TARGET SQL CASE STUDY BY PRANAV CHADDA

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

1. Data type of columns in a table
2. Time period for which the data is given
3. Cities and States of customers ordered during the given period

1. Data type of columns in a table

```
SELECT *
FROM information_schema.customers
WHERE table_name = 'scaler-ds-ml-de-372817.TARGET';
```

Filter Enter property name or value

Field name	Type	Mode	Collation	Default Value	Policy Tags	D
geolocation_zip_code_prefix	INTEGER	NULLABLE				
geolocation_lat	FLOAT	NULLABLE				
geolocation_lng	FLOAT	NULLABLE				
geolocation_city	STRING	NULLABLE				
geolocation_state	STRING	NULLABLE				

EDIT SCHEMA VIEW ROW ACCESS POLICIES

Filter Enter property name or value

Field name	Type	Mode	Collation	Default Value	Policy Tags	Descri
order_id	STRING	NULLABLE				
order_item_id	INTEGER	NULLABLE				
product_id	STRING	NULLABLE				
seller_id	STRING	NULLABLE				
shipping_limit_date	TIMESTAMP	NULLABLE				

EDIT SCHEMA VIEW ROW ACCESS POLICIES

Filter Enter property name or value

Field name	Type	Mode	Collation	Default Value	Policy Tags
review_id	STRING	NULLABLE			
order_id	STRING	NULLABLE			
review_score	INTEGER	NULLABLE			
review_comment_title	STRING	NULLABLE			
review_creation_date	TIMESTAMP	NULLABLE			

EDIT SCHEMA VIEW ROW ACCESS POLICIES

Filter Enter property name or value

Field name	Type	Mode	Collation	Default Value	Policy Tags
order_id	STRING	NULLABLE			
customer_id	STRING	NULLABLE			
order_status	STRING	NULLABLE			
order_purchase_timestamp	TIMESTAMP	NULLABLE			
order_approved_at	TIMESTAMP	NULLABLE			

EDIT SCHEMA VIEW ROW ACCESS POLICIES

Filter Enter property name or value

Field name	Type	Mode	Collation	Default Value	Policy Tags	Descrip
order_id	STRING	NULLABLE				
payment_sequential	INTEGER	NULLABLE				
payment_type	STRING	NULLABLE				
payment_installments	INTEGER	NULLABLE				
payment_value	FLOAT	NULLABLE				

EDIT SCHEMA VIEW ROW ACCESS POLICIES

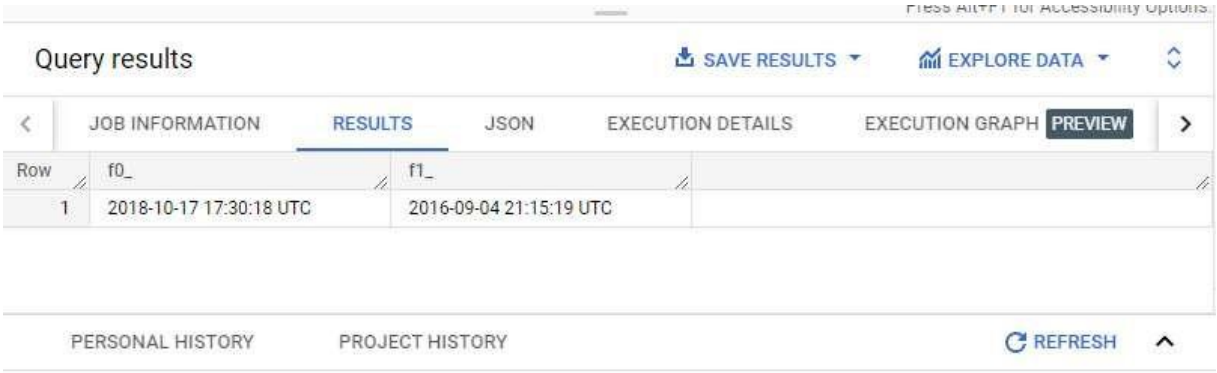
Filter Enter property name or value

Field name	Type	Mode	Collation	Default Value	Policy Tags	De
product_id	STRING	NULLABLE				
product_category	STRING	NULLABLE				
product_name_length	INTEGER	NULLABLE				
product_description_length	INTEGER	NULLABLE				
product_photos_qty	INTEGER	NULLABLE				

EDIT SCHEMA VIEW ROW ACCESS POLICIES

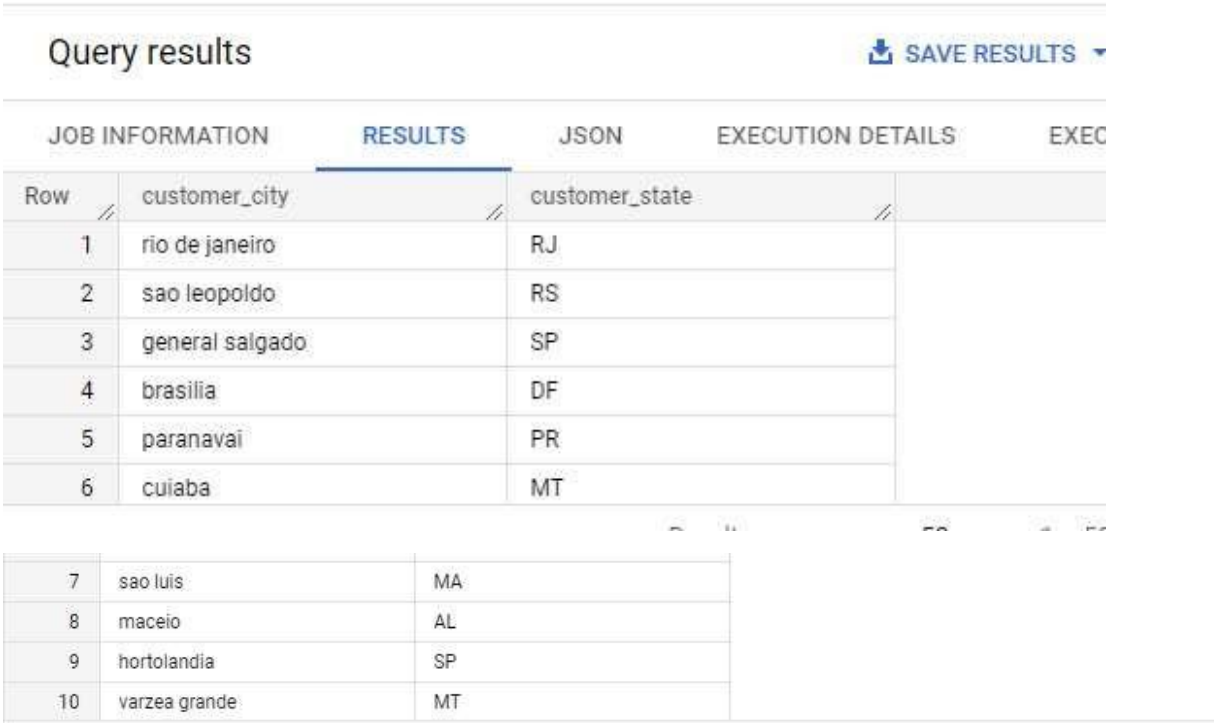
2. Time period for which the data is given

```
SELECT
MAX(order_purchase_timestamp),
MIN(order_purchase_timestamp) FROM `scaler-ds-ml-de-372817.TARGET.orders`
```



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

```
SELECT DISTINCT
  c.customer_city,
  c.customer_state
FROM `TARGET.orders` as o
INNER JOIN `TARGET.customers` as c
ON o.customer_id = c.customer_id;
```



Row	customer_city	customer_state
1	rio de janeiro	RJ
2	sao leopoldo	RS
3	general salgado	SP
4	brasilia	DF
5	paranavai	PR
6	cuiaba	MT
7	sao luis	MA
8	maceio	AL
9	hortolandia	SP
10	varzea grande	MT

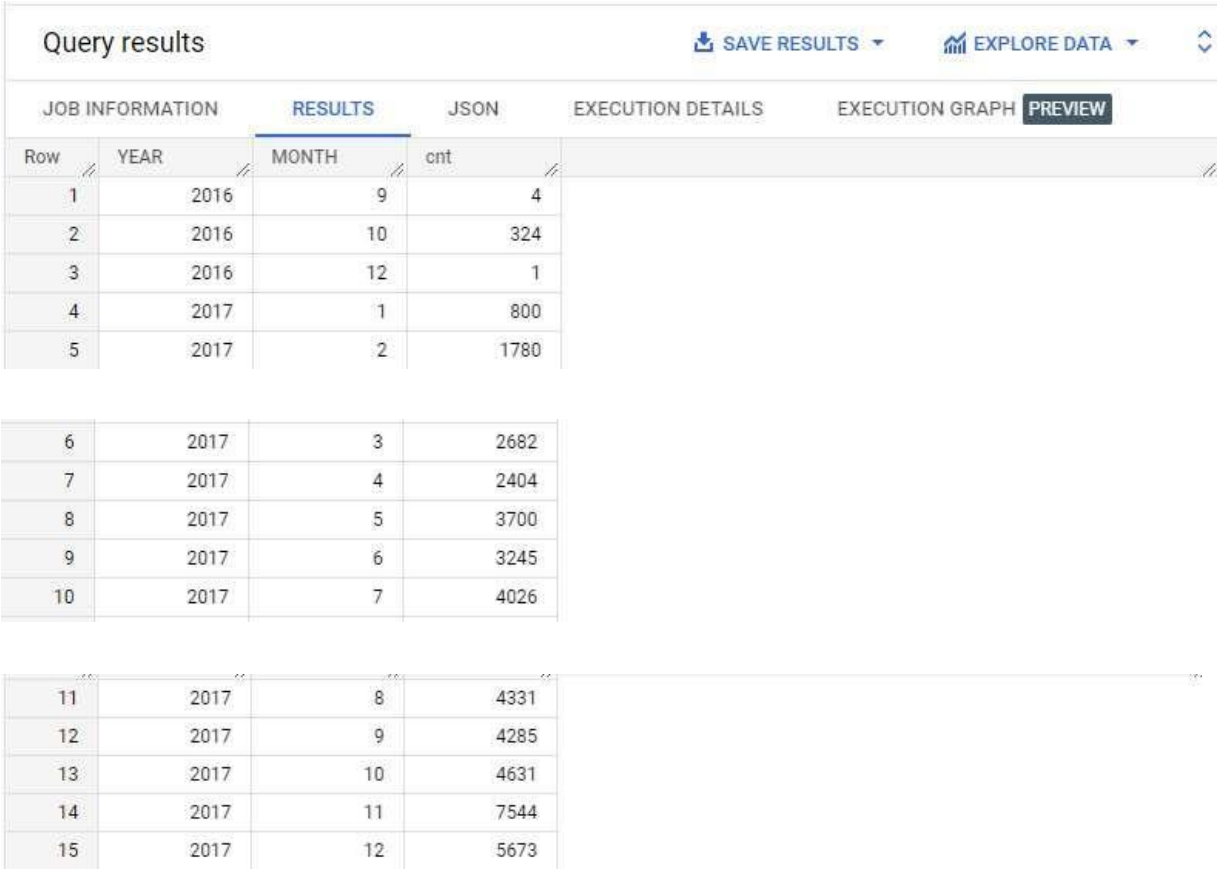
Q. 2.In-depth Exploration:

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

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?

```
SELECT YEAR,MONTH ,cnt

FROM

(SELECT

        EXTRACT(year FROM order_purchase_timestamp) AS YEAR,
```

```
        Extract(month From order_purchase_timestamp) AS MONTH,
        COUNT(order_id) AS  cnt
    FROM `scaler-ds-ml-de-372817.TARGET.orders`
    GROUP BY EXTRACT(year FROM order_purchase_timestamp),
        EXTRACT(month From order_purchase_timestamp)
    ) T
ORDER BY YEAR,MONTH ;
```



Query results

SAVE RESULTS | EXPLORE DATA

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

Row	YEAR	MONTH	cnt
1	2016	9	4
2	2016	10	324
3	2016	12	1
4	2017	1	800
5	2017	2	1780
6	2017	3	2682
7	2017	4	2404
8	2017	5	3700
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

Row	YEAR	MONTH	cnt
16	2018	1	7269
17	2018	2	6728
18	2018	3	7211
19	2018	4	6939
20	2018	5	6873
21	2018	6	6167
22	2018	7	6292
23	2018	8	6512
24	2018	9	16
25	2018	10	4

Inference :

a) It can be seen clearly from the data that there is a growing trend in e commerce in Brazil with slight fluctuations or exceptions.
b) Almost all months show good orders but still “September” , “October” , “November” are the months where no exceptional increase in orders is seen.

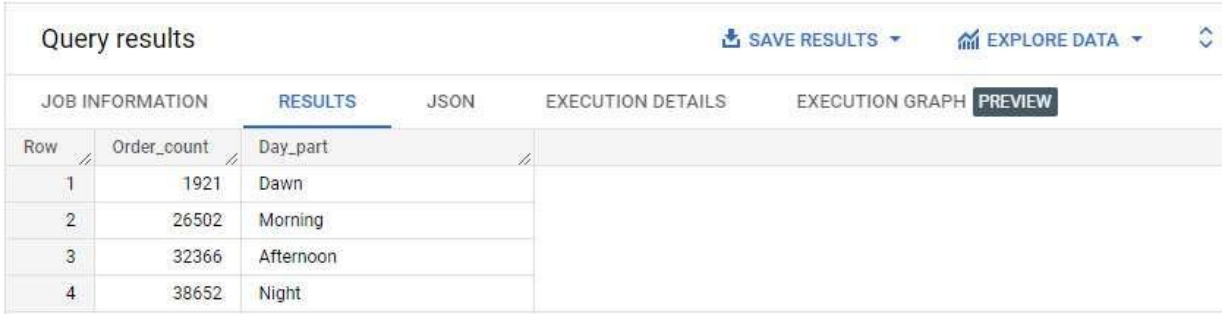
Recommendation:

- The trend of e-commerce is on an increase in the Brazil and hence the TARGET company must try to add new and other products which are otherwise excluded from the e -commerce industry. Company must come up with new offers, ideas and ways to satisfy the customers to keep this increasing e-commerce trend alive in Brazil for its bright economic future.

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

```
SELECT

  COUNT(DISTINCT order_id) as Order_count,

  Day_part

FROM

    (SELECT

            order_id,
```

```
    CASE

    WHEN  EXTRACT(hour FROM order_purchase_timestamp) >= 5 AND EXTRACT (hour
FROM order_purchase_timestamp) <= 7 THEN 'Dawn'

    WHEN  EXTRACT(hour FROM order_purchase_timestamp) >= 7 AND EXTRACT (hour
FROM order_purchase_timestamp) <= 12 THEN 'Morning'

    WHEN  EXTRACT(hour FROM order_purchase_timestamp) >= 12 AND EXTRACT (hour
FROM order_purchase_timestamp) <= 17 THEN 'Afternoon'

   ELSE  'Night'

   END as Day_Part

  FROM TARGET.orders) Y

GROUP BY Day_Part

ORDER BY COUNT(DISTINCT order_id);
```



Query results

SAVE RESULTS | EXPLORE DATA

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

Row	Order_count	Day_part
1	1921	Dawn
2	26502	Morning
3	32366	Afternoon
4	38652	Night

Inference:

- Hence from the above data it is clear that Brazilian customers tend to buy more at Night.

Recommendations :

- The buying of the Brazilian customers shows a Night trend and hence the TARGET company is advised to take care of the stocks and should ensure that stocks are not exhausted till evening and if get exhausted , the stocks need to be updated Also the closing time of

the stores can be extended. More staff needs to be available for the complete assistance of the huge night crowd.

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

1. Get month on month orders by states

```
SELECT

   year, month , COUNT(order_id)

FROM

      (SELECT

            EXTRACT (Month from o.order_purchase_timestamp) as month,

            EXTRACT (Year from o.order_purchase_timestamp) as year,

            Order_id

            FROM `TARGET.orders`  o

            JOIN `TARGET.customers` c

            ON o.customer_id = c.customer_id)
```

```
GROUP BY year,month

ORDER BY year,month;
```

Query results

SAVE RESULTS EXPLORE DATA

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

Row	year	month	f0_
1	2016	9	4
2	2016	10	324
3	2016	12	1
4	2017	1	800
5	2017	2	1780
6	2017	3	2682
7	2017	4	2404
8	2017	5	3700
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
18	2018	3	7211
19	2018	4	6939
20	2018	5	6873
21	2018	6	6167
22	2018	7	6292
23	2018	8	6512
24	2018	9	16

Inference :

- From the data above it is clear that , month on month sales show increasing trend for every year from 2016 to 2018.

Recommendations:

- As it is clear that the month on month sales are showing increasing trend over 2016 - 2018, it can be predicted that this trend may continue in the future with same pace and hence TARGET company is advised to be ready with maximum stock possible and avoid any shortage.

2. Distribution of customers across the states in Brazil

```
SELECT
   customer_state as State,
   COUNT(customer_id) as Total_customers
FROM TARGET.customers
GROUP BY customer_state
ORDER BY Total_customers DESC;
```

Row	State	Total_customers
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
10	GO	2020

Inference:

-From the above output we get the distribution of customers across all the states in Brazil . Also the top 5 states with the highest customers are

1) SP (41746) 2) RJ (12852) 3) MG (11635) 4) RS (5466) 5) PR (5045)

Recommendations:

- As the top 5 states giving more customers are identified, the company now needs to move its focus on the immediate states following th top 5 states and try to cater the needs of the customers from those states to increase the business.

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

```
WITH money_move_analysis as

(SELECT payment_value,

  EXTRACT(month FROM o.order_purchase_timestamp) as month,

  EXTRACT( year FROM o.order_purchase_timestamp) as year

 FROM `TARGET.payments` as p

 JOIN `TARGET.orders` as o
```

```
 ON p.order_id = o.order_id),
monthly_purchase as
(SELECT
       Month,
       Year,
       COUNT(payment_value) as nu_of_orders,
       AVG(payment_value ) as cost FROM money_move_analysis
  GROUP BY year, month
  HAVING month <= 8 and year BETWEEN 2017 and 2018)
SELECT
       a.month ,
       a.year,
       b.year,
       a.cost,
       b.cost,
       a.nu_of_orders,
       b.nu_of_orders,
       ((b.cost - a.cost) / a.cost) * 100 as  Percentage_change
  FROM monthly_purchase a
  JOIN monthly_purchase b
  ON a.month = b.month and a.year <> b.year
  ORDER BY a.year,a.month
  limit 8;
```

Row	month	year	year_1	cost	cost_1	nu_of_orders	nu_of_orders_1	Percentage_change
1	1	2017	2018	162.9271058823...	147.42882189...	850	7563	-9.512403661961...
2	2	2017	2018	154.7762513255...	142.75939873...	1886	6952	-7.764015789543...
3	3	2017	2018	158.57017976736	154.37328541...	2837	7512	-2.646710979016...
4	4	2017	2018	162.5002061454...	161.01893189...	2571	7209	-0.911552231170...
5	5	2017	2018	150.3343864097...	161.73540995...	3944	7135	7.583776282650...
6	6	2017	2018	148.7998777648...	159.50778937...	3436	6419	7.196183069028...
7	7	2017	2018	137.2209682649...	163.90667742...	4317	6507	19.44725321262...
8	8	2017	2018	148.2189714285	152.64636010	4550	6698	2.987059373203

Inference:

- From the above result it is clearly seen that, the percentage change in cost of orders has been negative at start but gradually then has shown an increasing trend. Also , the number of orders month on month from 2017 to 2018 have shown a steady increment.

Recommendations:

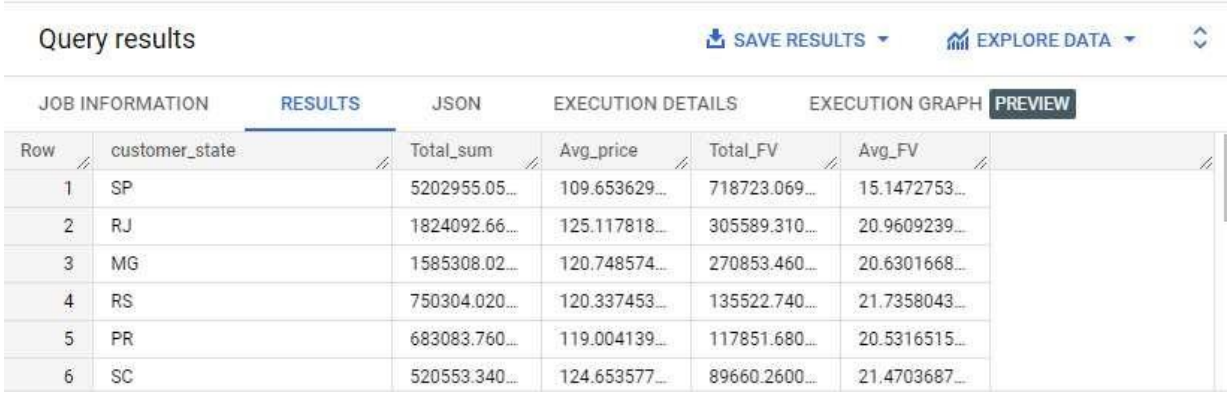
- The month on month orders vol. has shown increment from 2017-2018 and this shows that the TARGET company needs to take care of the stock updation and increment.

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

```
SELECT

  customer_state,

  SUM(price) as Total_sum,

  AVG(price) as Avg_price,

  SUM(freight_value) as Total_FV,

  AVG(freight_value) as Avg_freight_value
```

```
FROM `TARGET.order_items` as oi

JOIN `TARGET.orders` as o

ON oi.order_id = o.order_id

JOIN `TARGET.customers`as c

ON o.customer_id = c.customer_id

GROUP BY customer_state
```

Top 5 according to Total_sum



Query results

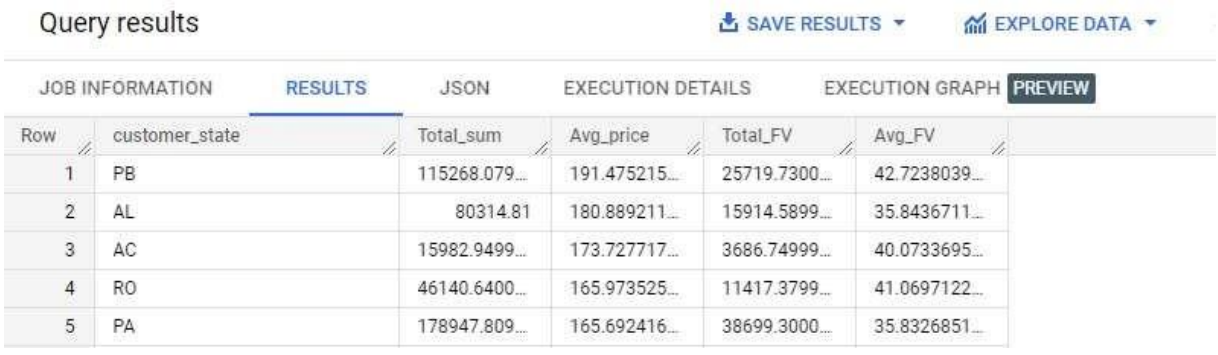
SAVE RESULTS | EXPLORE DATA

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

Row	customer_state	Total_sum	Avg_price	Total_FV	Avg_FV
1	SP	5202955.05...	109.653629...	718723.069...	15.1472753...
2	RJ	1824092.66...	125.117818...	305589.310...	20.9609239...
3	MG	1585308.02...	120.748574...	270853.460...	20.6301668...
4	RS	750304.020...	120.337453...	135522.740...	21.7358043...
5	PR	683083.760...	119.004139...	117851.680...	20.5316515...
6	SC	520553.340...	124.653577...	89660.2600...	21.4703687...

```
ORDER BY Total_sum DESC
```

Top 5 according to avg_sum



Query results

SAVE RESULTS | EXPLORE DATA

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

Row	customer_state	Total_sum	Avg_price	Total_FV	Avg_FV
1	PB	115268.079...	191.475215...	25719.7300...	42.7238039...
2	AL	80314.81	180.889211...	15914.5899...	35.8436711...
3	AC	15982.9499...	173.727717...	3686.74999...	40.0733695...
4	RO	46140.6400...	165.973525...	11417.3799...	41.0697122...
5	PA	178947.809...	165.692416...	38699.3000...	35.8326851...

```
ORDER BY Avg_sum DESC
```

Top 5 according to Total_FV

Query results

SAVE RESULTS EXPLORE DATA

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

Row	customer_state	Total_sum	Avg_price	Total_FV	Avg_FV
1	SP	5202955.05...	109.653629...	718723.069...	15.1472753...
2	RJ	1824092.66...	125.117818...	305589.310...	20.9609239...
3	MG	1585308.02...	120.748574...	270853.460...	20.6301668...
4	RS	750304.020...	120.337453...	135522.740...	21.7358043...
5	PR	683083.760...	119.004139...	117851.680...	20.5316515...
6	BA	511349.990...	134.601208...	100156.679...	26.3639589...

```
ORDER BY Total_FV DESC
```

Top 5 according to Avg_FV

Query results

SAVE RESULTS EXPLORE DATA

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

Row	customer_state	Total_sum	Avg_price	Total_FV	Avg_FV
1	RR	7829.42999...	150.565961...	2235.19000...	42.9844230...
2	PB	115268.079...	191.475215...	25719.7300...	42.7238039...
3	RO	46140.6400...	165.973525...	11417.3800...	41.0697122...
4	AC	15982.9499...	173.727717...	3686.75000...	40.0733695...
5	PI	86914.0799...	160.358081...	21218.2	39.1479704...
6	MA	119648.219...	145.204150...	31523.7700...	38.2570024...

```
ORDER BY Avg_FV DESC ;
```

- Same Query with the use of CTE

```
WITH CTE1 as

(SELECT

  customer_state,

  SUM(price) as Total_sum,

  AVG(price) as Avg_price,
```

```
  SUM(freight_value) as Total_FV,
  AVG(freight_value) as Avg_FV
FROM `TARGET.order_items` as oi
JOIN `TARGET.orders` as o
ON oi.order_id = o.order_id
JOIN `TARGET.customers`as c
ON o.customer_id = c.customer_id
GROUP BY customer_state)

SELECT
  customer_state,
  Total_sum,
  Avg_price,
  Total_FV,
  Avg_FV
FROM CTE1
ORDER BY customer_state;
```

Inference :

- From the above state wise data of

  SUM and AVERAGE of the Price and Freight_value ,

  we get information of the top 5 states ordered according to

  SUM(price) , AVG(price), SUM(freight_value) and AVG(freight_value).

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

1. Calculate days between purchasing, delivering and estimated delivery

```
WITH CTE1 AS
(
  SELECT
  EXTRACT(day FROM order_purchase_timestamp) as day,
  order_id,
  order_purchase_timestamp,
  order_estimated_delivery_date,
  order_delivered_customer_date,
  DATE_DIFF(order_estimated_delivery_date,order_purchase_timestamp,day) as
App_Estimated,
  DATE_DIFF(order_delivered_customer_date,order_purchase_timestamp,day) as
App_deliver,
  DATE_DIFF(order_delivered_customer_date,order_estimated_delivery_date,day) as
Est_deliver

FROM `TARGET.orders`
)
SELECT
MAX(App_Estimated) as Est_time,
MAX(App_deliver) as Del_time,
MAX(Est_deliver) as Est_time
FROM CTE1;
```

Inference:

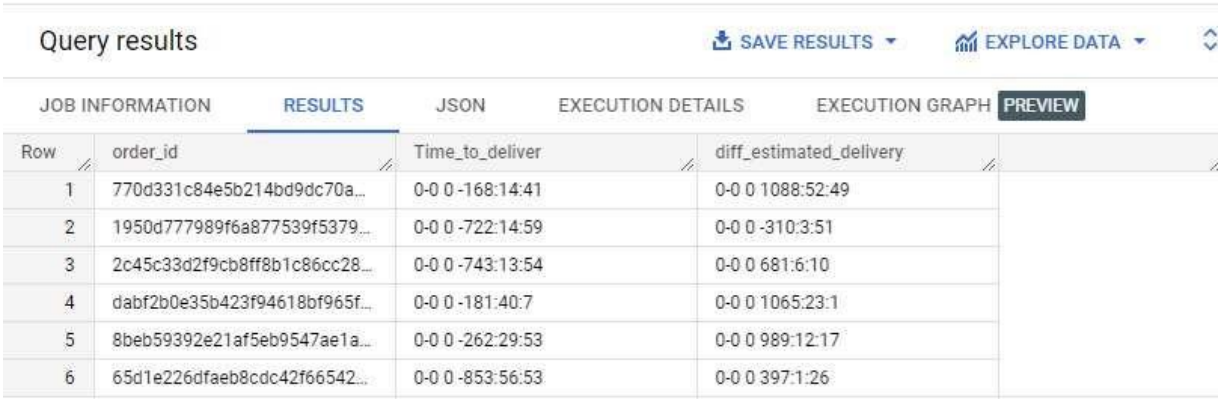
- From the data obtained we get to know that

a) The maximum time from the order purchased to estimated delivery is 155 days
b) The maximum time from order purchase to order delivered is 209 days
c) The maximum time from order estimated and delivered is 188 days

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

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

```
SELECT

  order_id,

  (order_purchase_timestamp      - order_delivered_customer_date)    AS
Time_to_deliver,

  (order_estimated_delivery_date - order_delivered_customer_date)    AS
diff_estimated_delivery

FROM `TARGET.orders`

WHERE order_purchase_timestamp IS NOT NULL AND order_delivered_customer_date IS NOT
NULL AND order_estimated_delivery_date IS NOT NULL;
```



Query results

SAVE RESULTS | EXPLORE DATA

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

Row	order_id	Time_to_deliver	diff_estimated_delivery
1	770d331c84e5b214bd9dc70a...	0-0 0 -168:14:41	0-0 0 1088:52:49
2	1950d777989f6a877539f5379...	0-0 0 -722:14:59	0-0 0 -310:3:51
3	2c45c33d2f9cb8ff8b1c86cc28...	0-0 0 -743:13:54	0-0 0 681:6:10
4	dabf2b0e35b423f94618bf965f...	0-0 0 -181:40:7	0-0 0 1065:23:1
5	8beb59392e21af5eb9547ae1a...	0-0 0 -262:29:53	0-0 0 989:12:17
6	65d1e226dfaeb8cdc42f66542...	0-0 0 -853:56:53	0-0 0 397:1:26

7	c158e9806f85a33877bdfd4f60...	0-0 0 -565:3:54	0-0 0 228:49:34
8	b60b53ad0bb7dacacf2989fe2...	0-0 0 -311:9:0	0-0 0 -133:12:27
9	c830f223aae08493ebecb52f2...	0-0 0 -309:37:20	0-0 0 298:32:10
10	a8aa2cd070eeac7e4368cae3d...	0-0 0 -173:39:35	0-0 0 24:37:40
11	813c55ce9b6baa8f879e064fbf...	0-0 0 -295:52:57	0-0 0 231:5:24
12	44558a1547e448b41c48c4087...	0-0 0 -44:13:3	0-0 0 126:59:55

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

```
SELECT
  c.customer_state,
  AVG(freight_value) AS Avg_FV,
  AVG(order_purchase_timestamp      - order_delivered_customer_date)    AS
Time_to_deliver,
  AVG(order_estimated_delivery_date - order_delivered_customer_date)    AS
Diff_estimated_delivery
FROM `TARGET.customers` as c
JOIN `TARGET.orders` as o
ON  c.customer_id = o.customer_id
JOIN `TARGET.order_items` as oi
ON oi.order_id = o.order_id
GROUP BY c.customer_state
ORDER BY Avg_FV desc;
```

Query results

SAVE RESULTS | EXPLORE DATA

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

Row	customer_state	Avg_FV	Time_to_deliver	Diff_estimated_delivery
1	RR	42.9844230...	0-0 0 -677:32:39.391304347	0-0 0 422:50:27.608695652
2	PB	42.7238039...	0-0 0 -494:8:22.412969283	0-0 0 296:55:4.755972696
3	RO	41.0697122...	0-0 0 -473:44:41.212454212	0-0 0 464:11:10.890109890
4	AC	40.0733695...	0-0 0 -497:10:23.516483516	0-0 0 487:59:17.450549450
5	PI	39.1479704...	0-0 0 -465:13:58.927342256	0-0 0 260:27:6.619502868
6	MA	38.2570024...	0-0 0 -519:34:4.800	0-0 0 221:24:4.645
7	TO	37.2466031...	0-0 0 -418:45:40.490322580	0-0 0 279:36:27.777419354
8	SE	36.6531688...	0-0 0 -515:12:59.317333333	0-0 0 223:49:3.408
9	AL	35.8436711...	0-0 0 -587:44:21.852459016	0-0 0 193:22:34.871194379
10	PA	35.8326851...	0-0 0 -570:5:50.211574952	0-0 0 325:39:52.679316888

Inference:

The above retrieved data gives us the information of the Top 5 states according to the AVG_Freight_Value along with their Time_to_deliver and Diff_estimated_delivery.

4. Sort the data to get the following:

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

```
(SELECT

  c.customer_state,

  AVG(freight_value) AS AVG_FV

FROM `TARGET.order_items` AS oi

JOIN `TARGET.orders` AS o

ON oi.order_id = o.order_id

JOIN `TARGET.customers` AS c

ON o.customer_id = c.customer_id

GROUP BY c.customer_state

ORDER BY AVG_FV

LIMIT 5)

UNION ALL

(SELECT

  c.customer_state,

  AVG(freight_value) AS AVG_FV

FROM `TARGET.order_items` AS oi

JOIN `TARGET.orders` AS o

ON oi.order_id = o.order_id

JOIN `TARGET.customers` AS c
```

```
ON o.customer_id = c.customer_id

GROUP BY c.customer_state

ORDER BY AVG_FV DESC

LIMIT 5)

ORDER BY AVG_FV DESC
```

Query results

SAVE RESULTS | EXPLORE DATA

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

Row	customer_state	AVG_FV
1	RR	42.9844230...
2	PB	42.7238039...
3	RO	41.0697122...
4	AC	40.0733695...
5	PI	39.1479704...
6	DF	21.0413549...
7	RJ	20.9609239...
8	MG	20.6301668...
9	PR	20.5316515...
10	SP	15.1472753...

Inference :

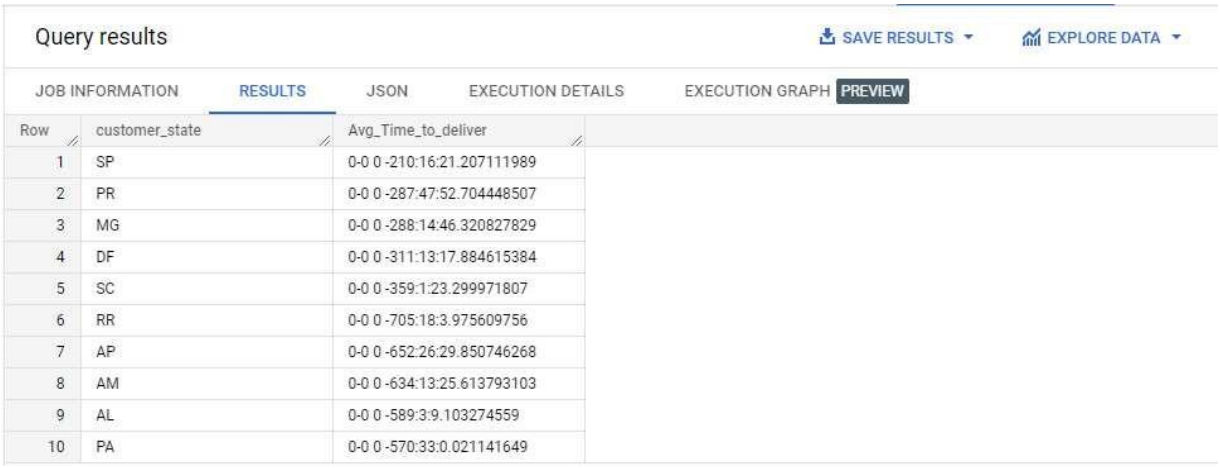
Thus the above table gives us the Top 5 states with highest AVG_Freight_value(row no. 1 - 5) and Top 5 states with the lowest AVG_Freight_value(row no. 6 - 10)

Recommendations:

The TARGET company needs to focus more on the top 5 states which have low freight_value and should try to increase their business in those states and try to capture that void market.

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

```
(SELECT

     c.customer_state,

    AVG (order_purchase_timestamp - order_delivered_customer_date)    AS
Avg_Time_to_deliver

FROM `TARGET.orders` AS o

JOIN `TARGET.customers` AS c

ON o.customer_id = c.customer_id

GROUP BY c.customer_state

ORDER BY Avg_Time_to_deliver

LIMIT 5)

UNION ALL

(SELECT

     c.customer_state,

    AVG (order_purchase_timestamp - order_delivered_customer_date)    AS
Avg_Time_to_deliver

FROM `TARGET.orders` AS o

JOIN `TARGET.customers` AS c

ON o.customer_id = c.customer_id

GROUP BY c.customer_state

ORDER BY Avg_Time_to_deliver DESC

LIMIT 5)
```



Row	customer_state	Avg_Time_to_deliver
1	SP	0-0 0 -210:16:21.207111989
2	PR	0-0 0 -287:47:52.704448507
3	MG	0-0 0 -288:14:46.320827829
4	DF	0-0 0 -311:13:17.884615384
5	SC	0-0 0 -359:1:23.299971807
6	RR	0-0 0 -705:18:3.975609756
7	AP	0-0 0 -652:26:29.850746268
8	AM	0-0 0 -634:13:25.613793103
9	AL	0-0 0 -589:3:9.103274559
10	PA	0-0 0 -570:33:0.021141649

INFERENCE :

> Thus the above table gives us the Top 5 states with highest AVG_Time_to_deliver (row no. 1 - 5) and Top 5 states with the lowest AVG_Time_to_deliver (row no. 6 - 10).

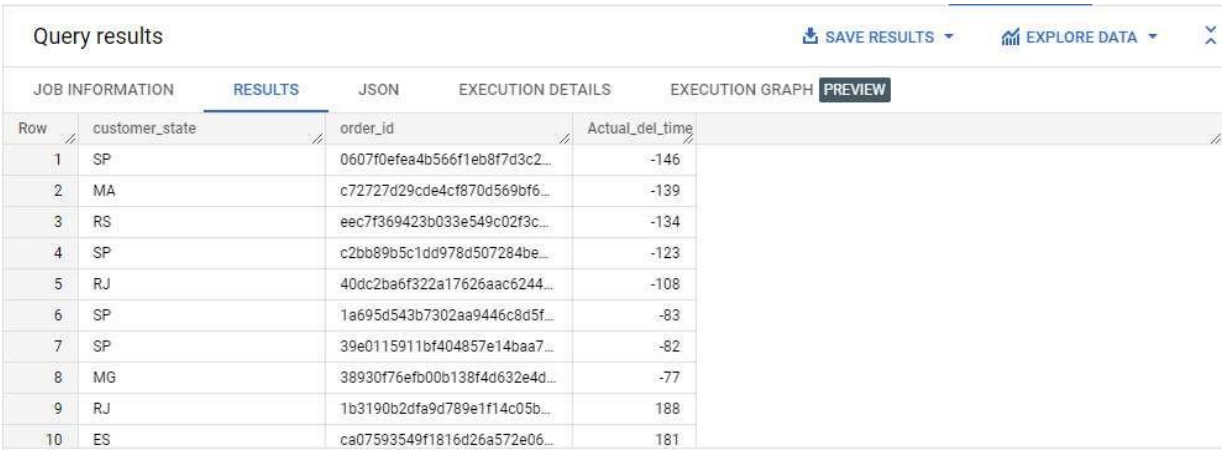
Recommendations:

The TARGET company is advised to improve itself on reducing the avg time to deliver , which may possibly lead to customer satisfaction and increase in the business can be seen ultimately.

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

```
(SELECT

  c.customer_state,

  o.order_id,

  DATE_DIFF(order_delivered_customer_date,order_estimated_delivery_date,day) AS
Actual_del_time

FROM `TARGET.orders` as o
```

```
JOIN `TARGET.customers` as c

ON o.customer_id = c.customer_id

WHERE order_delivered_customer_date IS NOT NULL AND order_estimated_delivery_date IS
NOT NULL

ORDER  BY Actual_del_time DESC

LIMIT 5)



UNION ALL



(SELECT

  c.customer_state,

  o.order_id,

  DATE_DIFF(order_delivered_customer_date,order_estimated_delivery_date,day) AS
Actual_del_time

FROM `TARGET.orders` as o

JOIN `TARGET.customers` as c

ON o.customer_id = c.customer_id

WHERE order_delivered_customer_date IS NOT NULL AND order_estimated_delivery_date IS
NOT NULL

ORDER  BY Actual_del_time

LIMIT 8);
```



Row	customer_state	order_id	Actual_del_time
1	SP	0607f0efea4b566f1eb8f7d3c2...	-146
2	MA	c72727d29cde4cf870d569bf6...	-139
3	RS	eec7f369423b033e549c02f3c...	-134
4	SP	c2bb89b5c1dd978d507284be...	-123
5	RJ	40dc2ba6f322a17626aac6244...	-108
6	SP	1a695d543b7302aa9446c8d5f...	-83
7	SP	39e0115911bf404857e14baa7...	-82
8	MG	38930f76efb00b138f4d632e4d...	-77
9	RJ	1b3190b2dfa9d789e1f14c05b...	188
10	ES	ca07593549f1816d26a572e06...	181

Q.6) Payment type analysis:

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

```
(SELECT

    EXTRACT(year FROM o.order_purchase_timestamp) as Year,

    EXTRACT(month FROM o.order_purchase_timestamp) as Month,

    COUNT(o.order_id) as Total_orders,

    p.payment_type

FROM `TARGET.orders`AS o

JOIN `TARGET.payments`AS p

ON o.order_id = p.order_id

GROUP BY year,month,p.payment_type

ORDER BY year,month)
```

Query results

SAVE RESULTS | EXPLORE DATA

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

Row	Year	Month	Total_orders	payment_type
1	2016	9	3	credit_card
2	2016	10	254	credit_card
3	2016	10	63	UPI
4	2016	10	23	voucher
5	2016	10	2	debit_card
6	2016	12	1	credit_card
7	2017	1	583	credit_card
8	2017	1	197	UPI
9	2017	1	61	voucher
10	2017	1	9	debit_card

Inference:

- Data of month over month count of orders for different payments type is displayed above for the first 10 entries the total orders per payment type are as below
    1. Credit card = 841
    2. UPI = 260
    3. Debit card = 11
    4. Voucher = 84

    Thus the order of payments type is credit card > upi > voucher > debit card.

Recommendations :

- Hence from the above inference it is advised that the Target company should focus on giving more incentives or offers related to the payments done through the credit cards , because the trend shows that people prefer more and more usage of the Credit card option for their payments.

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

```
SELECT
  p.payment_installments,
  COUNT(o.order_id) as Total_orders
FROM `TARGET.payments` as p
JOIN `TARGET.orders` as o
ON p.order_id = o.order_id
GROUP BY p.payment_installments
ORDER BY p.payment_installments
```

Query results

SAVE RESULTS | EXPLORE DATA

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

Row	payment_installments	Total_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
10	9	644

Row	payment_installments	Total_orders
11	10	5328
12	11	23
13	12	133
14	13	16
15	14	15
16	15	74
17	16	5
18	17	8
19	18	27
20	20	17

20	20	17
21	21	3
22	22	1
23	23	1
24	24	18

Inference:

- Thus we have obtained above the count of orders according to the no. payment installments. The decreasing order to of the orders count for the number of payments installments is

  1(52546) > 2(12413) > 3(10461) > 4(7098) > 10(5328) > 5(5239)

Recommendations:

- Seeing the trend of Total orders according to the payment_installments , the TARGET company is advised to offer special benefits to the customers who male payments in 1 installments since their number is the largest also it shd strive hard to give benefits to payments installments category 2 and 3 also to atleast continue the trend which is not very poor.