Introduction:
This report analyzes Target's e-commerce operations in Brazil between 2016 and 2018. It explores customer behavior, order trends, payments, freight, and delivery performance using SQL-based analysis. The goal is to extract actionable insights and provide recommendations for improving operations and customer satisfaction.

Business Case Study: Target Brazil

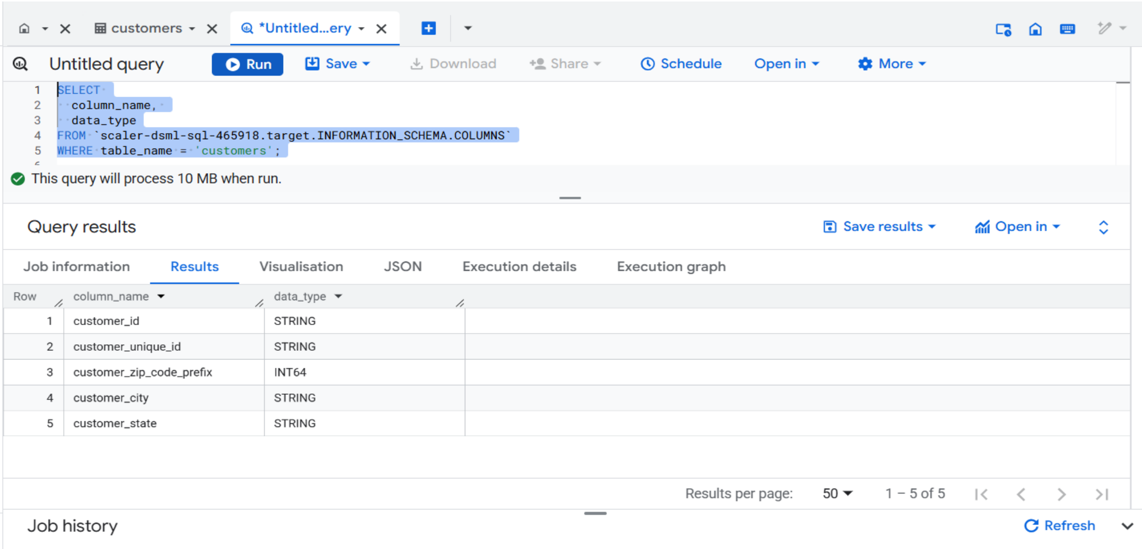
STUDENT NAME – AYUSH SAINI

BATCH- DSML_jul25_SQL_MWF

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 `scaler-dsml-sql-465918.target.INFORMATION_SCHEMA.COLUMNS`
WHERE table_name = 'customers';
```



```
SELECT
  column_name,
  data_type
FROM `scaler-dsml-sql-465918.target.INFORMATION_SCHEMA.COLUMNS`
WHERE table_name = 'customers';
```

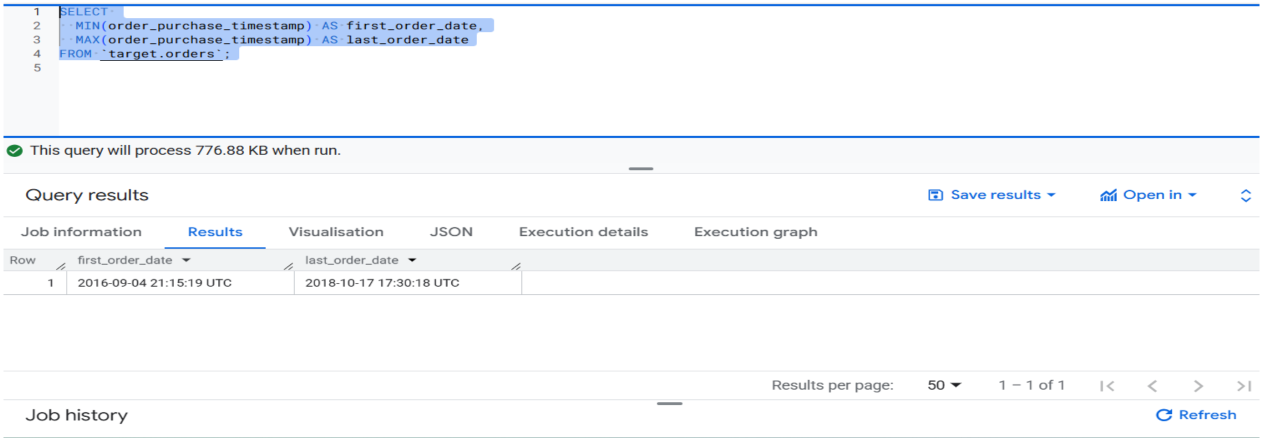
This query will process 10 MB when run.

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

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

```
 SELECT

MIN(order_purchase_timestamp) AS first_order_date,
MAX(order_purchase_timestamp) AS last_order_date
FROM `target.orders`
```



```
SELECT
  MIN(order_purchase_timestamp) AS first_order_date,
  MAX(order_purchase_timestamp) AS last_order_date
FROM `target.orders`;
```

This query will process 776.88 KB when run.

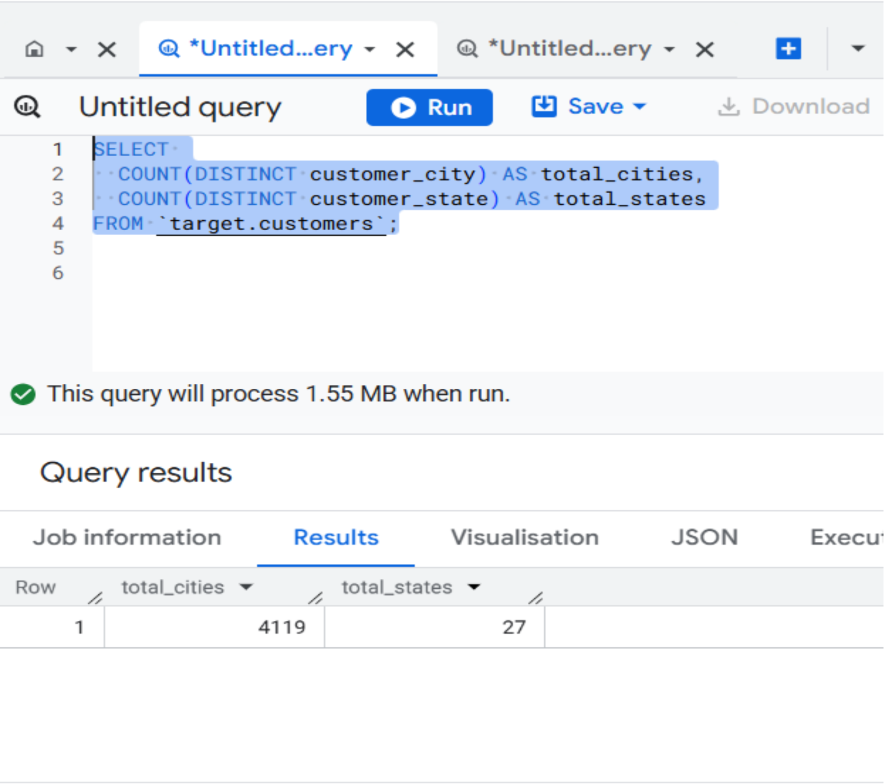
Query results

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

Results per page: 50 1 – 1 of 1

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

```
SELECT
COUNT(DISTINCT customer_city) AS total_cities,
COUNT(DISTINCT customer_state) AS total_states
FROM `target.customers`;
```



```
SELECT
  COUNT(DISTINCT customer_city) AS total_cities,
  COUNT(DISTINCT customer_state) AS total_states
FROM `target.customers`;
```

This query will process 1.55 MB when run.

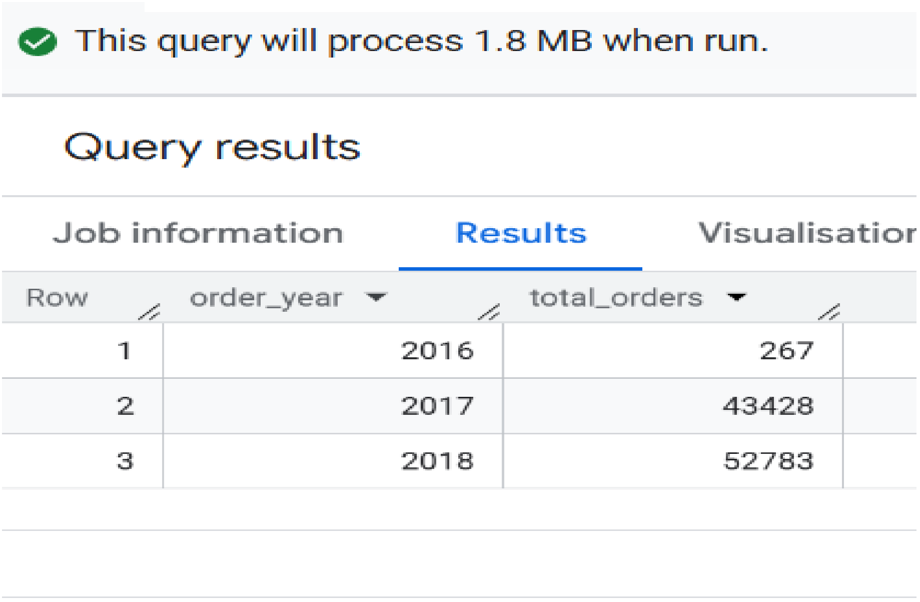
Query results

Row	total_cities	total_states
1	4119	27

In-depth Exploration

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

```
 SELECT
EXTRACT(YEAR FROM TIMESTAMP(order_purchase_timestamp )) AS order_year,
COUNT(*) AS total_orders
FROM target.orders
WHERE order_status = 'delivered'
GROUP BY order_year
ORDER BY order_year;
```



This query will process 1.8 MB when run.

Query results

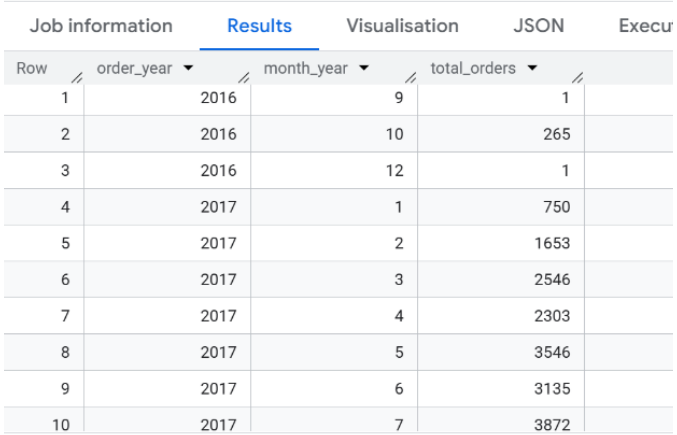
Row	order_year	total_orders
1	2016	267
2	2017	43428
3	2018	52783

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

Yes, I can see monthly seasonality in terms of the no of orders, as there are approx. no sales in month of September, December and October have very little sales in Year 2016.Then from January 2017 onwards there is increase in no of sales with constant growth.

```
SELECT
 EXTRACT(YEAR FROM TIMESTAMP(order_purchase_timestamp )) AS order_year,
 EXTRACT(MONTH FROM TIMESTAMP(order_purchase_timestamp)) AS month_year,
 COUNT(*) AS total_orders
 FROM target.orders
```

```
WHERE order_status = 'delivered'
GROUP BY order_year,month_year
ORDER BY order_year;
```



Row	order_year	month_year	total_orders
1	2016	9	1
2	2016	10	265
3	2016	12	1
4	2017	1	750
5	2017	2	1653
6	2017	3	2546
7	2017	4	2303
8	2017	5	3546
9	2017	6	3135
10	2017	7	3872

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

   a. 0-6 hrs : Dawn

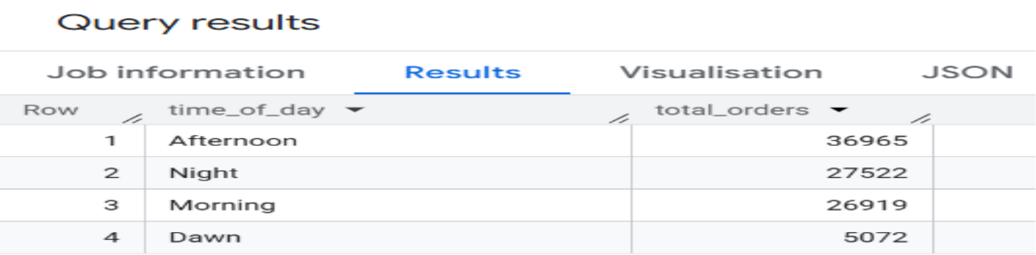
   b. 7-12 hrs : Mornings

   c. 13-18 hrs : Afternoon

   d. 19-23 hrs : Night

In Afternoon,Bazilian customers place most of their orders

```
SELECT
CASE
WHEN EXTRACT(HOUR FROM TIMESTAMP(order_purchase_timestamp) ) BETWEEN 0 AND 6
THEN 'Dawn'
WHEN EXTRACT(HOUR FROM TIMESTAMP(order_purchase_timestamp) ) BETWEEN 7 AND 12
THEN 'Morning'
WHEN EXTRACT(HOUR FROM TIMESTAMP(order_purchase_timestamp) ) BETWEEN 13 AND 18
THEN 'Afternoon'
WHEN EXTRACT(HOUR FROM TIMESTAMP(order_purchase_timestamp) ) BETWEEN 19 AND 23
THEN 'Night'
END AS time_of_day,
COUNT(order_id) AS total_orders
FROM `target.orders`
WHERE order_status = 'delivered'
GROUP BY time_of_day
order by total_orders DESC;
```

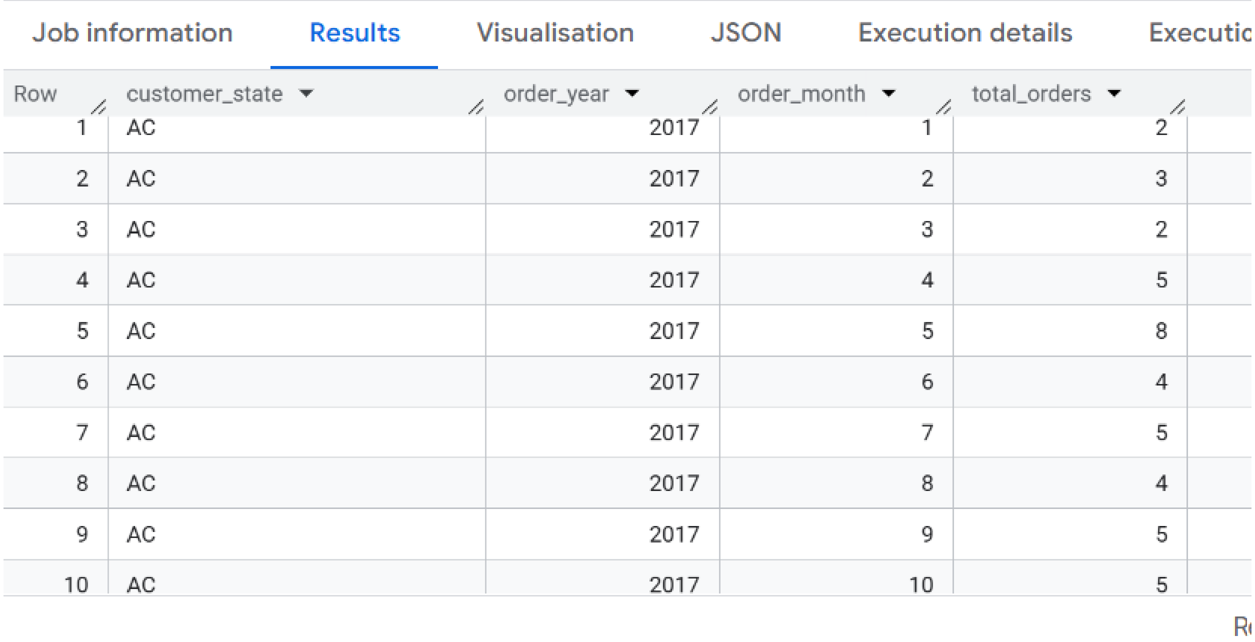


Row	time_of_day	total_orders
1	Afternoon	36965
2	Night	27522
3	Morning	26919
4	Dawn	5072

Evolution of E-commerce Orders in Brazil

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

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

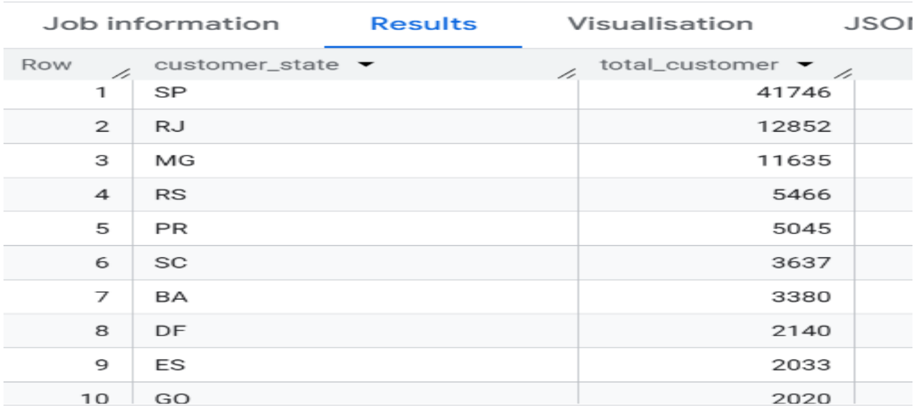


Row	customer_state	order_year	order_month	total_orders
1	AC	2017	1	2
2	AC	2017	2	3
3	AC	2017	3	2
4	AC	2017	4	5
5	AC	2017	5	8
6	AC	2017	6	4
7	AC	2017	7	5
8	AC	2017	8	4
9	AC	2017	9	5
10	AC	2017	10	5

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

```
select customer_state,
COUNT(distinct customer_id) AS total_customer
FROM target.customers
GROUP BY customer_state
```

```
ORDER BY total_customer DESC
```



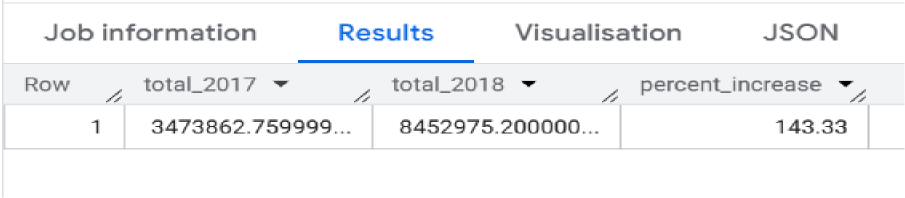
Row	customer_state	total_customer
1	SP	41746
2	RJ	12852
3	MG	11635
4	RS	5466
5	PR	5045
6	SC	3637
7	BA	3380
8	DF	2140
9	ES	2033
10	GO	2020

Impact on Economy

1. 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 yearly_costs AS(
   SELECT
  EXTRACT (YEAR FROM TIMESTAMP(o.order_purchase_timestamp)) AS order_year,
  SUM(p.payment_value) AS total_payment_value
  FROM `target.orders` o
   JOIN `target.payments` p
  ON o.order_id = p.order_id
  WHERE o.order_status = 'delivered'
  AND EXTRACT (MONTH FROM TIMESTAMP(o.order_purchase_timestamp)) BETWEEN 1 AND
8
  AND EXTRACT (YEAR FROM TIMESTAMP (o.order_purchase_timestamp)) IN (2017,
2018)
  GROUP BY order_year
)
SELECT
MAX(CASE WHEN order_year = 2017 THEN total_payment_value END) AS total_2017,
MAX(CASE WHEN order_year =  2018 THEN total_payment_value END) AS total_2018,
ROUND(
  ((MAX(CASE WHEN order_year = 2018 THEN total_payment_value END) -
  MAX(CASE WHEN order_year = 2017 THEN total_payment_value END )) /
  MAX(CASE WHEN order_year = 2017 THEN total_payment_value END)
  )
  * 100, 2
) AS percent_increase
```

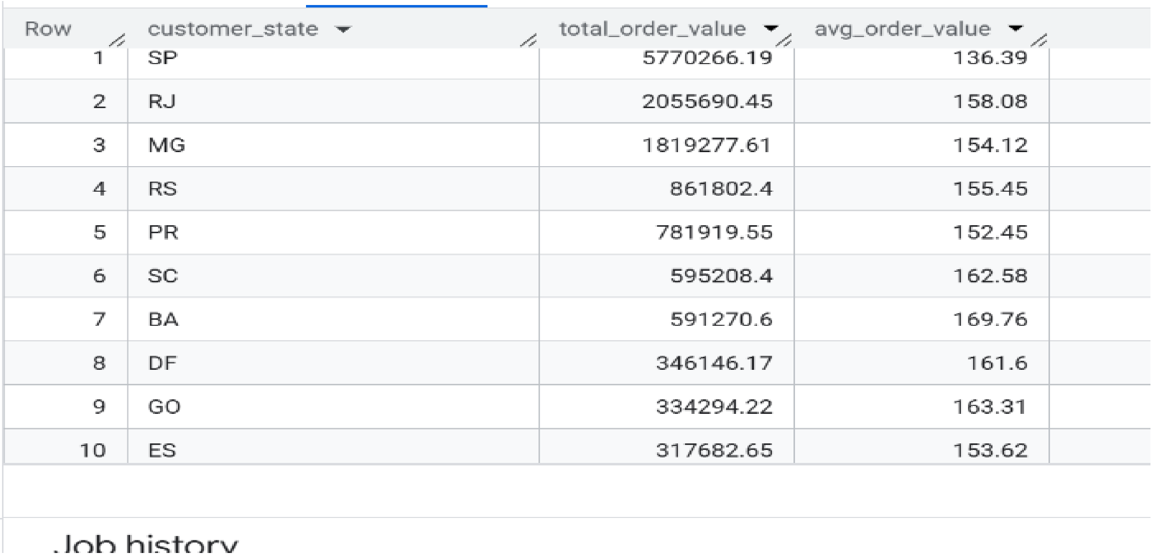
```
FROM yearly_costs;
```



Row	total_2017	total_2018	percent_increase
1	3473862.759999...	8452975.200000...	143.33

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

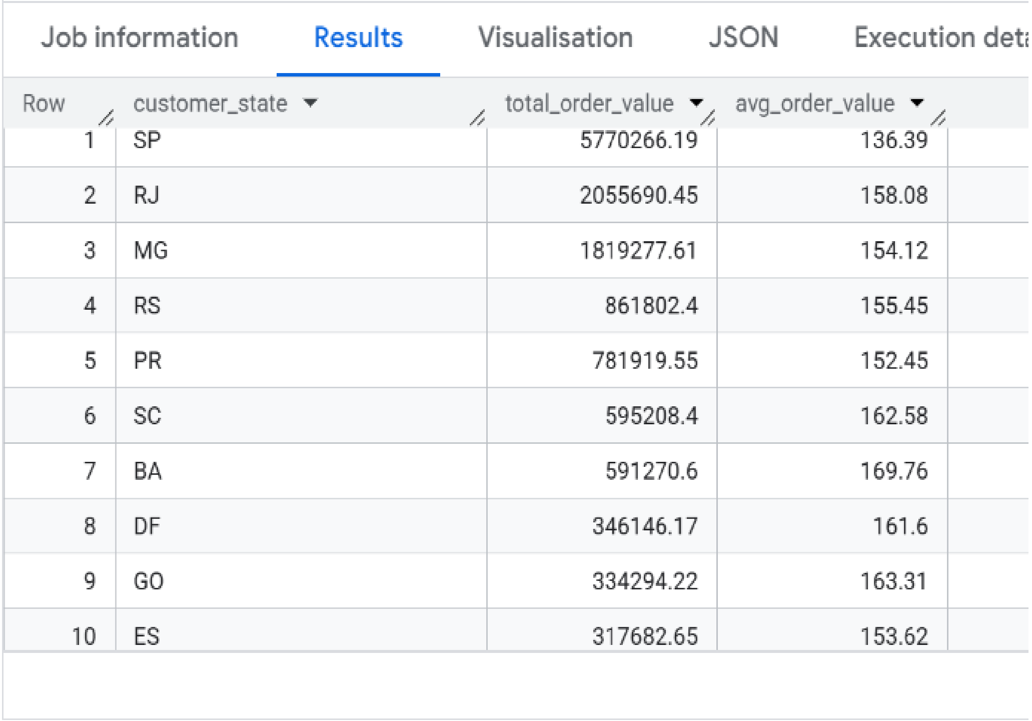
```
SELECT
c.customer_state,
ROUND(SUM(p.payment_value), 2) AS total_order_value,
ROUND(AVG(p.payment_value), 2) AS avg_order_value
FROM `target.customers` c
JOIN `target.orders` o
ON c.customer_id = o.customer_id
JOIN `target.payments` p
ON o.order_id = p.order_id
WHERE o.order_status = 'delivered'
group by c.customer_state
order by total_order_value DESC ;
```



Row	customer_state	total_order_value	avg_order_value
1	SP	5770266.19	136.39
2	RJ	2055690.45	158.08
3	MG	1819277.61	154.12
4	RS	861802.4	155.45
5	PR	781919.55	152.45
6	SC	595208.4	162.58
7	BA	591270.6	169.76
8	DF	346146.17	161.6
9	GO	334294.22	163.31
10	ES	317682.65	153.62

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

```
SELECT
c.customer_state,
ROUND(SUM(oi.freight_value), 2) AS total_freight_value,
ROUND(AVG(oi.freight_value), 2) AS avg_freight_value
FROM `target.customers` c
JOIN `target.orders` o
ON c.customer_id = o.customer_id
JOIN `target.order_items` oi
ON o.order_id = oi.order_id
WHERE o.order_status = 'delivered'
group by c.customer_state
order by total_freight_value DESC ;
```



Row	customer_state	total_order_value	avg_order_value
1	SP	5770266.19	136.39
2	RJ	2055690.45	158.08
3	MG	1819277.61	154.12
4	RS	861802.4	155.45
5	PR	781919.55	152.45
6	SC	595208.4	162.58
7	BA	591270.6	169.76
8	DF	346146.17	161.6
9	GO	334294.22	163.31
10	ES	317682.65	153.62

Analysis Based on Sales, Freight, and Delivery Time

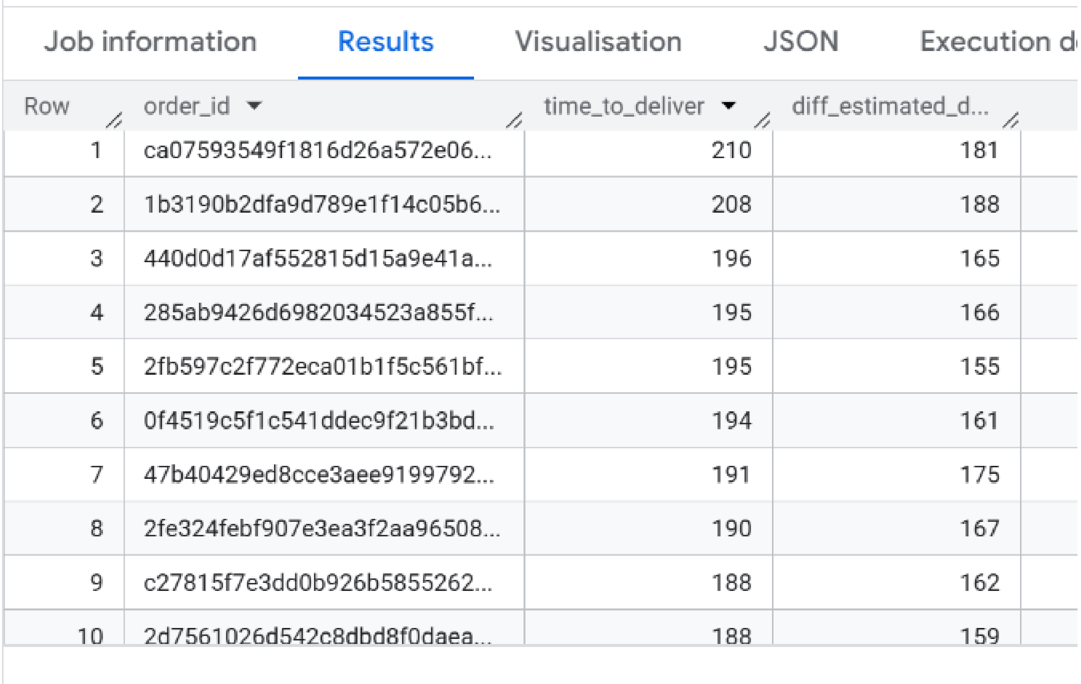
1.Find the no. of days taken to deliver each order from the order's purchase date as delivery time.

Also, calculate the difference (in days) between the estimated & actual delivery date of an order.

Do this in a single query.

```
       SELECT
   o.order_id,
   DATE_DIFF(DATE(o.order_delivered_customer_date),
DATE(o.order_purchase_timestamp),      DAY) AS time_to_deliver,
  DATE_DIFF(DATE(o.order_delivered_customer_date),
DATE(o.order_estimated_delivery_date), DAY) AS diff_estimated_delivery
FROM `target.orders` o
```

```
WHERE o.order_status = 'delivered'
  AND o.order_delivered_customer_date IS NOT NULL
  AND o.order_estimated_delivery_date IS NOT NULL
ORDER BY time_to_deliver DESC;
```



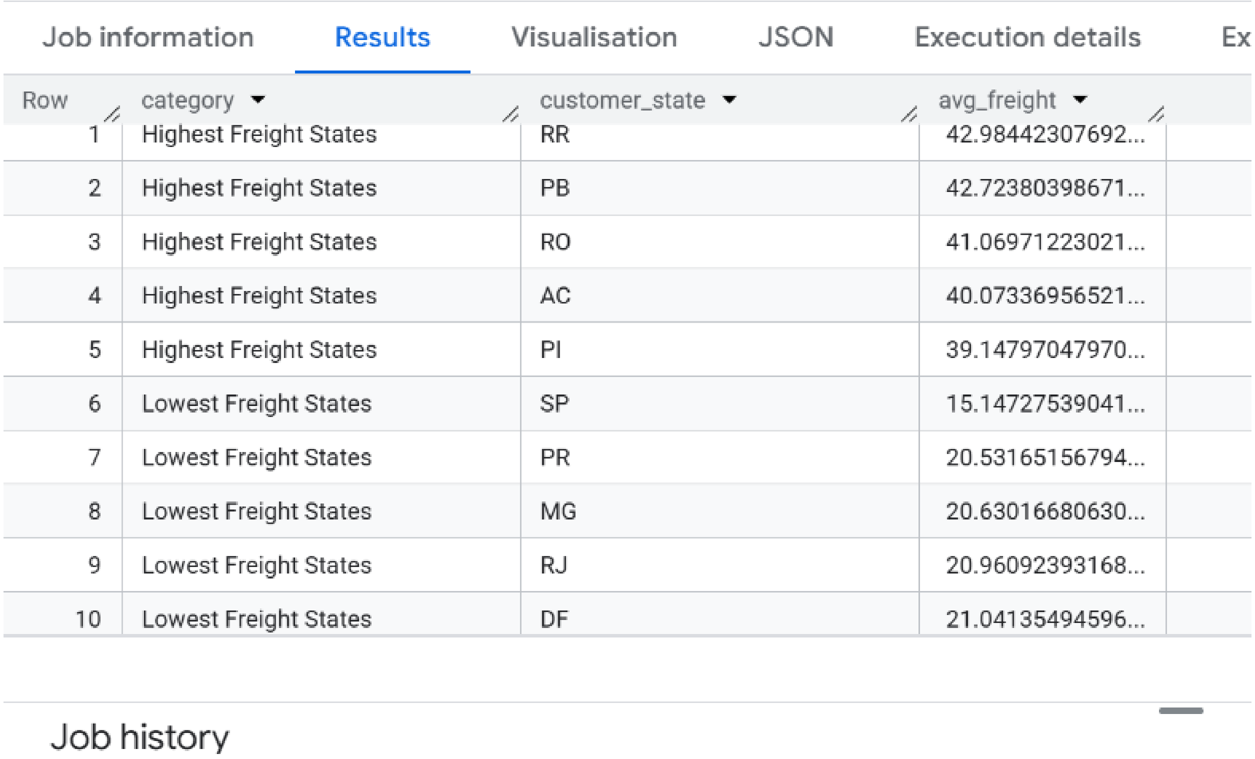
Row	order_id	time_to_deliver	diff_estimated_d...
1	ca07593549f1816d26a572e06...	210	181
2	1b3190b2dfa9d789e1f14c05b6...	208	188
3	440d0d17af552815d15a9e41a...	196	165
4	285ab9426d6982034523a855f...	195	166
5	2fb597c2f772eca01b1f5c561bf...	195	155
6	0f4519c5f1c541ddec9f21b3bd...	194	161
7	47b40429ed8cce3aee9199792...	191	175
8	2fe324febf907e3ea3f2aa96508...	190	167
9	c27815f7e3dd0b926b5855262...	188	162
10	2d7561026d542c8dbd8f0daea...	188	159

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

```
            WITH freight_per_state AS (
    SELECT
    c.customer_state,
    AVG(oi.freight_value) AS avg_freight
  FROM `scaler-dsml-sql-465918.target.order_items` oi
  JOIN `scaler-dsml-sql-465918.target.orders` o
    ON oi.order_id = o.order_id
  JOIN `scaler-dsml-sql-465918.target.customers` c
    ON o.customer_id = c.customer_id
  GROUP BY c.customer_state
),

highest AS (
  SELECT 'Highest Freight States' AS category, customer_state,
avg_freight
  FROM freight_per_state
  ORDER BY avg_freight DESC
  LIMIT 5
),
lowest AS (
  SELECT 'Lowest Freight States' AS category, customer_state,
avg_freight
  FROM freight_per_state
  ORDER BY avg_freight ASC
  LIMIT 5
)

SELECT * FROM highest
UNION ALL
SELECT * FROM lowest;
```



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

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

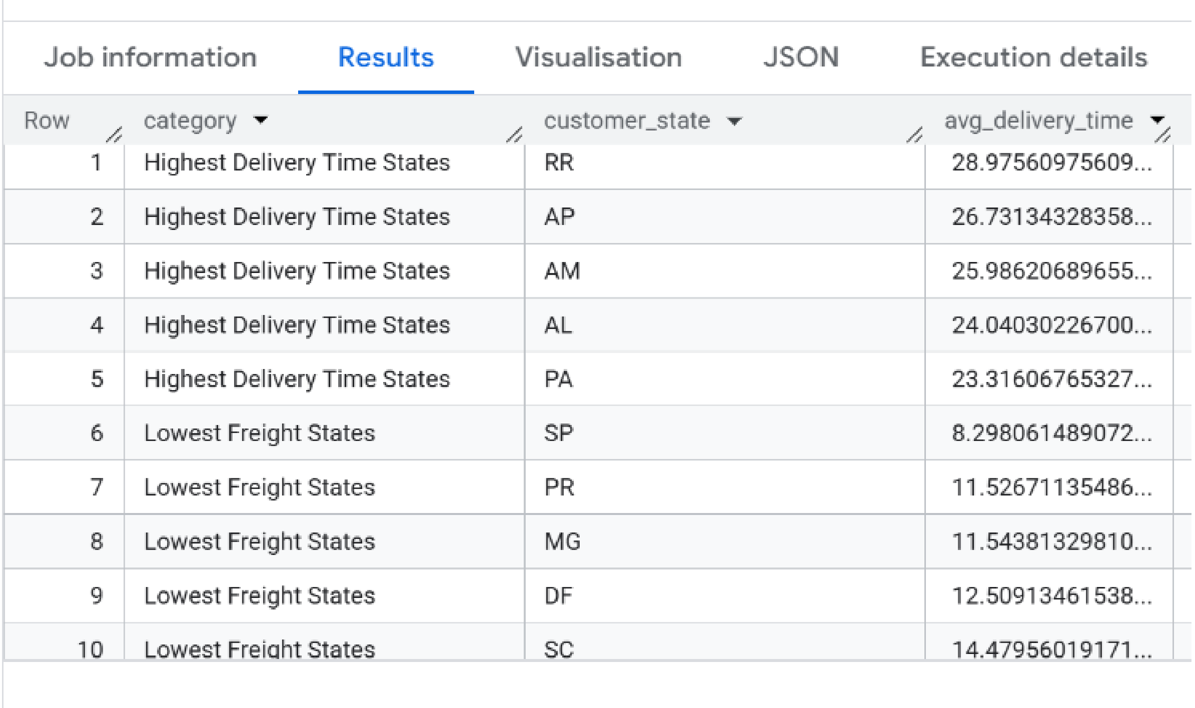
```
WITH delivery_time AS (
  SELECT
    c.customer_state,
    AVG(DATE_DIFF(o.order_delivered_customer_date,o.order_purchase_timestamp,
DAY)) AS avg_delivery_time

  FROM `scaler-dsml-sql-465918.target.orders` o
   JOIN `scaler-dsml-sql-465918.target.customers` c
    ON o.customer_id = c.customer_id
  WHERE o.order_delivered_customer_date IS NOT NULL
  AND o.order_purchase_timestamp IS NOT NULL
  GROUP BY c.customer_state

),
highest AS (
  SELECT 'Highest Delivery Time States' AS category, customer_state,
avg_delivery_time
  FROM delivery_time
  ORDER BY avg_delivery_time DESC
  LIMIT 5
),

lowest AS (
  SELECT 'Lowest Freight States' AS category, customer_state,
avg_delivery_time
  FROM delivery_time
  ORDER BY avg_delivery_time ASC
  LIMIT 5
)

SELECT * FROM highest
UNION ALL
SELECT * FROM lowest;
```



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

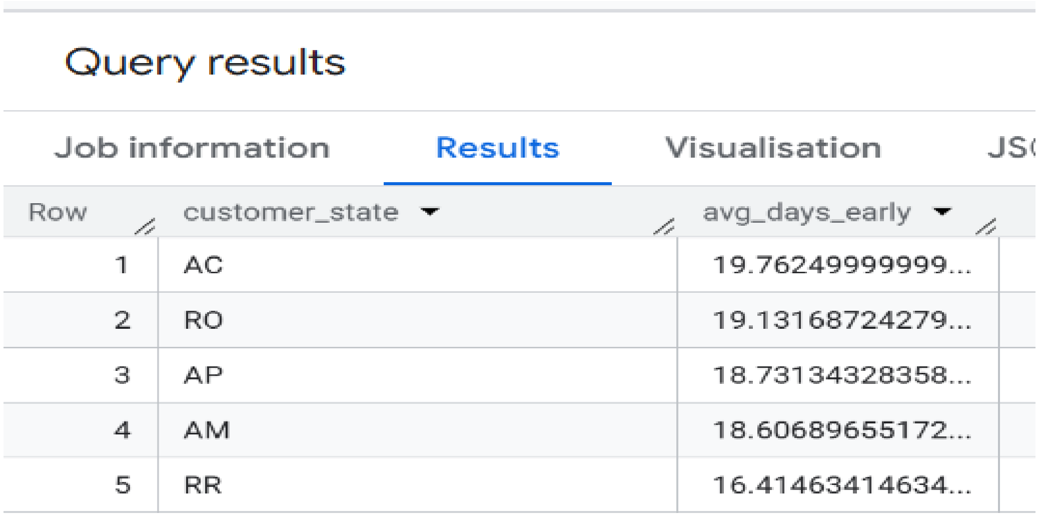
4. Find out the top 5 states where the order delivery is really fast as compared to the estimated date of delivery.
You can use the difference between the averages of actual & estimated delivery date to figure out how fast the delivery was for

each state.

```
SELECT
    c.customer_state,

AVG(DATE_DIFF(o.order_estimated_delivery_date,o.order_delivered_customer_date,
DAY)) AS avg_days_early

  FROM `scaler-dsml-sql-465918.target.orders` o
   JOIN `scaler-dsml-sql-465918.target.customers` c
    ON o.customer_id = c.customer_id
  WHERE o.order_delivered_customer_date IS NOT NULL
  AND o.order_purchase_timestamp IS NOT NULL
  GROUP BY c.customer_state
  order BY avg_days_early DESC
  LIMIT 5;
```

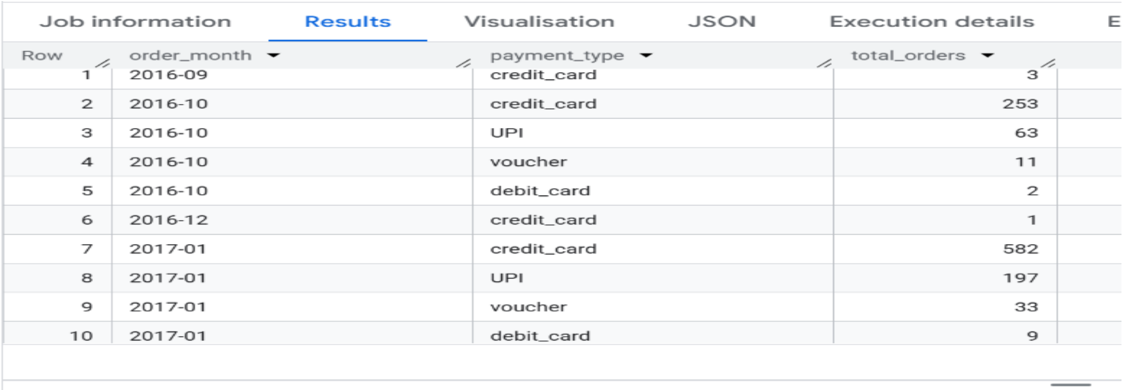


Row	customer_state	avg_days_early
1	AC	19.76249999999...
2	RO	19.13168724279...
3	AP	18.73134328358...
4	AM	18.60689655172...
5	RR	16.41463414634...

Payments Analysis

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

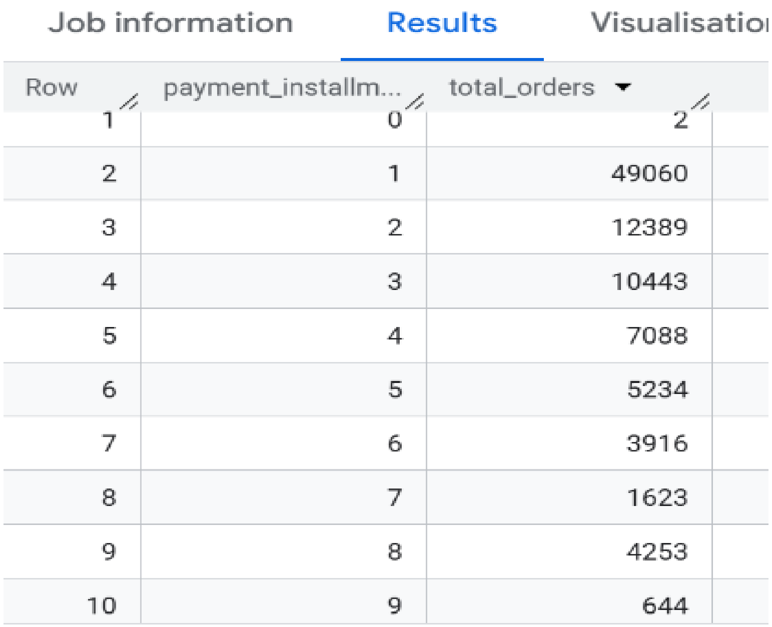
```
 SELECT
FORMAT_TIMESTAMP('%Y-%m', o.order_purchase_timestamp) AS order_month,
p.payment_type,
COUNT(DISTINCT o.order_id) AS total_orders
FROM `scaler-dsml-sql-465918.target.orders` o
JOIN `scaler-dsml-sql-465918.target.payments` p
ON o.order_id = p.order_id
WHERE o.order_purchase_timestamp IS NOT NULL
GROUP BY order_month, p.payment_type
ORDER BY order_month, total_orders DESC ;
```



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

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

```
SELECT p.payment_installments ,
COUNT(DISTINCT p.order_id) AS total_orders
FROM `target.payments` p
group by p.payment_installments
order by p.payment_installments
```



Row	payment_installm...	total_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

Actionable Insights & Recommendations

1. Customer Ordering Trends

Insight:
- Orders are steadily growing year-on-year with clear peaks in seasonal months.

Recommendation:
- Invest in seasonal campaigns and ensure inventory readiness before peak demand.

Action Items:
- Launch targeted marketing campaigns 1–2 months before peak sales periods.
- Improve demand forecasting using historical order trends.
- Ensure warehouses are well-stocked in regions with consistently high demand.

2. Time of Day for Orders

Insight:
- Most orders are placed in the Afternoon (13–18 hrs) and Night (19–23 hrs).

Recommendation:
- Optimize advertising spend and customer engagement during these time slots.

Action Items:
- Schedule digital ads, SMS/email campaigns, and app push notifications in afternoon & evening slots.
- Offer limited-time flash sales in the evenings to drive conversions.

3. Regional Distribution

Insight:
- Customers are highly concentrated in Southeast states (SP, RJ, MG).
- Northern and Northeastern states show relatively lower penetration.

Recommendation:
- Optimize logistics hubs and marketing spend for the Southeast.
- Expand awareness and offers in underpenetrated regions.

Action Items:
- Strengthen warehouses and last-mile delivery partners in Southeast Brazil.
- Run localized marketing campaigns for North/Northeast states to grow presence.

4. Freight & Delivery Performance

Insight:
- Some states incur higher freight costs, discouraging repeat orders.
- Many orders are delivered faster than estimated, but this isn’t highlighted to customers.

Recommendation:
- Reduce freight costs through better partnerships.
- Use early delivery as a competitive marketing advantage.

Action Items:
- Negotiate better rates with logistics providers in high-cost regions.
- Introduce free/discounted shipping thresholds (e.g., orders above R$150).
- Update customer communication: “Your order arrived 2 days earlier than expected .”

5. Payments & Installments

Insight:
- Majority of payments are via credit card (single installment).
- A noticeable share use multi-installments, especially for high-value items.

Recommendation:
- Promote installment flexibility to encourage higher-value purchases.

Action Items:
- Partner with banks to offer 0% EMI options.
- Highlight “Buy Now, Pay Later” schemes on product pages.

Conclusion

This analysis of Target’s operations in Brazil highlights key trends in customer behavior, payments, freight, and delivery. Orders are growing steadily, customers prefer credit card payments, and Southeast states dominate in sales volume. While delivery times vary across states, many orders are delivered earlier than expected – an opportunity for improving customer communication. Freight costs remain high in certain regions, which can be optimized through better logistics partnerships. The actionable recommendations provided in this report should guide Target in enhancing its customer experience, optimizing logistics, and strengthening its competitive edge in the Brazilian market.