Name – Krishna Vishwakarma

Batch – DSML Sept Beginner 2023

Business Case: Target SQL

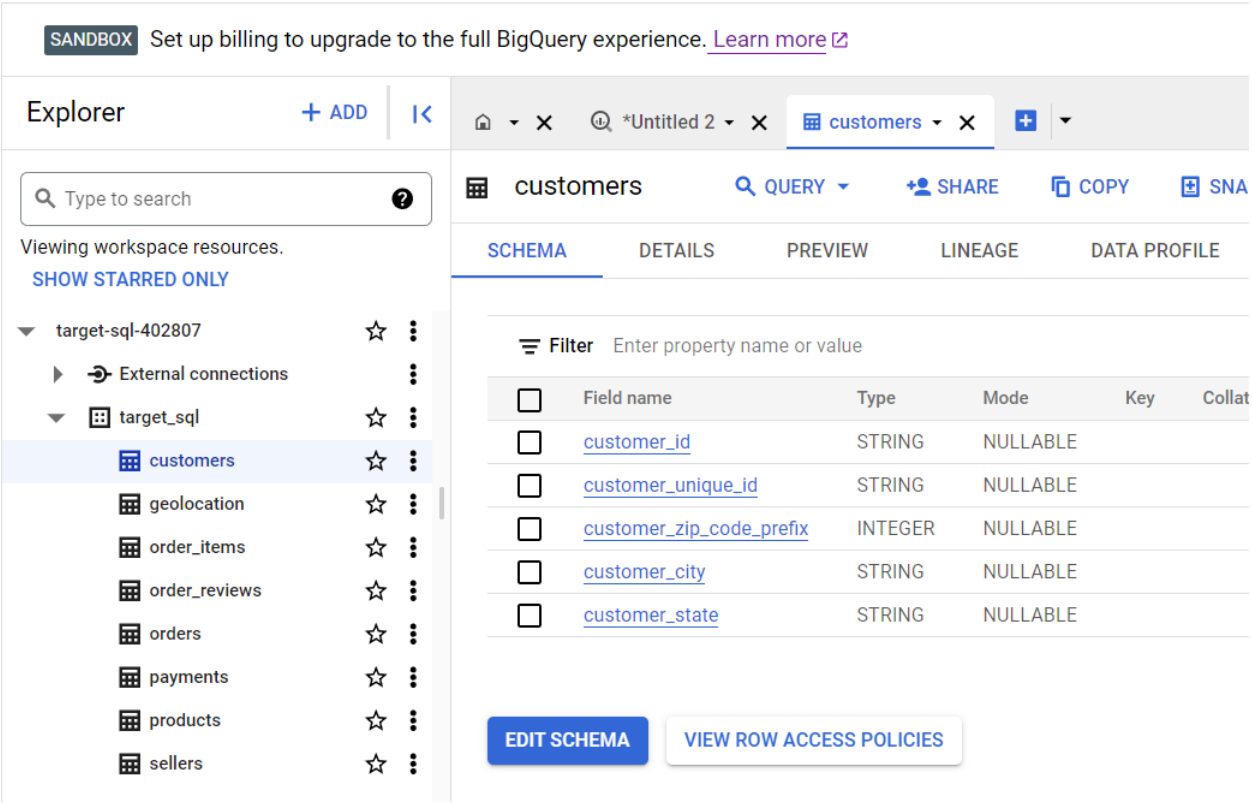
Problem Statement:

Assuming you are a data analyst/ scientist at Target, you have been assigned the task of analyzing the given dataset to extract valuable insights and provide actionable recommendations.

What does 'good' look like?

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

    A. Data type of all columns in the "customers" table.

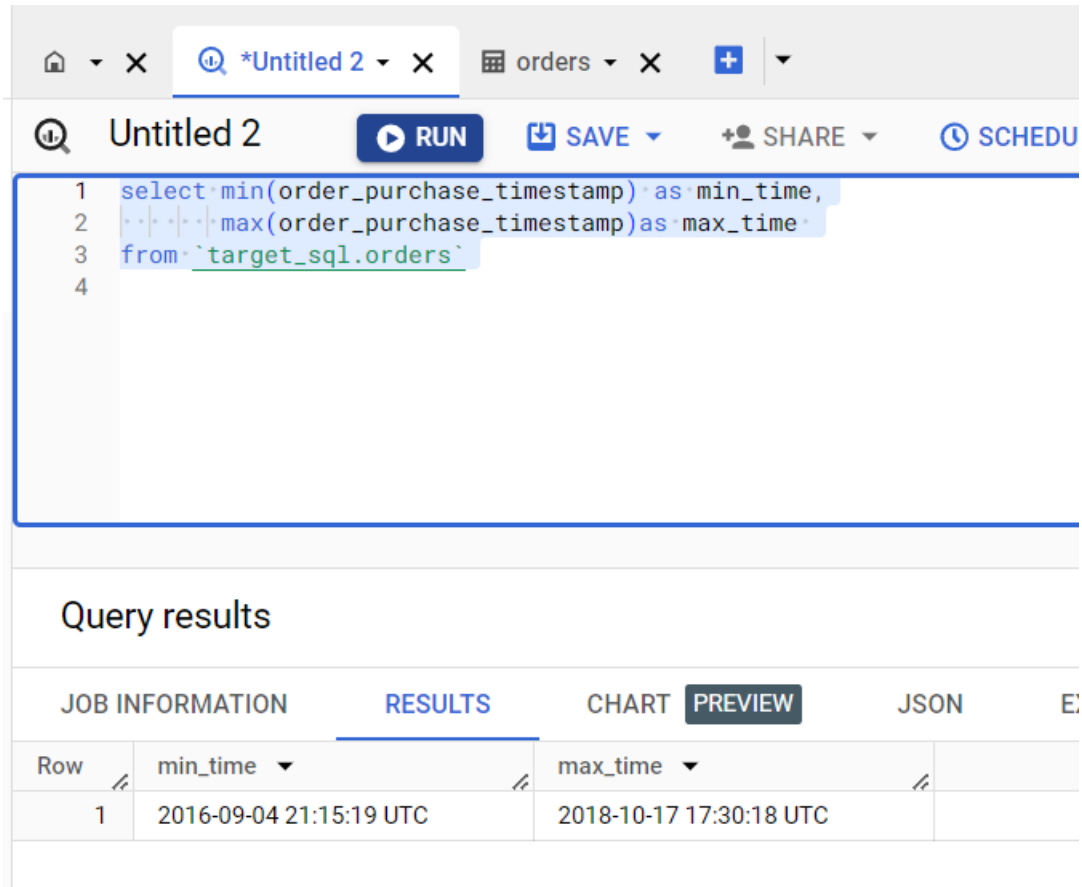


Sol : Here I imported all the tables under target_sql project where there is table name is customers.

In SCHEMA tab we can see the field name and there Data Type.

B. Get the time range between which the orders were placed.

Syntax -
```
select min(order_purchase_timestamp) as min_time,
       max(order_purchase_timestamp)as max_time
from `target_sql.orders`
```



Insights – Orders are placed between 4th Sep 2016 to 17th Oct 2018.

So basically we can say Data we received from client it range from Sep 2016 to Oct 2018 i.e. 2 year data.

C. Count the Cities & States of customers who ordered during the given period.

Syntax -
```
select count(distinct (customer_city)) as city,
```

```
      count(distinct(customer_state))as state
from `target_sql.customers`c
inner join `target_sql.orders` o  on c.customer_id = o.customer_id
```

```
select count(distinct (customer_city)) as city,
      count(distinct(customer_state))as state
from `target_sql.customers`c
inner join `target_sql.orders` o  on c.customer_id = o.customer_id
```

Query results

JOB INFORMATION | RESULTS | CHART PREVIEW | JSON | EXEC

Row	city	state
1	4119	27

Insights – From this we get to know count of City & States from where we get orders.

There are 27 states and 4119 city where we received order over period of time.

So basically we can see the regular customers from this States and city are more placing orders.

Suggestion - we can target more city and state for business.

2. **In-depth Exploration:**

   A. Is there a growing trend in the no. of orders placed over the past years?

Syntax -

```
select extract(Year from order_purchase_timestamp) as order_year,
      extract(Month from order_purchase_timestamp) as order_month,
      count(distinct(order_id)) as order_count
from `target_sql.orders`
group by order_year, order_month
order by order_year, order_month
```

```
select extract(Year from order_purchase_timestamp) as order_year,
       extract(Month from order_purchase_timestamp) as order_month,
       count(distinct(order_id)) as order_count
from `target_sql.orders`
group by order_year, order_month
order by order_year, order_month
```

Query results

JOB INFORMATION | RESULTS | CHART PREVIEW | JSON | EXI

Row	order_year	order_month	order_count
1	2016	9	4
2	2016	10	324
3	2016	12	1
4	2017	1	800

Insights – From this can see the there is growing trend for orders over the year for 2016 to 2018.

There high no. order placed in 2017 & 2018.

Suggestion – We can find out the months or year which orders are place in less quantity so we can find out reasons behind it and implement the strategy like offers and discount.

B. Can we see some kind of monthly seasonality in terms of the no. of orders being placed?

Syntax -
```
select extract(Year from order_purchase_timestamp) as order_year,
       extract(Month from order_purchase_timestamp) as order_month,
       count(distinct(order_id)) as order_count
from `target_sql.orders`
group by order_year, order_month
```

```
order by order_year,order_month
```

JOB INFORMATION | RESULTS | CHART PREVIEW | JS

Row	order_year	order_month	order_count
9	2017	6	3245
10	2017	7	4026
11	2017	8	4331
12	2017	9	4285
13	2017	10	4631
14	2017	11	7544
15	2017	12	5673
16	2018	1	7269
17	2018	2	6728

Insights – There is sudden spike on no. of orders for the November 2017 is 7544 & January 2018 is 7269 & March 2018 is 7211 because of festival season in Brazil.

Suggestions – During peak months arrange the warehouse nearby for stock arrangement so no delay in orders.

Manage the man power during this peak months.

For non-performing months we can throw some offers and discounts.

C. During what time of the day, do the Brazilian customers mostly place their orders? (Dawn, Morning, Afternoon or Night)
   a. 0-6 hrs : Dawn
   b. 7-12 hrs : Mornings
   c. 13-18 hrs : Afternoon
   d. 19-23 hrs : Night

Syntax - `select case`

```
        when extract(Hour from order_purchase_timestamp) between 0 and 6 then 'Dawn'
        when extract(Hour from order_purchase_timestamp) between 7 and 12 then 'Mornings'
        when extract(Hour from order_purchase_timestamp) between 13 and 18 then 'Afternoon'
        when extract(Hour from order_purchase_timestamp) between 19 and 23 then 'Night'
        ELSE 'unkown'
end as time_of_day,
count(distinct order_id) as order_count
from `target_sql.orders`
group by time_of_day
order by order_count desc
```

```
select case
      when extract(Hour from order_purchase_timestamp) between 0 and 6 then 'Dawn'
      when extract(Hour from order_purchase_timestamp) between 7 and 12 then 'Mornings'
      when extract(Hour from order_purchase_timestamp) between 13 and 18 then 'Afternoon
      when extract(Hour from order_purchase_timestamp) between 19 and 23 then 'Night'
      ELSE 'unkown'
end as time_of_day,
count(distinct order_id) as order_count
from `target_sql.orders`
group by time_of_day
```

Query results

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

Row	time_of_day	order_count
1	Afternoon	38135
2	Night	28331
3	Mornings	27733
4	Dawn	5242

Insight – From above analysis we get to know most of customers are placing orders in Afternoon that day time 1 to 6 then at Night time customer placing orders.

Suggestions – During peak hours we can arrange more man power and maintain the shifts breaks for them.

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

A. Get the month on month no. of orders placed in each state.

Syntax - select

```
     extract(Year from order_purchase_timestamp) as order_year,
     extract(Month from order_purchase_timestamp) as order_month,
     c.customer_state as state,
    count(Distinct(order_id)) as order_count
from `target_sql.orders` o
Join `target_sql.customers` c on o.customer_id=c.customer_id
group by 1,2,3
order by 1,2
```

Untitled 2 RUN SAVE SHARE SCHEDULE MORE

```
select
    extract(Year from order_purchase_timestamp) as order_year,
    extract(Month from order_purchase_timestamp) as order_month,
    c.customer_state as state,
   count(Distinct(order_id)) as order_count
from `target_sql.orders` o
Join `target_sql.customers` c on o.customer_id=c.customer_id
group by 1,2,3
order by 1,2
```

Query results

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

Row	order_year	order_month	state	order_count
1	2016	9	RR	1
2	2016	9	RS	1
3	2016	9	SP	2
4	2016	10	SP	113

Insights - From above analysis get to know SP state is having high volume of orders month on month.

Then RJ & MG states are also having high amount of orders.

It indicates most of revenue and active customers from these states.

Suggestion – identify the reason behind why other state are lacking behind in terms of orders.

B. How are the customers distributed across all the states?

Syntax –

```
select customer_state,

       count(distinct customer_unique_id) as cust_count,
       rank()over(order by count (distinct customer_unique_id) desc) as cust_rank
from `target_sql.customers`
group by customer_state
order by cust_rank
```

```sql
select customer_state,
       count(distinct customer_unique_id) as cust_count,
       rank()over(order by count (distinct customer_unique_id) desc) as cust_rank
from `target_sql.customers`
group by customer_state
order by cust_rank
```

Query results

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

Row	customer_state	cust_count	cust_rank
1	SP	40302	1
2	RJ	12384	2
3	MG	11259	3
4	RS	5277	4
5	PR	4882	5
6	SC	3534	6
7	BA	3277	7

Insights – From above analysis we get to know customers distribution over each state.

Top 5 states are SP, RJ, MG, RS, and PR.

Previously also seen that most of orders are coming from these states only.

Suggestion- We need to focus on states who are holding less customer count.

Throw some kind of attractive offers and fast delivery to attract customers.

4. **Impact on Economy: Analyze the money movement by e-commerce by looking at order prices, freight and others.**
   A. Get the % increase in the cost of orders from year 2017 to 2018 (include months between Jan to Aug only).
   You can use the "payment_value" column in the payments table to get the cost of orders.

Syntax -

```sql
select
    t1.order_year as year_2017,
```

```
      t2.order_year as year_2018,
      round(((t2.total_payment-t1.total_payment)/t1.total_payment),2)* 100 as
cost_increase_percentage
from
(select extract(Year from o1.order_purchase_timestamp) as order_year,
       extract (Month from o1.order_purchase_timestamp) as order_month,
       sum(p1.payment_value) as total_payment
from `target_sql.orders` o1
join `target_sql.payments` p1 on o1.order_id=p1.order_id
where extract(Year from o1.order_purchase_timestamp) = 2017
      and extract (Month from o1.order_purchase_timestamp) between 1 and 8
group by order_year,order_month) t1
join (
      select extract(Year from o2.order_purchase_timestamp) as order_year,
       extract (Month from o2.order_purchase_timestamp) as order_month,
       sum(p2.payment_value) as total_payment
from `target_sql.orders` o2
join `target_sql.payments` p2 on o2.order_id=p2.order_id
where extract(Year from o2.order_purchase_timestamp) = 2018
      and extract (Month from o2.order_purchase_timestamp) between 1 and 8
group by order_year,order_month
) t2 on t1.order_month=t2.order_month
order by cost_increase_percentage desc
```

Row	year_2017	year_2018	cost_increase_percentage
1	2017	2018	705.0
2	2017	2018	240.0
3	2017	2018	178.0
4	2017	2018	158.0
5	2017	2018	100.0
6	2017	2018	95.0
7	2017	2018	80.0
8	2017	2018	52.0

Insights – From above analysis we can see cost increase between 2017 & 2018.

It is from month of January to August.

B. Calculate the Total & Average value of order price for each state.

Syntax-

```
select c.customer_state,
       round(sum(p.payment_value),2) as total_price,
```

```
        round(avg(p.payment_value),2) as average_total
from
`target_sql.orders` o
join `target_sql.payments` p on o.order_id= p.order_id
join `target_sql.customers` c on o.customer_id=c.customer_id
group by c.customer_state
order by c.customer_state
```

Untitled 2 RUN SAVE SHARE SCHEDULI

```
1 select c.customer_state,
2         round(sum(p.payment_value),2) as total_price,
```

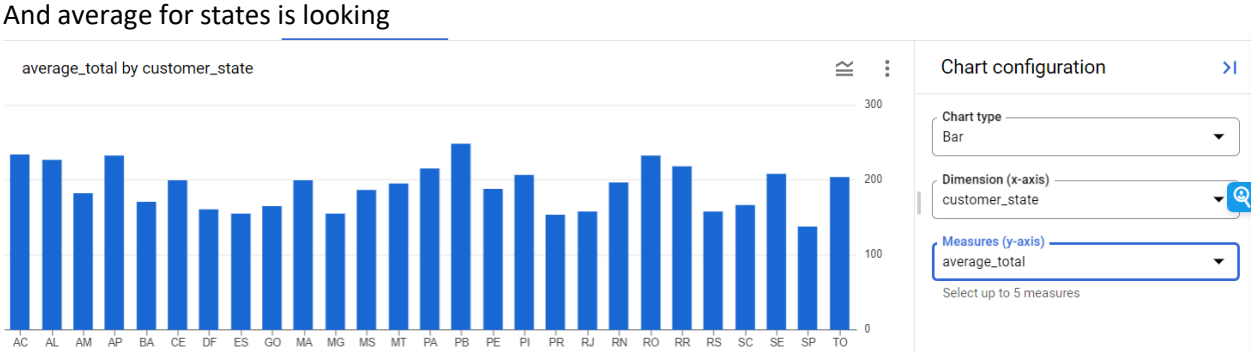
Query results

JOB INFORMATION | RESULTS | CHART PREVIEW | JSON | EXI

Row	customer_state	total_price	average_total
1	AC	19680.62	234.29
2	AL	96962.06	227.08
3	AM	27966.93	181.6
4	AP	16262.8	232.33
5	BA	616645.82	170.82
6	CE	279464.03	199.9
7	DF	355141.08	161.13
8	ES	325967.55	154.71
9	GO	350092.31	165.76

Insights – From above analysis we get to know there highest total price are coming from SP, RJ, & MG states.

And average for states is looking



C. Calculate the Total & Average value of order freight for each state.

Syntax –

```
SELECT c.customer_state,

       round(sum(oi.freight_value),2) as total_freight_value,
       round(avg(oi.freight_value),2) as average_freight_value
from
   `target_sql.orders` o
join `target_sql.order_items` oi on o.order_id=oi.order_id
join `target_sql.customers` c on o.customer_id= c.customer_id
group by c.customer_state
order by c.customer_state
```

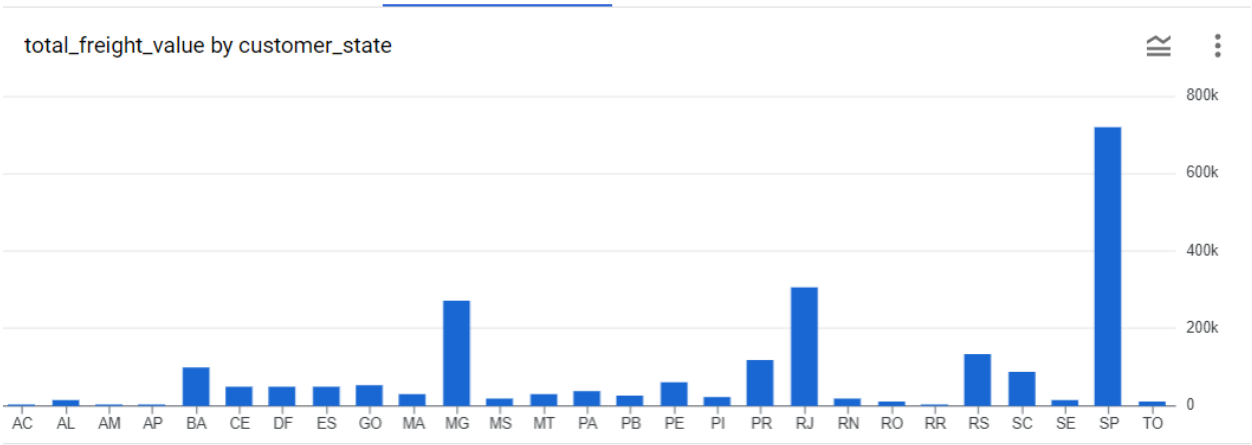
```
group by c.customer_state
order by c.customer_state
```

Query results

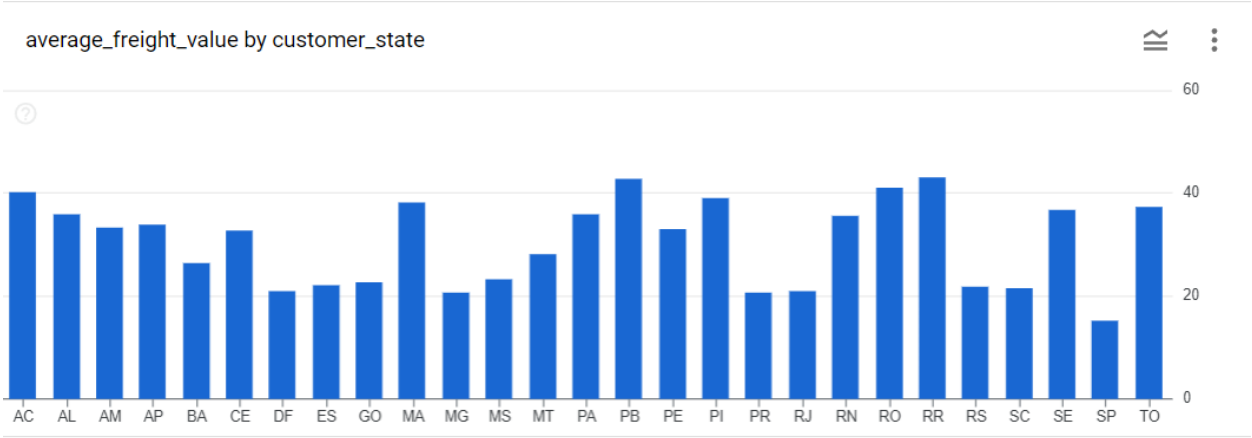
JOB INFORMATION | RESULTS | CHART PREVIEW | JSON

Row	customer_state	total_freight_value	average_freight_valu
1	AC	3686.75	40.07
2	AL	15914.59	35.84
3	AM	5478.89	33.21
4	AP	2788.5	34.01
5	BA	100156.68	26.36
6	CE	48351.59	32.71
7	DF	50625.5	21.04
8	ES	49764.6	22.06
9	GO	53114.98	22.77

Insights – from above analysis we get know states are high freight values are SP, RJ, MG



And average freight value for all states are



5. **Analysis based on sales, freight and delivery time.**

A. Find the no. of days taken to deliver each order from the order's purchase date as delivery time.
Also, calculate the difference (in days) between the estimated & actual delivery date of an order.
Do this in a single query.

You can calculate the delivery time and the difference between the estimated & actual delivery date using the given formula:

- **time_to_deliver** = order_delivered_customer_date - order_purchase_timestamp
- **diff_estimated_delivery** = order_estimated_delivery_date - order_delivered_customer_date

Syntax –

```
select
       date_diff(order_delivered_customer_date,order_purchase_timestamp,DAY) as
time_to_deliver,
       date_diff(order_estimated_delivery_date,order_purchase_timestamp,DAY) as
diff_estimated_delivery
from `target_sql.orders`
```

JOB INFORMATION | RESULTS | CHART | PREVIEW

Row	time_to_deliver	diff_estimated_delivery
1	null	50
2	null	6
3	null	44
4	null	54
5	null	56
6	null	54
7	null	56
8	null	41
9	null	3
10	null	2

Insights – From above analysis we can check the estimated day from the when we purchased order.

Also we can estimate the delivery dates as well.

Suggestions - As much early we deliver the orders, it will build good impact on customer experience.

B. Find out the top 5 states with the highest & lowest average freight value.

```
with avg_frieght_value as
(select  c.customer_state,
       round(avg(oi.freight_value),2) as average_freight_value,
       rank()over(order by avg(oi.freight_value) desc) as top_states
from `target_sql.orders` o
join `target_sql.order_items` oi on o.order_id=oi.order_id
join `target_sql.customers` c on o.customer_id=c.customer_id
group by c.customer_state)
select
     customer_state,
     average_freight_value
```

```
from
    avg_frieght_value
where
    top_states<=5
order by
       average_freight_value desc
```

JOB INFORMATION | RESULTS | CHART PREVIEW

Row	customer_state	average_freight_valu
1	RR	42.98
2	PB	42.72
3	RO	41.07
4	AC	40.07
5	PI	39.15

Insights – from above analysis we are getting to know top 5 states which having average high freight values comparing to others.

Suggestions – we can focus on how to reduce freight values.

Look for another vendors which are offering less transportations cost.

Syntax –

states who having lowest average freight value.

```
with avg_frieght_value as
(select  c.customer_state,
       round(avg(oi.freight_value),2) as average_freight_value,
       rank()over(order by avg(oi.freight_value) asc) as top_states
from `target_sql.orders` o
join `target_sql.order_items` oi on o.order_id=oi.order_id
join `target_sql.customers` c on o.customer_id=c.customer_id
group by c.customer_state)
select
     customer_state,
     average_freight_value
from
    avg_frieght_value
where
    top_states<=5
order by
       average_freight_value asc
```

JOB INFORMATION | RESULTS | CHART PREVIEW

Row	customer_state	average_freight_valu
1	SP	15.15
2	PR	20.53
3	MG	20.63
4	RJ	20.96
5	DF	21.04

Insights – from above analysis we can see the lowest freight value 5 states.

In this states we can push for more order numbers to generate more profit.

Suggestions – We can check for vendors for packing and transport to reduce freight values.

C. Find out the top 5 states with the highest & lowest average delivery time.

Syntax –

```
 select c.customer_state,

       round(avg(date_diff(order_delivered_customer_date,order_purchase_timestamp,DAY))) as
high_avg_delivery
from `target_sql.customers` c
join `target_sql.orders` o on c.customer_id=o.customer_id
group by c.customer_state
order by high_avg_delivery desc
limit 5
```

Row	customer_state	high_avg_delivery
1	RR	29.0
2	AP	27.0
3	AM	26.0
4	AL	24.0
5	PA	23.0

Insights - From above analysis we can the RR state I having highest delivery time as 29 day then AP having 27 days then AM having 26 days, AL having 24 days, PA having 23 days of delivery time.

Suggestions- we can built warehouses to those states so customers can easily get products with lowest delivery time. Need to focus on keep low delivery day time.

Syntax for Lowest delivery time –

```
select c.customer_state,

       round(avg(date_diff(order_delivered_customer_date,order_purchase_timestamp,DAY))) as
high_avg_delivery
from `target_sql.customers` c
join `target_sql.orders` o on c.customer_id=o.customer_id
group by c.customer_state
order by high_avg_delivery asc
limit 5
```

Row	customer_state	high_avg_delivery
1	SP	8.0
2	PR	12.0
3	MG	12.0
4	DF	13.0
5	SC	14.0

Insights – from above analysis we can see the SP state having low delivery time which is good from business perspective also PR & MG having 12 days average delivery time, DF & SC having the 13 & 14 days of delivery time respectively.

Suggestion – We can also reduce this much of average delivery time just for looking for new delivery vendors who can delivers as fast, increase man power for delivery.

D. Find out the top 5 states where the order delivery is really fast as compared to the estimated date of delivery.
   You can use the difference between the averages of actual & estimated delivery date to figure out how fast the delivery was for each state.

Syntax –

```
with avg_delivery_diff as

(
  select c.customer_state,
  avg(date_diff(order_delivered_customer_date,order_purchase_timestamp,DAY)) as
actual_avg_delivery,
  avg(date_diff(order_estimated_delivery_date,order_purchase_timestamp,DAY)) as
estimated_avg_delivery
```

```
  from `target_sql.customers` c
  join `target_sql.orders` o on c.customer_id=o.customer_id
  group by c.customer_state
)
select customer_state,
       round(estimated_avg_delivery - actual_avg_delivery) as delivery_diffrence
from
    avg_delivery_diff
order by
      delivery_diffrence
limit 5
```

Row	customer_state	delivery_diffrence
1	AL	8.0
2	SE	9.0
3	MA	9.0
4	ES	10.0
5	MS	10.0

Insights – from above analysis we can understand the delivery difference between the actual and estimated delivery date so, AL State Is having 8 days of delivery date comparing to estimated then SE & MA having 9 days of delivery difference then ES & MS having 10 days of delivery difference.

Suggestions – It is good to deliver product before estimated delivery date. We can work on the fast delivery mode transportations so it will help to get products as fast.

6. Analysis based on the payments:

A. Find the month on month no. of orders placed using different payment types.

Syntax –

```
select

     extract(year from o.order_purchase_timestamp) as order_year,
     extract(month from o.order_purchase_timestamp) as order_month,
     p.payment_type,
     count(*) as num_orders
from
    `target_sql.orders` o
    join `target_sql.payments` p on o.order_id=p.order_id
group by
       order_year, order_month, p.payment_type
order by
        order_year, order_month, p.payment_type
```

Row	order_year	order_month	payment_type	num_orders
1	2016	9	credit_card	3
2	2016	10	UPI	63
3	2016	10	credit_card	254
4	2016	10	debit_card	2
5	2016	10	voucher	23
6	2016	12	credit_card	1
7	2017	1	UPI	197

Insights – from above analysis we can see the most prefer payment type by customers over year and month. Credit card is the most used payment mode by customers then UPI & Voucher.

Suggestions - As credit card is most prefer mode of payment we can increase number of payment machines at shops so customers won't get delay in payment and don't need to wait for payment.

We can put some offers on different payment mode so they also utilize.

B. Find the no. of orders placed on the basis of the payment installments that have been paid.

Syntax -

```
 select

      payment_installments,
      count(*) as num_orders
from
    `target_sql.payments`
group by payment_installments
order by payment_installments
```

Row	payment_installment	num_orders
1	0	2
2	1	52546
3	2	12413
4	3	10461
5	4	7098
6	5	5239
7	6	3920
8	7	1626
9	8	4268

Insights – From above analysis we can see the payment installment going by order numbers.

So here are 1 payment installment having largest orders 52546 then 2 having 12413 and 22 & 23 are having only 1 orders respectively.

Suggestions – we can put attractive offers on payment installment on time so customers don't delay on installment payment.