TARGET SQL QUERY-RESULT

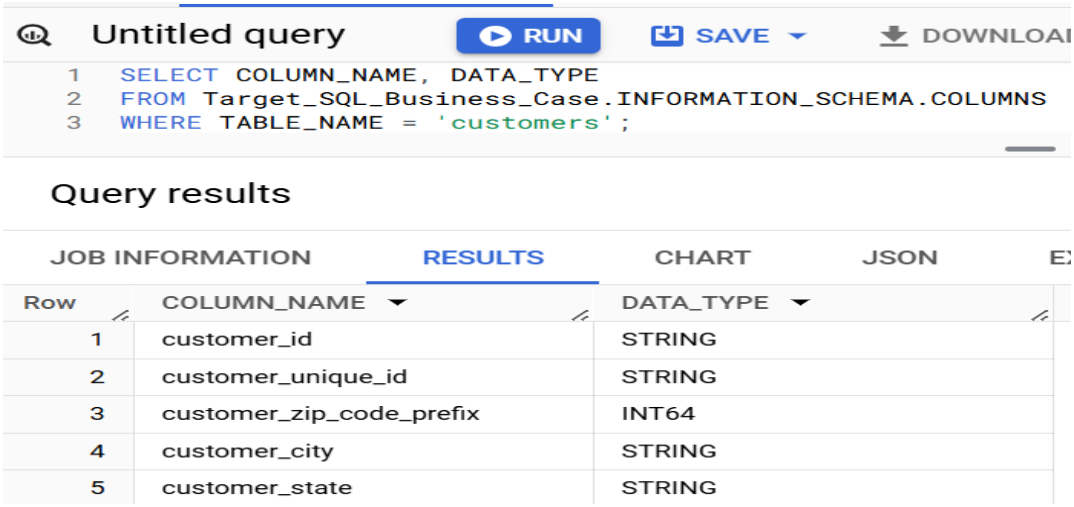
Aniket Chavan

9420395075

aniketc7303@gmail.com

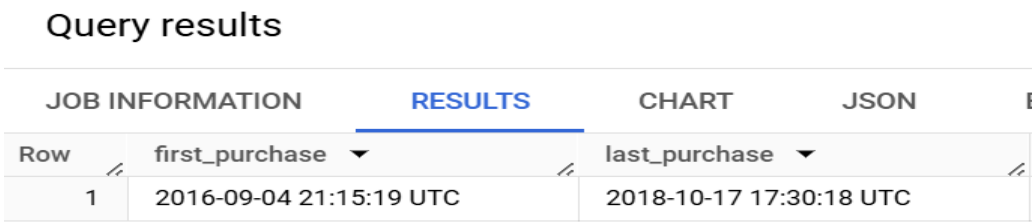
1. **Import the dataset and do usual exploratory analysis steps like checking the structure & characteristics of the dataset:**

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



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

```
SELECT
MIN(order_purchase_timestamp) AS first_purchase,
MAX(order_purchase_timestamp) AS last_purchase,
FROM `Target_SQL_Business_Case.orders`
```



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

```
select
COUNT(distinct customer_city) as city_count,
COUNT(distinct customer_state) as state_count
from Target_SQL_Business_Case.customers as c inner join `Target_SQL_Business_Case.orders` as
o
on c.customer_id = o.customer_id;
```

Query results

JOB INFORMATION | RESULTS | CHART

Row	city_count	state_count
1	4119	27

II. In-depth Exploration:

1. Is there a growing trend in the no. of orders placed over the past years?
   Query is same as 2nd question. As we can see growing trend from same table.

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

```
SELECT
  count(*) as order_count,
  extract(year from order_purchase_timestamp) as year,
  extract(month from order_purchase_timestamp) as month
from `Target_SQL_Business_Case.orders`
group by month, year
order by month, year;
```

Query results

JOB INFORMATION | RESULTS | CHART | JSON

Row	order_count	year	month
1	800	2017	1
2	7269	2018	1
3	1780	2017	2
4	6728	2018	2
5	2682	2017	3
6	7211	2018	3
7	2404	2017	4
8	6939	2018	4
9	3700	2017	5
10	6873	2018	5

Observation -

a. As here the query output is ordered by months and years we can easily find our answer. Yes there is a ***growing trend*** in the no. of orders per month every year. ***Also there is growth*** in no. of orders in a month ***as compare to the previous year***.
b. In terms of ***seasonality*** we can see here the ***growth in no. of orders*** in particular months of seasons per year.

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

```
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 'Morning'
    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 day_time,
  count(distinct order_id) as order_count
from `Target_SQL_Business_Case.orders`
group by day_time;
```

Query results

JOB INFORMATION | RESULTS | CHART | JSON

Row	day_time	order_count
1	Morning	27733
2	Dawn	5242
3	Afternoon	38135
4	Night	28331

Observation -

As we can see here, Brazilian customers mostly place their orders in the **Afternoon** which is **38135** nos. **Secondarily** their orders are get placed during **night** hours which is 28331 nos. **Consecutively** they use **morning** time to order with nos. of **27733**. After all **least** no. of orders get placed in **Dawn** time of the day with **5242** orders.

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

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

```
SELECT
  c.customer_state,
  count(o.order_id) as order_count,
  extract(year from o.order_purchase_timestamp) as order_year,
  extract(month from o.order_purchase_timestamp) as order_month
from `Target_SQL_Business_Case.customers` as c
  join `Target_SQL_Business_Case.orders` as o
  on c.customer_id = o.customer_id
GROUP BY c.customer_state, order_year, order_month
order by order_year, order_month, c.customer_state;
```

Query results

SAVE RESULT

JOB INFORMATION | RESULTS | CHART | JSON | EXECUTION DETAILS

Row	customer_state	order_count	order_year	order_month
1	RR	1	2016	9
2	RS	1	2016	9
3	SP	2	2016	9
4	AL	2	2016	10
5	BA	4	2016	10
6	CE	8	2016	10
7	DF	6	2016	10
8	ES	4	2016	10
9	GO	9	2016	10
10	MA	4	2016	10

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

```
select
  distinct customer_state,
  count(customer_id) as cust_count
from `Target_SQL_Business_Case.customers`
group by customer_state
```

Query results

JOB INFORMATION | RESULTS | CHART | JSON

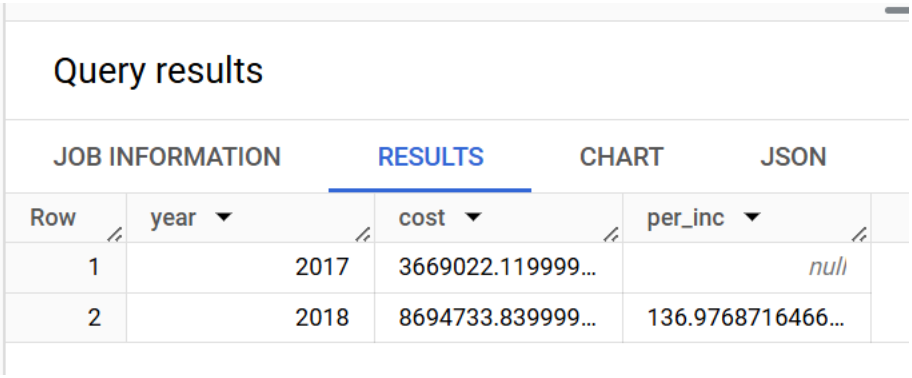
Row	customer_state	cust_count
1	RN	485
2	CE	1336
3	RS	5466
4	SC	3637
5	SP	41746
6	MG	11635
7	BA	3380
8	RJ	12852
9	GO	2020
10	MA	747

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

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

```
WITH CTE1 AS(
  SELECT
    *
  from `Target_SQL_Business_Case.orders` as o
    join `Target_SQL_Business_Case.payments` as p
    on o.order_id = p.order_id
  where extract(year from o.order_purchase_timestamp) between 2017 and 2018
    and extract(month from o.order_purchase_timestamp) between 1 and 8

),
CTE2 AS(
  SELECT
    extract(year from order_purchase_timestamp) as year,
    sum(payment_value) as cost
  from CTE1
  group by year
)
SELECT
  year, cost,
  (cost - lag(cost, 1) over(order by CTE2.year))*100/ lag(cost, 1) over(order by
CTE2.year) as per_inc
  from CTE2
```



Query results

JOB INFORMATION | RESULTS | CHART | JSON

Row	year	cost	per_inc
1	2017	3669022.119999...	null
2	2018	8694733.839999...	136.9768716466...

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

```
SELECT
  distinct c.customer_state,
  COUNT(distinct i.order_id) as unique_id,
  ROUND(SUM(i.price),3) as Total_amt,
  ROUND(SUM(i.price)/COUNT(distinct i.order_id),3) as avg_amt
FROM `Target_SQL_Business_Case.orders` as o
join `Target_SQL_Business_Case.order_items` as i
on o.order_id = i.order_id
join `Target_SQL_Business_Case.customers` as c
on o.customer_id = c.customer_id
GROUP BY c.customer_state
```

Query results

JOB INFORMATION | RESULTS | CHART | JSON | EXECUTION DETAI

Row	customer_state	unique_id	Total_amt	avg_amt
1	MT	903	156453.53	173.26
2	MA	740	119648.22	161.687
3	AL	411	80314.81	195.413
4	SP	41375	5202955.05	125.751
5	MG	11544	1585308.03	137.327
6	PE	1648	262788.03	159.459
7	RJ	12762	1824092.67	142.932
8	DF	2125	302603.94	142.402
9	RS	5432	750304.02	138.127
10	SE	345	58920.85	170.785

Observation -

Here we can see the total and average order prices for each state with state wise unique order ids. States with names SP, RJ and MG have highest order prices with highest order ids. So the average price is lesser as compare to other states having less total prices. State with name RR has 7829.43 as total price which is lowest of all the states.

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

```
SELECT
  distinct c.customer_state,
  COUNT(distinct i.order_id) as unique_id,
  ROUND(SUM(i.freight_value),3) as Total_amt,
  ROUND(SUM(i.freight_value)/COUNT(distinct i.order_id),3) as avg_amt
FROM `Target_SQL_Business_Case.orders` as o
join `Target_SQL_Business_Case.order_items` as i
on o.order_id = i.order_id
join `Target_SQL_Business_Case.customers` as c
on o.customer_id = c.customer_id
GROUP BY c.customer_state
```

Query results

JOB INFORMATION | RESULTS | CHART | JSON | EXECUTION DE

Row	customer_state	unique_id	Total_amt	avg_amt
1	MT	903	29715.43	32.907
2	MA	740	31523.77	42.6
3	AL	411	15914.59	38.722
4	SP	41375	718723.07	17.371
5	MG	11544	270853.46	23.463
6	PE	1648	59449.66	36.074
7	RJ	12762	305589.31	23.945
8	DF	2125	50625.5	23.824
9	RS	5432	135522.74	24.949
10	SE	345	14111.47	40.903

Observation -
We can say by watching this table, lesser the order no. needs to pay higher average fright charges for the order. On the contrary, as order nos. get increased it lowers the fright charges which customers needs to pay.

V. Analysis based on sales, freight and delivery time.

1. 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.

```
SELECT
  x.order_id,
  x.purchase_date,
  x.esti_deli_date,
  x.delivery_date,
  DATE_DIFF(x.delivery_date, x.purchase_date, DAY) as time_to_deliver,
  DATE_DIFF(x.esti_deli_date, x.delivery_date, DAY) as diff_estimated_delivery
from
(
  SELECT
    order_id,
    DATE(order_purchase_timestamp) as purchase_date,
    DATE(order_estimated_delivery_date) as esti_deli_date,
    DATE(order_delivered_customer_date) as delivery_date
  FROM `Target_SQL_Business_Case.orders`
) x
```

Query results

SAVE RESULTS

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

Row	order_id	purchase_date	esti_deli_date	delivery_date	time_to_deliver	diff_estimated_delive
1	770d331c84e5b21...	2016-10-07	2016-11-29	2016-10-14	7	46
2	1950d777989f6a8...	2018-02-19	2018-03-09	2018-03-21	30	-12
3	2c45c33d2f9cb8ff...	2016-10-09	2016-12-08	2016-11-09	31	29
4	dabf2b0e35b423f9...	2016-10-09	2016-11-30	2016-10-16	7	45
5	8beb59392e21af5...	2016-10-08	2016-11-30	2016-10-19	11	42
6	b60b53ad0bb7dac...	2017-05-10	2017-05-18	2017-05-23	13	-5
7	276e9ec344d3bf0...	2017-04-08	2017-05-18	2017-05-22	44	-4
8	1a0b31f08d0d7e8...	2017-04-11	2017-05-18	2017-04-18	7	30
9	cec8f5f7a13e5ab9...	2017-03-17	2017-05-18	2017-04-07	21	41
10	2d846c03073b1a4...	2017-05-10	2017-05-18	2017-05-25	15	-7

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

```
WITH T1 AS(
  SELECT
    distinct c.customer_state as state,
    ROUND(SUM(i.freight_value)/COUNT(distinct i.order_id),3) as avg_fright_amt
  FROM `Target_SQL_Business_Case.orders` as o
    join `Target_SQL_Business_Case.order_items` as i
    on o.order_id = i.order_id
    join `Target_SQL_Business_Case.customers` as c
    on o.customer_id = c.customer_id
  GROUP BY c.customer_state
  ORDER BY avg_fright_amt DESC
  LIMIT 5),
T2 AS(
  SELECT
    distinct c.customer_state as state,
    ROUND(SUM(i.freight_value)/COUNT(distinct i.order_id),3) as avg_fright_amt
  FROM `Target_SQL_Business_Case.orders` as o
    join `Target_SQL_Business_Case.order_items` as i
    on o.order_id = i.order_id
    join `Target_SQL_Business_Case.customers` as c
  on o.customer_id = c.customer_id
  GROUP BY c.customer_state
  ORDER BY avg_fright_amt ASC
  LIMIT 5)
(SELECT state, avg_fright_amt FROM T1) UNION ALL (SELECT state, avg_fright_amt FROM T2)
```

Query results

JOB INFORMATION | RESULTS | CHART | JSO

Row	state	avg_fright_amt
1	RR	48.591
2	PB	48.345
3	RO	46.224
4	AC	45.515
5	PI	43.039
6	SP	17.371
7	MG	23.463
8	PR	23.58
9	DF	23.824
10	RJ	23.945

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

```
With T1 as (
  Select
    c.customer_state,
    count(o.order_id) as No_of_Orders,
    sum(date_diff(o.order_delivered_customer_date, o.order_purchase_timestamp,
Day))/count(o.order_id) as Delivery_Time
  from `Target_SQL_Business_Case.orders` o
    inner join `Target_SQL_Business_Case.customers` c
    on o.customer_id = c.customer_id
    group by c.customer_state
    order by Delivery_Time Desc
    Limit 5),
T2 as (
  Select
    c.customer_state,
    count(o.order_id) as No_of_Orders,
    sum(date_diff(o.order_delivered_customer_date, o.order_purchase_timestamp,
Day))/count(o.order_id) as Delivery_Time
    from `Target_SQL_Business_Case.orders` o
    inner join `Target_SQL_Business_Case.customers` c
    on o.customer_id = c.customer_id
    group by c.customer_state
    order by 3 ASC
    Limit 5)
(Select Distinct customer_state as State, Delivery_Time as Average_Delivery_Time from T1)
union all
(Select Distinct customer_state as State, Delivery_Time as Average_Delivery_Time from T2)
```

Query results

JOB INFORMATION | RESULTS | CHART | JS

Row	State	Average_Delivery_Tir
1	SP	8.049393953911...
2	PR	11.24796828543...
3	MG	11.26600773528...
4	DF	12.15841121495...
5	SC	14.12125378058...
6	AP	26.33823529411...
7	RR	25.82608695652...
8	AM	25.45945945945...
9	AL	23.10895883777...
10	PA	22.62256410256...

4. Find out the top 5 states where the order delivery is really fast as compared to the estimated date of delivery.

```
Select State, sum(No_of_days) as No_of_days, count(order_id) as order_count,
  ROUND(sum(No_of_days)/count(order_id),2) as Fast_Delivery
from
  (Select State, Date_diff(Expected_Delivery_Date, Delivery_Date, Day) as No_of_days,
order_id
  from
    (Select c.customer_state as State,
      Extract(Date from o.order_delivered_customer_date) as Delivery_Date,
      Extract(Date from o.order_estimated_delivery_date) as Expected_Delivery_Date,
o.order_id
    from`Target_SQL_Business_Case.orders` o inner join `Target_SQL_Business_Case.customers` c
      on o.customer_id=c.customer_id
    where
      Extract(Date from o.order_delivered_customer_date) is not null
      and
      Extract(Date from o.order_delivered_customer_date) < Extract(Date from
o.order_estimated_delivery_date)))
    group by state
    order by 4
    limit 5
```

Query results

SAVE RESULTS

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

Row	State	No_of_days	order_count	Fast_Delivery
1	SP	463704	38108	12.17
2	ES	23358	1751	13.34
3	MS	8315	620	13.41
4	DF	25935	1933	13.42
5	SC	43239	3201	13.51

VI. Analysis based on the payments:

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

```
SELECT
  p.payment_type,
  EXTRACT(month from o.order_purchase_timestamp) as Month,
  EXTRACT(year from o.order_purchase_timestamp) as Year,
  COUNT(o.order_id) as Order_count
FROM `Target_SQL_Business_Case.orders` o
  inner join `Target_SQL_Business_Case.payments` p
  on o.order_id = p.order_id
GROUP BY Month, Year, p.payment_type
ORDER BY Month, Year
```

Query results

SAVE RESULT

JOB INFORMATION | RESULTS | CHART | JSON | EXECUTION DETAILS

Row	payment_type	Month	Year	Order_count
1	credit_card	1	2017	583
2	UPI	1	2017	197
3	voucher	1	2017	61
4	debit_card	1	2017	9
5	credit_card	1	2018	5520
6	UPI	1	2018	1518
7	voucher	1	2018	416
8	debit_card	1	2018	109
9	credit_card	2	2017	1356
10	UPI	2	2017	398

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

```
SELECT
  COUNT(order_id) as Order_count,
  payment_installments
FROM `Target_SQL_Business_Case.payments`
WHERE payment_installments > 1
GROUP BY payment_installments
```

Query results

JOB INFORMATION | RESULTS | CHART

Row	Order_count	payment_installment
1	12413	2
2	10461	3
3	7098	4
4	5239	5
5	3920	6
6	1626	7
7	4268	8
8	644	9
9	5328	10
10	23	11

Important Insights from the Data Set

1. The orders starts from 4th of September 2016 and ends on 17th October 2018.
2. There are total of 27 States and 8011 cities present in the data set.

3. November 2017 has the highest number of orders (7544).
4. Highest number of customers are from the State of SP (41746).

5. From 2017 to 2018, there is growth of 137 % in the cost of orders.

6. Year on Year Analysis shows maximum orders are placed in 2018 (54011), followed by 2017 (45101) and only 329 in 2016.

7. Month on Month Analysis also show increasing number of orders (positive trend) except in the months of September and October 2018.

8. There is significant increase in the number of orders placed from 2016 to 2017 and 2017 to 2018.

9. People of Brazil order significantly higher in the autumn and Winter Season as compared to other two seasons of the year.

10. Afternoon is the most preferred time period for Brazilians to order which is followed by Morning time.

11. Yes there is a growing trend in the number of orders placed over the past years. Implying that the company is growing and there is acceptance among the people of Brazil.

12. People of Brazil prefer Credit card payment mode as compared to other modes of payment. The data set shows credit card payment type is significantly higher than other three types of payments.

13. Installments of almost all the orders have been made successfully except in two cases. Means pendency's on account of due payments is almost nil where installment option is opted by the Brazilian customers.