Target-sql

1**. import the dataset and do the usual exploratory analysis steps like checking the structure and charcteristics of the dataset**

a.**data type of columns in a table**

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

c.Count the number of Cities and States in our dataset.

Sol:

1.a)

```
select
 column_name,data_type
from `Target`.information_schema.columns
where table_name='customers'
```

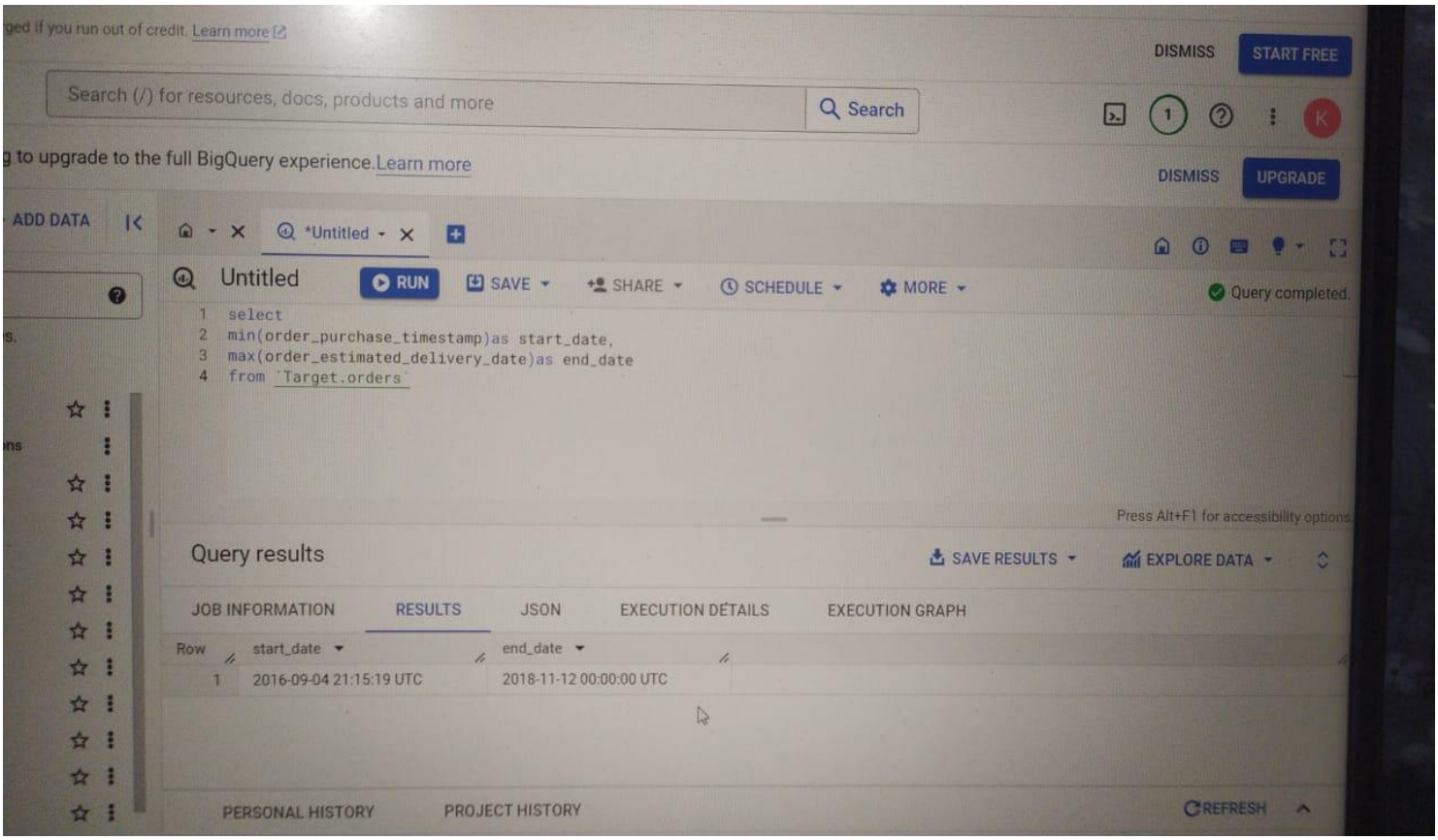
Field name	Type	Mode	K
customer_id	STRING	NULLABLE	
customer_unique_id	STRING	NULLABLE	
customer_zip_code_prefix	INTEGER	NULLABLE	
customer_city	STRING	NULLABLE	
customer_state	STRING	NULLABLE	

From above ; we get data of all columns by selecting all columns from customers table

1.b)

```
select
   min(order_purchase_timestamp)as start_date,
   max(order_estimated_delivery_date)as end_date
   from `Target.orders`
```

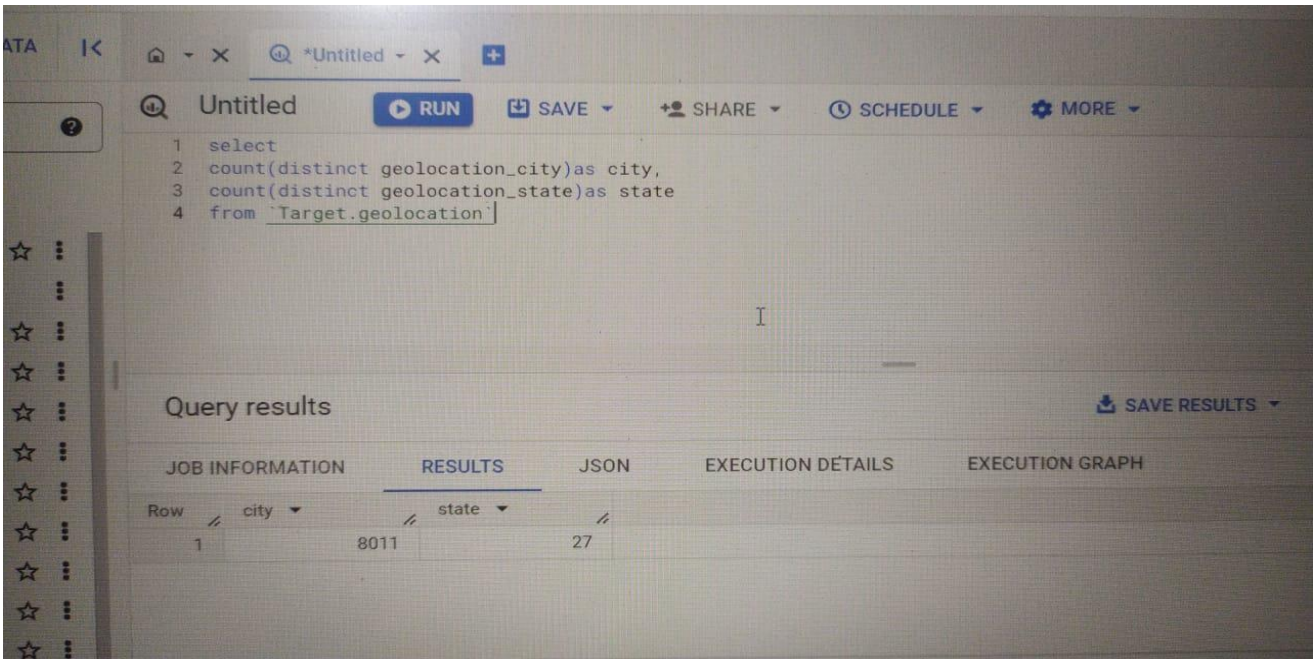
based on above data , we select minimum and maximum of order-purchase_timestamp and order_estimated_delivery columns to get time range between which the orders were placed.



1.c)

```
 select
count(distinct geolocation_city)as city,
count(distinct geolocation_state)as state
from `Target.geolocation`
```

from above ; we select geolocation_city,geolocation_state columns from geolocation table by using distinct to get unique details and by using count to get the no of cities and states



2. In-depth Exploration:

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

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

C .During what time of the day, do the Brazilian customers mostly place their orders? (Dawn, Morning, Afternoon or Night)

- 0-6 hrs : Dawn
- 7-12 hrs : Mornings
- 13-18 hrs : Afternoon
- 19-23 hrs : Night

Sol:

2.a) select

```
extract(YEAR from order_purchase_timestamp) as year,
extract(MONTH from order_purchase_timestamp) as month,
round(sum(p.payment_value), 2) as revenue
from `Target.orders` as o
inner join `Target.payments` as p
on o.order_id = p.order_id
group by year, month
order by year, month
```

from above; we use extract to get year and month details ,using inner join to get exact matching values of two tables to achieve growing trend in the no od orders placed over the past years

```
extract(YEAR from order_purchase_timestamp) as year,
```

Query results

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

Row	year	month	revenue
1	2016	9	252.24
2	2016	10	59090.48
3	2016	12	19.62
4	2017	1	138488.04
5	2017	2	291908.01
6	2017	3	449863.6
7	2017	4	417788.03
8	2017	5	592918.82
9	2017	6	511276.38

2.b)

```
select
extract(MONTH from order_purchase_timestamp) as month,
count(distinct order_id) as orders_count
from `Target.orders`
group by month
order by month
```

from above;we use extract and count to get the details of month and counting no of orders were placed and we see that no of monthly seasonality orders increasing and fluctualuations also from jan to august and then gradually orders were decreasing.

Query results

JOB INFORMATION | RESULTS | JSON | EXECUTION DETAILS

Row	month	orders_count
1	1	8069
2	2	8508
3	3	9893
4	4	9343
5	5	10573
6	6	9412
7	7	10318
8	8	10843
9	9	4305

JOB INFORMATION | RESULTS | JSON | EXECUTION DETAILS

Row	month	orders_count
4	4	9343
5	5	10573
6	6	9412
7	7	10318
8	8	10843
9	9	4305
10	10	4959
11	11	7544
12	12	5674

2.c)

```
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'
end as hour,
count(o.order_id) as order_count
from `Target.orders` as o
join `Target.customers` as c
on o.customer_id = c.customer_id
group by hour
order by order_count desc
```

based on above query; we observe that most of Brazilian customers place their orders in afternoon

Query results

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

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

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

a.Get the month on month no. of orders placed in each state.
b.How are the customers distributed across all the states?

Sol:

3.a)

```
   select customer_state, count(ord.order_id) as order_size,

    Concat(SUBSTR(CAST((order_purchase_timestamp) AS STRING),01,4),'-',
SUBSTR(CAST((order_purchase_timestamp) AS STRING),06,2)) as sales_span
    from `Target.customers` cust join `Target.orders` ord
     on cust.customer_id = ord.customer_id
     where order_status is not null
   group by sales_span, customer_state
   order by sales_span
```

based on above query; we get month on month order by states

Query results

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

Row	customer_state	order_size	sales_span
1	RR	1	2016-09
2	RS	1	2016-09
3	SP	2	2016-09
4	SP	113	2016-10
5	RS	24	2016-10
6	RJ	56	2016-10
7	MT	3	2016-10
8	GO	9	2016-10
9	MG	40	2016-10

3.b)

```
select
customer_state,
count(*)as total_no_of_purchases
from `Target.customers`
group by customer_state
order by total_no_of_purchases desc
```

From above querey ,we select customer_state column from customers table,in that we use count to get total no of purchases where customers distributed across all the states

Query results

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

Row	customer_state	total_no_of_purchase
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

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.
b. Calculate the Total & Average value of order price for each state.
c. Calculate the Total & Average value of order freight for each state.

Sol:
4.a)

```
select distinct
Round(Avg(price + freight_value),2) as Avg_Cost, SUBSTR(Cast(order_purchase_timestamp as
string),01,07) as payment_period
from `Target.order_items` oi join `Target.orders` o
on oi.order_id = o.order_id
where (extract(Date from order_purchase_timestamp) > "2018-01-01" and extract(Date from
order_purchase_timestamp) < '2018-08-31') or (extract(Date from order_purchase_timestamp) >
"2017-01-01" and extract(Date from order_purchase_timestamp) < '2017-08-31')
group by payment_period
order by payment_period
from above querey ; we get deatails of increase in the cost of orders from year 2017 to
2018
```

Query results

JOB INFORMATION | RESULTS | JSON | EXECUTION DETAILS | EXECUTION GRA

Row	Avg_Cost	payment_period
1	143.65	2017-01
2	146.74	2017-02
3	144.02	2017-03
4	153.66	2017-04
5	141.73	2017-05
6	140.37	2017-06
7	129.45	2017-07
8	136.03	2017-08
9	135.22	2018-01

4.c)

```
select  tbl1.customer_state,

Round(avg(tbl1.Price_with_Freight),2) as avg_price_with_Freight,

Round(Sum(tbl1.Price_with_Freight),2) as total_price_with_Freight
from
(select
 oi.order_id,
 (oi.price +  oi.freight_value) as Price_with_Freight,
  o.customer_id,
 c.customer_state
from `Target.order_items` oi join `Target.orders` o
on  oi.order_id = o.order_id join   `Target.customers` c
on o.customer_id = c.customer_id) tbl1
group by tbl1.customer_state
order by total_price_with_Freight desc;s
```

***based on above querey;**
we get total and avg value of freight price for each state

Query results

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

Row	customer_state	avg_price_with_Freig	total_price_with_Frei
1	SP	124.8	5921678.12
2	RJ	146.08	2129681.98
3	MG	141.38	1856161.49
4	RS	142.07	885826.76
5	PR	139.54	800935.44
6	BA	160.97	611506.67
7	SC	146.12	610213.6
8	DF	146.81	353229.44
9	GO	149.04	347706.93

4.b)

```
select  tbl1.customer_state,
Round(avg(tbl1.Price),2) as avg_price,
Round(Sum(tbl1.Price),2) as total_price
from
(select oi.order_id, oi.price as Price, o.customer_id,
c.customer_state
from `Target.order_items` oi join `Target.orders` o
on  oi.order_id = o.order_id join   `Target.customers` c
on o.customer_id = c.customer_id) tbl1
group by tbl1.customer_state
order by total_price desc;
```

***based on above querey ;we get total and avg value of order price for each state**

Query results

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

Row	customer_state	avg_price	total_price
1	SP	109.65	5202955.05
2	RJ	125.12	1824092.67
3	MG	120.75	1585308.03
4	RS	120.34	750304.02
5	PR	119.0	683083.76
6	SC	124.65	520553.34
7	BA	134.6	511349.99
8	DF	125.77	302603.94

Load more

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
b. Find out the top 5 states with the highest & lowest average freight value.
c. Find out the top 5 states with the highest & lowest average delivery time.
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.

Sol:

5.a)

```
select
order_purchase_timestamp,
timestamp_diff(order_delivered_customer_date, order_purchase_timestamp, Day) as
days_between_delivering,
timestamp_diff(order_estimated_delivery_date, order_purchase_timestamp, Day) as
days_estimated_to_deliver
from `Target.orders`
order by order_purchase_timestamp
```

Null values for orders which are yet to deliver, cancelled or for which delivery date was not recorded

Query results

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

Row	order_purchase_timestamp	days_between_delive	days_estimated_to_d
1	2016-09-04 21:15:19 UTC	null	45
2	2016-09-05 00:15:34 UTC	null	52
3	2016-09-13 15:24:19 UTC	null	16
4	2016-09-15 12:16:38 UTC	54	18
5	2016-10-02 22:07:52 UTC	null	22
6	2016-10-03 09:44:50 UTC	23	23
7	2016-10-03 16:56:50 UTC	24	34
8	2016-10-03 21:01:41 UTC	35	52
9	2016-10-03 21:13:36 UTC	30	56

time_to_delivery & diff_estimated_delivery :

```
select  order_purchase_timestamp, order_estimated_delivery_date,
order_delivered_customer_date,
timestamp_diff(order_delivered_customer_date, order_purchase_timestamp, Day) as
time_taken_to_deliver,
timestamp_diff(order_estimated_delivery_date, order_delivered_customer_date, Day) as
estimate_vs_delivery_time
from `Target.orders`
order by order_purchase_timestamp desc;
```

Query results

SAVE RESULTS

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

Row	order_purchase_timestamp	order_estimated_delivery_date	order_delivered_customer_date	time_taken_to_delive	estimate_vs_delivery
1	2018-10-17 17:30:18 UTC	2018-10-30 00:00:00 UTC	null	null	null
2	2018-10-16 20:16:02 UTC	2018-11-12 00:00:00 UTC	null	null	null
3	2018-10-03 18:55:29 UTC	2018-10-16 00:00:00 UTC	null	null	null
4	2018-10-01 15:30:09 UTC	2018-10-23 00:00:00 UTC	null	null	null
5	2018-09-29 09:13:03 UTC	2018-10-15 00:00:00 UTC	null	null	null
6	2018-09-26 08:40:15 UTC	2018-10-25 00:00:00 UTC	null	null	null
7	2018-09-25 11:59:18 UTC	2018-10-11 00:00:00 UTC	null	null	null
8	2018-09-20 13:54:16 UTC	2018-10-17 00:00:00 UTC	null	null	null
9	2018-09-17 17:21:16 UTC	2018-10-01 00:00:00 UTC	null	null	null

Results per page: 50

5.b)

States with highest freight value

```
select
  Round(avg(freight_value),2) as avg_freight_value,
  Round(avg(TIMESTAMP_DIFF(order_delivered_customer_date, order_purchase_timestamp,
Day)),2) as  time_to_delivery,
  Round(avg(TIMESTAMP_DIFF(order_estimated_delivery_date, order_purchase_timestamp,
Day)),2) as diff_estimated_delivery,
  c.customer_state
  from  `Target.order_items` oi join  `Target.orders` o
  on  oi.order_id = o.order_id join `Target.customers` c
  on  o.customer_id = c.customer_id group by c.customer_state
  order by avg_freight_value desc limit 5;
```

Query results

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

Row	avg_freight_value	time_to_delivery	diff_estimated_deliv	customer_state
1	42.98	27.83	45.98	RR
2	42.72	20.12	32.55	PB
3	41.07	19.28	38.65	RO
4	40.07	20.33	40.7	AC
5	39.15	18.93	29.92	PI

States with lowest freight value

```
select
Round(avg(freight_value),2) as avg_freight_value,
Round(avg(TIMESTAMP_DIFF(order_delivered_customer_date, order_purchase_timestamp, Day)),2)
as time_to_delivery,
Round(avg(TIMESTAMP_DIFF(order_estimated_delivery_date, order_purchase_timestamp, Day)),2)
as diff_estimated_delivery,
c.customer_state
from  `Target.order_items` oi join  `Target.orders` o
on  oi.order_id = o.order_id join `Target.customers` c
on  o.customer_id = c.customer_id group by c.customer_state
order by avg_freight_value limit 5;
```

Query results

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

Row	avg_freight_value	time_to_delivery	diff_estimated_deliv	customer_state
1	15.15	8.26	18.9	SP
2	20.53	11.48	24.38	PR
3	20.63	11.52	24.31	MG
4	20.96	14.69	26.1	RJ
5	21.04	12.5	24.19	DF

5.c)

States with higest average time to delivery

```
    select
Round(avg(freight_value),2) as avg_freight_value,
Round(avg(TIMESTAMP_DIFF(order_delivered_customer_date, order_purchase_timestamp, Day)),2)
as time_to_delivery,
Round(avg(TIMESTAMP_DIFF(order_estimated_delivery_date, order_purchase_timestamp, Day)),2)
as diff_estimated_delivery,
c.customer_state
from  `Target.order_items` oi join  `Target.orders` o
on  oi.order_id = o.order_id join `Target.customers` c
on  o.customer_id = c.customer_id group by c.customer_state
order by time_to_delivery desc limit 5;
```

Query results

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

ow	avg_freight_value	time_to_delivery	diff_estimated_deliv	customer_state
1	42.98	27.83	45.98	RR
2	34.01	27.75	45.49	AP
3	33.21	25.96	45.21	AM
4	35.84	23.99	32.18	AL
5	35.83	23.3	36.96	PA

States with lowest average time to delivery

```
 select
Round(avg(freight_value),2) as avg_freight_value,
Round(avg(TIMESTAMP_DIFF(order_delivered_customer_date, order_purchase_timestamp, Day)),2) as
time_to_delivery,
Round(avg(TIMESTAMP_DIFF(order_estimated_delivery_date, order_purchase_timestamp, Day)),2) as
diff_estimated_delivery,
c.customer_state
from  `Target.order_items` oi join  `Target.orders` o
on  oi.order_id = o.order_id join `Target.customers` c
on  o.customer_id = c.customer_id group by c.customer_state
order by time_to_delivery limit 5;
```

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

Row	avg_freight_value	time_to_delivery	diff_estimated_deliv	customer_state
1	15.15	8.26	18.9	SP
2	20.53	11.48	24.38	PR
3	20.63	11.52	24.31	MG
4	21.04	12.5	24.19	DF
5	21.47	14.52	25.51	SC

5.d)

fast deliver compare to estimated date

```
select
Round(avg(freight_value),2) as avg_freight_value,
Round(avg(TIMESTAMP_DIFF(order_delivered_customer_date, order_purchase_timestamp, Day)),2)
as time_to_delivery,
Round(avg(TIMESTAMP_DIFF(order_estimated_delivery_date, order_purchase_timestamp, Day)),2)
as estimated_time_to_delivery,
Round(avg(TIMESTAMP_DIFF(order_estimated_delivery_date, order_delivered_customer_date,
Day)),2) as estimate_vs_delivery_gap,
Round(Avg(case
when TIMESTAMP_DIFF(order_estimated_delivery_date, order_delivered_customer_date, Day) = 0
then 0
else((TIMESTAMP_DIFF(order_estimated_delivery_date, order_delivered_customer_date, Day) /
TIMESTAMP_DIFF(order_estimated_delivery_date, order_purchase_timestamp, Day)) * 100)End),2)
as percent_gap_estimate_vs_delivery,
c.customer_state
from  `Target.order_items` oi join  `Target.orders` o
on  oi.order_id = o.order_id
join  `Target.customers` c
on  o.customer_id = c.customer_id
where order_estimated_delivery_date is not null
group by c.customer_state
order by percent_gap_estimate_vs_delivery desc
limit 5;
```

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

Row	avg_freight_value	time_to_delivery	estimated_time_to_d	estimate_vs_delivery	percent_gap_estimat	customer_state
1	15.15	8.26	18.9	10.27	51.61	SP
2	20.53	11.48	24.38	12.53	49.97	PR
3	20.63	11.52	24.31	12.4	49.56	MG
4	40.07	20.33	40.7	20.01	48.31	AC
5	41.07	19.28	38.65	19.08	48.07	RO

6. Analysis based on the payments:

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

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

Sol:

6.a)

```
 select  tbl1.payment_type,

Concat(Case
When tbl1.month = 01 then 'Jan'
When tbl1.month = 02 then 'Feb'
When tbl1.month = 03 then 'Mar'
When tbl1.month = 04 then 'Apr'
When tbl1.month = 05 then 'May'
When tbl1.month = 06 then 'June'
When tbl1.month = 07 then 'July'
When tbl1.month = 08 then 'Aug'
When tbl1.month = 09 then 'Sep'
When tbl1.month = 10 then 'Oct'
When tbl1.month = 11 then 'Nov'
When tbl1.month = 12 then 'Dec' End,', ',
tbl1.year) as Month_of_order, tbl1.No_of_orders as order_count,
Sum(tbl1.No_of_orders) over(partition by tbl1.payment_type order by
tbl1.year, tbl1.month) as month_over_month_count_of_orders
from (select
count(ord.order_id) as No_of_orders, pmt.payment_type,
extract(year from ord.order_purchase_timestamp) as year,
extract(month from ord.order_purchase_timestamp) as month
from `Target.orders` ord join `Target.payments` pmt
on ord.order_id = pmt.order_id
group by year, month, pmt.payment_type order by year, month,
pmt.payment_type
) tbl1
order by tbl1.year, tbl1.month, payment_type;
```

Query results

SAVE RESULTS EXPLO

JOB INFORMATION RESULTS JSON EXECUTION DETAILS EXECUTION GRAPH

	payment_type	Month_of_order	order_count	month_over_month_c
1	credit_card	Sep, 2016	3	3
2	UPI	Oct, 2016	63	63
3	credit_card	Oct, 2016	254	257
4	debit_card	Oct, 2016	2	2
5	voucher	Oct, 2016	23	23
6	credit_card	Dec, 2016	1	258
7	UPI	Jan, 2017	197	260
8	credit_card	Jan, 2017	583	841

d more

Results per page: 50 1 – 50 of 90

6.b)

```
select
count(pmt.order_id) as order_count,
pmt.payment_installments
from `Target.payments` ord join `Target.payments` pmt
on ord.order_id = pmt.order_id
group by payment_installments
order by order_count;
```

Query results

JOB INFORMATION RESULTS JSON EXECUTION DETAILS EXECUTION GRAPH

Row	order_count	payment_installment
1	1	22
2	1	23
3	2	0
4	3	21
5	5	16
6	8	17
7	16	13
8	17	20
9	18	14

Insights :

From first question

a.data types of columns in a customers table is string and integer

b. time range between orders were placed on 2016-09-04-21:15:19 UTC to 2018-11-12 – 00:00:00 UTC

a.no of cities and states are 8011 and 27 respectively where count of cities were high and count of states were low

From second question

a.there is growing in a trend of increasing over past years by comparing with revenue

b. seasonality orders were increasing and fluctuations from jan to august ,and also gradually decreases aug to dec

c.most of the Brazilian customers place their orders in afternoon time

From third question

a.no of e-commerce orders were placed in each state from month to month it changes like in dec orders very high and jan very low

b.total no of purchases ordered by customerstate in sp is high about '41746'

From fourth question

a. the cost of orders increases from 2017 to 2018 and the highest avgcost is 153.66 and payment period is about 2017-04

b. highest total value of order price is 5202955.05 and avg value of order price is 134.6

c. higest total freight value of order price is 5921678.12 and avg freight value of order price is 160.97

From fifth question

a. no of orders time taken to delivery is high on date 2016 -10-03,21:13:36 UTC

b. states with highest freight value in customer state in "RR" and states with lowest freight value in customer state in "SP"

c. states with highest avg time to delivery in customerstate is “RR” along with time to delivery is 27.83 and diff estimated delivery is 45.98 and states lowest avg time to delivery in customer state is “SP” along with time to delivery is 8.26 and diff estimated delivery is 18.9

d. the order delivery is really fast as compared to estimated delivery is in top 5 staes are SP,PR,MG,AC,RO Respectively

<u>From sixth question</u>

a. No of orders placed using different payment types is “credit card’ having high- order count is 587 and month over month is 841

b. No of orders placed on the basis of high payment installaments is 23