BUSINESS CASE: TARGET SQL

SURBHI WAGDE

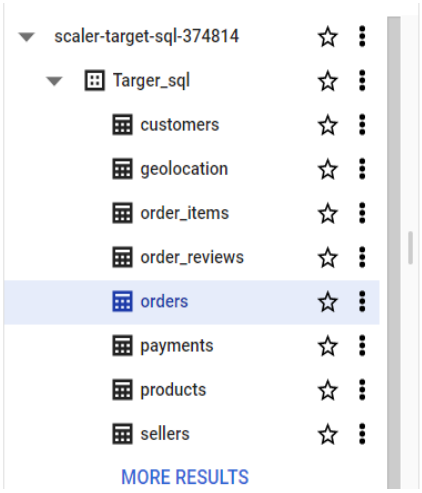
BATCH: DSML- JULY 2022

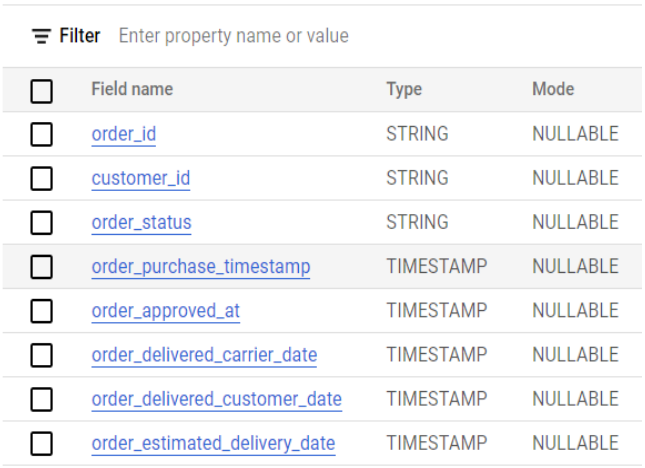
BEGINNER 1

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 – orders

Query:

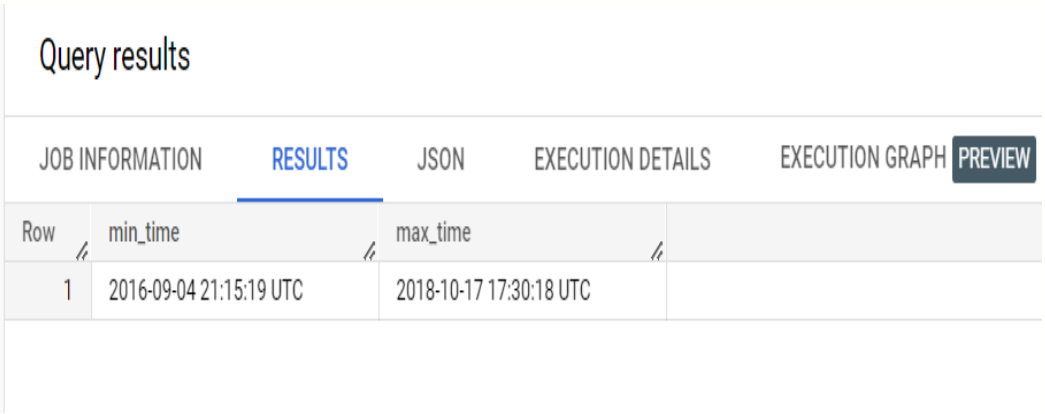




2.Time period for which the data is given

Query:

```
SELECT
    MIN(order_purchase_timestamp) AS min_time,
    MAX(order_purchase_timestamp) AS max_time
FROM `scaler-target-sql-374814.Targer_sql.orders`;
```



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

Query:

```
SELECT DISTINCT c.customer_state, c.customer_city

FROM `scaler-target-sql-374814.Targer_sql.orders` o
INNER JOIN `scaler-target-sql-374814.Targer_sql.customers` c
ON o.customer_id = c.customer_id;
```

Query results

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

Row	customer_state	customer_city
1	RN	acu
2	CE	ico
3	RS	ipe
4	CE	ipu
5	SC	ita
6	SP	itu
7	SP	jau
8	MG	luz
9	SP	poa
10	MG	uba
11	BA	una

1. 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 EXTRACT(month FROM order_purchase_timestamp) AS Month,
EXTRACT (year FROM order_purchase_timestamp) AS Year,
count(order_id) AS Total_orders
FROM `scaler-target-sql-374814.Targer_sql.orders`
GROUP BY Year,Month
ORDER BY Year,Month;
```

Query results

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

Row	Month	Year	Total_orders
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

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

Query:

```
SELECT count(order_id) as total_orders,
case
  when time(order_purchase_timestamp)between '05:00:01' and '06:00:00' then 'dawn'
  when time(order_purchase_timestamp)between '06:00:01' and '12:00:00' then 'morning'
  when time(order_purchase_timestamp)between '12:00:01' and '18:00:00' then 'afternoon'
  else 'night'
  end as time_of_day
FROM `scaler-target-sql-374814.Targer_sql.orders`
group by time_of_day
order by count(order_id);
```

Query results

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

Row	total_orders	time_of_day
1	188	dawn
2	22240	morning
3	38365	afternoon
4	38648	night

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

1. Get month on month orders by states

Query:

```
WITH case1 AS
(SELECT
  EXTRACT(year FROM order_purchase_timestamp) AS year,
  EXTRACT(month FROM order_purchase_timestamp) AS month,
  COUNT(o.order_id) AS total_orders,
  c.customer_state
 FROM `scaler-target-sql-374814.Targer_sql.orders` AS o
 INNER JOIN `scaler-target-sql-374814.Targer_sql.customers` AS c
 ON o.customer_id = c.customer_id
 GROUP BY 1,2,4),
case2 AS(
SELECT  year, month, total_orders, customer_state,
LAG(total_orders,1) OVER(PARTITION BY customer_state ORDER BY year,month) AS prev_month_order
s
FROM case1
ORDER BY year,month)
SELECT customer_state, year, month,total_orders,prev_month_orders,
      ROUND(((total_orders - prev_month_orders )/prev_month_orders)*100 ,2) AS month_on_month
FROM case2
ORDER BY customer_state,year, month;
```

Query results

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

Row	customer_state	year	month	total_orders	prev_month_orders	month_on_month
1	AC	2017	1	2	*null*	*null*
2	AC	2017	2	3	2	50.0
3	AC	2017	3	2	3	-33.33
4	AC	2017	4	5	2	150.0
5	AC	2017	5	8	5	60.0
6	AC	2017	6	4	8	-50.0
7	AC	2017	7	5	4	25.0
8	AC	2017	8	4	5	-20.0
9	AC	2017	9	5	4	25.0
10	AC	2017	10	6	5	20.0

2. Distribution of customers across the states in Brazil

Query:

```
SELECT
  customer_state,
  COUNT(DISTINCT customer_unique_id) AS total_customer
FROM `scaler-target-sql-374814.Targer_sql.customers`
GROUP BY customer_state;
```

Query results

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

Row	customer_state	total_customer
1	RN	474
2	CE	1313
3	RS	5277
4	SC	3534
5	SP	40302
6	MG	11259
7	BA	3277
8	RJ	12384
9	GO	1952
10	MA	726

4.Impact on Economy: Analyse 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

Query:

```
WITH case1 AS
(SELECT * FROM (
SELECT EXTRACT(year FROM order_purchase_timestamp) AS Year,
EXTRACT(month FROM order_purchase_timestamp) AS Month,
ROUND(SUM(payment_value),2) AS Total
FROM `scaler-target-sql-374814.Targer_sql.orders` o
INNER JOIN `scaler-target-sql-374814.Targer_sql.payments` p ON o.order_id=p.order_id
GROUP BY Year,Month ORDER BY Year,Month) a
WHERE a.Month IN (1,2,3,4,5,6,7,8) AND a.Year IN (2017,2018)),
case2 AS
(SELECT Year,ROUND(SUM(Total)) AS total FROM case1 GROUP BY Year ORDER BY Year)
SELECT *,ROUND((total - LAG(total) OVER(ORDER BY Year))/LAG(total) OVER(ORDER BY Year)*100,2)
 AS Percent_increase from case2;
```

Query results

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

Row	Year	total	Percent_increas
1	2017	3669022.0	*null*
2	2018	8694734.0	136.98

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

Query:

```
SELECT c.customer_state,

ROUND(AVG(oi.price),2) AS Average_price,
ROUND(SUM(oi.price),2) AS Total_price,
ROUND(AVG(oi.freight_value),2) AS Average_freight_value,
ROUND(SUM(oi.freight_value),2) AS Total_freight_value
FROM `scaler-target-sql-374814.Targer_sql.customers` c
INNER JOIN `scaler-target-sql-374814.Targer_sql.orders` o ON o.customer_id = c.customer_id
INNER JOIN `scaler-target-sql-374814.Targer_sql.order_items` oi ON o.order_id = oi.order_id
GROUP BY c.customer_state;
```

Query results

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

Row	customer_state	Average_price	Total_price	Average_freight	Total_freight_val
1	MT	148.3	156453.53	28.17	29715.43
2	MA	145.2	119648.22	38.26	31523.77
3	AL	180.89	80314.81	35.84	15914.59
4	SP	109.65	5202955.05	15.15	718723.07
5	MG	120.75	1585308.03	20.63	270853.46
6	PE	145.51	262788.03	32.92	59449.66
7	RJ	125.12	1824092.67	20.96	305589.31
8	DF	125.77	302603.94	21.04	50625.5
9	RS	120.34	750304.02	21.74	135522.74
10	SE	153.04	58920.85	36.65	14111.47

5. Analysis on sales, freight and delivery time

1. Calculate days between purchasing, delivering and estimated delivery

Query:

```
SELECT order_id,
(DATE_DIFF(order_delivered_customer_date,order_purchase_timestamp, DAY)) AS delivery_time,
(DATE_DIFF(order_estimated_delivery_date, order_purchase_timestamp, DAY)) AS estimated_delivery_ti
me,
(DATE_DIFF(order_estimated_delivery_date, order_delivered_customer_date, DAY)) AS diff_actaul_estima
ted
FROM `scaler-target-sql-374814.Targer_sql.orders`
WHERE order_status = 'delivered';
```

Query results

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

Row	order_id	delivery_time	estimated_deliv	diff_actaul_estin
1	635c894d068ac37e6e03dc54e...	30	32	1
2	3b97562c3aee8bdedcb5c2e45...	32	33	0
3	68f47f50f04c4cb6774570cfde...	29	31	1
4	276e9ec344d3bf029ff83a161c...	43	39	-4
5	54e1a3c2b97fb0809da548a59...	40	36	-4
6	fd04fa4105ee8045f6a0139ca5...	37	35	-1
7	302bb8109d097a9fc6e9cefc5...	33	28	-5
8	66057d37308e787052a32828...	38	32	-6
9	19135c945c554eebfd7576c73...	36	33	-2
10	4493e45e7ca1084efcd38ddeb...	34	33	0

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

Query:

```
SELECT order_id,

(DATE_DIFF(order_delivered_customer_date, order_purchase_timestamp, DAY)) AS time_to_delivery,
(DATE_DIFF(order_estimated_delivery_date, order_delivered_customer_date, DAY)) AS diff_estimated
_delivery
FROM `scaler-target-sql-374814.Targer_sql.orders`
WHERE order_status = 'delivered';
```

Query results

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

Row	order_id	time_to_delivery	diff_estimated_c
1	635c894d068ac37e6e03dc54e...	30	1
2	3b97562c3aee8bdedcb5c2e45...	32	0
3	68f47f50f04c4cb6774570cfde...	29	1
4	276e9ec344d3bf029ff83a161c...	43	-4
5	54e1a3c2b97fb0809da548a59...	40	-4
6	fd04fa4105ee8045f6a0139ca5...	37	-1
7	302bb8109d097a9fc6e9cefc5...	33	-5
8	66057d37308e787052a32828...	38	-6
9	19135c945c554eebfd7576c73...	36	-2
10	4493e45e7ca1084efcd38ddeb...	34	0

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

Query:

```
SELECT c.customer_state,
ROUND(AVG(oi.freight_value),2) AS average_value,
ROUND(AVG(a.time_to_delivery),2) AS avg_delivery_time,
ROUND(AVG(a.diff_estimated_delivery),2) AS avg_estimated_delivery
FROM `scaler-target-sql-374814.Targer_sql.customers` c
INNER JOIN
(SELECT *,
(DATE_DIFF(order_delivered_customer_date, order_purchase_timestamp, DAY)) AS time_to_delivery,
(DATE_DIFF(order_estimated_delivery_date, order_delivered_customer_date, DAY)) AS diff_estimated_de
livery
FROM `scaler-target-sql-374814.Targer_sql.orders`
WHERE order_status = 'delivered') AS a ON a.customer_id = c.customer_id
INNER JOIN `scaler-target-sql-374814.Targer_sql.order_items` AS oi ON oi.order_id = a.order_id
GROUP BY c.customer_state;
```

Query results

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

Row	customer_state	average_value	avg_delivery_time	avg_estimated_delivery
1	GO	22.56	14.95	11.37
2	SP	15.12	8.26	10.26
3	RS	21.61	14.71	13.2
4	BA	26.49	18.77	10.12
5	MG	20.63	11.51	12.4
6	MT	28.0	17.51	13.64
7	RJ	20.91	14.69	11.14
8	SC	21.51	14.52	10.66
9	SE	36.57	20.98	9.17
10	PE	32.69	17.79	12.55

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:

A. Top 5 states with highest average freight value

```
SELECT c.customer_state,

ROUND(AVG(oi.freight_value),2) AS average_value
FROM `scaler-target-sql-374814.Targer_sql.customers` c
INNER JOIN `scaler-target-sql-374814.Targer_sql.orders` o ON o.customer_id = c.customer_id
INNER JOIN `scaler-target-sql-374814.Targer_sql.order_items` oi ON o.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	average_value
1	RR	42.98
2	PB	42.72
3	RO	41.07
4	AC	40.07
5	PI	39.15

B. Top 5 states with lowest average freight value

```
SELECT c.customer_state,
ROUND(AVG(oi.freight_value),2) AS average_value
FROM `scaler-target-sql-374814.Targer_sql.customers` c
INNER JOIN `scaler-target-sql-374814.Targer_sql.orders` o ON o.customer_id = c.customer_id
INNER JOIN `scaler-target-sql-374814.Targer_sql.order_items` oi ON o.order_id = oi.order_id
GROUP BY c.customer_state
ORDER BY AVG(oi.freight_value)
LIMIT 5;
```

Query results

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

Row	customer_state	average_value
1	SP	15.15
2	PR	20.53
3	MG	20.63
4	RJ	20.96
5	DF	21.04

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

Query:

A. Top 5 states with highest average time to delivery

```
SELECT c.customer_state,
ROUND(AVG(a.time_to_delivery),2) AS avg_delivery_time
FROM `scaler-target-sql-374814.Targer_sql.customers` c
INNER JOIN
(SELECT *,
(DATE_DIFF(order_delivered_customer_date, order_purchase_timestamp, DAY)) AS time_to_delivery,
FROM `scaler-target-sql-374814.Targer_sql.orders`
```

```
WHERE order_status = 'delivered') AS a ON a.customer_id = c.customer_id
GROUP BY c.customer_state
ORDER BY AVG(a.time_to_delivery) DESC
LIMIT 5;
```

Query results

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

Row	customer_state	avg_delivery_time
1	RR	28.98
2	AP	26.73
3	AM	25.99
4	AL	24.04
5	PA	23.32

B. Top 5 states with lowest average time to delivery

```
SELECT c.customer_state,
ROUND(AVG(a.time_to_delivery),2) AS avg_delivery_time
FROM `scaler-target-sql-374814.Targer_sql.customers` c
INNER JOIN
(SELECT *,
(DATE_DIFF(order_delivered_customer_date, order_purchase_timestamp, DAY)) AS time_to_delivery,
FROM `scaler-target-sql-374814.Targer_sql.orders`
WHERE order_status = 'delivered') AS a ON a.customer_id = c.customer_id
GROUP BY c.customer_state
ORDER BY AVG(a.time_to_delivery)
LIMIT 5 ;
```

Query results

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

Row	customer_state	avg_delivery_time
1	SP	8.3
2	PR	11.53
3	MG	11.54
4	DF	12.51
5	SC	14.48

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

Query:

A. Top 5 states where delivery is fast to estimated date

```
SELECT c.customer_state,
ROUND(AVG(a.diff_estimated_delivery),2) AS avg_estimated_delivery
FROM `scaler-target-sql-374814.Targer_sql.customers` c
INNER JOIN
(SELECT *,
(DATE_DIFF(order_estimated_delivery_date, order_delivered_customer_date, DAY)) AS diff_estimated
_delivery,
FROM `scaler-target-sql-374814.Targer_sql.orders`
WHERE order_status = 'delivered') AS a ON a.customer_id = c.customer_id
GROUP BY c.customer_state
ORDER BY AVG(a.diff_estimated_delivery)
LIMIT 5;
```

Query results

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

Row	customer_state	avg_estimated_delivery
1	AL	7.95
2	MA	8.77
3	SE	9.17
4	ES	9.62
5	BA	9.93

A. Top 5 states where delivery is not so fast to estimated date

```
SELECT c.customer_state,
ROUND(AVG(a.diff_estimated_delivery),2) AS avg_estimated_delivery
FROM `scaler-target-sql-374814.Targer_sql.customers` c
INNER JOIN
(SELECT *,
(DATE_DIFF(order_estimated_delivery_date, order_delivered_customer_date, DAY)) AS diff_estimated_de
livery,
FROM `scaler-target-sql-374814.Targer_sql.orders`
WHERE order_status = 'delivered') AS a ON a.customer_id = c.customer_id
GROUP BY c.customer_state
ORDER BY AVG(a.diff_estimated_delivery) DESC
LIMIT 5;
```

Query results

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

Row	customer_state	avg_estimated_delivery
1	AC	19.76
2	RO	19.13
3	AP	18.73
4	AM	18.61
5	RR	16.41

6. Payment type analysis:

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

Query:

```
SELECT payment_type,
EXTRACT(month FROM order_purchase_timestamp) AS month,
EXTRACT(year FROM order_purchase_timestamp) AS year,
COUNT(DISTINCT(o.order_id)) AS total_orders
FROM `scaler-target-sql-374814.Targer_sql.orders` AS o
INNER JOIN `scaler-target-sql-374814.Targer_sql.payments` AS p ON o.order_id = p.order_id
GROUP BY payment_type, year, month;
```

Query results

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

Row	payment_type	month	year	total_orders
1	UPI	11	2017	1509
2	credit_card	12	2017	4363
3	UPI	2	2018	1325
4	credit_card	11	2017	5867
5	voucher	4	2017	115
6	credit_card	7	2017	3072
7	UPI	7	2017	845
8	credit_card	5	2018	5475
9	credit_card	10	2017	3510
10	credit_card	1	2018	5511

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

Query:

```
SELECT p.payment_installments,
COUNT(o.order_id) AS total_orders
FROM `scaler-target-sql-374814.Targer_sql.payments` p
INNER JOIN `scaler-target-sql-374814.Targer_sql.orders`  o ON p.order_id = o.order_id
GROUP BY  p.payment_installments;
```

Query results

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

INSIGHTS AND RECOMMENDATIONS:

- We can see that maximum number of orders are made during the afternoon (12PM- 6PM) and night(6PM-12AM), which is nearly more than three fourth of the total orders. So, Target can make sure that their website is running smoothly. They can launch different offer to attract more customer.
- The distribution of customers in Brazil is over the 27 states, out of which SP has 40302 customers which is also highest in number, second highest is RJ with 12384 customers, third is MG with 11259 customers. So, more warehouses and stores should be open in such places to reduce transportation cost which can also reduce delivery time.
- Target should work on customer experiences by taking their valuable feedback and work on the improvement of it. On some places estimated delivery time is too long, this cannot happen because such things directly affect the placement of order.
- Low customer base states like AP, AC, RR and more where customer count and sellers as less . Target should work on that area by doing campaign, marketing strategies, giving discounts, social media promotions to attract more people.