

## SQL Target Project

### Introduction:

The given dataset is about Target which is one of the world's most recognized brands and one of America's leading retailers. The given dataset is all about Target, Brazil over a time period of 2016 to 2018. The given dataset has 8 csv files to explore and has details about sellers, orders, customers, payments, reviews, etc.

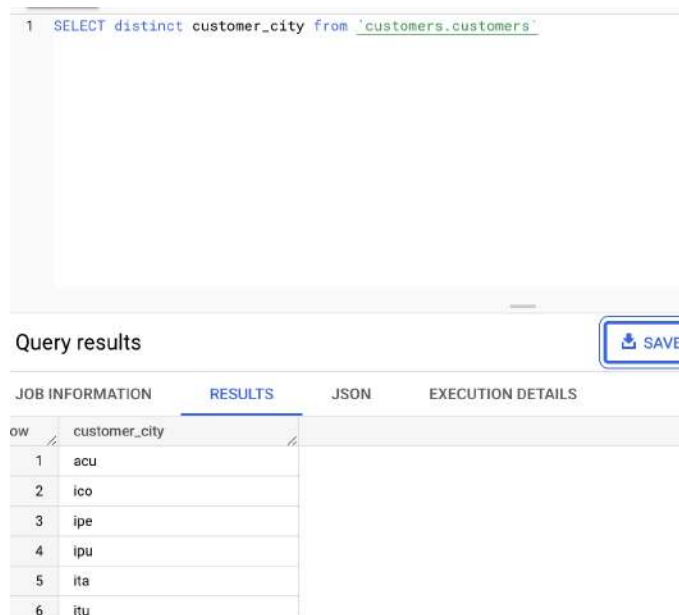
### Initial Analysis:

To find the Datatype:

Select column\_name as Field\_name, data\_type from information\_schema.columns where table\_schema = 'customers' and table\_name = 'customers'

To find distinct city:

**SELECT distinct customer\_city from  
Customers.customers**



The screenshot shows a SQL query execution interface. At the top, the query is displayed: `1 SELECT distinct customer_city from 'customers.customers'`. Below the query, there is a section titled "Query results" with a "SAVE" button. Underneath, there are tabs for "JOB INFORMATION", "RESULTS", "JSON", and "EXECUTION DETAILS". The "RESULTS" tab is selected, showing a table with two columns: "row" and "customer\_city". The table contains six rows of data.

row	customer_city
1	acu
2	ico
3	ipe
4	ipu
5	ita
6	itu

From the dataset we could get to know that Sellers are from 50 distinct cities and 23 distinct states while the customers are from 50 distinct cities, 27 distinct states. The geolocation has 50 distinct cities, 27 distinct states. There are about 50 different product categories, in which, the Art category with order\_item\_id 6 was the least purchased by customers and the bed table bath

with order\_item\_id 1 was the most purchased product. There are about 21 order\_item\_ids in total.

Field name	Type
customer_id	STRING
customer_unique_id	STRING
customer_zip_code_prefix	INTEGER
customer_city	STRING
customer_state	STRING

Field name	Type
seller_id	STRING
seller_zip_code_prefix	INTEGER
seller_city	STRING
seller_state	STRING

Field name	Type
geolocation_zip_code_prefix	INTEGER
geolocation_lat	FLOAT
geolocation_lng	FLOAT
geolocation_city	STRING
geolocation_state	STRING

Field name	Type
order_id	STRING
order_item_id	INTEGER
product_id	STRING
seller_id	STRING
shipping_limit_date	TIMESTAMP
price	FLOAT
freight_value	FLOAT

Field name	Type
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

Field name	Type
review_id	STRING
order_id	STRING
review_score	INTEGER
review_comment_title	STRING
review_creation_date	TIMESTAMP
review_answer_timestamp	TIMESTAMP

Field name	Type
order_id	STRING
payment_sequential	INTEGER
payment_type	STRING
payment_installments	INTEGER
payment_value	FLOAT

Field name	Type
product_id	STRING
product_category	STRING
product_name_length	INTEGER
product_description_length	INTEGER
product_photos_qty	INTEGER
product_weight_g	INTEGER
product_length_cm	INTEGER
product_height_cm	INTEGER
product_width_cm	INTEGER

To find about products purchased:

```
select order_item_id, count(distinct c.customer_id) as 'TotalCustomers',  
product_category, count(distinct p.product_id) as 'TotalProducts'  
from customers.customers as c join customers.orders as o on c.customer_id  
= o.customer_id join customers.order_items as ord on o.order_id = ord.order_id join  
customers.products as p on ord.product_id = p.product_id  
group by p.product_category, order_item_id  
order by count (distinct c.customer_id) desc
```

```

1 select order_item_id, count(distinct c.customer_id) as 'TotalCustomers', product_category, count(distinct p.
   product_id) as 'TotalProducts' from 'customers.customers' as c join 'customers.orders' as o on c.customer_id
   = o.customer_id join 'customers.order_items' as ord on o.order_id = ord.order_id join 'customers.products'
   as p on ord.product_id = p.product_id
2 group by p.product_category, order_item_id
3 order by count(distinct c.customer_id) desc

```

Press Alt+F1 for Accessibility Options.

## Query results

 SAVE RESULTS ▾

 EXPLORE DATA ▾



JOB INFORMATION		RESULTS	JSON	EXECUTION DETAILS	
Row	order_item_id	TotalCusto...	product_category	TotalProducts	
1	1	9311	bed table bath	2819	
2	1	8796	HEALTH BEAUTY	2381	
3	1	7681	sport leisure	2799	
4	1	6660	computer accessories	1604	
5	1	6355	Furniture Decoration	2488	
6	1	5829	housewares	2264	

To find about total orders and products:

```

select count(distinct o.order_id) as TotalOrders, count (distinct c.customer_id) as
TotalCustomers, count(distinct p.product_id) as TotalProducts
from customers.customers as c join customers.orders as o on c.customer_id = o.
customer_id join customers.order_items as ord on o.order_id = ord.order_id join
customers.products as p on ord.product_id = p.product_id

```

```

1 select count(distinct o.order_id) as 'TotalOrders', count(distinct c.customer_id) as 'TotalCustomers', count
  (distinct p.product_id) as 'TotalProducts' from 'customers.customers' as c join 'customers.orders' as o on c.
customer_id = o.customer_id join 'customers.order_items' as ord on o.order_id = ord.order_id join 'customers.
products' as p on ord.product_id = p.product_id

```

Press Alt+F1 for Accessibility Options.

Query results [SAVE RESULTS](#) [EXPLORE DATA](#)

JOB INFORMATION	RESULTS	JSON	EXECUTION DETAILS
Row	TotalOrders	TotalCusto...	TotalProducts
1	98666	98666	32951

6735 is the most expensive item under houseware category followed by PCs with 6729 purchased while 1.2 under health beauty category and 0.85 under construction tool category is the least expensive item. 17-10-2018 was the last order delivered and 17-10-2018 17:30:18 UTC was purchased. 2016-09-04 21:15:19 UTC was the first order purchased.

```

1 select count(*) as 'Count', avg(review_score) as 'Avgscore' from 'customers.order_reviews'
2 order by avg(review_score) desc
3
4
5
6
7 --where date(order_delivered_customer_date) is not null
8 -- , order_purchase_timestamp, order_delivered_customer_date, order_estimated_delivery_date
9 --time(order_purchase_timestamp) as 'Time', extract(month from order_purchase_timestamp) as 'Month',
10 --, date_diff(order_purchase_timestamp, order_delivered_customer_date, day) as 'time_to_delivery', date_diff
  (order_estimated_delivery_date, order_delivered_customer_date, day) as 'diff_estimated_delivery'
11

```

Press Alt+F1 for Accessibility Options.

Query results [SAVE RESULTS](#) [EXPLORE DATA](#)

JOB INFORMATION	RESULTS	JSON	EXECUTION DETAILS
Row	Count	Avgscore	
1	99224	4.08642062...	



Average review score given for the purchases by customers is 4.086. Most of the products are being sold by sellers from Sao Paulo.

```
1 select count(*) as 'Count', seller_city from 'customers.sellers'
2 group by seller_city
3 order by count(*) desc
4
5
6
7 --where date(order_delivered_customer_date) is not null
8 -- , order_purchase_timestamp, order_delivered_customer_date, order_estimated_delivery_date
9 --time(order_purchase_timestamp) as 'Time', extract(month from order_purchase_timestamp) as 'Month',
10 --, date_diff(order_purchase_timestamp, order_delivered_customer_date, day) as 'time_to_delivery', date_diff
    (order_estimated_delivery_date, order_delivered_customer_date, day) as 'diff_estimated_delivery'
11
```

Press Alt+F1 for Accessibility Options.

Query results [SAVE RESULTS](#) [EXPLORE DATA](#)

JOB INFORMATION		RESULTS	JSON	EXECUTION DETAILS
Row	Count	seller_city		
1	694	sao paulo		
2	127	curitiba		
3	96	rio de janeiro		
4	68	belo horizonte		
5	52	ribeirao preto		
6	50	quarulhos		

## In depth Exploration:

```
1 with x as(
2 select count(*) as 'Count', extract(date from order_purchase_timestamp) as 'Date', time
    (order_purchase_timestamp) as 'Time', case when time(order_purchase_timestamp) > '00:00:00' and time
    (order_purchase_timestamp) < '06:00:00' then 'Dawn'
3 when time(order_purchase_timestamp) > '06:00:00' and time(order_purchase_timestamp) < '12:00:00' then
    'Morning'
4 when time(order_purchase_timestamp) > '12:00:00' and time(order_purchase_timestamp) < '18:00:00' then
    'Evening'
5 else 'Night'
6 end 'DayTime'
7 from 'customers.orders'
8 group by DayTime, order_purchase_timestamp
9 order by Date
```

Press Alt+F1 for Accessibility Options.

Query results [SAVE RESULTS](#) [EXPLORE DATA](#)

JOB INFORMATION		RESULTS	JSON	EXECUTION DETAILS
Row	DayTime	Count		
1	Morning	22119		
2	Dawn	4723		
3	Evening	38129		
4	Night	33904		

Most purchases are made in the evening followed by night rather than in the morning hours. Least purchases are made at dawn.

```
1 with x as(
2   select count(*) as `Count`, extract(date from order_purchase_timestamp) as `Date`, time
   (order_purchase_timestamp) as `Time`, extract(month from order_purchase_timestamp) as `Month`
3   from `customers.orders`
4   group by Month, order_purchase_timestamp
5   order by Date)
6
7   select x.Month, count(*) as `Count` from x
8   group by x.Month
9   order by Count desc
10
```

Press Alt+F1 for Accessibility Options.

Query results [SAVE RESULTS](#) [EXPLORE DATA](#)

JOB INFORMATION		RESULTS	JSON	EXECUTION DETAILS
Row	Month	Count		
1	8	10780		
2	5	10516		
3	7	10256		
4	3	9825		
5	6	9374		
6	4	9297		

Most purchases were made in the 8th month i.e. August followed by may followed by July whereas least purchases are made in the month of september and october. There is a sudden decrease in purchase after August when grouped by month. In the year 2016 most purchases were made in October, in 2017 most of the purchases were made in November whereas in 2018 most of the purchases were made in March.

<pre> 1 with x as( 2 select count(*) as `Count`, extract(year from order_purchase_timestamp) as `Year`, extract(month from    order_purchase_timestamp) as `Month`, payment_value 3 from `customers.orders` as o join `customers.payments` as p on o.order_id = p.order_id 4 group by Year, Month, order_purchase_timestamp, payment_value) 5 6 7 select x.Year, x.Month, sum(x.payment_value) as `Totalvalue`, count(*) as `Count` from x 8 group by x.Year, x.Month 9 order by Totalvalue desc 10 11 --time(order_purchase_timestamp) as `Time`, extract(month from order_purchase_timestamp) as `Month`, 12 </pre>					
Press Alt+F1 for Accessibility Options.					
<div> <div>Query results</div> <div> <div>SAVE RESULTS</div> <div>EXPLORE DATA</div> </div> </div>					
JOB INFORMATION		RESULTS	JSON	EXECUTION DETAILS	
Row	Year	Month	Totalvalue	Count	
1	2017	11	1193578.98...	7813	
2	2018	3	1158618.79...	7473	
3	2018	4	1158409.36...	7166	
4	2018	5	1152587.55...	7078	
5	2018	1	1113872.15...	7528	
6	2018	7	1065132.26...	6475	

Evolution of E-commerce orders in the Brazil region:

Most purchases are made by people from Sao Paulo from SP state followed by Rio de janeiro from RJ state while least number of purchases are made in Caetanos from BA state, Jataizinho from PR state, Sao domingos do sul from RS state, Pirapemas from MA state and Alto bela vista from SC state.

```
1 WITH x AS(
2   select count(*) as 'Count', extract(date from order_purchase_timestamp) as 'Date', time
   (order_purchase_timestamp) as 'Time', extract(month from order_purchase_timestamp) as 'Month',
   customer_city, customer_state
3   from 'customers.orders' as o join customers.customers as c on o.customer_id = c.customer_id
4   group by Month, order_purchase_timestamp, customer_city, customer_state)
5
6
7   select x.customer_city, x.customer_state, count(*) as 'Count' from x
8   group by x.customer_city, x.customer_state
9   order by count desc
10
11
```

Press Alt+F1 for Accessibility Options.

### Query results

SAVE RESULTS

EXPLORE DATA

JOB INFORMATION		RESULTS	JSON	EXECUTION DETAILS	
Row	customer_city	customer_state	Count		
4305	lagoa santa	GO	1		
4306	caetanos	BA	1		
4307	jataizinho	PR	1		
4308	sao domingos do sul	RS	1		
4309	pirapemas	MA	1		
4310	alto bela vista	SC	1		

```
1 select extract(month from order_purchase_timestamp) as 'Month', customer_city, customer_state, count(*) as
   'Count' from customers.customers as c join customers.orders as o on c.customer_id = o.customer_id
2   group by extract(month from order_purchase_timestamp), customer_city, customer_state
3
4
5
6
7
8
9 --where date(order_delivered_customer_date) is not null
10 -- , order_purchase_timestamp, order_delivered_customer_date, order_estimated_delivery_date
11 --time(order_purchase_timestamp) as 'Time', extract(month from order_purchase_timestamp) as 'Month',
12 -- date diff(order_purchase_timestamp, order_delivered_customer_date, day) as 'time to delivery' date diff
   Press Alt+F1 for Accessibility Options.
```

### Query results

SAVE RESULTS

EXPLORE DATA

JOB INFORMATION		RESULTS	JSON	EXECUTION DETAILS	
Row	Month	customer_city	customer_state	Count	
1	1	acu	RN	1	
2	12	acu	RN	1	
3	5	acu	RN	1	
4	2	ico	CE	1	
5	3	ico	CE	1	
6	5	ico	CE	4	

Impact on Economy:

Highest payment value was 13664.08 made by a customer from RJ state in Rio de janeiro via Credit card followed by a value of 7274.88 from vila velha from the ES state via UPI and the value of 0.14 through voucher from SP state campinas city and Sao Paulo.

```
1 WITH x AS(
2   select count(*) as 'Count', extract(date from order_purchase_timestamp) as 'Date', time
   (order_purchase_timestamp) as 'Time', extract(month from order_purchase_timestamp) as 'Month',
   customer_city, customer_state, payment_value, payment_type
3   from 'customers.orders' as o join customers.customers as c on o.customer_id = c.customer_id join 'customers.
   payments' as p on o.order_id = p.order_id
4   group by Month, order_purchase_timestamp, customer_city, customer_state, payment_value, payment_type)
5
6
7   select x.customer_city, x.customer_state, x.payment_value, x.payment_type, count(*) as 'Count' from x
8   group by x.customer_city, x.customer_state, x.payment_value, x.payment_type
9   order by payment_value desc
10
```

Query results SAVE RESULTS EXPLORE DATA

JOB INFORMATION   RESULTS   JSON   EXECUTION DETAILS					
Row	customer_city	customer_state	payment_va...	payment_type	Count
1	rio de janeiro	RJ	13664.08	credit_card	1
2	vila velha	ES	7274.88	UPI	1
3	campo grande	MS	6929.31	credit_card	1
4	vitoria	ES	6922.21	UPI	1
5	marilia	SP	6726.66	UPI	1

```
1 WITH x AS(
2   select count(*) as 'Count', extract(date from order_purchase_timestamp) as 'Date', time
   (order_purchase_timestamp) as 'Time', extract(month from order_purchase_timestamp) as 'Month',
   customer_city, customer_state, payment_value, payment_type
3   from 'customers.orders' as o join customers.customers as c on o.customer_id = c.customer_id join 'customers.
   payments' as p on o.order_id = p.order_id
4   group by Month, order_purchase_timestamp, customer_city, customer_state, payment_value, payment_type)
5
6
7   select x.customer_city, x.customer_state, x.payment_value, x.payment_type, count(*) as 'Count' from x
8   group by x.customer_city, x.customer_state, x.payment_value, x.payment_type
9   order by payment_value desc
10
```

Query results SAVE RESULTS EXPLORE DATA

JOB INFORMATION   RESULTS   JSON   EXECUTION DETAILS					
Row	customer_city	customer_state	payment_va...	payment_type	Count
92401	santa barbara d'oeste	SP	0.15	voucher	1
92402	sao paulo	SP	0.15	credit_card	1
92403	sao paulo	SP	0.15	voucher	1
92404	campinas	SP	0.14	voucher	1
92405	sao paulo	SP	0.14	voucher	1

Customers has made 16097 purchases over the period from Sao Paulo for a total value of 2199273.6699999976 followed by 7159 purchases from Rio de janeiro for a value of

1159848.8099999912 and the least value of 20.7 from Polo petroquimico de triunfo and 20.42 from Sabaudia.

```

1 with x as(
2 select count(*) as 'Count', extract(date from order_purchase_timestamp) as 'Date', time
   (order_purchase_timestamp) as 'Time', extract(month from order_purchase_timestamp) as 'Month',
   customer_city, customer_state, payment_value, payment_type
3 from 'customers.orders' as o join customers.customers as c on o.customer_id = c.customer_id join 'customers.
   payments' as p on o.order_id = p.order_id
4 group by Month, order_purchase_timestamp, customer_city, customer_state, payment_value, payment_type)
5
6
7 select x.customer_city, sum(x.payment_value) as 'Totalvalue', count(*) as 'Count' from x
8 group by x.customer_city
9 order by Count desc
10

```

Press Alt+F1 for Accessibility Options.

**Query results** [SAVE RESULTS](#) [EXPLORE DATA](#)

JOB INFORMATION		RESULTS	JSON	EXECUTION DETAILS
Row	customer_city	Totalvalue	Count	
1	sao paulo	2199273.66...	16097	
2	rio de janeiro	1159848.80...	7159	
3	belo horizonte	421514.359...	2863	
4	brasilia	354036.840...	2187	
5	curitiba	247208.819...	1567	
6	campinas	214728.949...	1501	

The sales in the year 2016 has increased in count from 345 over a value of 59342.340000000026 to 47175 orders over a value of 7238583.33999996888 in 2017 to 55709 orders over a value of 8689008.9999998715 in 2018.



```

1 with x as(
2 select count(*) as `Count`, extract(year from order_purchase_timestamp) as `Year`, payment_value,
   payment_type
3 from `customers.orders` as o join `customers.payments` as p on o.order_id = p.order_id
4 group by Year, order_purchase_timestamp, payment_value, payment_type)
5
6
7 select x.Year, sum(x.payment_value) as `Totalvalue`, count(*) as `Count` from x
8 group by x.Year
9 order by Totalvalue desc
10
11 --time(order_purchase_timestamp) as `Time`, extract(month from order_purchase_timestamp) as `Month`,
12

```

Press Alt+F1 for Accessibility Options.

## Query results

[SAVE RESULTS](#)

[EXPLORE DATA](#)



JOB INFORMATION		RESULTS	JSON	EXECUTION DETAILS	
Row	Year	Totalvalue	Count		
1	2018	8689008.9999998715	55709		
2	2017	7238583.3399996888	47175		
3	2016	59342.340000000026	345		

```

1 with x as(
2 select count(*) as `Count`, extract(year from order_purchase_timestamp) as `Year`, extract(month from
   order_purchase_timestamp) as `Month`, payment_value, payment_type
3 from `customers.orders` as o join `customers.payments` as p on o.order_id = p.order_id
4 group by Year, order_purchase_timestamp, payment_value, payment_type)
5
6
7 select x.Year, sum(x.payment_value) as `Totalvalue`, x.payment_type, count(*) as `Count` from x
8 group by x.Year, x.payment_type
9 order by Totalvalue desc
10
11 --time(order_purchase_timestamp) as `Time`, extract(month from order_purchase_timestamp) as `Month`,
12

```

Press Alt+F1 for Accessibility Options.

## Query results

[SAVE RESULTS](#)

[EXPLORE DATA](#)



JOB INFORMATION		RESULTS	JSON	EXECUTION DETAILS	
Row	Year	Totalvalue	payment_type	Count	
1	2018	6854629.50...	credit_card	41946	
2	2017	5634452.67...	credit_card	34545	
3	2018	1462034.87...	UPI	10207	
4	2017	1395934.47...	UPI	9507	
5	2018	198022.479...	voucher	2449	
6	2018	174322.139...	debit_card	1104	

Over the period 2016-18 most of the payment is being made by credit card.

1	with x as(
2	select count(*) as 'Count', extract(year from order_purchase_timestamp) as 'Year', extract(month from
3	order_purchase_timestamp) as 'Month', payment_value
4	from 'customers.orders' as o join 'customers.payments' as p on o.order_id = p.order_id
5	where extract(month from order_purchase_timestamp) in (1,2,3,4,5,6,7,8) and extract(year from
6	order_purchase_timestamp) in (2017, 2018)
7	group by Year, Month, order_purchase_timestamp, payment_value)
8	select x.Year, sum(x.payment_value) as 'Totalvalue', count(*) as 'Count' from x
9	group by x.Year
10	order by Totalvalue desc
11	

Press Alt+F1 for Accessibility Options.

Query results
SAVE RESULTS
EXPLORE DATA

JOB INFORMATION		RESULTS	JSON	EXECUTION DETAILS
Row	Year	Totalvalue	Count	
1	2018	8683979.7899998687	55689	
2	2017	3662988.0399999255	24171	

In the year 2017 the total orders from the month Jan to Aug was 24171 over a value of 3662988.0399999255 and increased to 55689 over a value of 8683979.7899998687 in 2018. 57.82% increase in total cost of orders from 2017 to 2018 for the months Jan to August.

1	with x as(
2	select count(*) as 'Count', extract(year from order_purchase_timestamp) as 'Year', extract(month from
3	order_purchase_timestamp) as 'Month', payment_value
4	from 'customers.orders' as o join 'customers.payments' as p on o.order_id = p.order_id
5	where extract(month from order_purchase_timestamp) in (1,2,3,4,5,6,7,8) and extract(year from
6	order_purchase_timestamp) in (2017, 2018)
7	group by Year, Month, order_purchase_timestamp, payment_value)
8	select x.Year, x.Month, sum(x.payment_value) as 'Totalvalue', count(*) as 'Count' from x
9	group by x.Year, x.Month
10	order by Totalvalue desc
11	

Press Alt+F1 for Accessibility Options.

Query results
SAVE RESULTS
EXPLORE DATA

JOB INFORMATION		RESULTS	JSON	EXECUTION DETAILS
Row	Year	Month	Totalvalue	Count
1	2018	3	1158618.7999999891	7473
2	2018	4	1158409.3699999955	7166
3	2018	5	1152587.5599999984	7078
4	2018	1	1113872.150000002	7528
5	2018	7	1065132.2600000007	6475
6	2018	6	1022294.3599999971	6373



```

1 select avg(freight_value) as 'Meanfreight', sum(freight_value) as 'Totalfreight', avg(price) as 'Meanprice',
   sum(price) as 'Totalprice', customer_state from 'customers.customers' as c join 'customers.orders' as o on c.
   customer_id = o.customer_id join 'customers.order_items' as ord on o.order_id = ord.order_id
2 group by customer_state
3 order by customer_state
4
5
6
7
8 --where date(order_delivered_customer_date) is not null
9 -- , order_purchase_timestamp, order_delivered_customer_date, order_estimated_delivery_date
10 --time(order_purchase_timestamp) as 'Time', extract(month from order_purchase_timestamp) as 'Month',
11 -- date_diff(order_purchase_timestamp, order_delivered_customer_date, day) as 'time to delivery', date_diff

```

Press Alt+F1 for Accessibility Options.

## Query results

[SAVE RESULTS](#)
[EXPLORE DATA](#)


JOB INFORMATION      RESULTS      JSON      EXECUTION DETAILS

Row	Meanfreight	Totalfreight	Meanprice	Totalprice	customer_state
1	40.0733695...	3686.74999...	173.727717...	15982.9499...	AC
2	35.8436711...	15914.5899...	180.889211...	80314.81	AL
3	33.2053939...	5478.88999...	135.495999...	22356.8400...	AM
4	34.0060975...	2788.50000...	164.320731...	13474.2999...	AP
5	26.3639589...	100156.679...	134.601208...	511349.990...	BA
6	32.7142016...	48351.5899...	153.758261...	227254.709...	CE

## Analysis on sales, freight and delivery time:

```

1 select customer_state, avg(freight_value) as 'MeanFreight', avg(date_diff(order_purchase_timestamp,
   order_delivered_customer_date, day)) as 'Avg_time_to_del', avg(date_diff(order_estimated_delivery_date,
   order_delivered_customer_date, day)) as 'Avg_diff_estimated_del'
2 from 'customers.orders' as o join 'customers.order_items' as ord on o.order_id = ord.order_id join
   'customers.customers' as c on o.customer_id = c.customer_id
3 group by customer_state
4 order by avg(date_diff(order_purchase_timestamp, order_delivered_customer_date, day))
5 limit 5
6
7
8
9
10 --where date(order_delivered_customer_date) is not null

```

Press Alt+F1 for Accessibility Options.

## Query results

[SAVE RESULTS](#)
[EXPLORE DATA](#)


JOB INFORMATION      RESULTS      JSON      EXECUTION DETAILS

Row	customer_state	MeanFreight	Avg_time_to...	Avg_diff_est...
1	RR	42.9844230...	-27.826086...	17.4347826...
2	AP	34.0060975...	-27.753086...	17.4444444...
3	AM	33.2053939...	-25.963190...	18.9754601...
4	AL	35.8436711...	-23.992974...	7.97658079...
5	PA	35.8326851...	-23.301707...	13.3747628...

```

1 with x as(
2 | select order_id, customer_id, date_diff(order_purchase_timestamp, order_delivered_customer_date, day) as
   'time_to_delivery', date_diff(order_estimated_delivery_date, order_delivered_customer_date, day) as
   'diff_estimated_delivery'
3 from 'customers.orders'
4 )
5
6 select customer_state, avg(freight_value) as 'MeanFreight', avg(x.time_to_delivery) as 'Avg_time_to_del', avg
   (x.diff_estimated_delivery) as 'Avg_diff_estimated_del' from x join 'customers.order_items' as ord on x.
   order_id = ord.order_id join 'customers.customers' as c on x.customer_id = c.customer_id
7 group by customer_state
8 order by customer_state
9 --where date(order_delivered_customer_date) is not null

```

Press Alt+F1 for Accessibility Options.

## Query results

 SAVE RESULTS ▾

 EXPLORE DATA ▾



JOB INFORMATION		RESULTS	JSON	EXECUTION DETAILS	
Row	customer_state	MeanFreight	Avg_time_to_del	Avg_diff_estimated_del	
1	AC	40.073369565217405	-20.329670329670336	20.010989010989018	
2	AL	35.843671171171152	-23.992974238875881	7.9765807962529349	
3	AM	33.205393939393936	-25.963190184049076	18.975460122699381	
4	AP	34.006097560975618	-27.753086419753075	17.444444444444443	
5	BA	26.363958936562248	-18.774640238935675	10.119467825142538	
6	CE	32.714201623815995	-20.537166900420793	10.256661991584851	

```

2 | select order_id, customer_id, date_diff(order_purchase_timestamp, order_delivered_customer_date, day) as
   'time_to_delivery', date_diff(order_estimated_delivery_date, order_delivered_customer_date, day) as
   'diff_estimated_delivery'
3 from 'customers.orders'
4 )
5
6 select customer_state, avg(freight_value) as 'MeanFreight', avg(x.time_to_delivery) as 'Avg_time_to_del', avg
   (x.diff_estimated_delivery) as 'Avg_diff_estimated_del' from x join 'customers.order_items' as ord on x.
   order_id = ord.order_id join 'customers.customers' as c on x.customer_id = c.customer_id
7 group by customer_state
8 order by avg(freight_value) desc
9 limit 5
10 --where date(order_delivered_customer_date) is not null

```

Press Alt+F1 for Accessibility Options.

## Query results

 SAVE RESULTS ▾

 EXPLORE DATA ▾



JOB INFORMATION		RESULTS	JSON	EXECUTION DETAILS	
Row	customer_state	MeanFreight	Avg_time_to...	Avg_diff_est...	
1	RR	42.9844230...	-27.826086...	17.4347826...	
2	PB	42.7238039...	-20.119453...	12.1501706...	
3	RO	41.0697122...	-19.282051...	19.0805860...	
4	AC	40.0733695...	-20.329670...	20.0109890...	
5	PI	39.1479704...	-18.931166...	10.6826003...	

```

2 | select order_id, customer_id, date_diff(order_purchase_timestamp, order_delivered_customer_date, day) as
   | 'time_to_delivery', date_diff(order_estimated_delivery_date, order_delivered_customer_date, day) as
   | 'diff_estimated_delivery'
3 | from 'customers.orders'
4 | )
5 |
6 | select customer_state, avg(freight_value) as 'MeanFreight', avg(x.time_to_delivery) as 'Avg_time_to_del', avg
   | (x.diff_estimated_delivery) as 'Avg_diff_estimated_del' from x join 'customers.order_items' as ord on x.
   | order_id = ord.order_id join 'customers.customers' as c on x.customer_id = c.customer_id
7 | group by customer_state
8 | order by avg(freight_value)
9 | limit 5
10 | --where date(order_delivered_customer_date) is not null

```

Press Alt+F1 for Accessibility Options.

## Query results

 SAVE RESULTS ▾

 EXPLORE DATA ▾



JOB INFORMATION		RESULTS	JSON	EXECUTION DETAILS	
Row	customer_state	MeanFreight	Avg_time_to...	Avg_diff_est...	
1	SP	15.1472753...	-8.2596085...	10.2655943...	
2	PR	20.5316515...	-11.480793...	12.5338998...	
3	MG	20.6301668...	-11.515522...	12.3971510...	
4	RJ	20.9609239...	-14.689382...	11.1444931...	
5	DF	21.0413549...	-12.501486...	11.2747346...	

```

1 | select customer_state, avg(freight_value) as 'MeanFreight', avg(date_diff(order_purchase_timestamp,
   | order_delivered_customer_date, day)) as 'Avg_time_to_del', avg(date_diff(order_estimated_delivery_date,
   | order_delivered_customer_date, day)) as 'Avg_diff_estimated_del'
2 | from 'customers.orders' as o join 'customers.order_items' as ord on o.order_id = ord.order_id join
   | 'customers.customers' as c on o.customer_id = c.customer_id
3 | group by customer_state
4 | order by avg(date_diff(order_purchase_timestamp, order_delivered_customer_date, day)) desc, avg(date_diff
   | (order_estimated_delivery_date, order_delivered_customer_date, day)) desc
5 | limit 5
6 |
7 |
8 |
9 |

```

Press Alt+F1 for Accessibility Options.

## Query results

 SAVE RESULTS ▾

 EXPLORE DATA ▾



JOB INFORMATION		RESULTS	JSON	EXECUTION DETAILS	
Row	customer_state	MeanFreight	Avg_time_to...	Avg_diff_est...	
1	SP	15.1472753...	-8.2596085...	10.2655943...	
2	PR	20.5316515...	-11.480793...	12.5338998...	
3	MG	20.6301668...	-11.515522...	12.3971510...	
4	DF	21.0413549...	-12.501486...	11.2747346...	
5	SC	21.4703687...	-14.520985...	10.6688628...	

```

1 select distinct date(order_purchase_timestamp) as 'PurchaseDate', date(order_delivered_customer_date) as
   'CustomerDelDate', date(order_estimated_delivery_date) as 'EstimatedDelDate', date_diff(
   order_purchase_timestamp, order_delivered_customer_date, day) as 'time_to_delivery', date_diff(
   order_estimated_delivery_date, order_delivered_customer_date, day) as 'diff_estimated_delivery' from
   'customers.orders'
2 --where date(order_delivered_customer_date) is not null
3
4 --time(order_purchase_timestamp) as 'Time', extract(month from order_purchase_timestamp) as 'Month',
5
6

```

Press Alt+F1 for Accessibility Options.

## Query results

[SAVE RESULTS](#)

[EXPLORE DATA](#)



JOB INFORMATION		RESULTS	JSON	EXECUTION DETAILS	
Row	PurchaseDa...	CustomerDe...	EstimatedD...	time_to_deli...	diff_estimat...
1	2016-10-07	2016-10-14	2016-11-29	-7	45
2	2018-02-19	2018-03-21	2018-03-09	-30	-12
3	2016-10-09	2016-10-16	2016-11-30	-7	44
4	2016-10-08	2016-10-19	2016-11-30	-10	41
5	2017-04-11	2017-04-18	2017-05-18	-6	29
6	2017-03-17	2017-04-07	2017-05-18	-20	40

## Payment type analysis:

```

1 select payment_installments, count(*) as 'Count'
2 from 'customers.payments'
3 group by payment_installments
4 order by count(*) desc
5
6
7
8
9
10
11 --where date(order_delivered_customer_date) is not null
12 -- , order_purchase_timestamp, order_delivered_customer_date, order_estimated_delivery_date
13 --time(order_purchase_timestamp) as 'Time', extract(month from order_purchase_timestamp) as 'Month'

```

Press Alt+F1 for Accessibility Options.

## Query results

[SAVE RESULTS](#)

[EXPLORE DATA](#)



JOB INFORMATION		RESULTS	JSON	EXECUTION DETAILS	
Row	payment_installments	Count			
1	1	52546			
2	2	12413			
3	3	10461			
4	4	7098			
5	10	5328			
6	5	5239			



```

1 select payment_installments, count(*) as 'Count'
2 from 'customers.payments'
3 group by payment_installments
4 order by count(*)
5
6
7
8
9
10
11 --where date(order_delivered_customer_date) is not null
12 -- , order_purchase_timestamp, order_delivered_customer_date, order_estimated_delivery_date
13 --time(order_purchase_timestamp) as 'Time', extract(month from order_purchase_timestamp) as 'Month'

```

Press Alt+F1 for Accessibility Options.

## Query results

 SAVE RESULTS ▾

 EXPLORE DATA ▾



JOB INFORMATION		RESULTS	JSON	EXECUTION DETAILS
Row	payment_in...	Count		
1	22	1		
2	23	1		
3	0	2		
4	21	3		
5	16	5		
6	17	8		

```

1 select payment_type, count(*) as 'Count', extract(month from order_purchase_timestamp) as 'Month'
2 from 'customers.orders' as o join 'customers.payments' as p on o.order_id = p.order_id
3 group by extract(month from order_purchase_timestamp), payment_type
4
5
6
7
8
9
10 --where date(order_delivered_customer_date) is not null
11 -- , order_purchase_timestamp, order_delivered_customer_date, order_estimated_delivery_date
12 --time(order_purchase_timestamp) as 'Time', extract(month from order_purchase_timestamp) as 'Month',
13 -- date diff(order_purchase_timestamp, order_delivered_customer_date, day) as 'time to delivery', date diff

```

Press Alt+F1 for Accessibility Options.

## Query results

 SAVE RESULTS ▾

 EXPLORE DATA ▾



JOB INFORMATION		RESULTS	JSON	EXECUTION DETAILS
Row	payment_type	Count	Month	
1	UPI	1509	11	
2	credit_card	4378	12	
3	UPI	1723	2	
4	credit_card	5897	11	
5	voucher	572	4	
6	credit_card	7841	7	

## Insights:

From the given data we know that, Sellers are from 50 distinct cities and 23 distinct states while the customers are from 50 distinct cities, 27 distinct states. The geolocation has 50 distinct cities, 27 distinct states. There are about 50 different product categories, in which, the Art category with order\_item\_id 6 was the least purchased by customers and the bed table bath with order\_item\_id 1 was the most purchased product. There are about 21 order\_item\_ids in total. The average review score given by the customer for the purchases is 4.086. Most of the products are sold by sellers from Sao Paulo.

6735 is the most expensive item under houseware category followed by PCs with 6729 purchased while 1.2 under health beauty category and 0.85 under construction tool category is the least expensive item. 17-10-2018 was the last order delivered and 17-10-2018 17:30:18 UTC was purchased. 2016-09-04 21:15:19 UTC was the first order purchased. Most purchases are made in the evening followed by night rather than in the morning hours. Least purchases are made at dawn. Most purchases were made in the 8th month i.e. August followed by may followed by July whereas least purchases are made in the month of september and october. There is a sudden decrease in purchase after August when grouped by month. In the year 2016 most purchases were made in October, in 2017 most of the purchases were made in November whereas in 2018 most of the purchases were made in March.

Most purchases are made by people from Sao Paulo from SP state followed by Rio de janeiro from RJ state while least number of purchases are made in Caetanos from BA state, Jataizinho from PR state, Sao domingos do sul from RS state, Pirapemas from MA state and Alto bela vista from SC state. Highest payment value was 13664.08 made by a customer from RJ state in Rio de janeiro via Credit card followed by a value of 7274.88 from vila velha from the ES state via UPI and the value of 0.14 through voucher from SP state campinas city and Sao Paulo.

Customers has made 16097 purchases over the period from Sao Paulo for a total value of 2199273.6699999976 followed by 7159 purchases from Rio de janeiro for a value of 1159848.8099999912 and the least value of 20.7 from Polo petroquimico de triunfo and 20.42 from Sabaudia. The sales in the year 2016 has increased in count from 345 over a value of 59342.340000000026 to 47175 orders over a value of 7238583.3399996888 in 2017 to 55709 orders over a value of 8689008.9999998715 in 2018. Over the period 2016-18 most of the payment is being made by credit card.

In the year 2017 the total orders from the month Jan to Aug was 24171 over a value of 3662988.0399999255 and increased to 55689 over a value of 8683979.7899998687 in 2018. 57.82% increase in total cost of orders from 2017 to 2018 for the months Jan to August.

Most people pay the bill in a single installment. It is just one who pays the bill in 23 installments.

### **Recommendations:**

Bed table bath category items are most purchased and so the stock of these products can be increased and the art category products are the least purchased so these could be minimized. Most expensive items are sold under the houseware and PCs category which are not frequently

bought. So these products could be made least available and some discounts can be given probably to increase the sales.

The avg review score given by the customers for the purchases could be increased by giving even faster delivery of products, better product discounts, decreasing freight value, more credit card discounts since the most used payment option is credit card, etc.

New sellers could be deployed from Sao Paulo since most of the sellers are from there.

Most of the products like are being purchased in the evening time, so exciting offers and more products could be displayed for sale during these hours which is between 12:00 noon to 18:00 hours.

Sao Paulo from SP state followed by Rio de janeiro from RJ state is where most of the products are being purchased. These units could be enlarged and enhanced with more products.

Vouchers are being least used for purchases, so more vouchers could be distributed to the users when billed over a particular range so that the user visits the store next time again for purchase. This way sales as well as brand market could be improved.

Most new products could be launched during the period between August to March especially during November, December, March since most purchases are being made at that time which probably may be due to christmas and holiday season.