# Target

From company’s perspective:

● 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.

**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 **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 **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 **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 **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 **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 **reviews.csv** contain following features:

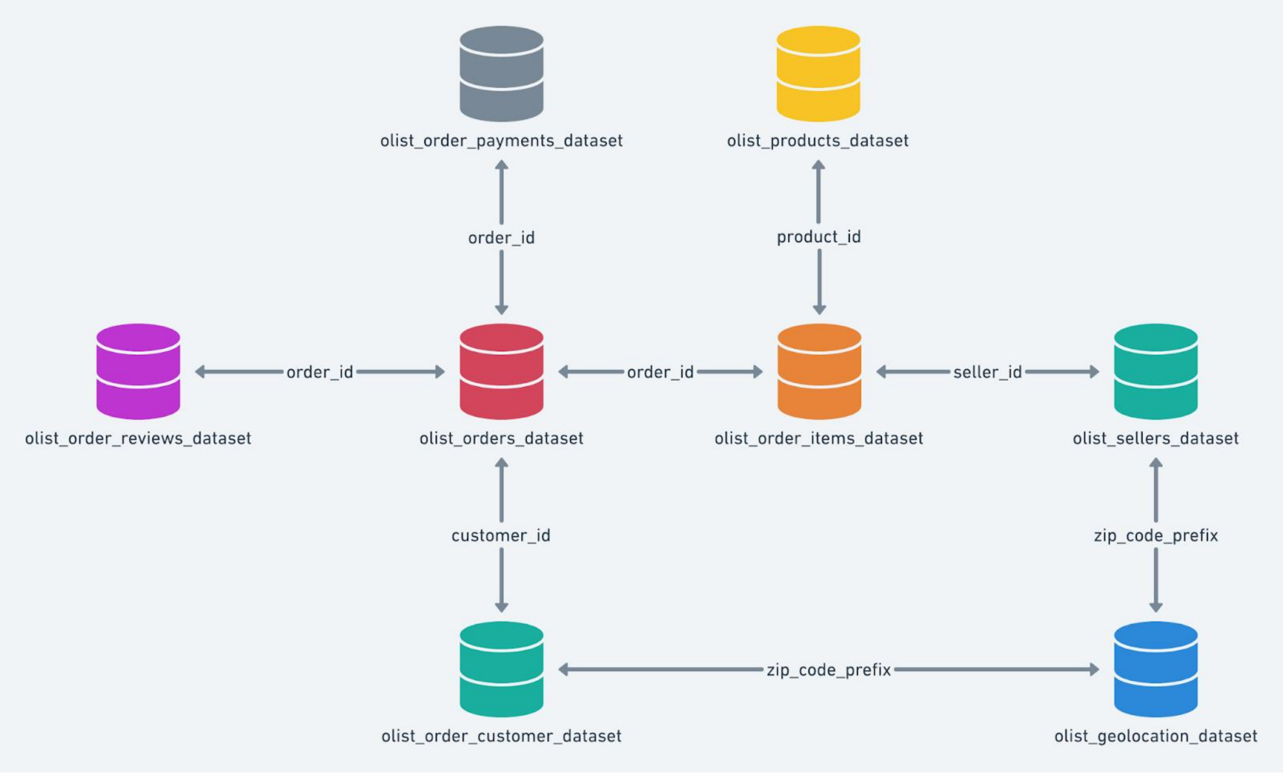
| **Features** | **Description** |
|---|---|
| review_id | ID of the review given on the product ordered by the order id |
| order_id | A Unique ID of order made by the consumers |
| review_score | Review score given by the customer for each order on a scale of 1-5 |
| review_comment_title | Title of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



## 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.

```
SELECT
  column_name,
  data_type
FROM
  `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**:- There are 5 columns in the customer table and what data type we can enter in the table is known to us.

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

```
select
  min(order_purchase_timestamp) as first_order_timestamp,
  max(order_purchase_timestamp) as last_order_timestamp,
from `Target.orders`;
```

| Row | first_order_timestamp | last_order_timestamp |
|---|---|---|
| 1 | 2016-09-04 21:15:19 UTC | 2018-10-17 17:30:18 UTC |

**Insights**:- The first order was placed on 2016 – 09 – 04 and the last order was placed on 2018 – 10 – 17.

### C. Count the number of Cities and States in our dataset.

```
select
  count(distinct(customer_city)) as num_city,
  count(distinct(customer_state)) as num_state
from `Target.customers`;
```

| Row | num_city | num_state |
|---|---|---|
| 1 | 4119 | 27 |

**Insights**:- We received orders from all the states in Brazil (i.e. 26 states (estados) and one federal district (distrito federal)) and received orders from around 74% of the total cities (i.e.  5,570 municipalities (source google)) in Brazil.

## II. In-depth Exploration:

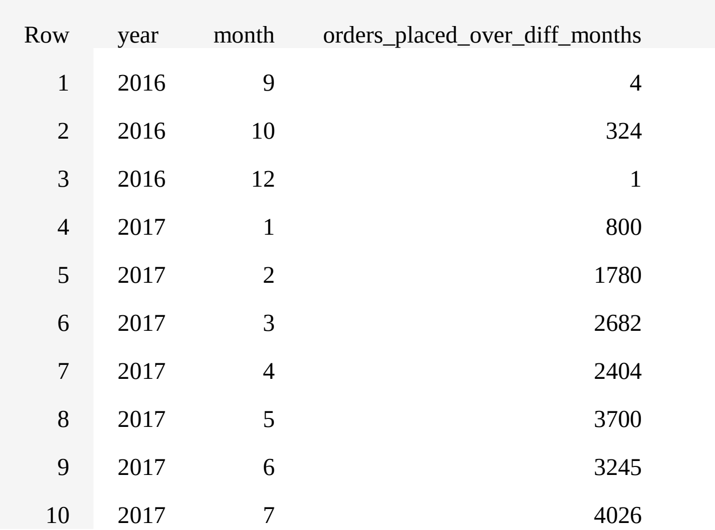
**Note**:- Solved **II** part by creating the view month_year_time.

```
create view `Target.month_year_time` as (
  select order_purchase_timestamp, extract(year from order_purchase_timestamp) as year,
  extract(month from order_purchase_timestamp) as month,
  extract(time from order_purchase_timestamp) as time
from `Target.orders`);
```

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

**Assumption**:- The data depicts the monthly orders placed across different years.

```
select year,month, count(*) as orders_placed_over_diff_months
from `Target.month_year_time`
group by year, month
order by year asc, month asc;
```



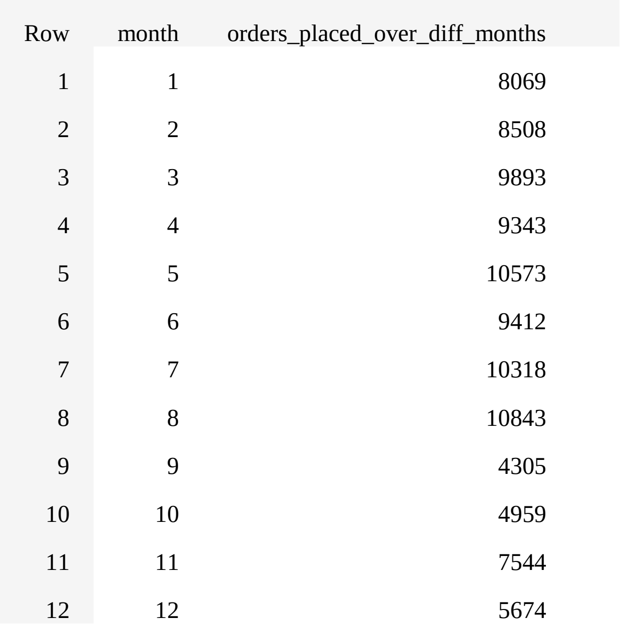
| Row | year | month | orders_placed_over_diff_months |
|---|---|---|---|
| 1 | 2016 | 9 | 4 |
| 2 | 2016 | 10 | 324 |
| 3 | 2016 | 12 | 1 |
| 4 | 2017 | 1 | 800 |
| 5 | 2017 | 2 | 1780 |
| 6 | 2017 | 3 | 2682 |
| 7 | 2017 | 4 | 2404 |
| 8 | 2017 | 5 | 3700 |
| 9 | 2017 | 6 | 3245 |
| 10 | 2017 | 7 | 4026 |

**Insights**:- The trend from the data is pretty clear that the number of orders that are placed are mostly increasing every month. But it is surprising to see that there are no orders in November 2016 (i.e. 2016 – 11) also number of orders decreased significantly in the months- December 2016 (2016 – 12), September 2018 (2018 – 9) and October 2018 (2018 – 10).

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

**Assumption**:- The data represents orders placed for each month, regardless of the year.

```
select month, count(*) as orders_placed_over_diff_months
from `Target.month_year_time`
group by  month
order by month asc;
```



| Row | month | orders_placed_over_diff_months |
|---|---|---|
| 1 | 1 | 8069 |
| 2 | 2 | 8508 |
| 3 | 3 | 9893 |
| 4 | 4 | 9343 |
| 5 | 5 | 10573 |
| 6 | 6 | 9412 |
| 7 | 7 | 10318 |
| 8 | 8 | 10843 |
| 9 | 9 | 4305 |
| 10 | 10 | 4959 |
| 11 | 11 | 7544 |
| 12 | 12 | 5674 |

**Insight**:- During 5th, 7th and 8th months (i.e. May, July and August) most of the orders are placed.

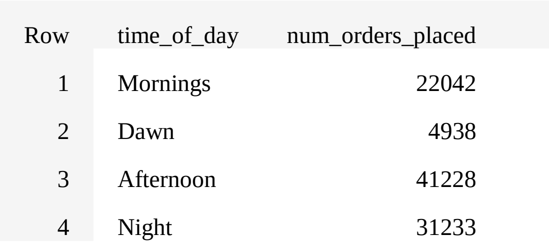
**C. During what time of the day, do the Brazilian customers mostly place their orders? (Dawn, Morning, Afternoon or Night)**

- 0-6 hrs : Dawn
- 7-12 hrs : Mornings
- 13-18 hrs : Afternoon
- 19-23 hrs : Night

**Assumption**:- Just to make time continuous, I have considered

- 0-6:30 hrs : Dawn
- 6:30-12 hrs : Mornings
- 12-18:30 hrs : Afternoon
- 18:30-24 hrs : Night

```
with cte as (select time,
case
when time between '00:00:00' and '06:30:00' then 'Dawn'
when time between '06:30:01' and '12:00:00' then 'Mornings'
when time between '12:00:01' and '18:30:00' then 'Afternoon'
else 'Night'
end as time_of_day
from `Target.month_year_time`)

select time_of_day, count(*) as num_orders_placed
from cte
group by time_of_day
```



| Row | time_of_day | num_orders_placed |
|---|---|---|
| 1 | Mornings | 22042 |
| 2 | Dawn | 4938 |
| 3 | Afternoon | 41228 |
| 4 | Night | 31233 |

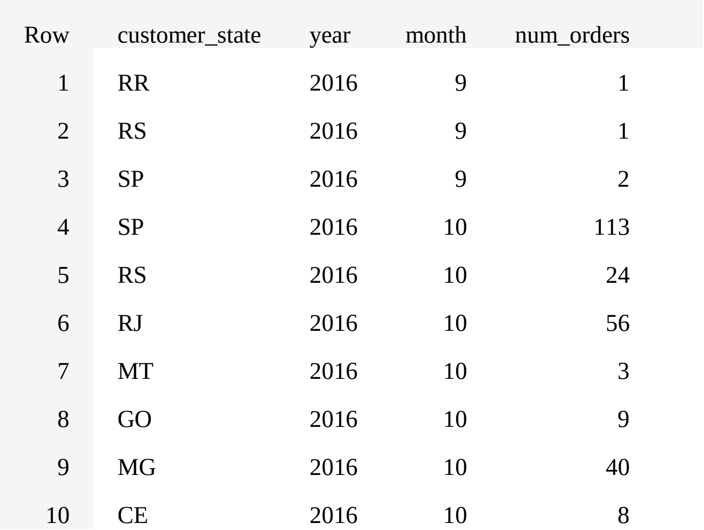
**Insight**:- Most of the orders are placed during Afternoon and least number of orders are placed during Dawn.

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

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

**Assumption**:- The data depicts the monthly orders placed across different years in different states.

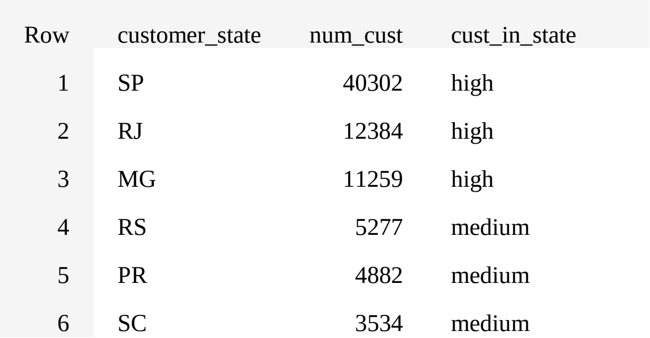
```
with cte as (
  select *,
  extract(year from order_purchase_timestamp) as year,
  extract(month from order_purchase_timestamp) as month
  from `Target.orders` o
  join `Target.customers` c
  on o.customer_id = c.customer_id)

select customer_state, year, month, count (*) as num_orders
from cte
group by customer_state, year,month
order by year asc, month asc
```



| Row | customer_state | year | month | num_orders |
|---|---|---|---|---|
| 1 | RR | 2016 | 9 | 1 |
| 2 | RS | 2016 | 9 | 1 |
| 3 | SP | 2016 | 9 | 2 |
| 4 | SP | 2016 | 10 | 113 |
| 5 | RS | 2016 | 10 | 24 |
| 6 | RJ | 2016 | 10 | 56 |
| 7 | MT | 2016 | 10 | 3 |
| 8 | GO | 2016 | 10 | 9 |
| 9 | MG | 2016 | 10 | 40 |
| 10 | CE | 2016 | 10 | 8 |

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

```
select
customer_state,
count(distinct(customer_unique_id)) as num_cust,
case
    when count(distinct(customer_unique_id)) > 10000 then 'high'
    when count(distinct(customer_unique_id)) between 1000 and 10000 then 'medium'
    else 'low'
end as cust_in_state
from `Target.customers`
group by customer_state
order by num_cust desc
```



| Row | customer_state | num_cust | cust_in_state |
|---|---|---|---|
| 1 | SP | 40302 | high |
| 2 | RJ | 12384 | high |
| 3 | MG | 11259 | high |
| 4 | RS | 5277 | medium |
| 5 | PR | 4882 | medium |
| 6 | SC | 3534 | medium |

| Row | customer_state | num_cust | cust_in_state |
|---|---|---|---|
| 7 | BA | 3277 | medium |
| 8 | DF | 2075 | medium |
| 9 | ES | 1964 | medium |
| 10 | GO | 1952 | medium |

**Insights**:- The data clearly shows that the states SP, RJ, and MG have the highest number of customers. To test the product's market potential, we could consider initiating experiments in these states initially.

## 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).*

**Assumption**:- Considered total payment value over the years for the required months.

```
with cte as (
  select *, extract(year from o.order_purchase_timestamp) year,
  extract(month from o.order_purchase_timestamp) as month
from `Target.payments` p
join `Target.orders` o
  on p.order_id = o.order_id
where
  extract(year from o.order_purchase_timestamp) in (2017, 2018) and
  extract(month from o.order_purchase_timestamp) in (1, 2, 3, 4, 5, 6, 7, 8))

select year, total, next_year_total, (next_year_total - total)*100/total as percentage_change
from (
  select *, lead(total) over(order by total) as next_year_total,
  from (
    select year, sum(payment_value) as total,
    from cte
    group by year)) tbl2
order by year
```

| Row | year | total | next_year_total | percentage_change |
|---|---|---|---|---|
| 1 | 2017 | 3669022.1200000118 | 8694733.83999979 | 136.97687164665447 |
| 2 | 2018 | 8694733.83999979 | *null* | *null* |

**Insights**:- The total payment value witnessed a substantial growth of approximately 137% between 2017 and 2018.

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

```
select
customer_state, round(sum(oi.price),2)as total_value,count(*) no_of_orders, round(avg(oi.price),2) as average_value
from `Target.orders` o
join `Target.customers` c
on o.customer_id = c.customer_id
join `Target.order_items` oi
on o.order_id = oi.order_id
group by c.customer_state
order by no_of_orders desc
```

| Row | customer_state | total_value | no_of_orders | average_value |
|---|---|---|---|---|
| 1 | SP | 5202955.05 | 47449 | 109.65 |
| 2 | RJ | 1824092.67 | 14579 | 125.12 |
| 3 | MG | 1585308.03 | 13129 | 120.75 |
| 4 | RS | 750304.02 | 6235 | 120.34 |
| 5 | PR | 683083.76 | 5740 | 119.0 |
| 6 | SC | 520553.34 | 4176 | 124.65 |
| 7 | BA | 511349.99 | 3799 | 134.6 |
| 8 | DF | 302603.94 | 2406 | 125.77 |
| 9 | GO | 294591.95 | 2333 | 126.27 |
| 10 | ES | 275037.31 | 2256 | 121.91 |

**Insights**:- We can emulate the successful strategies from SP, RJ, and MG to boost order numbers in other states. Additionally, focusing on MG, where the order count is lower than RJ but the average value is less than MG, could offer valuable insights.

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

```
select
customer_state, round(sum(oi.freight_value),2)as total_value,count(*) no_of_orders, round(avg(oi.freight_value),2) as average_value
from `Target.orders` o
join `Target.customers` c
on o.customer_id = c.customer_id
join `Target.order_items` oi
on o.order_id = oi.order_id
group by c.customer_state
order by average_value desc
```

| Row | customer_state | total_value | no_of_orders | average_value |
|---|---|---|---|---|
| 1 | RR | 2235.19 | 52 | 42.98 |
| 2 | PB | 25719.73 | 602 | 42.72 |
| 3 | RO | 11417.38 | 278 | 41.07 |
| 4 | AC | 3686.75 | 92 | 40.07 |
| 5 | PI | 21218.2 | 542 | 39.15 |
| 6 | MA | 31523.77 | 824 | 38.26 |
| 7 | TO | 11732.68 | 315 | 37.25 |
| 8 | SE | 14111.47 | 385 | 36.65 |
| 9 | AL | 15914.59 | 444 | 35.84 |
| 10 | PA | 38699.3 | 1080 | 35.83 |

**Insights**:- Reasons for higher average value of freight can be explored in the states having high average value. Geographical factors, market opportunities and other things can be explored in these states.

### 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.

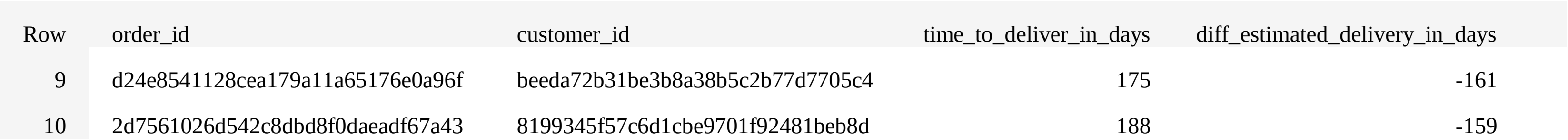
Calculate the delivery time and the difference between the estimated & actual delivery date using the given formula:

- **time_to_deliver** = order_delivered_customer_date - order_purchase_timestamp
- **diff_estimated_delivery** = order_estimated_delivery_date - order_delivered_customer_date

```
SELECT
    order_id,
    customer_id,
    #order_purchase_timestamp,
    #order_delivered_customer_date,
    #order_estimated_delivery_date,
    DATE_DIFF(order_delivered_customer_date, order_purchase_timestamp, DAY) AS time_to_deliver_in_days,
    DATE_DIFF(order_estimated_delivery_date, order_delivered_customer_date, DAY) AS diff_estimated_delivery_in_days
FROM
    `Target.orders`
WHERE
    order_status = 'delivered'
    AND order_delivered_customer_date IS NOT NULL
ORDER BY
    diff_estimated_delivery_in_days ASC
```

**Note**:- Didn't display order_purchase_timestamp, order_delivered_customer_date and order_estimated_delivery_date.

| Row | order_id | customer_id | time_to_deliver_in_days | diff_estimated_delivery_in_days |
|---|---|---|---|---|
| 1 | 1b3190b2dfa9d789e1f14c05b647a14a | d306426abe5fca15e54b645e4462dc7b | 208 | -188 |
| 2 | ca07593549f1816d26a572e06dc1eab6 | 75683a92331068e2d281b11a7866ba44 | 209 | -181 |
| 3 | 47b40429ed8cce3aee9199792275433f | cb2caaaead400c97350c37a3fc536867 | 191 | -175 |
| 4 | 2fe324febf907e3ea3f2aa9650869fa5 | 65b14237885b3972ebec28c0f7dd2220 | 189 | -167 |
| 5 | 285ab9426d6982034523a855f55a885e | 9cf2c3fa2632cee748e1a59ca9d09b21 | 194 | -166 |
| 6 | 440d0d17af552815d15a9e41abe49359 | 7815125148cfa1e8c7fee1ff7974f16c | 195 | -165 |
| 7 | c27815f7e3dd0b926b58552628481575 | f85e9ec0719b16dc4dd0edd438793553 | 187 | -162 |
| 8 | 0f4519c5f1c541ddec9f21b3bddd533a | 1a8a4a30dc296976717f44e7801fdeef | 194 | -161 |



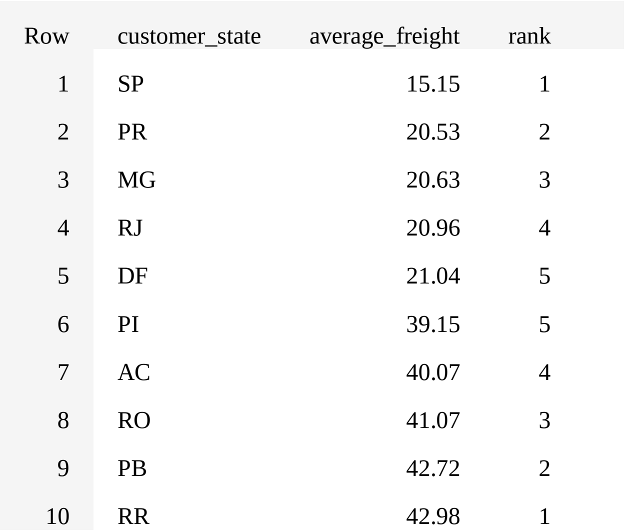
| Row | order_id | customer_id | time_to_deliver_in_days | diff_estimated_delivery_in_days |
|---|---|---|---|---|
| 9 | d24e8541128cea179a11a65176e0a96f | beeda72b31be3b8a38b5c2b77d7705c4 | 175 | -161 |
| 10 | 2d7561026d542c8dbd8f0daeadf67a43 | 8199345f57c6d1cbe9701f92481beb8d | 188 | -159 |

**Insights**:- Here positive Value (diff_estimated_delivery_in_days > 0) indicates that the order was delivered later than the estimated delivery date. While Negative Value (diff_estimated_delivery_in_days < 0) indicates that the order was delivered earlier than the estimated delivery date. To delve deeper into the analysis, it could be valuable to concentrate on orders that have experienced significant delays beyond the estimated delivery date. Addressing these instances of delay can contribute to enhancing the overall customer experience.

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

**Assumption**:- Shown average freight of the top 5 & the bottom 5 states arranged in increasing order of the average freight having top 5 and bottom 5 values (i.e rank) in the same column.

```
with cte as (
    select
        c.customer_state,
        round(avg(oi.freight_value), 2) as average_freight,
        dense_rank() over (order BY avg(oi.freight_value) desc) as rnk_high,
        dense_rank() over (order BY avg(oi.freight_value) asC) as rnk_low
    from `Target.orders` o
    join `Target.customers` c on o.customer_id = c.customer_id
    join `Target.order_items` oi on o.order_id = oi.order_id
    group by c.customer_state)
select
    customer_state,
    average_freight,
    case
        when rnk_low between 1 and 5 then rnk_low
        else rnk_high
    end as rank
from cte
where rnk_low between 1 and 5 or rnk_high between 1 and 5
order by average_freight;
```

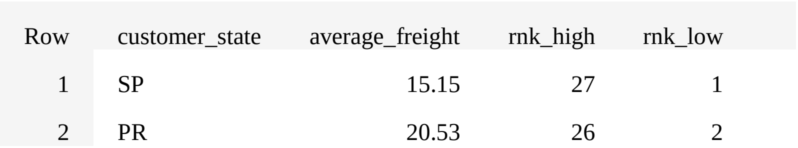


| Row | customer_state | average_freight | rank |
|---|---|---|---|
| 1 | SP | 15.15 | 1 |
| 2 | PR | 20.53 | 2 |
| 3 | MG | 20.63 | 3 |
| 4 | RJ | 20.96 | 4 |
| 5 | DF | 21.04 | 5 |
| 6 | PI | 39.15 | 5 |
| 7 | AC | 40.07 | 4 |
| 8 | RO | 41.07 | 3 |
| 9 | PB | 42.72 | 2 |
| 10 | RR | 42.98 | 1 |

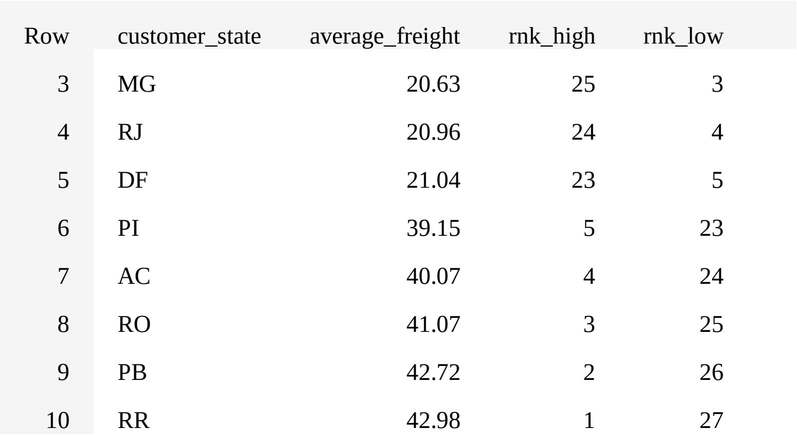
**Alternate Solution**:- Shown average freight of the top 5 & the bottom 5 states arranged in increasing order of the average freight having top 5 and bottom 5 values (i.e rank) in different columns.

**Note**:- Can also represent the same where both top 5 and bottom 5 states can be arranged together like done in the next part (i.e V(C)).

```
with cte as
(select c.customer_state, round(avg(oi.freight_value),2) as average_freight,
dense_rank() over(order by avg(oi.freight_value) desc) as rnk_high,
dense_rank() over(order by avg(oi.freight_value) asc) as rnk_low
from `Target.orders` o
join `Target.customers` c
on o.customer_id = c.customer_id
join `Target.order_items` oi
on o.order_id = oi.order_id
group by c.customer_state)

select customer_state, average_freight, rnk_high, rnk_low
from cte
where rnk_high in (1, 2, 3, 4, 5) or rnk_low in (1, 2, 3, 4, 5)
order by average_freight
```



| Row | customer_state | average_freight | rnk_high | rnk_low |
|---|---|---|---|---|
| 1 | SP | 15.15 | 27 | 1 |
| 2 | PR | 20.53 | 26 | 2 |



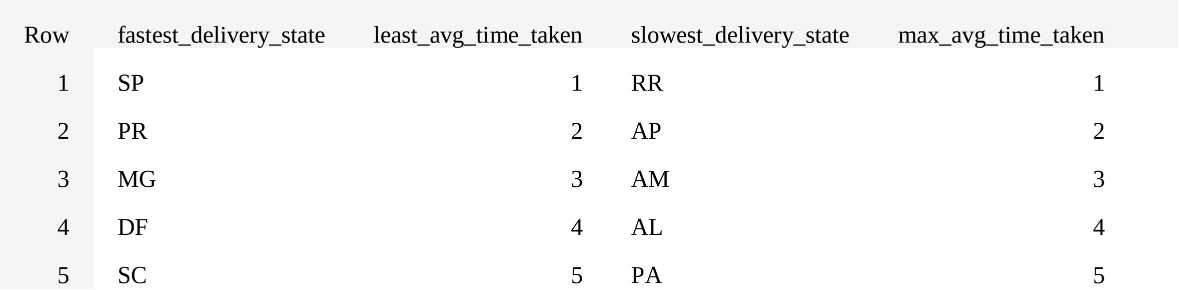
| Row | customer_state | average_freight | rnk_high | rnk_low |
|---|---|---|---|---|
| 3 | MG | 20.63 | 25 | 3 |
| 4 | RJ | 20.96 | 24 | 4 |
| 5 | DF | 21.04 | 23 | 5 |
| 6 | PI | 39.15 | 5 | 23 |
| 7 | AC | 40.07 | 4 | 24 |
| 8 | RO | 41.07 | 3 | 25 |
| 9 | PB | 42.72 | 2 | 26 |
| 10 | RR | 42.98 | 1 | 27 |

**Insight**:- Top 5 states with the highest freight value are RR, PB, RO, AC and PI & top 5 states with lowest average freight value are SP, PR, MG, RJ and DF.

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

**Assumption**:- Displayed top 5 & the bottom 5 states arranged in increasing order of the average delivery time for states with faster delivery and decreasing order of the average delivery time for states with slower delivery.

```
with cte as (
select customer_state, avg(time_to_deliver_in_days) as avg_time_to_deliver_in_days
from
(select
            order_id,
            customer_id,
            order_purchase_timestamp,
            order_delivered_customer_date, order_estimated_delivery_date,
            date_diff(order_delivered_customer_date, order_purchase_timestamp, day) as time_to_deliver_in_days,
        from `Target.orders`
        where order_status = 'delivered' and order_delivered_customer_date is not null) tbl
join `Target.customers` c
on tbl.customer_id = c.customer_id
group by c.customer_state)

select *
from(
select customer_state as fastest_delivery_state,
row_number() over(order by cte.avg_time_to_deliver_in_days) as least_avg_time_taken
from cte) l
join(
select customer_state as slowest_delivery_state,
row_number() over(order by cte.avg_time_to_deliver_in_days desc) as max_avg_time_taken
from cte)m
on l.least_avg_time_taken = m.max_avg_time_taken
limit 5
```



| Row | fastest_delivery_state | least_avg_time_taken | slowest_delivery_state | max_avg_time_taken |
|---|---|---|---|---|
| 1 | SP | 1 | RR | 1 |
| 2 | PR | 2 | AP | 2 |
| 3 | MG | 3 | AM | 3 |
| 4 | DF | 4 | AL | 4 |
| 5 | SC | 5 | PA | 5 |

**Insight**:- We can attract more customers in the states where average delivery time is more and can adapt best practices that are being followed by the top 5 states with least 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.**

```
with cte as(
select order_id, customer_id, order_purchase_timestamp, order_delivered_customer_date, order_estimated_delivery_date,
date_diff(order_estimated_delivery_date, order_delivered_customer_date, day) as diff_estimated_delivery_in_days
from `Target.orders`
where order_status = 'delivered' and order_delivered_customer_date is not null)

select *
from
(select c.customer_state, round(avg(diff_estimated_delivery_in_days),2) avg_time_diff,
dense_rank() over(order by avg(diff_estimated_delivery_in_days)) as rnk
from cte
join `Target.customers` c
on cte.customer_id = c.customer_id
group by c.customer_state) tbl
```

```
where rnk between 1 and 5
order by rnk asc
```

| Row | customer_state | avg_time_diff | rnk |
|---|---|---|---|
| 1 | AL | 7.95 | 1 |
| 2 | MA | 8.77 | 2 |
| 3 | SE | 9.17 | 3 |
| 4 | ES | 9.62 | 4 |
| 5 | BA | 9.93 | 5 |

**Insight**:- We can study these five states and implement the best practices that they are using to decrease average time to deliver.

## VI. Analysis based on the payments:

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

**Assumption**:- The data represents orders placed for each month, regardless of the year using different payment types.

```
select month, payment_type, count(*) as orders_placed
from (
  select *,EXTRACT(year FROM o.order_purchase_timestamp ) as year,
  EXTRACT(month FROM o.order_purchase_timestamp ) as month
  from `Target.orders` o
  join `Target.payments` p
  on o.order_id = p.order_id
  join `Target.customers` c
  on o.customer_id = c.customer_id) tbl
group by month, payment_type
order by month, orders_placed desc
```

| Row | month | payment_type | orders_placed |
|---|---|---|---|
| 1 | 1 | credit_card | 6103 |
| 2 | 1 | UPI | 1715 |
| 3 | 1 | voucher | 477 |
| 4 | 1 | debit_card | 118 |
| 5 | 2 | credit_card | 6609 |
| 6 | 2 | UPI | 1723 |
| 7 | 2 | voucher | 424 |
| 8 | 2 | debit_card | 82 |
| 9 | 3 | credit_card | 7707 |
| 10 | 3 | UPI | 1942 |

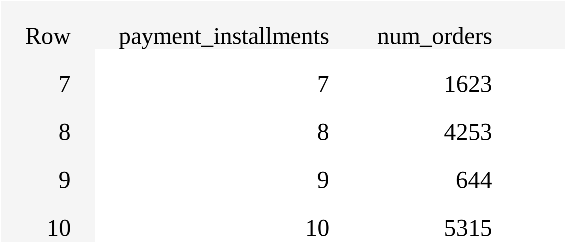
**Insight**:- Credit cards are the go-to payment method for most customers.

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

**Assumption**:- Considering payment_sequential greater than 0 and payment_installments greater than 0 and payment_value greater than 0 will give us the orders where at least one installment has been successfully paid. Also considering that we have to find orders placed based on the no. of payment_installments.

```
select payment_installments,
count(distinct order_id) as num_orders
from `Target.payments`
where payment_sequential > 0 and payment_installments > 0 and payment_value > 0
group by payment_installments
```

| Row | payment_installments | num_orders |
|---|---|---|
| 1 | 1 | 49057 |
| 2 | 2 | 12389 |
| 3 | 3 | 10443 |
| 4 | 4 | 7088 |
| 5 | 5 | 5234 |
| 6 | 6 | 3916 |



| Row | payment_installments | num_orders |
|---|---|---|
| 7 | 7 | 1623 |
| 8 | 8 | 4253 |
| 9 | 9 | 644 |
| 10 | 10 | 5315 |

**Insight**:- The majority of orders consist of a single payment installment.