

Data type:

- **Query:**

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

- **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

- **Insight:**

- Customer table structure contains unique customer id with location info (city & state) which can be useful for geo-analysis

First and last order timestamp:

- **Query:**

```
SELECT
MIN(order_purchase_timestamp) AS first_order_time,
MAX(order_purchase_timestamp) AS last_order_time
from target.orders;
```

- **Result:**

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

- **Insight:**

Order timeline spans for 2 years from **September 2016** to **August 2018**.

Cities and states:

SELECT

```
COUNT(DISTINCT customer_city) AS unique_cities,  
COUNT(DISTINCT customer_state) AS unique_states
```

FROM target.customers;

Job information Results Visualisation JSON Ex				
Row	unique_cities	unique_states		
1	4119	27		

Order trend:

- **Query**

SELECT

```
DATE_TRUNC(order_purchase_timestamp, MONTH) AS month,
```

```
COUNT(order_id) AS total_orders
```

FROM target.orders

```
WHERE order_status = 'delivered'
```

```
GROUP BY month
```

```
ORDER BY month;
```

- **Result:**

Row	month	total_orders	
1	2016-09-01 00:00:00 UTC	1	
2	2016-10-01 00:00:00 UTC	265	
3	2016-12-01 00:00:00 UTC	1	
4	2017-01-01 00:00:00 UTC	750	
5	2017-02-01 00:00:00 UTC	1653	
6	2017-03-01 00:00:00 UTC	2546	
7	2017-04-01 00:00:00 UTC	2303	
8	2017-05-01 00:00:00 UTC	3546	
9	2017-06-01 00:00:00 UTC	3135	
10	2017-07-01 00:00:00 UTC	3872	

- **Insights:**

1. There's a **steady increase in orders from 2016 → 2018**, peaking mid-2018.

2. Customer adoption of e-commerce was rising consistently over the years as per the result.

Monthly peak orders:

- **Query:**

```
SELECT  
  
    FORMAT_TIMESTAMP('%B', order_purchase_timestamp) AS month,  
  
    COUNT(order_id) AS total_orders  
  
FROM target.orders  
  
WHERE order_status = 'delivered'  
  
GROUP BY month  
  
ORDER BY total_orders DESC;
```

- **Result:**

Row	month ▼	total_orders ▼	
1	August	10544	
2	May	10295	
3	July	10031	
4	March	9549	
5	June	9234	
6	April	9101	
7	February	8208	
8	January	7819	
9	November	7289	
10	December	5514	

- **Insights:**

1. Orders tend to be at peak in August followed by May and July.

Orders with respect to time of the day:

- **Query:**

```
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'

    ELSE 'Night'

END AS time_of_day,
COUNT(order_id) AS total_orders
FROM target.orders
GROUP BY time_of_day
ORDER BY total_orders DESC;
```

- **Result:**

Row	time_of_day	total_orders
1	Afternoon	38135
2	Night	28331
3	Morning	27733
4	Dawn	5242

- **Insight:**

1. Majority of purchases happen in the **Afternoon** and **Night** when compared to Morning & Dawn .

Orders by state:

- **Query**

```
SELECT
  c.customer_state,
  DATE_TRUNC(o.order_purchase_timestamp, MONTH) AS 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, month
ORDER BY total_orders desc;
```

- **Result:**

1	SP	2018-08-01 00:00:00 UTC	3164	
2	SP	2018-05-01 00:00:00 UTC	3138	
3	SP	2018-04-01 00:00:00 UTC	3002	
4	SP	2018-01-01 00:00:00 UTC	2975	
5	SP	2018-03-01 00:00:00 UTC	2971	
6	SP	2017-11-01 00:00:00 UTC	2899	
7	SP	2018-06-01 00:00:00 UTC	2738	
8	SP	2018-07-01 00:00:00 UTC	2715	
9	SP	2018-02-01 00:00:00 UTC	2632	

- **Insights:**

1. **SP** state leads by a wide margin, followed by **RJ**, **MG** and **RS**.

Customer distribution over states:

- **Query:**

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

- **Result:**

Row	customer_state	unique_customers
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

- **Insight:**

1. Similar to the most orders, the most unique customers were from the states **SP, RJ, MG, RS.**

Percent increase in orders:

- **Query:**

```
WITH yearly_cost AS (  
    SELECT  
        EXTRACT(YEAR FROM o.order_purchase_timestamp) AS year,  
        SUM(p.payment_value) AS total_payment  
    FROM target.orders o  
    JOIN target.payments p ON o.order_id = p.order_id  
    WHERE EXTRACT(MONTH FROM o.order_purchase_timestamp) BETWEEN 1 AND 8  
    GROUP BY year  
)  
  
SELECT  
    ((MAX(total_payment) - MIN(total_payment)) / MIN(total_payment)) * 100 AS  
    pct_increase  
FROM yearly_cost;
```

- **Result:**

row	pct_increas
1	136.9768716466...

Total and average price per each state:

- **Query:**

```
SELECT
    c.customer_state,
    SUM(oi.price) AS total_order_price,
    AVG(oi.price) AS avg_order_price
FROM target.orders o
JOIN target.order_items oi ON o.order_id = oi.order_id
JOIN target.customers c ON o.customer_id = c.customer_id
GROUP BY c.customer_state
ORDER BY total_order_price DESC;
```

- **Result:**

Row	customer_state ▼	total_order_price ▼	avg_order_price ▼
1	SP	5202955.050001...	109.6536291597...
2	RJ	1824092.669999...	125.1178180945...
3	MG	1585308.029999...	120.7485741488...
4	RS	750304.0200000...	120.3374530874...
5	PR	683083.7600000...	119.0041393728...
6	SC	520553.3400000...	124.6535775862...
7	BA	511349.9900000...	134.6012082126...
8	DF	302603.9399999...	125.7705486284...
9	GO	294591.9499999...	126.2717316759...
10	ES	275037.3099999...	121.9137012411...

Total and average freight value per state:

- **Query:**

```
SELECT  
  
    c.customer_state,  
  
    SUM(oi.freight_value) AS total_freight,  
  
    AVG(oi.freight_value) AS avg_freight  
  
FROM target.orders o  
  
JOIN target.order_items oi ON o.order_id = oi.order_id  
  
JOIN target.customers c ON o.customer_id = c.customer_id  
  
GROUP BY c.customer_state  
  
ORDER BY total_freight DESC;
```

- **Result:**

Row	customer_state	total_freight	avg_freight
1	SP	718723.0700000...	15.14727539041...
2	RJ	305589.3100000...	20.96092393168...
3	MG	270853.4600000...	20.63016680630...
4	RS	135522.7399999...	21.73580433039...
5	PR	117851.6799999...	20.53165156794...
6	BA	100156.6800000...	26.36395893656...
7	SC	89660.2599999...	21.47036877394...
8	PE	59449.6600000001	32.91786267995...
9	GO	53114.98000000...	22.76681525932...
10	DF	50625.4999999...	21.04135494596...

Delivery time and estimated delivery difference:

- **Query:**

```
SELECT  
  
    o.order_id,  
  
    DATE_DIFF(o.order_delivered_customer_date, o.order_purchase_timestamp, DAY) AS  
time_to_deliver,  
  
    DATE_DIFF(o.order_estimated_delivery_date, o.order_delivered_customer_date, DAY) AS  
diff_estimated_delivery  
  
FROM target.orders o  
  
WHERE o.order_status = 'delivered';
```

- **Result:**

Row	order_id	time_to_deliver	diff_estimated_d...
1	bfb0f9bdef84302105ad712db...	54	-36
2	98974b076b01553d49ee64679...	43	6
3	c4b41c36dd589e901f6879f25a...	36	14
4	d2292ff2201e74c5db154d1b7a...	29	20
5	95e01270fcb9e986342340010...	30	19
6	ed8c7b1b3eb256c70ce0c7423...	44	5
7	5cc475c7c03290048eb2e742c...	68	-18
8	6b3ee7697a02619a0ace2b3f0a...	47	2
9	3b2ca3293a7ce539ea2379d70...	43	7
10	b2f92b2f7047cd8b35580d629d...	43	7

- **Insight:**

1. Estimated delivery is often longer than actual delivery (i.e) logistics are better than promised which is good for customer satisfaction.

Top 5 states per average freight value:

- **Query:**

```
SELECT
    c.customer_state,
    AVG(oi.freight_value) AS avg_freight
FROM target.orders o
JOIN target.order_items oi ON o.order_id = oi.order_id
JOIN target.customers c ON o.customer_id = c.customer_id
GROUP BY c.customer_state
ORDER BY avg_freight DESC
```

- **Result:**

Row	customer_state ▼	avg_freight ▼	
1	RR	42.98442307692...	
2	PB	42.72380398671...	
3	RO	41.06971223021...	
4	AC	40.07336956521...	
5	PI	39.14797047970...	

Bottom 5 states per average freight value:

- **Query:**

```
SELECT
    c.customer_state,
    AVG(oi.freight_value) AS avg_freight
FROM target.orders o
JOIN target.order_items oi ON o.order_id = oi.order_id
JOIN target.customers c ON o.customer_id = c.customer_id
GROUP BY c.customer_state
ORDER BY avg_freight ASC
```

- **Result:**

Row	customer_state	avg_freight
1	SP	15.14727539041...
2	PR	20.53165156794...
3	MG	20.63016680630...
4	RJ	20.96092393168...
5	DF	21.04135494596...

- **Insights:**

1. Top freight states: RR, PB, RO
2. Lowest freight states: SP, PR, MG.

Top 5 states with lowest delivery time:

- **Query:**

```

SELECT
    c.customer_state,
    AVG(DATE_DIFF(o.order_delivered_customer_date, o.order_purchase_timestamp, DAY))
    AS avg_delivery_time
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 BY avg_delivery_time ASC
LIMIT 5;

```

Result:

Row	customer_state ▼	avg_delivery_time ▼
1	SP	8.298093544722...
2	PR	11.52671135486...
3	MG	11.54218777523...
4	DF	12.50913461538...
5	SC	14.47518330513...

Top 5 states with highest delivery times:

- Query:**

```
SELECT
    c.customer_state,
    AVG(DATE_DIFF(o.order_delivered_customer_date, o.order_purchase_timestamp, DAY))
    AS avg_delivery_time
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 BY avg_delivery_time DESC
LIMIT 5;
```

- Result:**

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...

Top 5 states with faster delivery time:

- **Query:**

```
SELECT
    c.customer_state,
    AVG(DATE_DIFF(o.order_estimated_delivery_date, o.order_delivered_customer_date,
DAY)) AS avg_diff
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 BY avg_diff DESC
LIMIT 5;
```

- **Result:**

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

- **Insights:**

- Fastest deliveries in SP, PR, MG.
- Slowest deliveries in RR, AP, AM.
- In many states, deliveries are completed 2–3 days earlier than estimated which Builds trust & improves NPS.

Month on Month orders per payment type:

- **Query:**

```
SELECT
    DATE_TRUNC(o.order_purchase_timestamp, MONTH) AS month,
    p.payment_type,
    COUNT(DISTINCT o.order_id) AS total_orders
FROM target.orders o
JOIN target.payments p ON o.order_id = p.order_id
GROUP BY month, p.payment_type
ORDER BY month, total_orders DESC;
```

- **Result:**

Row	month ▼	payment_type ▼	total_orders ▼
1	2016-09-01 00:00:00 UTC	credit_card	3
2	2016-10-01 00:00:00 UTC	credit_card	253
3	2016-10-01 00:00:00 UTC	UPI	63
4	2016-10-01 00:00:00 UTC	voucher	11
5	2016-10-01 00:00:00 UTC	debit_card	2
6	2016-12-01 00:00:00 UTC	credit_card	1
7	2017-01-01 00:00:00 UTC	credit_card	582
8	2017-01-01 00:00:00 UTC	UPI	197
9	2017-01-01 00:00:00 UTC	voucher	33
10	2017-01-01 00:00:00 UTC	debit_card	9

Total orders as per the number of installments:

- **Query:**

```
SELECT
    p.payment_installments,
    COUNT(DISTINCT o.order_id) AS total_orders
FROM payments p
JOIN orders o ON o.order_id = p.order_id
WHERE p.payment_installments >= 1
GROUP BY p.payment_installments
ORDER BY total_orders DESC;
```

- **Result:**

Row	payment_installm...	total_orders
1	1	49060
2	2	12389
3	3	10443
4	4	7088
5	10	5315
6	5	5234
7	8	4253
8	6	3916
9	7	1623
10	9	644

- **Insights:**

1. Credit card payment dominates often with multiple installments.
2. Some usage of boleto and debit card is also seen.
3. Many customers pay in 3–6 installments, some orders even have 10+.

Recommendations:

- Strengthen marketing & logistics presence in SP, RJ, and MG where demand is highest.
- Track cities with small orders, but wide coverage for future regional expansion opportunities.
- Run ads/promotions during Afternoon and Night when customers shop the most.
- Scale servers, support, and delivery staff in the peak months to handle holiday demands.
- Push seasonal offers in quieter months to smooth sales fluctuations.
- Use personalized notifications around customer's active hours (afternoon/night) for higher conversion.
- Focus logistics and delivery hubs in SP, RJ, MG, RS.
- Launch localized campaigns in low sales states to improve adoption.
- Improve the inventory and warehouse structures across the states to maintain and reduce the delivery time.
- Provide more discounts in the low sales state to improve the business in the region.
- Focus on same day or one day deliveries in the top states such as SP, RJ, etc to improve customer satisfaction and repeated online orders.
- Continue supporting installment payments, as they drive conversion rates in Brazil.
- Partner with fintech providers for better installment/credit options (lower interest, longer plans).
- Offer discounts/cashback for one-time payments to improve cash flow.
- Educate customers on digital wallets / new payment types to diversify beyond credit cards.