Problem Statement:

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

What does 'good' look like?

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

1. Data type of all columns in the "customers" table.

Structure

Field name	Type	Mode	Key	Collation
customer_id	STRING	NULLABLE	-	-
customer_unique_id	STRING	NULLABLE	-	-
customer_zip_code_prefix	INTEGER	NULLABLE	-	-
customer_city	STRING	NULLABLE	-	-
customer_state	STRING	NULLABLE	-	-

Sample Content of Customer table

Row	customer_id	customer_unique_id	customer_zip_code_	customer_city	customer_state
1	0735e7e4298a2ebbb4664934...	fcb003b1bdc0df64b4d065d9b...	59650	acu	RN
2	903b3d86e3990db01619a4eb...	46824822b15da44e983b021d...	59650	acu	RN
3	38c97666e962d4fea7fd6a83e...	b6108acc674ae5c99e29adc10...	59650	acu	RN
4	77c2f46cf580f4874c9a5751c2...	402cce5c0509000eed9e77fec...	63430	ico	CE
5	4d3ef4cfffb8ad4767c199c36a...	6ba00666ab7eada5ceec279b2...	63430	ico	CE
6	3000841b86e1fbe9493b52324...	796a0b1a21f597704057184a1...	63430	ico	CE
7	3c325415ccc7e622c66dec4bc...	05d1d2d9f0161c5f397ce7fc77...	63430	ico	CE
8	04f3a7b250e3be964f01bf22bc...	c34585a0276ecc5e4fb03de75...	63430	ico	CE

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

```
select
      min(order_purchase_timestamp) as Minimum,
      max(order_purchase_timestamp) as Maximum
from
      `target.orders`;
```

```
1  select min(order_purchase_timestamp) as Minimum, max(order_purchase_timestamp)as Maximum from `target.orders`;
```

Query results

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

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

Insights:

- Time range in which orders were placed is from 4, September 2016 to 17, October 2018.

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

```
select
      count(distinct customer_state) Count_of_State,
      count(distinct customer_city) Count_of_Cities
from
      (select * from `target.customers` inner join `target.orders`
using(customer_id)) s;
```

```
1  select count(distinct customer_state) Count_of_State, count(distinct customer_city) Count_of_Cities
2  from (select *
3  from `target.customers` inner join `target.orders`
4  using(customer_id)) s ;
```

Query results

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

Row	Count_of_State	Count_of_Cities
1	27	4119

Insights:

- From 27 different states and 4119 different cities order were placed during the given period.

2. **In-depth Exploration:**

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

```
select
      extract(year from order_purchase_timestamp) Year,
      count(*) Count_of_orders
from `target.orders`
```

```
group by extract(year from order_purchase_timestamp)
order by year;
```

Row	Year	Count_of_orders
1	2016	329
2	2017	45101
3	2018	54011

Insights:

- Though in 2016, 4 months are considered and in 2018 records are till month of October. It is clearly visible that, number of orders have been increases in each coming year.

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

```
select
      extract(month from order_purchase_timestamp) Month,
      FORMAT_DATETIME("%B", DATETIME(order_purchase_timestamp)) Month_Name,
      count(*) Count_of_orders
from `target.orders`
group by
      extract(Month from order_purchase_timestamp),
      FORMAT_DATETIME("%B", DATETIME(order_purchase_timestamp))
order by Month;
```

Row	Month	Month_Name	Count_of_orders
1	1	January	8069
2	2	February	8508
3	3	March	9893
4	4	April	9343
5	5	May	10573
6	6	June	9412
7	7	July	10318
8	8	August	10843
9	9	September	4305
10	10	October	4959
11	11	November	7544
12	12	December	5674

Insights:

- From the result it has been observed that, on an average first half of the year is the period where more orders are placed compared to latter half of the year.
- Specifically, the month of August, May and July are the months in order from highest orders to lowest orders placed.
- In September and October, lowest numbers of orders are placed.

3. 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

```
select
      Time_of_day,
      count(*) Count_of_orders
from
      (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
'Mornings'
      when extract(hour from order_purchase_timestamp) between 13 and 18 then
'Afternoon'
      when extract(hour from order_purchase_timestamp) between 19 and 23 then
'Night'
      end Time_of_day
from `target.orders`) s
group by
      Time_of_day
order by
      count(*);
```

Row	Time_of_day	Count_of_orders
1	Dawn	5242
2	Mornings	27733
3	Night	28331
4	Afternoon	38135

Insights:

- From the result it is seen that most orders are placed in the afternoon between 1 p.m. to 6 p.m. and in the dawn, least orders are placed.
- It can be inferred that customers are more relaxed after having lunch and prefer to order more.

- Customers can be given more deals during afternoon time to encourage more customers to order.
- More deals, offers can be given in the night and morning time to attract more orders.

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

1. Get the month on month no. of orders placed in each state.

```
select
      customer_state,
      extract(month from order_purchase_timestamp) Month,
      FORMAT_DATETIME("%B", DATETIME(order_purchase_timestamp)) Month_Name,
      count(*) Count_of_orders
from
      (select * from `target.customers` inner join `target.orders`
using(customer_id)) s
group by
      customer_state,
      extract(month from order_purchase_timestamp),
      FORMAT_DATETIME("%B", DATETIME(order_purchase_timestamp))
order by
      count(*) desc;
```

Row	customer_state	Month	Month_Name	Count_of_orders
1	SP	8	August	4982
2	SP	5	May	4632
3	SP	7	July	4381
4	SP	6	June	4104
5	SP	3	March	4047
6	SP	4	April	3967
7	SP	2	February	3357
8	SP	1	January	3351
9	SP	11	November	3012
10	SP	12	December	2357

Insights:

- 322 lines are returned.
- Result is sorted from most to least according to the count of orders placed in a month for a state.
- It is observed that, most orders are placed in month of august in a state having code SP.
- Customers in state having code SP are placing maximum orders in all months.

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

```
select
      customer_state,
      count(*) Customer_per_state_count
from `target.customers`
group by
      customer_state
order by
      count(*) desc;
```

Row	customer_state	Customer_per_state_
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
11	PE	1652

Insights:

- From the output it is observed that, there are total 27 states in Brazil.
- The state having code SP has most number of customer living in it.
- Followed by SP, State code having RJ and MG have 2nd and 3rd number of customers living in it i.e. 12852 and 11635.

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

1. 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.

```
with
cte1 as
(select extract(year from o.order_purchase_timestamp) year,
round(sum(p.payment_value),2) Total from `target.orders` o
inner join `target.payments` p
using(order_id)
where  extract(month from o.order_purchase_timestamp) between 1 and 8
group by extract(year from o.order_purchase_timestamp)
having year in (2017,2018)
),
cte2 as
```

```
(
  select *,
  lead(total)over(order by total desc) previous_total
  from cte1
  order by year desc
  limit 1
 ),
 cte3 as
 (
  select round(((total-previous_total)/previous_total*100),2)
percent_increase
  from cte2
 )

select * from cte3;
```

Row	percent_increase
1	136.98

Insights:

- From the output, it is seen that from 2017 to 2018 there is 136.98% hike in the cost of orders for months from January to August.
- For year 2018 and 2017, the cost of order was 8694733.84 and 3669022.12 respectively.

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

```
select
      c.customer_state State,
      round(sum(i.price),2) Total_Order_price,
      round(avg(i.price),2) Average_Order_Price
from `target.customers` c inner join `target.orders`o
using(customer_id)
inner join `target.order_items` i
using (order_id)
group by
      c.customer_state
order by
      total_order_price DESC,
      Average_order_price;
```

Row	State	Total_Order_price	Average_Order_Price
1	SP	5202955.05	109.65
2	RJ	1824092.67	125.12
3	MG	1585308.03	120.75
4	RS	750304.02	120.34
5	PR	683083.76	119.0
6	SC	520553.34	124.65
7	BA	511349.99	134.6
8	DF	302603.94	125.77
9	GO	294591.95	126.27
10	ES	275037.31	121.91

Insights:

- 27 rows are returned.
- It is observed that state having code SP is having highest Total order price among all state. Highest amount of orders is placed in SP state.
- It is further noticed that, total order price and average order price is nearly about inversely proportional to each other in orders.

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

```
select
      c.customer_state State,
      round(sum(i.freight_value),2) Total_Freight_value,
      round(avg(i.Freight_value),2) Average_freight_value
from `target.customers` c
inner join `target.orders`o
using(customer_id)
inner join `target.order_items` i
using (order_id)
group by
      c.customer_state
order by
      total_freight_value DESC,
      Average_freight_value;
```

Row	State	Total_Freight_value	Average_freight_valu
1	SP	718723.07	15.15
2	RJ	305589.31	20.96
3	MG	270853.46	20.63
4	RS	135522.74	21.74
5	PR	117851.68	20.53
6	BA	100156.68	26.36
7	SC	89660.26	21.47
8	PE	59449.66	32.92
9	GO	53114.98	22.77
10	DF	50625.5	21.04

Insights:

- 27 rows are returned.
- It is observed that state having code SP is having highest Freight value among all state.
- It is further noticed that, total Freight value and average Freight value is nearly about inversely proportional to each other in orders. Highest freight value has lowest average freight value and vice versa.

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

1. 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.

   You can calculate the delivery time and the difference between the estimated & actual delivery date using the given formula:
   - **time_to_deliver** = order_delivered_customer_date - order_purchase_timestamp
   - **diff_estimated_delivery** = order_delivered_customer_date - order_estimated_delivery_date

```
select  order_purchase_timestamp,
        order_delivered_carrier_date,
        order_delivered_customer_date,
        order_estimated_delivery_date,
        timestamp_diff(order_delivered_carrier_date,order_purchase_timestamp,
day) time_to_deliver,
        if(timestamp_diff(order_delivered_carrier_date,order_estimated_delive
ry_date,day)<0,-
timestamp_diff(order_delivered_carrier_date,order_estimated_delivery_date,day
```

```
),timestamp_diff(order_delivered_carrier_date,order_estimated_delivery_date,d
ay)) diff_estimated_delivery
from `target.orders`;
```

Row	order_purchase_timestamp	order_delivered_carrier_date	order_delivered_customer_date	order_estimated_delivery_date	time_to_deliver	diff_estimated_delivery
1	2018-07-11 20:24:49 UTC	2018-07-31 14:10:00 UTC	null	2018-08-01 00:00:00 UTC	19	0
2	2017-12-09 10:16:45 UTC	2017-12-18 17:43:38 UTC	null	2018-01-29 00:00:00 UTC	9	41
3	2018-06-13 18:44:19 UTC	2018-06-14 15:45:00 UTC	null	2018-07-24 00:00:00 UTC	0	39
4	2018-08-10 15:14:50 UTC	2018-08-13 13:44:00 UTC	null	2018-08-17 00:00:00 UTC	2	3
5	2017-05-13 21:23:34 UTC	2017-05-20 07:43:42 UTC	null	2017-06-27 00:00:00 UTC	6	37
6	2018-03-08 07:06:35 UTC	2018-03-09 17:01:37 UTC	null	2018-04-19 00:00:00 UTC	1	40
7	2018-08-05 07:21:56 UTC	2018-08-06 13:21:00 UTC	null	2018-08-09 00:00:00 UTC	1	2
8	2018-08-05 17:00:00 UTC	2018-08-06 15:18:00 UTC	null	2018-08-09 00:00:00 UTC	0	2
9	2018-05-16 13:03:16 UTC	2018-05-18 10:43:00 UTC	null	2018-06-25 00:00:00 UTC	1	37

Insights:

- From the result, it has been observed that, order is delivered to the customer before estimated delivery.

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

```
with cte as
(
select customer_state, round(avg(freight_value),2) Average_freight_value,
'Top 5 maximum' min_max from `target.order_items`
inner join `target.orders`
using (order_id)
inner join `target.customers`
using(customer_id)
group by customer_state
order by Average_freight_value desc
limit 5
),
cte2 as
(
select customer_state, round(avg(freight_value),2) Average_freight_value,
'Top 5 minimum' min_max from `target.order_items`
inner join `target.orders`
using (order_id)
inner join `target.customers`
using(customer_id)
group by customer_state
order by Average_freight_value
limit 5
)
select * from cte
union all
select * from cte2;
```

Row	customer_state	Average_freight_valu	min_max
1	RR	42.98	Top 5 maximum
2	PB	42.72	Top 5 maximum
3	RO	41.07	Top 5 maximum
4	AC	40.07	Top 5 maximum
5	PI	39.15	Top 5 maximum
6	SP	15.15	Top 5 minimum
7	PR	20.53	Top 5 minimum
8	MG	20.63	Top 5 minimum
9	RJ	20.96	Top 5 minimum
10	DF	21.04	Top 5 minimum

Insights:

- 10 rows are returned.
- Highest average freight value is 42.98
- Lowest average freight value is 15.15

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

```
with
cte as
(
select
customer_state,
round(avg(timestamp_diff(order_delivered_carrier_date,order_purchase_timestam
p,day)),2) Average_delivery_time_in_days,
'Top 5 maximum' min_max
from `target.order_items`
inner join `target.orders`
using (order_id)
inner join `target.customers`
using(customer_id)
group by customer_state
order by Average_delivery_time_in_days desc
limit 5
),
cte2 as
(
select
customer_state,
round(avg(timestamp_diff(order_delivered_carrier_date,order_purchase_timestam
p,day)),2) Average_delivery_time_in_days,
'Top 5 minimum' min_max
from `target.order_items`
inner join `target.orders`
```

```
using (order_id)
inner join `target.customers`
using(customer_id)
group by customer_state
order by Average_delivery_time_in_days
limit 5
)
select * from cte2
union all
select * from cte;
```

Row	customer_state	Average_delivery_time_in_days	min_max
1	RR	4.63	Top 5 maximum
2	MA	3.4	Top 5 maximum
3	SE	3.25	Top 5 maximum
4	RN	3.2	Top 5 maximum
5	AL	3.15	Top 5 maximum
6	AM	2.29	Top 5 minimum
7	RO	2.34	Top 5 minimum
8	GO	2.62	Top 5 minimum
9	MS	2.72	Top 5 minimum
10	MT	2.72	Top 5 minimum

Insights:

- 10 rows are returned.
- Highest average delivery time is 4.63 days
- Lowest average delivery time is 2.29 days

4. Find out the top 5 states where the order delivery is really fast as compared to the estimated date of delivery.
   You can use the difference between the averages of actual & estimated delivery date to figure out how fast the delivery was for each state.

```
with cte1 as
(
  select
        customer_id,
        customer_state
  from
        `target.customers`
),
cte2 as
(
    -- calcalate difference in estimate date and actual date
    -- positive value indicate EARLY delivery and negative indicate LATE
delivery
```

```sql
  select
        customer_id,
        date_diff(date(order_estimated_delivery_date),date(order_delivered_cu
stomer_date),day) difference,
  from `target.orders`
  where
        order_delivered_customer_date is not null
),
cte3 as
(
  select *
  from
      cte1 a
  inner join cte2
  using(customer_id)
  order by a.customer_state
),
cte4 as
(
  select
        customer_state, sum(difference) Total_number_of_days
  from cte3
  group by
        customer_state
  order by
        count(*) Desc
),
cte5 as
(
  select
        customer_state, count(*)
  from
        cte3
  group by
        customer_state
  order by
        count(*) desc
)

select * from cte4
limit 5;
```

Row	customer_state	Total_number_of_days
1	SP	448538
2	RJ	145356
3	MG	150349
4	RS	74337
5	PR	65546

Insights:

- 96476 out of 99441 customers received the delivery.
- From state SP total 40495 orders are ordered which is highest.
- State SP has done earliest deliveries by 448538 days among all states.
- State RJ, MG, RS, PR stand on 2nd, 3rd, 4th and 5th place accordingly.
- It is also observed that State SP has most number of deliveries ordered.
- As difference between 1st and 2nd rank is of 303182 days, it clearly indicates that state SP has very good network, infrastructure for delivery of orders.
- The performance in State SP is very good in terms of number of orders and in delivery of products.

6. **Analysis based on the payments:**

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

```
with cte1 as
(
    select
        distinct order_id
    from
        `target.payments`
),
cte2 as
(
    select
        order_id,
        order_purchase_timestamp
    from
        `target.orders`
),
cte3 as
(
    select order_id, payment_type
    from `target.payments`
),
cte4 as
(
    select
        *
    from
        cte1
```

```
    left join cte2
    using(order_id)
    left join cte3
    using(order_id)
    order by
        order_id
),
cte5 as
(
    select
        format_timestamp('%m',order_purchase_timestamp) Month,
        payment_type,
        count(*) Payment_type_count_per_month
    from
        cte4
    group by
        Month,
        payment_type
    order by
        Month,
        payment_type
)

select * from cte5;
```

Row	Month	payment_type	Payment_type_count_per_month
1	01	UPI	1715
2	01	credit_card	6103
3	01	debit_card	118
4	01	voucher	477
5	02	UPI	1723
6	02	credit_card	6609
7	02	debit_card	82
8	02	voucher	424
9	03	UPI	1942
10	03	credit_card	7707

Insights:

- 96440 distinct orders are shown.
- 5 different types of payment types are used which credit card, voucher, debit card, UPI and not defined.
- Total records are 103886
- Majority customers used credit card for payment which is 76795. At second position UPI (19784) is used for payment. Voucher is used for 5775 orders and debit card is used for 1529 payments. 3 payment types are not defined.

- From the output it is clearly visible that in every month payment done by credit card is at top followed by UPI.

2. Find the no. of orders placed on the basis of the payment instalments that have been paid.

```
with

cte1 as

(

  select  order_id,

          count(*) count,

          max(payment_sequential) paid

  from `target.payments`

  group by order_id

  having count(*) = max(payment_sequential)

  order by count(*) desc

),

cte2 as

(

  select

          count(order_id) No_of_orders

  from cte1

)

select * from cte2;
```

Row	No_of_orders
1	99360

Insights:

- 99360 orders out of 99440 have fully paid their instalments.