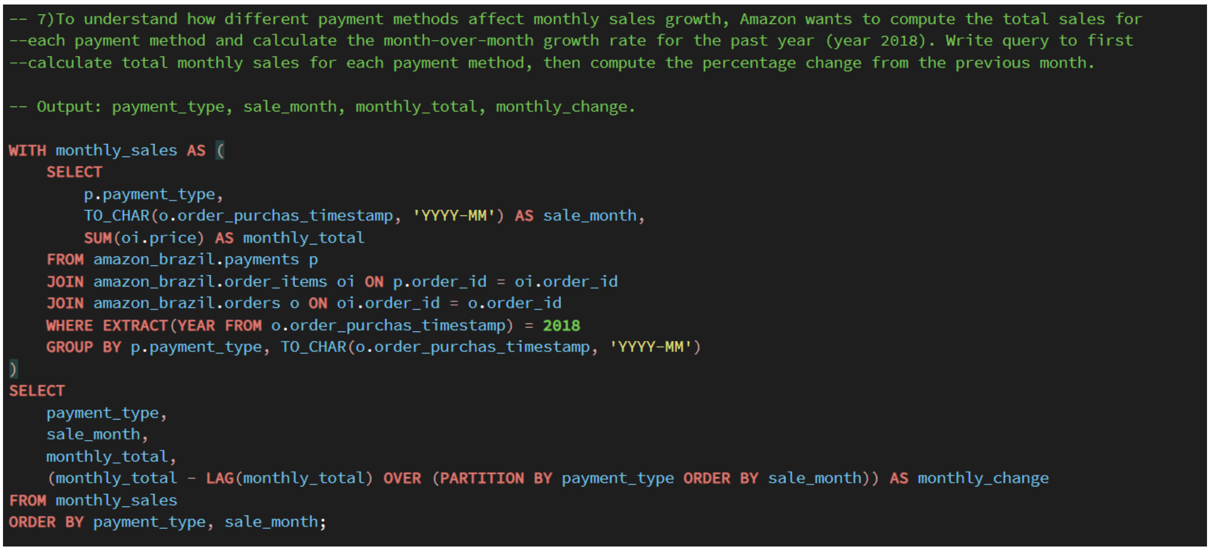
	product_id character varying	sale_month text	total_sales numeric
1	00066f42aeeb9f3007548bb9d3f33c38	2018-05	101.65
2	00088930e925c41fd95ebfe695fd2655	2017-12	129.9
3	0009406fd7479715e4bef61dd91f2462	2017-12	229
4	000b8f95fcb9e0096488278317764d19	2018-08	117.8
5	000d9be29b5207b54e86aa1b1ac548...	2018-04	199
6	0011c512eb256aa0dbbb544d8dffcf6e	2017-12	52
7	00126f27c813603687e6ce486d909d01	2017-09	498
8	001795ec6f1b187d37335e1c4704762e	2017-10	38.9
9	001795ec6f1b187d37335e1c4704762e	2017-11	116.7
10	001795ec6f1b187d37335e1c4704762e	2017-12	350.1



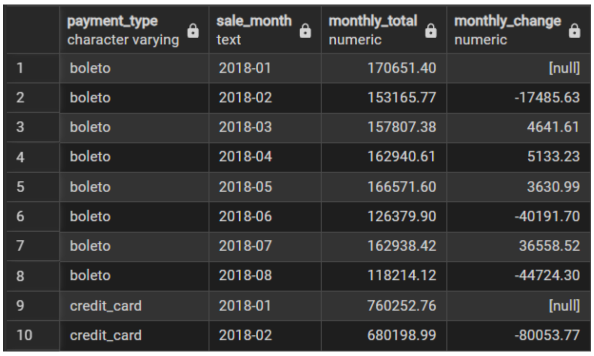
	product_id character varying	sale_month text	total_sales numeric
62089	ffb2a783fe9381154e13bb3457651996	2017-02	249.9
62090	ffb2e8c1ddc7c3e590d2bc4c91de53e1	2018-05	48.99
62091	ffb530bd30bcc1bd903e4da723faa5e6	2018-01	39.97
62092	ffb530bd30bcc1bd903e4da723faa5e6	2018-03	83.94
62093	ffb530bd30bcc1bd903e4da723faa5e6	2018-06	129.91
62094	ffb64e34a37740dafb6c88f1abd1fa61	2017-08	106.2
62095	ffb97eb64c6fe1baada2410288c04457	2017-10	548.97
62096	ffb97eb64c6fe1baada2410288c04457	2018-02	738.96
62097	ffb97eb64c6fe1baada2410288c04457	2018-03	1118.76
62098	ffb97eb64c6fe1baada2410288c04457	2018-04	1308.66

6) Product Lifecycle Cumulative Sales Analysis

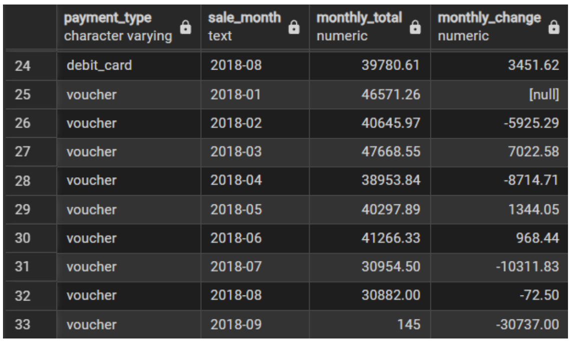
- **Analysis:** Tracks cumulative monthly sales per product to show lifecycle growth.
- **Recommendation:** Focus marketing and inventory on consistently growing products and Use discounts or bundling to boost slow or declining products.



```sql
-- 7)To understand how different payment methods affect monthly sales growth, Amazon wants to compute the total sales for
--each payment method and calculate the month-over-month growth rate for the past year (year 2018). Write query to first
--calculate total monthly sales for each payment method, then compute the percentage change from the previous month.

-- Output: payment_type, sale_month, monthly_total, monthly_change.

WITH monthly_sales AS (
    SELECT
        p.payment_type,
        TO_CHAR(o.order_purchas_timestamp, 'YYYY-MM') AS sale_month,
        SUM(oi.price) AS monthly_total
    FROM amazon_brazil.payments p
    JOIN amazon_brazil.order_items oi ON p.order_id = oi.order_id
    JOIN amazon_brazil.orders o ON oi.order_id = o.order_id
    WHERE EXTRACT(YEAR FROM o.order_purchas_timestamp) = 2018
    GROUP BY p.payment_type, TO_CHAR(o.order_purchas_timestamp, 'YYYY-MM')
)
SELECT
    payment_type,
    sale_month,
    monthly_total,
    (monthly_total - LAG(monthly_total) OVER (PARTITION BY payment_type ORDER BY sale_month)) AS monthly_change
FROM monthly_sales
ORDER BY payment_type, sale_month;
```



	payment_type character varying	sale_month text	monthly_total numeric	monthly_change numeric
1	boleto	2018-01	170651.40	[null]
2	boleto	2018-02	153165.77	-17485.63
3	boleto	2018-03	157807.38	4641.61
4	boleto	2018-04	162940.61	5133.23
5	boleto	2018-05	166571.60	3630.99
6	boleto	2018-06	126379.90	-40191.70
7	boleto	2018-07	162938.42	36558.52
8	boleto	2018-08	118214.12	-44724.30
9	credit_card	2018-01	760252.76	[null]
10	credit_card	2018-02	680198.99	-80053.77



	payment_type character varying	sale_month text	monthly_total numeric	monthly_change numeric
24	debit_card	2018-08	39780.61	3451.62
25	voucher	2018-01	46571.26	[null]
26	voucher	2018-02	40645.97	-5925.29
27	voucher	2018-03	47668.55	7022.58
28	voucher	2018-04	38953.84	-8714.71
29	voucher	2018-05	40297.89	1344.05
30	voucher	2018-06	41266.33	968.44
31	voucher	2018-07	30954.50	-10311.83
32	voucher	2018-08	30882.00	-72.50
33	voucher	2018-09	145	-30737.00

7) Payment Method Growth Analysis

- **Analysis:** Query highlights monthly sales and growth trends for each payment type in 2018.
- **Recommendation:** Include % change for deeper insight and ensure months are date-sorted for accurate trend analysis.