Q1. Business Case: Target SQL

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

1. *Data type of columns in a table*

1. customers.csv

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

2. geolocation.csv

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

3. order_items.csv

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

4. payments.csv

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

5. reviews.csv

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

6. orders.csv

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

7. products.csv

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

8. sellers.csv

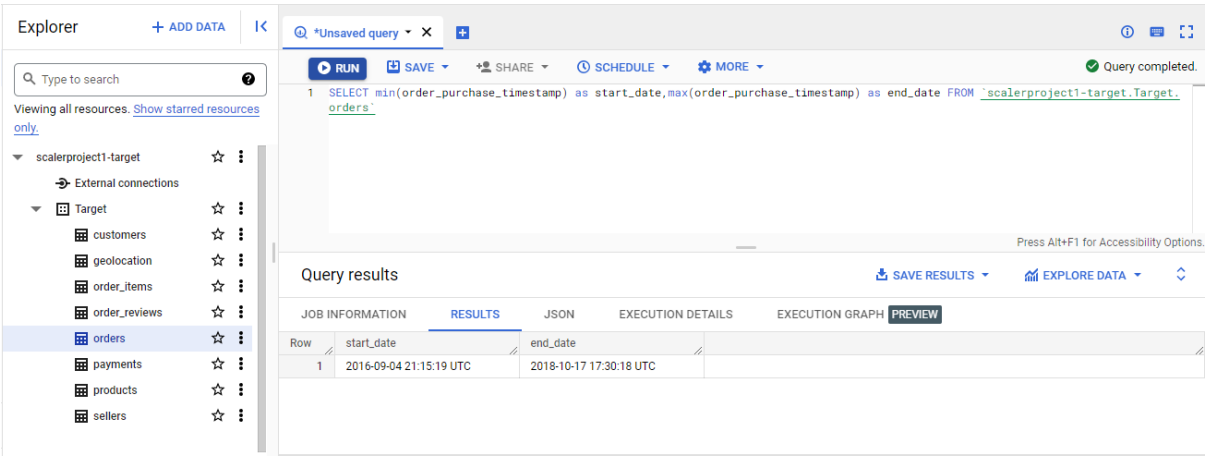
Field name	Type
seller_id	STRING
seller_zip_code_prefix	INTEGER
seller_city	STRING
seller_state	STRING

2. *Time period for which the data is given*

QUERY:

```
SELECT min(order_purchase_timestamp) as start_date,max(order_purchase_timestamp)
as end_date FROM `scalerproject1-target.Target.orders`
```

RESULT:

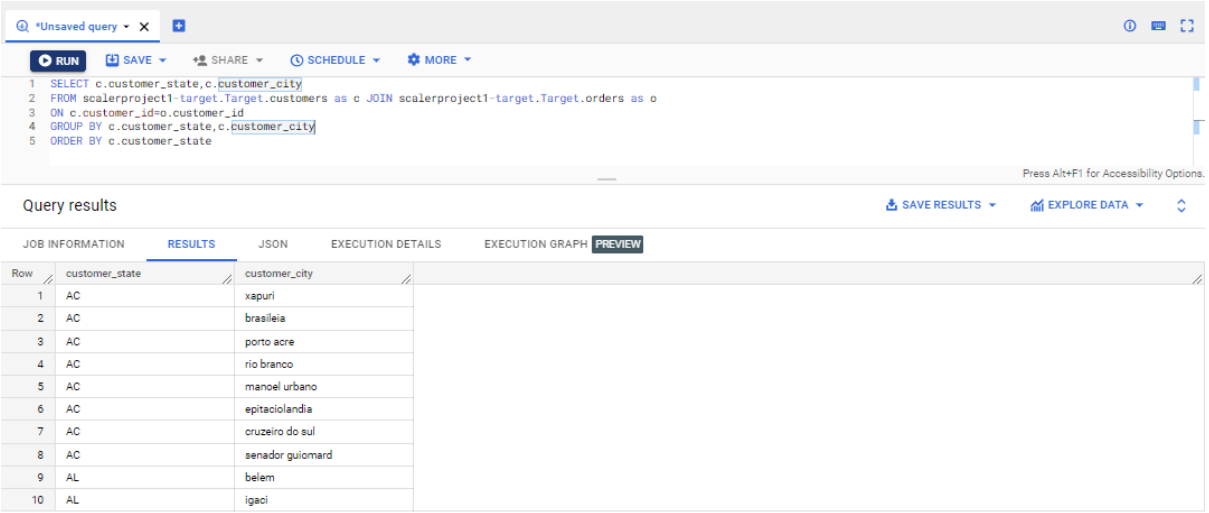


2. *Cities and States covered in the dataset*

QUERY:

```
SELECT c.customer_state,c.customer_city
FROM scalerproject1-target.Target.customers as c JOIN scalerproject1-
target.Target.orders as o
ON c.customer_id=o.customer_id
GROUP BY c.customer_state,c.customer_city
ORDER BY c.customer_state
```

RESULT:



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

2. In-depth Exploration:

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?*

QUERY:

```
SELECT *,RANK() OVER (partition by Year_of_purchase ORDER BY Total_payment DESC) AS
RANKER from
(SELECT EXTRACT(YEAR FROM o.order_purchase_timestamp) AS Year_of_purchase,EXTRACT(M
ONTH FROM o.order_purchase_timestamp) AS Month_of_purchase,SUM(p.payment_value) AS
Total_Payment
FROM scalerproject1-target.Target.payments as p LEFT JOIN scalerproject1-
target.Target.orders as o
ON p.order_id=o.order_id

GROUP BY Year_of_purchase,Month_of_purchase)
```

RESULT:

RUN SAVE SHARE SCHEDULE MORE

```
SELECT *,RANK() OVER (partition by Year_of_purchase ORDER BY Total_payment DESC) AS RANKER from
(SELECT EXTRACT(YEAR FROM o.order_purchase_timestamp) AS Year_of_purchase,EXTRACT(MONTH FROM o.order_purchase_timestamp) AS Month_of_purchase,SUM(p.
payment_value) AS Total_Payment
```

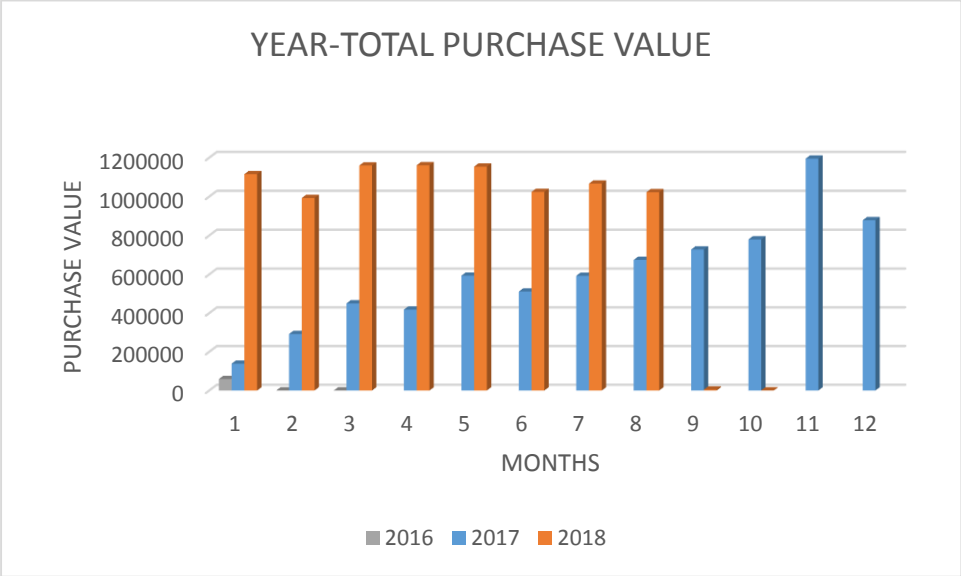
Press Alt+F1 for Accessibility Options.

Query results SAVE RESULTS EXPLORE DATA

JOB INFORMATION RESULTS JSON EXECUTION DETAILS EXECUTION GRAPH PREVIEW

Row	Year_of_purchas	Month_of_purch	Total_Payment	RANKER
1	2017	11	1194882.80...	1
2	2018	4	1160785.48...	1
3	2016	10	59090.4800...	1
4	2017	12	878401.480...	2
5	2018	3	1159652.11...	2
6	2016	9	252.24	2
7	2017	10	779677.880...	3
8	2018	5	1153982.15...	3
9	2016	12	19.62	3
10	2017	9	727762.450...	4

Results per page: 50 1 – 25 of 25



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

QUERY:

```
SELECT count(*) as Number_of_purchases,
CASE
WHEN time_purchase>=4 and time_purchase<6 THEN "dawn"
WHEN time_purchase>=6 and time_purchase<12 THEN "morning"
WHEN time_purchase>=12 and time_purchase<19 THEN "afternoon"
ELSE "night" end as time_of_purchase
FROM
(select
extract(HOUR from order_purchase_timestamp)
AS time_purchase
FROM scalerproject1-target.Target.orders)
GROUP BY time_of_purchase
ORDER BY Number_of_purchases DESC
```

RESULT:

```
SELECT count(*) as Number_of_purchases,
CASE
WHEN time_purchase>=4 and time_purchase<6 THEN "dawn"
WHEN time_purchase>=6 and time_purchase<12 THEN "morning"
WHEN time_purchase>=12 and time_purchase<19 THEN "afternoon"
ELSE "night" end as time_of_purchase
FROM
(select
extract(HOUR from order_purchase_timestamp)
AS time_purchase
FROM scalerproject1-target.Target.orders)
GROUP BY time_of_purchase
ORDER BY Number_of_purchases DESC
```

Query results

Row	Number_of_purc	time_of_purchase
1	44130	afternoon
2	32677	night
3	22240	morning
4	394	dawn

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

1. *Get month on month orders by region, states*

QUERY:

```
SELECT EXTRACT(MONTH FROM o.order_purchase_timestamp) AS month_of_purchase,c
.customer_city as city ,c.customer_state as state,
c.customer_zip_code_prefix as region,count(*) as No_of_purchases
FROM scalerproject1-target.Target.orders as o JOIN scalerproject1-
target.Target.customers as c
ON c.customer_id=o.customer_id
GROUP BY region,city,state,month_of_purchase
ORDER BY month_of_purchase,city
```

RESULT:

Query results

Row	month_of_purch	city	state	region	No_of_purchase
1	1	abadiania	GO	72940	1
2	1	abreu e lima	PE	53525	1
3	1	acarau	CE	62580	2
4	1	acopiara	CE	63560	2
5	1	acreuna	GO	75960	1
6	1	acu	RN	59650	1
7	1	adamantina	SP	17800	1
8	1	adolfo	SP	15230	1
9	1	afonso claudio	ES	29600	2
10	1	agronomica	SC	89188	1

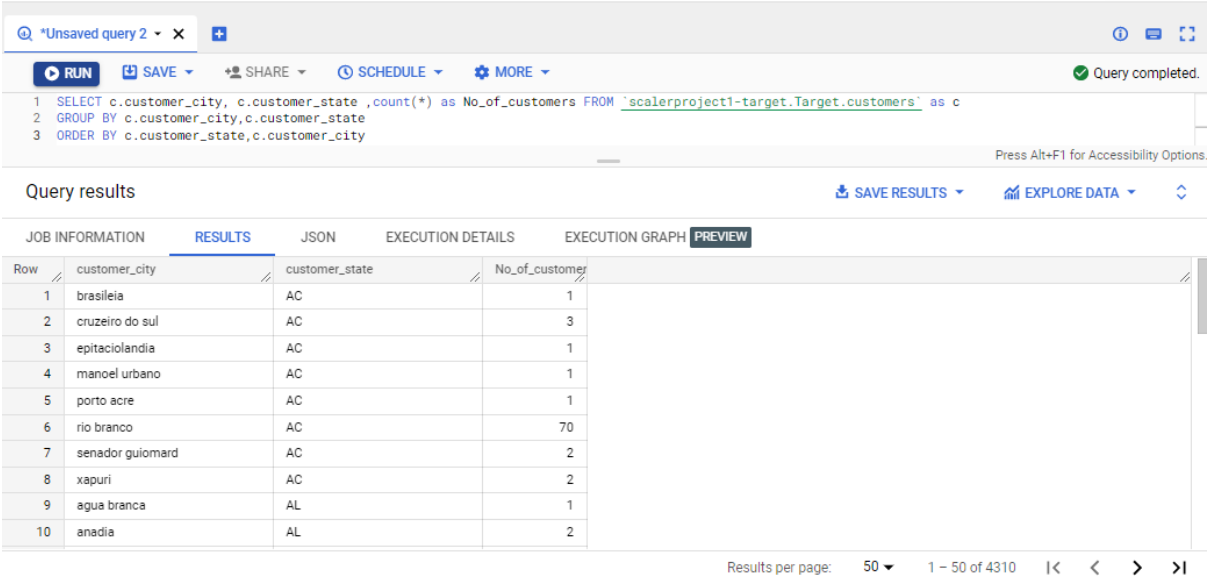
Load more

2. *How are customers distributed in Brazil*

QUERY:

```
SELECT c.customer_city, c.customer_state ,count(*) as No_of_customers FROM `
scalerproject1-target.Target.customers` as c
GROUP BY c.customer_city,c.customer_state
```

RESULT:



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

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

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

QUERY:

```
with purchase_table as
(SELECT Year_of_purchase,sum(Total_Payment) as Yearly_sum FROM
(SELECT SUM(p.payment_value) AS Total_Payment,EXTRACT(MONTH FROM o.order_pur
chase_timestamp) AS Month_of_purchase,
EXTRACT(YEAR FROM o.order_purchase_timestamp) AS Year_of_purchase
FROM scalerproject1-target.Target.payments as p LEFT JOIN scalerproject1-
target.Target.orders as o
ON p.order_id=o.order_id

GROUP BY Year_of_purchase,Month_of_purchase
HAVING Month_of_purchase>=1 and Month_of_purchase<=8
ORDER BY Year_of_purchase)
GROUP BY Year_of_purchase
ORDER BY Year_of_purchase)
SELECT ((y2.Yearly_sum-
y1.Yearly_sum)/y1.Yearly_sum)*100 as Increase_in_cost_order  FROM purchase_t
able as y1 cross join purchase_table as y2
WHERE y2.Year_of_purchase>y1.year_of_purchase
```

RESULT:

RUN SAVE SHARE SCHEDULE MORE This query will process 8.14 MB when run.

```
with purchase_table as
(SELECT Year_of_purchase,sum(Total_Payment) as Yearly_sum FROM
(SELECT SUM(p.payment_value) AS Total_Payment,EXTRACT(MONTH FROM o.order_purchase_timestamp) AS Month_of_purchase,
EXTRACT(YEAR FROM o.order_purchase_timestamp) AS Year_of_purchase
FROM scalerproject1-target.Target.payments as p LEFT JOIN scalerproject1-target.Target.orders as o
ON p.order_id=o.order_id

GROUP BY Year_of_purchase,Month_of_purchase
HAVING Month_of_purchase>=1 and Month_of_purchase<=8
ORDER BY Year_of_purchase)
GROUP BY Year_of_purchase
ORDER BY Year_of_purchase)
SELECT ((y2.Yearly_sum-y1.Yearly_sum)/y1.Yearly_sum)*100 as Increase_in_cost_order  FROM purchase_table as y1 cross join purchase_table as y2
WHERE y2.Year_of_purchase>y1.year_of_purchase
```

Press Alt+F1 for Accessibility Options.

Query results SAVE RESULTS EXPLORE DATA

JOB INFORMATION RESULTS JSON EXECUTION DETAILS EXECUTION GRAPH PREVIEW

Row	Increase_in_cost
1	136.976871...

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

QUERY:

```
SELECT c.customer_state as State,ROUND(AVG(oi.price),3) AS Mean_price, ROUND
(SUM(oi.price),3) as Total_price,ROUND(AVG(oi.freight_value),3) AS Mean_frei
ght_value,ROUND(SUM(oi.freight_value),3) as Toatal_frieght_value

FROM `scalerproject1-
target.Target.order_items` as oi LEFT JOIN `scalerproject1-
target.Target.orders` as o
ON oi.order_id=o.order_id
LEFT JOIN scalerproject1-target.Target.customers as c
ON o.customer_id=c.customer_id
GROUP BY c.customer_state
```

RESULT:

RUN MORE SAVE SHARE SCHEDULE Query completed.

```
SELECT c.customer_state as State,ROUND(AVG(oi.price),3) AS Mean_price, ROUND(SUM(oi.price),3) as Total_price,ROUND(AVG(oi.freight_value),3) AS Mean_freight_value,ROUND(SUM(oi.
freight_value),3) as Toatal_frieght_value
FROM `scalerproject1-target.Target.order_items` as oi LEFT JOIN `scalerproject1-target.Target.orders` as o
ON oi.order_id=o.order_id
LEFT JOIN scalerproject1-target.Target.customers as c
ON o.customer_id=c.customer_id
GROUP BY c.customer_state
ORDER BY c.customer_state
```

Press Alt+F1 for Accessibility Options.

Query results SAVE RESULTS EXPLORE DATA

JOB INFORMATION RESULTS JSON EXECUTION DETAILS EXECUTION GRAPH PREVIEW

Row	State	Mean_price	Total_price	Mean_freight_va	Toatal_frieght_va
1	AC	173.728	15982.95	40.073	3686.75
2	AL	180.889	80314.81	35.844	15914.59
3	AM	135.496	22356.84	33.205	5478.89
4	AP	164.321	13474.3	34.006	2788.5
5	BA	134.601	511349.99	26.364	100156.68
6	CE	153.758	227254.71	32.714	48351.59
7	DF	125.771	302603.94	21.041	50625.5
8	ES	121.914	275037.31	22.059	49764.6
9	GO	126.272	294591.95	22.767	53114.98
10	MA	145.204	119648.22	38.257	31523.77
11	MG	120.749	1585308.03	20.63	270853.46

5. Analysis on sales, freight and delivery time

1. Calculate days between purchasing, delivering and estimated delivery

2. Create columns:

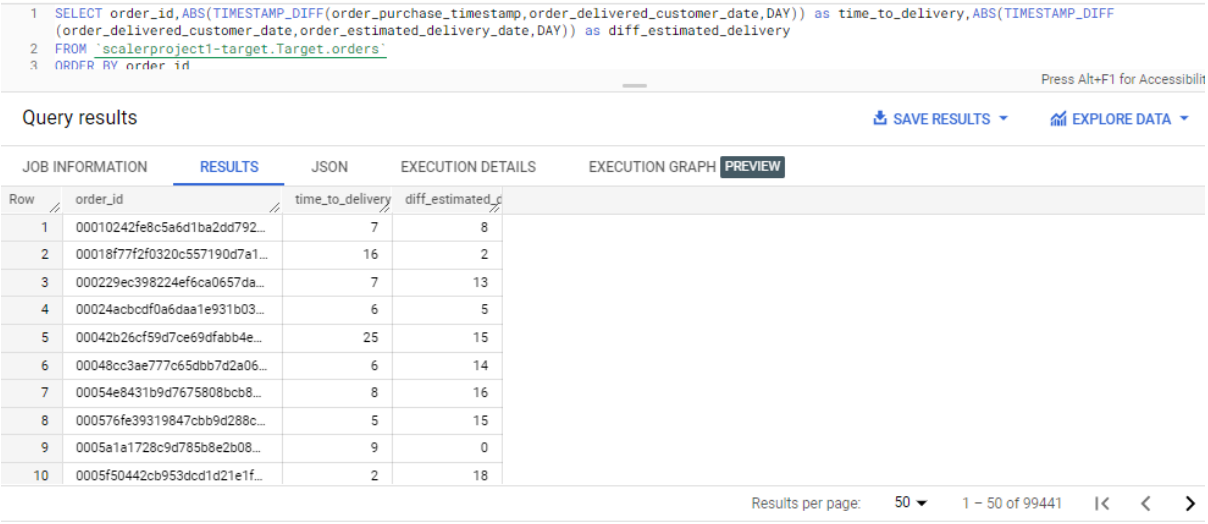
- *time_to_delivery = order_purchase_timestamp-order_delivered_customer_date*
- *diff_estimated_delivery = order_estimated_delivery_date-order_delivered_customer_date*

(Since the above equation will be giving a negative answer as delivery date is more than purchase date, I have used absolute function in the query)

QUERY:

```
SELECT order_id,ABS(TIMESTAMP_DIFF(order_purchase_timestamp,order_delivered_custome
r_date,DAY)) as time_to_delivery,ABS(TIMESTAMP_DIFF(order_delivered_customer_date,o
rder_estimated_delivery_date,DAY)) as diff_estimated_delivery
FROM `scalerproject1-target.Target.orders`
ORDER BY order_id
```

RESULT:



```
SELECT order_id,ABS(TIMESTAMP_DIFF(order_purchase_timestamp,order_delivered_customer_date,DAY)) as time_to_delivery,ABS(TIMESTAMP_DIFF
(order_delivered_customer_date,order_estimated_delivery_date,DAY)) as diff_estimated_delivery
FROM `scalerproject1-target.Target.orders`
ORDER BY order_id
```

Press Alt+F1 for Accessibilit

Query results — SAVE RESULTS — EXPLORE DATA

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

Row	order_id	time_to_delivery	diff_estimated_d
1	00010242fe8c5a6d1ba2dd792...	7	8
2	00018f77f2f0320c557190d7a1...	16	2
3	000229ec398224ef6ca0657da...	7	13
4	00024acbcdf0a6daa1e931b03...	6	5
5	00042b26cf59d7ce69dfabb4e...	25	15
6	00048cc3ae777c65dbb7d2a06...	6	14
7	00054e8431b9d7675808bcb8...	8	16
8	000576fe39319847cbb9d288c...	5	15
9	0005a1a1728c9d785b8e2b08...	9	0
10	0005f50442cb953dcd1d21e1f...	2	18

Results per page: 50 — 1 – 50 of 99441

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

QUERY:

```
SELECT ABS(AVG(time_to_delivery)) AS Avg_delivery_time,ABS(AVG( diff_estimated_delivery)) a
s Diff_in_delivery_time,AVG(freight_value) as Avg_freight_value,customer_state FROM
(SELECT *,TIMESTAMP_DIFF(order_purchase_timestamp,order_delivered_customer_date,DAY) as tim
e_to_delivery,TIMESTAMP_DIFF(order_delivered_customer_date,order_estimated_delivery_date,DA
Y) as diff_estimated_delivery
 FROM `scalerproject1-target.Target.orders`AS o JOIN scalerproject1-
target.Target.customers AS c
 ON o.customer_id=c.customer_id
 JOIN scalerproject1-target.Target.order_items AS oi
 ON o.order_id=oi.order_id
 )
```

```
GROUP BY customer_state
ORDER BY customer_state
```

RESULT:

```
SELECT ABS(AVG(time_to_delivery)) AS Avg_delivery_time,ABS(AVG( diff_estimated_delivery)) as Diff_in_delivery_time,AVG(freight_value) as
Avg_freight_value,customer_state FROM
(SELECT *,TIMESTAMP_DIFF(order_purchase_timestamp,order_delivered_customer_date,DAY) as time_to_delivery,TIMESTAMP_DIFF(order_delivered_customer_date,
```

Press Alt+F1 for Accessibility Optic

Query results

SAVE RESULTS EXPLORE DATA

JOB INFORMATION RESULTS JSON EXECUTION DETAILS EXECUTION GRAPH PREVIEW

Row	Avg_delivery_tim	Diff_in_delivery_	Avg_freight_valu	customer_state
1	20.3296703...	20.0109890...	40.0733695...	AC
2	23.9929742...	7.97658079...	35.8436711...	AL
3	25.9631901...	18.9754601...	33.2053939...	AM
4	27.7530864...	17.4444444...	34.0060975...	AP
5	18.7746402...	10.1194678...	26.3639589...	BA
6	20.5371669...	10.2566619...	32.7142016...	CE
7	12.5014861...	11.2747346...	21.0413549...	DF
8	15.1928089...	9.76853932...	22.0587765...	ES
9	14.9481774...	11.3728590...	22.7668152...	GO
10	21.2037500...	9.10999999...	38.2570024...	MA

Results per page: 50 1 – 27 of 27

4. *Sort the data to get the following:*

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

QUERY:

```
SELECT c.customer_state as State, ROUND(AVG(oi.freight_value),3) AS Mean_freight_value
FROM `scalerproject1-target.Target.order_items` as oi LEFT JOIN `scalerproject1-
target.Target.orders` as o
ON oi.order_id=o.order_id
LEFT JOIN scalerproject1-target.Target.customers as c
ON o.customer_id=c.customer_id
GROUP BY c.customer_state
ORDER BY Mean_freight_value
LIMIT 5
```

RESULT:

*Unsaved query 3 *Unsaved query 5

RUN SAVE SHARE SCHEDULE MORE

```
SELECT c.customer_state as State, ROUND(AVG(oi.freight_value),3) AS Mean_freight_value
FROM `scalerproject1-target.Target.order_items` as oi LEFT JOIN `scalerproject1-target.Target.orders` as o
ON oi.order_id=o.order_id
LEFT JOIN scalerproject1-target.Target.customers as c
ON o.customer_id=c.customer_id
GROUP BY c.customer_state
ORDER BY Mean_freight_value
LIMIT 5
```

Press Alt+F1 for Accessibility Options.

Query results

SAVE RESULTS EXPLORE DATA

JOB INFORMATION RESULTS JSON EXECUTION DETAILS EXECUTION GRAPH PREVIEW

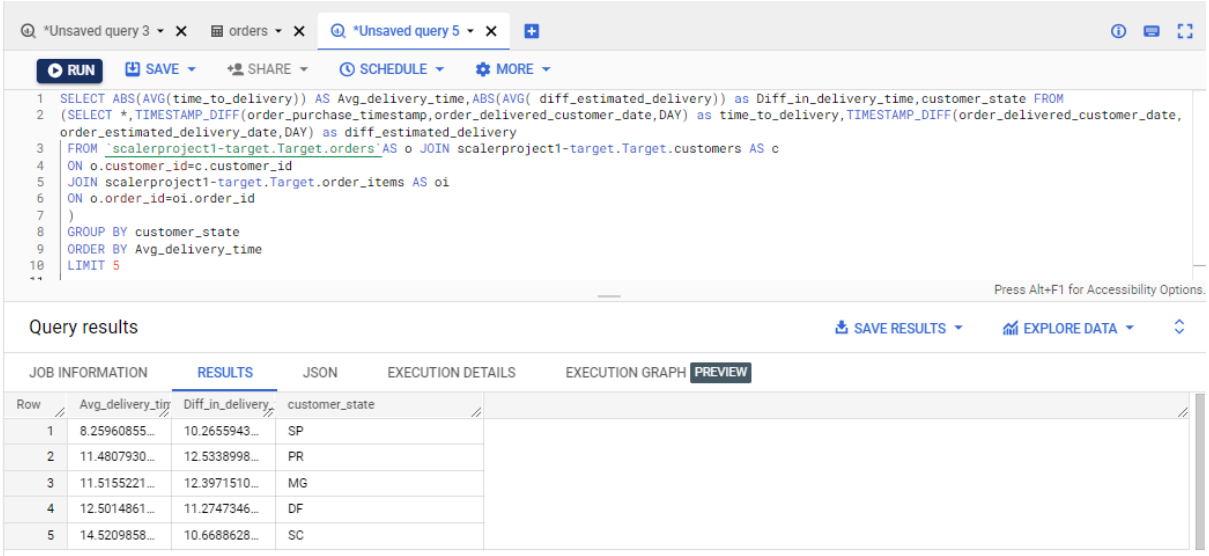
Row	State	Mean_freight_va
1	SP	15.147
2	PR	20.532
3	MG	20.63
4	RJ	20.961
5	DF	21.041

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

QUERY:

```
SELECT ABS(AVG(time_to_delivery)) AS Avg_delivery_time,ABS(AVG( diff_estimated_delivery)) a
s Diff_in_delivery_time,customer_state FROM
(SELECT *,TIMESTAMP_DIFF(order_purchase_timestamp,order_delivered_customer_date,DAY) as tim
e_to_delivery,TIMESTAMP_DIFF(order_delivered_customer_date,order_estimated_delivery_date,DA
Y) as diff_estimated_delivery
 FROM `scalerproject1-target.Target.orders`AS o JOIN scalerproject1-
target.Target.customers AS c
 ON o.customer_id=c.customer_id
 JOIN scalerproject1-target.Target.order_items AS oi
 ON o.order_id=oi.order_id
 )
 GROUP BY customer_state
 ORDER BY Avg_delivery_time
 LIMIT 5
```

RESULT:



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

QUERY:

```
SELECT ABS(AVG(time_to_delivery)) AS Avg_delivery_time,ABS(AVG( diff_estimated_delivery)) a
s Diff_in_delivery_time,customer_state FROM
(SELECT *,TIMESTAMP_DIFF(order_purchase_timestamp,order_delivered_customer_date,DAY) as tim
e_to_delivery,TIMESTAMP_DIFF(order_delivered_customer_date,order_estimated_delivery_date,DA
Y) as diff_estimated_delivery
 FROM `scalerproject1-target.Target.orders`AS o JOIN scalerproject1-
target.Target.customers AS c
 ON o.customer_id=c.customer_id
 JOIN scalerproject1-target.Target.order_items AS oi
 ON o.order_id=oi.order_id
 )
 GROUP BY customer_state
 ORDER BY Diff_in_delivery_time
 LIMIT 5
```

RESULT:

*Unsaved query 3 | orders | *Unsaved query 5

RUN | SAVE | SHARE | SCHEDULE | MORE

```
SELECT ABS(AVG(time_to_delivery)) AS Avg_delivery_time,ABS(AVG( diff_estimated_delivery)) as Diff_in_delivery_time,customer_state FROM
(SELECT *,TIMESTAMP_DIFF(order_purchase_timestamp,order_delivered_customer_date,DAY) as time_to_delivery,TIMESTAMP_DIFF(order_delivered_customer_date,
order_estimated_delivery_date,DAY) as diff_estimated_delivery
FROM `scalerproject1-target.Target.orders`AS o JOIN scalerproject1-target.Target.customers AS c
ON o.customer_id=c.customer_id
JOIN scalerproject1-target.Target.order_items AS oi
ON o.order_id=oi.order_id
)
GROUP BY customer_state
ORDER BY Diff_in_delivery_time
LIMIT 5
```

Press Alt+F1 for Accessibility Options.

Query results | SAVE RESULTS | EXPLORE DATA

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

Row	Avg_delivery_tim	Diff_in_delivery_	customer_state
1	23.9929742...	7.97658079...	AL
2	21.2037500...	9.10999999...	MA
3	20.9786666...	9.16533333...	SE
4	15.1928089...	9.76853932...	ES
5	18.7746402...	10.1194678...	BA

6. *Payment type analysis:*

1. Month over Month count of orders for different payment types

QUERY:

```
SELECT EXTRACT(MONTH FROM o.order_purchase_timestamp) AS month_of_purchase,p.paymen
t_type,count(*) as No_of_purchases

FROM scalerproject1-target.Target.orders as o JOIN scalerproject1-
target.Target.payments as p
ON o.order_id=p.order_id
GROUP BY month_of_purchase,p.payment_type
ORDER BY month_of_purchase,p.payment_type
```

RESULT:

RUN | SAVE | SHARE | SCHEDULE | MORE

```
SELECT EXTRACT(MONTH FROM o.order_purchase_timestamp) AS month_of_purchase,p.payment_type,count(*) as No_of_purchases
FROM scalerproject1-target.Target.orders as o JOIN scalerproject1-target.Target.payments as p
ON o.order_id=p.order_id
GROUP BY month_of_purchase,p.payment_type
ORDER BY month_of_purchase,p.payment_type
```

Press Alt+F1 for Accessibility Options.

Query results | SAVE RESULTS | EXPLORE DATA

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

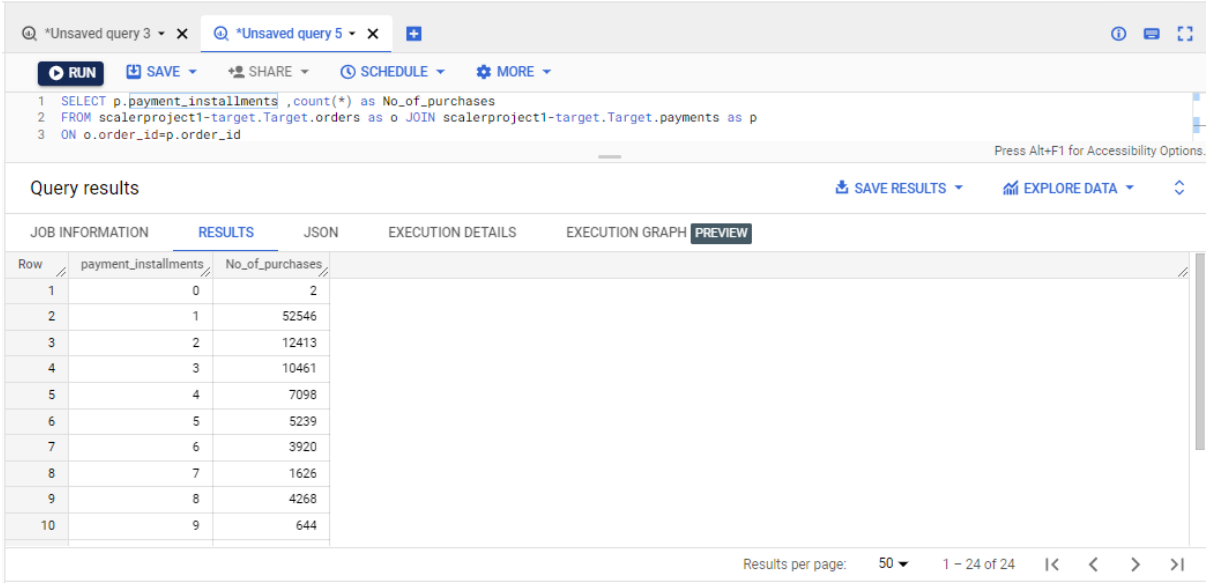
Row	month_of_purch	payment_type	No_of_purchase
1	1	UPI	1715
2	1	credit_card	6103
3	1	debit_card	118
4	1	voucher	477
5	2	UPI	1723
6	2	credit_card	6609
7	2	debit_card	82
8	2	voucher	424
9	3	UPI	1942
10	3	credit_card	7707

2. Count of orders based on the no. of payment installments

QUERY:

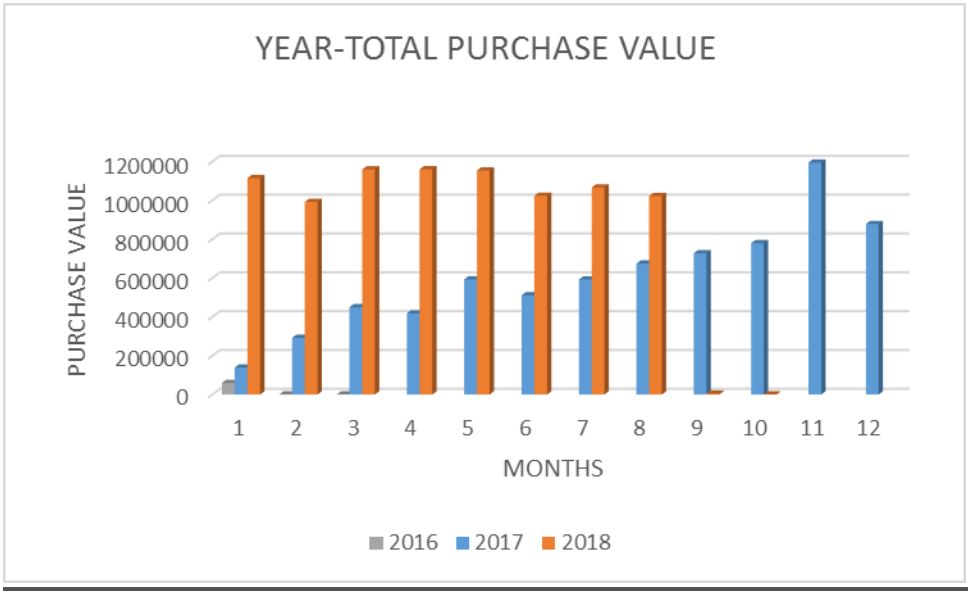
```
SELECT p.payment_installments ,count(*) as No_of_purchases
FROM scalerproject1-target.Target.orders as o JOIN scalerproject1-
target.Target.payments as p
ON o.order_id=p.order_id
GROUP BY payment_installments
ORDER BY payment_installments
```

RESULT:



Row	payment_installments	No_of_purchases
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

Actionable Insights



- The above graph shows the purchase value, i.e. the monthly total purchase value.

But since the data of 2016 and 2018 is not available for all the months, we couldn't get a proper comparison. But 2017 shows an increase in purchase along the year.

Also the purchase value of 2018 is more than respective months of 2017. That shows an increment across years.

There is an increment of 136% in purchase value in 2018 compared to 2017.

- As per the analysis it is shown that most of the Brazilian customers tends to do purchases during afternoon.
- Most of the customers are purchasing with single installments of payment.
- Among the payment types debit cards shows lesser number of transactions whereas credit card shows maximum
- Most of the customers are purchasing during afternoon

Recommendations

- Providing more offers in collaboration with debit cards and target may help in increasing customers purchase
- The data of 2016 and 2018 doesn't seem to be reliable as data from some months are not available.