Retail– Case Study

Problem Statement:

Assuming you are a data analyst/ scientist at Big Retail Firm, you have been assigned the task of analyzing the given dataset to extract valuable insights and provide actionable recommendations.

The data is available in 8 different csv files:

1. customers.csv
2. geolocation.csv
3. order_items.csv
4. payments.csv
5. reviews.csv
6. orders.csv
7. products.csv
8. sellers.csv

The column description for these csv files is given below.

The customers.csv contain following features:

Features	**Description**
customer_id	ID of the consumer who made the purchase
customer_unique_id	Unique ID of the consumer
customer_zip_code_prefix	Zip Code of consumer’s location
customer_city	Name of the City from where order is made
customer_state	State Code from where order is made (Eg. são paulo - SP)

The orders.csv contain following features:

Features	**Description**
order_id	A Unique ID of order made by the consumers
customer_id	ID of the consumer who made the purchase

order_status	Status of the order made i.e. delivered, shipped, etc.
order_purchase_timestamp	Timestamp of the purchase
order_delivered_carrier_date	Delivery date at which carrier made the delivery
order_delivered_customer_date	Date at which customer got the product
order_estimated_delivery_date	Estimated delivery date of the products

The order_items.csv contain following features:

Features	**Description**
order_id	A Unique ID of order made by the consumers
order_item_id	A Unique ID given to each item ordered in the order
product_id	A Unique ID given to each product available on the site
seller_id	Unique ID of the seller registered in Target
shipping_limit_date	The date before which the ordered product must be shipped
price	Actual price of the products ordered
freight_value	Price rate at which a product is delivered from one point to another

The payments.csv contain following features:

Features	**Description**
order_id	A Unique ID of order made by the consumers
payment_sequential	Sequences of the payments made in case of EMI
payment_type	Mode of payment used (Eg. Credit Card)

payment_installments	Number of installments in case of EMI purchase
payment_value	Total amount paid for the purchase order

The geolocations.csv contain following features:

Features	**Description**
geolocation_zip_code_prefix	First 5 digits of Zip Code
geolocation_lat	Latitude
geolocation_lng	Longitude
geolocation_city	City
geolocation_state	State

The sellers.csv contains following features:

Features	**Description**
seller_id	Unique ID of the seller registered
seller_zip_code_prefix	Zip Code of the seller’s location
seller_city	Name of the City of the seller
seller_state	State Code (Eg. são paulo - SP)

The reviews.csv contain following features:

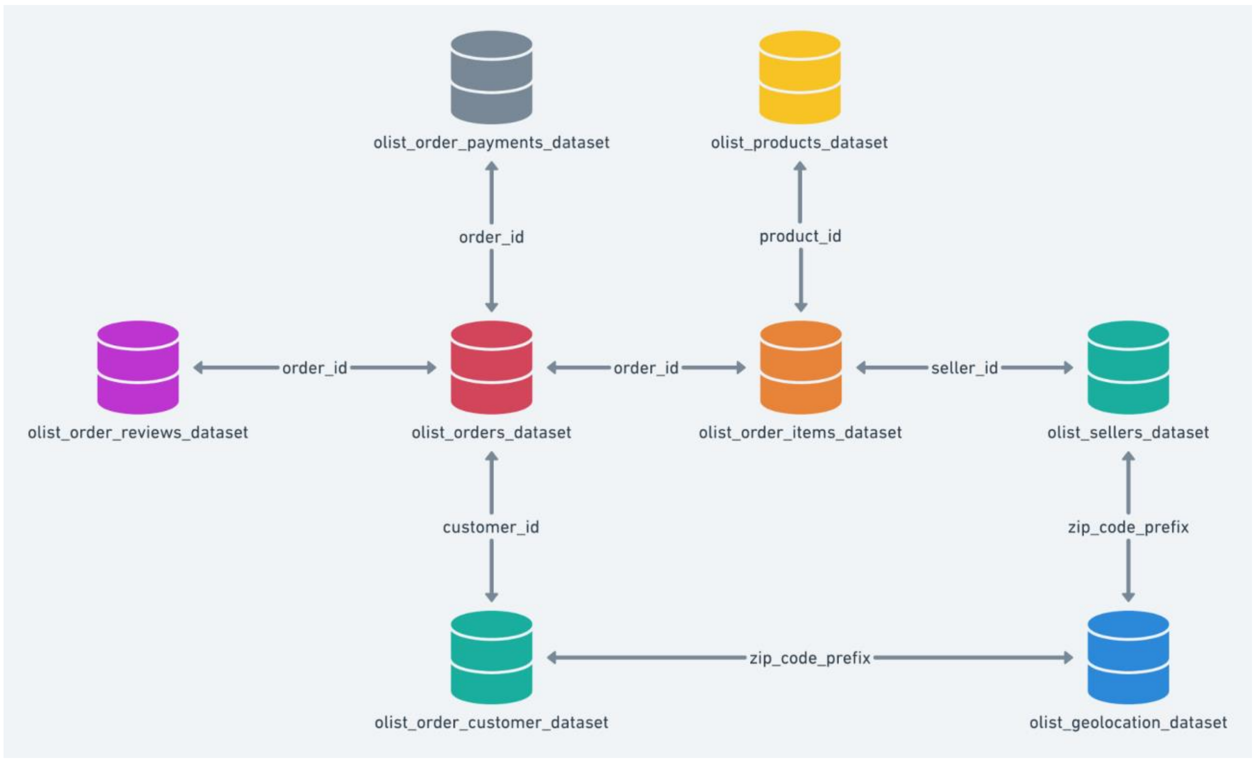
Features	**Description**
review_id	ID of the review given on the product ordered by the order id
order_id	A Unique ID of order made by the consumers
review_score	Review score given by the customer for each order on a scale of 1-5

review_comment_title	Title of the review
review_comment_message	Review comments posted by the consumer for each order
review_creation_date	Timestamp of the review when it is created
review_answer_timestamp	Timestamp of the review answered

The products.csv contain following features:

Features	**Description**
product_id	A Unique identifier for the proposed project.
product_category_name	Name of the product category
product_name_lenght	Length of the string which specifies the name given to the products ordered
product_description_lenght	Length of the description written for each product ordered on the site
product_photos_qty	Number of photos of each product ordered available on the shopping portal
product_weight_g	Weight of the products ordered in grams
product_length_cm	Length of the products ordered in centimeters
product_height_cm	Height of the products ordered in centimeters
product_width_cm	Width of the product ordered in centimeters

Dataset schema:



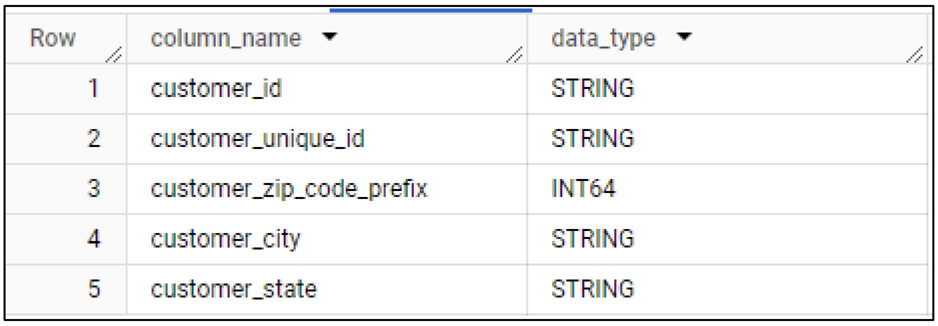
What does 'good' look like?

I. **Import the dataset and do usual exploratory analysis steps like checking the structure & characteristics of the dataset:**

   A. **Data type of all columns in the "customers" table.**

Ans.

```
select column_name, data_type
from scalar-bignner-batch.Target.INFORMATION_SCHEMA.COLUMNS
where table_name = 'customers';
```



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

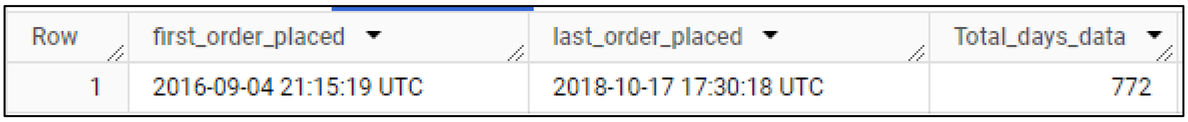
Insights:

Customers table mostly contains datatype as ‘STRING’ and only zip-code prefix is of type integer.

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

Ans.

```
select min(order_purchase_timestamp) as first_order_placed,
  max(order_purchase_timestamp) as last_order_placed,
  date_diff(max(order_purchase_timestamp), min(order_purchase_timestamp), day) as To-
tal_days_data
from scalar-bignner-batch.Target.orders
```



Row	first_order_placed	last_order_placed	Total_days_data
1	2016-09-04 21:15:19 UTC	2018-10-17 17:30:18 UTC	772

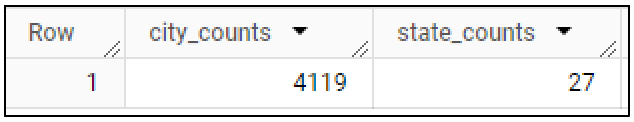
Insights:

The First order in the given dataset the placed on 04th September, 2016 in evening at 21:15:19 UTC and the last order got placed on 17th October, 2018 in afternoon at 17:30:18 UTC, that’s **range of 772 days**.

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

Ans.

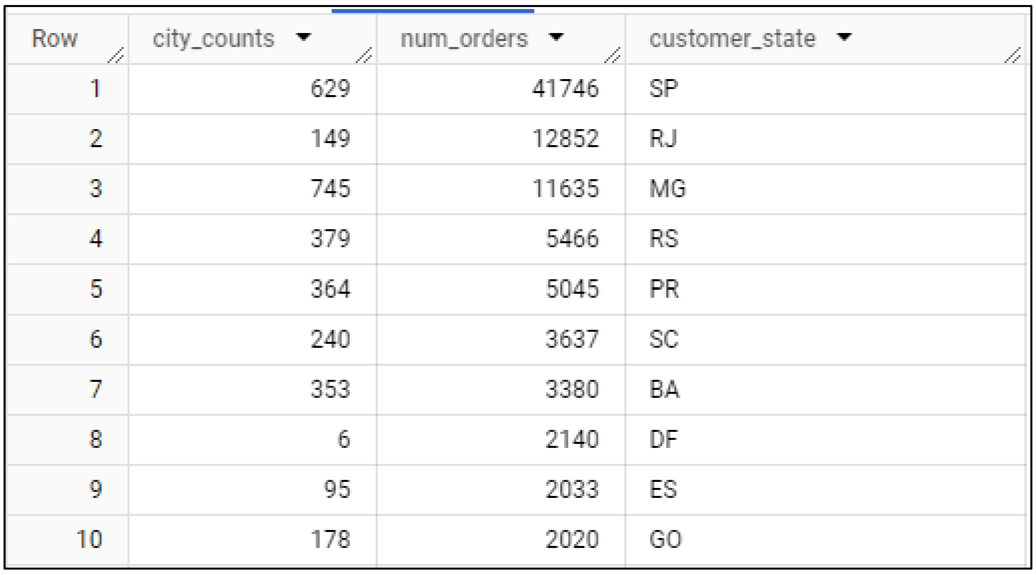
```
select count(distinct customer_city) as city_counts,
  count(distinct customer_state) as state_counts
from scalar-bignner-batch.Target.customers c
join scalar-bignner-batch.Target.orders o using(customer_id);
```



Row	city_counts	state_counts
1	4119	27

For better insights:

```
select count(distinct customer_city) as city_counts,
  count(*) as num_orders,
  customer_state
from scalar-bignner-batch.Target.customers c
join scalar-bignner-batch.Target.orders o using(customer_id)
group by customer_state
order by num_orders desc;
```



Row	city_counts	num_orders	customer_state
1	629	41746	SP
2	149	12852	RJ
3	745	11635	MG
4	379	5466	RS
5	364	5045	PR
6	240	3637	SC
7	353	3380	BA
8	6	2140	DF
9	95	2033	ES
10	178	2020	GO

Insights:

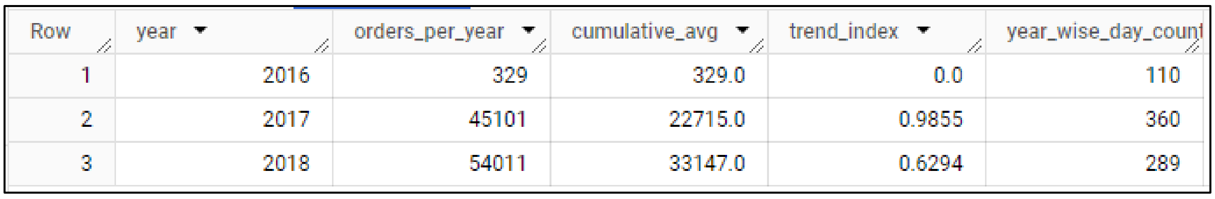
Total number of cities from which orders got placed counts for 4119 spread-across 27 states in Brazil. Among them, highest number of orders are made from SP state (with 41746 orders in 629 different cities) followed by RJ and MG states.

II. **In-depth Exploration:**

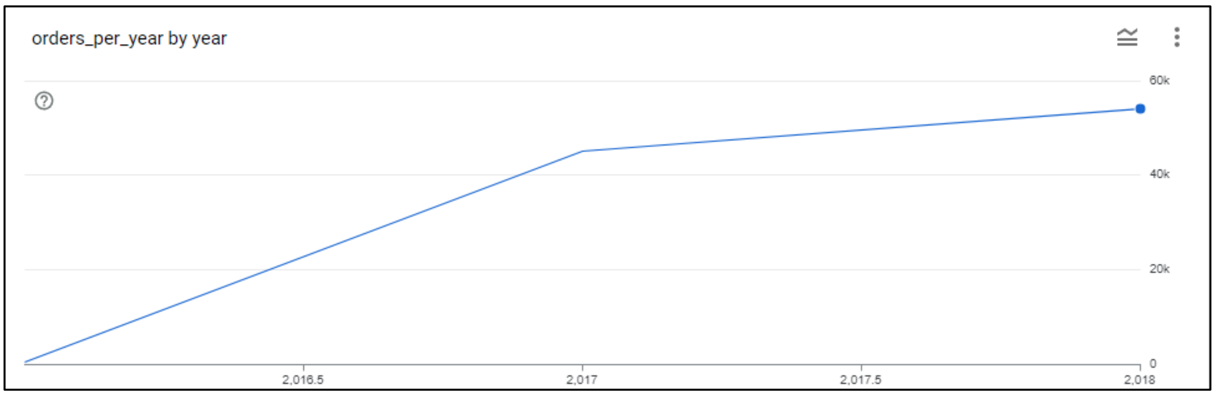
A. **Is there a growing trend in the no. of orders placed over the past years?**

Ans.

```
with cte as (
  select
    extract(YEAR from order_purchase_timestamp) as year,
    count(*) as orders_per_year,
    min(order_purchase_timestamp) as first_order_of_year,
    max(order_purchase_timestamp) as last_order_of_year,
    date_diff(max(order_purchase_timestamp), min(order_purchase_timestamp), day)
year_wise_day_count
  from scalar-bignner-batch.Target.orders
  group by year
),

yearly_avg as (select *,
  avg(orders_per_year) over(order by year) as cumulative_avg
from cte)

select
  year, orders_per_year, cumulative_avg,
  round((yearly_avg.orders_per_year - cumulative_avg) / cumulative_avg, 4) as trend_index,
  year_wise_day_count
from yearly_avg;
```



Row	year	orders_per_year	cumulative_avg	trend_index	year_wise_day_count
1	2016	329	329.0	0.0	110
2	2017	45101	22715.0	0.9855	360
3	2018	54011	33147.0	0.6294	289



Insights:

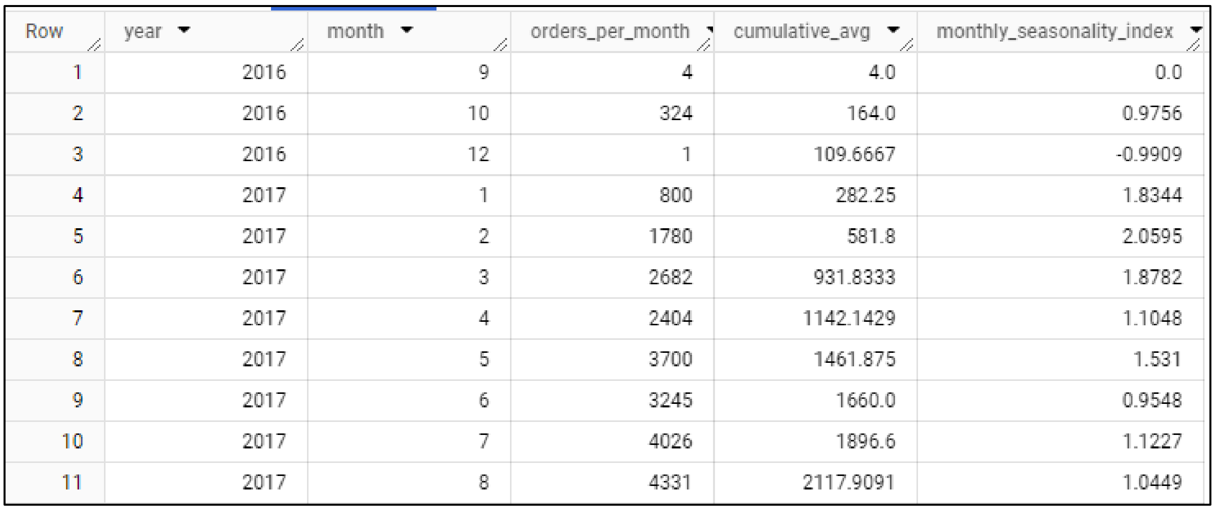
Yes, we can see significate increase in order trends from year 2016 to 2018 in Brazil market, we can see in 2016 over a period of 110 days total orders placed were only 329 (maybe Target started its business in Brazil around this time! And was new to Brazilian market).

We can see huge jump in orders in positive direction from year 2017 to 2018, where order trend index is giving positive values.

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

Ans.

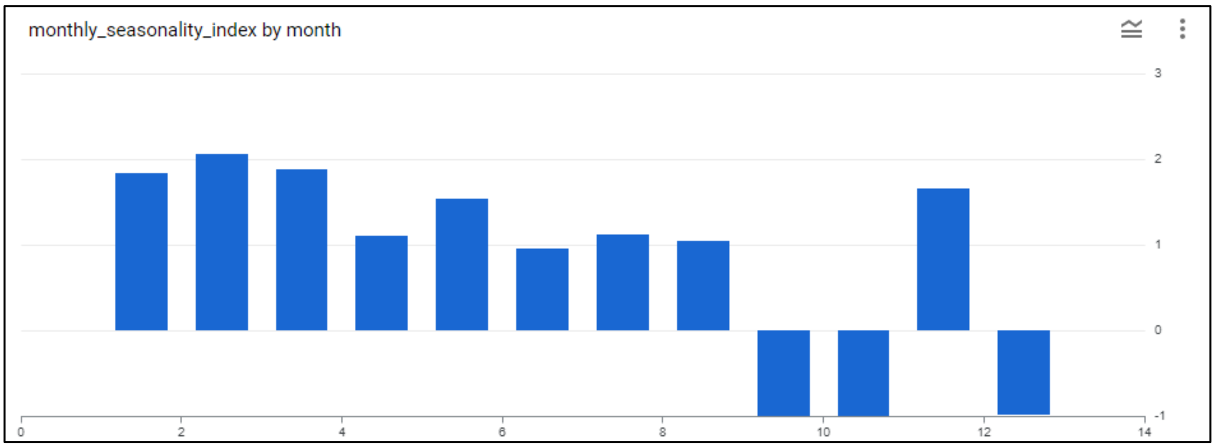
```
with cte as (
  select
    extract(MONTH from order_purchase_timestamp) as month,
    Extract(YEAR from order_purchase_timestamp) as year,
    count(*) as orders_per_month,
  from scalar-bignner-batch.Target.orders
  group by month, year
),

monthly_avg as (select *,
  round(avg(orders_per_month) over(order by year, month), 4) as cumulative_avg
from cte)

select
  year, month, orders_per_month,cumulative_avg,
  round((monthly_avg.orders_per_month - cumulative_avg) / cumulative_avg, 4) as
monthly_seasonality_index
from monthly_avg;
```



Row	year	month	orders_per_month	cumulative_avg	monthly_seasonality_index
1	2016	9	4	4.0	0.0
2	2016	10	324	164.0	0.9756
3	2016	12	1	109.6667	-0.9909
4	2017	1	800	282.25	1.8344
5	2017	2	1780	581.8	2.0595
6	2017	3	2682	931.8333	1.8782
7	2017	4	2404	1142.1429	1.1048
8	2017	5	3700	1461.875	1.531
9	2017	6	3245	1660.0	0.9548
10	2017	7	4026	1896.6	1.1227
11	2017	8	4331	2117.9091	1.0449

Insights:

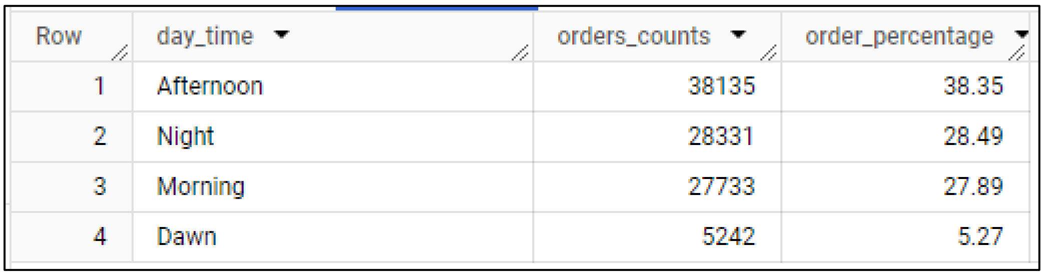
Yes, we are able to see monthly seasonality in number of orders while looking at index. Basically, index value greater than 1 means seasonality was above cumulative average which can be seen for **month Jan, Mar, Feb of year 2017 and 2018** which seems to have peaks. These peaks can also be visible in below given chart.



C. **During what time of the day, do the Brazilian customers mostly place their orders? (Dawn, Morning, Afternoon or Night)**
   - **0-6 hrs: Dawn**
   - **7-12 hrs: Mornings**
   - **13-18 hrs: Afternoon**
   - **19-23 hrs: Night**

Ans.

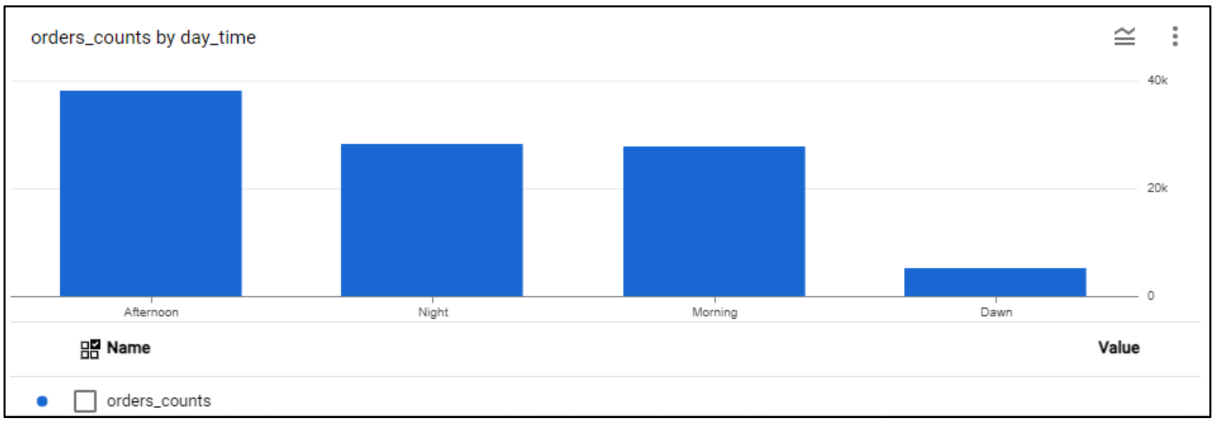
```
select
  case
    when extract(hour from order_purchase_timestamp) between 0 and 6 then 'Dawn'
    when extract(hour from order_purchase_timestamp) between 7 and 12 then 'Morning'
    when extract(hour from order_purchase_timestamp) between 13 and 18 then 'Afternoon'
    else 'Night'
  end as day_time,
  count(*) as orders_counts,
  round(count(*) * 100/ (select count(*) from scalar-bignner-batch.Target.orders), 2) as
order_percentage
from scalar-bignner-batch.Target.orders
group by day_time
order by orders_counts desc;
```



Row	day_time	orders_counts	order_percentage
1	Afternoon	38135	38.35
2	Night	28331	28.49
3	Morning	27733	27.89
4	Dawn	5242	5.27

Insights:

During Afternoon Brazilian customers tends to place orders the most, followed by Night and then Morning. Very few customers place order during Dawn as its sleeping time.

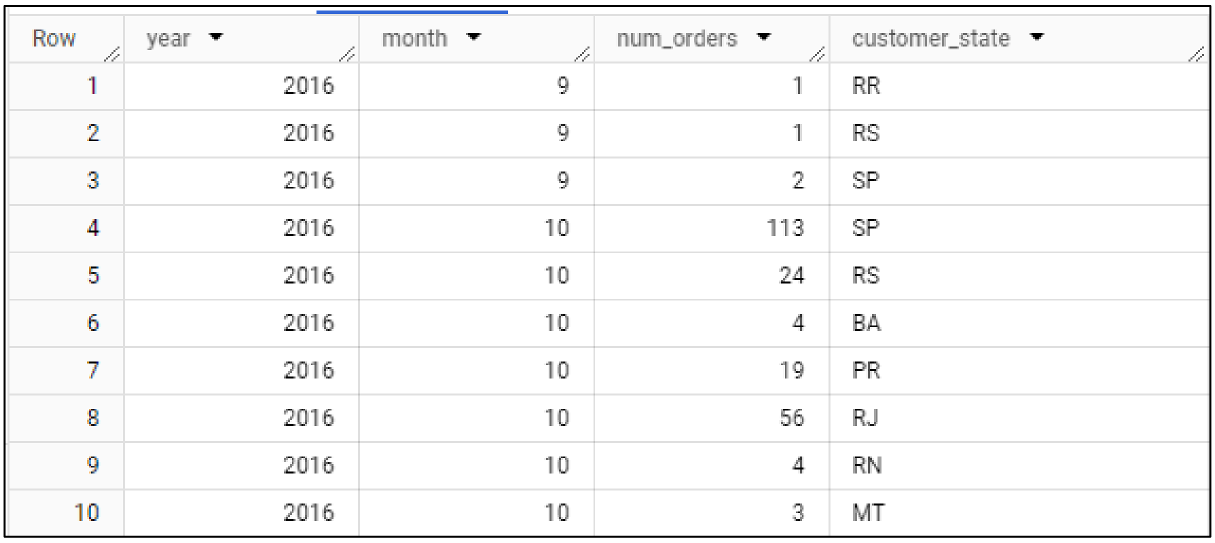


III. **Evolution of E-commerce orders in the Brazil region:**

A. **Get the month-on-month no. of orders placed in each state.**

Ans.

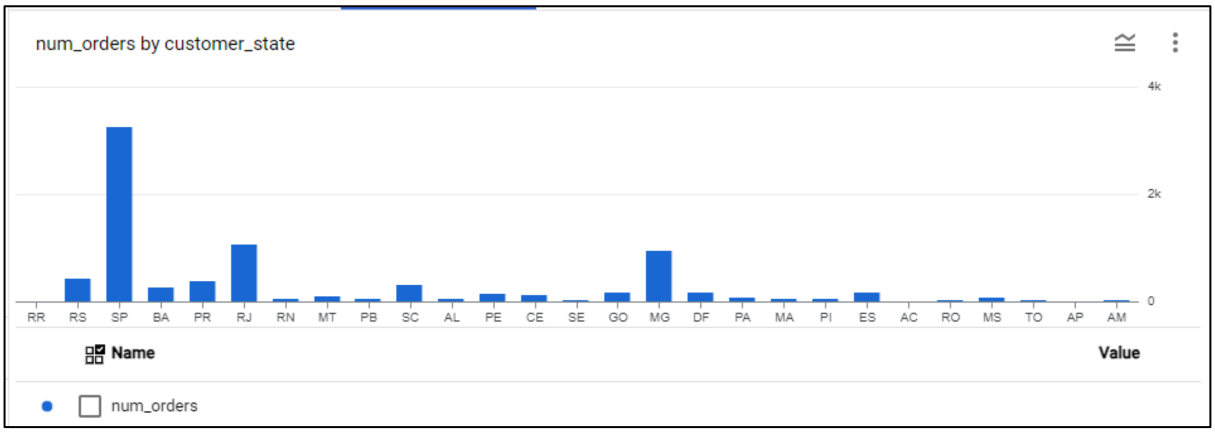
```
select extract(YEAR from o.order_purchase_timestamp) year,
  extract(MONTH from o.order_purchase_timestamp) month,
  count(*) as num_orders,
  customer_state
from scalar-bignner-batch.Target.customers c
join scalar-bignner-batch.Target.orders o using(customer_id)
group by customer_state, month, year
order by year, month;
```



Row	year	month	num_orders	customer_state
1	2016	9	1	RR
2	2016	9	1	RS
3	2016	9	2	SP
4	2016	10	113	SP
5	2016	10	24	RS
6	2016	10	4	BA
7	2016	10	19	PR
8	2016	10	56	RJ
9	2016	10	4	RN
10	2016	10	3	MT

Insights:

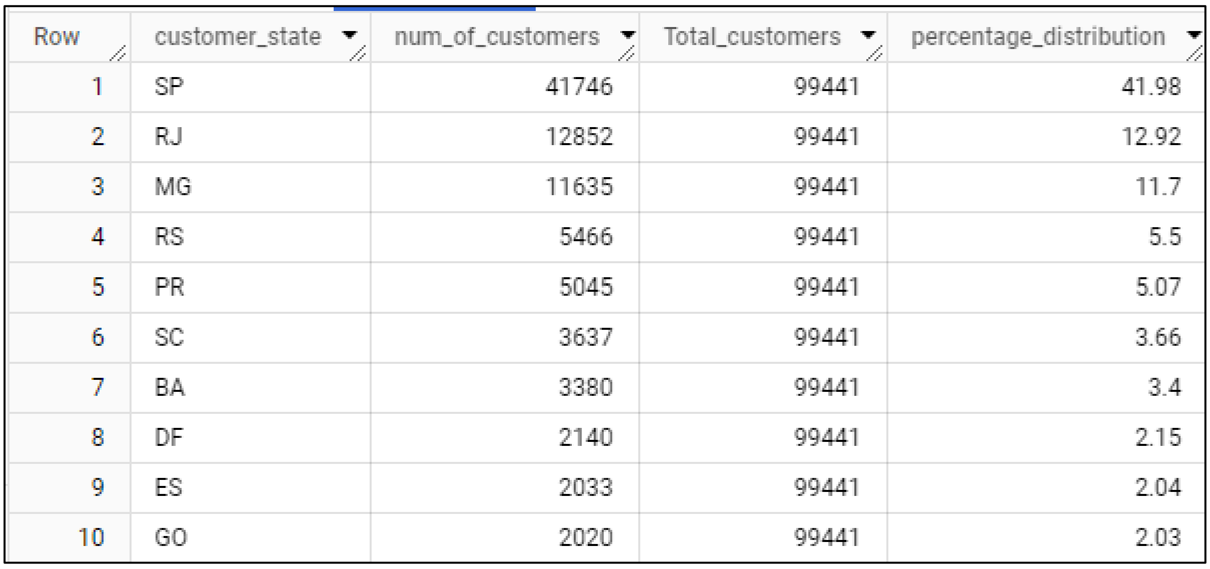
State-wise MoM shows that highest number of orders are being place in SP states as shown in below graphs - SP followed by RJ and MG and from below graph we can observe there is huge difference in number of orders placed in SP then any order state.



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

Ans.

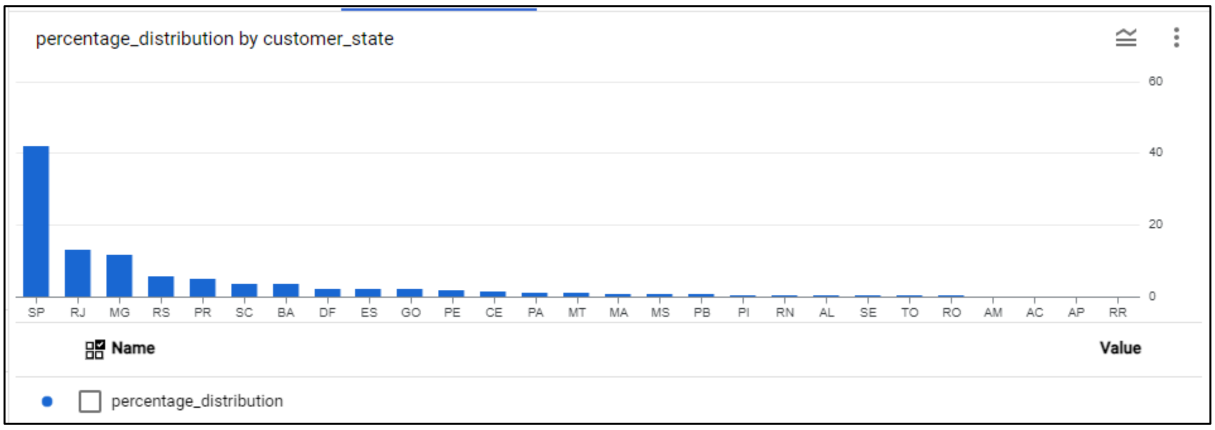
```
select *,
  sum(state_wise_cust.num_of_customers) over() as Total_customers,
  round(state_wise_cust.num_of_customers * 100 / sum(state_wise_cust.num_of_customers)
over(), 2) as percentage_distribution
from (
  select customer_state,
    count(customer_id) as num_of_customers
  from scalar-bignner-batch.Target.customers
  group by customer_state
) state_wise_cust
order by state_wise_cust.num_of_customers desc
```



Row	customer_state	num_of_customers	Total_customers	percentage_distribution
1	SP	41746	99441	41.98
2	RJ	12852	99441	12.92
3	MG	11635	99441	11.7
4	RS	5466	99441	5.5
5	PR	5045	99441	5.07
6	SC	3637	99441	3.66
7	BA	3380	99441	3.4
8	DF	2140	99441	2.15
9	ES	2033	99441	2.04
10	GO	2020	99441	2.03

Insights:

Customer distribution is biased towards SP state which is having almost 42% of the total customers present in the dataset whereas states like RR, AP, AC, AM and RO (states with bottom 5 number of customers) combined doesn't even constitute even 1% of total customers.

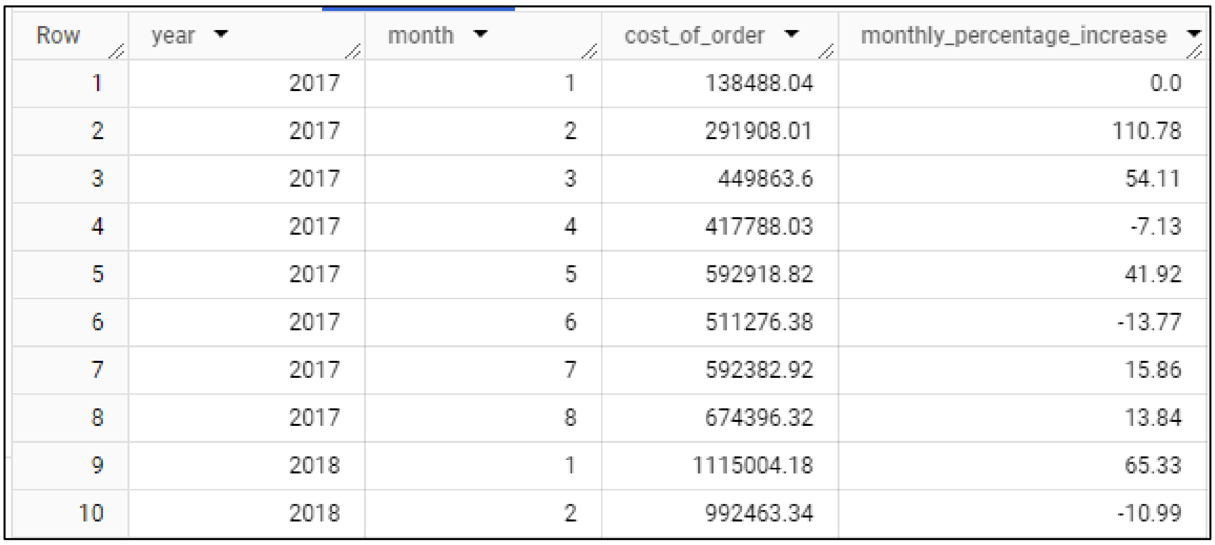


IV. **Impact on Economy: Analyze the money movement by e-commerce by looking at order prices, freight and others.**

A. **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.

Ans.

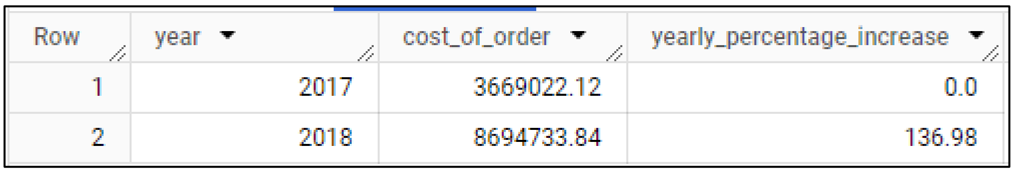
```
with cte as (
  select
    extract(YEAR from o.order_purchase_timestamp) as year,
    extract(MONTH from o.order_purchase_timestamp) as month,
    round(sum(payment_value), 2) cost_of_order
  from scalar-bignner-batch.Target.payments p
  join scalar-bignner-batch.Target.orders o using(order_id)
  where extract(YEAR from o.order_purchase_timestamp) between 2017 and 2018
  and extract(MONTH from o.order_purchase_timestamp) between 01 and 08
  group by year, month
),

monthly_costs as (select *,
  round(lag(cost_of_order) over(order by year, month), 2) prev_month_cost
from cte
order by year, month)

select year, month, monthly_costs.cost_of_order,
  ifnull(round((monthly_costs.cost_of_order - prev_month_cost) * 100 / prev_month_cost, 2),
0) as monthly_percentage_increase
from monthly_costs
```



Row	year	month	cost_of_order	monthly_percentage_increase
1	2017	1	138488.04	0.0
2	2017	2	291908.01	110.78
3	2017	3	449863.6	54.11
4	2017	4	417788.03	-7.13
5	2017	5	592918.82	41.92
6	2017	6	511276.38	-13.77
7	2017	7	592382.92	15.86
8	2017	8	674396.32	13.84
9	2018	1	1115004.18	65.33
10	2018	2	992463.34	-10.99

In case, we are only looking for yearly data only ignoring the monthly changes, we can slightly change the above query by removing the month from grouping and by changing few names of variables for better readability and understandability. Below is yearly percent change using only Jan to Aug data:

```
with cte2 as (
  select
    extract(YEAR from o.order_purchase_timestamp) as year,
    round(sum(payment_value), 2) cost_of_order
  from scalar-bignner-batch.Target.payments p
  join scalar-bignner-batch.Target.orders o using(order_id)
  where extract(YEAR from o.order_purchase_timestamp) between 2017 and 2018
  and extract(MONTH from o.order_purchase_timestamp) between 01 and 08
  group by year
),

yearly_costs as (select *,
  round(lag(cost_of_order) over(order by year), 2) prev_year_cost
from cte2
order by year)

select year, yearly_costs.cost_of_order,
  ifnull(round((yearly_costs.cost_of_order - prev_year_cost) * 100 / prev_year_cost, 2), 0)
as yearly_percentage_increase
from yearly_costs;
```



Row	year	cost_of_order	yearly_percentage_increase
1	2017	3669022.12	0.0
2	2018	8694733.84	136.98

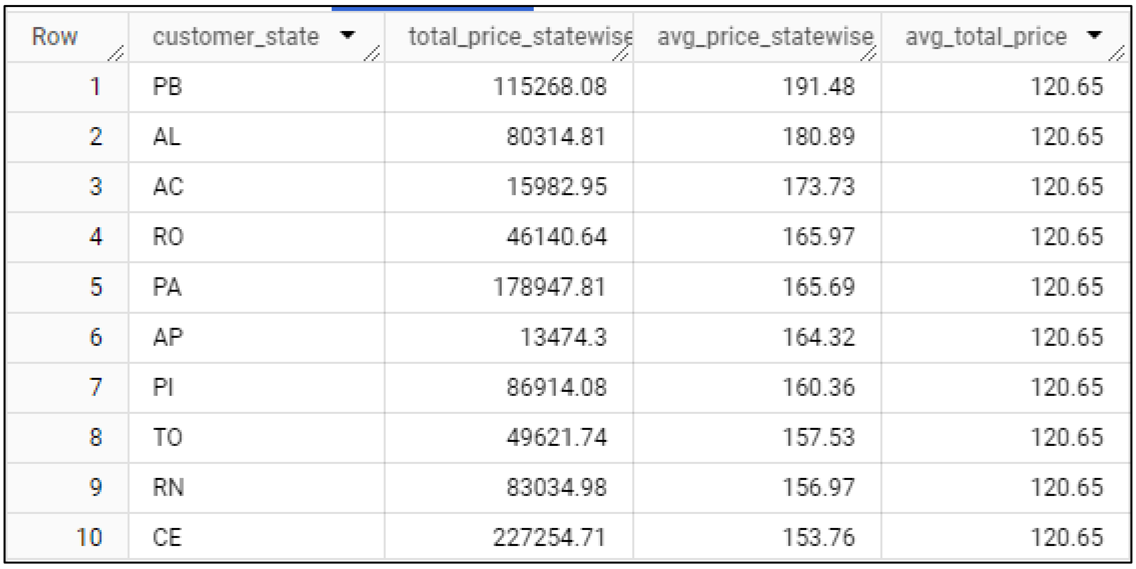
Insights:

We can see pretty good increase for the month of 2017-02 as increment is 110%. For the month of April (2017-04) there is a drop of 7% in cost and similar situation in 2017-05 and 2018-02 with drop of 13.77% and almost 11% respectively. Overall year wise there is almost 137% increment shown.

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

Ans.

```
select c.customer_state,
  round(sum(item.price), 2) as total_price_statewise,
  round(avg(item.price), 2) avg_price_statewise,
  (select round(avg(price), 2) from scalar-bignner-batch.Target.order_items) avg_to-
tal_price
from scalar-bignner-batch.Target.order_items item
join scalar-bignner-batch.Target.orders o using(order_id)
join scalar-bignner-batch.Target.customers c using(customer_id)
group by c.customer_state
order by avg_price_statewise desc;
```



Row	customer_state	total_price_statewise	avg_price_statewise	avg_total_price
1	PB	115268.08	191.48	120.65
2	AL	80314.81	180.89	120.65
3	AC	15982.95	173.73	120.65
4	RO	46140.64	165.97	120.65
5	PA	178947.81	165.69	120.65
6	AP	13474.3	164.32	120.65
7	PI	86914.08	160.36	120.65
8	TO	49621.74	157.53	120.65
9	RN	83034.98	156.97	120.65
10	CE	227254.71	153.76	120.65

Insights:

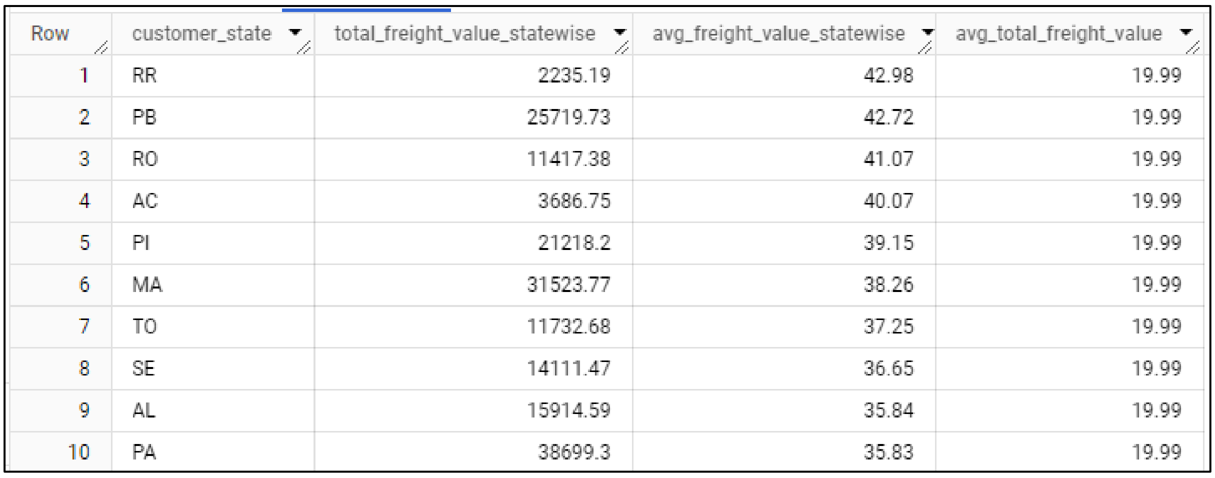
State-wise average price is more than the average total price for most of the states, The highest average price is for the state PB at 191.48 against the average total of 120.65.

Highest total price is with state SP as it holds highest number of orders and customers, yet SP is having average price at 109.65 which is lowest among all states.

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

Ans.

```
select c.customer_state,
  round(sum(item.freight_value), 2) as total_freight_value_statewise,
  round(avg(item.freight_value), 2) avg_freight_value_statewise,
  (select round(avg(freight_value), 2) from scalar-bignner-batch.Target.order_items)
avg_total_freight_value
from scalar-bignner-batch.Target.order_items item
join scalar-bignner-batch.Target.orders o using(order_id)
join scalar-bignner-batch.Target.customers c using(customer_id)
group by c.customer_state
order by avg_freight_value_statewise desc;
```



Row	customer_state	total_freight_value_statewise	avg_freight_value_statewise	avg_total_freight_value
1	RR	2235.19	42.98	19.99
2	PB	25719.73	42.72	19.99
3	RO	11417.38	41.07	19.99
4	AC	3686.75	40.07	19.99
5	PI	21218.2	39.15	19.99
6	MA	31523.77	38.26	19.99
7	TO	11732.68	37.25	19.99
8	SE	14111.47	36.65	19.99
9	AL	15914.59	35.84	19.99
10	PA	38699.3	35.83	19.99

Insights:

States with low number of customers and orders like RR, PB is having very high average freight value - around 42 to 43 (delivery price) compared to average total value of 19.99 whereas states with higher orders like SP having as low as 15.15 average freight value.

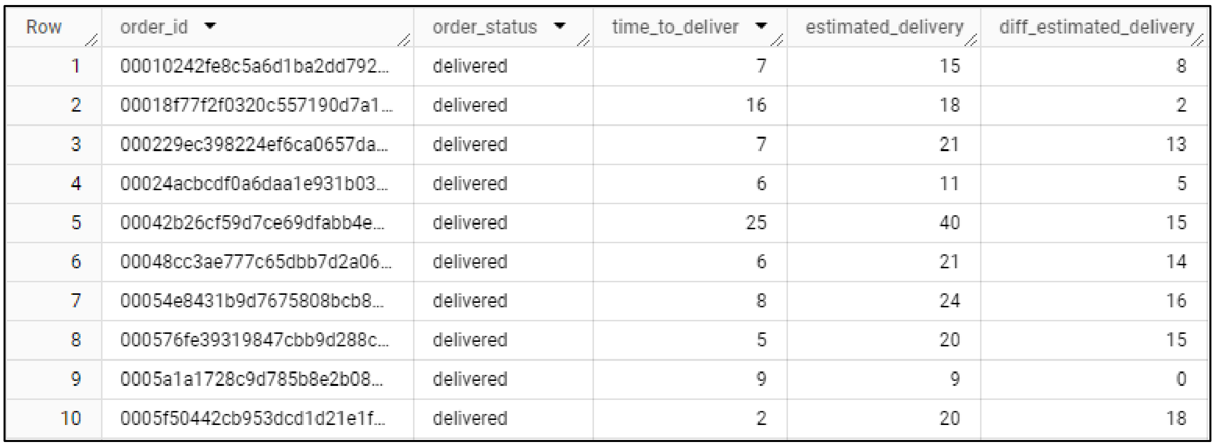
V. **Analysis based on sales, freight and delivery time.**

A. **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.

Ans.

```
select order_id,
  order_status,
  date_diff(order_delivered_customer_date, order_purchase_timestamp, day) time_to_deliver,
  date_diff(order_estimated_delivery_date, order_purchase_timestamp, day)
estimated_delivery,
  date_diff(order_estimated_delivery_date, order_delivered_customer_date, day)
diff_estimated_delivery
from scalar-bignner-batch.Target.orders
order by order_id
```

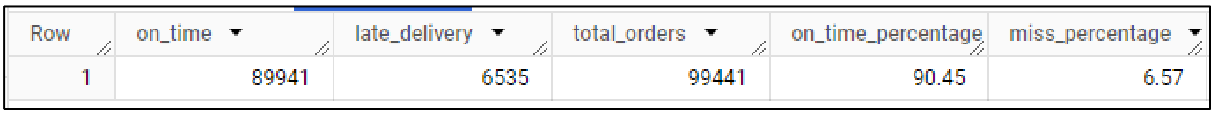


Row	order_id	order_status	time_to_deliver	estimated_delivery	diff_estimated_delivery
1	00010242fe8c5a6d1ba2dd792...	delivered	7	15	8
2	00018f77f2f0320c557190d7a1...	delivered	16	18	2
3	000229ec398224ef6ca0657da...	delivered	7	21	13
4	00024acbcdf0a6daa1e931b03...	delivered	6	11	5
5	00042b26cf59d7ce69dfabb4e...	delivered	25	40	15
6	00048cc3ae777c65dbb7d2a06...	delivered	6	21	14
7	00054e8431b9d7675808bcb8...	delivered	8	24	16
8	000576fe39319847cbb9d288c...	delivered	5	20	15
9	0005a1a1728c9d785b8e2b08...	delivered	9	9	0
10	0005f50442cb953dcd1d21e1f...	delivered	2	20	18

Insights:

Well, this doesn't give much of meaningful insights other than higher the difference in estimated delivery faster the order arrives at customer's doorstep. So, let's check further using above results as our base table how much orders delivered on time and how much got delayed.

```
with cte as (select order_id,
  order_status,
  date_diff(order_delivered_customer_date, order_purchase_timestamp, day) time_to_deliver,
  date_diff(order_estimated_delivery_date, order_purchase_timestamp, day)
estimated_delivery,
  date_diff(order_estimated_delivery_date, order_delivered_customer_date, day)
diff_estimated_delivery
from scalar-bignner-batch.Target.orders
order by order_id)

select
  sum(case when diff_estimated_delivery >= 0 then 1 end) on_time,
  sum(case when diff_estimated_delivery < 0 then 1 end) late_delivery,
  count(*) total_orders,
  round(sum(case when diff_estimated_delivery >= 0 then 1 end) * 100 / count(*), 2)
on_time_percentage,
  round(sum(case when diff_estimated_delivery < 0 then 1 end) * 100 / count(*), 2)
miss_percentage
from cte;
```



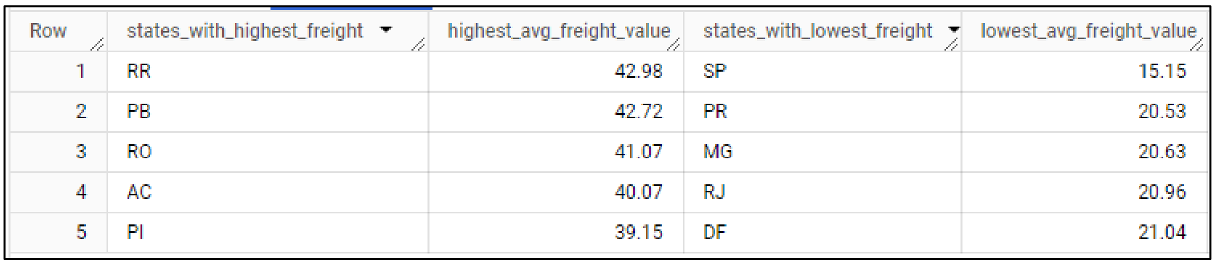
Row	on_time	late_delivery	total_orders	on_time_percentage	miss_percentage
1	89941	6535	99441	90.45	6.57

So, basically around 90% orders got delivered within estimated time period and around 6.5% missed estimated time period and most probably remaining of the percentage either having shipped or got cancelled at the time of data extraction.

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

Ans.

```
with highest_freight as (select * ,
  row_number() over(order by avg_freight_value_statewise desc) as a
from
  (select c.customer_state,
    round(avg(item.freight_value), 2) avg_freight_value_statewise
  from scalar-bignner-batch.Target.order_items item
  join scalar-bignner-batch.Target.orders o using(order_id)
  join scalar-bignner-batch.Target.customers c using(customer_id)
  group by c.customer_state
) high),

lowest_freight as (select * ,
  row_number() over(order by avg_freight_value_statewise) as a
from
  (select c.customer_state,
    round(avg(item.freight_value), 2) avg_freight_value_statewise
  from scalar-bignner-batch.Target.order_items item
  join scalar-bignner-batch.Target.orders o using(order_id)
  join scalar-bignner-batch.Target.customers c using(customer_id)
  group by c.customer_state
) low)

select highest_freight.customer_state as states_with_highest_freight,
  highest_freight.avg_freight_value_statewise highest_avg_freight_value,
  lowest_freight.customer_state as states_with_lowest_freight,
  lowest_freight.avg_freight_value_statewise lowest_avg_freight_value
from highest_freight join lowest_freight on highest_freight.a = lowest_freight.a
limit 5;
```



Row	states_with_highest_freight	highest_avg_freight_value	states_with_lowest_freight	lowest_avg_freight_value
1	RR	42.98	SP	15.15
2	PB	42.72	PR	20.53
3	RO	41.07	MG	20.63
4	AC	40.07	RJ	20.96
5	PI	39.15	DF	21.04

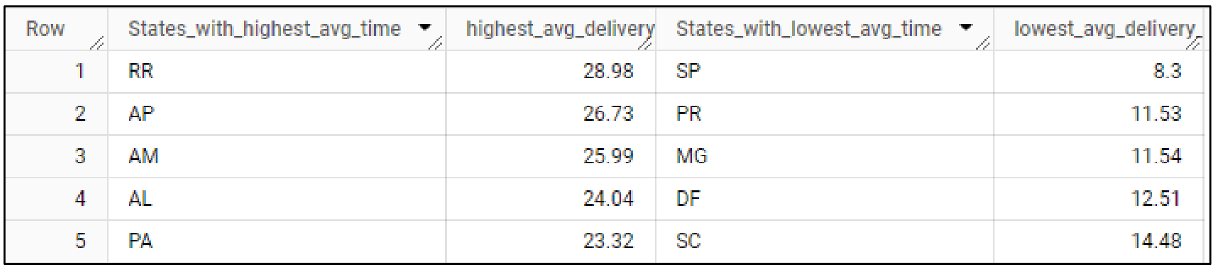
Insights:

States like RR, PB, RO, AC, PI are with the highest average freight value ranging between 39.15 and 42.98 whereas states like SP, PR, MG, RJ, DF are with lowest average ranging between 15.15 and 21.04

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

Ans.

```sql
with highest_avg_del_time as (select *,
  row_number() over(order by avg_delivery_time desc) numbers
from (
  select t.customer_state,
    round(avg(time_to_deliver), 2) avg_delivery_time
  from (
    select c.customer_state,
      date_diff(order_delivered_customer_date, order_purchase_timestamp, day)
time_to_deliver
    from scalar-bignner-batch.Target.orders o
    join scalar-bignner-batch.Target.customers c using(customer_id)
  ) t
  group by t.customer_state
)),

lowest_avg_del_time as (select *,
  row_number() over(order by avg_delivery_time) numbers
from (
  select t.customer_state,
    round(avg(time_to_deliver), 2) avg_delivery_time
  from (
    select c.customer_state,
      date_diff(order_delivered_customer_date, order_purchase_timestamp, day)
time_to_deliver
    from scalar-bignner-batch.Target.orders o
    join scalar-bignner-batch.Target.customers c using(customer_id)
  ) t
  group by t.customer_state
))

select highest_avg_del_time.customer_state as States_with_highest_avg_time,
  highest_avg_del_time.avg_delivery_time as highest_avg_delivery_time,
  lowest_avg_del_time.customer_state as States_with_lowest_avg_time,
  lowest_avg_del_time.avg_delivery_time as lowest_avg_delivery_time
from highest_avg_del_time join lowest_avg_del_time using(numbers)
limit 5;
```



Row	States_with_highest_avg_time	highest_avg_delivery	States_with_lowest_avg_time	lowest_avg_delivery
1	RR	28.98	SP	8.3
2	AP	26.73	PR	11.53
3	AM	25.99	MG	11.54
4	AL	24.04	DF	12.51
5	PA	23.32	SC	14.48

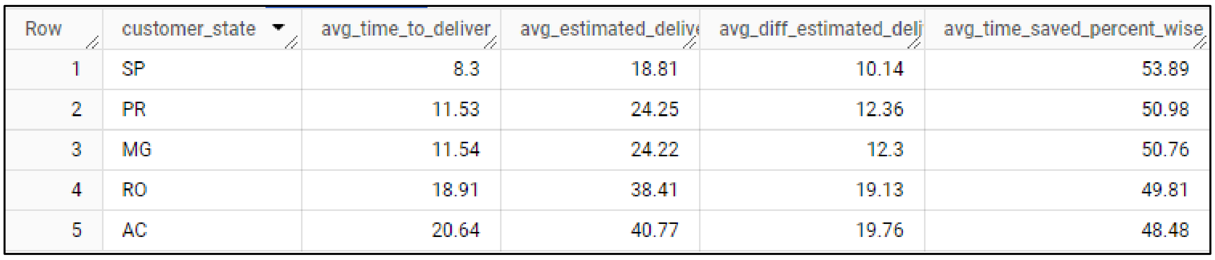
Insights:

States like RR, AP, AM, AL, and PA seems to be having highest average delivery time with days reaching up to 30 days (23 days – 29 days on average). It's most likely customers from these states will order less due to huge waiting time involved.
Whereas states like SP, PR, MG, DF, and SC having lowest average delivery time, making customers to order more.

D. Find out the top 5 states where the order delivery is really fast as compared to the estimated date of delivery.

Ans.

```
select c.customer_state,
  round(avg(date_diff(order_delivered_customer_date, order_purchase_timestamp, day)), 2)
avg_time_to_deliver,
  round(avg(date_diff(order_estimated_delivery_date, order_purchase_timestamp, day)), 2)
avg_estimated_delivery,
  round(avg(date_diff(order_estimated_delivery_date, order_delivered_customer_date, day)),
2) avg_diff_estimated_delivery,
  round(avg(date_diff(order_estimated_delivery_date, order_delivered_customer_date, day)) *
100 / avg(date_diff(order_estimated_delivery_date, order_purchase_timestamp, day)), 2)
avg_time_saved_percent_wise
from scalar-bignner-batch.Target.orders o
join scalar-bignner-batch.Target.customers c using(customer_id)
group by customer_state
order by avg_time_saved_percent_wise desc
limit 5
```



Row	customer_state	avg_time_to_deliver	avg_estimated_delive	avg_diff_estimated_deli	avg_time_saved_percent_wise
1	SP	8.3	18.81	10.14	53.89
2	PR	11.53	24.25	12.36	50.98
3	MG	11.54	24.22	12.3	50.76
4	RO	18.91	38.41	19.13	49.81
5	AC	20.64	40.77	19.76	48.48

Insights:

When compared to estimated delivery dates, on an average these states like SP, PR, MG, RO, AC delivers orders very fast.

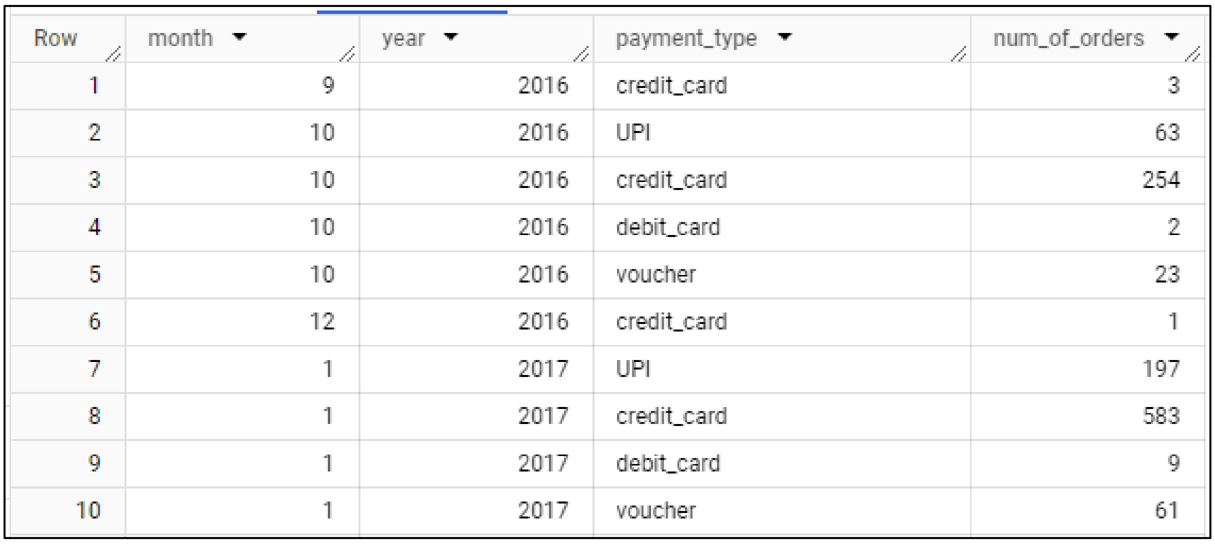
If we take out SP state as an example, we can see on an average order gets delivered 10.14 days before the estimated delivery time which means 53.89% of estimated time got saved.
Similarly, in case of AC (which holds 5th place), average difference between actual to estimated time is 19.76 days which means it saved 48.48% of estimated time making it 5th fastest state to deliver orders.

VI. Analysis based on the payments:

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

Ans.

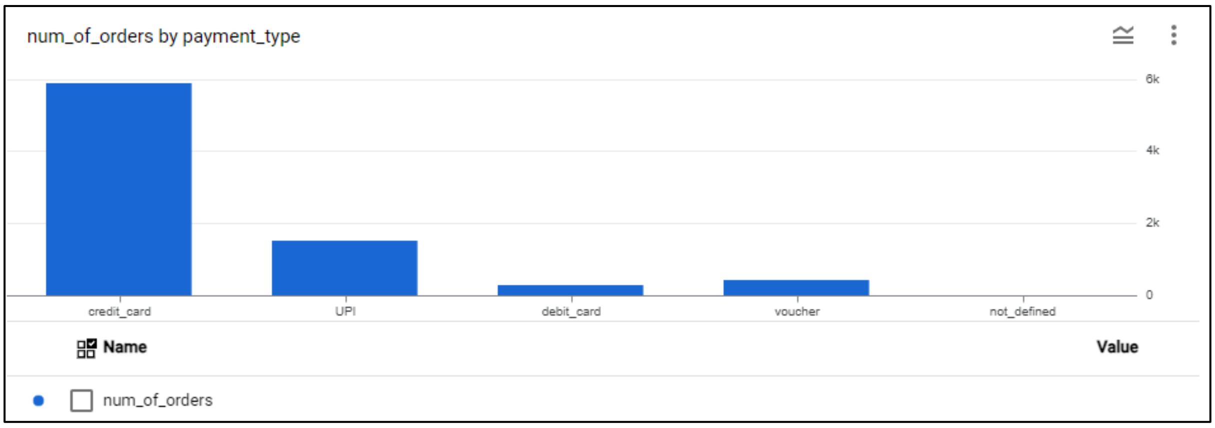
```
select extract(MONTH from o.order_purchase_timestamp) month,
  extract(YEAR from o.order_purchase_timestamp) year,
   p.payment_type,
   count(*) as num_of_orders
from scalar-bignner-batch.Target.orders o
join scalar-bignner-batch.Target.payments p using(order_id)
group by year, month, p.payment_type
order by year, month, p.payment_type
```



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

Insights:

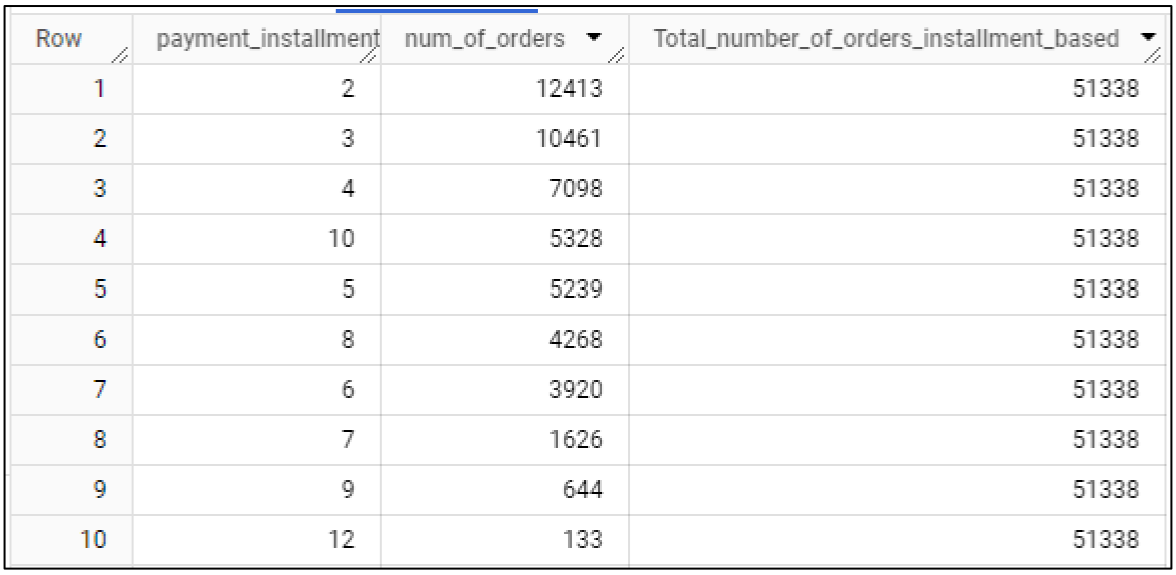
Credit card seems to be most popular mode of payment among customers followed by UPI, Voucher and then Debit card.



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

Ans.

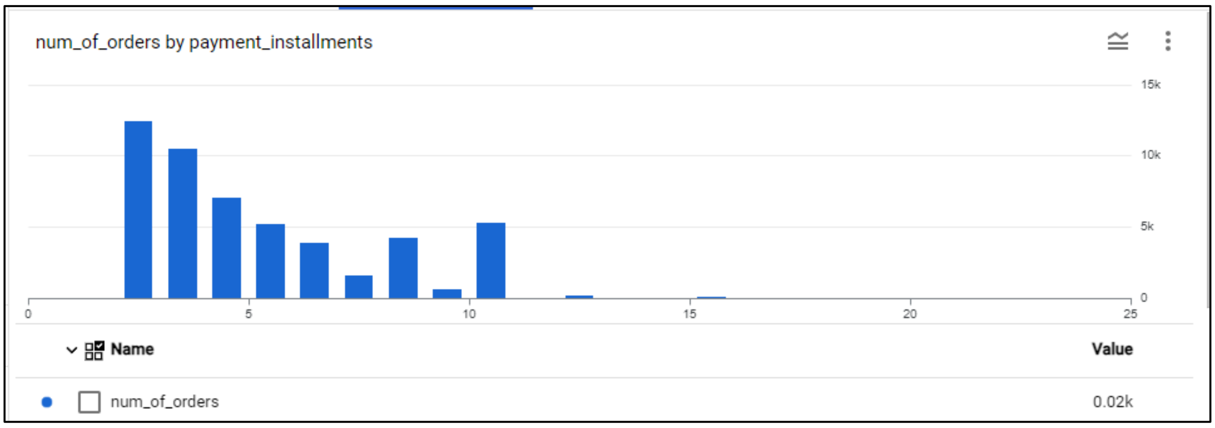
```
select payment_installments,
  count(*) num_of_orders,
  sum(count(*)) over() Total_number_of_orders_installment_based
from scalar-bignner-batch.Target.payments
where payment_installments > 1 and payment_sequential > 0
group by payment_installments
order by num_of_orders desc
```



Row	payment_installment	num_of_orders	Total_number_of_orders_installment_based
1	2	12413	51338
2	3	10461	51338
3	4	7098	51338
4	10	5328	51338
5	5	5239	51338
6	8	4268	51338
7	6	3920	51338
8	7	1626	51338
9	9	644	51338
10	12	133	51338

Insights:

Number of orders paid via installments are 51338. Customers mostly opted the 2 and 3 installment options. Higher the number of installments less more it is.



Overall Insights and Summary:

- We are having 772 days data of dates between 2016-09-04 and 2018-10-17 in Brazil.

- Dataset contains details of customer's orders ordered from 4119 cities spread across 27 states of Brazil.
- Trend Index on cumulative average for year 2017 shows 0.9855 (normalized value) and 0.6294 proving to have a positive growth trend in number of orders.
- Monthly Seasonality Index clearly shows how some have seasonality above cumulative average (index value greater than 1).
- We also came to conclude the Brazilian customers mostly make orders during Afternoon (38.35% of total orders), followed by Night and Morning. During Dawn very few orders were placed (only 5.27% of total orders).
- Month-on-month state-wise data shows that highest number of orders were placed from SP state, followed by RJ and MG.
- Upon checking the customer distribution across states, we came to know that almost 42% of total customers are from SP, hence there is possibility of bias towards SP state. SP is followed by RG with 12.92% and MG with 11.7% of total customer base.
- Upon checking cost of order yearly data (including months between Jan to Aug only), we could see an overall percentage increase of 136.98 % which shows good growth economically.
- PB is having highest average price of 191.48, followed by AL and AC with 180.89 and 173.73 respectively. Whereas Lowest average price is with SP at 109.85.
- RR and PB are having very high freight value (delivery price) - 42.98 and 42.72 respectively followed by RO, AC, PI whereas SP having lowest at 15.15 only followed by PR, MG, RJ, DF.
- Based on delivery time, almost 90.45% of orders got delivered within estimated delivery date whereas 6.57% missed the estimated date. Remaining percentage is of those orders which either got cancelled or were yet to be delivered at the time of data extraction.
- States with highest average delivery time (slowest delivery) are RR with 29 days average delivery time followed by AP, AM, AL, PA whereas states with shortest delivery time is SP which takes on average of 8.3 days followed by PR, MG, DF, SC.
- When comparing delivery time with estimated time, states with 5 fastest order delivery are SP, PR, MG along with RO and AC joining in at 4th and 5th place.
- While looking at on month-on-month orders placed using payment types, we can clearly say credit card is most famous among customers followed by UPI, Voucher, and then Debit card.
- Total 51338 are the total number of orders which have been paid using EMI or installments.

Recommendations:

- In order to increase the customer base Target should focus in states like RG and MG where it has good customer base but not good enough when compare to SP state.
- Focusing on SP, RG, and MG should be of higher priority as it offers significant market potential.
- Work on optimizing the delivery time in states with higher average delivery time like RR, AP etc.
- During dawn, we see less orders, Target can explore the opportunities by offering special discounts and promotions to attract early customers.

- Improve logistics services, target may explore some local delivery partners to enhance customer satisfaction by providing faster delivery services.
- Consider to re-adjust average pricing and freight costs in order to remain competitive. Especially, in states with too high freight costs or too low average prices.
- Consider diversifying customer base by using different kind of market strategies in different states, don't just neglect them.
- Consider exploring the payment options, offer some incentives with other payments options to broad the customer base and ensure to provide EMI options as there is a significant number of orders paid using installments.