Business Case: Target SQL

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

   - **Data type of all columns in the "customers" table.**

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

Query results

JOB INFORMATION | RESULTS | CHART | PREVIEW | JSON

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

   - **Get the time range between which the orders were placed.**

```
SELECT
MIN(order_purchase_timestamp) AS start_time,
MAX(order_purchase_timestamp) AS end_time
FROM
`Target.orders` ;
```

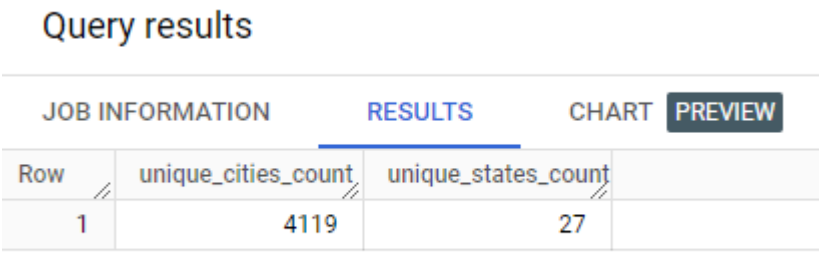
Query results

JOB INFORMATION | RESULTS | CHART | PREVIEW | JSON

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

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

```
SELECT
COUNT(DISTINCT c.customer_city) AS unique_cities_count,
COUNT(DISTINCT c.customer_state) AS unique_states_count
FROM `Target.customers` c
JOIN `Target.orders` o ON
c.customer_id = o.customer_id
WHERE
o.order_purchase_timestamp BETWEEN TIMESTAMP("2016-09-04") AND
TIMESTAMP("2018-10-17");
```



2. **In-depth Exploration:**

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

```
WITH OrderYears AS (
SELECT
EXTRACT(YEAR FROM order_purchase_timestamp) AS order_year,
COUNT(DISTINCT order_id) AS total_orders
FROM
`Target.orders`
GROUP BY
order_year
ORDER BY
order_year
)SELECT
order_year,
total_orders,
CASE
WHEN total_orders > LAG(total_orders) OVER (ORDER BY order_year)
THEN 'Increasing'
WHEN total_orders = LAG(total_orders) OVER (ORDER BY order_year)
THEN  'Stable'
ELSE 'Decreasing'
END AS trend
FROM
OrderYears
ORDER BY
order_year;
```

Query results

JOB INFORMATION | RESULTS | CHART | PREVIEW | JSON | EX

Row	order_year	total_orders	trend
1	2016	329	Decreasing
2	2017	45101	Increasing
3	2018	54011	Increasing

INSIGHTS:

- ➔ The order_purchase_timestamp from the Target.orders table is used by the query to efficiently segregate and count the unique orders placed each year.
- ➔ We can see that from 2016 the order kept increasing in 2017 and again in 2018 the total number of orders increased. There is a growing trend here.

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

```
WITH MonthlyOrders AS (
SELECT
EXTRACT(YEAR FROM order_purchase_timestamp) AS order_year,
EXTRACT(MONTH FROM order_purchase_timestamp) AS order_month,
COUNT(DISTINCT order_id) AS total_orders
FROM
`Target.orders`
GROUP BY
order_year, order_month
ORDER BY
order_year, order_month
)

SELECT
order_year,
order_month,
total_orders,
LAG(total_orders) OVER (ORDER BY order_year, order_month)
AS previous_month_orders,
CASE
WHEN total_orders > LAG(total_orders) OVER (ORDER BY
order_year, order_month) THEN 'Increasing'
WHEN total_orders < LAG(total_orders) OVER (ORDER BY
order_year, order_month) THEN 'Decreasing'
ELSE 'Stable'
END AS trend
FROM
MonthlyOrders
ORDER BY
order_year, order_month;
```

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

Row	order_year	order_month	total_orders	previous_month_orde	trend
1	2016	9	4	*null*	Stable
2	2016	10	324	4	Increasing
3	2016	12	1	324	Decreasing
4	2017	1	800	1	Increasing
5	2017	2	1780	800	Increasing
6	2017	3	2682	1780	Increasing
7	2017	4	2404	2682	Decreasing
8	2017	5	3700	2404	Increasing
9	2017	6	3245	3700	Decreasing
10	2017	7	4026	3245	Increasing
11	2017	8	4331	4026	Increasing
12	2017	9	4285	4331	Decreasing

INSIGHTS:

- ➔ The order_purchase_timestamp from the Target.orders table is used by the query to efficiently segregate and count the unique orders placed each month and determine whether the no. of orders is increasing, decreasing or stable from the previous month.
- ➔ There is a variable trend here where orders keep increasing, decreasing.

RECOMMENDATIONS:

- ➔ To encourage purchases and increase customer traffic during peak and trough months, implement customized promotional techniques, discounts, or offers based on the seasonality.
- ➔ Improve customer loyalty plans, personalized recommendations, and customer interaction tactics to keep customers and promote year-round repeat business.

- **During what time of the day, do the Brazilian customers mostly place their orders? (Dawn, Morning, Afternoon or Night)**
  - o 0-6 hrs: Dawn
  - o 7-12 hrs: Mornings
  - o 13-18 hrs: Afternoon
  - o 19-23 hrs: 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 time_of_day,
COUNT(o.order_id) AS order_count
FROM
`Target.orders` o  JOIN `Target.customers` c ON o.customer_id = c.customer_id
WHERE
c.customer_state = 'DF'
GROUP BY
time_of_day
ORDER BY
order_count DESC;
```

Query results

JOB INFORMATION | RESULTS | CHART | PREVIEW | JSON

Row	time_of_day	order_count
1	Afternoon	820
2	Night	614
3	Morning	608
4	Dawn	98

INSIGHTS:

- ➔ This insight helps understand customer behaviour, preferences, and daily routines in terms of shopping activities.
- ➔ As we can see the order count is more in Afternoon, the people tend to buy more, compared to morning and night where the trend is the same. At dawn there is a smaller number of orders.

RECOMMENDATIONS:

- ➔ To optimize reach, engagement, and conversion rates among customers, it is recommended to launch focused marketing campaigns, promotions, and advertising activities during peak time windows. Afternoon in this case.

➔ Improve the online shopping experience, website performance and customer support availability during peak hours to facilitate seamless transactions, reduce card failures, and drive repeat purchases.

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

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

```
WITH MonthlyOrdersBrazil AS (

SELECT
c.customer_state,
EXTRACT(YEAR FROM o.order_purchase_timestamp) AS order_year,
EXTRACT(MONTH FROM o.order_purchase_timestamp) AS order_month,
COUNT(DISTINCT o.order_id) AS total_orders
FROM
`Target.orders` AS o
JOIN
`Target.customers` AS c
ON
o.customer_id = c.customer_id
WHERE
c.customer_state = 'DF'
GROUP BY
c.customer_state,
order_year,
order_month
ORDER BY
c.customer_state,
order_year,
order_month
)
SELECT
customer_state,
order_year,
order_month,
total_orders
FROM
MonthlyOrdersBrazil
ORDER BY
customer_state,
order_year,
order_month;
```

INSIGHTS:

➔ From the below output we see that there is an increase in no. of orders placed in Brazil during 2017, though there was a slight decrease in April, for the rest of the months we can see an increase in trend of no. of orders placed. During the last November and December, we saw a decrease in sales.

➔ In the year 2018, even in the initial Two months, there was an increased trend, we can see the no. of orders placed moved between 145-150 a month.

RECOMMENDATIONS:

➔ Keep an eye on, evaluate, and use data-driven insights to guide strategic choices, maximize operational effectiveness, and match your company's plans to changing consumer tastes, market trends, and competitive dynamics in Brazil's e-commerce environment.

➔ To maintain current clients, promote recurring business, and cultivate enduring connections within the designated states, improve customer engagement tactics, loyalty initiatives, and customized experiences.

Query results

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

Row	customer_state	order_year	order_month	total_orders
1	DF	2016	10	6
2	DF	2017	1	13
3	DF	2017	2	24
4	DF	2017	3	57
5	DF	2017	4	35
6	DF	2017	5	64
7	DF	2017	6	70
8	DF	2017	7	77
9	DF	2017	8	87
10	DF	2017	9	97
11	DF	2017	10	98
12	DF	2017	11	168

- **How are the customers distributed across all the states?**

```
SELECT
customer_state,
COUNT(DISTINCT customer_id) AS total_customers,
ROUND(COUNT(DISTINCT customer_id) / (SELECT COUNT(DISTINCT customer_id)
```

```
FROM `Target.customers`) * 100, 2) AS percentage_of_total_customers
FROM
`Target.customers`
GROUP BY
customer_state
ORDER BY
total_customers DESC;
```

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

Row	customer_state	total_customers	percentage_of_total_
1	SP	41746	41.98
2	RJ	12852	12.92
3	MG	11635	11.7
4	RS	5466	5.5
5	PR	5045	5.07
6	SC	3637	3.66
7	BA	3380	3.4
8	DF	2140	2.15
9	ES	2033	2.04
10	GO	2020	2.03
11	PE	1652	1.66
12	CE	1336	1.34

INSIGHT:

➔ Here we see that country state SP has the most number of customers with a percentage total of 41.98, total customers of SP is 41,746. The country state RR has least no. of customers with 0.05 percentage, total customers of RR is 46.

➔ The total number of customers is 99,441.

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

    - **Get the % increase in the cost of orders from year 2017 to 2018 (include months between Jan to Aug only).**
      You can use the "payment value" column in the payments table to get the cost of orders.

```
WITH MonthlyPayments AS (
SELECT
EXTRACT(MONTH FROM o.order_purchase_timestamp) AS payment_month,
EXTRACT(YEAR FROM o.order_purchase_timestamp) AS payment_year,
SUM(p.payment_value) AS total_payment_value
```

```sql
 FROM
`Target.payments` AS p
 JOIN
`Target.orders` AS o
 ON
 p.order_id = o.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
 GROUP BY
 payment_year, payment_month
  )

 SELECT
 payment_month,
 2018 AS current_year,
 2017 AS previous_year,
 current_year_total_payment,
 previous_year_total_payment,
 CASE
 WHEN previous_year_total_payment != 0 THEN
 ((current_year_total_payment - previous_year_total_payment) /
 previous_year_total_payment) * 100
 ELSE
 0
 END AS percentage_increase
 FROM (
 SELECT
 payment_month,
 SUM(CASE WHEN payment_year = 2018 THEN total_payment_value ELSE 0
 END) AS current_year_total_payment,
 SUM(CASE WHEN payment_year = 2017 THEN total_payment_value ELSE 0
 END) AS previous_year_total_payment
 FROM
 MonthlyPayments
 GROUP BY
 payment_month
 ) AS t
 ORDER BY
 payment_month;
```

Query results

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

Row	payment_month	current_year	previous_year	current_year_total_pa	previous_year_total_	percentage_increase
1	1	2018	2017	1115004.180000...	138488.0399999...	705.1266954171...
2	2	2018	2017	992463.3400000...	291908.0099999...	239.9918145445...
3	3	2018	2017	1159652.119999...	449863.6000000...	157.7786066709...
4	4	2018	2017	1160785.480000...	417788.0300000...	177.8407701149...
5	5	2018	2017	1153982.150000...	592918.8200000...	94.62734375677...
6	6	2018	2017	1023880.499999...	511276.3800000...	100.2596912456...
7	7	2018	2017	1066540.750000...	592382.9200000...	80.04245463390...
8	8	2018	2017	1022425.319999...	674396.3200000...	51.60600520477...

INSIGHT:

➔ Comparing the payments made in total (payment_value) for the months of January through August in the respective years 2017 and 2018. This comparison helps to clarify the rise or fall in payments.

➔ As we see, the percentage increase in January from 2017 to 2018 is the highest, compared to all other months. When compared to 2017 the percentage increase shows an overall decreasing trend.

RECOMMENDATIONS:

➔ To discover possible obstacles, such as market saturation, investigate months where payments increased by a small proportion. Then, use techniques like boosting product availability, price strategies, or marketing campaigns to boost growth and address underlying problems.

➔ By providing individualized experiences, value-added services, and support throughout the payment process, you can cultivate client relationships, loyalty, and trust. To engage customers, address their concerns, and create enduring connections that promote loyalty, referrals, and revenue growth, put in place customer feedback methods, reward programs, and communication techniques.

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

```
SELECT
c.customer_state,
ROUND(SUM(oi.price), 2) AS total_order_price_per_state,
ROUND(AVG(oi.price), 2) AS avg_order_price_per_state
FROM
`Target.orders` AS o
JOIN
`Target.customers` AS c
```

```
ON
o.customer_id = c.customer_id
JOIN
`Target.order_items` AS oi
ON
o.order_id = oi.order_id
GROUP BY
c.customer_state
ORDER BY
total_order_price_per_state DESC;
```

JOB INFORMATION | RESULTS | CHART PREVIEW | JSON

Row	customer_state	total_order_price_per	avg_order_price_per
1	SP	5202955.05	109.65
2	RJ	1824092.67	125.12
3	MG	1585308.03	120.75
4	RS	750304.02	120.34
5	PR	683083.76	119.0
6	SC	520553.34	124.65
7	BA	511349.99	134.6
8	DF	302603.94	125.77
9	GO	294591.95	126.27
10	ES	275037.31	121.91

INSIGHTS:

- ➔ By connecting the orders, customers, and order_items columns, the SQL query yields a breakdown of the average order price per state as well as the total order price. From this we can analyse the economic value generated from each state in terms of total sales and average transaction value.

- ➔ Can identify states that contribute significantly to overall revenue and assess the purchasing power or consumer behaviour patterns within each region. Variations in average order values between states could be different customer demographics, lifestyle choices, or factors impacting consumer spending power.

RECOMMENDATIONS:

- ➔ To create stable marketing campaigns, promotional offers that appeal to the requirements, tastes, and purchasing habits of consumers in these areas, concentrate on the states with higher total order prices. Develop tactics that are tailored to maximize income production from high-value states and to take use of the economic potential.

➔ Evaluate the average order price across states to refine pricing strategies, adjust product assortments, and optimize inventory management. Identify opportunities to introduce premium products, bundle offerings.

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

```
SELECT
c.customer_state,
ROUND(SUM(oi.freight_value), 2) AS total_freight_per_state,
ROUND(AVG(oi.freight_value), 2) AS avg_freight_per_state
FROM
`Target.orders` AS o
JOIN
`Target.customers` AS c
ON
o.customer_id = c.customer_id
JOIN
`Target.order_items` AS oi
ON
o.order_id = oi.order_id
GROUP BY
c.customer_state
ORDER BY
total_freight_per_state DESC;
```

INSIGHTS:

➔ This analysis allows firms to find areas with greater transportation costs by highlighting variances in shipping expenses between geographies. By examining the distribution of freight costs per state, we can assess the impact on efficiency, profitability, and customer satisfaction.

RECOMMENDATIONS:

➔ Assess the states that have greater average and total freight expenses to maximize logistics. To reduce transportation costs and boost cost-efficiency, look into ways to combine shipments, make use of regional distribution hubs.

➔ Develop ties with shippers, logistics companies, and other supply chain participants to work together on creative projects, pooled resources, and shipping options. Give priority to client happiness, communication, by offering precise shipping estimates, live tracking, and active notifications all the way through the delivery process.

JOB INFORMATION | RESULTS | CHART | PREVIEW | JSON

Row	customer_state	total_freight_per_sta	avg_freight_per_state
1	SP	718723.07	15.15
2	RJ	305589.31	20.96
3	MG	270853.46	20.63
4	RS	135522.74	21.74
5	PR	117851.68	20.53
6	BA	100156.68	26.36
7	SC	89660.26	21.47
8	PE	59449.66	32.92
9	GO	53114.98	22.77
10	DF	50625.5	21.04
11	ES	49764.6	22.06
12	CE	48351.59	32.71

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

- **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.

  You can calculate the delivery time and the difference between the estimated & actual delivery date using the given formula:
    - **time_to_deliver = order_delivered_customer_date - order_purchase_timestamp**
    - **diff_estimated_delivery = order_delivered_customer_date - order_estimated_delivery_date**

```
SELECT
order_id,
DATE_DIFF(order_delivered_customer_date, order_purchase_timestamp, DAY)
AS delivery_time,
ABS(DATE_DIFF(order_estimated_delivery_date, order_delivered_customer_date
,DAY)) AS diff_estimated_delivery
FROM
`Target.orders`
ORDER BY delivery_time desc;
```

JOB INFORMATION | RESULTS | CHART PREVIEW | JSON | E

Row	order_id	delivery_time	diff_estimated_delive
1	ca07593549f1816d26a572e06...	209	181
2	1b3190b2dfa9d789e1f14c05b...	208	188
3	440d0d17af552815d15a9e41a...	195	165
4	0f4519c5f1c541ddec9f21b3bd...	194	161
5	285ab9426d6982034523a855f...	194	166
6	2fb597c2f772eca01b1f5c561b...	194	155
7	47b40429ed8cce3aee9199792...	191	175
8	2fe324febf907e3ea3f2aa9650...	189	167
9	2d7561026d542c8dbd8f0daea...	188	159
10	437222e3fd1b07396f1d9ba8c...	187	144
11	c27815f7e3dd0b926b5855262...	187	162
12	dfe5f68118c2576143240b8d7...	186	153

INSIGHTS:

➔ The query evaluates the accuracy and reliability of the delivery timeline provided to customers. This comparison enables businesses to assess their ability to meet customer expectations, manage delivery commitments, and address potential discrepancies or delays in the fulfillment process.

RECOMMENDATIONS:

➔ Priority on customer communication, transparency, and happiness by offering precise delivery estimates, proactive notifications, and real-time tracking tools. By delivering orders on time and consistently, you can manage expectations, handle complaints, and build customer trust.

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

```
WITH StateFreightAvg AS (
SELECT
c.customer_state,
AVG(oi.freight_value) AS avg_freight_value
FROM
`Target.orders` AS o
JOIN
`Target.customers` AS c
 ON
```

```
 o.customer_id = c.customer_id
 JOIN
 `Target.order_items` AS oi
 ON
 o.order_id = oi.order_id
 GROUP BY
 c.customer_state
)

(SELECT
customer_state,
avg_freight_value
FROM
StateFreightAvg
ORDER BY
avg_freight_value DESC
LIMIT 5)

UNION ALL

(SELECT
 customer_state,
 avg_freight_value
FROM
StateFreightAvg
ORDER BY
avg_freight_value ASC
LIMIT 5);
```

INSIGHTS:

- ➔ The query identifies the top 5 states with the highest average freight values and the bottom 5 states with the lowest average freight values . The results are combined using the UNION ALL operator to provide insights into the regional variations in shipping costs, logistics efficiency, and operational performance across states.

RECOMMENDATIONS:

- ➔ Foster collaboration and alignment among cross-functional teams, departments, and stakeholders to optimize shipping strategies, reduce operational costs, and boost customer satisfaction across high and low-ranking states.
- ➔ To drive continuous improvement, innovation, and excellence inside the business while establishing a culture of accountability, transparency, and

customer-centricity, encourage knowledge sharing, adoption of best practices, and collaborative problem-solving.

JOB INFORMATION | RESULTS | CHART PREVIEW

Row	customer_state	avg_freight_value
1	RR	42.98442307692...
2	PB	42.72380398671...
3	RO	41.06971223021...
4	AC	40.07336956521...
5	PI	39.14797047970...
6	SP	15.14727539041...
7	PR	20.53165156794...
8	MG	20.63016680630...
9	RJ	20.96092393168...
10	DF	21.04135494596...

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

```
WITH DeliveryData AS (
SELECT
 c.customer_state,
 AVG(DATE_DIFF(order_delivered_customer_date, order_purchase_timestamp,
 DAY)) AS avg_delivery_time
 FROM
 `Target.orders` AS o
 JOIN
 `Target.customers` AS c
  ON
  o.customer_id = c.customer_id
  GROUP BY
   c.customer_state
 )

 ( SELECT
   customer_state,
   avg_delivery_time
   FROM
   DeliveryData
   ORDER BY
   avg_delivery_time DESC
   LIMIT 5 )
```

```
UNION ALL

( SELECT
  customer_state,
  avg_delivery_time
  FROM
  DeliveryData
  ORDER BY
  avg_delivery_time ASC
  LIMIT 5 );
```

INSIGHTS:

➔ The average delivery times in each state vary significantly among regions, according to the SQL query. These discrepancies may be due to variations in supply chain operations, transportation networks, and logistics efficiency. Knowing which states have the longest and shortest average delivery times will help you understand the variables that affect service quality, customer happiness, and delivery speed in different areas.

RECOMMENDATIONS:

➔ Use real-time tracking features, predictive modeling tools, and sophisticated analytics to improve routing, resource allocation, and strategy adaptation based on real-time data, consumer preferences, and market dynamics across several areas.

JOB INFORMATION | RESULTS | CHART | PREVIEW

Row	customer_state	avg_delivery_time
1	RR	28.97560975609...
2	AP	26.73134328358...
3	AM	25.98620689655...
4	AL	24.04030226700...
5	PA	23.31606765327...
6	SP	8.298061489072...
7	PR	11.52671135486...
8	MG	11.54381329810...
9	DF	12.50913461538...
10	SC	14.47956019171...

- **Find out the top 5 states where the order delivery is really fast as compared to the estimated date of delivery.**
  You can use the difference between the averages of actual & estimated delivery date to figure out how fast the delivery was for each state.

```
WITH DeliverySpeedData AS (
SELECT
c.customer_state,
AVG(DATE_DIFF(order_delivered_customer_date, order_estimated_delivery_date,
DAY)) AS avg_delivery_speed,
RANK() OVER (ORDER BY AVG(DATE_DIFF(order_delivered_customer_date,
order_estimated_delivery_date, DAY)) ASC) AS rank_speed
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
 GROUP BY
 c.customer_state
)
 SELECT
 customer_state,
 avg_delivery_speed
 FROM
 DeliverySpeedData
 WHERE
 rank_speed <= 5
  ORDER BY
 avg_delivery_speed ASC;
```

INSIGHTS:

➔ The SQL query reveals the top 5 states where orders are delivered considerably quicker than the predicted delivery date, based on the negative average difference between the actual and estimated delivery dates. This analysis focuses on areas with effective fulfillment procedures, efficient transportation networks, and efficient logistics that lead to quicker delivery times and higher customer satisfaction.

JOB INFORMATION | RESULTS | CHART | PREVIEW

Row	customer_state	avg_delivery_speed
1	AC	-19.7625000000...
2	RO	-19.1316872427...
3	AP	-18.7313432835...
4	AM	-18.6068965517...
5	RR	-16.4146341463...

6. **Analysis based on the payments**

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

```
SELECT
FORMAT_TIMESTAMP('%Y-%m', o.order_purchase_timestamp) AS order_month,
p.payment_type,
COUNT(DISTINCT o.order_id) AS num_orders
FROM
`Target.orders` AS o
JOIN
`Target.payments` AS p
ON
o.order_id = p.order_id
GROUP BY
order_month,
p.payment_type
ORDER BY
order_month,
p.payment_type;
```

INSIGHTS:

➔ Through the analysis of monthly order breakdowns by payment type, you can spot seasonal variations, trends, and patterns that affect customer payment behavior, purchase choices, and transactional activity all year long. This kind of information enables businesses to plan ahead, maximize inventory, and coordinate marketing campaigns with holidays, peak seasons, and other events that affect consumer payment preferences and shopping habits.

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

Row	order_month	payment_type	num_orders
1	2016-09	credit_card	3
2	2016-10	UPI	63
3	2016-10	credit_card	253
4	2016-10	debit_card	2
5	2016-10	voucher	11
6	2016-12	credit_card	1
7	2017-01	UPI	197
8	2017-01	credit_card	582
9	2017-01	debit_card	9
10	2017-01	voucher	33
11	2017-02	UPI	398
12	2017-02	credit_card	1347

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

```
SELECT
payment_installments,
COUNT(DISTINCT order_id) AS num_orders
FROM
`Target.payments`
GROUP BY
payment_installments
ORDER BY
 payment_installments;
```

JOB INFORMATION | RESULTS | CHART

Row	payment_installment	num_orders
1	0	2
2	1	49060
3	2	12389
4	3	10443
5	4	7088
6	5	5234
7	6	3916
8	7	1623
9	8	4253
10	9	644

INSIGHTS:

- ➔ The analysis highlights the distribution of orders across different payment installment options, enabling organizations to identify the most commonly chosen installment plans, payment terms, and financing options utilized by customers within the marketplace.
- ➔ This insight allows businesses to tailor their product offerings, pricing strategies, and promotional campaigns to meet the diverse needs and preferences of customers seeking flexible payment solutions and budget-friendly alternatives.