BUSINESS CASE: TARGET SQL

Submitted By:

Karan Viswakarma

karanviswa308@gmail.com

+91-9777496373

INDEX

Business Case: Target SQL

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

1.1. Data type of columns in a table:

QUERY:

```
SELECT TABLE_NAME, COLUMN_NAME, DATA_TYPE FROM bcs1target.INFORMATION_SCHEMA.COLUMNS;
```

Query results

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

Row	TABLE_NAME	COLUMN_NAME	DATA_TYPE
1	order_items	order_id	STRING
2	order_items	order_item_id	INT64
3	order_items	product_id	STRING
4	order_items	seller_id	STRING
5	order_items	shipping_limit_date	TIMESTAMP
6	order_items	price	FLOAT64
7	order_items	freight_value	FLOAT64
8	sellers	seller_id	STRING
9	sellers	seller_zip_code_prefix	INT64
10	sellers	seller_city	STRING
11	sellers	seller_state	STRING
12	reviews	review_id	STRING
13	reviews	order_id	STRING
14	reviews	review_score	INT64
15	reviews	review_comment_title	STRING

1.2. Perioderiod for which the data is given:

QUERY:

```
SELECT MIN(order_purchase_timestamp) AS min_time_of_purchase,
       MAX(order_estimated_delivery_date) AS max_time_of_purchase
FROM `bcs1target.orders`, `bcs1target.order_items`;
```

Query results

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

Row	min_time_of_purchase	max_time_of_purchase
1	2016-09-04 21:15:19 UTC	2018-11-12 00:00:00 UTC

1.3. Cities and States covered in the dataset

QUERY:

```
SELECT DISTINCT customer_state, customer_city
FROM `bcs1target.customer`
GROUP BY 1, 2 ORDER BY 1, 2;
```

Query results

JOB INFORMATION | RESULTS | JSON | EXECUTION DETAILS

Row	customer_state	customer_city
1	AC	brasileia
2	AC	cruzeiro do sul
3	AC	epitaciolandia
4	AC	manoel urbano
5	AC	porto acre
6	AC	rio branco
7	AC	senador guiomard
8	AC	xapuri
9	AL	agua branca
10	AL	anadia

2. In-depth Exploration:

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

QUERY:

```
SELECT *, ROUND(((orders_count - prev_order_count) / prev_order_count) * 100, 2) AS order_growth_r
ate_percent FROM
  (SELECT *, LAG(orders_count) OVER(ORDER BY YEAR, MONTH) AS prev_order_count FROM
    (SELECT EXTRACT(YEAR FROM order_purchase_timestamp) AS YEAR, EXTRACT(MONTH FROM order_purchase
_timestamp) AS MONTH, COUNT(*) AS orders_count
    FROM `bcs1target.orders`
    WHERE order_status = 'delivered'
    GROUP BY 1, 2 ORDER BY 1, 2) AS BASE1 ORDER BY YEAR, MONTH) AS BASE2;
```

Query results

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

Row	YEAR	MONTH	orders_count	prev_order_count	order_growth_rate_percent
1	2016	9	1	*null*	*null*
2	2016	10	265	1	26400.0
3	2016	12	1	265	-99.62
4	2017	1	750	1	74900.0
5	2017	2	1653	750	120.4
6	2017	3	2546	1653	54.02
7	2017	4	2303	2546	-9.54
8	2017	5	3546	2303	53.97
9	2017	6	3135	3546	-11.59
10	2017	7	3872	3135	23.51

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

QUERY:

```
SELECT COUNTIF((TIME(order_purchase_timestamp) >= '05:00:00' AND TIME(order_purchase_timestamp) <
'06:00:00')) AS dawn_orders_count_5am_6am,
    COUNTIF((TIME(order_purchase_timestamp) >= '06:00:00' AND TIME(order_purchase_timestamp) < '12
:00:00')) AS morning_orders_count_6am_12pm,
    COUNTIF((TIME(order_purchase_timestamp) >= '12:00:00' AND TIME(order_purchase_timestamp) < '18
:00:00')) AS afternoon_orders_count_12pm_6pm,
    COUNTIF((TIME(order_purchase_timestamp) >= '18:00:00' AND TIME(order_purchase_timestamp) <= '2
3:59:59') OR (TIME(order_purchase_timestamp) >= '00:00:00' AND TIME(order_purchase_timestamp) < '0
5:00:00')) AS night_orders_count_6pm_5am
 FROM `bcs1target.orders`;
```

Query results

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

Row	dawn_orders_count_5am_6am	morning_orders_count_6am_12pm	afternoon_orders_count_12pm_6pm	night_orders_count_6pm_5am
1	188	22240	38361	38652

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

3.1. Get month-on-month orders by region, states

QUERY:

```
SELECT *, ROUND(((orders_count - prev_orders_count) / prev_orders_count) * 100, 2) AS orders_count
_growth_rate FROM
  (SELECT *, LAG(orders_count) OVER(PARTITION BY customer_state, customer_city ORDER BY YEAR, MONT
H) AS prev_orders_count FROM
    (SELECT C.customer_state, C.customer_city, BASE1.YEAR, BASE1.MONTH, COUNT(*) AS orders_count F
ROM `bcs1target.customer` AS C
      JOIN
      (SELECT *, EXTRACT(MONTH FROM order_purchase_timestamp) AS MONTH, EXTRACT(YEAR FROM order_pu
rchase_timestamp) AS YEAR FROM `bcs1target.orders`
      WHERE order_status = 'delivered') AS BASE1 ON C.customer_id = BASE1.customer_id GROUP BY 1,
2, 3, 4)) AS BASE2;
```

Query results

SAVE RESULTS

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

Row	customer_state	customer_city	YEAR	MONTH	orders_count	prev_orders_count	orders_count_growth_rate
1	AC	rio branco	2017	1	2	null	null
2	AC	rio branco	2017	2	2	2	0.0
3	AC	rio branco	2017	3	2	2	0.0
4	AC	rio branco	2017	4	4	2	100.0
5	AC	rio branco	2017	5	8	4	100.0
6	AC	rio branco	2017	6	4	8	-50.0
7	AC	rio branco	2017	7	5	4	25.0
8	AC	rio branco	2017	8	4	5	-20.0
9	AC	rio branco	2017	9	3	4	-25.0
10	AC	rio branco	2017	10	4	3	33.33

3.2. How are customers distributed in Brazil

QUERY:

```
SELECT customer_state, customer_city ,
      COUNT(DISTINCT customer_id) AS count_customer_id,
      COUNT(DISTINCT customer_unique_id) AS count_customer_unique_id
FROM `bcs1target.customer`
GROUP BY 1, 2
ORDER BY 1, 2;
```

Query results

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

Row	customer_state	customer_city	count_customer_id	count_customer_unique_id
1	AC	brasileia	1	1
2	AC	cruzeiro do sul	3	3
3	AC	epitaciolandia	1	1
4	AC	manoel urbano	1	1
5	AC	porto acre	1	1
6	AC	rio branco	70	66
7	AC	senador guiomard	2	2
8	AC	xapuri	2	2
9	AL	agua branca	1	1
10	AL	anadia	2	2

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

4.1. Get % increase in cost of orders from 2017 to 2018 (include months between Jan to Aug only)

QUERY:

```
WITH TABLE1 AS
(SELECT ROUND(SUM(price + freight_value), 2) AS total_cost_2017
FROM (SELECT O.*, OI.* FROM `bcs1target.orders` AS O JOIN `bcs1target.order_items` AS OI ON O.orde
r_id = OI.order_id
WHERE O.order_status = 'delivered' AND (EXTRACT(YEAR FROM O.order_purchase_timestamp) = 2017) AND
EXTRACT(MONTH FROM O.order_purchase_timestamp) BETWEEN 1 AND 8)),
TABLE2 AS (SELECT ROUND(SUM(price + freight_value),2) AS total_cost_2018
FROM (SELECT O.*, OI.* FROM `bcs1target.orders` AS O JOIN `bcs1target.order_items` AS OI ON O.orde
r_id = OI.order_id
WHERE O.order_status = 'delivered' AND (EXTRACT(YEAR FROM O.order_purchase_timestamp) = 2018) AND
EXTRACT(MONTH FROM O.order_purchase_timestamp) BETWEEN 1 AND 8))
SELECT T1.total_cost_2017, T2.total_cost_2018, ROUND(((T2.total_cost_2018 - T1.total_cost_2017) /
T1.total_cost_2017) * 100, 2) AS cost_growth_rate
FROM TABLE1 AS T1 CROSS JOIN TABLE2 AS T2;
```

Query results

JOB INFORMATION | RESULTS | JSON | EXECUTION DETAILS

Row	total_cost_2017	total_cost_2018	cost_growth_rate
1	3472898.25	8451584.77	143.36

4.2. Mean & Sum of price and freight value by customer state

QUERY:

```
SELECT C.customer_state, ROUND(AVG(OI.price + OI.freight_value), 2) AS avg_cost, ROUND(SUM(price +
 freight_value), 2) AS sum_cost
FROM `bcs1target.customer` AS C JOIN `bcs1target.orders` AS O ON C.customer_id = O.customer_id
      JOIN
  `bcs1target.order_items` AS OI ON O.order_id = OI.order_id
WHERE O.order_status = 'delivered'
GROUP BY 1;
```

Query results

JOB INFORMATION | RESULTS | JSON | EXECUTION DETAILS

Row	customer_state	avg_cost	sum_cost
1	GO	146.78	334212.35
2	SP	124.22	5769703.15
3	RS	140.44	861472.79
4	BA	160.5	591137.81
5	MG	140.82	1818891.67
6	MT	174.76	181224.42
7	RJ	145.33	2055401.57
8	SC	145.26	595127.78
9	SE	187.44	70289.13
10	PE	176.96	308972.05

5. Analysis of sales, freight, and delivery time

5.1. Calculate days between purchasing, delivering, and estimated delivery

QUERY:

```
SELECT order_id, order_purchase_timestamp, order_delivered_customer_date, order_estimated_delivery
_date,
        (TIMESTAMP_DIFF(order_delivered_customer_date,order_purchase_timestamp, DAY)) AS time_to_d
elivery,
       (TIMESTAMP_DIFF(order_estimated_delivery_date, order_purchase_timestamp, DAY)) AS diff_esti
mated_delivery,
        (TIMESTAMP_DIFF(order_estimated_delivery_date, order_delivered_customer_date, DAY)) AS dif
f_estdel_actdel
FROM bcs1target.orders
WHERE order_status = 'delivered';
```

Query results

SAVE RESULTS EXPLORE DATA

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

Row	order_id	order_purchase_timestamp	order_delivered_customer_date	order_estimated_delivery_date	time_to_delivery	diff_estimated_delivery	diff_estdel_actdel
1	cec8f5f7a13e5ab934a486ec9eb713c8	2017-03-17 15:56:47 UTC	2017-04-07 13:14:56 UTC	2017-05-18 00:00:00 UTC	20	61	40
2	58527ee4726911bee84a0f42cdd797c1	2017-03-20 11:01:17 UTC	2017-03-30 14:04:04 UTC	2017-05-18 00:00:00 UTC	10	58	48
3	10ed5499d1623638ee810eff1deccded	2017-03-21 13:38:25 UTC	2017-04-18 13:52:43 UTC	2017-05-18 00:00:00 UTC	28	57	29
4	818996ea247803ddc123789f2bd6046b	2018-08-20 15:56:23 UTC	2018-08-29 22:52:40 UTC	2018-10-04 00:00:00 UTC	9	44	35
5	d195cac9ccaa1394ede717d38d075fac	2018-08-12 18:14:29 UTC	2018-08-23 02:08:44 UTC	2018-10-04 00:00:00 UTC	10	52	41
6	64eeb35d3ade7fcdff9fbb1ca5175bcf	2018-08-16 07:55:32 UTC	2018-08-23 00:09:45 UTC	2018-10-04 00:00:00 UTC	6	48	41
7	2691ae869f13b10f3d356461b4311c73	2018-08-22 22:39:54 UTC	2018-08-29 19:11:48 UTC	2018-10-04 00:00:00 UTC	6	42	35
8	1cd147d1c0fe18f3b742a353396c44a7	2018-08-20 17:04:34 UTC	2018-08-29 16:41:59 UTC	2018-10-04 00:00:00 UTC	8	44	35
9	b36d2e6b1781d380e140608a4e831277	2018-08-09 19:17:50 UTC	2018-08-22 18:04:27 UTC	2018-10-04 00:00:00 UTC	12	55	42
10	88ab6b0ede7f19c65b5b71771b88254f	2018-08-13 12:12:46 UTC	2018-08-29 20:58:39 UTC	2018-10-04 00:00:00 UTC	16	51	35

5.2. Create columns:

- time_to_delivery = order_purchase_timestamp-order_delivered_customer_date
- diff_estimated_delivery = order_estimated_delivery_date-order_delivered_customer_date

QUERY:

```
SELECT order_id, order_purchase_timestamp, order_delivered_customer_date, order_estimated_delivery
_date,
        (TIMESTAMP_DIFF(order_delivered_customer_date,order_purchase_timestamp, DAY)) AS time_to_d
elivery,
        (TIMESTAMP_DIFF(order_estimated_delivery_date, order_delivered_customer_date, DAY)) AS dif
f_estimated_delivery
FROM bcs1target.orders
WHERE order_status = 'delivered';
```

Query results

SAVE RESULTS

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

Row	order_id	order_purchase_timestamp	order_delivered_customer_date	order_estimated_delivery_date	time_to_delivery	diff_estimated_delivery
1	635c894d068ac37e6e03dc54eccb6189	2017-04-15 15:37:38 UTC	2017-05-16 14:49:55 UTC	2017-05-18 00:00:00 UTC	30	1
2	3b97562c3aee8bdedcb5c2e45a50d5e1	2017-04-14 22:21:54 UTC	2017-05-17 10:52:15 UTC	2017-05-18 00:00:00 UTC	32	0
3	68f47f50f04c4cb6774570cfde3a9aa7	2017-04-16 14:56:13 UTC	2017-05-16 09:07:47 UTC	2017-05-18 00:00:00 UTC	29	1
4	276e9ec344d3bf029ff83a161c6b3ce9	2017-04-08 21:20:24 UTC	2017-05-22 14:11:31 UTC	2017-05-18 00:00:00 UTC	43	-4
5	54e1a3c2b97fb0809da548a59f64c813	2017-04-11 19:49:45 UTC	2017-05-22 16:18:42 UTC	2017-05-18 00:00:00 UTC	40	-4
6	fd04fa4105ee8045f6a0139ca5b49f27	2017-04-12 12:17:08 UTC	2017-05-19 13:44:52 UTC	2017-05-18 00:00:00 UTC	37	-1
7	302bb8109d097a9fc6e9cefc5917d1f3	2017-04-19 22:52:59 UTC	2017-05-23 14:19:48 UTC	2017-05-18 00:00:00 UTC	33	-5
8	66057d37308e787052a32828cd007e58	2017-04-15 19:22:06 UTC	2017-05-24 08:11:57 UTC	2017-05-18 00:00:00 UTC	38	-6
9	19135c945c554eebfd7576c733d5ebdd	2017-07-11 14:09:37 UTC	2017-08-16 20:19:32 UTC	2017-08-14 00:00:00 UTC	36	-2
10	4493e45e7ca1084efcd38ddebf174dda	2017-07-11 20:56:34 UTC	2017-08-14 21:37:08 UTC	2017-08-14 00:00:00 UTC	34	0

5.3. Group data by state, take mean of freight_value, time_to_delivery, diff_estimated_delivery

QUERY:

```
SELECT C.customer_state,
      AVG(OI.freight_value) AS avg_freight_value,
      AVG(BASE1.time_to_delivery) AS avg_time_to_delivery,
      AVG(BASE1.diff_estimated_delivery) AS avg_diff_estimated_delivery
FROM `bcs1target.customer` AS C
      JOIN
      (SELECT *,
            TIMESTAMP_DIFF(order_delivered_customer_date, order_purchase_timestamp, DAY) AS time_t
o_delivery,
            TIMESTAMP_DIFF(order_estimated_delivery_date, order_purchase_timestamp, DAY) AS diff_e
stimated_delivery,
      FROM `bcs1target.orders`
      WHERE order_status = 'delivered') AS BASE1 ON C.customer_id = BASE1.customer_id
      JOIN
      `bcs1target.order_items` AS OI ON BASE1.order_id = OI.order_id
GROUP BY C.customer_state;
```

Query results

JOB INFORMATION | RESULTS | JSON | EXECUTION DETAILS | EXECUTION GRAPI

Row	customer_state	avg_freight_value	avg_time_to_delivery	avg_diff_estimated_delivery
1	GO	22.56	14.95	26.63
2	SP	15.12	8.26	18.87
3	RS	21.61	14.71	28.27
4	BA	26.49	18.77	29.18
5	MG	20.63	11.51	24.26
6	MT	28.0	17.51	31.48
7	RJ	20.91	14.69	26.08
8	SC	21.51	14.52	25.51
9	SE	36.57	20.98	30.42
10	PE	32.69	17.79	30.67

5.4. Sort the data to get the following:

5.4.1. Top 5 states with highest/lowest average freight value - sort in desc/asc limit 5

QUERY FOR Top 5 states with highest avg_freight_value:

```
SELECT C.customer_state, ROUND(AVG(OI.freight_value), 2) AS avg_freight_value,
FROM `bcs1target.customer` AS C
      JOIN
      (SELECT *,
      FROM `bcs1target.orders`
      WHERE order_status = 'delivered') AS BASE1 ON C.customer_id = BASE1.customer_id
      JOIN
      `bcs1target.order_items` AS OI ON BASE1.order_id = OI.order_id
GROUP BY C.customer_state
ORDER BY AVG(OI.freight_value) DESC LIMIT 5;
```

Query results

JOB INFORMATION | RESULTS | JSON | EXEC

Row	customer_state	avg_freight_value
1	PB	43.09
2	RR	43.09
3	RO	41.33
4	AC	40.05
5	PI	39.12

QUERY FOR Top 5 states with lowest avg_freight_value:

```
SELECT C.customer_state, ROUND(AVG(OI.freight_value), 2) AS avg_freight_value,
FROM `bcs1target.customer` AS C
      JOIN
      (SELECT *,
      FROM `bcs1target.orders`
      WHERE order_status = 'delivered') AS BASE1 ON C.customer_id = BASE1.customer_id
      JOIN
      `bcs1target.order_items` AS OI ON BASE1.order_id = OI.order_id
GROUP BY C.customer_state
ORDER BY AVG(OI.freight_value) ASC
LIMIT 5;
```

Query results

JOB INFORMATION | RESULTS | JSON

Row	customer_state	avg_freight_value
1	SP	15.12
2	PR	20.47
3	MG	20.63
4	RJ	20.91
5	DF	21.07

5.4.2. Top 5 states with highest/lowest average time to delivery

QUERY for the top 5 states with the highest average time to delivery:

```
SELECT C.customer_state,  ROUND(AVG(BASE1.time_to_delivery), 2) AS avg_time_to_delivery,
FROM `bcs1target.customer` AS C
      JOIN
      (SELECT *,
            TIMESTAMP_DIFF(order_delivered_customer_date, order_purchase_timestamp, DAY) AS time_t
o_delivery,
      FROM `bcs1target.orders`
      WHERE order_status = 'delivered') AS BASE1 ON C.customer_id = BASE1.customer_id
GROUP BY C.customer_state
ORDER BY AVG(BASE1.time_to_delivery) DESC
```

```
LIMIT 5;
```

Query results

JOB INFORMATION | RESULTS | JSON | EX

Row	customer_state	avg_time_to_delivery
1	RR	28.98
2	AP	26.73
3	AM	25.99
4	AL	24.04
5	PA	23.32

QUERY for the top 5 states with the lowest average time to delivery:

```
SELECT C.customer_state, ROUND(AVG(BASE1.time_to_delivery), 2) AS avg_time_to_delivery,
FROM `bcs1target.customer` AS C
      JOIN
      (SELECT *,
            TIMESTAMP_DIFF(order_delivered_customer_date, order_purchase_timestamp, DAY) AS time_t
o_delivery
      FROM `bcs1target.orders`
      WHERE order_status = 'delivered') AS BASE1 ON C.customer_id = BASE1.customer_id
GROUP BY C.customer_state
ORDER BY AVG(BASE1.time_to_delivery) ASC
LIMIT 5;
```

Query results

JOB INFORMATION | RESULTS | JSON | EXECUTIC

Row	customer_state	avg_time_to_delivery
1	SP	8.3
2	PR	11.53
3	MG	11.54
4	DF	12.51
5	SC	14.48

5.4.3. Top 5 states where delivery is really fast/ not so fast compared to the estimated date

QUERY for the top 5 states where delivery is really fast compared to the estimated date

```
SELECT C.customer_state, ROUND(AVG(BASE1.diff_estdel_actdel), 2) AS avg_daydiff_estdel_actdel
FROM `bcs1target.customer` AS C
      JOIN
      (SELECT *,
            TIMESTAMP_DIFF(order_estimated_delivery_date, order_delivered_customer_date, DAY) AS d
iff_estdel_actdel
      FROM `bcs1target.orders`
```

```
      WHERE order_status = 'delivered') AS BASE1 ON C.customer_id = BASE1.customer_id
GROUP BY C.customer_state
ORDER BY AVG(BASE1.diff_estdel_actdel) DESC
LIMIT 5;
```

Query results

JOB INFORMATION | RESULTS | JSON | EXECUT

Row	customer_state	avg_daydiff_estdel_actdel
1	AC	19.76
2	RO	19.13
3	AP	18.73
4	AM	18.61
5	RR	16.41

QUERY for top 5 states where delivery is NOT SO FAST compared to the estimated date

```
SELECT C.customer_state, ROUND(AVG(BASE1.diff_estdel_actdel), 2) AS avg_daydiff_estdel_actdel
FROM `bcs1target.customer` AS C
      JOIN
      (SELECT *,
            TIMESTAMP_DIFF(order_estimated_delivery_date, order_delivered_customer_date, DAY) AS d
iff_estdel_actdel
      FROM `bcs1target.orders`
      WHERE order_status = 'delivered') AS BASE1 ON C.customer_id = BASE1.customer_id
GROUP BY C.customer_state
ORDER BY AVG(BASE1.diff_estdel_actdel) ASC
LIMIT 5;
```

Query results

JOB INFORMATION | RESULTS | JSON

Row	customer_state	avg_daydiff_estdel_actdel
1	AL	7.95
2	MA	8.77
3	SE	9.17
4	ES	9.62
5	BA	9.93

6. Payment type analysis:

6.1. Month over Month count of orders for different payment types:

QUERY

```
SELECT BASE2.*, ROUND(((BASE2.payment_type_count - BASE2.prev_count) / BASE2.prev_count) * 100, 2)
 AS count_growth_rate_percent
FROM
  (SELECT payment_type, YEAR, MONTH, payment_type_count,
        LAG(payment_type_count) OVER(PARTITION BY payment_type ORDER BY YEAR, MONTH) AS prev_count
  FROM
    (SELECT DISTINCT P.payment_type, O.YEAR, O.MONTH,
          COUNT(*) OVER (PARTITION BY P.payment_type, O.YEAR, O.MONTH ORDER BY O.YEAR, O.MONTH) AS
 payment_type_count
    FROM `bcs1target.payments` AS P
      JOIN
      (SELECT order_id, EXTRACT(MONTH FROM order_purchase_timestamp) AS MONTH, EXTRACT(YEAR FROM o
rder_purchase_timestamp) AS YEAR,
      FROM `bcs1target.orders`
      WHERE order_status = 'delivered') AS O ON P.order_id = O.order_id) AS BASE1) AS BASE2
```

Query results

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

Row	payment_type	YEAR	MONTH	payment_type_count	prev_count	count_growth_rate_percent
1	UPI	2016	10	51	null	null
2	UPI	2017	1	188	51	268.63
3	UPI	2017	2	371	188	97.34
4	UPI	2017	3	565	371	52.29
5	UPI	2017	4	474	565	-16.11
6	UPI	2017	5	740	474	56.12
7	UPI	2017	6	689	740	-6.89
8	UPI	2017	7	811	689	17.71
9	UPI	2017	8	902	811	11.22
10	UPI	2017	9	868	902	-3.77

6.2. Distribution of payment installments and count of orders

QUERY

```
SELECT P.payment_installments, COUNT(*) AS orders_count
FROM `bcs1target.payments` AS P
  JOIN
  (SELECT *
    FROM `bcs1target.orders`
    WHERE order_status = 'delivered') AS O ON P.order_id = O.order_id
```

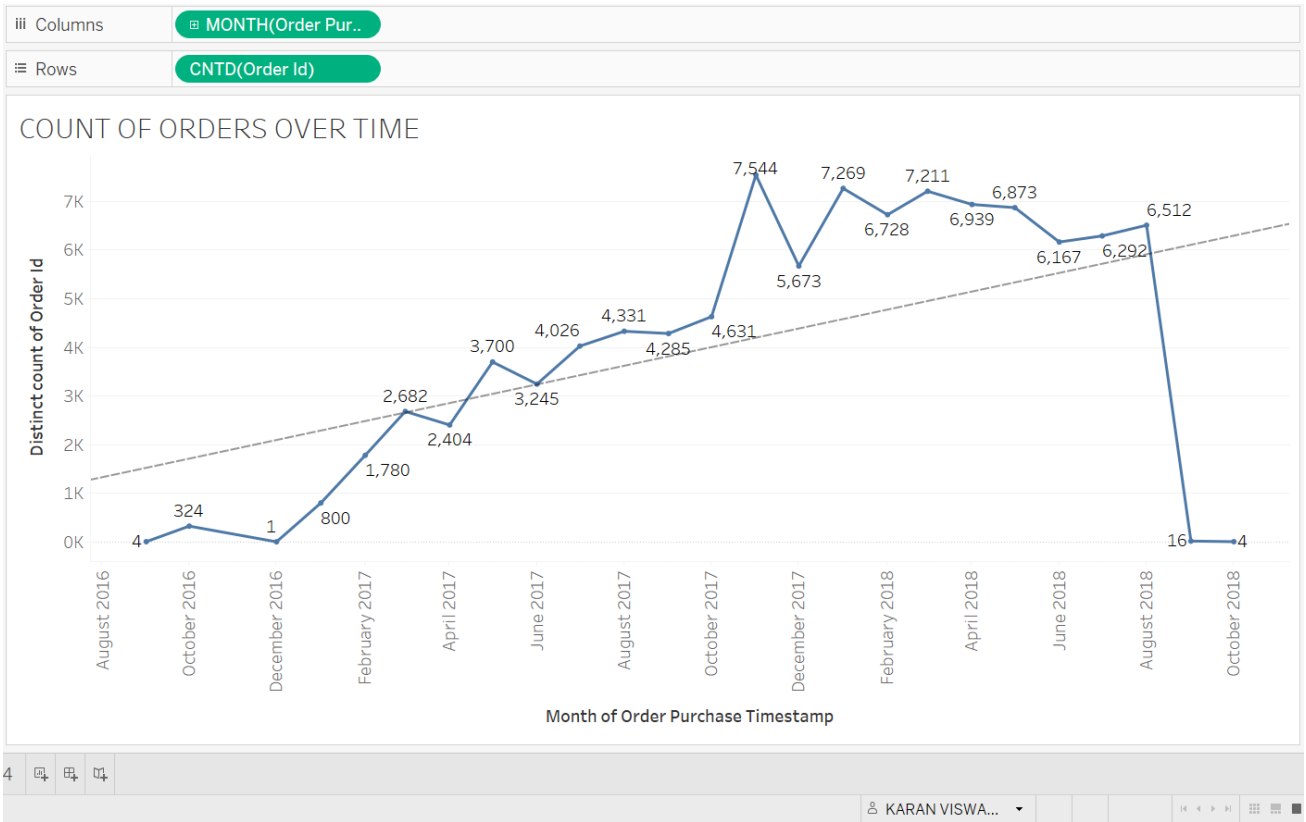
```
GROUP BY 1;
```

Query results

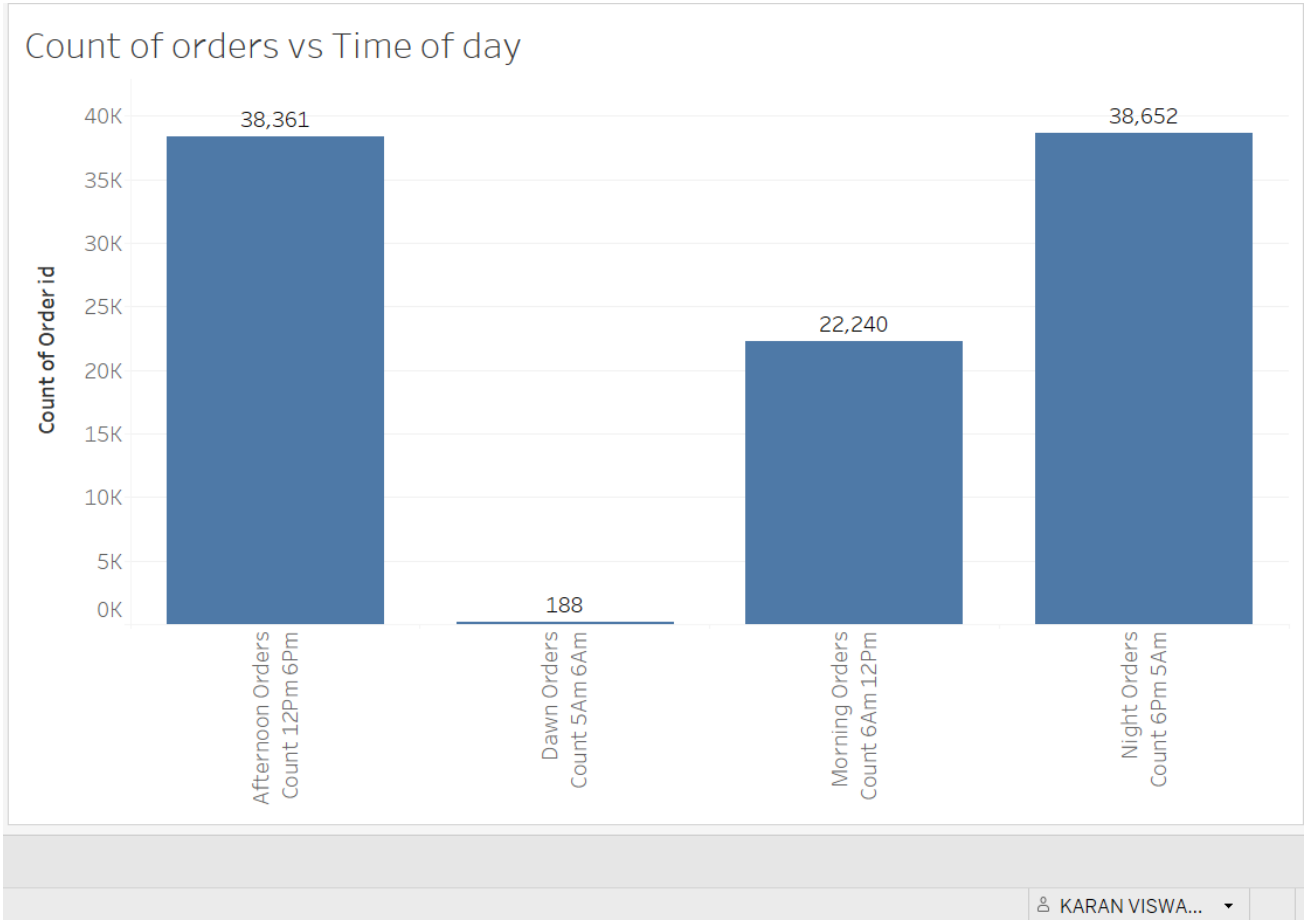
JOB INFORMATION | RESULTS | JSON

Row	payment_installments	orders_count
1	0	2
2	1	50929
3	2	12075
4	3	10164
5	4	6891
6	5	5095
7	6	3804
8	7	1563
9	8	4136
10	9	618

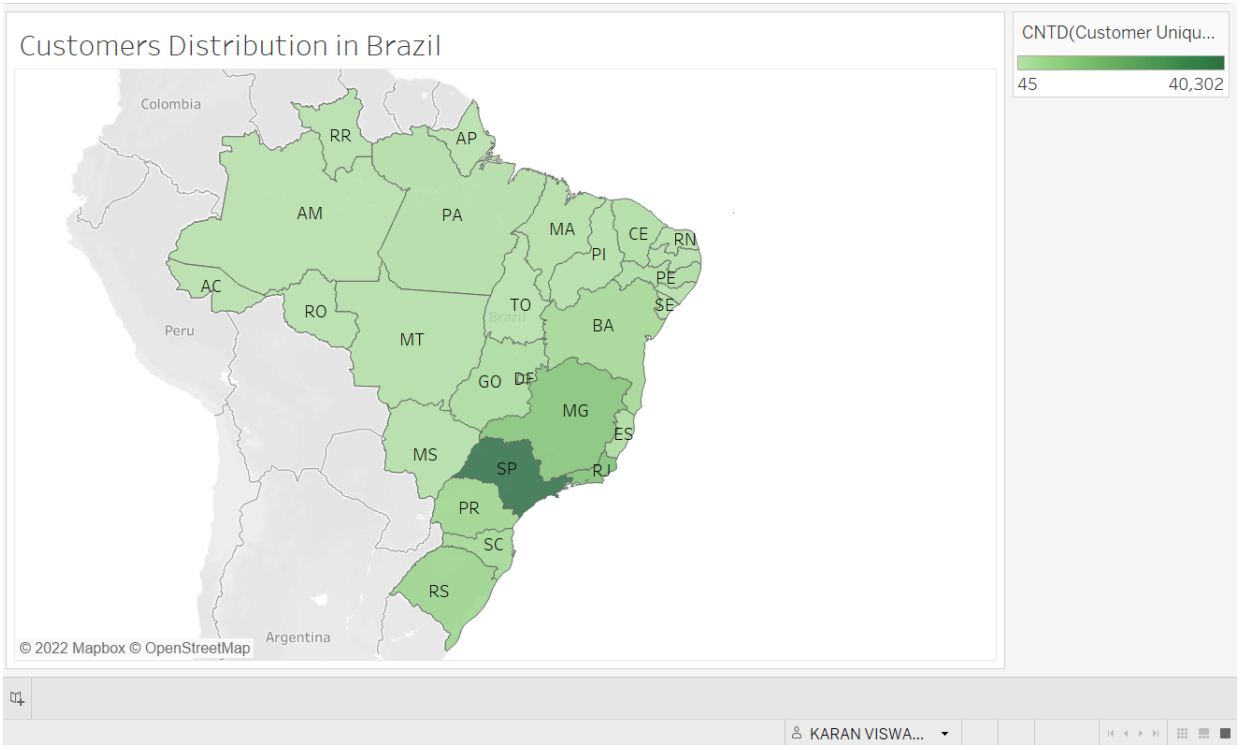
7. Actionable Insights



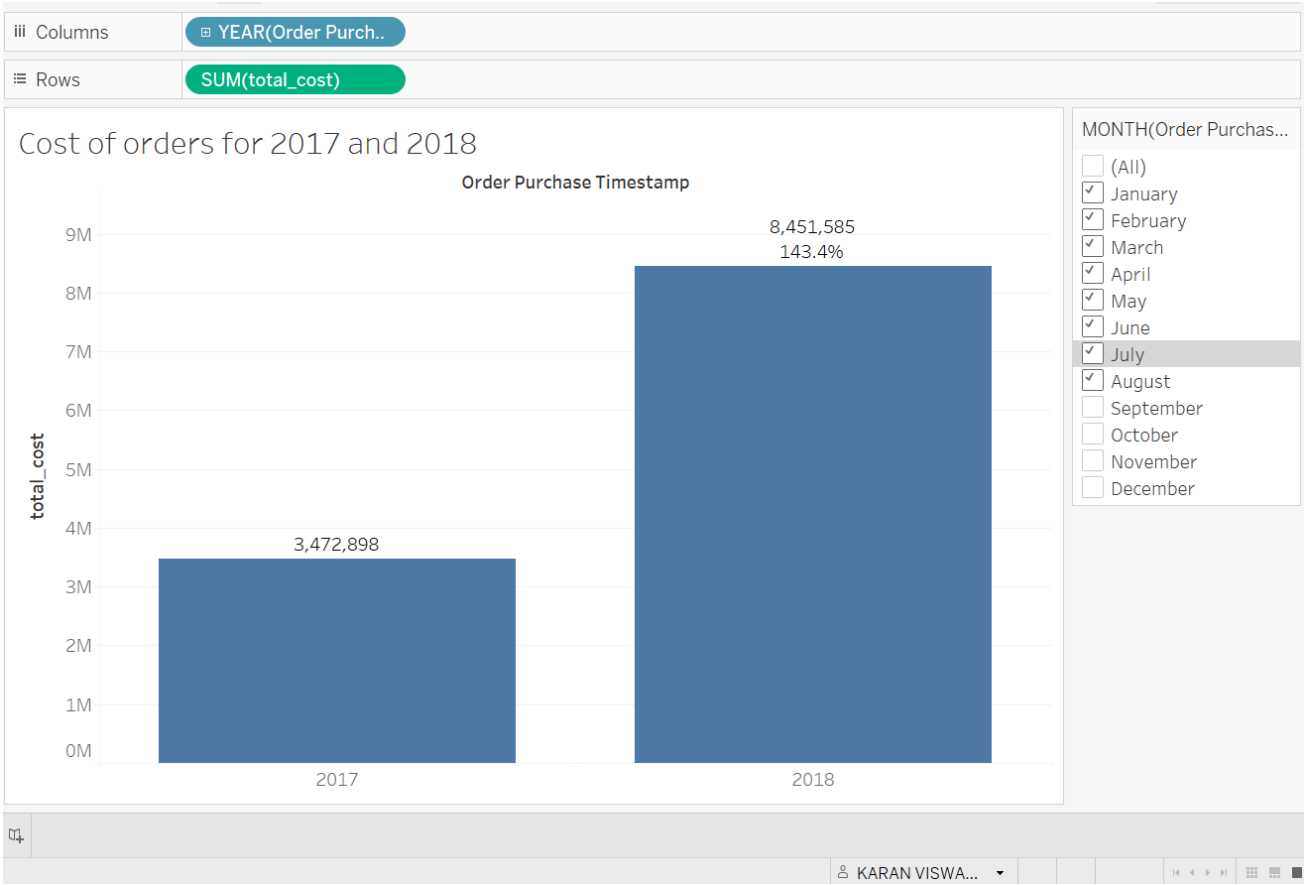
- It can be inferred from the above graph that there is a growing trend for America's leading retailer, Target over the years for which the data is given i.e. from the year 2016 to 2018. The peak of the orders count occurred in November month of 2018 while there was a huge decrease in the orders count from August 2018 to September 2018.



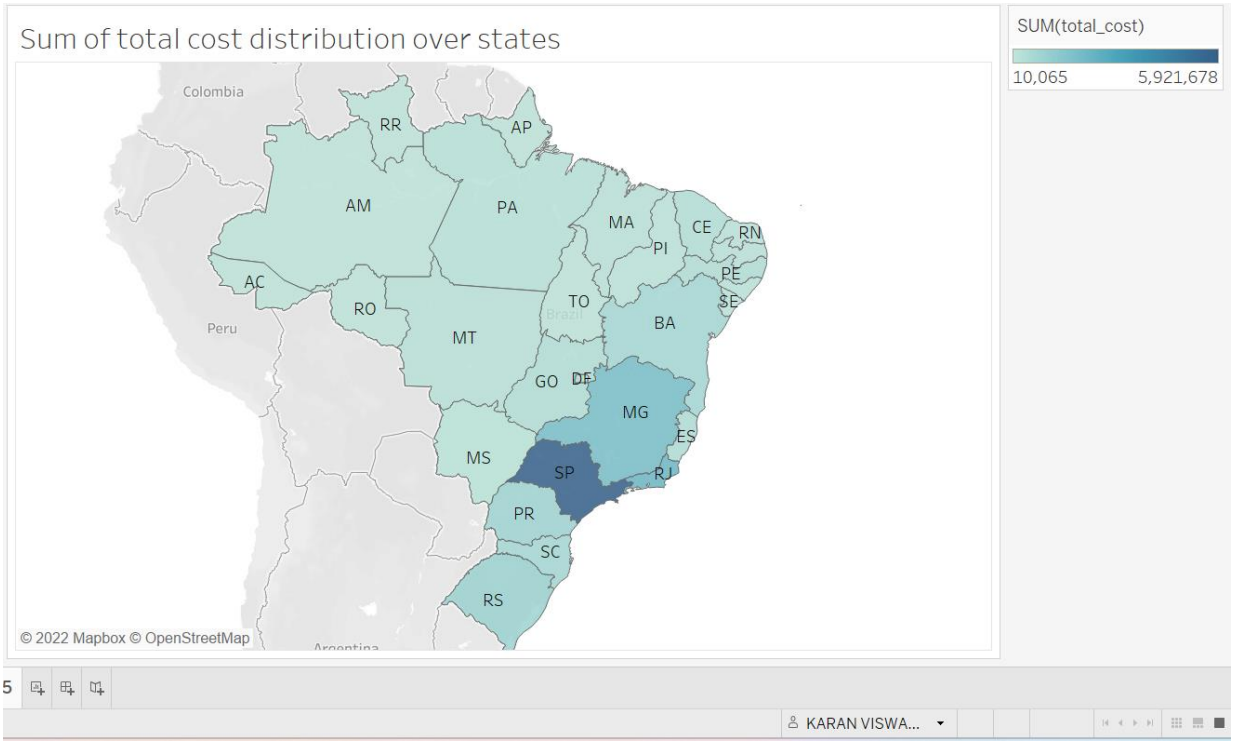
- It can be seen from the above graph that the customers prefer making orders at night and afternoon rather than at dawn and morning. More than three fourth of the total orders are made during these two times of the day.



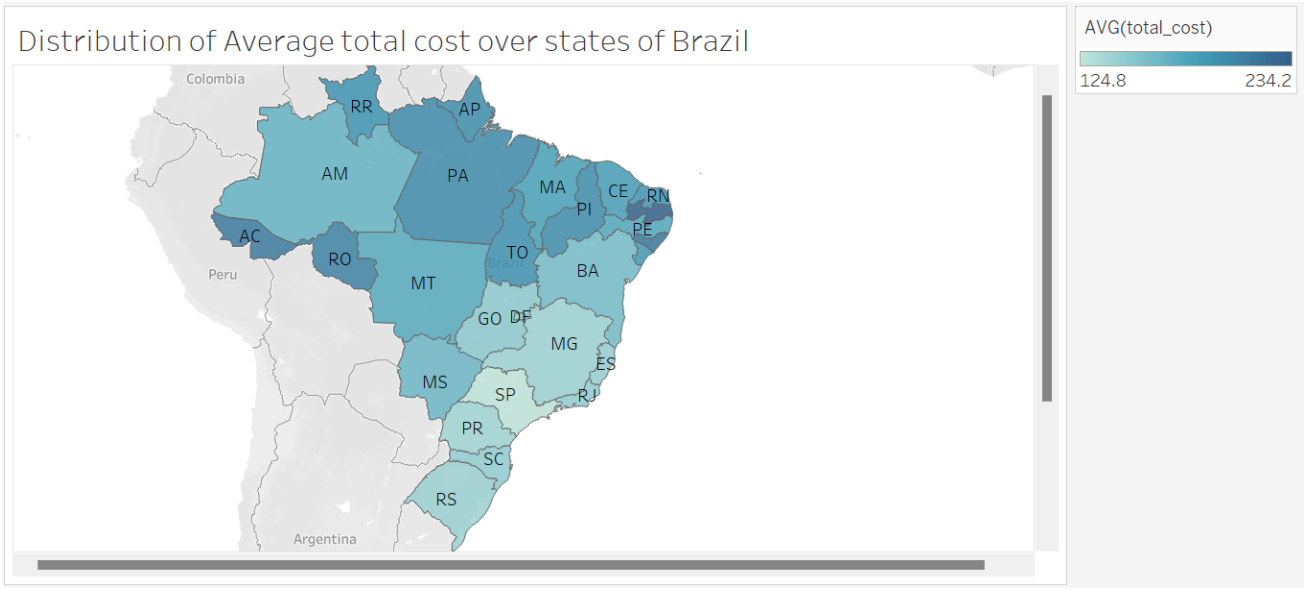
- The above graph shows the distribution of customers over 27 states in Brazil. Sao Paulo has the highest number of customers (40302) followed by Rio De Janeiro (12384) and Minas Gerais(11259).



- A growth rate of 143.4% is seen on the sum of total cost i.e. ∑(price + freight value) of orders from 2017 to 2018 when only the order of months from Jan to Aug is considered.

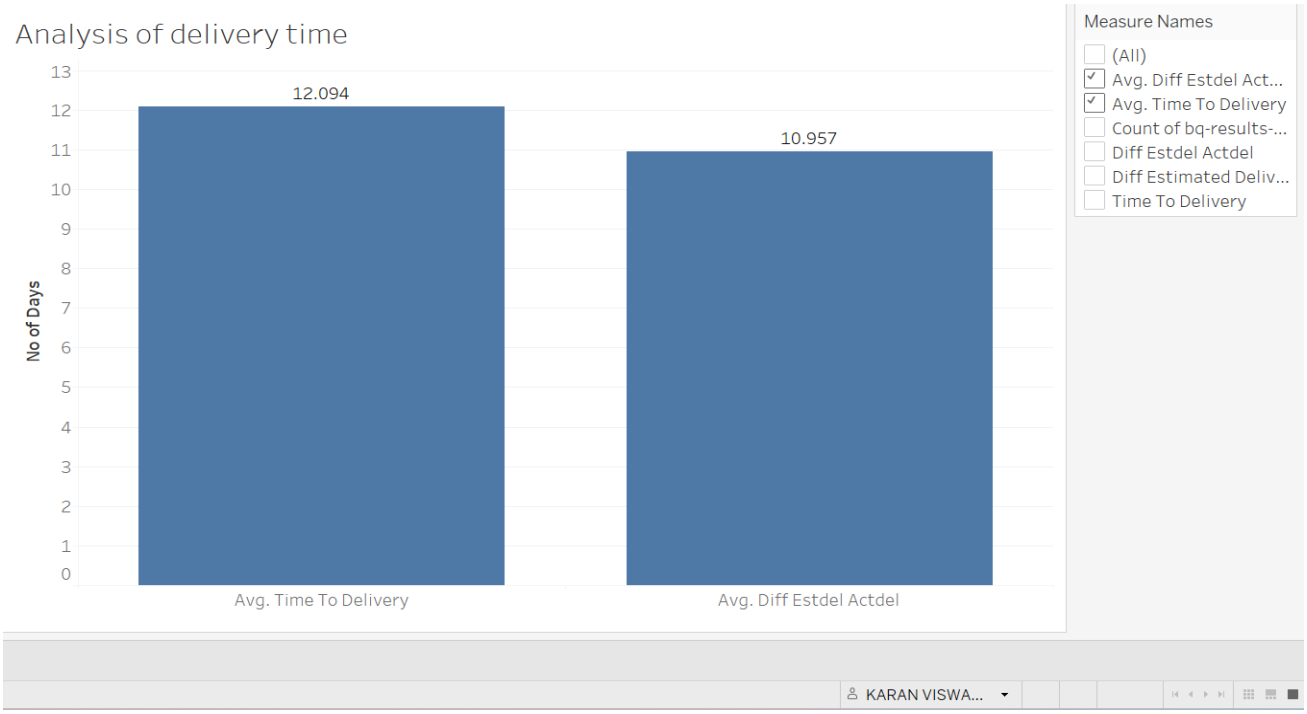


• The above map shows the distribution of the sum of total cost across 27 states of Brazil. The highest sum of the total cost for all the successfully delivered items is for the state of Sao Paulo (R$ 5769703.15) followed by Rio De Janeiro (R$ 2055401.57) and Minas Gerais (R$1818891.67) and the lowest figures are for the states Roraima (R$9039.52), Amapa (R$16141.81), Acre (R$19575.33).

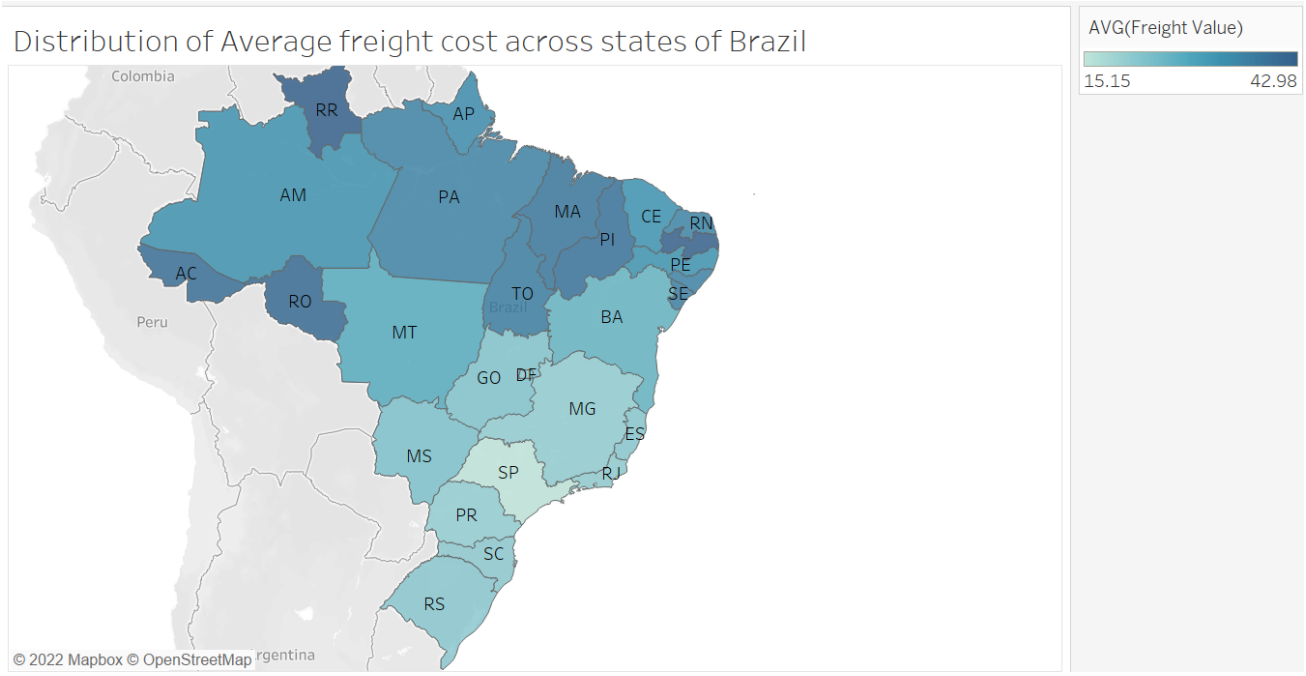


The average total cost is highest for the state Paraiba (R$ 235.22) followed by Alagoas (R$ 220.54) and Acre (R$ 215.11) and is the lowest for the states Sao Paulo (R$124.22), Parana (R$138.38) and Rio Grande do Sul (R$140.44). [Only the records of the successfully delivered items are considered

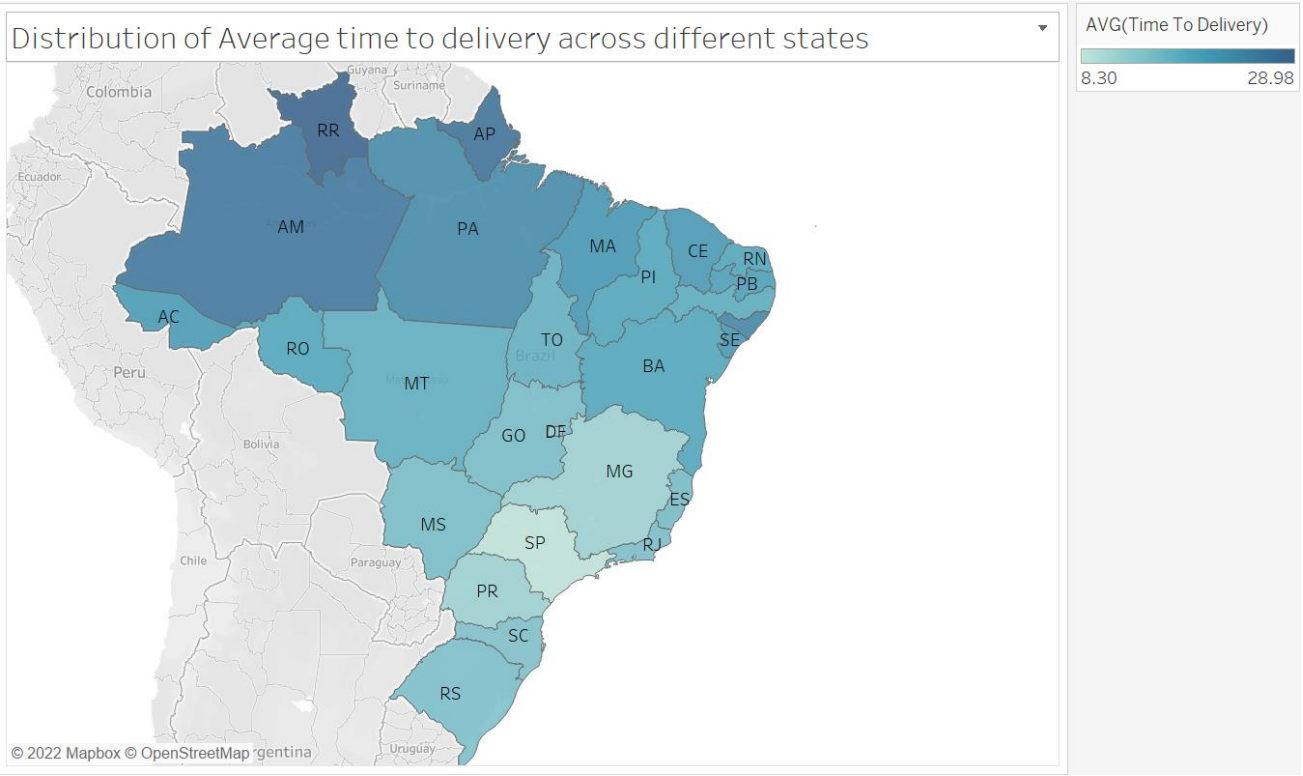
for the above analysis.]



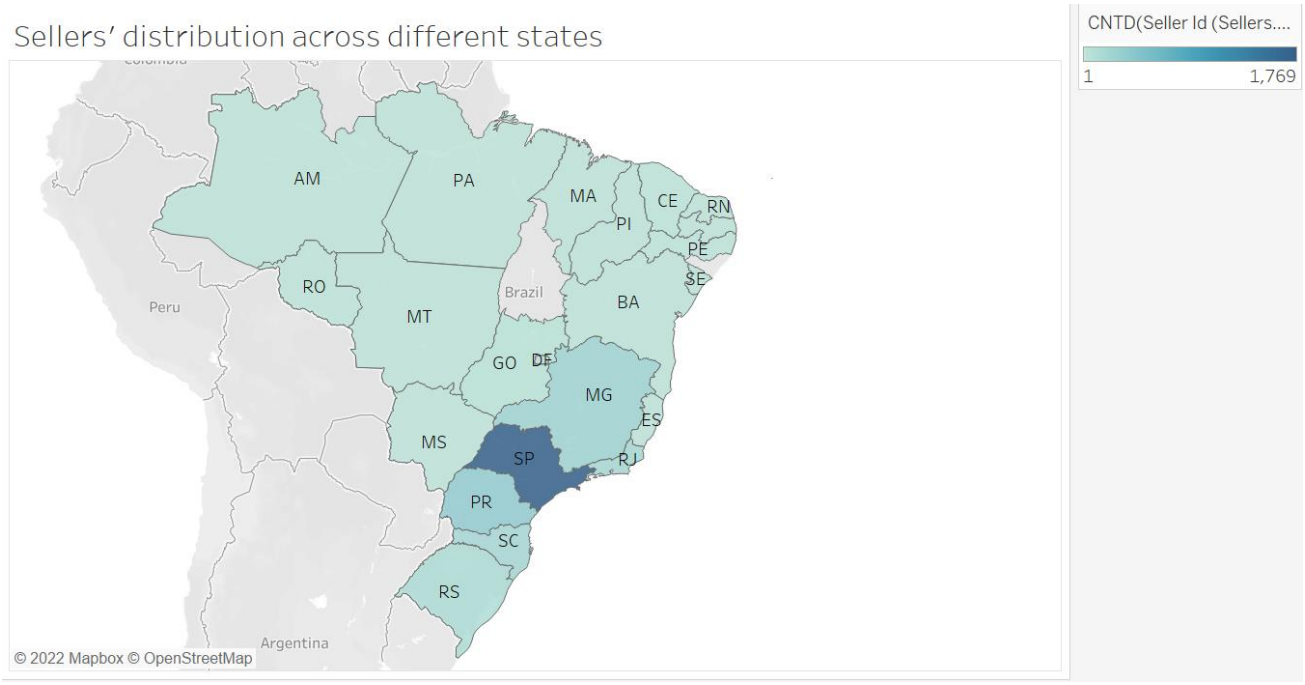
- From the above, the average days to deliver an item is 12.09 days and the orders are delivered at an average of 10.96 days before the estimated delivery date for all the successfully delivered items.



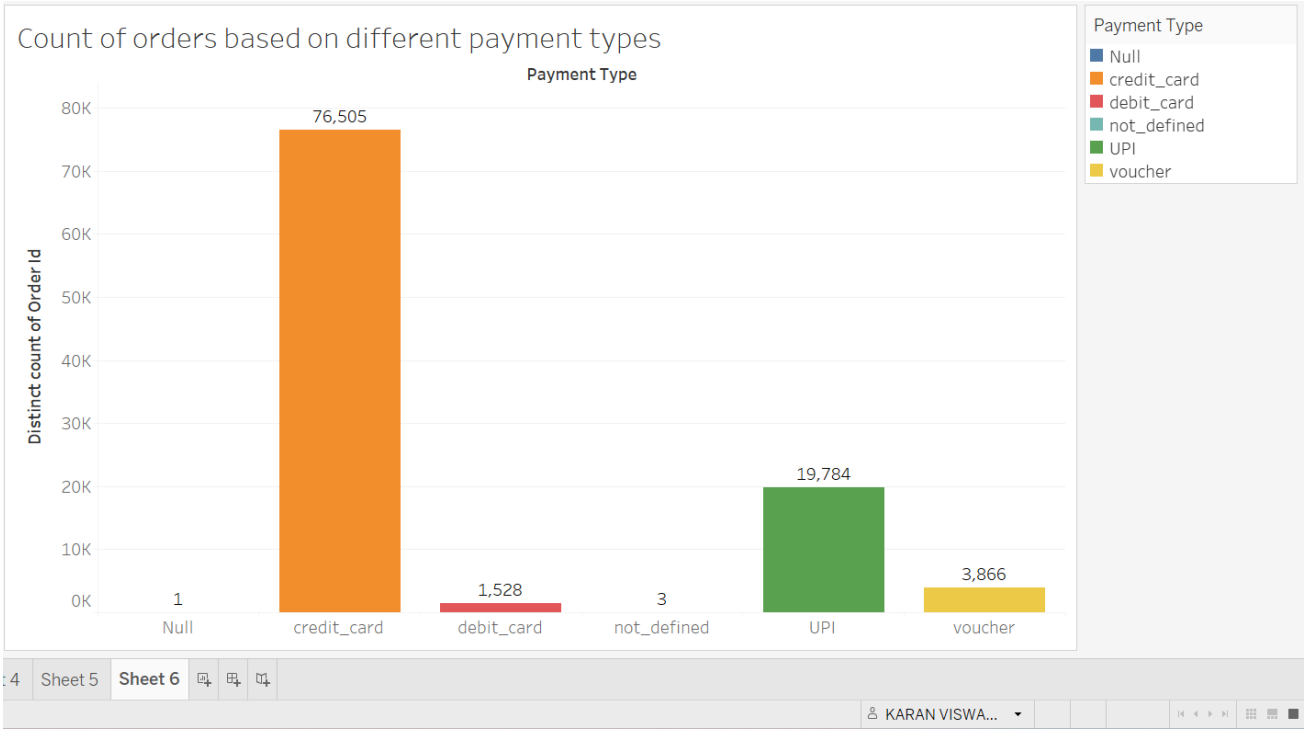
- The above graph shows the variation in average freight cost within different states of Brazil. The south-eastern states have the least values of average freight cost which shows that there is smooth transportation connectivity as well as a strong sellers network in these regions. Because of this, the freight cost is on the higher side for the northern states of the country.



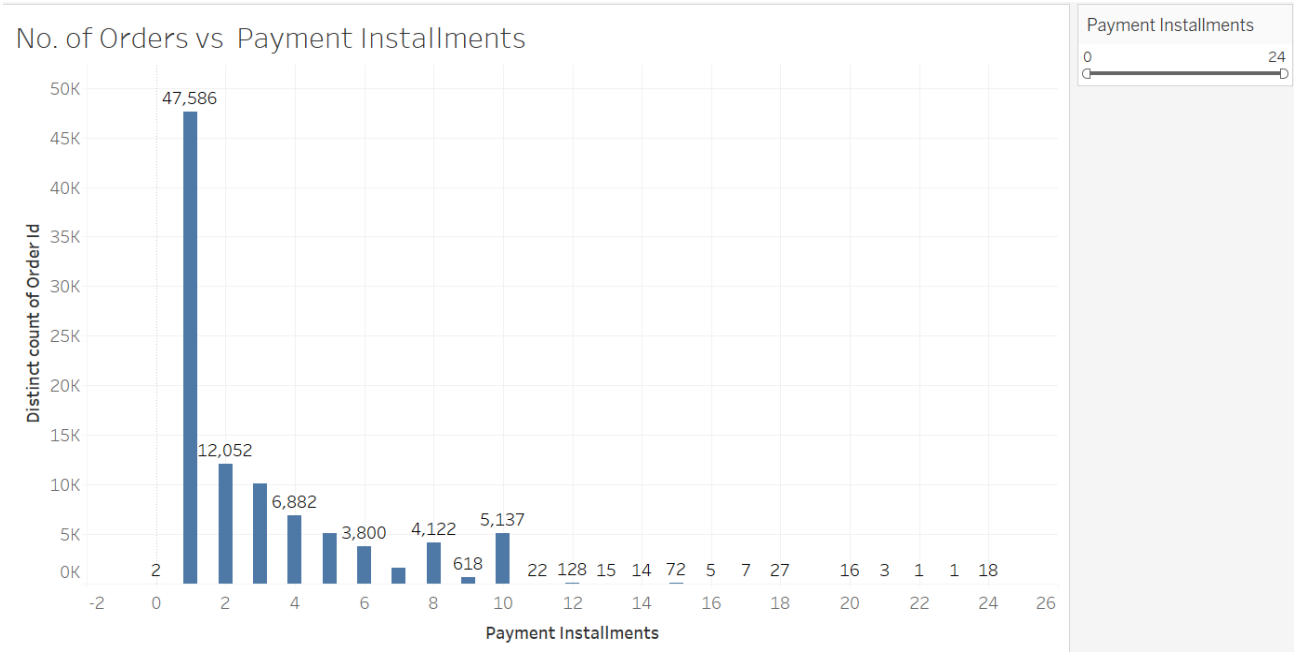
- The above variations are similar to the variations in the average freight cost. The customers of the northern states have to wait longer to receive the shipment as compared to the customers in the southern states.



- As can be seen from the above geographical map that most of the sellers are accumulated in the southern region of Brazil. The delivery time and the freight cost of the southern regions are therefore on the lower side than for the northern regions.



- It can be inferred from the above chart that credit card is the most preferred payment type by customers in Brazil followed by UPI and voucher.



- It can be seen in the graph that payment of a significant number of orders is made in small installments.

8. Recommendations:

- The monthly orders growth rate can be as high as 62.77% (Nov 2017) during the peak seasons. So to meet high demands, inventory levels must be adjusted accordingly to ensure that all the items are stocked at optimal levels. Else Target will lose sales from potential customers and the consumers will likely seek competitors to get what they need. By losing these sales, Target will also lose out on profits.
- Target should work on discount pricing strategies before the peak seasons to acquire new customers from the northern regions of Brazil where the customer count is very low, retain the customers in the southern parts of the state, increase sales, and promote new products. This will multiply the profit that Target normally makes.
- Since most of the orders are made at night and afternoon (77.45% of total orders made), Target has to make sure that their website runs smoothly during these times of the day. It should work on website personalization techniques for each customer based on their past interactions and preferences to improve their online shopping experience.
- As a significant number of orders (66.55% of total orders) come from three south-eastern states of Brazil i.e. Sao Paulo, Rio de Janeiro, and Minas Gerais so to reduce the freight cost and delivery time of the products, Target has to continue building a strong seller network to improve customer buying experience in such regions.
- To attract more customers from the northern regions of Brazil, Target has to work on the existing customer experience by reducing average freight cost and average time to delivery. Target has to take some measures to improve the supply base by attracting more vendors in these regions. There is ample scope for making profits from northern regions as well.
- Target has to develop a good social omnipresence. It should have footprints across all social media platforms to reach new potential customers and sellers. Since the count of customers and sellers is very less in most of the northern regions of Brazil.