Name - Elakala Sai Kumar
Phone - +917989089283

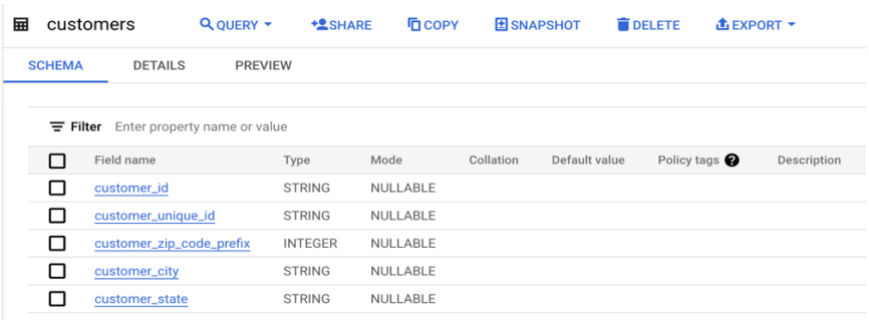
Business Case: Target SQL

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

I want to use GCP environment (Big Query) to work on this dataset. From the schema I can determine that order table is the main table which is used to establish relation with other tables with the help of primary key etc.,

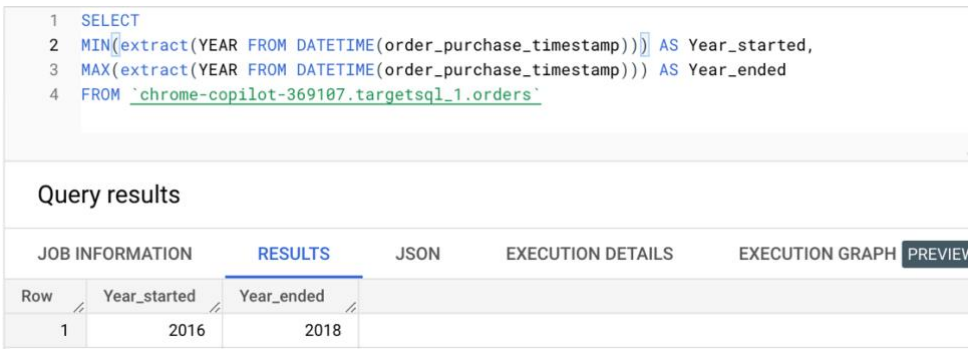
1.1.Data type of columns in a table

To determine what datatype a column belongs to we have built in analysis in GCP which determines datatype of **each and every column in a table**.Similar to the below screenshot.



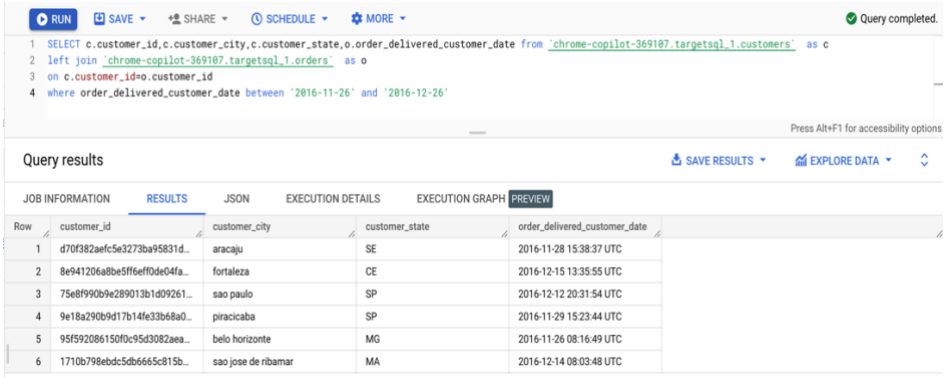
1.2. Time period for which the data is given

From the question I can understand that dataset I have has the records from the year **2016 to 2018.**



1.3. Cities and States of customers ordered during the given period

For knowing the cities and state of a customer with in a date time range I have used left join between customers and order table.



```
SELECT c.customer_id,c.customer_city,c.customer_state,o.order_delivered_customer_date from `chrome-copilot-369107.targetsql_1.customers` as c
left join `chrome-copilot-369107.targetsql_1.orders` as o
on c.customer_id=o.customer_id
where order_delivered_customer_date between '2016-11-26' and '2016-12-26'
```

Query results

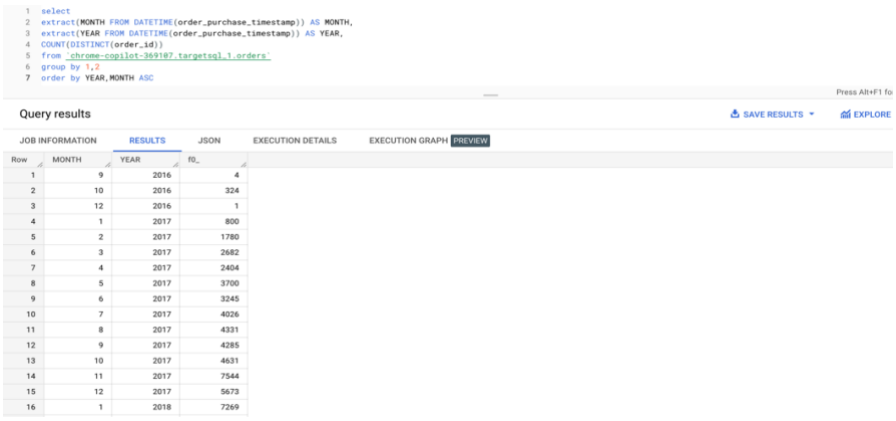
Row	customer_id	customer_city	customer_state	order_delivered_customer_date
1	d70f382aefc5e3273ba95831d...	aracaju	SE	2016-11-28 15:38:37 UTC
2	8e941206a8be5ff6eff0de04fa...	fortaleza	CE	2016-12-15 13:35:55 UTC
3	75e8f990b9e289013b1d09261...	sao paulo	SP	2016-12-12 20:31:54 UTC
4	9e18a290b9d17b14fe33b68a0...	piracicaba	SP	2016-11-29 15:23:44 UTC
5	95f592086150f0c95d3082aea...	belo horizonte	MG	2016-11-26 08:16:49 UTC
6	1710b798ebdc5db6665c815b...	sao jose de ribamar	MA	2016-12-14 08:03:48 UTC

2. In-depth Exploration:

One possible way of analysis we can do in this section is with Date-Time analysis

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

I had considered month from order_purchase_timestamp column to analyse the seasonal trend in the dataset with the number of orders customers purchased. In my analysis I can say trend in inclined in the **last quarter of the year for 2017 i.e., (9,10,11) months and for 2018 its (5,6,7)months**.



```
select
extract(MONTH FROM DATETIME(order_purchase_timestamp)) AS MONTH,
extract(YEAR FROM DATETIME(order_purchase_timestamp)) AS YEAR,
COUNT(DISTINCT(order_id))
from `chrome-copilot-369107.targetsql_1.orders`
group by 1,2
order by YEAR,MONTH ASC
```

Query results

Row	MONTH	YEAR	f0_
1	9	2016	4
2	10	2016	324
3	12	2016	1
4	1	2017	800
5	2	2017	1780
6	3	2017	2682
7	4	2017	2404
8	5	2017	3700
9	6	2017	3245
10	7	2017	4026
11	8	2017	4331
12	9	2017	4285
13	10	2017	4631
14	11	2017	7544
15	12	2017	5673
16	1	2018	7269

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

I had considered hour from order_purchase_timestamp column to analyse the times of the day in which number of orders made by customer. In my analysis I can say customers are more interested to place orders from Morning 10:00 A.m. to Night 10:00 P.M.

```
select
extract(HOUR FROM DATETIME(order_purchase_timestamp)) AS hour,
COUNT(*) as usage
from `chrome-copilot-369107.targetsql_1.orders` as o
group by hour
order by hour asc
```

Query results

SAVE RESULTS

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

Row	hour	usage
5	4	206
6	5	188
7	6	502
8	7	1231
9	8	2967
10	9	4785
11	10	6177
12	11	6578
13	12	5995
14	13	6518
15	14	6569
16	15	6454
17	16	6675
18	17	6150
19	18	5769
20	19	5982
21	20	6193
22	21	6217
23	22	5816
24	23	4123

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

3.1. **Get month on month orders by states**

To know the month on month order by states I have applied Date time and order by function here and also applied mom calculation to know the difference.

```
select t.* from (select *, round((((Total/mom)-1)*100),1) as mom_percentage from(select *
,lag(Total) over(partition by State order by year,month asc ) as mom from (select
c.customer_state as State,extract(year from o.order_purchase_timestamp) as Year,
extract(month from o.order_purchase_timestamp) as Month, count(o.order_id) as Total from
`chrome-copilot-369107.targetsql_1.orders` as o inner join `chrome-copilot-
369107.targetsql_1.customers` as c on o.customer_id=c.customer_id
where o.order_purchase_timestamp is not null
group by State,Year,Month)base1)base2
order by Year,Month,State asc)t
where t.mom is not null
```

```
select t.* from (select *, round((((Total/mom)-1)*100),1) as mom_percentage from(select * ,lag(Total) over(partition by State order by year,month asc ) as mom from (select c.
customer_state as State,extract(year from o.order_purchase_timestamp) as Year, extract(month from o.order_purchase_timestamp) as Month, count(o.order_id) as Total from
`chrome-copilot-369107.targetsql_1.orders` as o inner join `chrome-copilot-369107.targetsql_1.customers` as c on o.customer_id=c.customer_id
where o.order_purchase_timestamp is not null
group by State,Year,Month)base1)base2
order by Year,Month,State asc)t
where t.mom is not null
```

Press Alt+F1 for acces

Query results

SAVE RESULTS | EXPLORE DATA

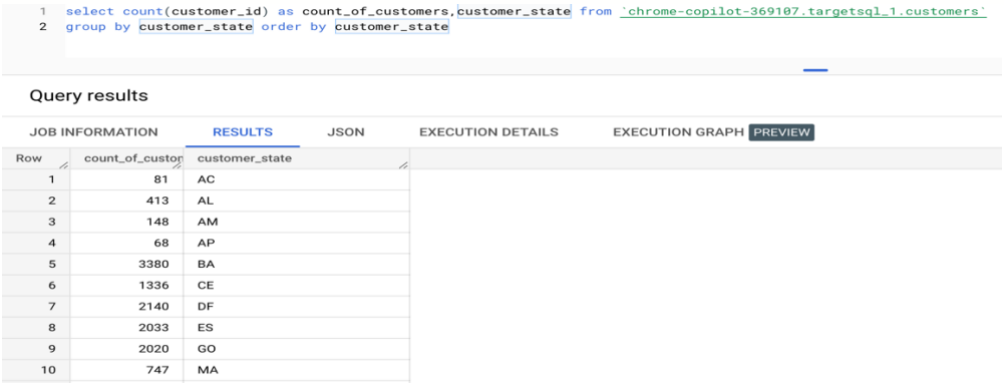
JOB INFORMATION | RESULTS | JSON | EXECUTION DETAILS | EXECUTION GRAPH PREVIEW

Row	State	Year	Month	Total	mom	mom_percentag
1	RR	2016	10	1	1	0.0
2	RS	2016	10	24	1	2300.0
3	SP	2016	10	113	2	5550.0
4	PR	2016	12	1	19	-94.7
5	AL	2017	1	2	2	0.0
6	BA	2017	1	25	4	525.0
7	CE	2017	1	9	8	12.5
8	DF	2017	1	13	6	116.7
9	ES	2017	1	12	4	200.0
10	GO	2017	1	18	9	100.0

3.2. **Distribution of customers across the states in Brazil**

I have analysed distribution of customers across the states with Distinct function.

```
select count(customer_id) as count_of_customers,customer_state from `chrome-copilot-369107.targetsql_1.customers`
group by customer_state order by customer_state
```



Query results

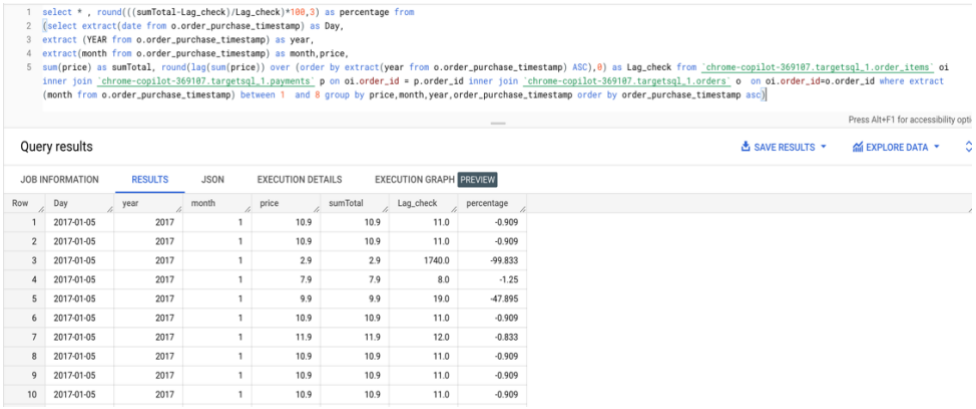
JOB INFORMATION | RESULTS | JSON | EXECUTION DETAILS | EXECUTION GRAPH PREVIEW

Row	count_of_custor	customer_state
1	81	AC
2	413	AL
3	148	AM
4	68	AP
5	3380	BA
6	1336	CE
7	2140	DF
8	2033	ES
9	2020	GO
10	747	MA

4. **Impact on Economy: Analyze the money movement 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) - You can use "payment_value" column in payments table**

select * , round(((sumTotal-Lag_check)/Lag_check)*100,3) as percentage from (select extract(date from o.order_purchase_timestamp) as Day,extract (YEAR from o.order_purchase_timestamp) as year,extract(month from o.order_purchase_timestamp) as month,price,sum(price) as sumTotal, round(lag(sum(price)) over (order by extract(year from o.order_purchase_timestamp) ASC),0) as Lag_check from `chrome-copilot-369107.targetsql_1.order_items` oi inner join `chrome-copilot-369107.targetsql_1.payments` p on oi.order_id = p.order_id inner join `chrome-copilot-369107.targetsql_1.orders` o on oi.order_id=o.order_id where extract(month from o.order_purchase_timestamp) between 1 and 8 group by price,month,year,order_purchase_timestamp order by order_purchase_timestamp asc)



Query results

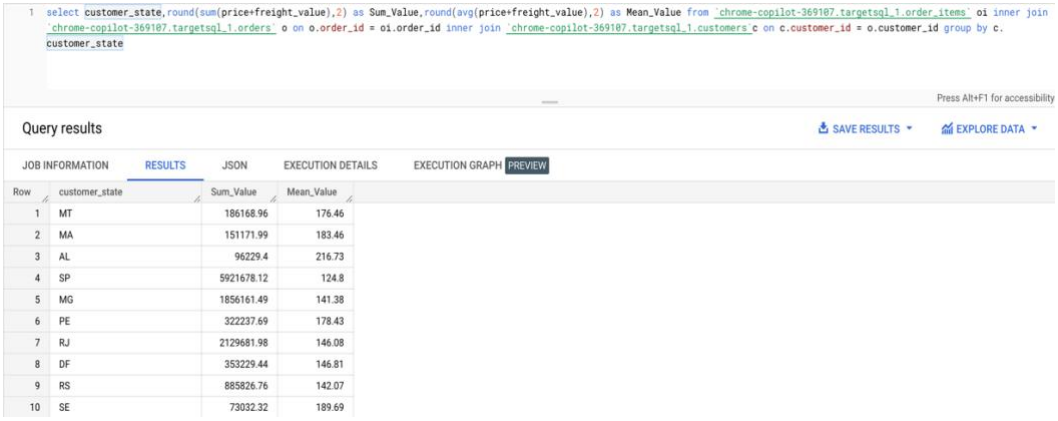
SAVE RESULTS | EXPLORE DATA

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

Row	Day	year	month	price	sumTotal	Lag_check	percentage
1	2017-01-05	2017	1	10.9	10.9	11.0	-0.909
2	2017-01-05	2017	1	10.9	10.9	11.0	-0.909
3	2017-01-05	2017	1	2.9	2.9	1740.0	-99.833
4	2017-01-05	2017	1	7.9	7.9	8.0	-1.25
5	2017-01-05	2017	1	9.9	9.9	19.0	-47.895
6	2017-01-05	2017	1	10.9	10.9	11.0	-0.909
7	2017-01-05	2017	1	11.9	11.9	12.0	-0.833
8	2017-01-05	2017	1	10.9	10.9	11.0	-0.909
9	2017-01-05	2017	1	10.9	10.9	11.0	-0.909
10	2017-01-05	2017	1	10.9	10.9	11.0	-0.909

1. 4.2. **Mean & Sum of price and freight value by customer state**

```
select customer_state,round(sum(price+freight_value),2) as Sum_Value,round(avg(price+freight_value),2) as Mean_Value from `chrome-copilot-369107.targetsql_1.order_items` oi inner join `chrome-copilot-369107.targetsql_1.orders` o on o.order_id = oi.order_id inner join `chrome-copilot-369107.targetsql_1.customers`c on c.customer_id = o.customer_id group by c.customer_state
```



Query results

SAVE RESULTS | EXPLORE DATA

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

Row	customer_state	Sum_Value	Mean_Value
1	MT	186168.96	176.46
2	MA	151171.99	183.46
3	AL	96229.4	216.73
4	SP	5921678.12	124.8
5	MG	1856161.49	141.38
6	PE	322237.69	178.43
7	RJ	2129681.98	146.08
8	DF	353229.44	146.81
9	RS	885826.76	142.07
10	SE	73032.32	189.69

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

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

```
select order_id,extract(DATE FROM DATETIME(order_purchase_timestamp)) AS Purchased,extract(DATE FROM DATETIME(order_delivered_customer_date)) AS Delivered,extract(DATE FROM DATETIME(order_estimated_delivery_date)) AS Estimated_delivery,DATE_DIFF(order_estimated_delivery_date,order_delivered_customer_date,DAY) AS DIFFERENCE from `chrome-copilot 369107.targetsql_1.orders` where order_purchase_timestamp is not null and order_delivered_customer_date is not null and order_estimated_delivery_date is not null
```

```
select order_id,
extract(DATE FROM DATETIME(order_purchase_timestamp)) AS Purchased,
extract(DATE FROM DATETIME(order_delivered_customer_date)) AS Delivered,
extract(DATE FROM DATETIME(order_estimated_delivery_date)) AS Estimated_delivery,
DATE_DIFF(order_estimated_delivery_date,order_delivered_customer_date,DAY) AS DIFFERENCE
from `chrome-copilot-369107.targetsql_1.orders`
where order_purchase_timestamp is not null and order_delivered_customer_date is not null and order_estimated_delivery_date is not nu
```

Query results

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

Row	order_id	Purchased	Delivered	Estimated_delivery	DIFFERENCE
1	770d331c84e5b214bd9dc70a...	2016-10-07	2016-10-14	2016-11-29	45
2	1950d777989f6a877539f5379...	2018-02-19	2018-03-21	2018-03-09	-12
3	dabf2b0e35b423f94618bf965f...	2016-10-09	2016-10-16	2016-11-30	44
4	8beb59392e21af5eb9547ae1a...	2016-10-08	2016-10-19	2016-11-30	41
5	b60b53ad0bb7dacacf2989fe2...	2017-05-10	2017-05-23	2017-05-18	-5
6	276e9ec344d3bf029ff83a161c...	2017-04-08	2017-05-22	2017-05-18	-4
7	1a0b31f08d0d7e87935b819ed...	2017-04-11	2017-04-18	2017-05-18	29
8	cec8f5f7a13e5ab934a486ec9e...	2017-03-17	2017-04-07	2017-05-18	40
9	54e1a3c2b97fb0809da548a59...	2017-04-11	2017-05-22	2017-05-18	-4
10	58527ee4726911bee84a0f42c...	2017-03-20	2017-03-30	2017-05-18	48

5.2(a)

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

```
select order_id,extract(DATE FROM DATETIME(order_purchase_timestamp)) AS
Order_Purchased,extract(DATE FROM DATETIME(order_delivered_customer_date))
ASOrder_Delivered,date_diff(order_purchase_timestamp,order_delivered_customer_date,Day) as
Time_To_Delivery from `chrome-copilot 369107.targetsql_1.orders`where
order_delivered_customer_date is not null
order by Order_Purchased,Order_Delivered asc limit 100
```

```
select order_id,
extract(DATE FROM DATETIME(order_purchase_timestamp)) AS Order_Purchased,
extract(DATE FROM DATETIME(order_delivered_customer_date)) AS Order_Delivered,
date_diff(order_purchase_timestamp,order_delivered_customer_date,Day) as Time_To_Delivery
from `chrome-copilot-369107.targetsql_1.orders`
where order_delivered_customer_date is not null
order by Order_Purchased,Order_Delivered asc
limit 100
```

Query results SAVE

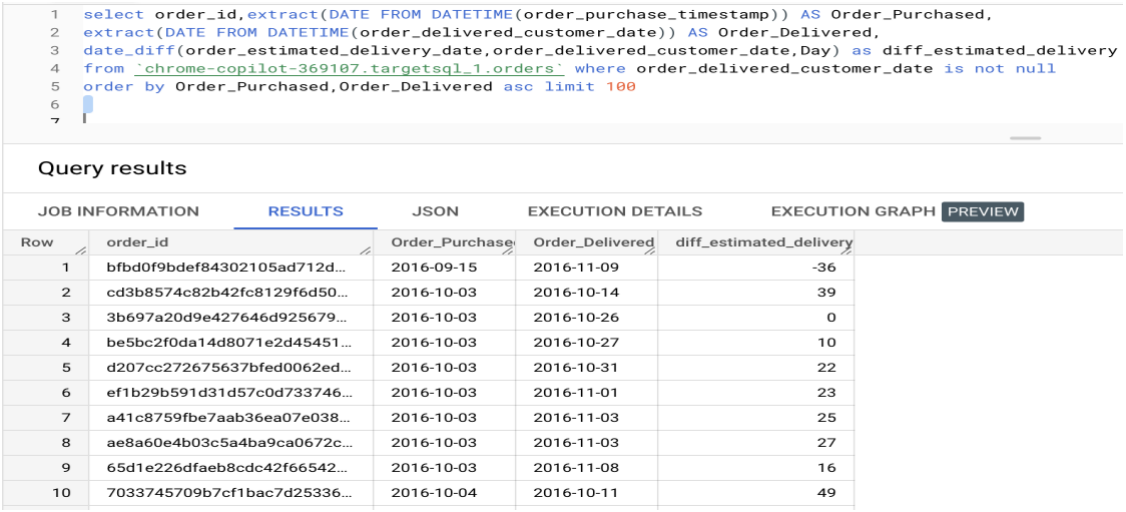
JOB INFORMATION | RESULTS | JSON | EXECUTION DETAILS | EXECUTION GRAPH | PREVIEW

Row	order_id	Order_Purchase	Order_Delivered	Time_To_Deliver
1	bfbd0f9bdef84302105ad712d...	2016-09-15	2016-11-09	-54
2	cd3b8574c82b42fc8129f6d50...	2016-10-03	2016-10-14	-10
3	3b697a20d9e427646d925679...	2016-10-03	2016-10-26	-23
4	be5bc2f0da14d8071e2d45451...	2016-10-03	2016-10-27	-24
5	d207cc272675637bfed0062ed...	2016-10-03	2016-10-31	-27
6	ef1b29b591d31d57c0d733746...	2016-10-03	2016-11-01	-28
7	a41c8759fbe7aab36ea07e038...	2016-10-03	2016-11-03	-30
8	ae8a60e4b03c5a4ba9ca0672c...	2016-10-03	2016-11-03	-30
9	65d1e226dfaeb8cdc42f66542...	2016-10-03	2016-11-08	-35
10	7033745709b7cf1bac7d25336...	2016-10-04	2016-10-11	-7

5.2(b)

diff_estimated_delivery = order_estimated_delivery_date-order_delivered_customer_date

```
select order_id,extract(DATE FROM DATETIME(order_purchase_timestamp)) AS Order_Purchased,
extract(DATE FROM DATETIME(order_delivered_customer_date)) AS Order_Delivered,
date_diff(order_estimated_delivery_date,order_delivered_customer_date,Day) as diff_estimated_delivery
from `chrome-copilot-369107.targetsql_1.orders` where order_delivered_customer_date is not null
order by Order_Purchased,Order_Delivered asc limit 100
```



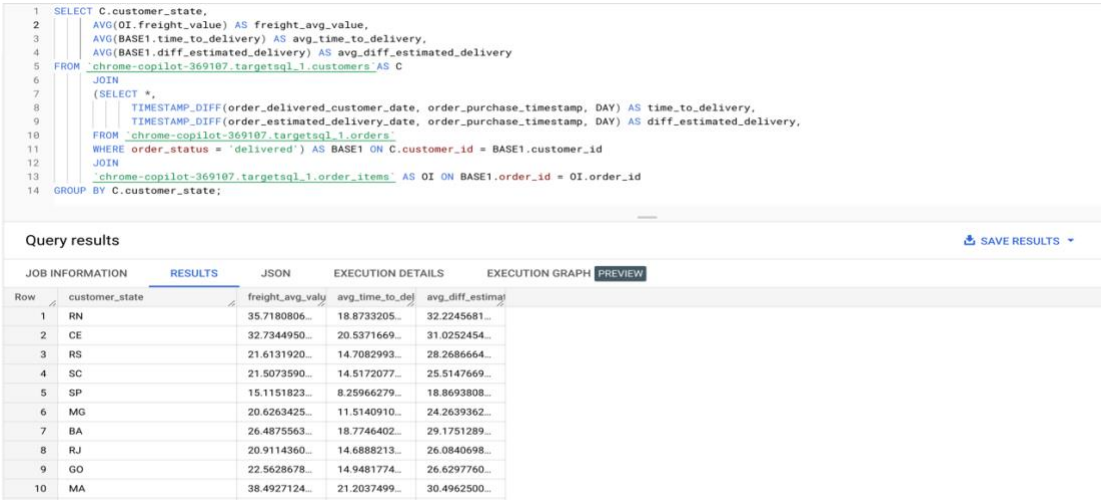
Query results

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

Row	order_id	Order_Purchase	Order_Delivered	diff_estimated_delivery
1	bfbd0f9bdef84302105ad712d...	2016-09-15	2016-11-09	-36
2	cd3b8574c82b42fc8129f6d50...	2016-10-03	2016-10-14	39
3	3b697a20d9e427646d925679...	2016-10-03	2016-10-26	0
4	be5bc2f0da14d8071e2d45451...	2016-10-03	2016-10-27	10
5	d207cc272675637bfed0062ed...	2016-10-03	2016-10-31	22
6	ef1b29b591d31d57c0d733746...	2016-10-03	2016-11-01	23
7	a41c8759fbe7aab36ea07e038...	2016-10-03	2016-11-03	25
8	ae8a60e4b03c5a4ba9ca0672c...	2016-10-03	2016-11-03	27
9	65d1e226dfaeb8cdc42f66542...	2016-10-03	2016-11-08	16
10	7033745709b7cf1bac7d25336...	2016-10-04	2016-10-11	49

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

```
SELECT C.customer_state,
AVG(OI.freight_value) AS freight_avg_value,
AVG(BASE1.time_to_delivery) AS avg_time_to_delivery,
AVG(BASE1.diff_estimated_delivery) AS avg_diff_estimated_delivery
FROM `chrome-copilot-369107.targetsql_1.customers`AS C
JOIN
(SELECT *,
TIMESTAMP_DIFF(order_delivered_customer_date, order_purchase_timestamp, DAY) AS
time_to_delivery,
TIMESTAMP_DIFF(order_estimated_delivery_date, order_purchase_timestamp, DAY) AS
diff_estimated_delivery,
FROM `chrome-copilot-369107.targetsql_1.orders`
WHERE order_status = 'delivered') AS BASE1 ON C.customer_id = BASE1.customer_id JOIN
`chrome-copilot-369107.targetsql_1.order_items` AS OI ON BASE1.order_id = OI.order_id
GROUP BY C.customer_state;
```



```
SELECT C.customer_state,
       AVG(OI.freight_value) AS freight_avg_value,
       AVG(BASE1.time_to_delivery) AS avg_time_to_delivery,
       AVG(BASE1.diff_estimated_delivery) AS avg_diff_estimated_delivery
FROM `chrome-copilot-369107.targetsql_1.customers` AS C
    JOIN
    (SELECT *,
            TIMESTAMP_DIFF(order_delivered_customer_date, order_purchase_timestamp, DAY) AS time_to_delivery,
            TIMESTAMP_DIFF(order_estimated_delivery_date, order_purchase_timestamp, DAY) AS diff_estimated_delivery,
     FROM `chrome-copilot-369107.targetsql_1.orders`
     WHERE order_status = 'delivered') AS BASE1 ON C.customer_id = BASE1.customer_id
     JOIN
     `chrome-copilot-369107.targetsql_1.order_items` AS OI ON BASE1.order_id = OI.order_id
GROUP BY C.customer_state;
```

Query results

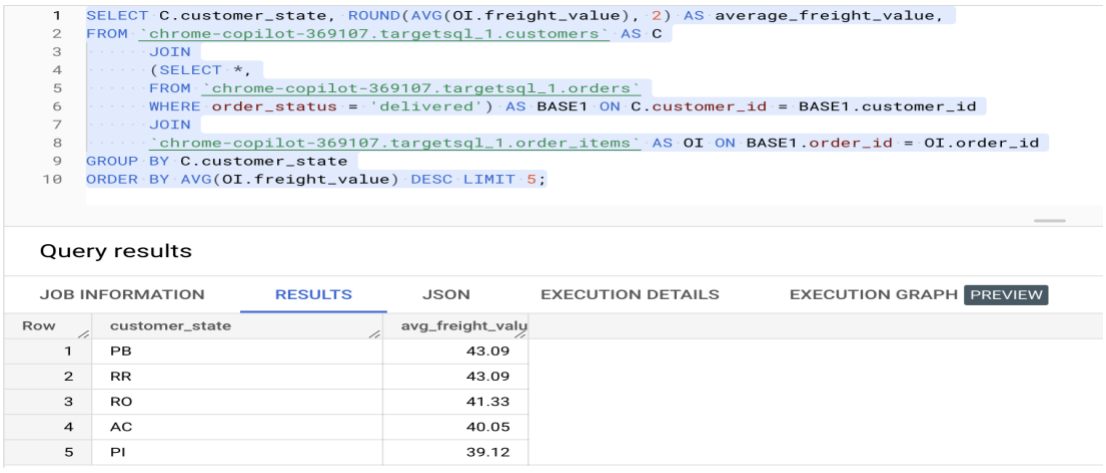
SAVE RESULTS

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

Row	customer_state	freight_avg_valu	avg_time_to_del	avg_diff_estimat
1	RN	35.7180806...	18.8733205...	32.2245681...
2	CE	32.7344950...	20.5371669...	31.0252454...
3	RS	21.6131920...	14.7082993...	28.2686664...
4	SC	21.5073590...	14.5172077...	25.5147669...
5	SP	15.1151823...	8.25966279...	18.8693808...
6	MG	20.6263425...	11.5140910...	24.2639362...
7	BA	26.4875563...	18.7746402...	29.1751289...
8	RJ	20.9114360...	14.6888213...	26.0840698...
9	GO	22.5628678...	14.9481774...	26.6297760...
10	MA	38.4927124...	21.2037499...	30.4962500...

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

```
SELECT C.customer_state, ROUND(AVG(OI.freight_value), 2) AS average_freight_value,FROM
`chrome-copilot-369107.targetsql_1.customers` AS C JOIN
(SELECT *,
FROM `chrome-copilot-369107.targetsql_1.orders`
WHERE order_status = 'delivered') AS BASE1 ON C.customer_id = BASE1.customer_id
JOIN
`chrome-copilot-369107.targetsql_1.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;
```



```
SELECT C.customer_state, ROUND(AVG(OI.freight_value), 2) AS average_freight_value,
FROM `chrome-copilot-369107.targetsql_1.customers` AS C
    JOIN
    (SELECT *,
    FROM `chrome-copilot-369107.targetsql_1.orders`
    WHERE order_status = 'delivered') AS BASE1 ON C.customer_id = BASE1.customer_id
    JOIN
    `chrome-copilot-369107.targetsql_1.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;
```

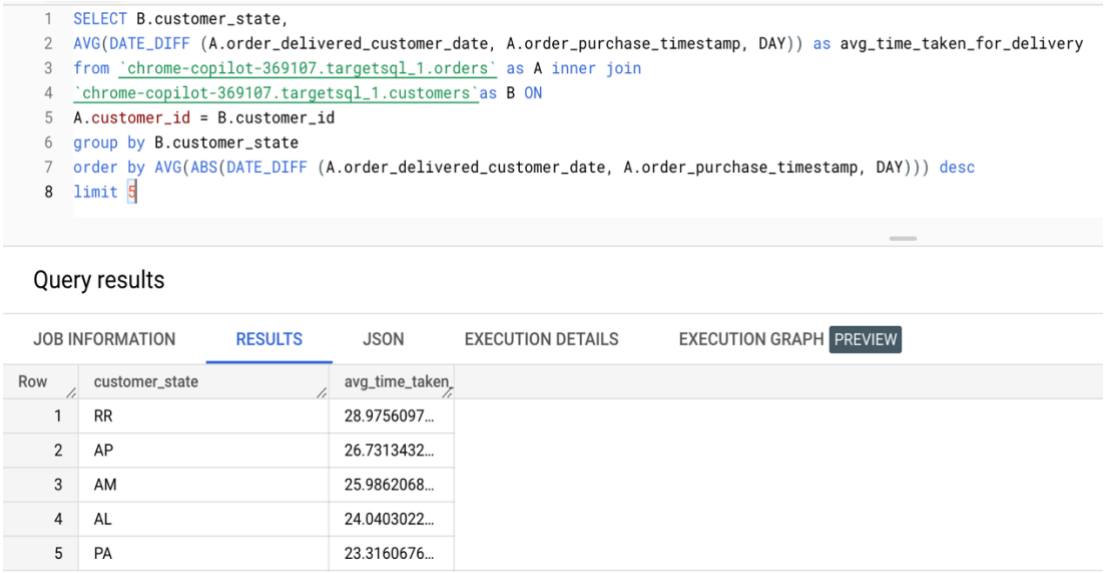
Query results

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

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

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

```
SELECT B.customer_state,
AVG(DATE_DIFF (A.order_delivered_customer_date, A.order_purchase_timestamp, DAY)) as
avg_time_taken_for_delivery
from `chrome-copilot-369107.targetsql_1.orders` as A inner join
`chrome-copilot-369107.targetsql_1.customers`as B ON
A.customer_id = B.customer_id
group by B.customer_state
order by AVG(ABS(DATE_DIFF (A.order_delivered_customer_date, A.order_purchase_timestamp,
DAY))) desc
limit 5
```



```
1  SELECT B.customer_state,
2  AVG(DATE_DIFF (A.order_delivered_customer_date, A.order_purchase_timestamp, DAY)) as avg_time_taken_for_delivery
3  from `chrome-copilot-369107.targetsql_1.orders` as A inner join
4  `chrome-copilot-369107.targetsql_1.customers`as B ON
5  A.customer_id = B.customer_id
6  group by B.customer_state
7  order by AVG(ABS(DATE_DIFF (A.order_delivered_customer_date, A.order_purchase_timestamp, DAY))) desc
8  limit 5
```

Query results

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

Row	customer_state	avg_time_taken_
1	RR	28.9756097...
2	AP	26.7313432...
3	AM	25.9862068...
4	AL	24.0403022...
5	PA	23.3160676...

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

```
SELECT B.customer_state,
ABS(AVG(DATE_DIFF (A.order_delivered_customer_date, A.order_purchase_timestamp, DAY))) as
total_time_of_delivery,
ABS(AVG(DATE_DIFF (A.order_estimated_delivery_date, A.order_purchase_timestamp, DAY))) as
estimated_delivery_time,
AVG(DATE_DIFF (A.order_estimated_delivery_date, A.order_delivered_customer_date,DAY)) as
diff_estmtd_and_delivey
FROM `chrome-copilot-369107.targetsql_1.orders` as A inner join
`chrome-copilot-369107.targetsql_1.customers`as B ON
```

```
A.customer_id = B.customer_id
group by B.customer_state
order by AVG(DATE_DIFF (A.order_estimated_delivery_date, A.order_delivered_customer_date,DAY))
desc
limit 5 ;
```

```
SELECT B.customer_state,
ABS(AVG(DATE_DIFF (A.order_delivered_customer_date, A.order_purchase_timestamp, DAY))) as total_time_of_delivery,
ABS(AVG(DATE_DIFF (A.order_estimated_delivery_date, A.order_purchase_timestamp, DAY))) as estimated_delivery_time,
AVG(DATE_DIFF (A.order_estimated_delivery_date, A.order_delivered_customer_date,DAY)) as diff_estmtd_and_delivey
FROM `chrome-copilot-369107.targetsql_1.orders` as A inner join
`chrome-copilot-369107.targetsql_1.customers`as B ON
A.customer_id = B.customer_id
group by B.customer_state
order by AVG(DATE_DIFF (A.order_estimated_delivery_date, A.order_delivered_customer_date,DAY)) desc
limit 5 ;
```

Query results

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

Row	customer_state	total_time_of_delivery	estimated_delivery_time	diff_estmtd_and
1	AC	20.637500000000...	40.765432098765423	19.7625000...
2	RO	18.913580246913...	38.4071146245059	19.1316872...
3	AP	26.731343283582...	45.70588235294117 4	18.7313432...
4	AM	25.986206896551...	44.756756756756765	18.6068965...
5	RR	28.975609756097...	46.173913043478251	16.4146341...

6. **Payment type analysis:**

6.1

Month over Month count of orders for different payment types

```
select *, round(((((order_cnt-mom))/mom)* 100),2) as mom_Percentage from (select *, lag(order_cnt)
over(partition by payment_type order by Year ASC, month ASC ) as mom from (select DISTINCT
year,month,payment_type,count(order_id)over(partition by payment_type, year,month) order_cnt from
(select distinct o.order_id, extract(year from o.order_purchase_timestamp) as year, extract(month from
o.order_purchase_timestamp) as month,p.payment_type as payment_type from `chrome-copilot-
369107.targetsql_1.orders` o
join `chrome-copilot-369107.targetsql_1.payments` p on o.order_id=p.order_id )b1)b2)b3
where mom is not null
order by payment_type
```

```
select *, round(((((order_cnt-mom))/mom)* 100),2) as mom_Percentage from (select *, lag(order_cnt) over(partition by payment_type order by Year ASC, month ASC ) as mom from (select
DISTINCT year,month,payment_type,count(order_id)over(partition by payment_type, year,month) order_cnt from (select distinct o.order_id, extract(year from o.order_purchase_timestamp) as
year, extract(month from o.order_purchase_timestamp) as month,p.payment_type as payment_type from `chrome-copilot-369107.targetsql_1.orders` o
join `chrome-copilot-369107.targetsql_1.payments` p on o.order_id=p.order_id )b1)b2)b3
where mom is not null
order by payment_type
```

Press Alt+F1 for accessibility

Query results | SAVE RESULTS | EXPLORE DATA

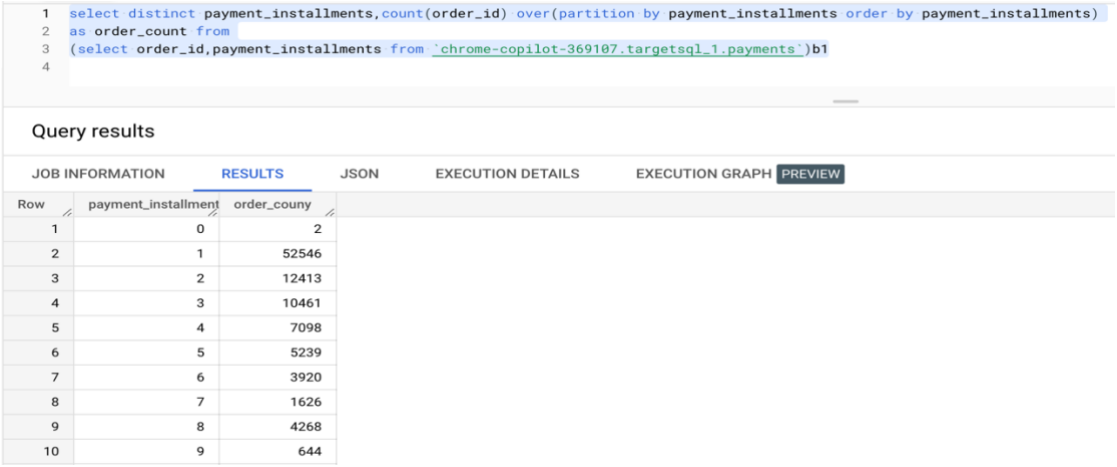
JOB INFORMATION | RESULTS | JSON | EXECUTION DETAILS | EXECUTION GRAPH | PREVIEW

Row	year	month	payment_type	order_cnt	mom	mom_Percentag
1	2017	1	UPI	197	63	212.7
2	2017	2	UPI	398	197	102.03
3	2017	3	UPI	590	398	48.24
4	2017	4	UPI	496	590	-15.93
5	2017	5	UPI	772	496	55.65
6	2017	6	UPI	707	772	-8.42
7	2017	7	UPI	845	707	19.52
8	2017	8	UPI	938	845	11.01
9	2017	9	UPI	903	938	-3.73
10	2017	10	UPI	993	903	9.97

6.2

Count of orders based on the no. of payment installments

```
select distinct payment_installments,count(order_id) over(partition by payment_installments order by payment_installments)
as order_count from
(select order_id,payment_installments from `chrome-copilot-369107.targetsql_1.payments`)b1
```



Query results

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

Row	payment_installment	order_couny
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
10	9	644