


# Business Case: E-Commerce SQL

## Context:

XYZ.Ltd is one of the world's most recognized brands and one of America's leading E-Commerce retailers. XYZ.Ltd makes itself a preferred shopping destination by offering outstanding value, inspiration, innovation, and an exceptional guest experience that no other retailer can deliver.

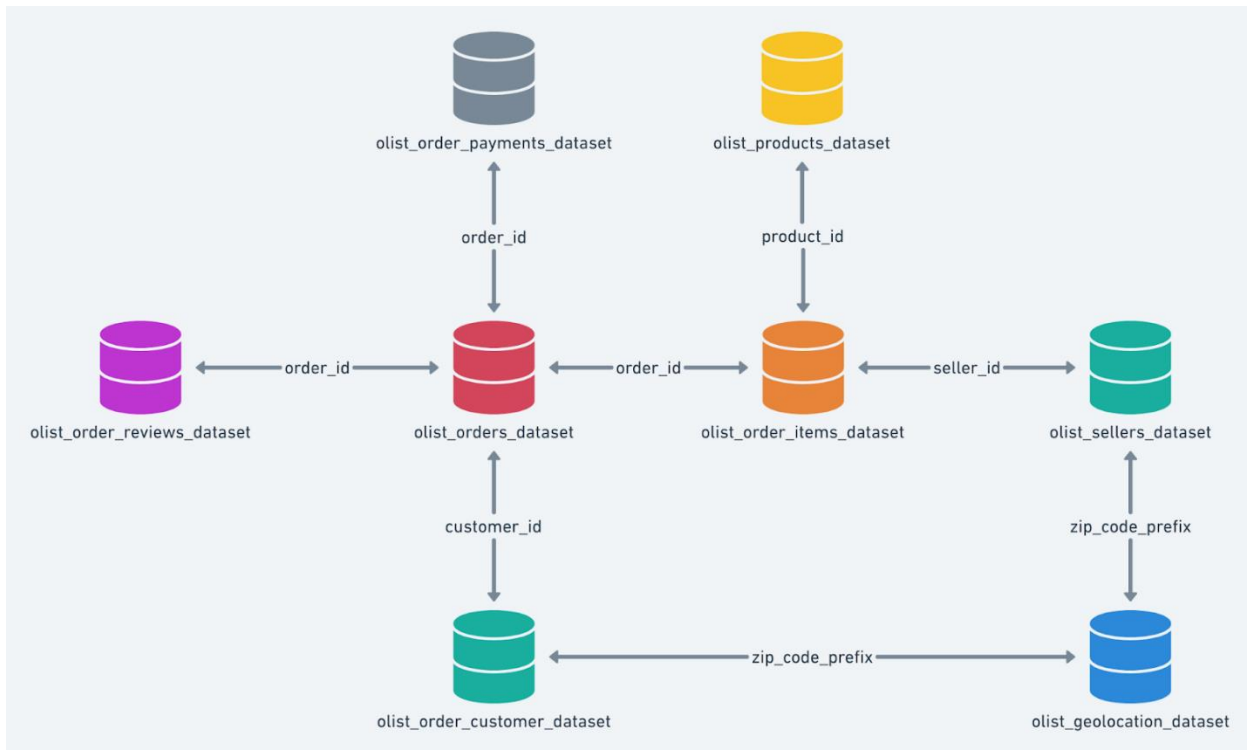
This business case has information of 100k orders from 2016 to 2018 made at XYZ.Ltd in Brazil. Its features allow viewing an order from multiple dimensions: from order status, price, payment and freight performance to customer location, product attributes and finally reviews written by customers.

Editor used to Query the DATA	 BigQuery Google Big Query Sandbox
Project Name	Business Case E-Commerce SQL
Project ID	business-case-e-commerce-sql
Data set Name	Ecomm_Data
Data set ID	business-case-e-commerce-sql.Ecomm_Data

## Data Set Details:

Table Name	Table ID
Customers	Ecomm_Data.customers
geolocation	Ecomm_Data.geolocation
order_items	Ecomm_Data.order_items
order_reviews	Ecomm_Data.order_reviews
orders	Ecomm_Data.orders
payments	Ecomm_Data.payments
products	Ecomm_Data.products
sellers	Ecomm_Data.sellers

High level overview of relationship between datasets:



## Initial exploration of dataset

1. Let us get an overview of the data we are dealing with i.e., column names and their data types in each table

SQL Query:

```
SELECT
  table_name,column_name,data_type
FROM
  `business-case-e-commerce-sql.Ecomm_Data`.INFORMATION_SCHEMA.COLUMNS
```

Query Result:

Row	table_name	column_name	data_type
1	order_items	order_id	STRING
2	order_items	order_item_id	INT64
3	order_items	product_id	STRING
4	order_items	seller_id	STRING
5	order_items	shipping_limit_date	TIMESTAMP
6	order_items	price	FLOAT64
7	order_items	freight_value	FLOAT64
8	sellers	seller_id	STRING
9	sellers	seller_zip_code_prefix	INT64
10	sellers	seller_city	STRING
11	sellers	seller_state	STRING
12	geolocation	geolocation_zip_code_prefix	INT64
13	geolocation	geolocation_lat	FLOAT64
14	geolocation	geolocation_lng	FLOAT64
15	geolocation	geolocation_city	STRING

Row	table_name	column_name	data_type
16	geolocation	geolocation_state	STRING
17	products	product_id	STRING
18	products	product_category	STRING
19	products	product_name_length	INT64
20	products	product_description_length	INT64
21	products	product_photos_qty	INT64
22	products	product_weight_g	INT64
23	products	product_length_cm	INT64
24	products	product_height_cm	INT64
25	products	product_width_cm	INT64
26	orders	order_id	STRING
27	orders	customer_id	STRING
28	orders	order_status	STRING
29	orders	order_purchase_timestamp	TIMESTAMP
30	orders	order_approved_at	TIMESTAMP

Row	table_name	column_name	data_type
31	orders	order_delivered_carrier_date	TIMESTAMP
32	orders	order_delivered_customer_date	TIMESTAMP
33	orders	order_estimated_delivery_date	TIMESTAMP
34	payments	order_id	STRING
35	payments	payment_sequential	INT64
36	payments	payment_type	STRING
37	payments	payment_installments	INT64
38	payments	payment_value	FLOAT64
39	customers	customer_id	STRING
40	customers	customer_unique_id	STRING
41	customers	customer_zip_code_prefix	INT64
42	customers	customer_city	STRING
43	customers	customer_state	STRING
44	order_reviews	review_id	STRING
45	order_reviews	order_id	STRING
46	order_reviews	review_score	INT64
47	order_reviews	review_comment_title	STRING
48	order_reviews	review_creation_date	TIMESTAMP
49	order_reviews	review_answer_timestamp	TIMESTAMP

The above results help us to better understand the tables and their columns which will further help us in querying the data

## 2. What is the time period for which the data has been provided??

To get to know the time period of the data, we need get start date and end date of the data and the difference between them

SQL Query:

```
SELECT
EXTRACT(DATE FROM MIN(order_purchase_timestamp)) AS start_date,
EXTRACT(DATE FROM MAX(order_purchase_timestamp)) AS end_date,
CONCAT(TIMESTAMP_DIFF(MAX(order_purchase_timestamp),MIN(order_purchase_timestamp),day)," days") AS time_period_of_data
FROM `Ecomm_Data.orders`
```

Query Result:

Row	start_date	end_date	time_period_of_data
1	2016-09-04	2018-10-17	772 days

3. Provide list of all the states and their cites from which customers ordered

SQL Query:

```
SELECT DISTINCT customer_state, customer_city
FROM `Ecomm_Data.customers`
order by customer_state, customer_city
```

Query Result:

Row	customer_state	customer_city
1	AC	brasileia
2	AC	cruzeiro do sul
3	AC	epitaciolandia
4	AC	manoel urbano
5	AC	porto acre
6	AC	rio branco
7	AC	senador guiomard
8	AC	xapuri
9	AL	agua branca
10	AL	anadia
11	AL	arapiraca
12	AL	atalaia
13	AL	barra de santo antonio

Load more

Results per page: 50 1 – 50 of 4310

4. What is the count of distinct states and cities of customers

SQL Query:

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

Query Result:

Row	total_states	total_cities
1	27	4119

So, there are total 27 states and 4119 cities from where customers have ordered products.

5. Provide list of all the product categories

SQL Query:

```
SELECT DISTINCT product_category  
FROM `Ecomm_Data.products`  
ORDER BY product_category
```

Query Result:

Row	product_category
1	null
2	Agro Industria e Comercio
3	Art
4	Arts and Crafts
5	Bags Accessories
6	Blu Ray DVDs
7	CITTE AND UPHACK FURNITURE
8	CONSTRUCTION SECURITY TO...
9	Casa Construc�o
10	Christmas articles
11	Construction Tools Construction
12	Construction Tools Garden
13	Construction Tools Illumination
14	Construction Tools Tools
15	Cool Stuff

Results per page: 50 1 – 50 of 74

6. In how many categories are products categorized?

SQL Query:

```
SELECT COUNT(DISTINCT product_category) AS no_of_product_categories
FROM `Ecomm_Data.products`
```

Query Result:

Row	no_of_product_categories
1	73

There are total 73 Product categories

7. List down top ten customers id who have contributed highest to the sales.

SQL Query:

```
SELECT
  o.customer_id,
  SUM(p.payment_value) AS Total_order_amount
FROM `Ecomm_Data.orders` As o
```

```

JOIN `Ecomm_Data.payments` As p
ON o.order_id = p.order_id
GROUP BY o.customer_id
ORDER BY Total_order_amount DESC
LIMIT 10

```

Query Result:

Row	customer_id	Total_order_amount
1	1617b1357756262bfa56ab541c47bc...	13664.08
2	ec5b2ba62e574342386871631fafd3fc	7274.88
3	c6e2731c5b391845f6800c97401a43...	6929.31
4	f48d464a0baaea338cb25f816991ab1f	6922.21
5	3fd6777bbce08a352fddd04e4a7cc8f6	6726.66
6	05455dfa7cd02f13d132aa7a6a9729c6	6081.54
7	df55c14d1476a9a3467f131269c2477f	4950.34
8	e0a2412720e9ea4f26c1ac985f6a7358	4809.44
9	24bbf5fd2f2e1b359ee7de94defc4a15	4764.34
10	3d979689f636322c62418b6346b1c6...	4681.78

## In-depth Exploration

### 1. Is there growing trend on e-commerce in Brazil ?

We can come to know whether e-commerce trend is growing in Brazil by getting yearly sales and checking whether the sales are increasing or decreasing

SQL Query:

```

SELECT EXTRACT(year FROM order_purchase_timestamp) AS year,
ROUND(SUM(payment_value),2) as sales,
FROM `Ecomm_Data.orders` as o
JOIN `Ecomm_Data.payments` as p
ON o.order_id = p.order_id
GROUP BY year
ORDER BY year

```

Query Result:



Row	year	sales
1	2016	59362.34
2	2017	7249746.73
3	2018	8699763.05

From the above table we can definitely say the trend of e-commerce is growing in Brazil.

## 2. Get monthly sales for each year

SQL Query:

```
SELECT EXTRACT(year FROM order_purchase_timestamp) AS year,
       EXTRACT(month FROM order_purchase_timestamp) AS month,
       FORMAT_DATETIME("%B", DATETIME(order_purchase_timestamp)) AS month_name,
       ROUND(SUM(payment_value),2) AS sales,
FROM `Ecomm_Data.orders` AS o
JOIN `Ecomm_Data.payments` AS p
ON o.order_id = p.order_id
GROUP BY year,month_name,month
ORDER BY year,month
```

Query Result:

Row	year	month	month_name	sales
1	2016	9	September	252.24
2	2016	10	October	59090.48
3	2016	12	December	19.62
4	2017	1	January	138488.04
5	2017	2	February	291908.01
6	2017	3	March	449863.6
7	2017	4	April	417788.03
8	2017	5	May	592918.82
9	2017	6	June	511276.38
10	2017	7	July	592382.92
11	2017	8	August	674396.32
12	2017	9	September	727762.45
13	2017	10	October	779677.88
14	2017	11	November	1194882.8
15	2017	12	December	878401.18

The query gives us sales per month.

### 3. Find out top 2 months with highest sales each year

SQL Query:

```
SELECT *
FROM(
  SELECT year,month,sales, DENSE_RANK() OVER(PARTITION BY year ORDER BY sales DESC) AS
sales_rank
  FROM(
    SELECT EXTRACT(year FROM order_purchase_timestamp) AS year,
      FORMAT_DATETIME("%B", DATETIME(order_purchase_timestamp)) AS month,
      ROUND(SUM(payment_value),2) AS sales,
    FROM `Ecomm_Data.orders` AS o
    JOIN `Ecomm_Data.payments` AS p
    ON o.order_id = p.order_id
    GROUP BY year,month)) AS a
WHERE a.sales_rank < 3
ORDER BY year,sales_rank
```

Query Result:

Row	year	month	sales	sales_rank
1	2016	October	59090.48	1
2	2016	September	252.24	2
3	2017	November	1194882.8	1
4	2017	December	878401.48	2
5	2018	April	1160785.48	1
6	2018	March	1159652.12	2

4. What time do Brazilian customers tend to buy (Dawn, Morning, Afternoon or Night)?

To get to know Brazilian customers tend to buy most of the time we have look into order purchase timings

SQL Query:

```
SELECT
*,DENSE_RANK() OVER(ORDER BY no_of_purchase_orders DESC) AS highest_orders_rank
FROM(SELECT
CASE
WHEN EXTRACT (time from order_purchase_timestamp) BETWEEN "00:00:00" AND "06:00:00"
THEN "dawn"
WHEN EXTRACT (time from order_purchase_timestamp) BETWEEN "06:00:01" AND "12:00:00"
THEN "morning"
WHEN EXTRACT (time from order_purchase_timestamp) BETWEEN "12:00:01" AND "18:00:00"
THEN "afternoon"
ELSE "night"
END AS time_in_day, COUNT(1) as no_of_purchase_orders,
from `Ecomm_Data.orders`
group by time_in_day)
ORDER BY no_of_purchase_orders DESC
```

Query Result:

Row	time_in_day	no_of_purchase_orders	highest_orders_rank
1	afternoon	38365	1
2	night	34096	2
3	morning	22240	3
4	dawn	4740	4

From the above result we can clearly say that Brazilian customers tend to buy more during afternoon and night times.

5. Get the months where sales where higher than the previous month along with the sales values.

SQL Query:

```
SELECT *
FROM(
SELECT *,
LAG(month_name) OVER(ORDER BY year,month) AS prev_month_name,
LAG(sales,1) OVER(ORDER BY year,month) AS prev_month_sales
FROM(
```

```

SELECT EXTRACT(year FROM order_purchase_timestamp) AS year,
       EXTRACT(month FROM order_purchase_timestamp) AS month,
       FORMAT_DATETIME("%B", DATETIME(order_purchase_timestamp)) AS month_name,
       ROUND(SUM(payment_value),2) AS sales
FROM `Ecomm_Data.orders` AS o
JOIN `Ecomm_Data.payments` AS p
ON o.order_id = p.order_id
GROUP BY year,month_name,month
) AS t1 ) AS t
WHERE t.sales > t.prev_month_sales
ORDER BY year,month

```

Query Result:

Row	year	month	month_name	sales	prev_month_name	prev_month_sales
1	2016	10	October	59090.48	September	252.24
2	2017	1	January	138488.04	December	19.62
3	2017	2	February	291908.01	January	138488.04
4	2017	3	March	449863.6	February	291908.01
5	2017	5	May	592918.82	April	417788.03
6	2017	7	July	592382.92	June	511276.38
7	2017	8	August	674396.32	July	592382.92
8	2017	9	September	727762.45	August	674396.32
9	2017	10	October	779677.88	September	727762.45

Results per page: 50 1 – 14 of 14

## Evolution of E-commerce orders in the Brazil region

1. Get month on month orders by states

To get state wise month on month orders count we have to firstly join orders and customers table and then group by states, year and month and then count the number of orders per month per state

SQL Query:

```
SELECT
c.customer_state AS states,
EXTRACT(year FROM order_purchase_timestamp) AS year,
EXTRACT(month FROM order_purchase_timestamp) AS month,
FORMAT_DATETIME("%B", DATETIME(order_purchase_timestamp)) AS month_name,
COUNT(o.order_id) AS no_of_orders
FROM `Ecomm_Data.customers` AS c
LEFT JOIN `Ecomm_Data.orders` AS o
ON c.customer_id = o.customer_id
GROUP BY c.customer_state,year,month_name,month
ORDER BY c.customer_state,year,month
```

Query Result:

Row	states	year	month	month_name	no_of_orders
1	AC	2017	1	January	2
2	AC	2017	2	February	3
3	AC	2017	3	March	2
4	AC	2017	4	April	5
5	AC	2017	5	May	8
6	AC	2017	6	June	4
7	AC	2017	7	July	5
8	AC	2017	8	August	4
9	AC	2017	9	September	5
10	AC	2017	10	October	6
11	AC	2017	11	November	5
12	AC	2017	12	December	5
13	AC	2018	1	January	6
14	AC	2018	2	February	3
15	AC	2018	3	March	2

## 2. Distribution of customers across the states in Brazil

We can get this information by finding the percentage of total customers in states of Brazil

SQL Query:

```
SELECT customer_state, COUNT(*) AS no_of_customers_per_state,  
ROUND((((COUNT(*)/(SELECT COUNT(*) FROM `Ecomm_Data.customers`))*100),2) AS percent_  
distribution_of_customers  
FROM `Ecomm_Data.customers`  
GROUP BY customer_state  
ORDER BY percent_distribution_of_customers DESC
```

Query result:

Row	customer_state	no_of_customers_per_state	percent_distribution_of_customers
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
13	PA	975	0.98
14	MT	907	0.91

From above results we can clearly say that states SP, RJ, MG contribute to approx. 65% of all customers

3. What is over all combined average delivery timings for the states SP, RJ, MG

SQL Query:

```
SELECT
CONCAT(ROUND(AVG(TIMESTAMP_DIFF(order_delivered_customer_date,order_purchase_timestamp,day)),2)," days") AS Average_time_to_delivery_SP_RJ_MG
FROM `Ecomm_Data.orders` AS o
JOIN `Ecomm_Data.customers` AS c
ON o.customer_id = c.customer_id
WHERE c.customer_state IN ("SP","RJ","MG")
```

Query Result:

Row	Average_time_to_delivery_SP_RJ_MG
1	10.13 days

## Impact on Economy

1. Get % increase in cost of orders from 2017 to 2018 (include months between Jan to Aug only) - You can use "payment\_value" column in payments table

To find the % increase in cost of orders from 2017 to 2018 ,we must first find out the total cost of orders for 2017 and 2018 for the months Jan to Aug

SQL Query:

```
SELECT
(EXTRACT(year FROM order_purchase_timestamp)) AS year,
ROUND(SUM(payment_value),2) AS cost_of_orders_Jan_to_Aug
FROM `Ecomm_Data.orders` AS o
JOIN `Ecomm_Data.payments` AS p
ON o.order_id = p.order_id
WHERE (EXTRACT(month FROM order_purchase_timestamp) BETWEEN 1 AND 8) AND
      (EXTRACT(year FROM order_purchase_timestamp) BETWEEN 2017 AND 2018)
GROUP BY year
ORDER BY year
```

Query Result:

Row	year	cost_of_orders_Jan_to_Aug
1	2017	3669022.12
2	2018	8694733.84

Now we can find the percentage increase in the cost of orders by using below formula

percentage increase = increase ÷ original number × 100

here,

increase = 2018 cost of orders – 2017 cost of orders

increase = 8694733.84 - 3669022.12

increase = 5025711.72

original number = 3669022.12

percentage increase = 5025711.72 ÷ 3669022.12 \* 100

percentage increase = 136.98%

we can clearly see there huge increase of 137% from 2017 to 2018(Jan to Aug) , which clearly indicates that e-commerce section of target is growing rapidly and people of Brazil adapting faster to online shopping and even the overall purchasing power has increased.

## 2. Mean & Sum of price and freight value by customer state

To get customer state wise Mean and sum values for price and freight value we need to join customers, orders and order\_item tables

SQL Query:

```
SELECT
c.customer_state,ROUND(AVG(oi.price),2) AS mean_price, ROUND(SUM(oi.price),2) AS sum_of
_price,
ROUND(AVG(oi.freight_value),2) AS mean_freight_value,ROUND(SUM(oi.freight_value),2) AS s
um_of_freight_value
FROM `Ecomm_Data.customers` AS c
JOIN `Ecomm_Data.orders` AS o
ON c.customer_id = o.customer_id
JOIN `Ecomm_Data.order_items` AS oi
```



```

ON o.order_id = oi.order_id
GROUP BY c.customer_state
ORDER BY c.customer_state

```

Query Result:

Row	customer_state	mean_price	sum_of_price	mean_freight_value	sum_of_freight_value
10	MA	145.2	119648.22	38.26	31523.77
11	MG	120.75	1585308.03	20.63	270853.46
12	MS	142.63	116812.64	23.37	19144.03
13	MT	148.3	156453.53	28.17	29715.43
14	PA	165.69	178947.81	35.83	38699.3
15	PB	191.48	115268.08	42.72	25719.73
16	PE	145.51	262788.03	32.92	59449.66
17	PI	160.36	86914.08	39.15	21218.2
18	PR	119.0	683083.76	20.53	117851.68
19	RJ	125.12	1824092.67	20.96	305589.31
20	RN	156.97	83034.98	35.65	18860.1
21	RO	165.97	46140.64	41.07	11417.38
22	RR	150.57	7829.43	42.98	2235.19
23	RS	120.34	750304.02	21.74	135522.74
24	SC	124.65	520553.34	21.47	89660.26

Results per page: 50 1 - 27 of 27

## Analysis on sales, freight, and delivery time

1. Calculate days between purchasing, delivering and estimated delivery

SQL Query:

```

SELECT order_id,
order_purchase_timestamp,order_delivered_customer_date,
order_estimated_delivery_date,
CONCAT(TIMESTAMP_DIFF(order_delivered_customer_date,order_purchase_timestamp,day),"
days") AS days_bw_purchase_delivery,

```

```

CONCAT(TIMESTAMP_DIFF(order_estimated_delivery_date,order_purchase_timestamp,day),"
days") AS days_bw_purchase_estimated_delivery
FROM `Ecomm_Data.orders`
WHERE order_delivered_customer_date IS NOT NULL

```

Query Result:

Row	order_id	order_purchase_timestamp	order_delivered_customer_date	order_estimated_delivery_date	days_bw_purchase_delivery	days_bw_purchase_estimated_delivery
1	770d331...	2016-10-07 14:...	2016-10-14 15:0...	2016-11-29 00:...	7 days	52 days
2	2c45c33...	2016-10-09 15:...	2016-11-09 14:5...	2016-12-08 00:...	30 days	59 days
3	dabf2b0...	2016-10-09 00:...	2016-10-16 14:3...	2016-11-30 00:...	7 days	51 days
4	8beb593...	2016-10-08 20:...	2016-10-19 18:4...	2016-11-30 00:...	10 days	52 days
5	65d1e22...	2016-10-03 21:...	2016-11-08 10:5...	2016-11-25 00:...	35 days	52 days
6	cec8f5f7...	2017-03-17 15:...	2017-04-07 13:1...	2017-05-18 00:...	20 days	61 days
7	58527ee...	2017-03-20 11:...	2017-03-30 14:0...	2017-05-18 00:...	10 days	58 days
8	10ed549...	2017-03-21 13:...	2017-04-18 13:5...	2017-05-18 00:...	28 days	57 days
9	818996e...	2018-08-20 15:...	2018-08-29 22:5...	2018-10-04 00:...	9 days	44 days
10	d195cac...	2018-08-12 18:...	2018-08-23 02:0...	2018-10-04 00:...	10 days	52 days
11	64eeb35...	2018-08-16 07:...	2018-08-23 00:0...	2018-10-04 00:...	6 days	48 days
12	2691ae8...	2018-08-22 22:...	2018-08-29 19:1...	2018-10-04 00:...	6 days	42 days
13	1cd147d...	2018-08-20 17:...	2018-08-29 16:4...	2018-10-04 00:...	8 days	44 days

Results per page: 50 1 – 50 of 96476

2. Find time\_to\_delivery & diff\_estimated\_delivery. Formula for the same given below:

- time\_to\_delivery = order\_purchase\_timestamp - order\_delivered\_customer\_date
- diff\_estimated\_delivery = order\_estimated\_delivery\_date - order\_delivered\_customer\_date

a. SQL Query:

```

SELECT order_id,
order_purchase_timestamp,order_delivered_customer_date,
CONCAT(TIMESTAMP_DIFF(order_delivered_customer_date,order_purchase_timestamp,da
y)," days") AS Time_to_delivery,

```

```
FROM `Ecomm_Data.orders`
WHERE order_delivered_customer_date IS NOT NULL
```

Query Result:

Row	order_id	order_purchase_timestamp	order_delivered_customer_date	Time_to_delivery
1	1950d777989f6a877539f5379...	2018-02-19 19:48:52 UTC	2018-03-21 22:03:51 UTC	30 days
2	2c45c33d2f9cb8ff8b1c86cc28...	2016-10-09 15:39:56 UTC	2016-11-09 14:53:50 UTC	30 days
3	65d1e226dfaeb8cdc42f66542...	2016-10-03 21:01:41 UTC	2016-11-08 10:58:34 UTC	35 days
4	635c894d068ac37e6e03dc54e...	2017-04-15 15:37:38 UTC	2017-05-16 14:49:55 UTC	30 days
5	3b97562c3aee8bdedcb5c2e45...	2017-04-14 22:21:54 UTC	2017-05-17 10:52:15 UTC	32 days
6	68f47f50f04c4cb6774570cfde...	2017-04-16 14:56:13 UTC	2017-05-16 09:07:47 UTC	29 days
7	276e9ec344d3bf029ff83a161c...	2017-04-08 21:20:24 UTC	2017-05-22 14:11:31 UTC	43 days
8	54e1a3c2b97fb0809da548a59...	2017-04-11 19:49:45 UTC	2017-05-22 16:18:42 UTC	40 days
9	fd04fa4105ee8045f6a0139ca5...	2017-04-12 12:17:08 UTC	2017-05-19 13:44:52 UTC	37 days
10	302bb8109d097a9fc6e9cefc5...	2017-04-19 22:52:59 UTC	2017-05-23 14:19:48 UTC	33 days
11	66057d37308e787052a32828...	2017-04-15 19:22:06 UTC	2017-05-24 08:11:57 UTC	38 days
12	19135c945c554eebfd7576c73...	2017-07-11 14:09:37 UTC	2017-08-16 20:19:32 UTC	36 days
13	4493e45e7ca1084efcd38ddeb...	2017-07-11 20:56:34 UTC	2017-08-14 21:37:08 UTC	34 days
14	70c77e51e0f179d75a64a6141...	2017-07-13 21:03:44 UTC	2017-08-25 19:41:53 UTC	42 days

Results per page: 50 ▼ 1 – 50 of 96476 |<

b. SQL Query:

```
SELECT order_id,
order_estimated_delivery_date,order_delivered_customer_date,
CONCAT(TIMESTAMP_DIFF(order_estimated_delivery_date,order_delivered_customer_date,day)," days") AS diff_estimated_delivery ,
FROM `Ecomm_Data.orders`
WHERE order_delivered_customer_date IS NOT NULL
```

Query Result:

Row	order_id	order_estimated_delivery_date	order_delivered_customer_date	diff_estimated_delivery
1	770d331c84e5b214bd9dc70a1...	2016-11-29 00:00:00 UTC	2016-10-14 15:07:11 UTC	45 days
2	1950d777989f6a877539f5379...	2018-03-09 00:00:00 UTC	2018-03-21 22:03:51 UTC	-12 days
3	dabf2b0e35b423f94618bf965f...	2016-11-30 00:00:00 UTC	2016-10-16 14:36:59 UTC	44 days
4	8beb59392e21af5eb9547ae1a...	2016-11-30 00:00:00 UTC	2016-10-19 18:47:43 UTC	41 days
5	b60b53ad0bb7dacacf2989fe2...	2017-05-18 00:00:00 UTC	2017-05-23 13:12:27 UTC	-5 days
6	276e9ec344d3bf029ff83a161c...	2017-05-18 00:00:00 UTC	2017-05-22 14:11:31 UTC	-4 days
7	1a0b31f08d0d7e87935b819ed...	2017-05-18 00:00:00 UTC	2017-04-18 08:18:11 UTC	29 days
8	cec8f5f7a13e5ab934a486ec9e...	2017-05-18 00:00:00 UTC	2017-04-07 13:14:56 UTC	40 days
9	54e1a3c2b97fb0809da548a59...	2017-05-18 00:00:00 UTC	2017-05-22 16:18:42 UTC	-4 days
10	58527ee4726911bee84a0f42c...	2017-05-18 00:00:00 UTC	2017-03-30 14:04:04 UTC	48 days
11	302bb8109d097a9fc6e9cefc5...	2017-05-18 00:00:00 UTC	2017-05-23 14:19:48 UTC	-5 days
12	10ed5499d1623638ee810eff1...	2017-05-18 00:00:00 UTC	2017-04-18 13:52:43 UTC	29 days
13	cb837ba275cf8ffa9ded7e18f7...	2017-05-18 00:00:00 UTC	2017-05-22 20:35:54 UTC	-4 days
14	66057d37308e787052a32828...	2017-05-18 00:00:00 UTC	2017-05-24 08:11:57 UTC	-6 days

Results per page: 50 1 – 50 of 96476

- Group data by state, take mean of freight\_value, time\_to\_delivery, diff\_estimated\_delivery

SQL Query:

```

SELECT
c.customer_state,
ROUND(AVG(oi.freight_value),2) AS mean_freight_value,
CONCAT(ROUND(AVG(TIMESTAMP_DIFF(order_delivered_customer_date,order_purchase_timestamp,day)),2)," days") AS mean_time_to_delivery,
CONCAT(ROUND(AVG(TIMESTAMP_DIFF(order_estimated_delivery_date,order_delivered_customer_date,day)),2)," days") AS mean_diff_estimated_delivery
FROM `Ecomm_Data.customers` AS c
JOIN `Ecomm_Data.orders` AS o
ON c.customer_id = o.customer_id
JOIN `Ecomm_Data.order_items` AS oi
ON o.order_id = oi.order_id
GROUP BY c.customer_state
ORDER BY c.customer_state

```

#### Query Result:

Row	customer_state	mean_freight_value	mean_time_to_delivery	mean_diff_estimated_delivery
1	AC	40.07	20.33 days	20.01 days
2	AL	35.84	23.99 days	7.98 days
3	AM	33.21	25.96 days	18.98 days
4	AP	34.01	27.75 days	17.44 days
5	BA	26.36	18.77 days	10.12 days
6	CE	32.71	20.54 days	10.26 days
7	DF	21.04	12.5 days	11.27 days
8	ES	22.06	15.19 days	9.77 days
9	GO	22.77	14.95 days	11.37 days
10	MA	38.26	21.2 days	9.11 days
11	MG	20.63	11.52 days	12.4 days
12	MS	23.37	15.11 days	10.34 days
13	MT	28.17	17.51 days	13.64 days
14	PA	35.83	23.3 days	13.37 days
15	PB	42.72	20.12 days	12.15 days

Results per page: 50 ▼ 1 – 27 of 27

4. What is the overall mean of freight\_value, time\_to\_delivery, diff\_estimated\_delivery

SQL Query:

```
SELECT
ROUND(AVG(oi.freight_value),2) AS mean_freight_value,
CONCAT(ROUND(AVG(TIMESTAMP_DIFF(order_delivered_customer_date,order_purchase_timestamp,day)),2)," days") AS mean_time_to_delivery,
CONCAT(ROUND(AVG(TIMESTAMP_DIFF(order_estimated_delivery_date,order_delivered_customer_date,day)),2)," days") AS mean_diff_estimated_delivery
FROM `Ecomm_Data.orders` AS o
JOIN `Ecomm_Data.order_items` AS oi
ON o.order_id = oi.order_id
```

Query Result:

Row	mean_freight_va	mean_time_to_delivery	mean_diff_estimated_delivery
1	19.99	12.01 days	11.11 days

Considering present time for the above result we can clearly say that there is still large scope to improve delivery network.

## 5. Get Top 5 states with highest/lowest average freight value

SQL Query:

(For top 5 Highest)

```
SELECT  
c.customer_state,ROUND(AVG(oi.freight_value),2) AS Top5_Highest_Average_freight_value  
FROM `Ecomm_Data.customers` AS c  
JOIN `Ecomm_Data.orders` AS o  
ON c.customer_id = o.customer_id  
JOIN `Ecomm_Data.order_items` AS oi  
ON o.order_id = oi.order_id  
GROUP BY c.customer_state  
ORDER BY Top5_Highest_Average_freight_value DESC  
LIMIT 5
```

Query Result:

Row	customer_state	Top5_Highest_Average_freight_value
1	RR	42.98
2	PB	42.72
3	RO	41.07
4	AC	40.07
5	PI	39.15

SQL Query:

(For top 5 Lowest)

```
SELECT
c.customer_state,ROUND(AVG(oi.freight_value),2) AS Top5_Lowest_Average_freight_value
FROM `Ecomm_Data.customers` AS c
JOIN `Ecomm_Data.orders` AS o
ON c.customer_id = o.customer_id
JOIN `Ecomm_Data.order_items` AS oi
ON o.order_id = oi.order_id
GROUP BY c.customer_state
ORDER BY Top5_Lowest_Average_freight_value ASC
LIMIT 5
```

Query Result:

Row	customer_state	Top5_Lowest_Average_freight_value
1	SP	15.15
2	PR	20.53
3	MG	20.63
4	RJ	20.96
5	DF	21.04

## 6. Top 5 states with highest/lowest average time to delivery

SQL Query:

(For Top 5 Highest time to delivery)

```
SELECT
c.customer_state,
CONCAT(ROUND(AVG(TIMESTAMP_DIFF(order_delivered_customer_date,order_purchase_timestamp,day)),2)," days") AS Top5_Highest_Average_time_to_delivery
FROM `Ecomm_Data.customers` AS c
JOIN `Ecomm_Data.orders` AS o
ON c.customer_id = o.customer_id
```

```

GROUP BY c.customer_state
ORDER BY ROUND(AVG(TIMESTAMP_DIFF(order_delivered_customer_date,order_purchase_timestamp,day))) DESC
LIMIT 5

```

Query Result:

Row	customer_state	Top5_Highest_Average_time_to_delivery
1	RR	28.98 days
2	AP	26.73 days
3	AM	25.99 days
4	AL	24.04 days
5	PA	23.32 days

SQL Query:

(For Top 5 Lowest time to delivery)

```

SELECT
c.customer_state,
CONCAT(ROUND(AVG(TIMESTAMP_DIFF(order_delivered_customer_date,order_purchase_timestamp,day)),2)," days") AS Top5_Lowest_Average_time_to_delivery
FROM `Ecomm_Data.customers` AS c
JOIN `Ecomm_Data.orders` AS o
ON c.customer_id = o.customer_id
GROUP BY c.customer_state
ORDER BY ROUND(AVG(TIMESTAMP_DIFF(order_delivered_customer_date,order_purchase_timestamp,day))) ASC
LIMIT 5

```

Query Result:



Row	customer_state	Top5_Lowest_Average_time_to_delivery
1	SP	8.3 days
2	PR	11.53 days
3	MG	11.54 days
4	DF	12.51 days
5	SC	14.48 days

## 7. Top 5 states where delivery is really fast/ not so fast compared to estimated date

Here in this case we will again use the concept  $\text{diff\_estimated\_delivery} = \text{order\_estimated\_delivery\_date} - \text{order\_delivered\_customer\_date}$ , which will provide us the difference between estimated date and actual date on which the order was delivered from which we can find out states in which delivery is fast and not fast

SQL Query:

(For states with fast delivery)

```
SELECT
c.customer_state,
CONCAT(ROUND(AVG(TIMESTAMP_DIFF(order_estimated_delivery_date,order_delivered_customer_date,day)),2)," days") AS Lowest_difference_in_delivery_timing
FROM `Ecomm_Data.customers` AS c
JOIN `Ecomm_Data.orders` AS o
ON c.customer_id = o.customer_id
GROUP BY c.customer_state
ORDER BY ROUND(AVG(TIMESTAMP_DIFF(order_estimated_delivery_date,order_delivered_customer_date,day)),2) ASC
LIMIT 5
```

Query Result:

Row	customer_state	Lowest_difference_in_delivery_timing
1	AL	7.95 days
2	MA	8.77 days
3	SE	9.17 days
4	ES	9.62 days
5	BA	9.93 days

SQL Query:

(For States with not so fast delivery)

```

SELECT
c.customer_state,
CONCAT(ROUND(AVG(TIMESTAMP_DIFF(order_estimated_delivery_date,order_delivered_customer_date,day)),2)," days") AS Highest_difference_in_delivery_timing
FROM `Ecomm_Data.customers` AS c
JOIN `Ecomm_Data.orders` AS o
ON c.customer_id = o.customer_id
GROUP BY c.customer_state
ORDER BY ROUND(AVG(TIMESTAMP_DIFF(order_estimated_delivery_date,order_delivered_customer_date,day)),2) DESC
LIMIT 5

```

Query Result:

Row	customer_state	Highest_difference_in_delivery_timing
1	AC	19.76 days
2	RO	19.13 days
3	AP	18.73 days
4	AM	18.61 days
5	RR	16.41 days

## Payment type analysis

1. Month over Month count of orders for different payment types

SQL Query:

```
SELECT EXTRACT(year FROM order_purchase_timestamp) AS year,
       EXTRACT(month FROM order_purchase_timestamp) AS month,
       FORMAT_DATETIME("%B", DATETIME(order_purchase_timestamp)) AS month_name,
       payment_type,
       COUNT(payment_type) AS count_of_payment_type
FROM `Ecomm_Data.orders` AS o
JOIN `Ecomm_Data.payments` AS p
ON o.order_id = p.order_id
GROUP BY year, month_name, month, payment_type
ORDER BY year, month, payment_type
```

Query Result:

Row	year	month	month_name	payment_type	count_of_payment_type
1	2016	9	September	credit_card	3
2	2016	10	October	UPI	63
3	2016	10	October	credit_card	254
4	2016	10	October	debit_card	2
5	2016	10	October	voucher	23
6	2016	12	December	credit_card	1
7	2017	1	January	UPI	197
8	2017	1	January	credit_card	583
9	2017	1	January	debit_card	9
10	2017	1	January	voucher	61
11	2017	2	February	UPI	398
12	2017	2	February	credit_card	1356
13	2017	2	February	debit_card	13
14	2017	2	February	voucher	119

Results per page: 50 ▼ 1 – 50 of 90 |< < > >

## 2. Count of orders based on the no. of payment installments

SQL Query:

```
SELECT payment_installments,COUNT(*) AS count_of_orders
FROM `Ecomm_Data.payments`
GROUP BY payment_installments
ORDER BY payment_installments
```

Query Result:

Row	payment_installments	count_of_orders
1	0	2
2	1	52546
3	2	12413
4	3	10461
5	4	7098
6	5	5239
7	6	3920
8	7	1626
9	8	4268
10	9	644
11	10	5328
12	11	23
13	12	133
14	13	16

Results per page: 50 ▼ 1 – 24 of 24 |< <

## 3. Over all Count of orders based on payment type

SQL Query:

```
SELECT payment_type,COUNT(*) AS count_of_orders
FROM `Ecomm_Data.payments`
GROUP BY payment_type
ORDER BY COUNT(*) DESC
```

Query Result:

Row	payment_type	count_of_orders
1	credit_card	76795
2	UPI	19784
3	voucher	5775
4	debit_card	1529
5	not_defined	3

From above result we can clearly say that credit card payment method accounts for more than 70% of order payments

## **Actionable Insights and Recommendations**

### 1. Insight:

There are total 27 states and 4119 cities from where customers have ordered products And  
There are total 73 Product categories

### 2. Insight:

Considering the year-on-year sales we can definitely say that e-commerce in Brazil is rapidly growing.

#### Recommendations:

We should take advantage of this e-commerce boom and try to focus on making online shopping much more convenient and easier for the customers. And also invest in improving over all e-commerce section of business

### 3. Insight:

Brazilian customers tend to buy more during afternoon and night times i.e., the peak hours in Brazilian online shopping

## Recommendations:

Since there is high tendency of buying at these times of the day, we can provide offers on multiple products based on their sales performance and can also give combo offers. This will help us to improve sales as well.

### 4. Insight:

States SP, RJ, MG accounts for almost 65% of the customer base And SP alone accounts for almost 42% of the customer base

## Recommendation:

We should focus on improving delivery networks in these states and if possible setup multiple warehouses in these state at different cities, again depending on the cities orders and customer base. This will drastically reduce the average delivery timings for these states which is currently 10.3 days.

### 5. Insight:

There is a huge increase of 137% in cost of orders from 2017 to 2018(Jan to Aug) , which clearly indicates that e-commerce section of target is growing rapidly and people of Brazil adapting faster to online shopping and even the overall purchasing power has increased.

## Recommendations:

We can spend more money on digital marketing which in turn will boost the sales, considering the fact that the sales are any ways growing but this small improvement can result in much more better results

### 6. Insight:

The average time for delivering and average difference in estimated delivery time and actual delivery time is approx. 12 days and 11 days respectively.

## Recommendations:

There is certainly scope for improvements in delivery timings which can be tackled again by improving delivery networks and setting up warehouses .

There is a huge difference between estimated delivery time and actual delivery timing. We need to improve the algorithm for estimated delivery date to be much more precise and make sure that we can try to bring the difference to 5 days at least.

## 7. Insight:

credit card payment method accounts for more than 70% of order payments

### Recommendation:

We can make changes in the UI and the shopping methodology which will smoothen the cart exit for the credit card users much easier.

We can partner up with credit card providers and provide offers for customers who are using that credit card provided by the partnered firm.