Business Case: Target SQL

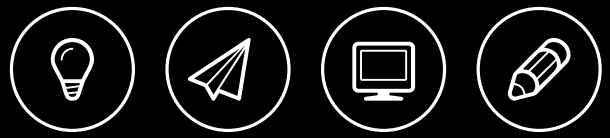
DATE:22-06-2023

CONTEXT

Target is a globally renowned brand and a prominent retailer in the United States. Target makes itself a preferred shopping destination by offering outstanding value, inspiration, innovation and an exceptional guest experience that no other retailer can deliver.

This particular business case focuses on the operations of Target in Brazil and provides insightful information about 100,000 orders placed between 2016 and 2018. The dataset offers a comprehensive view of various dimensions including the order status, price, payment and freight performance, customer location, product attributes, and customer reviews.

By analyzing this extensive dataset, it becomes possible to gain valuable insights into Target's operations in Brazil. The information can shed light on various aspects of the business, such as order processing, pricing strategies, payment and shipping efficiency, customer demographics, product characteristics, and customer satisfaction levels.



Problem Statement:
Assuming you are a data analyst/ scientist at Target, you have been assigned the task of analyzing the given dataset to extract valuable insights and provide actionable recommendations.

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

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

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

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

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

PERSONAL HISTORY PROJECT HISTORY

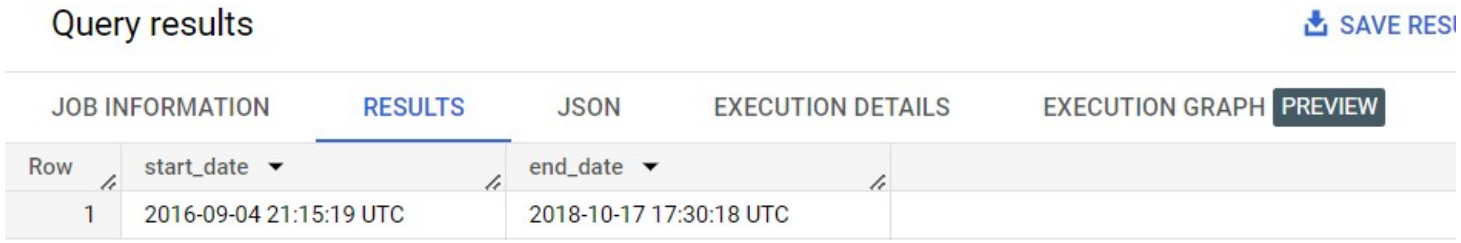
Insights:

1. The customer_id and customer_unique_id columns provide a unique identity for each customer.
2. The customer_zip_code_prefix column stores numerical values representing the zip code.
3. The customer_city and customer_state columns store data representing the city and state associated with each customer.
4. The data types of the columns indicate the kind of data stored in each column, facilitating appropriate data manipulation and analysis.

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

```
SELECT

        MIN(order_purchase_timestamp ) as start_date,
        MAX(order_purchase_timestamp ) as end_date
FROM
        `target.orders`
```


Query results

SAVE RES

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

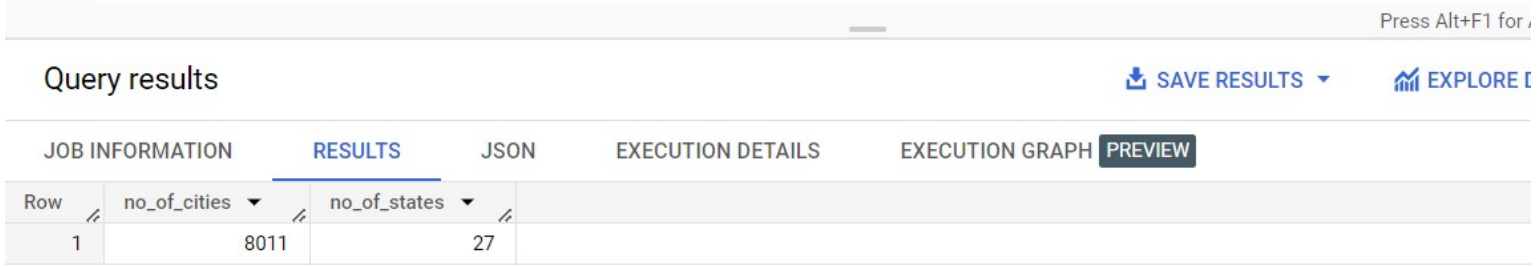
Row	start_date	end_date
1	2016-09-04 21:15:19 UTC	2018-10-17 17:30:18 UTC

Insights:

1. The Start_date is September 4, 2016, at 21:15:19 UTC.
2. The End_date is October 17, 2018, at 17:30:18 UTC.
3. The duration between the start and end dates is approximately 2 years, 1 month, 13 days

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

```
SELECT
  COUNT(DISTINCT geolocation_city) AS no_of_cities,
  COUNT(DISTINCT geolocation_state) AS no_of_states
FROM
  `target.geolocation`;
```



Insights:

1. The "no_of_cities" column has a value of 8011.
2. The "no_of_states" column has a value of 27.
3. These values indicate the number of cities and states represented in the data or context provided

2. In-depth Exploration:

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

```
SELECT
  EXTRACT(year FROM order_purchase_timestamp) AS year,
  EXTRACT(month FROM order_purchase_timestamp) AS month,
  COUNT(*) AS order_count
FROM
  `target.orders`
GROUP BY
  year,
  month
ORDER BY
  year, month
```

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

Row	year	month	order_count
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

Insights:

1. Order counts varied significantly from month to month. The highest order count was observed in July 2017 with 4,026 orders, while the lowest count was in December 2016 with only 1 order.
2. The year 2017 had a higher order count compared to 2016. This suggests that business Has increased and there was high demand from the customers

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

```
SELECT
        EXTRACT(month FROM order_purchase_timestamp) AS month,
        COUNT(*) AS order_count
FROM
      `target.orders`
GROUP BY
      month
ORDER BY
       month
```

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

Row	month	order_count
1	1	8069
2	2	8508
3	3	9893
4	4	9343
5	5	10573
6	6	9412
7	7	10318
8	8	10843
9	9	4305
10	10	4959
11	11	7544
12	12	5674

Insights:

1. The order counts generally show a fluctuating pattern throughout the year.
2. The months with the highest order counts are May (10,573 orders) and August (10,843 orders), indicating potential periods of increased sales or customer activity.

3. The months with the lowest order counts are September (4,305 orders) and October (4,959 orders), suggesting a decrease in demand during those months compared to the rest of the year.

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

```
SELECT
        a.time_category,
        COUNT(a.count_order) AS count
FROM (
          SELECT
          EXTRACT(HOUR FROM order_purchase_timestamp) AS hour,
          COUNT(order_id) AS count_order,
            CASE
            WHEN EXTRACT(HOUR FROM order_purchase_timestamp) >= 0 AND EXTRACT(HOUR FROM order_purchase_timestamp) < 6 THEN 'Dawn'
            WHEN EXTRACT(HOUR FROM order_purchase_timestamp) >= 6 AND EXTRACT(HOUR FROM order_purchase_timestamp) < 12 THEN 'Mornings'
            WHEN EXTRACT(HOUR FROM order_purchase_timestamp) >= 12 AND EXTRACT(HOUR FROM order_purchase_timestamp) < 18 THEN
        'Afternoon'
              WHEN EXTRACT(HOUR FROM order_purchase_timestamp) >= 18 AND EXTRACT(HOUR FROM order_purchase_timestamp) <= 23 THEN 'Night'
             END AS time_category
          FROM `target.orders`
          GROUP BY order_purchase_timestamp
        ) AS a
GROUP by a.time_category
ORDER by count DESC;
```

Insights:

1. The most frequent time category for orders is "Afternoon" with a count of 38,129. This suggests that a significant portion of orders occurs during the afternoon hours.
2. The second most common time category is "Night" with a count of 33,903. This indicates a substantial number of orders being placed during the nighttime.
3. "Mornings" is the third most frequent time category, with a count of 22,119. And the The least common time category for orders is "Dawn" with a count of 4,724.

Query results

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

Row	time_category	count
1	Afternoon	38129
2	Night	33903
3	Mornings	22119
4	Dawn	4724

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

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

```
SELECT
  c.customer_state,
  EXTRACT(MONTH FROM o.order_purchase_timestamp) AS Month,
  EXTRACT(YEAR FROM o.order_purchase_timestamp) AS Year,
  COUNT(*) AS Total_order
FROM
  `target.customers` AS c
JOIN
  `target.orders` AS o
ON
  c.customer_id = o.customer_id
GROUP BY
  Month,
  c.customer_state,
  Year
ORDER  BY
  Month, c.customer_state,year
```

Query results

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

Row	customer_state	Month	Year	Total_order
1	AC	1	2017	2
2	AC	1	2018	6
3	AL	1	2017	2
4	AL	1	2018	37
5	AM	1	2018	12
6	AP	1	2018	11
7	BA	1	2017	25
8	BA	1	2018	239
9	CE	1	2017	9
10	CE	1	2018	90
11	DF	1	2017	13
12	DF	1	2018	138

Insights:
Looking at the above results from the snipshot , u can come to the conclusion that :
The orders have slightly increased in the year 2018 compared to 2017 in all states.
For example

1. The state with the lowest total order count in January 2017 is "AC" with 2 orders.
2. In January 2018, the state with the highest total order count is "BA" with 239 orders.
3. The state "AL" shows an increase in total order count from January 2017 (2 orders) to January 2018 (37 orders).
4. "CE" also exhibits a significant increase in total order count from January 2017 (9 orders) to January 2018 (90 orders).

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

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

```
SELECT
  customer_state,
  COUNT(DISTINCT customer_unique_id) AS Total_unique_id
FROM
  `target.customers`
GROUP BY
  customer_state
```

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

Row	customer_state	Total_unique_id
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
11	PE	1609
12	PB	519

Insights:

1. The state with the highest total unique IDs is **SP** with 40,302. This suggests a high number of unique customers.
2. The state with the second-highest total unique IDs is **MG** with 11,259. This indicates a significant number of unique customers
3. "RJ" follows with 12,384 total unique IDs, indicating a considerable customer presence
4. Other states like **RS**, **SC**, and **BA** also have a substantial number of total unique IDs, suggesting a significant customer base in those regions.
5. States like **CE**,**GO**, and **PE** have a moderate number of total unique IDs, indicating a moderate customer presence in those areas.
6. States like **MA, PB,** and **RS** have a relatively lower number of total unique IDs, indicating a smaller customer base or lower customer activity in those regions

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

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

```
SELECT
        c.customer_state,
          ROUND (SUM(op.price),2) as Total_price,
          ROUND (AVG(op.price),2) as Avg_price
FROM
        `target.order_items` as op
JOIN
          `target.orders` as o
          ON op.order_id = o.order_id
JOIN
          `target.customers` as c
          ON o.customer_id = c.customer_id
GROUP BY
          c.customer_state
```

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

Row	customer_state	Total_price	Avg_price
1	SP	5202955.05	109.65
2	RJ	1824092.67	125.12
3	PR	683083.76	119.0
4	SC	520553.34	124.65
5	DF	302603.94	125.77
6	MG	1585308.03	120.75
7	PA	178947.81	165.69
8	BA	511349.99	134.6
9	GO	294591.95	126.27
10	RS	750304.02	120.34
11	TO	49621.74	157.53
12	AM	22356.84	135.5

Insights:

1. The state with the highest total price is **SP** with a **Total_price of 5,202,955.05.** This indicates a significant amount of sales or revenue generated from customers, with **AVG_price of 109.65**
2. **RJ** follows with a **Total_price** of 1,824,092.67, indicating substantial sales or revenue from customers ,with **AVG_price of 125.12**
3. States like **PR ,SC,and MG** also have notable total prices, suggesting a significant amount of sales or revenue generated from customers in those regions.
4. The state with the highest average price is **PA with an average price of 165.69.** This suggests that customers in **PA** tend to have higher individual transaction amounts compared to other states

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

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

```
SELECT
  c.customer_state,
  ROUND (SUM(op.freight_value),2) as Total_freight_value,
  ROUND (AVG(op.freight_value),2) as Avg_freight_value
FROM
  `target.order_items` as op
JOIN
  `target.orders` as o
  ON op.order_id = o.order_id
JOIN
  `target.customers` as c
  ON o.customer_id = c.customer_id
GROUP BY
  c.customer_state
```

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

Row	customer_state	Total_freight_value	Avg_freight_value
1	MT	29715.43	28.17
2	MA	31523.77	38.26
3	AL	15914.59	35.84
4	SP	718723.07	15.15
5	MG	270853.46	20.63
6	PE	59449.66	32.92
7	RJ	305589.31	20.96
8	DF	50625.5	21.04
9	RS	135522.74	21.74
10	SE	14111.47	36.65
11	PR	117851.68	20.53

Insights:

1. The state with the highest total freight value is **SP** with a **Total_freight_value of 718,723.07 and AVG_freight_value of 28.17** This indicates a significant amount of freight expenses associated with customers
2. MG follows with a **Total_freight_value** of **270,853.46**, **and AVG_freight_value of 28.17** indicating substantial freight costs related to customers

1. States like **RJ,RS, and DF** also have notable total freight values, suggesting significant freight expenses associated with customers in those regions.
2. The state with the highest average freight value is **MA** with an **AVG_freight_value** of **38.26.** This suggests that customers tend to have higher average freight costs compared to other states

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.
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_estimated_delivery_date - order_delivered_customer_date

```
SELECT
        order_id,
        order_purchase_timestamp,
        order_delivered_customer_date,
        order_estimated_delivery_date,
        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`
```

Query results

SAVE RESULTS EXPLORE DA

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

Row	order_id	order_purchase_timestamp	order_delivered_customer_date	order_estimated_delivery_date	time_to_deliver	diff_estimated_delivery
1	1950d777989f6a877539f5379...	2018-02-19 19:48:52 UTC	2018-03-21 22:03:51 UTC	2018-03-09 00:00:00 UTC	30	-12
2	2c45c33d2f9cb8ff8b1c86cc28...	2016-10-09 15:39:56 UTC	2016-11-09 14:53:50 UTC	2016-12-08 00:00:00 UTC	30	28
3	65d1e226dfaeb8cdc42f66542...	2016-10-03 21:01:41 UTC	2016-11-08 10:58:34 UTC	2016-11-25 00:00:00 UTC	35	16
4	635c894d068ac37e6e03dc54e...	2017-04-15 15:37:38 UTC	2017-05-16 14:49:55 UTC	2017-05-18 00:00:00 UTC	30	1
5	3b97562c3aee8bdedcb5c2e45...	2017-04-14 22:21:54 UTC	2017-05-17 10:52:15 UTC	2017-05-18 00:00:00 UTC	32	0
6	68f47f50f04c4cb6774570cfde...	2017-04-16 14:56:13 UTC	2017-05-16 09:07:47 UTC	2017-05-18 00:00:00 UTC	29	1
7	276e9ec344d3bf029ff83a161c...	2017-04-08 21:20:24 UTC	2017-05-22 14:11:31 UTC	2017-05-18 00:00:00 UTC	43	-4
8	54e1a3c2b97fb0809da548a59...	2017-04-11 19:49:45 UTC	2017-05-22 16:18:42 UTC	2017-05-18 00:00:00 UTC	40	-4
9	fd04fa4105ee8045f6a0139ca5...	2017-04-12 12:17:08 UTC	2017-05-19 13:44:52 UTC	2017-05-18 00:00:00 UTC	37	-1

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

```
SELECT
        c.customer_state,
        Round(AVG(ot.freight_value),2) AS avg_freight_value
FROM
        `target.customers` AS c
JOIN    `target.orders` AS o
ON      c.customer_id = o.customer_id
JOIN    `target.order_items` AS ot
ON      o.order_id = ot.order_id
WHERE
        ot.freight_value IS NOT NULL
GROUP BY
        c.customer_state
ORDER BY
        avg_freight_value desc
LIMIT 5
```

```
SELECT
        c.customer_state,
        Round(AVG(ot.freight_value),2) AS avg_freight_value
FROM
        `target.customers` AS c
JOIN    `target.orders` AS o
ON      c.customer_id = o.customer_id
JOIN    `target.order_items` AS ot
ON      o.order_id = ot.order_id
WHERE
        ot.freight_value IS NOT NULL
GROUP BY c.customer_state
ORDER BY avg_freight_value ASC
LIMIT 5
```

Query results

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

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

Query results

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

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

Insights:

1. The state with the highest average freight value is **RR** with an **AVG_freight_value** of **42.98.**
2. **PB** follows closely with an **AVG_freight_value of** of **42.72**.
3. States like **RO,AC, and PI** also have notable highest **AVG_freight_value**

4. The state with the lowest average freight value is **SP** with an **AVG_freight_value** of **15.15.**
5. **PR** follows with an lowest **AVG_freight_value** of **20.53.**
6. States like **MG,RJ,and DF** also have notable lowest **AVG_freight_value**

6. Aalysis based on the payments:

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

```
SELECT
  EXTRACT(month FROM o.order_purchase_timestamp) AS month,
  p.payment_type,
  COUNT(DISTINCT o.order_id) AS order_count
FROM
  `target.orders` AS o
  JOIN `target.payments` AS p ON o.order_id = p.order_id
GROUP BY
  month,
  p.payment_type
ORDER BY
  month,p.payment_type
```

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

Row	month	payment_type	order_count
1	1	UPI	1715
2	1	credit_card	6093
3	1	debit_card	118
4	1	voucher	337
5	2	UPI	1723
6	2	credit_card	6582
7	2	debit_card	82
8	2	voucher	288
9	3	UPI	1942
10	3	credit_card	7682
11	3	debit_card	109

Insights:

1. In the first month (January), the most popular payment type is **credit card with 6,093** orders. This is followed by , **UPI with 1,715** orders , **voucher with 337** orders, and **debit card with 118** orders.
2. In the second month (February), the order count for each payment type is as follows: **credit card - 6,582** orders **,UPI - 1,723 orders**, **debit card - 82 orders**, and **voucher - 288 orders.**
3. In the third month (March), the order count for each payment type is as follows: **credit card - 7,682** orders, **UPI - 1,942** orders, **debit card - 109**, and **voucher - 395.**

From this we can conclude that Credit Cards are the Most used payment type among all .

6. Aalysis based on the payments:

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

```
SELECT
  p.payment_installments,
  COUNT(DISTINCT o.order_id) AS order_count
FROM
  `target.orders` AS o
  JOIN `target.payments` AS p ON o.order_id = p.order_id
WHERE
  p.payment_value > 0
GROUP BY
  p.payment_installments
ORDER BY
  p.payment_installments
```

Query results

JOB INFORMATION | RESULTS | JSON

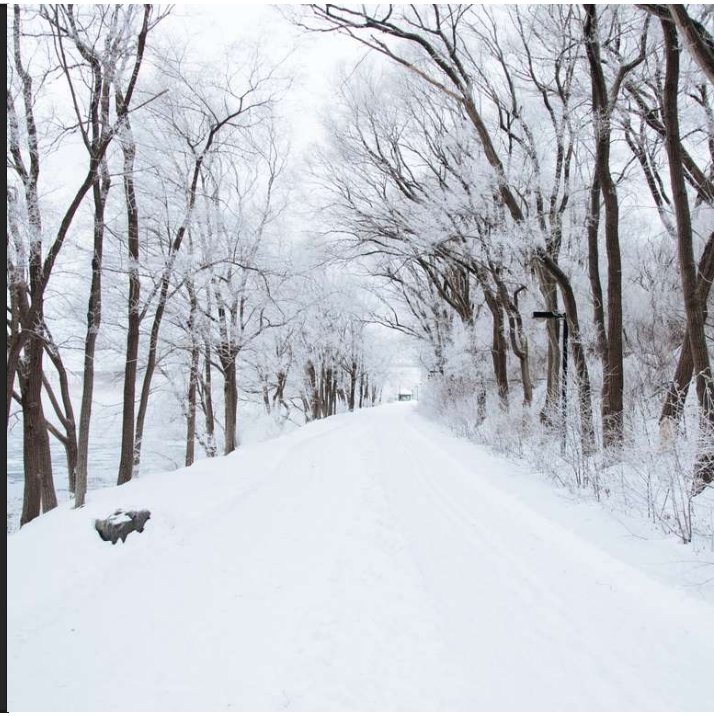
Row	payment_installment	order_count
1	0	2
2	1	49057
3	2	12389
4	3	10443
5	4	7088
6	5	5234
7	6	3916
8	7	1623
9	8	4253
10	9	644
11	10	5315
12	11	23
13	12	133
14	13	16

Insights:

1. The most common payment installment is option 1, with an order count of 49,057. This indicates that the majority of customers choose to pay for their orders in a single installment.

2. For orders with payment installments ranging from 2 to 9, the order counts gradually decrease as the number of installments increases. This suggests that fewer customers opt for installment-based payment plans as the number of installments increases.

OVERALL SUMMARY



The dataset provides insightful information on various aspects of orders, customers, and payments. It includes details such as monthly order counts by payment type, average freight values by customer state, time taken for order delivery, and order counts based on payment installments. The data highlights interesting trends, such as the popularity of certain payment methods, variations in freight costs across different states, and patterns in order delivery times.

THANK YOU

DATE：22-06-2023
DONE BY - KIRAN MJ