Target Case Study (SQL Project)

This case study is based on data received from E-commerce Company known as Target. Data collected is from 2016 to 2018. This case study answers questions that are required to understand working of business in detail.

The data that is available here cannot be shown because of NDA with the company. Queries that helped me achieved require output are written below. Also solution approach shows how the solution was achieved.

We are using Google Big Query to process data for our case study.

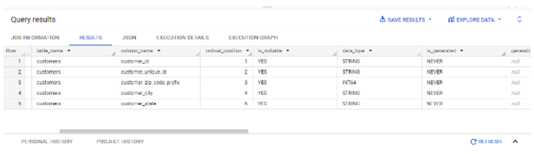
Import the dataset and do usual exploratory analysis steps like checking the structure & characteristics of the dataset:

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

   Solution Approach:
   - Here we need to find schema of all the columns mentioned in the table.
   - We can use simple Table_name.Information_schema.columns to get information about the columns.

   Query:

   ```
   SELECT * FROM Target.INFORMATION_SCHEMA.COLUMNS
   where table_name = 'customers'
   ```

   Query results

Row	table_name	column_name	ordinal_position	is_nullable	data_type	is_generated	generati
1	customers	customer_id	1	YES	STRING	NEVER	null
2	customers	customer_unique_id	2	YES	STRING	NEVER	null
3	customers	customer_zip_code_prefix	3	YES	INT64	NEVER	null
4	customers	customer_city	4	YES	STRING	NEVER	null
5	customers	customer_state	5	YES	STRING	NEVER	null

   Insights:
   From above results we can say only customer_zip_code_index dataype is INT64, rest all other columns are in string format.

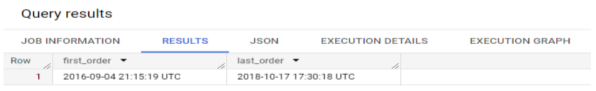
2. Get the time range between which the orders were placed.

   Solution Approach:
   - Here we are trying to find dates from which data ranges.
   - By using simple aggregation function we will be able to calculate the date and time for orders placed.

Query:

```
select
min(order_purchase_timestamp) as first_order,
max(order_purchase_timestamp) as last_order
from `target-bussiness-usecase.Target.orders`
```

Output:



Query results

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

Row	first_order	last_order
1	2016-09-04 21:15:19 UTC	2018-10-17 17:30:18 UTC

Insights:
Output of query concludes that first order and last order placed were 04-09-2016 and 17-10-2018 respectively.so the data is between these two dates.

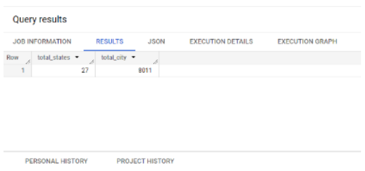
3. Count the number of Cities and States in our dataset.

   Solution Approach:
   - We have to get total number of cities and states where business is active.
   - For that purpose we can use count function to count distinct cities and states which are available in data.
   - The table we have to use is geolocation data, as it stores all the geographical information about business

   Query:

```
select
COUNT(DISTINCT geolocation_state) AS total_states,
COUNT(DISTINCT geolocation_city) as total_city
from `target-bussiness-usecase.Target.geolocation`
```

Output :



Query results

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

Row	total_states	total_city
1	27	8011

PERSONAL HISTORY | PROJECT HISTORY

Insights :

There are total 27 different states and 8011 total different cities in the given table.

2. **In-depth Exploration:**

1. Is there a growing trend in the no. of orders placed over the past years?

   Solution Approach:

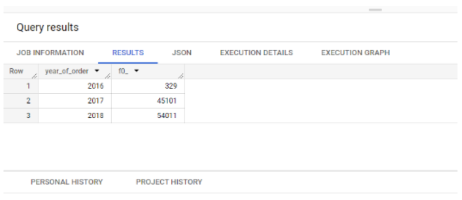
   - In this question we are trying to find how many orders were placed each year.
   - There is no column as order_year, so we are extracting year from order purchase timestamp column and renaming as year_of_order.
   - We create CTE for storing to new information with original values.
   - After that we use this column to group by year order and getting count of order_id to get number of orders that are being placed.

   Query:

```
with cte as (select *,
extract(year FROM order_purchase_timestamp) as year_of_order
from `target-bussiness-usecase.Target.orders`)

select year_of_order,count(order_id),
from cte
group by year_of_order
order by year_of_order
```

   Output:



Query results

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

Row	year_of_order	f0_
1	2016	329
2	2017	45101
3	2018	54011

PERSONAL HISTORY | PROJECT HISTORY

   Insights:

   From the above results we can see that number of orders have been increased from 2016 to 2018. There was major improvement from 2016 to 2017. We can see there is significant difference in 2017 to 2018 compared to 2016 to 2017.
   Orders were maximum for year 2018(54011) and minimum in 2016(329)

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

   Solution Approach:

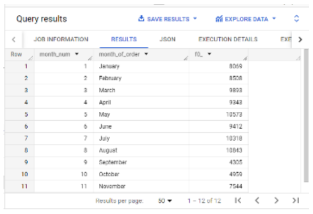
   - For this question we extract months from order_purchase_date column.
   - The months extracted will be number of months and month name regardless of year.
   - This is because we want number of orders that are placed each month every year.
   - First we create two columns that month num and month_of_order where we will be having numerical month number and actual month name.
   - We get these columns in CTE and after that we group by month of order and month num where we will be counting the number of orders that we were places using COUNT aggregation function.
   - This will provide us with number of orders placed every month.

   Query:

```
with cte as (select *,
extract(month from order_purchase_timestamp ) as month_num,
FORMAT_DATE('%B', order_purchase_timestamp) AS month_of_order
from `target-bussiness-usecase.Target.orders`)

select month_num, month_of_order,count(order_id)
from cte
group by month_of_order,month_num
order by month_num
```

   Output :



Query results

Row	month_num	month_of_order	f0_
1	1	January	8069
2	2	February	8508
3	3	March	9893
4	4	April	9343
5	5	May	10573
6	6	June	9412
7	7	July	10318
8	8	August	10843
9	9	September	4305
10	10	October	4959
11	11	November	7544

   Insights:

   Above output shows that order were peak in month of august. And order were lowest in month of September. The time period from May to august shows highest number of orders. This shows that year on year, this period gives more number of order and results in better sales and revenue.

3. During what time of the day, do the Brazilian customers mostly place their orders?

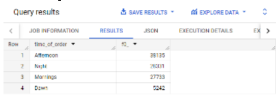
   Solution Approach:
   - Here we have to find at what time of day the orders where placed.
   - We use case when and extract to extract hour from order purchase timestamp column.
   - This will give us hour in 24 hour format of time. Which can later bin them using case when statement as shown in query.
   - Later we have to summarize data by using group by with newly created column.
   - This will also provide us number of orders that were being placed in that time frame.

   Query:

```
with cte as (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 'Mornings'
  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 time_of_order
from `target-bussiness-usecase.Target.orders`)

select time_of_order,count(order_purchase_timestamp)
from cte
group by time_of_order
order by count(order_purchase_timestamp) DESC
```

   Output



Row	time_of_order	f0_
1	Afternoon	38135
2	Night	28331
3	Mornings	27733
4	Dawn	5242

   Insights:

   From the above table we can see that mostly orders are placed in afternoon. Orders are placed least at time dawn. Orders frequency at time of night and mornings is nearly similar.

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

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

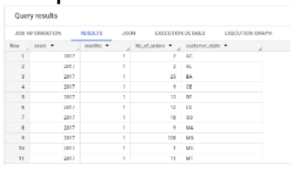
   Solution Approach:

   - Here we are to trying to find out how many orders were placed in each month in every state.
   - We need to join orders and customers table as orders table don't have state of delivery column.
   - After joining both the tables we extract month and year separately from order_date column.
   - We also get count of orders that were placed to get number of orders.
   - We use group by to summarize data according to month, year and state of customers belong to.

   Query:

```
SELECT extract(year from o.order_purchase_timestamp) as years,
extract(Month from o.order_purchase_timestamp) as months ,
count(o.order_id) as No_of_orders,
c.customer_state,
from `Target.orders` o join `Target.customers` c on o.customer_id = c.customer_id
group by months,years,customer_state
order by months,years,customer_state;
```

   Output



Query results

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

Row	years	months	No_of_orders	customer_state
1	2017	1	2	AC
2	2017	1	2	AL
3	2017	1	25	BA
4	2017	1	9	CE
5	2017	1	13	DF
6	2017	1	12	ES
7	2017	1	18	GO
8	2017	1	9	MA
9	2017	1	108	MG
10	2017	1	1	MS
11	2017	1	11	MT

   Insights:

   The above table helps compare the number of orders placed in every month of every year between 2016-2018 in different states in Brazil. These insights can help Target gain insights into sales, customer behaviour, if any, from which state it received maximum orders between 2016-2018, etc.

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

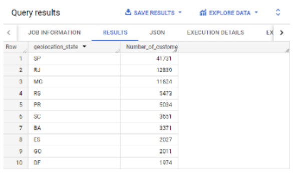
   Solution Approach :

   - We have summarize data according to state wise.
   - Here we have to join two tables as customer table and geolocation table using zip code prefix.
   - After joining, we will group by data according to geolocation state to get aggregated result.
   - We will select state and using count function we will count unique customer if from customer's table.
   - Lastly we will get total number of customers that are available in different states.

Query:

```
select g.geolocation_state, count(distinct c.customer_id) as Number_of_customers
from `Target.customers` c join `Target.geolocation` g
on c.customer_zip_code_prefix = g.geolocation_zip_code_prefix
group by g.geolocation_state
order by count(distinct c.customer_id) desc
limit 10
```

Output



Row	geolocation_state	Number_of_custome
1	SP	41731
2	RJ	12839
3	MG	11624
4	RS	5473
5	PR	5034
6	SC	3651
7	BA	3371
8	ES	2027
9	GO	2011
10	DF	1974

Insights:

The above results suggest that Target has most customers in the state SP. Insights like these can help Target manage their inventories. It also helps Target to understand the most preferable location to open new stores. This will also help them identify areas with potential for growth and better understand the preferences of their customers. Target can also use this information to introduce customized promotions and offers for their customers to increase the sales and customer satisfaction.

4. 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).

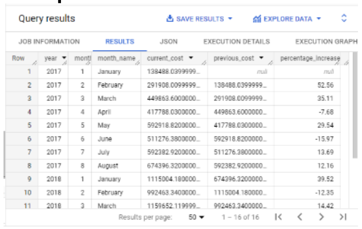
   Solution Approach :
   - First we join orders table with payment table using order_id.
   - After joining the two tables we filter out data in months from January to august and for year 2017 and 2018.
   - Also we create new columns for year,month and using sum function to get total cost of orders as current cost.
   - To summarize it we need to group by according to year and month and create CTE from this.
   - After we created CTE, we use lag function which is ordered by year and month to generate new column which has values of previous row.
   - Now we get the difference of orders from current cost column and previous cost column and multiply it by 100 and rounding them to two decimal place.
   - This new column will provide us with % cost increase or decrease month wise for year 2017 and 2017 for months January to August.

   Query:

```
with cte as (select year,month,month_name,sum(payment_value) current_cost,
from (select *,
extract(month from o.order_purchase_timestamp) as month,
extract(year from o.order_purchase_timestamp) as year,
FORMAT_DATE('%B', o.order_purchase_timestamp) AS month_name
from `target-bussiness-usecase.Target.orders` o join `Target.payments` p
on o.order_id=p.order_id
where extract(month from o.order_purchase_timestamp) <9 and extract(year from
o.order_purchase_timestamp) in (2017,2018))
group by year,month,month_name
order by year,month)

select *,
lag(cte.current_cost) over (order by year,month) as previous_cost,
round(((current_cost - (lag(cte.current_cost) over (order by
year,month)))/current_cost)*100,2) as percentage_increase
from cte
order by year,month
```

   Output



Query results

Row	year	montl	month_name	current_cost	previous_cost	percentage_increase
1	2017	1	January	138488.0399999...	null	null
2	2017	2	February	291908.0099999...	138488.0399999...	52.56
3	2017	3	March	449863.6000000...	291908.0099999...	35.11
4	2017	4	April	417788.0300000...	449863.6000000...	-7.68
5	2017	5	May	592918.8200000...	417788.0300000...	29.54
6	2017	6	June	511276.3800000...	592918.8200000...	-15.97
7	2017	7	July	592382.9200000...	511276.3800000...	13.69
8	2017	8	August	674396.3200000...	592382.9200000...	12.16
9	2018	1	January	1115004.180000...	674396.3200000...	39.52
10	2018	2	February	992463.3400000...	1115004.180000...	-12.35
11	2018	3	March	1159652.119999...	992463.3400000...	14.42

Results per page: 50 1 – 16 of 16

Insights:

From the above table we can conclude that there war increase in cost of 52% for month of February which was also highest cost as can be seen. The % cost decreases into following months. The % cost was decreased lowest in month of June 2017 which was negative. This shows that cost of orders was lower than preivous month.

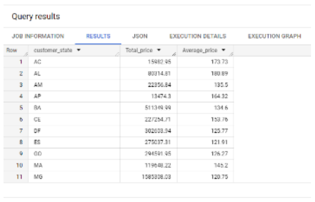
2. Calculate the Total & Average value of order price for each state.

   Solution Approach:
   - Here we are joining orders table and customers table.Also we will be joining orderitems table on ordered.
   - Now after joining all these tables we group by on customer state column from customers table
   - We want aggregated result for order price so we use sum and avg aggregation function to get total and average price by state.

   Query:

```
select c.customer_state,round(sum(oi.price),2) as Total_price,
round(avg(oi.price),2) as Average_price
from `Target.orders` o join `Target.customers` c on o.customer_id = c.customer_id
join `Target.order_items` oi on o.order_id=oi.order_id
group by c.customer_state
order by c.customer_state
```

   Output



Query results

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

Row	customer_state	Total_price	Average_price
1	AC	15982.95	173.73
2	AL	80314.81	180.89
3	AM	22356.84	135.5
4	AP	13474.3	164.32
5	BA	511349.99	134.6
6	CE	227254.71	153.76
7	DF	302603.94	125.77
8	ES	275037.31	121.91
9	GO	294591.95	126.27
10	MA	119648.22	145.2
11	MG	1585308.03	120.75

   Insights:

   From above table we can conclude that average price for customer is 180 to be maximum and maximum amount of total orders are 1585308.
   The average price and total price provide insight into the cost price of sales in each Brazilian state between 2016 and 2018. By tracking the average price and total price of sales in each Brazilian state over the course of 3 years, we can get an accurate picture of the cost price of goods and services in each state. This allows us to compare states and identify trends in pricing over time.

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

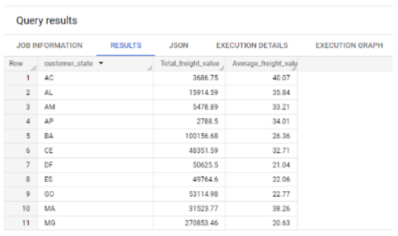
   Solution Approach:

   - Here we are joining orders table and customers table. Also we will be joining order items table on ordered.
   - Now after joining all these tables we group by on customer state column from customers table
   - We want aggregated result for freight value so we use sum and avg aggregation function to get total and average freight value by state.

   Query:

```
select c.customer_state,round(sum(oi.freight_value),2) as Total_freight_value,
round(avg(oi.freight_value),2) as Average_freight_value
from `Target.orders` o join `Target.customers` c on o.customer_id = c.customer_id
join `Target.order_items` oi on o.order_id=oi.order_id
group by c.customer_state
order by c.customer_state
```

   Output



Query results

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

Row	customer_state	Total_freight_value	Average_freight_valu
1	AC	3686.75	40.07
2	AL	15914.59	35.84
3	AM	5478.89	33.21
4	AP	2788.5	34.01
5	BA	100156.68	26.36
6	CE	48351.59	32.71
7	DF	50625.5	21.04
8	ES	49764.6	22.06
9	GO	53114.98	22.77
10	MA	31523.77	38.26
11	MG	270853.46	20.63

   Insights:

   From the above output we see average freight value is maximum for state AC. Also, the total freight value is maximum at state MG which also has lower Average freight value. This shows that we for lower total freight value Target pays more as average freight value and vice versa.
   By reducing the Average freight value and total freight value, a customer will have to pay less for shipping, thereby attracting more customers to the site. Lower costs can lead to higher sales and profitability. Additionally, lower shipping costs will make the company more competitive in the market and give it an edge over its competitors.

5. 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.

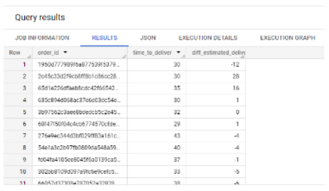
   Solution Approach:

   - Here we have to find how difference can be found in estimated and actual delivery time.
   - To calculate that we us SQL's in-built date_diff function to calculate difference of days between two dates.
   - First we calculate data difference from customer order date and estimated order delivery date
   - Then we calculate difference between order date and actual delivery date.This gives us comparison of difference in days of delivery.

   Query:

```
select order_id,
date_diff(order_delivered_customer_date, order_purchase_timestamp,day) as time_to_deliver,
date_diff(order_estimated_delivery_date, order_delivered_customer_date,day) as
diff_estimated_delivery
from `Target.orders`
```

   Output



Query results

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

Row	order_id	time_to_deliver	diff_estimated_deliv
1	1950d777989f6a877539f5379...	30	-12
2	2c45c33d2f9cb8ff8b1c86cc28...	30	28
3	65d1e226dfaeb8cdc42f66542...	35	16
4	635c894d068ac37e6e03dc54e...	30	1
5	3b97562c3aee8bdedcb5c2e45...	32	0
6	68f47f50f04c4cb6774570cfde...	29	1
7	276e9ec344d3bf029ff83a161c...	43	-4
8	54e1a3c2b97fb0809da548a59...	40	-4
9	fd04fa4105ee8045f6a0139ca5...	37	-1
10	302bb8109d097a9fcbe9cefc5...	33	-5
11	66057d37308e787052e32828	38	-6

   Insights:

   The table above shows the actual delivery time and the estimated delivery time in days format. We can see diff_estimated_delivery has some negative values but time to deliver doesn't have such value. The negative values indicate that orders were delivered late. This orders were shipped after the estimated delivery date. Some orders have long delivery times, and they are also late. Between 2016-2018, the maximum time it took to deliver a product to a Brazilian customer was 210 days, and the minimum was 0 days.

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

   Solution Approach:

   - At first we have to join orders and customers table on customer_id, also we join order_items table on order_id.
   - We will aggregate the data on customer state from customers table using Group By.
   - Now we select customer state from customers table, using avg aggregate function to average the cost of freight value and rename as average_freight_value.
   - Now we sort data using order by clause to get lowest average freight value. By default SQL provides result in ascending order.
   - To get results for highest average freight value, we will be using desc at end of order by clause.

   Query:

   For Highest :

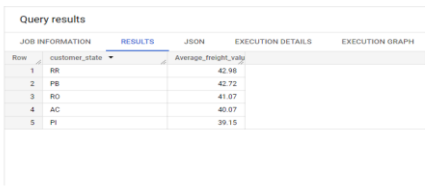
```
select c.customer_state,round(avg(oi.freight_value),2) as Average_freight_value
from `Target.orders` o join `Target.customers` c on o.customer_id = c.customer_id
join `Target.order_items` oi on o.order_id=oi.order_id
group by c.customer_state
order by Average_freight_value desc
limit 5
```

   For Lowest :

```
select c.customer_state,round(avg(oi.freight_value),2) as Average_freight_value
from `Target.orders` o join `Target.customers` c on o.customer_id = c.customer_id
join `Target.order_items` oi on o.order_id=oi.order_id
group by c.customer_state
order by Average_freight_value asc
limit 5
```

   Output :

   Highest :

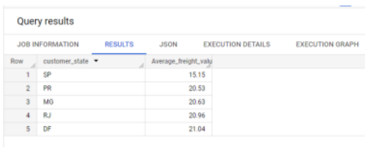


Query results

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

Row	customer_state	Average_freight_valu
1	RR	42.98
2	PB	42.72
3	RO	41.07
4	AC	40.07
5	PI	39.15

   Lowest :



Query results

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

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

Insights:

The freight value is the cost of transporting cargo from one place to another. Cargo size, weight, and distance between origin and delivery determine freight value. From the above results, it is observed that the average freight value was minimum for the state SP while maximum for the state RR in Brazil between 2016-2018. These insights can help Target understand its supply chain and exploring different modes of transport. This data can help Target optimize their supply chain and reduce their transport costs. They can also consider different transport modes to reduce their freight costs. Lastly, they can use this data to forecast and plan their supply chain more accurately.

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

   Solution Approach:
   - At first we join orders and customers table on customer id and group them by using customer state from customers table.
   - Now we select customer state and in-line calculation to find date difference in order purchase date and order delivery date and rename as time_to_deliver.
   - We sort data according to time_to_deliver column. By default result are sorted in ascending orders which will give us lowest average delivery time.
   - To get highest average delivery time we use desc at order by clause after time_to_deliver column name.

   Query:

   For Highest :

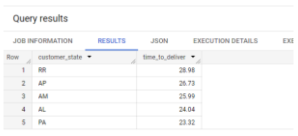
```
select c.customer_state,round(avg(date_diff(order_delivered_customer_date,
order_purchase_timestamp,day)),2) as time_to_deliver
from `Target.orders` o join `Target.customers` c on o.customer_id = c.customer_id
group by c.customer_state
order by time_to_deliver desc
limit 5
```

   For Lowest

```
select c.customer_state,round(avg(date_diff(order_delivered_customer_date,
order_purchase_timestamp,day)),2) as time_to_deliver
from `Target.orders` o join `Target.customers` c on o.customer_id = c.customer_id
group by c.customer_state
order by time_to_deliver asc
limit 5
select * from `Target.orders`
```

Output:

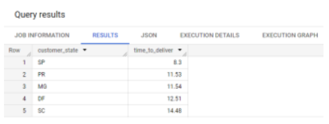
Highest :


Query results

JOB INFORMATION | RESULTS | JSON | EXECUTION DETAILS | EXE

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

Lowest :


Query results

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

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

Insights:

Based on the results above, one can conclude that the minimum average time it took to deliver a product in Brazil between 2016-2018 was 9 days. 29 days was the maximum time. The average time was 19 days. Delivery times varied according to the product type and customer location. It is clear that customers in remote areas faced longer delivery times than customers in metropolitan areas. In addition, the type of product was also a factor in delivery times

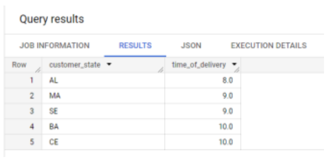
4. Find out the top 5 states where the order delivery is really fast as compared to the estimated date of delivery.

   Solution Approach:
   - Here we join orders and customers table on customer_id and aggregate them using group by function.
   - Now in select clause we have to do 2 calculations. Firstly we will calculate day difference between order purchase date and estimated delivery date.
   - Secondly we will calculate date difference between order purchase date and actual delivery date.
   - Now we get absolute difference of above both calculated values.
   - Lets now sort the data using order by clause and we will have required results.

   Query:

```
select c.customer_state,
round(avg(date_diff(order_estimated_delivery_date, order_purchase_timestamp,day))-
  avg(date_diff(order_delivered_customer_date, order_purchase_timestamp,day))) as
time_of_delivery
from `Target.orders` o join `Target.customers` c on o.customer_id = c.customer_id
group by c.customer_state
order by time_of_delivery asc
limit 5
```

   Output


Query results

JOB INFORMATION | RESULTS | JSON | EXECUTION DETAILS

Row	customer_state	time_of_delivery
1	AL	8.0
2	MA	9.0
3	SE	9.0
4	BA	10.0
5	CE	10.0

Insights:

According to the above results, AL had the fastest delivery time.In these states, orders have been delivered before the estimated delivery date.State AL had significantly faster delivery times compared to the other states.Furthermore, the data showed that AL consistently delivered orders before the estimated delivery date. This could be attributed to the state's superior logistics, which allowed them to optimize their delivery routes and cut down on shipping time.

6. **Analysis based on the payments:**

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

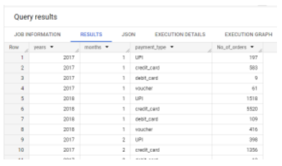
Solution Approach:

- Here we first join orders and payment table on order id column.
- After joining we extract month and year from order_purchase_timestamp.
- Now we use group by to get aggregated result for month , year , and payment_type.
- We also use aggregated function COUNT to get number of orders.

Query:

```
SELECT
extract(year from o.order_purchase_timestamp) as years,
extract(Month from o.order_purchase_timestamp) as months,p.payment_type ,
count(o.order_id) as No_of_orders,
from `Target.payments` as p join `Target.orders` as o using(order_id)
group by months,years,payment_type
order by months,years,payment_type,No_of_orders;
```

Output



Query results

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

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

Insights:

Credit cards were found to be the most preferred mode of payment for 2016, 2017 and 2018. This is likely due to the convenience and ease of use of credit cards. In addition, the higher levels of protection they offer from fraud and identity theft. Insights like these can assist Target in understanding how most of their customers make their payments. By giving customers cashbacks, vouchers, and other flexible payment options, Target can acquire more customers.

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

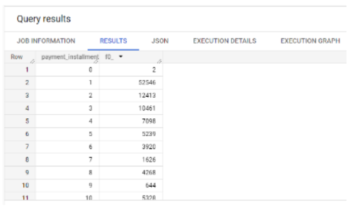
   Solution Approach:

   - At first we have to join orders table to payment table to get payment installations for each orders.
   - After joining the tables we group by on payment installation column.
   - Also we use count aggregate function to get number of orders that are being placed according to payment installations.

   Query:

```
select p.payment_installments, count(o.order_id)
from `Target.payments` as p join `Target.orders` as o using(order_id)
group by p.payment_installments
order by p.payment_installments
```

   Output



Query results

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

Row	payment_installment	f0_
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
11	10	5328

   Insights:

   From the above data we can conclude that customers mostly chose to do payment in 1 instalment. While only 2 customers chose to complete the payment without installations. Also we can see that customers mostly chose payment instalments to be 1, 2, 3 as they large number of customers.

Recommendations:

- According to seasonality graph we can see that most order were placed from april to june. Also at this time period cost of orders was also lower as compared to month February.
  This suggest that customers tend to order items when the order cost is lower.

- States like SP , RJ , MG have most number of customers. Also the minimum deliver time is for state SP. To expand business Target can use similar strategy to state like RJ,MG as State of SP.

- Most of customers chose to pay using credit card. So Target can work more on this payment type and include more offers, coupons to increase sales