Business Case: Target SQL

Context:

Target is a globally renowned brand and a prominent retailer in the United States. Target makes itself a preferred shopping destination by offering outstanding value, inspiration, innovation and an exceptional guest experience that no other retailer can deliver.

This particular business case focuses on the operations of Target in Brazil and provides insightful information about 100,000 orders placed between 2016 and 2018. The dataset offers a comprehensive view of various dimensions including the order status, price, payment and freight performance, customer location, product attributes, and customer reviews.

By analyzing this extensive dataset, it becomes possible to gain valuable insights into Target's operations in Brazil. The information can shed light on various aspects of the business, such as order processing, pricing strategies, payment and shipping efficiency, customer demographics, product characteristics, and customer satisfaction levels.

Problem Statement:

Assuming you are a data analyst/ scientist at Target, you have been assigned the task of analyzing 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:**
    a) Data type of all columns in the "customers" table.
    b) Get the time range between which the orders were placed.
    c) Count the number of Cities and States in our dataset.

Answer:

Q.1 (a) Data type of all columns in the "customers" table.

Answer:

- Query:

```
select table_name,column_name,data_type from
Target_sql.INFORMATION_SCHEMA.COLUMNS where table_schema="Target_sql" and
table_name="customers"
```

- Output Screenshot (top 10 rows):

Row	table_name	column_name	data_type
1	customers	customer_id	STRING
2	customers	customer_unique_id	STRING
3	customers	customer_zip_code_prefix	INT64
4	customers	customer_city	STRING
5	customers	customer_state	STRING

- Valuable insights: Not Applicable

Q.1 (b) Get the time range between which the orders were placed

Answer:

- Query:

```
select
  min(order_purchase_timestamp) as first_purchase_date,
  max(order_purchase_timestamp) as last_purchase_date
from `Target_sql.orders`
```

- Output Screenshot (top 10 rows):

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

- Valuable insights: orders were placed in between 4th Sep,2016 and 17th Oct,2018.

Q.1 (c) Count the number of Cities and States in our dataset.

Answer:

- Query:

```
select
  count(distinct geolocation_city) as city_count,
  count(distinct geolocation_state) as state_count
from `Target_sql.geolocation`
```

- Output Screenshot (top 10 rows):

Row	city_count	state_count
1	8011	27

- Valuable insights: City count is 8011 and state count is 27.

2. **In-depth Exploration:**

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

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

   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

Answer:

Q.2 (a) Is there a growing trend in the no. of orders placed over the past years?

Answer:

- Query:

```
select *, case
when order_count_per_year>ifNull(previous_year_order_count,0) then
"YES_Growing"
when order_count_per_year=ifNull(previous_year_order_count,0) then
"Same_as_Previous_Year"
else "Not_Growing"
end as is_Yearly_Growing
from
(
  select *,
  lag(final.order_count_per_year,1) over(order by final.Ordered_year asc) as
previous_year_order_count
  from
    (select t.Ordered_year,count(t.order_id) as order_count_per_year
      from
        (select order_id,extract(year from order_purchase_timestamp ) as
Ordered_year
          from `Target_sql.orders`)t
          group by t.Ordered_year
    ) final
    order by final.Ordered_year
)growing_Order_trend_analysis
```

- Output Screenshot (top 10 rows):

Row	Ordered_year	order_count_per_year	previous_year_order_count	is_Yearly_Growing
1	2016	329	*null*	YES_Growing
2	2017	45101	329	YES_Growing
3	2018	54011	45101	YES_Growing

- Valuable insights: Yes, Every Year, it is growing since 2016.

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

Answer:

- Query:

```
  with final_table as(
    select *, case
    when
order_count_per_month_year>ifNull(previous_month_same_year_order_count,0) then
"YES_Monthly_Growing"
    when
order_count_per_month_year=ifNull(previous_month_same_year_order_count,0) then
"Same_as_Previous_Month"
    else "Not_Growing"
    end as is_monthly_Growing
    from
    (
          select *,
          lag(final.order_count_per_month_year,1) over(order by
final.Ordered_year , Ordered_Month_in_year asc) as
previous_month_same_year_order_count
          from

          (
            select t.Ordered_year,t.Ordered_Month_in_year,count(t.order_id) as
order_count_per_month_year
                  from
              (select order_id,extract(year from order_purchase_timestamp ) as
Ordered_year ,
                extract(Month from order_purchase_timestamp ) as
Ordered_Month_in_year
                    from `Target_sql.orders`) t
                    group by t.Ordered_year,t.Ordered_Month_in_year
```

```
        )final
        order by final.Ordered_year,final.Ordered_Month_in_year
    )Monthly_growing_Order_trend_analysis
    order by
Monthly_growing_Order_trend_analysis.Ordered_Month_in_year,Monthly_growing_Ord
er_trend_analysis.Ordered_year
  )

 select distinct
f1.Ordered_year,f1.Ordered_Month_in_year,f1.is_monthly_Growing, "Season_Month"
as
is_Season_Month,f1.order_count_per_month_year,f1.previous_month_same_year_orde
r_count from final_table  f1
 where f1.Ordered_Month_in_year in (
    select distinct f2.Ordered_Month_in_year from final_table  f2 where
f2.Ordered_Month_in_year in (select distinct f3.Ordered_Month_in_year from
final_table  f3 where
  f3.is_monthly_Growing="YES_Monthly_Growing" ) group by
f2.Ordered_Month_in_year having count(distinct f2.is_monthly_Growing)=1)

  order by f1.Ordered_Month_in_year,f1.Ordered_year
```

- Output Screenshot (top 10 rows):

Row	Ordered_year	Ordered_Month_in_y	is_monthly_Growing	is_Season_Month	order_count_per_mo	previous_month_sar
1	2017	1	YES_Monthly_Growing	Season_Month	800	1
2	2018	1	YES_Monthly_Growing	Season_Month	7269	5673
3	2017	3	YES_Monthly_Growing	Season_Month	2682	1780
4	2018	3	YES_Monthly_Growing	Season_Month	7211	6728
5	2017	7	YES_Monthly_Growing	Season_Month	4026	3245
6	2018	7	YES_Monthly_Growing	Season_Month	6292	6167
7	2017	8	YES_Monthly_Growing	Season_Month	4331	4026
8	2018	8	YES_Monthly_Growing	Season_Month	6512	6292
9	2017	11	YES_Monthly_Growing	Season_Month	7544	4631

- Valuable insights: Season months are January,March,July,August and November in the year.

Q.2 (c) 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

Answer:

- Query:

```
with EXTRACT_TABLE AS
(Select order_id, order_purchase_timestamp, extract (HOUR from
order_purchase_timestamp ) as Purchased_Hour,
extract (MINUTE from order_purchase_timestamp ) as Purchased_Minutes
 from `Target_sql.orders` )
, Puchased_Time_Category_Table as
( SELECT ET.order_id, ET.order_purchase_timestamp,
ET.Purchased_Hour,ET.Purchased_Minutes,
  case
      when (ET.Purchased_Hour between 0 and 6 and ET.Purchased_Minutes between
0 and 59 )     then "0-6 hrs : Dawn"
      when (ET.Purchased_Hour between 7 and 12 and ET.Purchased_Minutes
between 0 and 59 )    then "7-12 hrs : Mornings"
      when (ET.Purchased_Hour between 13 and 18 and ET.Purchased_Minutes
between 0 and 59 )   then "13-18 hrs : Afternoon"
      when (ET.Purchased_Hour between 19 and 23 and ET.Purchased_Minutes
between 0 and 59 )   then "19-23 hrs : Night"

  end as Purchase_Time_Category

from EXTRACT_TABLE ET )

select ptct.Purchase_Time_Category,count(ptct.order_id) as
Order_Count_per_Category from Puchased_Time_Category_Table ptct group by
ptct.Purchase_Time_Category order by Order_Count_per_Category desc
```

- Output Screenshot (top 10 rows):

Row	Purchase_Time_Category	Order_Count_per_Category
1	13-18 hrs : Afternoon	38135
2	19-23 hrs : Night	28331
3	7-12 hrs : Mornings	27733
4	0-6 hrs : Dawn	5242

- Valuable insights: Maximum Order placed in Afternoon (13-18 hrs).

3. **Evolution of E-commerce orders in the Brazil region:**
   a) Get the month on month no. of orders placed in each state.
   b) How are the customers distributed across all the states?

Answer:

Q.3 (a) Get the month on month no. of orders placed in each state?

Answer:

- Query:

```
with join_table as ( select
o.order_id,o.order_purchase_timestamp,c.customer_state from
`Target_sql.orders` o join `Target_sql.customers` c on
o.customer_id=c.customer_id

), Extract_data as(
  select jt.*,extract(year from jt.order_purchase_timestamp ) as Ordered_year
,
                extract(Month from jt.order_purchase_timestamp ) as
Ordered_Month
      from join_table jt

)

select ED.customer_state,ED.Ordered_year,ED.Ordered_Month,count(ED.order_id)
as MOM_Order_count from Extract_data ED group by
 ED.customer_state,ED.Ordered_year,ED.Ordered_Month
order by ED.customer_state,ED.Ordered_year,ED.Ordered_Month
```

- Output Screenshot (top 10 rows):

Row	customer_state	Ordered_year	Ordered_Month	MOM_Order_count
1	AC	2017	1	2
2	AC	2017	2	3
3	AC	2017	3	2
4	AC	2017	4	5
5	AC	2017	5	8
6	AC	2017	6	4
7	AC	2017	7	5
8	AC	2017	8	4
9	AC	2017	9	5
10	AC	2017	10	6

- Valuable insights: NA.
- EXTRA_Insights: Top 10 States name having maximum order based on total_order_count are given below.

Row	customer_state	state_total_order_count	no_of_Months	state_Avg_order_count
1	SP	41746	24	1739.42
2	RJ	12852	23	558.78
3	MG	11635	22	528.86
4	RS	5466	22	248.45
5	PR	5045	22	229.32
6	SC	3637	22	165.32
7	BA	3380	21	160.95
8	DF	2140	21	101.9
9	ES	2033	21	96.81
10	GO	2020	21	96.19

Q.3 (b) How are the customers distributed across all the states?

Answer:

- Query:

```
select c.customer_state, count(c.customer_id) as
Customer_count_per_State,round((count(c.customer_id)/(select count(c1.customer_id) from
`Target_sql.customers` c1))*100,2) as Customer_Distribution_in_Percentage  from
`Target_sql.customers` c group by c.customer_state order by Customer_count_per_State desc
```

- Output Screenshot (top 10 rows):

Row	customer_state	Customer_count_per_State	Customer_Distribution_in_Percentage
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

- Valuable insights: Target have 41.98% customer base from State "SP". Then, two states (RJ and MG) are having 12.92% and 11.7 % customer base respectively. Rest 24 States are having appox. 33% of customer base.

4. **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.
   b) Calculate the Total & Average value of order price for each state.
   c) Calculate the Total & Average value of order freight for each state.

Answer:

Q.4 (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.

Answer:

- Query:

```
with Extract_data_table as
(select order_id , order_purchase_timestamp, Extract(Year from
order_purchase_timestamp) as Ordered_year,
Extract(Month from order_purchase_timestamp) as Ordered_Month_year
 from `Target_sql.orders` ) , filtered_extract_table as
 (
  select * from Extract_data_table where Ordered_Month_year between 1 and 8
and Ordered_year in(2018,2017)
 ) , Join_table as
 (
  select * from filtered_extract_table fet join `Target_sql.payments` pay on
fet.order_id=pay.order_id
 ),Yearly_cost_of_order as
 (select jt.Ordered_year,round(sum(jt.payment_value),2) as
Yearly_Total_ordered_Payment_Cost from Join_table as jt group by
jt.Ordered_year order by jt.Ordered_year ),Year_2017 as

(select Yearly_Total_ordered_Payment_Cost as Year_2017_Order_cost from
Yearly_cost_of_order Ycoo where Ycoo.Ordered_year=2017)
,Year_2018 as
(select Yearly_Total_ordered_Payment_Cost as Year_2018_Order_cost from
Yearly_cost_of_order Ycoo where Ycoo.Ordered_year=2018)
```

```
select Year_2017.Year_2017_Order_cost,Year_2018.Year_2018_Order_cost,Round((
(Year_2018.Year_2018_Order_cost - Year_2017.Year_2017_Order_cost)/
Year_2017.Year_2017_Order_cost ),4)*100 as
Yearly_OrderCost_increase_Percentage from Year_2017, Year_2018
```

- Output Screenshot (top 10 rows):

Row	Year_2017_Order_cost	Year_2018_Order_cost	Yearly_OrderCost_increase_Percentage
1	3669022.12	8694733.84	136.98

- Valuable insights: Total Order cost (Revenue) is increasing with almost 137% from Year 2017 to Year 2018.

Q.4 (b) Calculate the Total & Average value of order price for each state.

Answer:

- Query:

```
with order_customer_join as
(select o.order_id,o.customer_id , c.customer_state from `Target_sql.orders` o
join `Target_sql.customers` c on o.customer_id=c.customer_id ),
Order_item_sum_price as
(
  select oi.order_id,sum(oi.price)as total_order_price from
`Target_sql.order_items` oi group by oi.order_id
),
final_join_table as
(
  select ocj.*,oisp.* from order_customer_join ocj join Order_item_sum_price
oisp on ocj.order_id=oisp.order_id
)

select fjt.customer_state,Round(sum(fjt.total_order_price),2) as
State_wise_Total_OrderPrice, round(avg(fjt.total_order_price),2) as
State_wise_AVG_OrderPrice  from final_join_table fjt group by
fjt.customer_state order by State_wise_Total_OrderPrice
desc,State_wise_AVG_OrderPrice desc
```

- Output Screenshot (top 10 rows):

Row	customer_state	State_wise_Total_OrderPrice	State_wise_AVG_OrderPrice
1	SP	5202955.05	125.75
2	RJ	1824092.67	142.93
3	MG	1585308.03	137.33
4	RS	750304.02	138.13
5	PR	683083.76	136.67
6	SC	520553.34	144.12
7	BA	511349.99	152.28
8	DF	302603.94	142.4
9	GO	294591.95	146.78
10	ES	275037.31	135.82

- Valuable insights: Total_OrderPrice wise, top 5 states are SP,RJ,MG,RS and PR.
  -- AVG Order Price wise, top 5 states are PB,AP,AC,AL and RO as per shown below in table.

Row	customer_state	State_wise_Total_Ord	State_wise_AVG_Ord
1	PB	115268.08	216.67
2	AP	13474.3	198.15
3	AC	15982.95	197.32
4	AL	80314.81	195.41
5	RO	46140.64	186.8
6	PA	178947.81	184.48
7	TO	49621.74	177.86
8	PI	86914.08	176.3
9	MT	156453.53	173.26
10	RN	83034.98	172.27

Q.4 (c) Calculate the Total & Average value of order freight for each state.

Answer:

- Query:

```
with order_customer_join as
(select o.order_id,o.customer_id , c.customer_state from `Target_sql.orders` o join
`Target_sql.customers` c on o.customer_id=c.customer_id ), Order_item_sum_freight_value as
(
  select oi.order_id,sum(oi.freight_value)as total_order_freight_value from
`Target_sql.order_items` oi group by oi.order_id
),
final_join_table as
(
  select ocj.*,oisp.* from order_customer_join ocj join Order_item_sum_freight_value oisp
on ocj.order_id=oisp.order_id
)

select fjt.customer_state,Round(sum(fjt.total_order_freight_value),2) as
State_wise_Total_Order_freight_value, round(avg(fjt.total_order_freight_value),2) as
```

```
State_wise_AVG_Order_Freight_value from final_join_table fjt group by fjt.customer_state
order by State_wise_AVG_Order_Freight_value , State_wise_Total_Order_freight_value
```

- Output Screenshot (top 10 rows):

Row	customer_state	State_wise_Total_Order_freight_value	State_wise_AVG_Order_Freight_value
1	SP	718723.07	17.37
2	MG	270853.46	23.46
3	PR	117851.68	23.58
4	DF	50625.5	23.82
5	RJ	305589.31	23.95
6	ES	49764.6	24.58
7	SC	89660.26	24.82
8	RS	135522.74	24.95
9	GO	53114.98	26.46
10	MS	19144.03	27.0

- Valuable insights: AVG Freight Value wise, Top 5 states SP,MG,PR,DF and RJ are having least freight Value.

5. **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.

You can calculate the delivery time and the difference between the estimated & actual delivery date using the given formula:

1. **time_to_deliver** = order_delivered_customer_date - order_purchase_timestamp
2. **diff_estimated_delivery** = order_estimated_delivery_date - order_delivered_customer_date

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

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

d) 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.

Answer:

Q.5 (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.

Answer:

- Query:

```
select o.order_id,o.order_delivered_customer_date,o.order_purchase_timestamp,
datetime_Diff(o.order_delivered_customer_date, o.order_purchase_timestamp,day)
 as delivery_time_in_days, o.order_estimated_delivery_date,

(datetime_Diff(o.order_estimated_delivery_date,
o.order_delivered_customer_date,day)) as diff_estimated_delivery_in_Days

from `Target_sql.orders` o where o.order_delivered_customer_date is not null
```

- Output Screenshot (top 10 rows):

Row	order_id	order_delivered_customer_date	order_purchase_timestamp	delivery_time_in_days	order_estimated_delivery_date	diff_estimated_delivery_in_Days
1	1950d777989f6a877539f5379...	2018-03-21 22:03:51 UTC	2018-02-19 19:48:52 UTC	30	2018-03-09 00:00:00 UTC	-12
2	2c45c33d2f9cb8ff8b1c86cc28...	2016-11-09 14:53:50 UTC	2016-10-09 15:39:56 UTC	30	2016-12-08 00:00:00 UTC	28
3	65d1e226dfaeb8cdc42f66542...	2016-11-08 10:58:34 UTC	2016-10-03 21:01:41 UTC	35	2016-11-25 00:00:00 UTC	16
4	635c894d068ac37e6e03dc54e...	2017-05-16 14:49:55 UTC	2017-04-15 15:37:38 UTC	30	2017-05-18 00:00:00 UTC	1
5	3b97562c3aee8bdedcb5c2e45...	2017-05-17 10:52:15 UTC	2017-04-14 22:21:54 UTC	32	2017-05-18 00:00:00 UTC	0
6	68f47f50f04c4cb6774570cfde...	2017-05-16 09:07:47 UTC	2017-04-16 14:56:13 UTC	29	2017-05-18 00:00:00 UTC	1
7	276e9ec344d3bf029ff83a161c...	2017-05-22 14:11:31 UTC	2017-04-08 21:20:24 UTC	43	2017-05-18 00:00:00 UTC	-4
8	54e1a3c2b97fb0809da548a59...	2017-05-22 16:18:42 UTC	2017-04-11 19:49:45 UTC	40	2017-05-18 00:00:00 UTC	-4
9	fd04fa4105ee8045f6a0139ca5...	2017-05-19 13:44:52 UTC	2017-04-12 12:17:08 UTC	37	2017-05-18 00:00:00 UTC	-1
10	302bb8109d097a9fc6e9cefc5...	2017-05-23 14:19:48 UTC	2017-04-19 22:52:59 UTC	33	2017-05-18 00:00:00 UTC	-5

- Valuable insights: NA.

Q.5 (b) Find out the top 5 states with the highest & lowest average freight value.

Answer:

- Query:

```
with order_customer_join as
(select o.order_id,o.customer_id , c.customer_state from `Target_sql.orders` o join
`Target_sql.customers` c on o.customer_id=c.customer_id ), Order_item_sum_freight_value as
(
  select oi.order_id,sum(oi.freight_value)as total_order_freight_value from
`Target_sql.order_items` oi group by oi.order_id
),
final_join_table as
(
  select ocj.*,oisp.* from order_customer_join ocj join Order_item_sum_freight_value oisp
on ocj.order_id=oisp.order_id
)

(select fjt.customer_state, round(avg(fjt.total_order_freight_value),2) as
State_wise_AVG_Order_Freight_value,"top5Lowest" as is_highest_or_lowest from
final_join_table fjt group by fjt.customer_state
order by State_wise_AVG_Order_Freight_value asc limit 5)
union all
(select fjt.customer_state, round(avg(fjt.total_order_freight_value),2) as
State_wise_AVG_Order_Freight_value , "top5Highest" as is_highest_or_lowest from
final_join_table fjt group by fjt.customer_state
order by State_wise_AVG_Order_Freight_value desc limit 5)
order by State_wise_AVG_Order_Freight_value desc
```

- Output Screenshot (top 10 rows):

Row	customer_state	State_wise_AVG_Order_Freight_value	is_highest_or_lowest
1	RR	48.59	top5Highest
2	PB	48.35	top5Highest
3	RO	46.22	top5Highest
4	AC	45.52	top5Highest
5	PI	43.04	top5Highest
6	RJ	23.95	top5Lowest
7	DF	23.82	top5Lowest
8	PR	23.58	top5Lowest
9	MG	23.46	top5Lowest
10	SP	17.37	top5Lowest

- Valuable insights: NA.

Q.5 (c) Find out the top 5 states with the highest & lowest average delivery time.
Answer:

- Query:

```
with order_details as (select
o.order_id,o.order_status,o.customer_id,o.order_delivered_customer_date,o.order_purchase_ti
mestamp, datetime_Diff(o.order_delivered_customer_date, o.order_purchase_timestamp,day)  as
delivery_time_in_days
from `Target_sql.orders` o where o.order_delivered_customer_date is not null )
, order_state_join_details as
(
select od.*,c.customer_state from order_details od join `Target_sql.customers`c on
od.customer_id=c.customer_id
)

(select osjt.customer_state, round(avg(osjt.delivery_time_in_days),2) as
State_wise_AVG_Order_DeliveryTime,"top5fastest" as is_fastest_or_slowest from
order_state_join_details osjt group by osjt.customer_state
order by State_wise_AVG_Order_DeliveryTime asc limit 5)

union all

(select osjt.customer_state, round(avg(osjt.delivery_time_in_days),2) as
State_wise_AVG_Order_DeliveryTime,"top5Slowest" as is_fastest_or_slowest from
order_state_join_details osjt group by osjt.customer_state
order by State_wise_AVG_Order_DeliveryTime desc limit 5)

order by State_wise_AVG_Order_DeliveryTime asc
```

- Output Screenshot (top 10 rows):

Row	customer_state	State_wise_AVG_Order_DeliveryTime	is_fastest_or_slowest
1	SP	8.3	top5fastest
2	PR	11.53	top5fastest
3	MG	11.54	top5fastest
4	DF	12.51	top5fastest
5	SC	14.48	top5fastest
6	PA	23.32	top5Slowest
7	AL	24.04	top5Slowest
8	AM	25.99	top5Slowest
9	AP	26.73	top5Slowest
10	RR	28.98	top5Slowest

- Valuable insights: NA.

Q.5 (d) Find out the top 5 states where the order delivery is really fast as compared to the estimated date of delivery.

Answer:

- Query:

```
with order_delivery_time_details as
(select
o.order_id,o.order_status,o.customer_id,o.order_delivered_customer_date,o.order_purchase_ti
mestamp, datetime_Diff(o.order_delivered_customer_date, o.order_purchase_timestamp,day)  as
delivery_time_in_days, o.order_estimated_delivery_date,

(datetime_Diff(o.order_estimated_delivery_date, o.order_purchase_timestamp,day)) as
estimated_delivery_in_Days

from `Target_sql.orders` o where o.order_delivered_customer_date is not null ),

order_state_join_details as
(
select odtd.*,c.customer_state from order_delivery_time_details  odtd join
`Target_sql.customers`c on odtd.customer_id=c.customer_id
)

select osjt.customer_state, round(avg(osjt.delivery_time_in_days),2) as
AVG_Order_DeliveryTime_State_wise,
round(avg(osjt.estimated_delivery_in_Days),2) as
AVG_Order_Estimated_DeliveryTime_State_wise,
 round((round(avg(osjt.estimated_delivery_in_Days),2)-
round(avg(osjt.delivery_time_in_days),2)),2) as Diff_DeliveryTime_State_wise

 from order_state_join_details osjt group by osjt.customer_state
 order by Diff_DeliveryTime_State_wise desc limit 5
```

- Output Screenshot (top 10 rows):

Row	customer_state	AVG_Order_DeliveryTime	AVG_Order_Estimated_DeliveryTime	Diff_DeliveryTime
1	AC	20.64	40.72	20.08
2	RO	18.91	38.39	19.48
3	AP	26.73	45.87	19.14
4	AM	25.99	44.92	18.93
5	RR	28.98	45.63	16.65

- Valuable insights: Fastest Delivery than Estimated time wise, Top 5 States are AC,RO,AP,AM and RR.

6. **Analysis based on the payments:**
   a) Find the month on month no. of orders placed using different payment types.
   b) Find the no. of orders placed on the basis of the payment installments that have been paid.

Answer:

Q.6 (a) Find the month on month no. of orders placed using different payment types.

Answer:

- Query:

```
with final_join_table as
(select o.order_id,o.order_purchase_timestamp,Extract (month from
o.order_purchase_timestamp) as Ordered_month,
Extract (Year from o.order_purchase_timestamp) as Ordered_Year,p.payment_type from
`Target_sql.orders` o join `Target_sql.payments` p on o.order_id=p.order_id )

select fjt.Ordered_Year,fjt.Ordered_month,fjt.payment_type, count(distinct order_id) as
Order_count from final_join_table fjt group by
fjt.payment_type,fjt.Ordered_Year,fjt.Ordered_month
order by fjt.Ordered_Year,fjt.Ordered_month,fjt.payment_type limit 10
```

- Output Screenshot (top 10 rows):

Row	Ordered_Year	Ordered_month	payment_type	Order_count
1	2016	9	credit_card	3
2	2016	10	UPI	63
3	2016	10	credit_card	253
4	2016	10	debit_card	2
5	2016	10	voucher	11
6	2016	12	credit_card	1
7	2017	1	UPI	197
8	2017	1	credit_card	582
9	2017	1	debit_card	9
10	2017	1	voucher	33

- Valuable insights: NA.

Q.6 (b) Find the no. of orders placed on the basis of the payment installments that have been paid.

Answer:

- Query:

```
select count(distinct order_id) as Order_count_Installment_Payment from (select
order_id,count(*) as count_order from `Target_sql.payments`  group by order_id ) t where
count_order > 1
```

- Output Screenshot (top 10 rows):

Row	Order_count_I
1	2961

- Valuable insights: 2961 orders are paid on Installments mode.

Gain valuable insights into Target's operations in Brazil :

- To Increase sales Revenue :-
    1. Increase NEW Customer base Count: As most of customers are from the States - SP, RJ and MG, so considering population count, income and their spending pattern of state, we need to do marketing activity to attract and onboard new customers.
    2. AVG Order Value: Considering Highest avg order value i.e. 216 and whatever the avg order value benchmark of profitability, we need to increase the avg order value for all states by looking at their customers more needs, pricing offers and Financing EMI options.
    3. Increase Sales on Season Months: Considering Season months are January,March,July,August and November in the year and most preferable buying time are Afternoon,Night and Morning , So Target needs to come up with great selling proposition(which can be like more items listed on platform, pricing offers , Financing EMI options) for these months.
    4. Increase Awareness of EMI Option: As EMI option made purchase very manageable, Target needs to market this Financing option to their customers, so that they will use it and buy more products.
    5. Delivery Time : Make sure delivery time can be reduced to max. 7 days. Do more collaboration for Delivery.
- To Decrease Expenses :-
    1. Decrease Freight value: Do more collaboration for Delivery, increase sales/orders from same pincode area, Onboard more Nearest Seller based on ordered products.