1.1) Data type of columns in a table

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

Query results

JOB INFORMATION | RESULTS | JSON | EXECUTION DET.

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

```
SELECT column_name,data_type
FROM   `Target.INFORMATION_SCHEMA.COLUMNS`
WHERE   table_name = 'geolocation';
```

Query results

JOB INFORMATION | RESULTS | JSON | EXECUTION DET.

Row	column_name	data_type
1	geolocation_zip_code_prefix	INT64
2	geolocation_lat	FLOAT64
3	geolocation_lng	FLOAT64
4	geolocation_city	STRING
5	geolocation_state	STRING

```
SELECT column_name,data_type
FROM   `Target.INFORMATION_SCHEMA.COLUMNS`
WHERE   table_name = 'order_items';
```

Query results

JOB INFORMATION | RESULTS | JSON | EXECUTION DET.

Row	column_name	data_type
1	order_id	STRING
2	order_item_id	INT64
3	product_id	STRING
4	seller_id	STRING
5	shipping_limit_date	TIMESTAMP
6	price	FLOAT64
7	freight_value	FLOAT64

```
SELECT column_name,data_type
FROM   `Target.INFORMATION_SCHEMA.COLUMNS`
WHERE   table_name = 'order_reviews';
```

Query results

JOB INFORMATION | RESULTS | JSON | EXECUTION DET.

Row	column_name	data_type
1	review_id	STRING
2	order_id	STRING
3	review_score	INT64
4	review_comment_title	STRING
5	review_creation_date	TIMESTAMP
6	review_answer_timestamp	TIMESTAMP

```
SELECT column_name,data_type
FROM   `Target.INFORMATION_SCHEMA.COLUMNS`
WHERE   table_name = 'orders';
```

Query results

JOB INFORMATION | RESULTS | JSON | EXECUTION DET.

Row	column_name	data_type
1	order_id	STRING
2	customer_id	STRING
3	order_status	STRING
4	order_purchase_timestamp	TIMESTAMP
5	order_approved_at	TIMESTAMP
6	order_delivered_carrier_date	TIMESTAMP
7	order_delivered_customer_date	TIMESTAMP
8	order_estimated_delivery_date	TIMESTAMP

```
SELECT column_name,data_type
FROM   `Target.INFORMATION_SCHEMA.COLUMNS`
WHERE   table_name = 'payments';
```

Query results

JOB INFORMATION | RESULTS | JSON | EXECUTION DET

Row	column_name	data_type
1	order_id	STRING
2	payment_sequential	INT64
3	payment_type	STRING
4	payment_installments	INT64
5	payment_value	FLOAT64

```
SELECT column_name,data_type
FROM   `Target.INFORMATION_SCHEMA.COLUMNS`
WHERE   table_name = 'products';
```

Query results

JOB INFORMATION | RESULTS | JSON | EXECUTION DET.

Row	column_name	data_type
1	product_id	STRING
2	product_category	STRING
3	product_name_length	INT64
4	product_description_length	INT64
5	product_photos_qty	INT64
6	product_weight_g	INT64
7	product_length_cm	INT64
8	product_height_cm	INT64
9	product_width_cm	INT64

```
SELECT column_name,data_type
FROM   `Target.INFORMATION_SCHEMA.COLUMNS`
WHERE   table_name = 'sellers';
```

Query results

JOB INFORMATION | RESULTS | JSON | EXECUTION DET.

Row	column_name	data_type
1	seller_id	STRING
2	seller_zip_code_prefix	INT64
3	seller_city	STRING
4	seller_state	STRING

1.2) Time period for which the data is given

```
SELECT
  MIN(order_purchase_timestamp) AS `First Order Purchase Date`,
  MAX(order_purchase_timestamp) AS `Last Order Purchase Date`
FROM `Target.orders`;
```

Query results

JOB INFORMATION | RESULTS | JSON | EXECUTION DET

Row	First Order Purchase Date	Last Order Purchase Date
1	2016-09-04 21:15:19 UTC	2018-10-17 17:30:18 UTC

1.3) Cities and States of customers ordered during the given period

```
SELECT customer_city, customer_state
FROM `Target.customers`;
```

Query results

JOB INFORMATION | RESULTS | JSON | EXECUTION DET.

Row	customer_city	customer_state
1	acu	RN
2	acu	RN
3	acu	RN
4	ico	CE
5	ico	CE
6	ico	CE
7	ico	CE
8	ico	CE
9	ico	CE
10	ico	CE

```
SELECT
  count(distinct customer_city) AS `Total cities`,
  count(distinct customer_state) AS `Total states`
FROM `Target.customers`;
```

Query results

JOB INFORMATION | RESULTS

Row	Total cities	Total states
1	4119	27

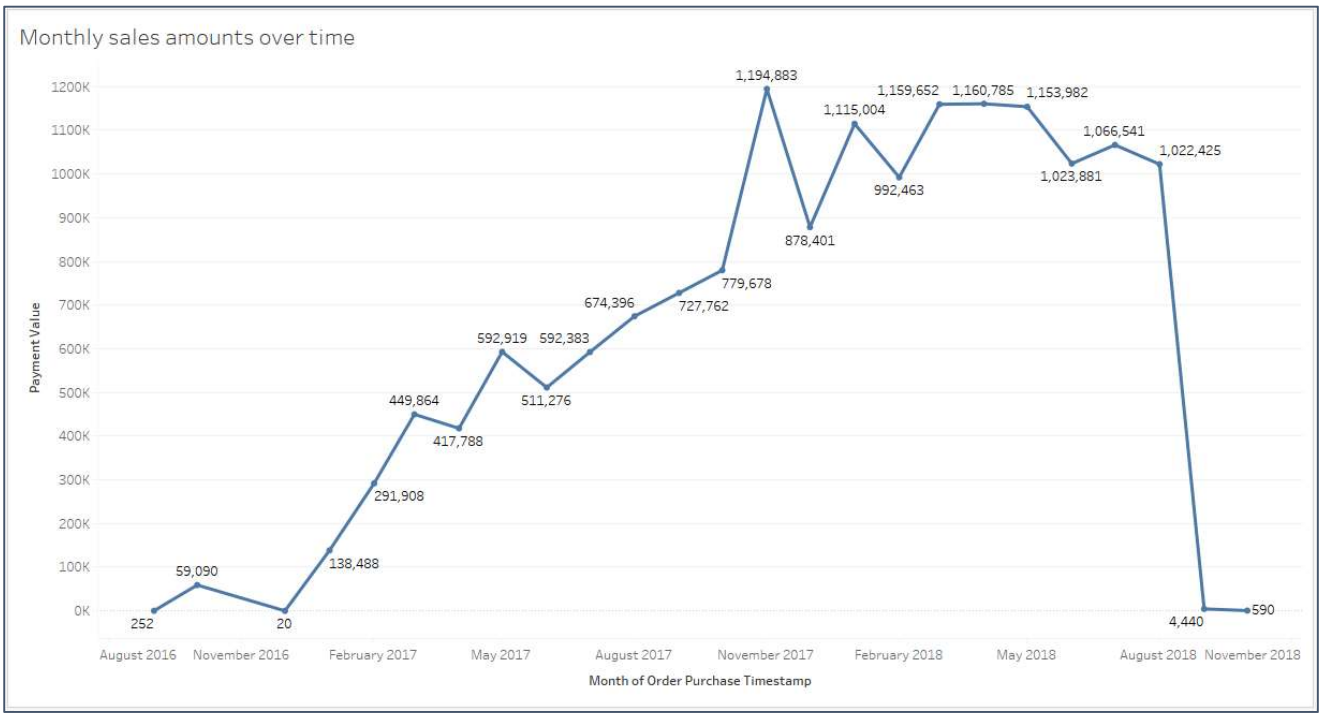
2.1) Is there a growing trend on e-commerce in Brazil? How can we describe a complete scenario? Can we see some seasonality with peaks at specific months?

```
SELECT
  EXTRACT(YEAR FROM o.order_purchase_timestamp) AS order_year,
  EXTRACT(MONTH FROM o.order_purchase_timestamp) AS order_month,
  ROUND(SUM(p.payment_value), 4) AS total_amount,
  COUNT(o.order_id) AS order_count
FROM `Target.orders` o
  INNER JOIN `Target.payments` p
  ON o.order_id = p.order_id
GROUP BY order_year, order_month
ORDER BY order_year, order_month;
```

Query results

JOB INFORMATION | RESULTS | JSON | EXECUTION DET

Row	order_year	order_month	total_amount	order_count
1	2016	9	252.24	3
2	2016	10	59090.48	342
3	2016	12	19.62	1
4	2017	1	138488.04	850
5	2017	2	291908.01	1886
6	2017	3	449863.6	2837
7	2017	4	417788.03	2571
8	2017	5	592918.82	3944
9	2017	6	511276.38	3436
10	2017	7	592382.92	4317



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

```
SELECT
  CASE
    WHEN EXTRACT(HOUR FROM order_purchase_timestamp) BETWEEN 0 AND 5 THEN 'Dawn'
    WHEN EXTRACT(HOUR FROM order_purchase_timestamp) BETWEEN 6 AND 11 THEN 'Morning'
    WHEN EXTRACT(HOUR FROM order_purchase_timestamp) BETWEEN 12 AND 17 THEN 'Afternoon'
    ELSE 'Night'
  END AS order_purchase_time,
  COUNT(*) AS order_purchase_count
FROM `Target.orders`
GROUP BY order_purchase_time
ORDER BY order_purchase_count DESC;
```

Query results

JOB INFORMATION | RESULTS | JSON

Row	order_purchase_time	order_purchase_count
1	Afternoon	38361
2	Night	34100
3	Morning	22240
4	Dawn	4740

3.1) Get month on month orders by states

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

Query results

JOB INFORMATION | RESULTS | JSON | EXECUTION DETAILS | EXECU

Row	Month	Year	customer_state	Orders
1	9	2016	RR	1
2	9	2016	RS	1
3	9	2016	SP	2
4	10	2016	SP	113
5	10	2016	RS	24
6	10	2016	RJ	56
7	10	2016	MT	3
8	10	2016	GO	9
9	10	2016	MG	40
10	10	2016	CE	8

3.2) Distribution of customers across the states in Brazil

```
SELECT
  c.customer_state,
  COUNT(*) AS customer_count
FROM `Target.customers` c
GROUP BY c.customer_state
ORDER BY customer_count DESC;
```

Query results

JOB INFORMATION | RESULTS | JSON

Row	customer_state	customer_count
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

4.1) Get % increase in cost of orders from 2017 to 2018 (include months between Jan to Aug only)

```
WITH year_costs AS (
  SELECT
    EXTRACT(YEAR FROM o.order_purchase_timestamp) AS order_year,
    FORMAT_TIMESTAMP('%B', o.order_purchase_timestamp) AS order_month,
    SUM(p.payment_value) AS total_cost
  FROM `Target.orders` o
  INNER JOIN `Target.payments` p
  ON o.order_id = p.order_id
  WHERE EXTRACT(YEAR FROM o.order_purchase_timestamp) IN (2017, 2018)
    AND EXTRACT(MONTH FROM o.order_purchase_timestamp) BETWEEN 1 AND 8
  GROUP BY order_year, order_month
)
SELECT year_costs_2017.order_month,
  ROUND((year_costs_2018.total_cost - year_costs_2017.total_cost) / year_costs_2017.total_cost
* 100, 4) AS cost_increase_percentage
FROM year_costs year_costs_2017
JOIN year_costs year_costs_2018
ON year_costs_2017.order_month = year_costs_2018.order_month
WHERE year_costs_2017.order_year = 2017
  AND year_costs_2018.order_year = 2018;
```

Query results

JOB INFORMATION | RESULTS | JSON | EXECUTIOI

Row	order_month	cost_increase_percentage
1	April	177.8408
2	June	100.2597
3	May	94.6273
4	August	51.606
5	July	80.0425
6	February	239.9918
7	March	157.7786
8	January	705.1267

4.2) Mean & Sum of price and freight value by customer state

```
SELECT
  c.customer_state,
  ROUND(AVG(oi.price), 4) AS mean_price,
  ROUND(SUM(oi.price), 4) AS sum_price,
  ROUND(AVG(oi.freight_value), 4) AS mean_freight_value,
  ROUND(SUM(oi.freight_value), 4) AS sum_freight_value
FROM `Target.order_items` oi
  INNER JOIN `Target.orders` o
  ON o.order_id = oi.order_id
  INNER JOIN `Target.customers` c
  ON o.customer_id = c.customer_id
GROUP BY c.customer_state
ORDER BY c.customer_state;
```

Query results

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

Row	customer_state	mean_price	sum_price	mean_freight_value	sum_freight_value
1	AC	173.7277	15982.95	40.0734	3686.75
2	AL	180.8892	80314.81	35.8437	15914.59
3	AM	135.496	22356.84	33.2054	5478.89
4	AP	164.3207	13474.3	34.0061	2788.5
5	BA	134.6012	511349.99	26.364	100156.68
6	CE	153.7583	227254.71	32.7142	48351.59
7	DF	125.7705	302603.94	21.0414	50625.5
8	ES	121.9137	275037.31	22.0588	49764.6
9	GO	126.2717	294591.95	22.7668	53114.98
10	MA	145.2042	119648.22	38.257	31523.77

5.1) Calculate days between purchasing, delivering and estimated delivery

```
SELECT
  o.order_id,
  TIMESTAMP_DIFF(order_delivered_customer_date, order_purchase_timestamp, DAY)
AS  time_to_delivery,
  TIMESTAMP_DIFF(order_estimated_delivery_date, order_delivered_customer_date, DAY) AS
diff_estimated_delivery
FROM `Target.orders`;
```

Query results

JOB INFORMATION | RESULTS | JSON | EXECUTION DETAILS

Row	order_id	time_to_delivery	diff_estimated_delivery
1	1950d777989f6a877539f5379...	30	-12
2	2c45c33d2f9cb8ff8b1c86cc28...	30	28
3	65d1e226dfaeb8cdc42f66542...	35	16
4	635c894d068ac37e6e03dc54e...	30	1
5	3b97562c3aee8bdedcb5c2e45...	32	0
6	68f47f50f04c4cb6774570cfde...	29	1
7	276e9ec344d3bf029ff83a161c...	43	-4
8	54e1a3c2b97fb0809da548a59...	40	-4
9	fd04fa4105ee8045f6a0139ca5...	37	-1
10	302bb8109d097a9fc6e9cefc5...	33	-5

5.5) Top 5 states with highest/lowest average freight value

```
SELECT
  c.customer_state,
  ROUND(AVG(oi.freight_value), 4) AS average_freight_value,
  AVG(TIMESTAMP_DIFF(order_delivered_customer_date, order_purchase_timestamp, DAY))
AS  time_to_delivery,
  AVG(TIMESTAMP_DIFF(order_estimated_delivery_date, order_delivered_customer_date, DAY)) AS
diff_estimated_delivery
FROM `Target.order_items` oi
  INNER JOIN `Target.orders` o
  ON o.order_id = oi.order_id
  INNER JOIN `Target.customers` c
  ON o.customer_id = c.customer_id
GROUP BY c.customer_state
ORDER BY average_freight_value ASC
LIMIT 5;
```

Query results

JOB INFORMATION | RESULTS | JSON | EXECUTION DETAILS | EXECUT

Row	customer_state	average_freight_value	time_to_delivery	diff_estimated_delivery
1	SP	15.1473	8.25960855241909	10.26559438451439
2	PR	20.5317	11.480793060718735	12.533899805275263
3	MG	20.6302	11.515522180072811	12.397151041263502
4	RJ	20.9609	14.689382157500321	11.14449314293797
5	DF	21.0414	12.501486199575384	11.274734607218704

```
SELECT
  c.customer_state,
  ROUND(AVG(oi.freight_value), 4) AS average_freight_value,
  AVG(TIMESTAMP_DIFF(order_delivered_customer_date, order_purchase_timestamp, DAY))
AS  time_to_delivery,
  AVG(TIMESTAMP_DIFF(order_estimated_delivery_date, order_delivered_customer_date, DAY)) AS
diff_estimated_delivery
FROM `Target.order_items` oi
  INNER JOIN `Target.orders` o
  ON o.order_id = oi.order_id
  INNER JOIN `Target.customers` c
  ON o.customer_id = c.customer_id
GROUP BY c.customer_state
ORDER BY average_freight_value DESC
LIMIT 5;
```

Query results

JOB INFORMATION | RESULTS | JSON | EXECUTION DETAILS | EXECUT

Row	customer_state	average_freight_value	time_to_delivery	diff_estimated_delivery
1	RR	42.9844	27.826086956521738	17.434782608695652
2	PB	42.7238	20.119453924914676	12.15017064846416
3	RO	41.0697	19.282051282051292	19.080586080586084
4	AC	40.0734	20.329670329670336	20.010989010989018
5	PI	39.148	18.931166347992352	10.682600382409184

5.6) Top 5 states with highest/lowest average time to delivery

```
SELECT
  c.customer_state,
  ROUND(AVG(oi.freight_value), 4) AS average_freight_value,
  AVG(TIMESTAMP_DIFF(order_delivered_customer_date, order_purchase_timestamp, DAY))
AS  `avg_time_to_delivery`,
  AVG(TIMESTAMP_DIFF(order_estimated_delivery_date, order_delivered_customer_date, DAY)) AS
`avg_diff_estimated_delivery`
FROM `Target.order_items` oi
  INNER JOIN `Target.orders` o
  ON o.order_id = oi.order_id
  INNER JOIN `Target.customers` c
  ON o.customer_id = c.customer_id
GROUP BY c.customer_state
ORDER BY avg_time_to_delivery ASC
LIMIT 5;
```

Query results

JOB INFORMATION | RESULTS | JSON | EXECUTION DETAILS | EXECUTION

Row	customer_state	average_freight_value	avg_time_to_delivery	avg_diff_estimated_delivery
1	SP	15.1473	8.25960855241909	10.26559438451439
2	PR	20.5317	11.480793060718735	12.533899805275263
3	MG	20.6302	11.515522180072811	12.397151041263502
4	DF	21.0414	12.501486199575384	11.274734607218704
5	SC	21.4704	14.520985846754517	10.6688628599317

```
SELECT
  c.customer_state,
  ROUND(AVG(oi.freight_value), 4) AS average_freight_value,
  AVG(TIMESTAMP_DIFF(order_delivered_customer_date, order_purchase_timestamp, DAY))
AS  `avg_time_to_delivery`,
  AVG(TIMESTAMP_DIFF(order_estimated_delivery_date, order_delivered_customer_date, DAY)) AS
`avg_diff_estimated_delivery`
FROM `Target.order_items` oi
  INNER JOIN `Target.orders` o
  ON o.order_id = oi.order_id
  INNER JOIN `Target.customers` c
  ON o.customer_id = c.customer_id
GROUP BY c.customer_state
ORDER BY avg_time_to_delivery DESC
LIMIT 5;
```

Query results

JOB INFORMATION | RESULTS | JSON | EXECUTION DETAILS | EXECUTION

Row	customer_state	average_freight_value	avg_time_to_delivery	avg_diff_estimated_delivery
1	RR	42.9844	27.826086956521738	17.434782608695652
2	AP	34.0061	27.753086419753075	17.444444444444443
3	AM	33.2054	25.963190184049076	18.975460122699381
4	AL	35.8437	23.992974238875881	7.9765807962529349
5	PA	35.8327	23.301707779886126	13.37476280834913

5.7) Top 5 states where delivery is really fast/ not so fast compared to estimated date

```
SELECT
  c.customer_state,
  ROUND(AVG(oi.freight_value), 4) AS average_freight_value,
  AVG(TIMESTAMP_DIFF(order_delivered_customer_date, order_purchase_timestamp, DAY))
AS  `avg_time_to_delivery`,
  AVG(TIMESTAMP_DIFF(order_estimated_delivery_date, order_delivered_customer_date, DAY)) AS
`avg_diff_estimated_delivery`
FROM `Target.order_items` oi
  INNER JOIN `Target.orders` o
  ON o.order_id = oi.order_id
  INNER JOIN `Target.customers` c
  ON o.customer_id = c.customer_id
GROUP BY c.customer_state
ORDER BY avg_diff_estimated_delivery ASC
LIMIT 5;
```

Query results

JOB INFORMATION | RESULTS | JSON | EXECUTION DETAILS | EXECUTION

Row	customer_state	average_freight_value	avg_time_to_delivery	avg_diff_estimated_delivery
1	AL	35.8437	23.992974238875881	7.976580796252918
2	MA	38.257	21.20375	9.1100000000000119
3	SE	36.6532	20.978666666666683	9.1653333333333329
4	ES	22.0588	15.192808988764023	9.7685393258427116
5	BA	26.364	18.774640238935675	10.119467825142568

```
SELECT
  c.customer_state,
  ROUND(AVG(oi.freight_value), 4) AS average_freight_value,
  AVG(TIMESTAMP_DIFF(order_delivered_customer_date, order_purchase_timestamp, DAY))
AS  `avg_time_to_delivery`,
  AVG(TIMESTAMP_DIFF(order_estimated_delivery_date, order_delivered_customer_date, DAY)) AS
`avg_diff_estimated_delivery`
FROM `Target.order_items` oi
  INNER JOIN `Target.orders` o
  ON o.order_id = oi.order_id
  INNER JOIN `Target.customers` c
  ON o.customer_id = c.customer_id
GROUP BY c.customer_state
ORDER BY avg_diff_estimated_delivery DESC
LIMIT 5;
```

Query results

JOB INFORMATION | RESULTS | JSON | EXECUTION DETAILS | EXECUTION

Row	customer_state	average_freight_value	avg_time_to_delivery	avg_diff_estimated_delivery
1	AC	40.0734	20.329670329670336	20.010989010989018
2	RO	41.0697	19.282051282051292	19.080586080586084
3	AM	33.2054	25.963190184049076	18.975460122699381
4	AP	34.0061	27.753086419753075	17.444444444444443
5	RR	42.9844	27.826086956521738	17.434782608695652

6.1) Month over Month count of orders for different payment types

```
SELECT
  FORMAT_TIMESTAMP('%Y-%m', order_purchase_timestamp) AS month,
  payment_type,
  COUNT(o.order_id) AS order_count
FROM `Target.orders` o
  INNER JOIN `Target.payments` p
  ON o.order_id = p.order_id
GROUP BY month, payment_type
ORDER BY month ASC;
```

Query results

JOB INFORMATION | RESULTS | JSON | EXECUTION DETAILS | EXEC

Row	month	payment_type	order_count
1	2016-09	credit_card	3
2	2016-10	credit_card	254
3	2016-10	UPI	63
4	2016-10	voucher	23
5	2016-10	debit_card	2
6	2016-12	credit_card	1
7	2017-01	credit_card	583
8	2017-01	UPI	197
9	2017-01	voucher	61
10	2017-01	debit_card	9

6.2) Count of orders based on the no. of payment installments

```
SELECT
  payment_installments,
  COUNT(*) AS order_count
FROM `Target.payments`
GROUP BY payment_installments;
```

Query results

JOB INFORMATION | RESULTS

Row	payment_installments	order_count
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

7) Actionable Insights

i. Time period for the given dataset:

   First Order Purchase date -> 2016-09-04 21:15:19 UTC
   Last Order Purchase date -> 2018-10-17 17:30:18 UTC

ii. There are a total of 27 different states and 4119 different cities given in the dataset.

iii. There is an uprising trend on e-commerce starting from January 2016, and declining trend towards the end starting from September, 2018.

iv. The highest order count (7863) with highest sales (1,194,883) was in November 2017.

v. Brazilian customers tend to buy most orders in the afternoon.

vi. The most customers out of Brazilian states belong to state ‘SP’.

vii. Between January and August from 2017 to 2018, April has the highest % increase in cost of orders.

viii. State ‘SP’ has the lowest average freight value while state ‘RR’ has the highest average freight value.

ix. State ‘SP’ takes the minimum time for delivery while state ‘RR’ takes the maximum time for delivery.

x. State ‘AL’ takes the minimum delivery delay while state ‘AC’ takes the maximum delivery delay.

xi. Customers buy most products through ‘credit_card’ payment_type.

xii. Customers mostly purchase products with only 1 payment_installments.

8) Recommendations

i. Optimize shipping and delivery processes by partnering with reliable logistics providers to ensure timely and cost-effective deliveries.

ii. Communicate delivery updates and tracking information to customers to enhance transparency and manage their expectations.

iii. Conduct market research to identify popular products, localize product features, packaging, and pricing to align with local market demands.

iv. Engage in sustainability efforts, support local communities, or contribute to social causes that are important to Brazilian consumers. Communicate your brand's commitment to social responsibility to build trust and loyalty.

v. Conduct regular competitive analysis to stay updated on market trends, pricing strategies, and competitors' offerings.