Business Case: Target SQL

1. Import the dataset and do usual exploratory analysis steps like checking the structure & characteristics of the dataset

a) ***Data type of columns in a table***

S.No.	Table	Columns	Datatype
1	columns	customer_id	Varchar(50)
2	columns	customer_unique_id	Varchar(50)
3	columns	customer_zip_code_prefix	Int
4	columns	customer_city	Varchar(50)
5	columns	customer_state	Varchar(50)
6	sellers	seller_id	Varchar(50)
7	sellers	seller_zip_code_prefix	Varchar(50)
8	sellers	seller_city	Varchar(50)
9	sellers	seller_state	Varchar(50)
10	order_items	order_id	Varchar(50)
11	order_items	order_item_id	Varchar(50)
12	order_items	product_id	Varchar(50)
13	order_items	seller_id	Varchar(50)
14	order_items	shipping_limit_date	datetime
15	order_items	price	Float8
16	order_items	freight_value	Float8
17	geolocations	geolocation_zip_code_prefix	int
18	geolocations	geolocation_lat	Float8
19	geolocations	geolocation_lng	Float8
20	geolocations	geolocation_city	Varchar(50)
21	geolocations	geolocation_state	Varchar(50)
22	payments	order_id	Varchar(50)
23	payments	payment_sequential	Int
24	payments	payment_type	Varchar(50)
25	payments	payment_installments	Int
26	payments	payment_value	Float8
27	orders	order_id	Varchar(50)
28	orders	customer_id	Varchar(50)
29	orders	order_status	Varchar(50)
30	orders	order_purchase_timestamp	Datetime
31	orders	order_delivered_carrier_date	Datetime
32	orders	order_delivered_customer_date	Datetime
33	orders	order_estimated_delivery_date	Datetime
34	reviews	review_id	Varchar(50)
35	reviews	order_id	Varchar(50)
36	reviews	review_score	Int
37	reviews	review_comment_title	Varchar(50)
38	reviews	review_comment_message	Varchar(50)
39	reviews	review_creation_date	Datetime
40	reviews	review_answer_timestamp	Datetime

41	products	product_id	Varchar(50)
42	products	product_category_name	Varchar(50)
43	products	product_name_length	int
44	products	product_description_length	int
45	products	product_photos_qty	int
46	products	product_weight_g	int
47	products	product_length_cm	int
48	products	product_height_cm	int
49	products	product_width_cm	int

b) ***Time period for which the data is given***

The data is for a period:

2016-09-04 to 2018-11-12

c) ***Cities and States of customers ordered during the given period***

CODE:

```
SELECT DISTINCT
    customer_city
FROM target.customers
ORDER BY customer_city ASC
```

Result up to 10 rows:

Query results

SAVE RESULTS EXPLOR

JOB INFORMATION | RESULTS | JSON | EXECUTION DETAILS | EXECUTION GRAPH PREVIEW

Row	customer_city
1	abadia dos dourados
2	abadiania
3	abaete
4	abaetetuba
5	abaiara
6	abaira
7	abare
8	abatia
9	abdon batista
10	abelardo luz

Results per page: 50 1 – 50 of 4119

CODE:

```
SELECT DISTINCT
    customer_state
```

```
FROM target.customers
ORDER BY customer_state ASC
```

Result up to 10 rows:

Query results

SAVE RESULTS EXPLOR

JOB INFORMATION | RESULTS | JSON | EXECUTION DETAILS | EXECUTION GRAPH PREVIEW

Row	customer_state
1	AC
2	AL
3	AM
4	AP
5	BA
6	CE
7	DF
8	ES
9	GO
10	MA

Results per page: 50 1 – 27 of 27

2. In-depth Exploration:

a) ***Is there a growing trend on e-commerce in Brazil? How can we describe a complete scenario? Can we see some seasonality with peaks at specific months?***

CODE:

```
SELECT DISTINCT

    EXTRACT(YEAR FROM o.order_purchase_timestamp) AS year,
    EXTRACT(MONTH FROM o.order_purchase_timestamp) AS month,
    COUNT(*) as no_of_orders
FROM target.orders o
GROUP BY year,month
ORDER BY year,month
```

Result up to 10 rows:

Query results

SAVE RESULTS EXPLOI

JOB INFORMATION RESULTS JSON EXECUTION DETAILS EXECUTION GRAPH PREVIEW

Row	year	month	no_of_orders
1	2016	9	4
2	2016	10	324
3	2016	12	1
4	2017	1	800
5	2017	2	1780
6	2017	3	2682
7	2017	4	2404
8	2017	5	3700
9	2017	6	3245
10	2017	7	4026

Results per page: 50 1 – 25 of 25

Insight/Observation:

As can be seen from the data, we do see a growing trend of ecommerce in Brazil which peaks around mid-year months, i.e., 4th to 8th month. This could be attributed to several cultural and social factors. In Brazil, the period from April to August marks the fall and winter season, and during this time, people tend to spend more time at home and indoors. This means they are more likely to engage with indoor activities like online shopping. Additionally, the months of June and July include popular Brazilian holidays such as São João and Independence Day, which may drive more online shopping activity.

Recommendation:

To capitalize on this surge in sales, Target could focus on creating a strong social media strategy during this period. They could create seasonal campaigns that are focused on fall and winter products such as warm clothing, heaters, and home decor items that align with the cultural and social context of this period. For example, they could create a series of social media posts and ads featuring cozy winter outfits or home decor products that are designed to be used during the fall and winter seasons.

b) ***What time do Brazilian customers tend to buy (Dawn, Morning, Afternoon or Night)?***

NOTE: Assuming: Dawn – 00:00-06:00, Morning – 06:00-12:00, Afternoon – 12:00-18:00, Night - 18:00-23:59

CODE:

```
SELECT
    CASE
  WHEN EXTRACT(HOUR FROM CAST(order_purchase_timestamp AS TIMESTAMP)) BETWEEN 0 AND 6 T
HEN 'Dawn'
  WHEN EXTRACT(HOUR FROM CAST(order_purchase_timestamp AS TIMESTAMP)) BETWEEN 6 AND 12
THEN 'Morning'
  WHEN EXTRACT(HOUR FROM CAST(order_purchase_timestamp AS TIMESTAMP)) BETWEEN 12 AND 18
THEN 'Afternoon'
  WHEN EXTRACT(HOUR FROM CAST(order_purchase_timestamp AS TIMESTAMP)) BETWEEN 18 AND 23
THEN 'Night'
 END AS purchase_time,
        COUNT(*) AS total_purchases
        FROM
            target.orders o
        JOIN target.customers c
        ON o.customer_id = c.customer_id

        GROUP BY
            purchase_time
        ORDER BY
            total_purchases DESC;
```

Result up to 10 rows:

Query results

JOB INFORMATION | RESULTS | JSON

Row	purchase_time	total_purchases
1	Afternoon	38135
2	Night	28331
3	Morning	27733
4	Dawn	5242

<u>Insight/Observation:</u>

From the data it can be observed that the Brazilians tend to shop more during the afternoon period with the night time being a far 2nd and Morning a closer 3rd.

Recommendation:

The majority of customers are buying during the afternoon and night, so Target should optimize its marketing campaigns accordingly, targeting its advertising efforts during the afternoon and night, which are the peak times for shopping. This can include running targeted social media campaigns and offering promotions during these times.

3. Evolution of E-commerce orders in the Brazil region:

a) *Get month on month orders by states*

CODE:

```
SELECT DISTINCT
  c.customer_state,
  EXTRACT(MONTH FROM o.order_purchase_timestamp) AS month,
  COUNT(*) AS count_
FROM target.orders o
JOIN target.customers c
ON o.customer_id = c.customer_id
GROUP BY c.customer_state,month
ORDER BY c.customer_state,month
```

Result up to 10 rows:

Query results

SAVE RESULTS EXPLO

JOB INFORMATION RESULTS JSON EXECUTION DETAILS EXECUTION GRAPH PREVIEW

Row	customer_state	month	count_
1	AC	1	8
2	AC	2	6
3	AC	3	4
4	AC	4	9
5	AC	5	10
6	AC	6	7
7	AC	7	9
8	AC	8	7
9	AC	9	5
10	AC	10	6

Results per page: 50 1 – 50 of 322

Insight/Observation:

Observation of data suggests that the sale across all states surges during the mid-year months, i.e., 4th to 8th months.

Recommendation:

They could create a series of social media posts and ads featuring cozy winter outfits or home decor products that are designed to be used during the fall and winter seasons. Additionally, they could leverage the Brazilian holidays in their social media campaigns, offering exclusive discounts and promotions to customers during these periods.

b) ***Distribution of customers across the states in Brazil***

CODE:

```
SELECT
  customer_state,
  COUNT(*) AS count_of_customers
FROM target.customers
GROUP BY customer_state
ORDER BY count_of_customers DESC
```

Result up to 10 rows:

Query results

SAVE RESULTS EXPLOI

JOB INFORMATION | RESULTS | JSON | EXECUTION DETAILS | EXECUTION GRAPH PREVIEW

Row	customer_state	count_of_custon
1	SP	41746
2	RJ	12852
3	MG	11635
4	RS	5466
5	PR	5045
6	SC	3637
7	BA	3380
8	DF	2140
9	ES	2033
10	GO	2020

Results per page: 50 1 – 27 of 27

Insight/Observation:

Here, it can be seen that the top 3 states with the highest number of sales are São Paulo (SP), Rio de Janeiro (RJ), and Minas Gerais (MG), with SP having the highest number of sales by a large margin. This might be because of the fact that SP is the most populous state in Brazil and is home to many large cities and urban areas., leading to higher concentration of potential customers.

Recommendation:

To design an appropriate growth engine for further business expansion in Brazil, Target can focus on building brand awareness and loyalty in the top 3 states with the highest sales, especially São Paulo. This could involve targeted advertising and promotions on social media platforms popular in Brazil leveraging appropriate performance marketing tools, such as Facebook and Instagram ads, as well as partnering with local influencers and bloggers to showcase their products.

4. **Impact on Economy: Analyse the money movement by e-commerce by looking at order prices, freight and others.**

a) ***Get % increase in cost of orders from 2017 to 2018 (include months between Jan to Aug only) - You can use "payment_value" column in payments table***

CODE:

```
SELECT

ROUND(((SUM(CASE WHEN EXTRACT(YEAR FROM CAST(o.order_purchase_timestamp AS DATE)) = 2018
AND EXTRACT(MONTH FROM CAST(o.order_purchase_timestamp AS DATE)) BETWEEN 1 AND 8 THEN p.p
ayment_value ELSE 0 END)

- SUM(CASE WHEN EXTRACT(YEAR FROM CAST(o.order_purchase_timestamp AS DATE)) = 2017 AND EXT
RACT(MONTH FROM CAST(o.order_purchase_timestamp AS DATE)) BETWEEN 1 AND 8 THEN p.payment_v
alue ELSE 0 END))

/ (SUM(CASE WHEN EXTRACT(YEAR FROM CAST(o.order_purchase_timestamp AS DATE)) = 2017 AND EX
TRACT(MONTH FROM CAST(o.order_purchase_timestamp AS DATE)) BETWEEN 1 AND 8 THEN p.payment_
value ELSE 0 END))),2) * 100 AS increase_percentage

FROM target.orders o

JOIN target.payments p

ON o.order_id = p.order_id
```

Result up to 10 rows:

Query results

JOB INFORMATION RESULTS

Row	increase_percentage
1	137.0

Insight/Observation:

We can observe that there is a 137% increase in total sales for the Target Ecommerce in Brazil from FY17 to FY18. (For the data pertaining to Jan to Aug months). This seems to be a significant growth given its only the second year of Target in the country. This indicates that Target has been successful in acquiring weighty market share in the early days.

Recommendation:

Target should continue to focus on better customer experience because down the road retaining the customers is going to be more important than just keep gaining new ones. Hence, a push for a better loyalty program may be a good strategy.

b) ***Mean & Sum of price and freight value by customer state***

CODE:

```
SELECT

    c.customer_state,
    ROUND(AVG(oi.price),2) AS avg_price,
    ROUND(AVG(oi.freight_value),2) AS avg_freight_value,
    ROUND(SUM(oi.price),2) AS total_price,
    ROUND(SUM(oi.freight_value),2) AS total_freight_value
FROM target.customers c
JOIN target.orders o
ON o.customer_id = c.customer_id
JOIN target.order_items oi
ON oi.order_id = o.order_id
GROUP BY customer_state
ORDER BY avg_price DESC
```

Result up to 10 rows:

Query results

SAVE RESULTS EXPLOI

JOB INFORMATION RESULTS JSON EXECUTION DETAILS EXECUTION GRAPH PREVIEW

Row	customer_state	avg_price	avg_freight_valu	total_price	total_freight_val
1	PB	191.48	42.72	115268.08	25719.73
2	AL	180.89	35.84	80314.81	15914.59
3	AC	173.73	40.07	15982.95	3686.75
4	RO	165.97	41.07	46140.64	11417.38
5	PA	165.69	35.83	178947.81	38699.3
6	AP	164.32	34.01	13474.3	2788.5
7	PI	160.36	39.15	86914.08	21218.2
8	TO	157.53	37.25	49621.74	11732.68
9	RN	156.97	35.65	83034.98	18860.1
10	CE	153.76	32.71	227254.71	48351.59

Results per page: 50 1 – 27 of 27

CODE:

```
    SELECT
        c.customer_state,
        ROUND(AVG(oi.price),2) AS avg_price,
        ROUND(AVG(oi.freight_value),2) AS avg_freight_value,
    ROUND(SUM(oi.price),2) AS total_price,
    ROUND(SUM(oi.freight_value),2) AS total_freight_value
FROM target.customers c
JOIN target.orders o
ON o.customer_id = c.customer_id
JOIN target.order_items oi
ON oi.order_id = o.order_id
GROUP BY customer_state
ORDER BY avg_freight_value DESC
```

Result up to 10 rows:

Query results

SAVE RESULTS EXPLOI

JOB INFORMATION RESULTS JSON EXECUTION DETAILS EXECUTION GRAPH PREVIEW

Row	customer_state	avg_price	avg_freight_valu	total_price	total_freight_val
1	RR	150.57	42.98	7829.43	2235.19
2	PB	191.48	42.72	115268.08	25719.73
3	RO	165.97	41.07	46140.64	11417.38
4	AC	173.73	40.07	15982.95	3686.75
5	PI	160.36	39.15	86914.08	21218.2
6	MA	145.2	38.26	119648.22	31523.77
7	TO	157.53	37.25	49621.74	11732.68
8	SE	153.04	36.65	58920.85	14111.47
9	AL	180.89	35.84	80314.81	15914.59
10	PA	165.69	35.83	178947.81	38699.3

Results per page: 50 1 – 27 of 27

CODE:

```sql
SELECT
    c.customer_state,
    ROUND(AVG(oi.price),2) AS avg_price,
    ROUND(AVG(oi.freight_value),2) AS avg_freight_value,
    ROUND(SUM(oi.price),2) AS total_price,
    ROUND(SUM(oi.freight_value),2) AS total_freight_value
FROM target.customers c
JOIN target.orders o
ON o.customer_id = c.customer_id
JOIN target.order_items oi
ON oi.order_id = o.order_id
GROUP BY customer_state
ORDER BY total_price DESC
```

Result up to 10 rows:

Query results

SAVE RESULTS EXPLOI

JOB INFORMATION RESULTS JSON EXECUTION DETAILS EXECUTION GRAPH PREVIEW

Row	customer_state	avg_price	avg_freight_valu	total_price	total_freight_val
1	SP	109.65	15.15	5202955.05	718723.07
2	RJ	125.12	20.96	1824092.67	305589.31
3	MG	120.75	20.63	1585308.03	270853.46
4	RS	120.34	21.74	750304.02	135522.74
5	PR	119.0	20.53	683083.76	117851.68
6	SC	124.65	21.47	520553.34	89660.26
7	BA	134.6	26.36	511349.99	100156.68
8	DF	125.77	21.04	302603.94	50625.5
9	GO	126.27	22.77	294591.95	53114.98
10	ES	121.91	22.06	275037.31	49764.6

Results per page: 50 1 – 27 of 27

CODE:

```sql
SELECT
    c.customer_state,
    ROUND(AVG(oi.price),2) AS avg_price,
    ROUND(AVG(oi.freight_value),2) AS avg_freight_value,
    ROUND(SUM(oi.price),2) AS total_price,
    ROUND(SUM(oi.freight_value),2) AS total_freight_value
FROM target.customers c
JOIN target.orders o
ON o.customer_id = c.customer_id
JOIN target.order_items oi
ON oi.order_id = o.order_id
GROUP BY customer_state
ORDER BY total_freight_value DESC
```

Result up to 10 rows:

Query results

SAVE RESULTS EXPLOI

JOB INFORMATION RESULTS JSON EXECUTION DETAILS EXECUTION GRAPH PREVIEW

Row	customer_state	avg_price	avg_freight_valu	total_price	total_freight_valu
1	SP	109.65	15.15	5202955.05	718723.07
2	RJ	125.12	20.96	1824092.67	305589.31
3	MG	120.75	20.63	1585308.03	270853.46
4	RS	120.34	21.74	750304.02	135522.74
5	PR	119.0	20.53	683083.76	117851.68
6	BA	134.6	26.36	511349.99	100156.68
7	SC	124.65	21.47	520553.34	89660.26
8	PE	145.51	32.92	262788.03	59449.66
9	GO	126.27	22.77	294591.95	53114.98
10	DF	125.77	21.04	302603.94	50625.5

Results per page: 50 1 – 27 of 27

Insight/Observation:

We know when it comes to sheer volume of sales, SP, RJ and MG are the states that top the chart. While, PB, AL and AC seem to be getting more higher ticket orders which suggests a financially well-off customer base in those regions.

Recommendation:

It would be recommended that Target further investigates more about the states like SP, RJ and MG through conducting customer surveys, analysing the competition, and identifying potential opportunities for expansion and try and find a duplicate audience in other states as well

5. Analysis on sales, freight and delivery time:

a) ***Calculate days between purchasing, delivering and estimated delivery***

CODE:

```
SELECT
    order_id,
order_purchase_timestamp,
EXTRACT(DAY FROM order_estimated_delivery_date) -
EXTRACT(DAY FROM order_purchase_timestamp) AS days_to_deliver
FROM target.orders
ORDER BY order_purchase_timestamp
```

Result up to 10 rows:

Query results

SAVE RESULTS EXPLO

JOB INFORMATION | RESULTS | JSON | EXECUTION DETAILS | EXECUTION GRAPH PREVIEW

Row	order_id	order_purchase_timestamp	days_to_deliver
1	2e7a8482f6fb09756ca50c10d...	2016-09-04 21:15:19 UTC	16
2	e5fa5a7210941f7d56d0208e4...	2016-09-05 00:15:34 UTC	23
3	809a282bbd5dbcabb6f2f724fc...	2016-09-13 15:24:19 UTC	17
4	bfbd0f9bdef84302105ad712db...	2016-09-15 12:16:38 UTC	-11
5	71303d7e93b399f5bcd537d12...	2016-10-02 22:07:52 UTC	23
6	3b697a20d9e427646d925679...	2016-10-03 09:44:50 UTC	24
7	be5bc2f0da14d8071e2d45451...	2016-10-03 16:56:50 UTC	4
8	65d1e226dfaeb8cdc42f66542...	2016-10-03 21:01:41 UTC	22
9	a41c8759fbe7aab36ea07e038...	2016-10-03 21:13:36 UTC	26
10	d207cc272675637bfed0062ed...	2016-10-03 22:06:03 UTC	20

Results per page: 50 1 – 50 of 99441

b) Find time_to_delivery & diff_estimated_delivery. Formula for the same given below:

1. ***time_to_delivery = order_purchase_timestamp-order_delivered_customer_date***
2. ***diff_estimated_delivery = order_estimated_delivery_date-order_delivered_customer_date***

CODE:

```
SELECT
    order_id,
    order_purchase_timestamp,
TIMESTAMP_DIFF(order_delivered_customer_date,order_purchase_timestamp,DAY) AS time_to_delivery,
TIMESTAMP_DIFF(order_estimated_delivery_date,order_delivered_customer_date, DAY) AS diff_estimated
_delivery
FROM target.orders
ORDER BY order_purchase_timestamp
```

Result up to 10 rows:

Query results

SAVE RESULTS EXPLO

JOB INFORMATION | RESULTS | JSON | EXECUTION DETAILS | EXECUTION GRAPH PREVIEW

Row	order_id	order_purchase_timestamp	time_to_delivery	diff_estimated_c
1	2e7a8482f6fb09756ca50c10d...	2016-09-04 21:15:19 UTC	null	null
2	e5fa5a7210941f7d56d0208e4...	2016-09-05 00:15:34 UTC	null	null
3	809a282bbd5dbcabb6f2f724fc...	2016-09-13 15:24:19 UTC	null	null
4	bfbd0f9bdef84302105ad712db...	2016-09-15 12:16:38 UTC	54	-36
5	71303d7e93b399f5bcd537d12...	2016-10-02 22:07:52 UTC	null	null
6	3b697a20d9e427646d925679...	2016-10-03 09:44:50 UTC	23	0
7	be5bc2f0da14d8071e2d45451...	2016-10-03 16:56:50 UTC	24	10
8	65d1e226dfaeb8cdc42f66542...	2016-10-03 21:01:41 UTC	35	16
9	a41c8759fbe7aab36ea07e038...	2016-10-03 21:13:36 UTC	30	25
10	d207cc272675637bfed0062ed...	2016-10-03 22:06:03 UTC	27	22

Results per page: 50 1 – 50 of 99441

c) ***Group data by state, take mean of freight_value, time_to_delivery, diff_estimated_delivery***

CODE:

```
SELECT
    c.customer_state,
ROUND(AVG(oi.freight_value),2) AS avg_freight_value,
ROUND(AVG(TIMESTAMP_DIFF(o.order_delivered_customer_date,o.order_purchase_timestamp,DAY)),2) AS avg_time_to_delivery,
ROUND(AVG(TIMESTAMP_DIFF(o.order_estimated_delivery_date,o.order_delivered_customer_date, DAY)),2) AS avg_diff_estimated_delivery
FROM target.customers c
JOIN target.orders o
ON o.customer_id = c.customer_id
JOIN target.order_items oi
ON o.order_id = oi.order_id
GROUP BY c.customer_state
```

Result up to 10 rows:

Query results

SAVE RESULTS EXPLORE DATA

JOB INFORMATION RESULTS JSON EXECUTION DETAILS EXECUTION GRAPH PREVIEW

Row	customer_state	avg_freight_value	avg_time_to_delivery	avg_diff_estimated_delivery
1	MT	28.17	17.51	13.64
2	MA	38.26	21.2	9.11
3	AL	35.84	23.99	7.98
4	SP	15.15	8.26	10.27
5	MG	20.63	11.52	12.4
6	PE	32.92	17.79	12.55
7	RJ	20.96	14.69	11.14
8	DF	21.04	12.5	11.27
9	RS	21.74	14.71	13.2
10	SE	36.65	20.98	9.17

Results per page: 50 1 – 27 of 27

d) ***Sort the data to get the following:***

CODE:

```
SELECT

    c.customer_state,
ROUND(AVG(oi.freight_value),2) AS avg_freight_value,

  ROUND(AVG(TIMESTAMP_DIFF(o.order_delivered_customer_date,o.order_purchase_timestamp,DAY)),2)
AS avg_time_to_delivery,
 ROUND(AVG(TIMESTAMP_DIFF(o.order_estimated_delivery_date,o.order_delivered_customer_date, DAY)),2
) AS avg_diff_estimated_delivery
FROM target.customers c
JOIN target.orders o
ON o.customer_id = c.customer_id
JOIN target.order_items oi
ON o.order_id = oi.order_id
GROUP BY c.customer_state
ORDER BY avg_freight_value
```

Result up to 10 rows:

Query results

SAVE RESULTS EXPLORE DATA

JOB INFORMATION RESULTS JSON EXECUTION DETAILS EXECUTION GRAPH PREVIEW

Row	customer_state	avg_freight_valu	avg_time_to_deli	avg_diff_estimat
1	SP	15.15	8.26	10.27
2	PR	20.53	11.48	12.53
3	MG	20.63	11.52	12.4
4	RJ	20.96	14.69	11.14
5	DF	21.04	12.5	11.27
6	SC	21.47	14.52	10.67
7	RS	21.74	14.71	13.2
8	ES	22.06	15.19	9.77
9	GO	22.77	14.95	11.37
10	MS	23.37	15.11	10.34

Results per page: 50 1 – 27 of 27

e) ***Top 5 states with highest average freight value - sort in desc limit 5***

CODE:

```
SELECT
    c.customer_state,
ROUND(AVG(oi.freight_value),2) AS avg_freight_value,
ROUND(AVG(TIMESTAMP_DIFF(o.order_delivered_customer_date,o.order_purchase_timestamp,DAY)),2) AS
avg_time_to_delivery,
ROUND(AVG(TIMESTAMP_DIFF(o.order_estimated_delivery_date,o.order_delivered_customer_date, DAY)),2
) AS avg_diff_estimated_delivery
FROM target.customers c
JOIN target.orders o
ON o.customer_id = c.customer_id
JOIN target.order_items oi
ON o.order_id = oi.order_id
GROUP BY c.customer_state
ORDER BY avg_freight_value DESC
LIMIT 5
```

Result up to 5 rows:

Query results

SAVE RESULTS EXPLORE DATA

JOB INFORMATION RESULTS JSON EXECUTION DETAILS EXECUTION GRAPH PREVIEW

Row	customer_state	avg_freight_valu	avg_time_to_deli	avg_diff_estimat
1	RR	42.98	27.83	17.43
2	PB	42.72	20.12	12.15
3	RO	41.07	19.28	19.08
4	AC	40.07	20.33	20.01
5	PI	39.15	18.93	10.68

f) ***Top 5 states with lowest average freight value - sort in asc limit 5***

CODE:

```
SELECT
    c.customer_state,
ROUND(AVG(oi.freight_value),2) AS avg_freight_value,
ROUND(AVG(TIMESTAMP_DIFF(o.order_delivered_customer_date,o.order_purchase_timestamp,DAY)),2) AS
avg_time_to_delivery,
ROUND(AVG(TIMESTAMP_DIFF(o.order_estimated_delivery_date,o.order_delivered_customer_date, DAY)),2
) AS avg_diff_estimated_delivery
FROM target.customers c
JOIN target.orders o
ON o.customer_id = c.customer_id
JOIN target.order_items oi
ON o.order_id = oi.order_id
GROUP BY c.customer_state
ORDER BY avg_freight_value ASC
LIMIT 5
```

Result up to 5 rows:

Query results

SAVE RESULTS EXPLORE DATA

JOB INFORMATION RESULTS JSON EXECUTION DETAILS EXECUTION GRAPH PREVIEW

Row	customer_state	avg_freight_valu	avg_time_to_deli	avg_diff_estimat
1	RR	42.98	27.83	17.43
2	PB	42.72	20.12	12.15
3	RO	41.07	19.28	19.08
4	AC	40.07	20.33	20.01
5	PI	39.15	18.93	10.68

g) ***Top 5 states with highest average time to delivery***

CODE:

```
SELECT
    c.customer_state,
ROUND(AVG(oi.freight_value),2) AS avg_freight_value,
ROUND(AVG(TIMESTAMP_DIFF(o.order_delivered_customer_date,o.order_purchase_timestamp,DAY)),2) AS
avg_time_to_delivery,
ROUND(AVG(TIMESTAMP_DIFF(o.order_estimated_delivery_date,o.order_delivered_customer_date, DAY)),2
) AS avg_diff_estimated_delivery
FROM target.customers c
JOIN target.orders o
ON o.customer_id = c.customer_id
JOIN target.order_items oi
ON o.order_id = oi.order_id
GROUP BY c.customer_state
ORDER BY avg_time_to_delivery DESC
LIMIT 5
```

Result up to 5 rows:

Query results

SAVE RESULTS EXPLORE DATA

JOB INFORMATION | RESULTS | JSON | EXECUTION DETAILS | EXECUTION GRAPH PREVIEW

Row	customer_state	avg_freight_value	avg_time_to_delivery	avg_diff_estimated_delivery
1	RR	42.98	27.83	17.43
2	AP	34.01	27.75	17.44
3	AM	33.21	25.96	18.98
4	AL	35.84	23.99	7.98
5	PA	35.83	23.3	13.37

h) Top 5 states with lowest average time to delivery

CODE:

```
SELECT
    c.customer_state,
ROUND(AVG(oi.freight_value),2) AS avg_freight_value,
ROUND(AVG(TIMESTAMP_DIFF(o.order_delivered_customer_date,o.order_purchase_timestamp,DAY)),2) AS
avg_time_to_delivery,
ROUND(AVG(TIMESTAMP_DIFF(o.order_estimated_delivery_date,o.order_delivered_customer_date, DAY)),2
) AS avg_diff_estimated_delivery
FROM target.customers c
JOIN target.orders o
ON o.customer_id = c.customer_id
JOIN target.order_items oi
ON o.order_id = oi.order_id
GROUP BY c.customer_state
ORDER BY avg_time_to_delivery ASC
LIMIT 5
```

Result up to 5 rows:

Query results

SAVE RESULTS EXPLORE DATA

JOB INFORMATION RESULTS JSON EXECUTION DETAILS EXECUTION GRAPH PREVIEW

Row	customer_state	avg_freight_valu	avg_time_to_deli	avg_diff_estimat
1	SP	15.15	8.26	10.27
2	PR	20.53	11.48	12.53
3	MG	20.63	11.52	12.4
4	DF	21.04	12.5	11.27
5	SC	21.47	14.52	10.67

i) Top 5 states where delivery is really fast compared to estimated date

CODE:

```
SELECT
    c.customer_state,
AVG(TIMESTAMP_DIFF(o.order_delivered_customer_date,o.order_purchase_timestamp,DAY)) AS avg_time_to_delivery,
AVG(TIMESTAMP_DIFF(o.order_estimated_delivery_date,o.order_purchase_timestamp, DAY)) AS avg_diff_estimated_delivery,
(AVG(TIMESTAMP_DIFF(o.order_delivered_customer_date,o.order_purchase_timestamp,DAY)))/(AVG(TIMESTAMP_DIFF(o.order_estimated_delivery_date,o.order_purchase_timestamp, DAY))) as ratio
    FROM target.customers c
    JOIN target.orders o
    ON o.customer_id = c.customer_id
    GROUP BY c.customer_state
    ORDER BY ratio DESC
    LIMIT 5
```

Result up to 10 rows:

Query results

JOB INFORMATION RESULTS JSON EXECUTION DETAILS EXECUT

Row	customer_state	avg_time_to_delivery	avg_diff_estimated_delivery	ratio
1	AL	24.04	32.23	0.75
2	MA	21.12	30.11	0.7
3	SE	21.03	30.35	0.69
4	CE	20.82	30.94	0.67
5	BA	18.87	29.04	0.65

j) Top 5 states where delivery is not so fast compared to estimated date

CODE:

```
SELECT
    c.customer_state,
ROUND(AVG(TIMESTAMP_DIFF(o.order_delivered_customer_date,o.order_purchase_timestamp,DAY)),2) AS
avg_time_to_delivery,
ROUND(AVG(TIMESTAMP_DIFF(o.order_estimated_delivery_date,o.order_purchase_timestamp, DAY)),2) AS
avg_diff_estimated_delivery,
ROUND((AVG(TIMESTAMP_DIFF(o.order_delivered_customer_date,o.order_purchase_timestamp,DAY)))/(AV
G(TIMESTAMP_DIFF(o.order_estimated_delivery_date,o.order_purchase_timestamp, DAY))),2) as ratio
FROM target.customers c
JOIN target.orders o
ON o.customer_id = c.customer_id
GROUP BY c.customer_state
ORDER BY ratio ASC
```

Result up to 10 rows:

Query results

JOB INFORMATION | RESULTS | JSON | EXECUTION DETAILS | EXECUT

Row	customer_state	avg_time_to_deli	avg_diff_estimat	ratio
1	SP	8.3	18.81	0.44
2	PR	11.53	24.25	0.48
3	MG	11.54	24.22	0.48
4	RO	18.91	38.41	0.49
5	AC	20.64	40.77	0.51

6. Payment type analysis:

a) ***Month over Month count of orders for different payment types:***

CODE:

```
SELECT
  EXTRACT(MONTH FROM o.order_purchase_timestamp) AS month,
  COUNT(CASE WHEN p.payment_type = 'credit card' THEN 1 END) AS credit_card,
  COUNT(CASE WHEN p.payment_type = 'UPI' THEN 1 END) AS upi,
  COUNT(CASE WHEN p.payment_type = 'debit card' THEN 1 END) AS debit_card,
  COUNT(CASE WHEN p.payment_type = 'voucher' THEN 1 END) AS voucher,
  COUNT(CASE WHEN p.payment_type = 'not defined' THEN 1 END) AS not_defined
FROM target.orders AS o
JOIN target.payments AS p
ON o.order_id = p.order_id
GROUP BY month
ORDER BY month
```

Result up to 10 rows:

JOB INFORMATION | RESULTS | JSON | EXECUTION DETAILS | EXECUTION GRAPH PREVIEW

Row	month	credit_card	upi	debit_card	voucher	not_defined
1	1	0	1715	0	477	0
2	2	0	1723	0	424	0
3	3	0	1942	0	591	0
4	4	0	1783	0	572	0
5	5	0	2035	0	613	0
6	6	0	1807	0	563	0
7	7	0	2074	0	645	0
8	8	0	2077	0	589	0
9	9	0	903	0	302	0
10	10	0	1056	0	318	0

Results per page: 50 1 – 12 of 12

Insight/Observation:

The data suggests that only UPI and Vouchers are the payment types being used by the Brazilian customers. Now, UPI is a payment type used in India so, it's unlikely that Brazilian customers are using UPI as payment type, but if the data suggests it, this tells one of the two things about the customers. Either the Brazilians are slow at adopting to the traditional payment techniques of credit card and debit card, or they have leapfrogged the technology into using the UPI.

b) Count of orders based on the no. of payment instalments:

CODE:

```
SELECT
  p.payment_installments,
COUNT(*) AS count_of_orders

FROM target.orders o

JOIN target.payments p
ON o.order_id = p.order_id
GROUP BY p.payment_installments
ORDER BY count_of_orders DESC
```

Result up to 10 rows:

Query results

SAVE RESULTS EXPLOI

JOB INFORMATION RESULTS JSON EXECUTION DETAILS EXECUTION GRAPH PREVIEW

Row	payment_installm	count_of_orders
1	1	52546
2	2	12413
3	3	10461
4	4	7098
5	10	5328
6	5	5239
7	8	4268
8	6	3920
9	7	1626
10	9	644

Results per page: 50 1 – 24 of 24

Insight/Observation:

Here, we can see that mostly the payments ate being done in a single instalment itself which is an indicator of a financially strong customer segment that is getting attracted towards the company's products on its ecommerce platform. Though, we see a large number of orders being paid in 2, 3 or 4 instalments too, which indicates an audience base that prefers paying in multiple payments.

Recommendation:

Looking at the number of orders paid in multiple instalments, the company needs to accommodate, if it is not already, Buy Now Pay Later facilities by having third party collaborations. This might help increase those numbers. Also, Target can incentivise the customers to pay in a single payment by offering attractive discounts.