Target SQL Business Case

Vishal Kamath

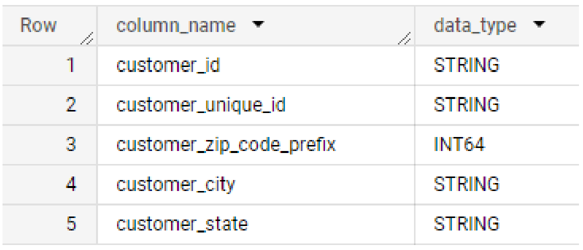
1. Import the dataset and do usual exploratory analysis steps like checking the structure & characteristics of the dataset.

A. Data type of all columns in the “customers” table.

i. **Query:**

```
SELECT
  column_name,
  data_type
FROM
  `target`.INFORMATION_SCHEMA.COLUMNS
WHERE
  table_name = 'customers';
```

ii. **Result:**



Row	column_name	data_type
1	customer_id	STRING
2	customer_unique_id	STRING
3	customer_zip_code_prefix	INT64
4	customer_city	STRING
5	customer_state	STRING

iii. **Insights:** NA

iv. **Recommendations:** NA

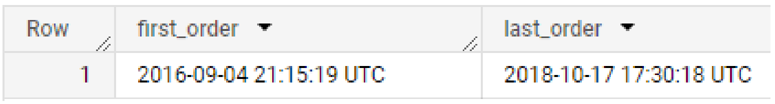
v. **Assumptions:** NA

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

i. **Query:**

```
SELECT
 MIN(order_purchase_timestamp) AS first_order,
 MAX(order_purchase_timestamp) AS last_order
FROM
 `target.orders`;
```

ii. **Result:**



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

iii. **Insights:** NA

iv. **Recommendations:** NA

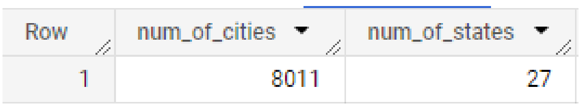
v. **Assumptions:** NA

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

i. **Query:**

```
SELECT
 COUNT(DISTINCT geolocation_city) AS num_of_cities,
 COUNT(DISTINCT geolocation_state) AS num_of_states
FROM
 target.geolocation`;
```

ii. **Result:**



Row	num_of_cities	num_of_states
1	8011	27

iii. **Insights:** NA

iv. **Recommendations:** NA

v. **Assumptions:** I am assuming that we need to consider all the states and cities even though there might not be any customers from those cities, hence choosing the table “geolocation”

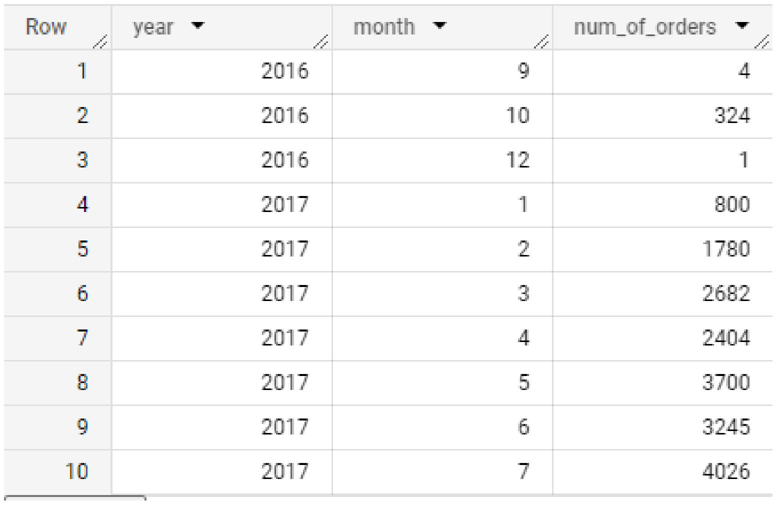
2. In-depth Exploration

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

i. **Query:**

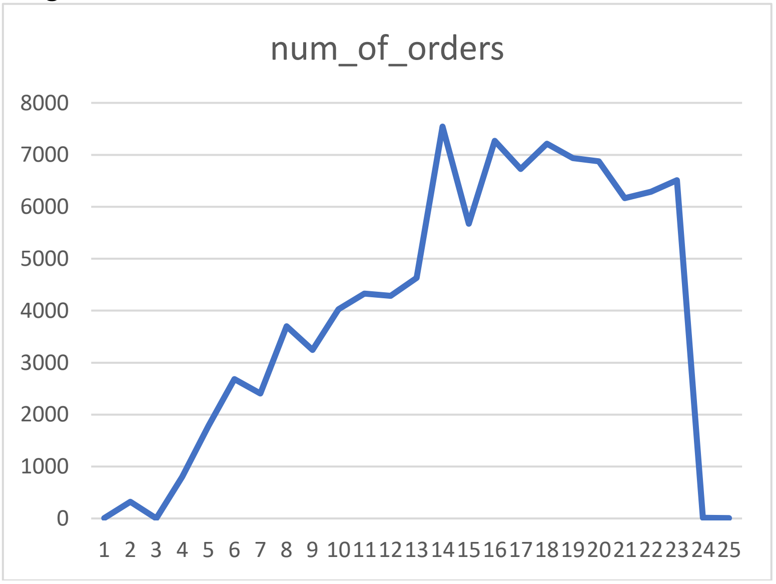
```
SELECT
  year,
  month,
  COUNT(month) AS num_of_orders
FROM (
  SELECT
    order_purchase_timestamp,
    EXTRACT(year
    FROM
      order_purchase_timestamp) AS year,
    EXTRACT(month
    FROM
      order_purchase_timestamp) AS month,
  FROM
    `target.orders`) AS orders_table
GROUP BY
  year,
  month
ORDER BY
  year,
  month;
```

ii. **Result:**



Row	year	month	num_of_orders
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

iii. **Insights:**



We see that the number of orders, in general, has increased month on month till Nov 2017 reaching a peak of 7544 then it has been consolidating around 6500 till Aug 2018 and then we see a sudden fall in the number of orders in the month of September and October 2018.

iv. **Recommendations:** Target should try to replicate the same which worked in getting increasing orders from Jan 2017 to Nov 2017

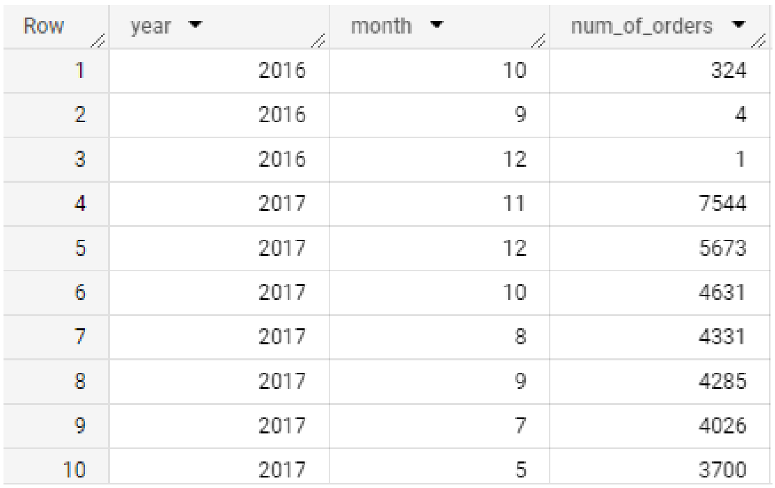
v. **Assumptions:** NA

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

i. **Query:**

```
SELECT year, month, COUNT(order_id) AS num_of_orders
FROM `target.orders_info`
GROUP BY year, month
ORDER BY year, num_of_orders DESC;
```

ii. **Result:**



Row	year	month	num_of_orders
1	2016	10	324
2	2016	9	4
3	2016	12	1
4	2017	11	7544
5	2017	12	5673
6	2017	10	4631
7	2017	8	4331
8	2017	9	4285
9	2017	7	4026
10	2017	5	3700

iii. **Insights:** With just 3 months data available in 2016, the number of orders peak in October. In 2017, the top 3 months are November, December and October. In 2018, January, March and April are the top 3 months. With the limited data available I don't see any monthly seasonality in terms of number of orders being placed.

iv. **Recommendations:** NA

v. **Assumptions:** NA

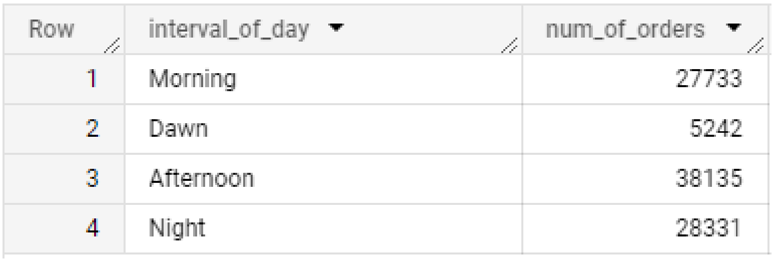
C. During what time of the day, do the Brazilian customers mostly place their orders? (Dawn, Morning, Afternoon or Night)

- 0-6 hrs : Dawn
- 7-12 hrs : Mornings
- 13-18 hrs : Afternoon
- 19-23 hrs : Night

i. **Query:**

```
WITH cte AS (
  SELECT
    CASE
      WHEN hour BETWEEN 0 AND 6 THEN 'Dawn'
      WHEN hour BETWEEN 7 AND 12 THEN 'Morning'
      WHEN hour BETWEEN 13 AND 18 THEN 'Afternoon'
      WHEN hour BETWEEN 19 AND 23 THEN 'Night'
    END AS interval_of_day
  FROM `target.orders_info`
)
SELECT interval_of_day, COUNT(*) AS num_of_orders
FROM cte
GROUP BY interval_of_day;
```

ii. **Result:**



Row	interval_of_day	num_of_orders
1	Morning	27733
2	Dawn	5242
3	Afternoon	38135
4	Night	28331

iii. **Insights:** We can see that that the customers order the most during afternoon and least during dawn with mornings and night, being almost same, in the middle .

iv. **Recommendations:** NA

v. **Assumptions:** NA

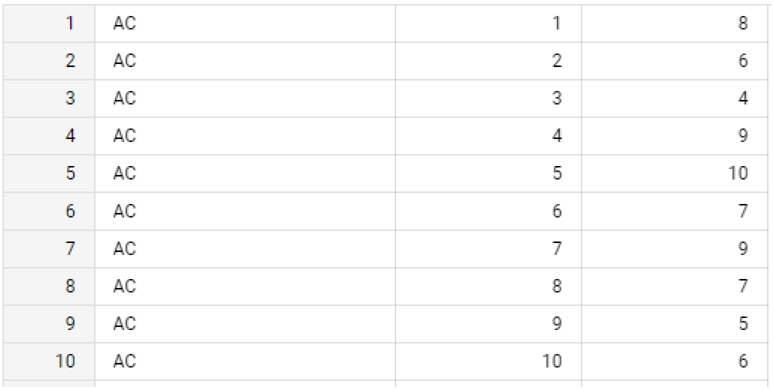
3. Evolution of E-commerce orders in the Brazil region:

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

i. **Query:**

```
WITH cte AS (
  SELECT O.order_id, C.customer_state, O.month
  FROM `target.orders_info` AS O INNER JOIN `target.customers` C ON O.customer_id =
C.customer_id)
SELECT customer_state, month, COUNT(order_id) AS num_of_orders
FROM cte
GROUP BY customer_state, month
ORDER BY customer_state, month;
```

ii. **Result:**



1	AC	1	8
2	AC	2	6
3	AC	3	4
4	AC	4	9
5	AC	5	10
6	AC	6	7
7	AC	7	9
8	AC	8	7
9	AC	9	5
10	AC	10	6

iii. **Insights:** NA

iv. **Recommendations:** NA

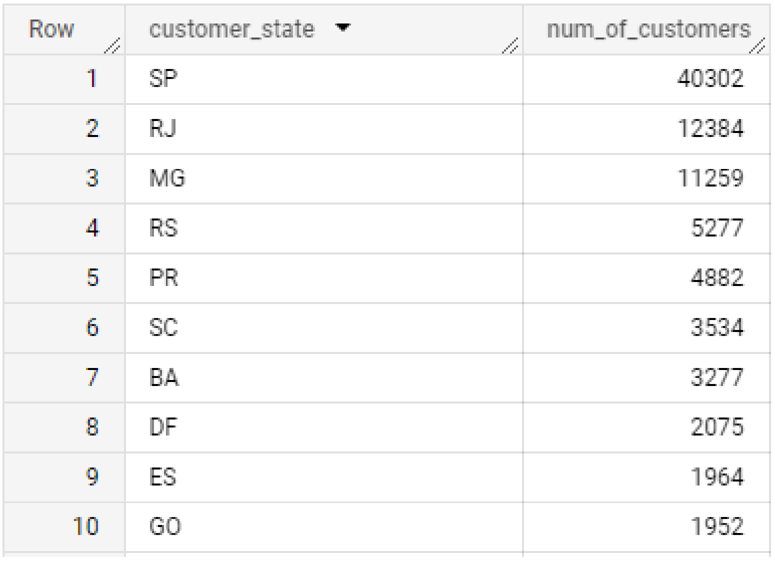
v. **Assumptions:** NA

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

i. **Query:**

```
SELECT customer_state, COUNT(DISTINCT customer_unique_id) AS num_of_customers
FROM `target.customers`
GROUP BY customer_state
ORDER BY num_of_customers DESC;
```

ii. **Result:**



Row	customer_state	num_of_customers
1	SP	40302
2	RJ	12384
3	MG	11259
4	RS	5277
5	PR	4882
6	SC	3534
7	BA	3277
8	DF	2075
9	ES	1964
10	GO	1952

iii. **Insights:** São Paulo has the largest number of customers followed not so closely by Rio de Janeiro and Minas Gerais. Roraima has the least number of customers.

iv. **Recommendations:** NA

v. **Assumptions:** NA

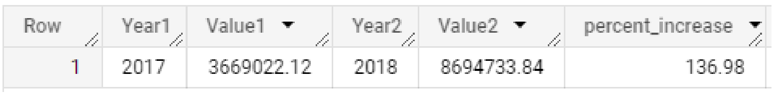
4. Impact on Economy: Analyze the money movement by e-commerce by looking at order prices, freight and others.

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

i. **Query:**

```
WITH cte AS (
 SELECT O.year, ROUND(SUM(P.payment_value), 2) AS total_payment_value
 FROM `target.orders_info` AS O INNER JOIN `target.payments` AS P
 USING (order_id)
 WHERE O.year BETWEEN 2017 AND 2018 AND O.month BETWEEN 1 AND 8
 GROUP BY year)
SELECT
  T1.year AS Year1, T1.total_payment_value AS Value1,
  T2.year AS Year2, T2.total_payment_value AS Value2,
  ROUND(((T2.total_payment_value -
T1.total_payment_value)/T1.total_payment_value)*100,2) AS percent_increase
FROM cte AS T1, cte AS T2
WHERE T1.year < T2.year;
```

ii. **Result:**



Row	Year1	Value1	Year2	Value2	percent_increase
1	2017	3669022.12	2018	8694733.84	136.98

iii. **Insights:** The cost of orders has increased by around 137% from year 2017 to 2018, which is a huge increase, more than double.

iv. **Recommendations:** NA

v. **Assumptions:** NA

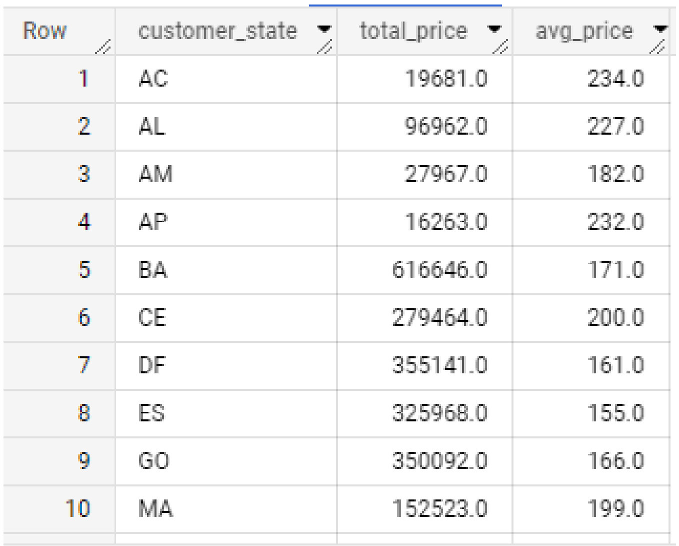
B. Calculate the Total & Average value of order price for each state.

i. **Query:**

```
SELECT
  C.customer_state,
  ROUND(SUM(P.payment_value), 0) AS total_price,
  ROUND(AVG(P.payment_value), 0) AS avg_price
FROM `target.orders_info` AS O INNER JOIN `target.payments` AS P ON O.order_id =
P.order_id
INNER JOIN `target.customers` AS C ON O.customer_id = C.customer_id
GROUP BY C.customer_state
```

```
ORDER BY C.customer_state;
```

ii. **Result:**



Row	customer_state	total_price	avg_price
1	AC	19681.0	234.0
2	AL	96962.0	227.0
3	AM	27967.0	182.0
4	AP	16263.0	232.0
5	BA	616646.0	171.0
6	CE	279464.0	200.0
7	DF	355141.0	161.0
8	ES	325968.0	155.0
9	GO	350092.0	166.0
10	MA	152523.0	199.0

iii. **Insights:** The total price is highest for São Paulo and this is expected as the number of customers are also highest in São Paulo. The average price is around 200

iv. **Recommendations:** NA

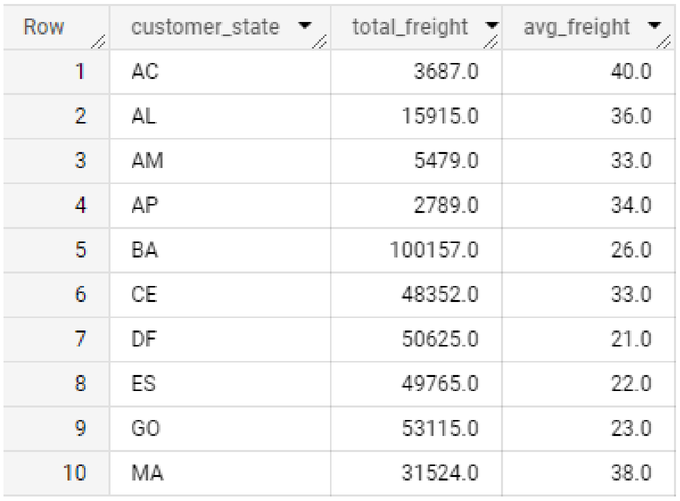
v. **Assumptions:** NA

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

i. **Query:**

```
SELECT customer_state, total_freight, avg_freight
FROM `target.freight_info`
ORDER BY customer_state;
```

ii. **Result:**



Row	customer_state	total_freight	avg_freight
1	AC	3687.0	40.0
2	AL	15915.0	36.0
3	AM	5479.0	33.0
4	AP	2789.0	34.0
5	BA	100157.0	26.0
6	CE	48352.0	33.0
7	DF	50625.0	21.0
8	ES	49765.0	22.0
9	GO	53115.0	23.0
10	MA	31524.0	38.0

iii. **Insights:** The total freight, following the similar pattern as total price, is highest for São Paulo. The average is around 30.

iv. **Recommendations:** NA

v. **Assumptions:** NA

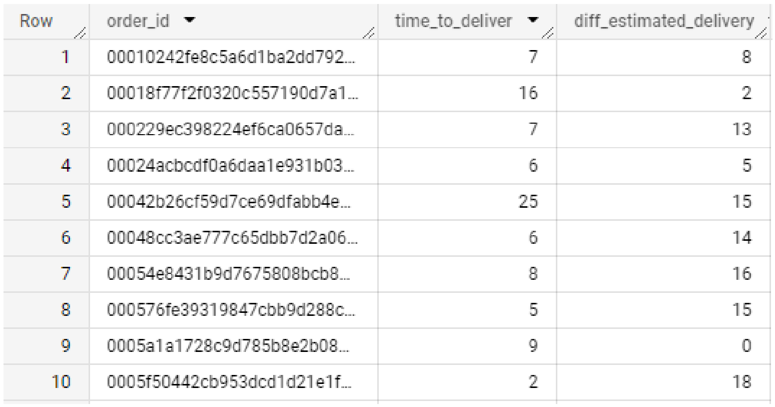
5. Analysis based on sales, freight and delivery time.

A. 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.

i. **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`
WHERE order_status = 'delivered'
ORDER BY order_id;
```

ii. **Result:**



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

iii. **Insights:** NA

iv. **Recommendations:** NA

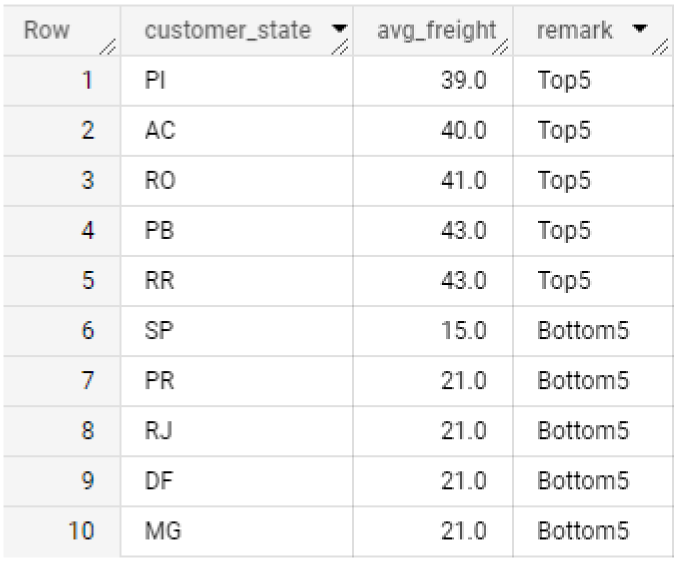
v. **Assumptions:** NA

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

i. **Query:**

```
WITH cte1 AS (
 SELECT customer_state, avg_freight, 'Bottom5' AS remark
 FROM `target.freight_info`
 ORDER BY avg_freight
 LIMIT 5),
cte2 AS (
 SELECT customer_state, avg_freight, 'Top5' AS remark
 FROM `target.freight_info`
 ORDER BY avg_freight desc
 LIMIT 5)
SELECT customer_state, avg_freight, remark FROM cte1
UNION ALL
SELECT customer_state, avg_freight, remark FROM cte2
ORDER BY remark DESC, avg_freight;
```

ii. **Result:**



Row	customer_state	avg_freight	remark
1	PI	39.0	Top5
2	AC	40.0	Top5
3	RO	41.0	Top5
4	PB	43.0	Top5
5	RR	43.0	Top5
6	SP	15.0	Bottom5
7	PR	21.0	Bottom5
8	RJ	21.0	Bottom5
9	DF	21.0	Bottom5
10	MG	21.0	Bottom5

iii. **Insights:** NA

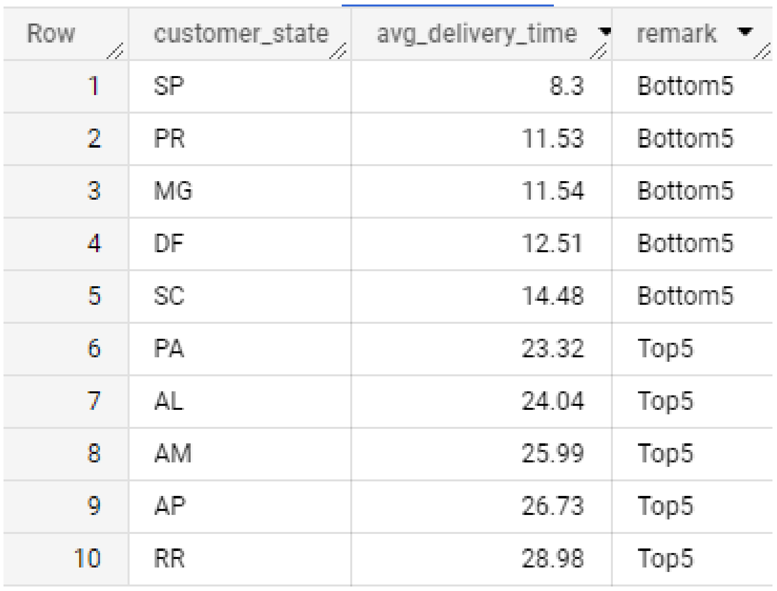
iv. **Recommendations:** NA
v. **Assumptions:** NA

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

i. **Query:**

```
WITH cte1 AS (
  SELECT
    customer_state,
    ROUND(AVG(DATE_DIFF(O.order_delivered_customer_date, O.order_purchase_timestamp,
day)),2) AS avg_delivery_time
  FROM `target.orders` AS O INNER JOIN `target.customers` AS C ON O.customer_id =
C.customer_id
  WHERE O.order_status = 'delivered'
  GROUP BY customer_state),
cte2 AS (
  SELECT
    customer_state,
    avg_delivery_time,
    ROW_NUMBER() OVER(ORDER BY avg_delivery_time desc) AS TopRnk,
    ROW_NUMBER() OVER(ORDER BY avg_delivery_time) AS BottomRnk
  FROM cte1)
SELECT
  customer_state,
  avg_delivery_time,
  CASE
    WHEN TopRnk <= 5 THEN 'Top5'
    WHEN BottomRnk <= 5 THEN 'Bottom5'
  END AS remark
FROM cte2
WHERE TopRnk <= 5 OR BottomRnk <= 5;
```

ii. **Result:**



Row	customer_state	avg_delivery_time	remark
1	SP	8.3	Bottom5
2	PR	11.53	Bottom5
3	MG	11.54	Bottom5
4	DF	12.51	Bottom5
5	SC	14.48	Bottom5
6	PA	23.32	Top5
7	AL	24.04	Top5
8	AM	25.99	Top5
9	AP	26.73	Top5
10	RR	28.98	Top5

iii. **Insights:** NA
iv. **Recommendations:** NA
v. **Assumptions:** NA

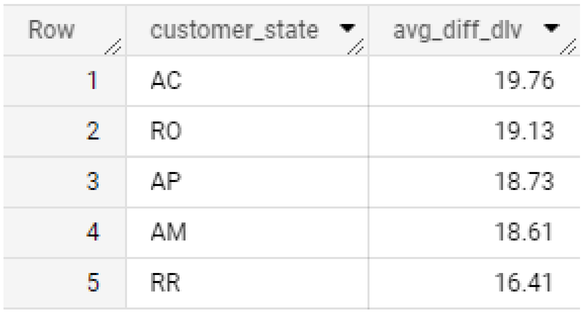
D. Find out the top 5 states where the order delivery is really fast as compared to the estimated date of delivery.

i. **Query:**

```
WITH cte AS(
  SELECT
```

```
    customer_id,
    DATE_DIFF(order_estimated_delivery_date, order_delivered_customer_date, day) AS
diff_delv
  FROM `target.orders`
  WHERE order_status = 'delivered'
)
SELECT C.customer_state, ROUND(AVG(CT.diff_delv),2) AS avg_diff_dlv
FROM cte AS CT INNER JOIN `target.customers` AS C USING (customer_id)
GROUP BY C.customer_state
ORDER BY avg_diff_dlv DESC
LIMIT 5;
```

ii. **Result:**



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

iii. **Insights:** The top state has order delivery 20 days faster than the estimate date.

iv. **Recommendations:** NA

v. **Assumptions:** NA

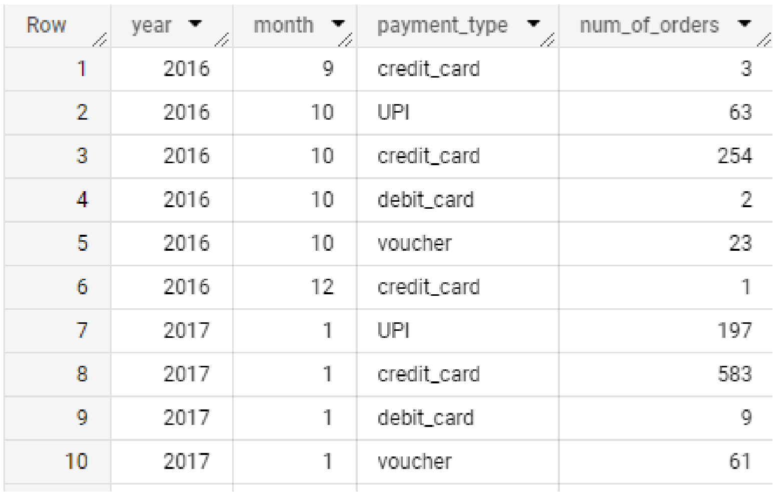
6. Analysis based on the payments:

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

i. **Query:**

```
SELECT
 O.year,
 O.month,
 P.payment_type,
 COUNT(O.order_id) AS num_of_orders
FROM `target.orders_info` AS O INNER JOIN `target.payments` AS P
USING (order_id)
GROUP BY O.year, O.month, P.payment_type
ORDER BY O.year, O.month, P.payment_type;
```

ii. **Result:**



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

iii. **Insights:** Majority of the orders are placed using credit card followed by UPI

iv. **Recommendations:** NA

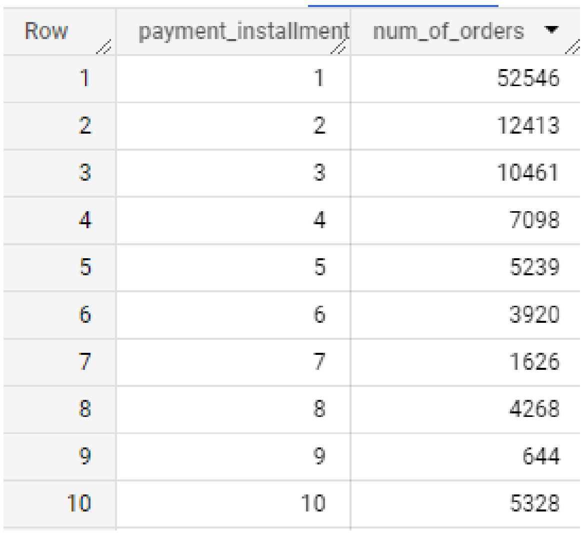
v. **Assumptions:** NA

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

i. **Query:**

```
SELECT payment_installments, COUNT(order_id) AS num_of_orders
FROM `target.payments`
WHERE payment_installments >= 1
GROUP BY payment_installments
ORDER BY payment_installments
```

ii. **Result:**



Row	payment_installment	num_of_orders
1	1	52546
2	2	12413
3	3	10461
4	4	7098
5	5	5239
6	6	3920
7	7	1626
8	8	4268
9	9	644
10	10	5328

iii. **Insights:** Around 50% of the orders placed have at least one installment paid.

iv. **Recommendations:** NA

v. **Assumptions:** NA