

## Q1. Exploration of Dataset

### 1) DATATYPE OF TABLE-

#### 1.customers table -



**Query -** SELECT column\_name, data\_type  
FROM information\_schema.columns  
WHERE table\_name = 'customers';

**Output -**

	COLUMN_NAME	DATA_TYPE	
▶	customer_id	varchar	
●	customer_name	varchar	
●	phone	varchar	
●	address	varchar	
●	city	varchar	
●	state	varchar	
●	postal_code	varchar	
●	country	varchar	
●	customer_id	text	
●	customer_unique_id	text	
●	customer_zip_cod...	text	
●	customer_city	text	
●	customer_state	text	

#### 2.geolocation table -

SELECT column\_name, data\_type  
FROM information\_schema.columns  
WHERE table\_name = 'geolocation';

Result Grid   Filter Rows: <input type="text" value="Search"/>			
	COLUMN_NAME	DATA_TYPE	
▶	geolocation_zip_code_prefix	text	
●	geolocation_lat	double	
●	geolocation_lng	double	
●	geolocation_city	text	
●	geolocation_state	text	
●			
●			
●			

### 3.order item table -

```
SELECT column_name, data_type  
FROM information_schema.columns  
WHERE table_name = 'order_items';
```

	COLUMN_NAME	DATA_TYPE	
▶	order_id	text	
■	order_item_id	int	
■	product_id	text	
■	seller_id	text	
■	shipping_limit_date	text	
■	price	double	
■	freight_value	double	
■			
■			

### 4.order reviews table -

```
SELECT column_name, data_type  
FROM information_schema.columns  
WHERE table_name = 'order_reviews';
```

	COLUMN_NAME	DATA_TYPE	
▶	review_id	text	
■	order_id	text	
■	review_score	int	
■	review_comment_title	text	
■	review_creation_date	text	
■	review_answer_timestamp	text	
■			

### 5.orders table -

```
SELECT column_name, data_type  
FROM information_schema.columns  
WHERE table_name = 'orders';
```

	COLUMN_NAME	DATA_TYPE	
▶	order_id	text	
	customer_id	text	
	order_status	text	
	order_purchase_timestamp	text	
	order_approved_at	text	
	order_delivered_carrier_date	text	
	order_delivered_customer_date	text	
	order_estimated_delivery_date	text	

## 6.payments table -

```
SELECT column_name, data_type
FROM information_schema.columns
WHERE table_name = 'payments';
```

	COLUMN_NAME	DATA_TYPE	
▶	order_id	text	
	payment_sequential	int	
	payment_type	text	
	payment_installments	int	
	payment_value	double	

## 7.products table -

```
SELECT column_name, data_type
FROM information_schema.columns
WHERE table_name = 'products';
```

	COLUMN_NAME	DATA_TYPE	
▶	product_id	varchar	
	product_name	varchar	
	product_description	varchar	
	product_code	varchar	
	PRODUCT_LINE	varchar	
	price	float	
	product_id	varchar	
	product_name	varchar	
	product_description	varchar	
	product_id	text	
	product category	text	
	product_name_le...	int	
	product_descriptio...	int	
	product_photos_qty	int	
	product_weight_g	int	
	product_length_cm	int	
	product_height_cm	int	
	product_width_cm	int	

## 8.sellers table -

```
SELECT column_name, data_type
FROM information_schema.columns
WHERE table_name = 'sellers';
```

	COLUMN_NAME	DATA_TYPE	
▶	seller_id	text	
	seller_zip_code_prefix	text	
	seller_city	text	
	seller_state	text	

## 2) TIME PERIOD FOR WHICH DATA IS GIVEN -

**Query -** SELECT MIN(Year(order\_purchase\_timestamp)) as start\_date,  
MAX(Year(order\_delivered\_customer\_date)) as end\_date from orders;

**Output -**

	start_date	end_date	
▶	2016	2018	

The time period for which the data is given is **2016 - 2018**

## 3) CITIES AND STATES OF CUSTOMERS ORDERED DURING THIS PERIOD -

**Query -** SELECT customer\_city, customer\_state FROM customers  
JOIN orders USING(customer\_id)  
WHERE order\_purchase\_timestamp BETWEEN  
(  
SELECT MIN(Year(order\_purchase\_timestamp)) FROM orders) AND  
(SELECT MAX(Year(order\_delivered\_customer\_date)) FROM orders  
);

	customer_city	customer_state	
▶	sao paulo	SP	
	sumare	SP	
	sao jose dos pinhais	PR	
	porto alegre	RS	
	contagem	MG	
	guaruja	SP	
	sao paulo	SP	
	sao paulo	SP	
	serrinha	BA	
	rio de janeiro	RJ	

## Q2.In Depth Exploration

### 1) GROWING TREND OF E-COMMERCE

**Query -** SELECT YEAR(order\_purchase\_timestamp) as year,  
MONTH(order\_purchase\_timestamp) as month, count(order\_id) AS order\_count,  
row\_number() over(order by count(order\_id) desc) AS ranks  
FROM orders  
GROUP BY year, month  
ORDER BY year, month;

**Output -**

	year	month	order_count	ranks	
▶	2016	9	4	23	
■	2016	10	324	21	
■	2016	12	1	25	
■	2017	1	800	20	
■	2017	2	1780	19	
■	2017	3	2682	17	
■	2017	4	2404	18	
■	2017	5	3700	15	
■	2017	6	3245	16	
■	2017	7	4026	14	

From the above table we can come to the conclusion that **there is for sure a growing trend on e-commerce in Brazil over the years.**

-Seasonal time comes around **months 5 - 8 (May - August)** with good no of orders every year and a slight decline after that .

Peaks at specific months comes in **Month 8 (August)**, which has a high no of order every year.

**Recommendation** - It will be advised to increase the stock of products during these months (May - August).

### 2)TIME BRAZILIAN CUSTOMERS TEND TO BUY

**Query -**

```
WITH a AS (  
  SELECT CASE  
    WHEN hour(order_purchase_timestamp) BETWEEN 0 AND 6 THEN 'Dawn'  
    WHEN hour(order_purchase_timestamp) BETWEEN 6 AND 12 THEN 'Morning'  
    WHEN hour(order_purchase_timestamp) BETWEEN 12 AND 18 THEN 'Afternoon'  
    WHEN hour(order_purchase_timestamp) BETWEEN 18 AND 24 THEN 'Night'  
  END as time  
  FROM orders )  
SELECT a.time, count(a.time) AS count
```

```
FROM a
GROUP BY a.time
ORDER BY count DESC;
```

**Output -**

	time	count
▶	Afternoon	38135
	Night	28331
	Morning	27733
	Dawn	5242

**Inference -**

The time at which Brazilian customers tend to buy stuff are in the following order :

- 1.Afternoon
- 2.Night
- 3.Morning
- 4.Dawn

**Afternoon** is the time most Brazilian customers tend to buy stuff.

**Recommendation** - It can be advised to increase the number of staff during Afternoon. And reduce the staff during dawn.

### **Q3. Evolution of E-commerce orders**

#### **1) MONTH ON MONTH ORDERS BY STATES**

**Query -**

```
SELECT Year(order_purchase_timestamp) AS year,
Month(order_purchase_timestamp) AS month, customer_state,
count(order_id) as order_count
FROM orders o
JOIN customers c USING(customer_id)
GROUP BY year, month, customer_state
ORDER BY year, month;
```

**Output -**

	year	month	customer_state	order_count	
▶	2016	9	RR	1	
	2016	9	RS	1	
	2016	9	SP	2	
	2016	10	AL	2	
	2016	10	BA	4	
	2016	10	CE	8	
	2016	10	DF	6	
	2016	10	ES	4	
	2016	10	GO	9	
	2016	10	MA	4	

**Inference** - Highest no of orders is in 11/2017 (November 2017)  
- Lowest no of orders is in 12/2016 (December 2016)

## 2) DISTRIBUTION OF CUSTOMERS ACROSS STATES IN BRAZIL

**Query -**

```
SELECT customer_state, count(customer_unique_id) as customer_count  
FROM customers  
GROUP BY customer_state;
```

**Output -**

	customer_state	customer_count	
▶	SP	41746	
	SC	3637	
	MG	11635	
	PR	5045	
	RJ	12852	
	RS	5466	
	PA	975	
	GO	2020	
	ES	2033	
	BA	3380	

**Inference** - Maximum no of customers is in state SP  
- Least no of customers are in state RR.

## Q4. Impact on Economy

### 1) % INCREASE IN COST OF ORDERS

#### Query -

```
SELECT YEAR(order_purchase_timestamp) as year, MONTH(order_purchase_timestamp)
as month,
ROUND(SUM(payment_value)) as payment_value
FROM orders o
JOIN payments p using(order_id)
WHERE YEAR(order_purchase_timestamp) IN (2017,2018) AND
MONTH(order_purchase_timestamp) NOT IN (9,10,11,12)
GROUP BY year, month
ORDER BY year, month;
```

#### Output -

	year	month	payment_value	
▶	2017	1	138488	
	2017	2	291908	
	2017	3	449864	
	2017	4	417788	
	2017	5	592919	
	2017	6	511276	
	2017	7	592383	
	2017	8	674396	
	2018	1	1115004	
	2018	2	992463	

### 2) MEAN & SUM OF PRICE AND FREIGHT VALUES BY CUSTOMER STATE

#### Query -

```
SELECT c.customer_state, ROUND(avg(oi.price)) AS mean_price ,
ROUND(avg(oi.freight_value)) AS mean_freight_value,
ROUND(sum(oi.price)) AS sum_price, ROUND(sum(oi.freight_value)) AS
sum_freight_value
FROM customers c
JOIN orders o using(customer_id)
JOIN order_items oi using(order_id)
GROUP BY c.customer_state;
```

#### Output -

	customer_state	mean_price	mean_freight_value	sum_price	sum_freight_value	
▶	SP	110	15	5202955	718723	
	RS	120	22	750304	135523	
	AP	164	34	13474	2788	
	SC	125	21	520553	89660	
	BA	135	26	511350	100157	
	MS	143	23	116813	19144	
	RJ	125	21	1824093	305589	
	PI	160	39	86914	21218	
	MG	121	21	1585308	270853	
	ES	122	22	275037	49765	



## Q5. Analysis on Sales, Freight and Delivery Time

### 1) DAYS BETWEEN PURCHASING, DELIVERING AND ESTIMATED DELIVERY

Query -

```
SELECT
    DATEDIFF(order_delivered_customer_date,order_purchase_timestamp) AS
    purchase_to_delivery,
    DATEDIFF(order_estimated_delivery_date,order_delivered_customer_date) AS
    delivery_to_estimated_delivery
FROM orders;
```

Output -

	purchase_to_delivery	delivery_to_estimated_delivery	
▶	8	8	
	14	6	
	9	18	
	14	13	
	3	10	
	17	6	
	NULL	NULL	
	10	12	
	10	32	
	18	7	

### 2) FIND time\_to\_delivery and diff\_estimated\_delivery

Query -

```
SELECT
    DATEDIFF(order_delivered_customer_date,order_purchase_timestamp) AS
    time_to_delivery,
    DATEDIFF(order_estimated_delivery_date, order_delivered_customer_date) AS
    diff_estimated_delivery
FROM orders;
```

Output -

	time_to_delivery	diff_estimated_delivery	
▶	8	8	
	14	6	
	9	18	
	14	13	
	3	10	
	17	6	
	NULL	NULL	
	10	12	
	10	32	
	18	7	

### 3) GROUP DATA BY STATE, TAKE MEAN OF FRIEGHT VALUE, time\_to\_delivery AND diff\_estimated\_delivery

#### Query -

```
WITH a AS (
  SELECT order_id, DATEDIFF(order_delivered_customer_date,order_purchase_timestamp)
  AS time_to_delivery,
  DATEDIFF(order_estimated_delivery_date, order_delivered_customer_date) AS
  diff_estimated_delivery
  FROM orders),
```

```
  b as (
    SELECT c.customer_state, oi.order_id,SUM(oi.freight_value) AS freight_value
    FROM order_items oi
    JOIN orders o using(order_id)
    JOIN customers c using(customer_id)
    GROUP BY oi.order_id, c.customer_state)
```

```
SELECT b.customer_state,ROUND(AVG(b.freight_value)) AS mean_freight_value,
ROUND(AVG(a.time_to_delivery)) as avg_time_to_delivery,
ROUND(AVG(a.diff_estimated_delivery)) as avg_estimated_time
FROM b
JOIN a using(order_id)
GROUP BY b.customer_state;
```

	customer_state	mean_freight_value	avg_time_to_deliv...	avg_estimated_ti...	
▶	GO	26	16	12	
●	MG	23	12	13	
	SP	17	9	11	
●	MS	27	16	11	
	RJ	24	15	12	
●	MA	43	22	10	
	BA	30	19	11	
●	RS	25	15	14	
	PR	24	12	13	
●	SC	25	15	12	

#### 4),5) A.TOP 5 STATES WITH HIGHEST AVERAGE FREIGHT VALUE

##### Query -

```
WITH a AS (  
  SELECT order_id, DATEDIFF(order_delivered_customer_date,order_purchase_timestamp)  
  AS time_to_delivery,  
  DATEDIFF(order_estimated_delivery_date, order_delivered_customer_date) AS  
  diff_estimated_delivery  
  FROM orders),
```

```
  b as (  
    SELECT c.customer_state, oi.order_id,SUM(oi.freight_value) AS freight_value  
    FROM order_items oi  
    JOIN orders o using(order_id)  
    JOIN customers c using(customer_id)  
    GROUP BY oi.order_id, c.customer_state)
```

```
SELECT b.customer_state,ROUND(AVG(b.freight_value)) AS mean_freight_value,  
ROUND(AVG(a.time_to_delivery)) as avg_time_to_delivery,  
ROUND(AVG(a.diff_estimated_delivery)) as avg_diff_estimated_delivery  
FROM b  
JOIN a using(order_id)  
GROUP BY b.customer_state  
ORDER BY mean_freight_value DESC  
LIMIT 5;
```

##### Output -

	customer_state	mean_freight_value	avg_time_to_delivery	avg_diff_estimated_delivery	
▶	RR	49	29	17	
■	PB	48	20	13	
■	AC	46	21	21	
■	RO	46	19	20	
■	MA	43	22	10	

#### 5) B.TOP 5 STATES WITH LOWEST AVERAGE FREIGHT VALUE -

##### Query -

```
WITH a AS (  
  SELECT order_id, DATEDIFF(order_delivered_customer_date,order_purchase_timestamp)  
  AS time_to_delivery,  
  DATEDIFF(order_estimated_delivery_date, order_delivered_customer_date) AS  
  diff_estimated_delivery  
  FROM orders),
```

```
  b as (  
    SELECT c.customer_state, oi.order_id,SUM(oi.freight_value) AS freight_value  
    FROM order_items oi  
    JOIN orders o using(order_id)  
    JOIN customers c using(customer_id)
```

GROUP BY oi.order\_id, c.customer\_state)

```
SELECT b.customer_state,ROUND(AVG(b.freight_value)) AS mean_freight_value,  
ROUND(AVG(a.time_to_delivery)) as avg_time_to_delivery,  
ROUND(AVG(a.diff_estimated_delivery)) as avg_diff_estimated_delivery  
FROM b  
JOIN a using(order_id)  
GROUP BY b.customer_state  
ORDER BY mean_freight_value ASC  
LIMIT 5;
```

	customer_state	mean_freight_value	avg_time_to_delivery	avg_diff_estimated_delivery	
▶	SP	17	9	11	
	MG	23	12	13	
	RJ	24	15	12	
	DF	24	13	12	
	PR	24	12	13	

## 6) A.TOP 5 STATES WITH HIGHEST AVERAGE TIME TO DELIVERY

### Query -

```
WITH a AS (  
SELECT order_id, DATEDIFF(order_delivered_customer_date,order_purchase_timestamp)  
AS time_to_delivery,  
DATEDIFF(order_estimated_delivery_date, order_delivered_customer_date) AS  
diff_estimated_delivery  
FROM orders),
```

```
b as (  
SELECT c.customer_state, oi.order_id,SUM(oi.freight_value) AS freight_value  
FROM order_items oi  
JOIN orders o using(order_id)  
JOIN customers c using(customer_id)  
GROUP BY oi.order_id, c.customer_state)
```

```
SELECT b.customer_state,ROUND(AVG(b.freight_value)) AS mean_freight_value,  
ROUND(AVG(a.time_to_delivery)) as avg_time_to_delivery,  
ROUND(AVG(a.diff_estimated_delivery)) as avg_diff_estimated_delivery  
FROM b  
JOIN a using(order_id)  
GROUP BY b.customer_state  
ORDER BY avg_time_to_delivery DESC  
LIMIT 5;
```

## Output -

	customer_state	mean_freight_value	avg_time_to_delivery	avg_diff_estimated_delivery
	RR	49	29	17
	AP	41	27	20
	AM	37	26	20
	AL	39	25	9
	PA	40	24	14

## 6) A.TOP 5 STATES WITH LOWEST AVERAGE TIME TO DELIVERY

### Query -

```
WITH a AS (  
  SELECT order_id, DATEDIFF(order_delivered_customer_date,order_purchase_timestamp)  
  AS time_to_delivery,  
  DATEDIFF(order_estimated_delivery_date, order_delivered_customer_date) AS  
  diff_estimated_delivery  
  FROM orders),
```

```
b as (  
  SELECT c.customer_state, oi.order_id,SUM(oi.freight_value) AS freight_value  
  FROM order_items oi  
  JOIN orders o using(order_id)  
  JOIN customers c using(customer_id)  
  GROUP BY oi.order_id, c.customer_state)
```

```
SELECT b.customer_state,ROUND(AVG(b.freight_value)) AS mean_freight_value,  
ROUND(AVG(a.time_to_delivery)) as avg_time_to_delivery,  
ROUND(AVG(a.diff_estimated_delivery)) as avg_diff_estimated_delivery  
FROM b  
JOIN a using(order_id)  
GROUP BY b.customer_state  
ORDER BY avg_time_to_delivery ASC  
LIMIT 5;
```

## Output -

	customer_state	mean_freight_value	avg_time_to_delivery	avg_diff_estimated_delivery
►	SP	17	9	11
	PR	24	12	13
	MG	23	12	13
	DF	24	13	12
	RJ	24	15	12

## 6) A.TOP 5 STATES WHERE DELIVERY IS REALLY FAST

### Query -

```
WITH a AS (  
  SELECT order_id, DATEDIFF(order_delivered_customer_date,order_purchase_timestamp)  
  AS time_to_delivery,  
  DATEDIFF(order_estimated_delivery_date, order_delivered_customer_date) AS  
  diff_estimated_delivery  
  FROM orders),  
  
  b as (  
    SELECT c.customer_state, oi.order_id,SUM(oi.freight_value) AS freight_value  
    FROM order_items oi  
    JOIN orders o using(order_id)  
    JOIN customers c using(customer_id)  
    GROUP BY oi.order_id, c.customer_state)  
  
SELECT b.customer_state,ROUND(AVG(b.freight_value)) AS mean_freight_value,  
ROUND(AVG(a.time_to_delivery)) as avg_time_to_delivery,  
ROUND(AVG(a.diff_estimated_delivery)) as avg_diff_estimated_delivery  
FROM b  
JOIN a using(order_id)  
GROUP BY b.customer_state  
ORDER BY avg_diff_estimated_delivery ASC  
LIMIT 5;
```

### Output -

	customer_state	mean_freight_value	avg_time_to_delivery	avg_diff_estimated_delivery
▶	AL	39	25	9
■	ES	25	16	10
■	SE	41	21	10
■	MA	43	22	10
■	CE	36	21	11

## 6) B.TOP 5 STATES WHERE DELIVERY IS NOT SO FAST

### Query -

```
WITH a AS (  
  SELECT order_id, DATEDIFF(order_delivered_customer_date,order_purchase_timestamp)  
  AS time_to_delivery,  
  DATEDIFF(order_estimated_delivery_date, order_delivered_customer_date) AS  
  diff_estimated_delivery  
  FROM orders),  
  
  b as (  
    SELECT c.customer_state, oi.order_id,SUM(oi.freight_value) AS freight_value  
    FROM order_items oi  
    JOIN orders o using(order_id)  
    JOIN customers c using(customer_id)  
    GROUP BY oi.order_id, c.customer_state)
```

```

SELECT b.customer_state,ROUND(AVG(b.freight_value)) AS mean_freight_value,
ROUND(AVG(a.time_to_delivery)) as avg_time_to_delivery,
ROUND(AVG(a.diff_estimated_delivery)) as avg_diff_estimated_delivery
FROM b
JOIN a using(order_id)
GROUP BY b.customer_state
ORDER BY avg_diff_estimated_delivery DESC
LIMIT 5;

```

**Output -**

	customer_state	mean_freight_value	avg_time_to_delivery	avg_diff_estimated_delivery	
▶	AC	46	21	21	
	RO	46	19	20	
	AM	37	26	20	
	AP	41	27	20	
	RR	49	29	17	

## Q6. Payment Type Analysis

### 1) MONTH OVER MONTH COUNT OF ORDERS FOR DIFFERENT PAYMENT TYPES

Query -

```

SELECT YEAR(order_purchase_timestamp) AS year, MONTH(order_purchase_timestamp)
AS month, payment_type, count(payment_type) AS order_count
FROM orders o
JOIN payments p using(order_id)
GROUP BY year, month, payment_type
ORDER BY year, month;

```

Output -

	year	month	payment_type	order_count	
▶	2016	9	credit_card	3	
	2016	10	credit_card	254	
	2016	10	debit_card	2	
	2016	10	UPI	63	
	2016	10	voucher	23	
	2016	12	credit_card	1	
	2017	1	credit_card	583	
	2017	1	debit_card	9	
	2017	1	UPI	197	
	2017	1	voucher	61	

Inference -Highest No of payments through credit card

Recommendation - give attractive offers for other payment options

## 2) COUNT OF ORDERS BASED ON THE NO. OF PAYMENT INSTALLMENTS

Query -

```
SELECT payment_installments, count(order_id) AS order_count  
FROM payments  
GROUP BY payment_installments;
```

Output -

	payment_installments	order_count	
▶	8	4268	
■	1	52546	
■	2	12413	
■	3	10461	
■	6	3920	
■	5	5239	
■	4	7098	
■	10	5328	
■	7	1626	
■	12	133	

Inference - Most people prefer to pay in a single installment