E-commerce Business Analytics KG

Analyst: Gabriela
Email: gabrielaalves0108@gmail.com
LinkedIn: /in/gabrielaalves
Date of Creation: 2026-04-22

Table of Contents

Platform Setup
- Installing the required Library
- Importing Packages
- Google Colab User Authentication for Drive, Cloud Storage
- GCP Integration: BigQuery - Colab
- Querying the BigQuery Ecommerce database

Section 1: Exploratory Analysis
- Query 1.1: Information Schema
- Query 1.2: Calculating Duration and Time Range of Orders
- Query 1.3: Unique Order Statuses

Section 2: In-depth Exploration
- Query 2.1: Order Value Trend
- Query 2.2: Monthly Sales Trend

Section 3: Evolution of E-commerce
- Query 3.1: Month-on-Month Orders Placed by Items Analysis
- Query 3.2: Customers Distribution across all the Regions

Section 4: Impact on Economy
- Query 4.1: Percentage Increase in Order Costs from 2017 to 2018 (January to August)
- Query 4.2: Analyzing the Money Movement of E-Commerce on Last Years Holidays, Freight, and Taxes

Section 5: Analysis based on Sales, Freight and Delivery time.
- Query 5.1: Analyzing Order Delivery Time and Estimation Accuracy
- Query 5.2: Top 5 Highest Avg Freight Value
- Query 5.3: Top 5 Lowest Avg Freight Value
- Query 5.4: Top 5 Highest Avg Delivery Time
- Query 5.5: Top 5 Lowest Avg Delivery Time
- Query 5.6: Identifying Top 5 States with Faster than Estimated Order Deliveries

Section 6: Analysis based on the Payments
- Query 6.1: Month-on-Month Analysis of Orders Placed by Payment Types
- Query 6.2: Analysis of Orders Based on Payment Installments Field

Section 7: RFM Analysis.
- Query 7.1: RFM Scores
- Query 7.2: Customer Segmentation

Section 8: Strategizing the Opportunity For Market Capture
- Recommendations

Introduction

Target, a globally renowned brand, has expanded its operations to Brazil, offering a unique shopping experience to its customers. With an extensive dataset covering 100,000 orders placed between 2017 and 2018, there is an opportunity to gain valuable insights into Target's operations in Brazil. This analysis aims to provide actionable recommendations to enhance order processing, pricing strategies, payment and shipping efficiency, and overall customer satisfaction.

Objective

The primary objective of this analysis is to provide Target with actionable insights derived from the dataset, enabling informed decision-making to enhance operational efficiency, customer satisfaction, and overall business performance in the Brazilian market.

Tech Stack

- Google BigQuery for querying and opening large datasets.
- VSC Microsoft SQL for formatting SQL queries in Colab environment.
- Vega-Lite/Altair Tableau for dashboard operations.
- Data Visualization
  - Google Colab
  - Tableau Public

Installing the required Library

```
!pip install sqlalchemy_bigquery
```

```
!pip install graphviz
```

Importing Packages

```
%%capture --no-stdout
!pip install -U -q google.cloud.bigquery
!pip install -U -q google.cloud.bigquery[ipy]
!pip install graphviz
```

Google Colab User Authentication for Drive, Cloud Storage

This code authenticates a user in Google Colab, enabling access to Google services like Drive, Cloud Storage, and other resources.

```
!rm -f google-cloud-sdk.tar.gz
!rm -rf /usr/local/gcloud /usr/bin/gcloud
!rm -rf ~/.config/gcloud
```

GCP Integration: BigQuery - Colab

In Colab, users can take advantage of the **google.colab.data_table** package to display large pandas dataframes interactively.
Additionally, accessing the BigQuery datasets is made possible by using the `%%bigquery` command, enabled within the Google Cloud Platform environment. This enables users to retrieve data through BigQuery SQL queries directly from their Colab notebook.

```
!rm -f /usr/bin/gcloud
```

```
%%bigquery --project=ecommerce-analytics-kg-2026 --verbose --dry_run
SELECT CURRENT_DATE() AS today_date
```

Querying the BigQuery Ecommerce Database

Section 1: Exploratory Analysis

Query 1.1 : Information Schema

INFORMATION_SCHEMA provides metadata about database objects and enables schema validation. It helps in understanding the database structure, particularly with externally hosted tables using primary and foreign keys. It also provides insights into each column, including its data type, "is_nullable" characteristics, and more.

```
%%bigquery --project=ecommerce-analytics-kg-2026 --verbose --dry_run
SELECT table_catalog, table_schema, table_name, column_name, ordinal_position, is_nullable, data_type
FROM `olist_ecommerce_dataset.INFORMATION_SCHEMA.COLUMNS`
ORDER BY table_name, ordinal_position;
```

```
%%bigquery --project=ecommerce-analytics-kg-2026 --verbose
SELECT table_catalog, table_schema, table_name, column_name, ordinal_position, is_nullable, data_type
FROM `olist_ecommerce_dataset.INFORMATION_SCHEMA.COLUMNS`
ORDER BY table_name, ordinal_position;
```

	table_catalog	table_schema	table_name	column_name	ordinal_position	is_nullable	data_type
1	ecommerce-analytics-kg-2026	olist_ecommerce_dataset	customers	customer_id	1	YES	STRING
2	ecommerce-analytics-kg-2026	olist_ecommerce_dataset	customers	customer_unique_id	2	YES	STRING
3	ecommerce-analytics-kg-2026	olist_ecommerce_dataset	customers	customer_zip_code_prefix	3	YES	STRING
4	ecommerce-analytics-kg-2026	olist_ecommerce_dataset	customers	customer_city	4	YES	STRING
5	ecommerce-analytics-kg-2026	olist_ecommerce_dataset	customers	customer_state	5	YES	STRING
6	ecommerce-analytics-kg-2026	olist_ecommerce_dataset	geolocation	geolocation_zip_code_prefix	1	YES	INT64
7	ecommerce-analytics-kg-2026	olist_ecommerce_dataset	geolocation	geolocation_lat	2	YES	FLOAT64
8	ecommerce-analytics-kg-2026	olist_ecommerce_dataset	geolocation	geolocation_lng	3	YES	FLOAT64
9	ecommerce-analytics-kg-2026	olist_ecommerce_dataset	geolocation	geolocation_city	4	YES	STRING
10	ecommerce-analytics-kg-2026	olist_ecommerce_dataset	geolocation	geolocation_state	5	YES	STRING
11	ecommerce-analytics-kg-2026	olist_ecommerce_dataset	product_category_name_translation	product_category_name	1	YES	STRING
12	ecommerce-analytics-kg-2026	olist_ecommerce_dataset	product_category_name_translation	product_category_name_english	2	YES	STRING
13	ecommerce-analytics-kg-2026	olist_ecommerce_dataset	orders	order_id	1	YES	STRING
14	ecommerce-analytics-kg-2026	olist_ecommerce_dataset	orders	customer_id	2	YES	STRING
15	ecommerce-analytics-kg-2026	olist_ecommerce_dataset	orders	order_status	3	YES	STRING
16	ecommerce-analytics-kg-2026	olist_ecommerce_dataset	orders	order_purchase_timestamp	4	YES	TIMESTAMP
17	ecommerce-analytics-kg-2026	olist_ecommerce_dataset	orders	order_approved_at	5	YES	TIMESTAMP
18	ecommerce-analytics-kg-2026	olist_ecommerce_dataset	orders	order_delivered_carrier_date	6	YES	TIMESTAMP
19	ecommerce-analytics-kg-2026	olist_ecommerce_dataset	orders	order_delivered_customer_date	7	YES	TIMESTAMP
20	ecommerce-analytics-kg-2026	olist_ecommerce_dataset	orders	order_estimated_delivery_date	8	YES	TIMESTAMP
21	ecommerce-analytics-kg-2026	olist_ecommerce_dataset	order_items	order_id	1	YES	STRING
22	ecommerce-analytics-kg-2026	olist_ecommerce_dataset	order_items	order_item_id	2	YES	INT64
23	ecommerce-analytics-kg-2026	olist_ecommerce_dataset	order_items	product_id	3	YES	STRING
24	ecommerce-analytics-kg-2026	olist_ecommerce_dataset	order_items	seller_id	4	YES	STRING
25	ecommerce-analytics-kg-2026	olist_ecommerce_dataset	order_items	shipping_limit_date	5	YES	TIMESTAMP
26	ecommerce-analytics-kg-2026	olist_ecommerce_dataset	order_items	price	6	YES	FLOAT64
27	ecommerce-analytics-kg-2026	olist_ecommerce_dataset	order_items	freight_value	7	YES	FLOAT64
28	ecommerce-analytics-kg-2026	olist_ecommerce_dataset	order_reviews	review_id	1	YES	STRING
29	ecommerce-analytics-kg-2026	olist_ecommerce_dataset	order_reviews	order_id	2	YES	STRING
30	ecommerce-analytics-kg-2026	olist_ecommerce_dataset	order_reviews	review_score	3	YES	INT64
31	ecommerce-analytics-kg-2026	olist_ecommerce_dataset	order_reviews	review_comment_title	4	YES	STRING

Rows per page: 10 | 1 – 10 of 31 | Page 1 of 3

Next steps: Generate code with ✨ | View recommended plots

BigQuery Schema Visualization Script

An **Entity-Relationship Diagram (ERD)** visually represents entities (objects), attributes (properties), and relationships in a database, aiding in understanding data structures and their interconnections within a system.

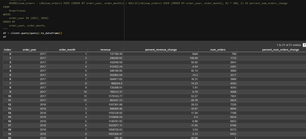
This code generates an Entity-Relationship Diagram (ERD) from a BigQuery ecommerce sales analysis. It groups the retrieved schema by table name, aggregates column names and data types, then creates a graph using the **Graphviz library**. Edges (lines) visually illustrate the relationships between tables and their columns. It also saves the ERD as "bq_ecommerce_erd.pdf" for easy sharing and review.

```
%%bigquery --project=ecommerce-analytics-kg-2026 --verbose
SELECT table_name, column_name, data_type
FROM `olist_ecommerce_dataset.INFORMATION_SCHEMA.COLUMNS`
ORDER BY table_name, ordinal_position;
```

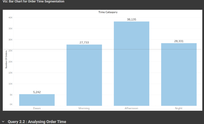
```
%%bigquery --project=ecommerce-analytics-kg-2026 --verbose
SELECT table_name, column_name, data_type
FROM `olist_ecommerce_dataset.INFORMATION_SCHEMA.COLUMNS`
ORDER BY table_name, ordinal_position;
```

```
import collections, collections.abc, pandas, graphviz, sys
df = _.sort_values(['table_name', 'ordinal_position'])

dot = graphviz.Graph(graph_attr={'rankdir': 'TB', 'splines': 'ortho', 'nodesep': '0.6', 'ranksep': '1.0'},
                     node_attr={'shape': 'box', 'fontsize': '10'})

for table_name, columns in df.groupby('table_name'):
    dot.node(table_name, f'{table_name}\n{"-"*20}\' + '\n'.join(f'{row.column_name}: {row.data_type}' for _, row in columns.iterrows()))

# Add edges between tables
tables = df['table_name'].unique()
for i, t1 in enumerate(tables):
    for t2 in tables[i+1:]:
        common = set(df[df['table_name'] == t1]['column_name']) & set(df[df['table_name'] == t2]['column_name'])
        for c in common:
            dot.edge(t1, t2, label=c)

dot.render('bq_ecommerce_erd', format='png', cleanup=True)

print('Saved: bq_ecommerce_erd.pdf')
```

```
Saved: bq_ecommerce_erd.pdf
```


customers
customer_id: STRING
customer_unique_id: STRING
customer_zip_code_prefix: STRING
customer_city: STRING
customer_state: STRING
geolocation
geolocation_zip_code_prefix: INT64
geolocation_lat: FLOAT64
geolocation_lng: FLOAT64
geolocation_city: STRING
geolocation_state: STRING
product_category_name_translation
product_category_name: STRING
product_category_name_english: STRING
orders
order_id: STRING
customer_id: STRING
order_status: STRING
order_purchase_timestamp: TIMESTAMP
order_approved_at: TIMESTAMP
order_delivered_carrier_date: TIMESTAMP
order_delivered_customer_date: TIMESTAMP
order_estimated_delivery_date: TIMESTAMP
order_items
order_id: STRING
order_item_id: INT64
product_id: STRING
seller_id: STRING
shipping_limit_date: TIMESTAMP
price: FLOAT64
freight_value: FLOAT64
order_reviews
review_id: STRING
order_id: STRING
review_score: INT64
review_comment_title: STRING
customer_id
customer_id
customer_id

01. Bar Chart for Order Time Segmentation



Query 2.2 : Analysing Order Time

```
with dtime AS(
select
extract(hour from order_purchase_timestamp) AS hour_of_day
FROM olist_orders
)
SELECT
hour_of_day,
count(hour_of_day) AS order_count
FROM dtime
GROUP BY hour_of_day
ORDER BY hour_of_day;
select count(order_id) from olist_orders;
```

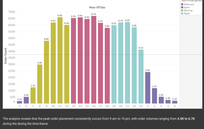
1 to 24 of 24 rows | Export | Explore

Index	hour_of_day	order_count
0	0	2763
1	1	1328
2	2	548
3	3	305
4	4	193
5	5	385
6	6	1164
7	7	2904
8	8	4329
9	9	5773
10	10	6843
11	11	7211
12	12	7039
13	13	7057
14	14	7212
15	15	7135
16	16	6571
17	17	5274
18	18	4232
19	19	4331
20	20	4746
21	21	4811
22	22	4482
23	23	3334

Rows per page: 100 1 - 24 of 24

Next steps: Generate code with ✓ View recommended plots

03. Bar Chart for Hourly Order Volume Distribution



The analysis reveals that the peak order placement consistently occurs from 9 am to 10 pm, with order volumes ranging from **4.34 to 6.78** lakhs the during the time frame.

Inference: The distribution illustrates the variations in order placements throughout the day, providing insights into customer behaviour and preferences during different time categories. The morning and Afternoon periods appear to be the busiest, while the Evening and Night categories experience lower order activity.

Section 3: Evolution of E-commerce

Monthly Orders Placed in Each State To understand the evolution of e-commerce orders in Brazil, we first examine the month-on-month number of orders placed in each state. This analysis provides insights into the underlying trends and potential variations across different regions.

Query 3.1 : Month-on-Month Orders Placed by State Analysis

```
with query1 AS(
SELECT
EXTRACT(YEAR FROM order_purchase_timestamp) AS order_year,
EXTRACT(MONTH FROM order_purchase_timestamp) AS order_month,
c.[illegible],
count(o.order_id) AS order_count
FROM olist_orders AS o JOIN olist_customers
AS c
ON o.customer_id = c.customer_id
GROUP BY
order_year, order_month, c.[illegible]
ORDER BY
order_year, order_month, c.[illegible]
),
query2 AS(
SELECT
order_year, order_month, c.[illegible],
order_count,
LAG(order_count, 1, 0) OVER(PARTITION BY c.[illegible]
ORDER BY order_year, order_month) AS prev_month_orders
FROM query1 AS c
),
query3 AS(
SELECT
order_year, order_month, c.[illegible],
order_count,
prev_month_orders,
CASE WHEN prev_month_orders = 0 THEN NULL
ELSE ROUND(CAST(order_count - prev_month_orders AS FLOAT) / prev_month_orders, 2)
END AS mom_growth_rate
FROM query2 AS c
)
SELECT
*
FROM query3 AS c
ORDER BY
c.[illegible], order_year, order_month;
```

1 to 100 of 100 rows | Export | Explore

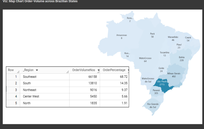
Index	order_year	order_month	c.[illegible]	order_count	prev_month_orders	mom_growth_rate
0	2016	9	Sao Paulo	1	0	NULL
1	2016	10	Rio de Janeiro	1	0	NULL
2	2016	10	Sao Paulo	2	1	1.00
3	2016	12	Sao Paulo	1	2	-0.50
4	2017	1	Distrito Federal	1	0	NULL
5	2017	1	Espirito Santo	1	0	NULL
6	2017	1	Minas Gerais	1	0	NULL
7	2017	1	Parana	1	0	NULL
8	2017	1	Rio de Janeiro	2	0	NULL
9	2017	1	Sao Paulo	34	0	NULL
10	2017	2	Alagoas	1	0	NULL
11	2017	2	Distrito Federal	1	1	0.00
12	2017	2	Espirito Santo	1	1	0.00
13	2017	2	Goias	2	0	NULL
14	2017	2	Mato Grosso	1	0	NULL
15	2017	2	Minas Gerais	5	1	4.00
16	2017	2	Parana	2	1	1.00
17	2017	2	Rio de Janeiro	7	2	2.50
18	2017	2	Rio Grande do Norte	1	0	NULL
19	2017	2	Sao Paulo	79	34	1.32
20	2017	3	Alagoas	2	1	1.00
21	2017	3	Ceara	3	0	NULL
22	2017	3	Distrito Federal	4	1	3.00
23	2017	3	Espirito Santo	9	1	8.00
24	2017	3	Goias	2	2	0.00
25	2017	3	Mato Grosso	1	1	0.00
26	2017	3	Minas Gerais	11	5	1.20
27	2017	3	Parana	5	2	1.50
28	2017	3	Pernambuco	1	0	NULL
29	2017	3	Rio de Janeiro	16	7	1.29
30	2017	3	Rio Grande do Norte	1	1	0.00
31	2017	3	Sao Paulo	154	79	0.95
32	2017	4	Alagoas	2	2	0.00
33	2017	4	Ceara	11	3	2.67
34	2017	4	Distrito Federal	5	4	0.25
35	2017	4	Espirito Santo	9	9	0.00
36	2017	4	Goias	19	2	8.50
37	2017	4	Mato Grosso	14	1	13.00
38	2017	4	Minas Gerais	59	11	4.36
39	2017	4	Parana	30	5	4.80
40	2017	4	Pernambuco	3	1	2.00
41	2017	4	Rio de Janeiro	64	16	3.00
42	2017	4	Rio Grande do Norte	3	1	2.00
43	2017	4	Sao Paulo	392	154	1.55
44	2017	5	Alagoas	3	2	0.50
45	2017	5	Ceara	20	11	0.82
46	2017	5	Distrito Federal	12	5	1.40
47	2017	5	Espirito Santo	15	9	0.67
48	2017	5	Goias	32	19	0.68
49	2017	5	Mato Grosso	14	14	0.00
50	2017	5	Minas Gerais	75	59	0.27
51	2017	5	Parana	43	30	0.43
52	2017	5	Pernambuco	11	3	2.67
53	2017	5	Rio de Janeiro	85	64	0.33
54	2017	5	Rio Grande do Norte	5	3	0.67
55	2017	5	Sao Paulo	692	392	0.77
56	2017	6	Alagoas	3	3	0.00
57	2017	6	Ceara	32	20	0.60
58	2017	6	Distrito Federal	13	12	0.08
59	2017	6	Espirito Santo	17	15	0.13
60	2017	6	Goias	35	32	0.09
61	2017	6	Mato Grosso	16	14	0.14
62	2017	6	Minas Gerais	84	75	0.12
63	2017	6	Parana	39	43	-0.09
64	2017	6	Pernambuco	18	11	0.64
65	2017	6	Rio de Janeiro	85	85	0.00
66	2017	6	Rio Grande do Norte	5	5	0.00
67	2017	6	Sao Paulo	709	692	0.02
68	2017	7	Alagoas	4	3	0.33
69	2017	7	Ceara	35	32	0.09
70	2017	7	Distrito Federal	15	13	0.15
71	2017	7	Espirito Santo	21	17	0.24
72	2017	7	Goias	40	35	0.14
73	2017	7	Mato Grosso	14	16	-0.13
74	2017	7	Minas Gerais	92	84	0.10
75	2017	7	Parana	44	39	0.13
76	2017	7	Pernambuco	20	18	0.11
77	2017	7	Rio de Janeiro	102	85	0.20
78	2017	7	Rio Grande do Norte	5	5	0.00
79	2017	7	Sao Paulo	821	709	0.16
80	2017	8	Alagoas	3	4	-0.25
81	2017	8	Ceara	39	35	0.11
82	2017	8	Distrito Federal	20	15	0.33
83	2017	8	Espirito Santo	29	21	0.38
84	2017	8	Goias	37	40	-0.07
85	2017	8	Mato Grosso	14	14	0.00
86	2017	8	Minas Gerais	84	92	-0.09
87	2017	8	Parana	36	44	-0.18
88	2017	8	Pernambuco	32	20	0.60
89	2017	8	Rio de Janeiro	100	102	-0.02
90	2017	8	Rio Grande do Norte	5	5	0.00
91	2017	8	Sao Paulo	870	821	0.06
92	2017	9	Alagoas	4	3	0.33
93	2017	9	Ceara	54	39	0.38
94	2017	9	Distrito Federal	16	20	-0.20
95	2017	9	Espirito Santo	24	29	-0.17
96	2017	9	Goias	32	37	-0.14
97	2017	9	Mato Grosso	19	14	0.36
98	2017	9	Minas Gerais	76	84	-0.10
99	2017	9	Parana	54	36	0.50

Rows per page: 100 1 - 100 of 100

Next steps: Generate code with ✓ View recommended plots

04. Stacked Bar of Chart Order Volume by Region Over Time

05. Map Chart Order Volume across Brazilian States



Row	Region	OrderVolumeMax	OrderPercentage
1	Southeast	60738	68.73
2	South	13810	15.63
3	Northeast	9016	10.20
4	Center-West	5400	6.08
5	North	1920	2.17

Inference:

- **The South-Core region** dominates with Sao Paulo, Rio de Janeiro, and Minas Gerais contributing to 68.7% of the total order volume.
- **Followed by the South region**, led by the Greater Sao Paulo Finland, and Santa Catarina, accounting for 15.63%.
- **Other regions**, including North-East, Center-West & North, contribute proportionally, while Northern states constitute the remaining small market share.

Query 3.2 : Customers Distribution across all the States

Customer Distribution Across States Additionally, we explore how customers are distributed across all the states in Brazil. This analysis helps in understanding the geographical distribution of customers and identifying regions with higher customer concentration.

By combining this with the order volume trends across states, we can gain valuable insights into the e-commerce landscape in Brazil and make informed decisions to optimize business strategies.

```
with c AS(
select
customer_state,
c.customer_id,
count(o.order_id)
FROM olist_customers AS c JOIN olist_orders
AS o
ON c.customer_id = o.customer_id
GROUP BY customer_state, c.customer_id
),
SELECT
c.customer_state,
count(c.customer_id) AS Number_of_customers,
SUM(c.Number_of_customers) OVER(ORDER BY Total_Customers
DESC
ROWS
BETWEEN UNBOUNDED PRECEDING AND CURRENT ROW) AS Running_Total,
SUM(c.Number_of_customers) OVER(ORDER BY Total_Customers
DESC
ROWS
BETWEEN UNBOUNDED PRECEDING AND CURRENT ROW)/
(SELECT SUM(Number_of_customers) FROM
(SELECT
c.customer_id,
count(c.customer_id) AS Number_of_customers
FROM c
GROUP BY c.customer_id
)
AS
c
)
AS Running%Total
FROM c
GROUP BY c.customer_state
ORDER BY Total_Customers
DESC;
```

1 to 27 of 27 rows | Export | Explore

Index	customer_state	city_name	region	Number_of_customers	Total_Customers	Running Total	Running%Total
0	SP	Sao Paulo	Southeast	41746	41746	41746	0.42
1	RJ	Rio de Janeiro	Southeast	12852	54598	54598	0.55
2	MG	Belo Horizonte	Southeast	11635	66233	66233	0.67
3	RS	Porto Alegre	South	5466	71699	71699	0.72
4	PR	Curitiba	South	5048	76747	76747	0.77
5	SC	Florianopolis	South	3637	80384	80384	0.81
6	BA	Salvador	Northeast	3380	83764	83764	0.84
7	DF	Brasilia	Center-West	2140	85904	85904	0.86
8	ES	Vitoria	Southeast	2033	87937	87937	0.89
9	GO	Goiania	Center-West	2020	89957	89957	0.90
10	PE	Recife	Northeast	1652	91609	91609	0.92
11	CE	Fortaleza	Northeast	1336	92945	92945	0.93
12	PA	Belem	Norte	975	93920	93920	0.94
13	MT	Cuiaba	Center-West	907	94827	94827	0.95
14	MA	Sao Luis	Northeast	747	95574	95574	0.96
15	MS	Campo Grande	Center-West	693	96267	96267	0.96
16	PB	Joao Pessoa	Northeast	536	96803	96803	0.97
17	RN	Natal	Northeast	485	97288	97288	0.97
18	AL	Maceio	Northeast	413	97701	97701	0.98
19	PI	Teresina	Northeast	411	98112	98112	0.98
20	SE	Aracaju	Northeast	350	98462	98462	0.98
21	TO	Palmas	Norte	280	98742	98742	0.99
22	RO	Porto Velho	Norte	246	98988	98988	0.99
23	AC	Rio Branco	Norte	81	99069	99069	0.99
24	AM	Manaus	Norte	148	99217	99217	0.99
25	AP	Macapa	Norte	68	99285	99285	0.99
26	RR	Boa Vista	Norte	46	99331	99331	0.99

Rows per page: 100 1 - 27 of 27

Next steps: Generate code with ✓ View recommended plots

06. RS/SS Rate Visualized in Map Chart Order Volume across Brazilian States

Inference:

The top 6 states in the **Southern and Southeastern** regions contribute around 80% of the customer base, while the remaining **20%** comes from 21 states in the **North and West** regions. This highlights the significance of focusing marketing and sales efforts on these key states to effectively reach the majority of customers.

Section 4: Impact on Economy

Analyze the money movement by e-commerce by looking at order prices, freight, and others.

- 1. Calculate the **total & average value of order** from year 2017 to 2018 (include month's between due to Aug only).
  - Use the **"payment_value"** column in the **payments** table to get the cost of orders.

Query 4.1 : Percentage Increase in Order Costs from 2017 to 2018 (January to August)

```
with query1 AS(
SELECT
EXTRACT(YEAR FROM o.order_purchase_ts) AS order_year,
SUM(CASE WHEN o.order_purchase_ts IS NOT NULL THEN
p.payment_value ELSE 0 END) AS total_order_cost,
SUM(CASE WHEN EXTRACT(YEAR FROM o.order_purchase_ts) = 2017 THEN p.payment_value ELSE 0 END) AS total_order_cost_2017,
SUM(CASE WHEN EXTRACT(YEAR FROM o.order_purchase_ts) = 2018 THEN p.payment_value ELSE 0 END) AS total_order_cost_2018
FROM
olist_orders AS o
JOIN
olist_payments AS p ON o.order_id = p.order_id
GROUP BY
order_year
HAVING
EXTRACT(YEAR FROM o.order_purchase_ts) IN (2017, 2018)
AND EXTRACT(MONTH FROM o.order_purchase_ts) BETWEEN 1 AND 8
),
SELECT
order_year, total_order_cost,
total_order_cost_2017, total_order_cost_2018,
ROUND((total_order_cost_2018 - total_order_cost_2017) / total_order_cost_2017) * 100, 2) AS percentage_difference
FROM
query1
ORDER BY
order_year
LIMIT 2;
```

1 to 2 of 2 rows | Export | Explore

Index	order_year	total_order_cost	total_order_cost_2017	total_order_cost_2018	percentage_difference
0	2017	5.7246E7	5.7246E7	7.8627E7	37.3
1	2018	8.3648E7	3.8E7	8.3648E7	120.2

Rows per page: 100 1 - 2 of 2

Next steps: Generate code with ✓ View recommended plots

08. Bar Chart 2017-2018 Order Value in Millions

09. 2017-2018 Grouped Bar Chart for Cost of Order Value in Millions

The cost of orders in January 2018 significantly increased by 120% compared to January 2017. This suggests a substantial surge in spending during this month.

The YoY percentage difference has too decreased from 100% to 37% in 8 months (upon rapidly, this suggests a significant reduction in the rate of growth. Market Saturation, or change in customer behaviour are some potential reasons for this decline.

Query 4.2 : Analyzing the Money movement of E-commerce w.r.t order-prices, Freight, and Others.

- 2. Calculate the **Total & Average value of order price** for each state.
- 3. Calculate the **Total & Average value of order freight** for each state.

```
with query1 AS(
SELECT
c.[illegible],
SUM(p.[illegible]) AS Total_Price,
AVG(p.[illegible]) AS AVG_Price,
SUM(o.[illegible]) AS Total_Freight,
AVG(o.[illegible]) AS AVG_Freight
FROM
olist_orders AS o
JOIN
olist_customers AS c ON o.customer_id = c.customer_id
JOIN
olist_order_items AS p ON o.[illegible] = p.[illegible]
GROUP BY
c.[illegible]
ORDER BY
c.[illegible]
),
SELECT
*
FROM query1 AS c
ORDER BY
c.[illegible];
```

1 to 27 of 27 rows | Export | Explore

Index	[illegible]	Total_Price	AVG_Price	Total_Freight	AVG_Freight
0	AC	19680.7	334.58	3686.75	75.8
1	AL	80314.81	180.89	15914.59	35.84
2	AM	22357.09	135.98	3686.96	22.43
3	AP	13474.3	164.78	2731.07	33.4
4	BA	511349.99	134.59	100156.68	26.37
5	CE	227254.71	153.76	48351.59	32.72
6	DF	302603.94	125.71	50625.5	21.03
7	ES	275037.31	121.91	49091.75	21.75
8	GO	294591.95	126.27	53114.98	22.77
9	MA	119648.22	145.85	31523.77	38.42
10	MG	1585308.03	120.75	270853.46	20.63
11	MS	324051.21	137.28	53482.96	22.66
12	MT	340086.42	156.74	73404.9	33.82
13	PA	178947.81	165.7	35832.99	33.18
14	PB	115268.08	191.82	25719.73	42.8
15	PE	592882.2	145.23	132451.64	32.44
16	PI	77548.27	170.8	18884.14	41.59
17	PR	619626.5	119.85	117851.68	22.8
18	RJ	1824092.67	158.24	293805.8	25.49
19	RN	96475.38	191.88	19860.75	39.5
20	RO	46140.64	165.97	11417.8	41.07
21	RR	8404.07	152.8	2235.19	40.64
22	RS	890261.63	135.8	217111.69	33.12
23	SC	589497.07	125.39	141444.48	30.08
24	SE	90315.05	153.25	27067.23	45.92
25	SP	5202955.05	137.46	718723.07	18.98
26	TO	49548.19	189.08	11732.99	44.77

Rows per page: 100 1 - 27 of 27

Next steps: Generate code with ✓ View recommended plots

1. First the no. of days taken to deliver each order from the order's purchase date to delivery time.
2. Also find out the difference in days between the estimated & actual delivery date of an order.
3. Find out the average delay. Some orders are delivered before the estimated delivery date. This is calculated by actual delivery date using the given formula:

```
avg(date_part('day', order_delivered_customer_date - order_estimated_delivery_date))
```

Section 5: Analysis based on Sales, Freight and Delivery time.

```
order_id,
order_purchase_timestamp,
order_approved_at,
order_delivered_carrier_date,
DATE_DIFF(order_delivered_customer_date, order_purchase_timestamp, DAY) AS Actual_Delivery_Time,
DATE_DIFF(order_estimated_delivery_date, order_purchase_timestamp, DAY) AS Estimated_Delivery_Time,
DATE_DIFF(order_delivered_customer_date, order_estimated_delivery_date, DAY) AS Delivery_Difference,
CASE
WHEN order_status = 'delivered'
THEN
'Delivered'
ELSE
'Not Delivered'
END AS Delivery_Status
FROM
`olist_order_dataset`
WHERE
order_status = 'delivered'
LIMIT
1000;
```

Running total number of rows 99441 scanned so far, 1000. Running bytes in quota: 8Kb.

	order_id	order_purchase_timestamp	order_approved_at	order_delivered_carrier_date	Actual_Delivery_Time	Estimated_Delivery_Time	Delivery_Difference	Delivery_Status
1	1a8f7f2f7a5a4a6d9f4e4e8d7f4d3d	2017-12-12 15:30:41+00:00	2017-12-14 11:10:14+00:00	2017-12-14 14:30:00+00:00	29	45	-16	Delivered
2	6b67c54cd7f7a4a4d9e7f4f4c7f7d7	2017-12-20 11:19:55+00:00	2017-12-20 11:30:14+00:00	2017-12-21 14:30:00+00:00	31	45	-14	Delivered
3	18a4f4f4d7f4f7f4d7f4d7f4d7f4d7d4	2018-01-12 15:00:18+00:00	2018-01-12 15:30:14+00:00	2018-01-18 14:30:00+00:00	6	29	-23	Delivered
4	4d7f4d7f4d7f4d7f4d7f4d7f4d7f4d7d	2018-01-12 20:48:31+00:00	2018-01-12 21:05:14+00:00	2018-01-18 14:30:00+00:00	6	29	-23	Delivered
5	7d7f4d7f4d7f4d7f4d7f4d7f4d7f4d7d	2018-01-16 12:36:48+00:00	2018-01-16 12:45:14+00:00	2018-01-25 14:30:00+00:00	9	29	-20	Delivered
6	8d7f4d7f4d7f4d7f4d7f4d7f4d7f4d7d	2018-01-16 21:25:21+00:00	2018-01-17 02:30:14+00:00	2018-01-25 14:30:00+00:00	9	29	-20	Delivered
7	9d7f4d7f4d7f4d7f4d7f4d7f4d7f4d7d	2018-01-18 12:48:30+00:00	2018-01-18 12:55:14+00:00	2018-01-25 14:30:00+00:00	7	29	-22	Delivered
8	1c1c1c1c1c1c1c1c1c1c1c1c1c1c1c1c	2017-05-19 10:17:43+00:00	2017-05-19 10:35:14+00:00	2017-06-20 20:30:00+00:00	31	29	2	Delivered
9	2c2c2c2c2c2c2c2c2c2c2c2c2c2c2c2c	2017-06-13 08:34:27+00:00	2017-06-13 08:45:14+00:00	2017-06-23 20:30:00+00:00	10	29	-13	Delivered
10	3c3c3c3c3c3c3c3c3c3c3c3c3c3c3c3c	2017-06-26 11:57:44+00:00	2017-06-26 12:05:14+00:00	2017-07-05 20:30:00+00:00	9	29	-20	Delivered
11	4c4c4c4c4c4c4c4c4c4c4c4c4c4c4c4c	2018-05-10 13:47:14+00:00	2018-05-10 13:55:14+00:00	2018-05-22 20:30:00+00:00	12	29	-17	Delivered
12	5c5c5c5c5c5c5c5c5c5c5c5c5c5c5c5c	2018-05-14 15:35:12+00:00	2018-05-14 15:45:14+00:00	2018-05-22 20:30:00+00:00	8	29	-21	Delivered
13	6c6c6c6c6c6c6c6c6c6c6c6c6c6c6c6c	2018-05-16 13:10:30+00:00	2018-05-16 13:25:14+00:00	2018-05-22 20:30:00+00:00	6	29	-23	Delivered
14	7c7c7c7c7c7c7c7c7c7c7c7c7c7c7c7c	2018-05-17 15:08:01+00:00	2018-05-17 15:15:14+00:00	2018-05-22 20:30:00+00:00	5	29	-24	Delivered
15	8c8c8c8c8c8c8c8c8c8c8c8c8c8c8c8c	2018-05-18 11:36:30+00:00	2018-05-18 11:45:14+00:00	2018-05-22 20:30:00+00:00	4	29	-25	Delivered

Next steps: Generate code with ✨ View recommended plots

```
select
order_id,
order_purchase_timestamp,
order_approved_at,
order_delivered_carrier_date,
DATE_DIFF(order_delivered_customer_date, order_purchase_timestamp, DAY) AS Actual_Delivery_Time,
DATE_DIFF(order_estimated_delivery_date, order_purchase_timestamp, DAY) AS Estimated_Delivery_Time,
DATE_DIFF(order_delivered_customer_date, order_estimated_delivery_date, DAY) AS Delivery_Difference,
CASE
WHEN order_status = 'delivered'
THEN
'Delivered'
ELSE
'Not Delivered'
END AS Delivery_Status
FROM
`olist_order_dataset`
WHERE
order_status = 'delivered'
LIMIT
1000;
```

Successfully saved results to table 'bqx-1-1-0-0-0

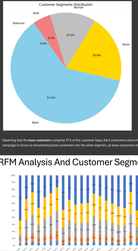
10866	20230-7551CPHCAMB7ca-TAN-PCS	Jan	2	107.5	1	5	5.96 Brown
10867	20230-7551CPHCAMB7ca-TAN-PCS	Jan	2	107.5	1	5	5.96 Brown
10868	20230-8551CPHCAMB7ca-TAN-PCS	Jan	2	203.1	1	5	5.97 Brown
10869	20230-9551CPHCAMB7ca-TAN-PCS	Jan	2	203.1	1	5	5.97 Brown
10870	20230-9551CPHCAMB7ca-TAN-PCS	Jan	1	107.5	1	5	2.98 Brown

10871 rows × 14 columns

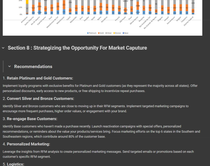
```
rfm = rfm_df.sort_values(by='Recency', ascending=True)

plt.figure(figsize=(12, 6))
sns.barplot(x='Customer_Segment', y='Recency', data=rfm, palette='Blues')
plt.title('Recency')
plt.xlabel('Customer_Segment')
plt.ylabel('Recency')
plt.xticks(rotation=45)

rfm = rfm_df.sort_values(by='Frequency', ascending=False)

plt.figure(figsize=(12, 6))
sns.barplot(x='Customer_Segment', y='Frequency', data=rfm, palette='Greens')
plt.title('Frequency')
plt.xlabel('Customer_Segment')
plt.ylabel('Frequency')
plt.xticks(rotation=45)

rfm = rfm_df.sort_values(by='Monetary', ascending=False)

plt.figure(figsize=(12, 6))
sns.barplot(x='Customer_Segment', y='Monetary', data=rfm, palette='Reds')
plt.title('Monetary')
plt.xlabel('Customer_Segment')
plt.ylabel('Monetary')
plt.xticks(rotation=45)

plt.suptitle('RFM Scores by Customer Segment', fontsize=18)
plt.tight_layout()
plt.show()
```

```
/tmp/ipykernel_1124/3844444444.py:6: FutureWarning:

Passing 'palette' without assigning 'hue' to Seaborn>v0.14.0 will be removed in v0.14.0. Assign the 'x' variable to 'hue' and set 'legend=False' for the same effect.

  sns.barplot(x='Customer_Segment', y='Recency', data=rfm, palette='Blues')
/usr/local/lib/python3.11/dist-packages/IPykernel/cp/_init.py:

Passing 'palette' without assigning 'hue' to Seaborn>v0.14.0 will be removed in v0.14.0. Assign the 'x' variable to 'hue' and set 'legend=False' for the same effect.

  sns.barplot(x='Customer_Segment', y='Frequency', data=rfm, palette='Greens')
/usr/local/lib/python3.11/dist-packages/IPykernel/cp/_init.py:

Passing 'palette' without assigning 'hue' to Seaborn>v0.14.0 will be removed in v0.14.0. Assign the 'x' variable to 'hue' and set 'legend=False' for the same effect.

  sns.barplot(x='Customer_Segment', y='Monetary', data=rfm, palette='Reds')
```

```
plt.figure(figsize=(8, 8))
plt.pie(customer_segments, labels=segment_counts.index, autopct='%1.1f%%', startangle=140, colors=['#ffd700', '#c0c0c0', '#cd7f32', '#ff6347', '#87ceeb'])
plt.title('Customer Segments Distribution')
plt.show()
```



Observing that the **base customers** comprise 57% of the customer base, the E-commerce store should position their marketing campaign to focus on transitioning base customers into the silver segment, as base customers are more susceptible to churn.



Section 8 : Strategizing the Opportunity For Market Caputure

Recommendations

1. Retain Platinum and Gold Customers:

Implement loyalty programs with exclusive benefits for Platinum and Gold customers (as they represent the majority across all states). Offer personalized discounts, early access to new products, or free shipping to incentivize repeat purchases.

2. Convert Silver and Bronze Customers:

Identify Silver and Bronze customers who are close to moving up in their RFM segments. Implement targeted marketing campaigns to encourage more frequent purchases, higher order values, or engagement with your brand.

3. Re-engage Base Customers:

Identify Base customers who haven't made a purchase recently. Launch reactivation campaigns with special offers, personalized recommendations, or reminders about the value your products/services bring. Focus marketing efforts on the top 5 states in the Southern and Northeastern regions, which collectively account 60% of the customer base.

4. Personalized Marketing:

Leverage the insights from RFM analysis to create personalized marketing messages. Send targeted emails or promotions based on each customer's specific RFM segment.

5. Logistics:

Optimize delivery processes to ensure consistent on-time delivery while minimizing late deliveries.

Cancel edit prompt | Cancel command here

Ln , Col , Spaces: 4 UTF-8 LF Cell

Ln , Col , Spaces: 4 UTF-8 LF Cell