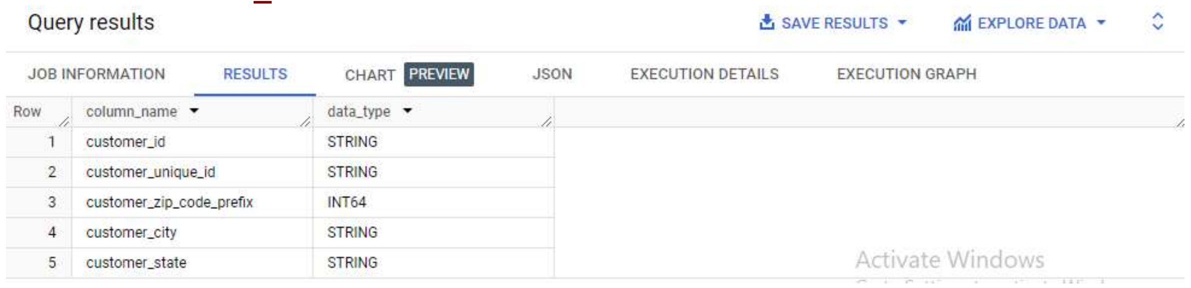
# BUSINESS CASE – TARGET SQL

**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.

```
ANS.SELECT column_name,data_type
    FROM target_sql.INFORMATION_SCHEMA.COLUMNS
    WHERE table_name='customers'
```

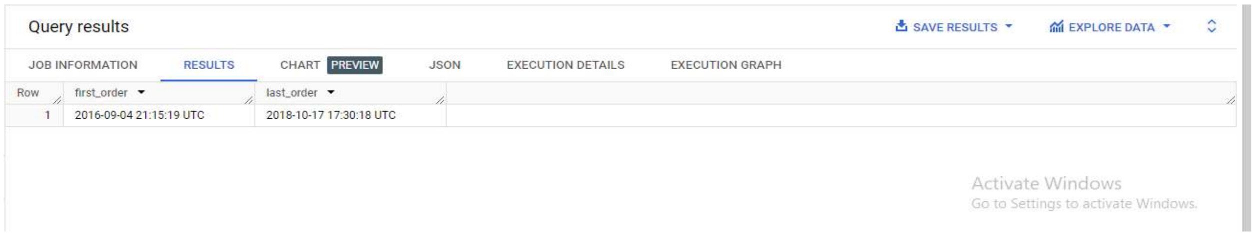

Query results

| 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 |

**<u>Insights</u>**: Here we have most of the column_name has data_type as string except customer_zip_code_prefix dataset as INT64

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

```
ANS: SELECT

  MIN(order_purchase_timestamp)AS first_order,
  MAX(order_purchase_timestamp)AS last_order
FROM
  `target_sql.orders`
```

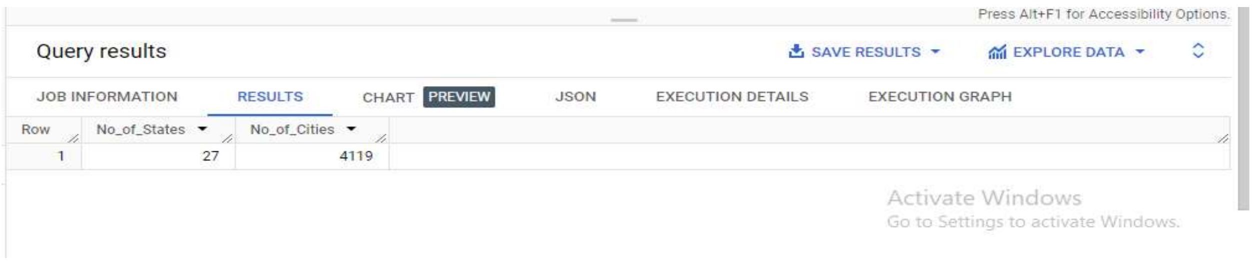

Query results

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

**<u>Insights:</u>** The first order is placed on 2016-09-04 and last order is placed on 2018-10-17 in our dataset.

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

```
ANS: SELECT

  count (DISTINCT(customer_state))AS No_of_States,
  count (DISTINCT(customer_city))AS No_of_Cities
FROM
  `target_sql.customers`
```



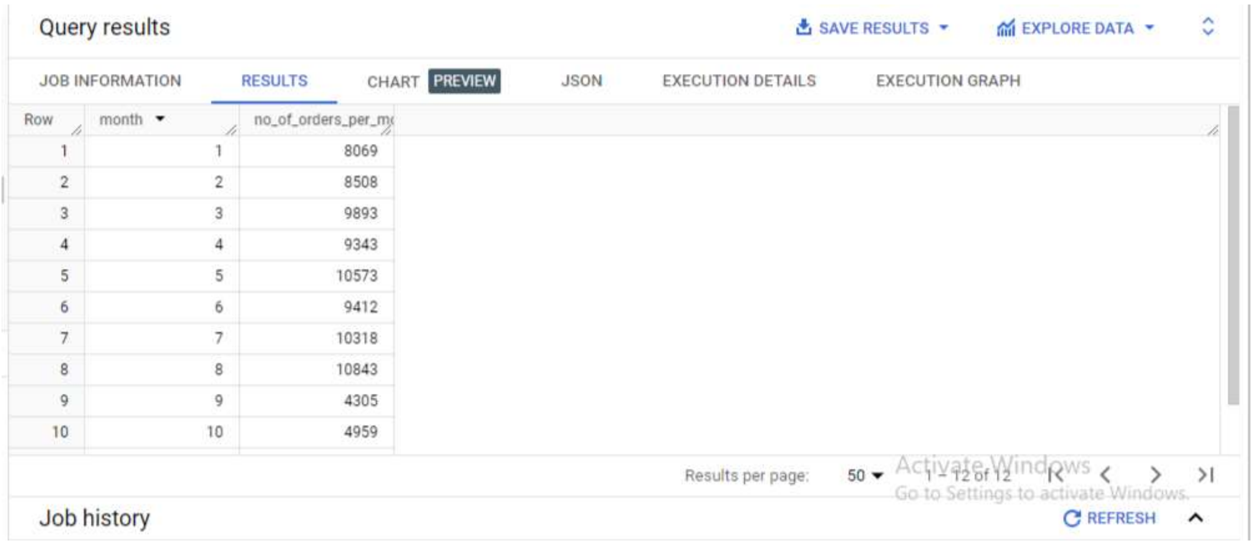
| Row | No_of_States | No_of_Cities |
|---|---|---|
| 1 | 27 | 4119 |

**Insights:** Total number of states where customers have ordered is 27 and total number of cities is 4119.

## 2. In-depth Exploration:

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

ANS:
```
SELECT extract(month from order_purchase_timestamp)AS
month,count(order_id)AS no_of_orders_per_month

FROM `target_sql.orders`
GROUP BY month
ORDER BY 1
```



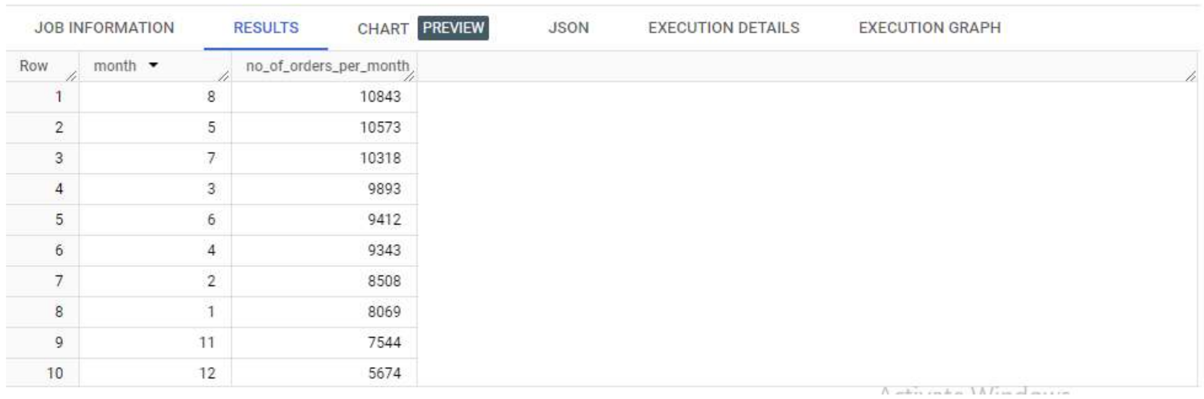
| Row | month | no_of_orders_per_m |
|---|---|---|
| 1 | 1 | 8069 |
| 2 | 2 | 8508 |
| 3 | 3 | 9893 |
| 4 | 4 | 9343 |
| 5 | 5 | 10573 |
| 6 | 6 | 9412 |
| 7 | 7 | 10318 |
| 8 | 8 | 10843 |
| 9 | 9 | 4305 |
| 10 | 10 | 4959 |

**Insights:** Here we have month on month growing of orders placed over last year.

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

ANS:
```
SELECT

  EXTRACT(month
  FROM
    order_purchase_timestamp)AS month,
  COUNT(order_id)AS no_of_orders_per_month
FROM
  `target_sql.orders`
```

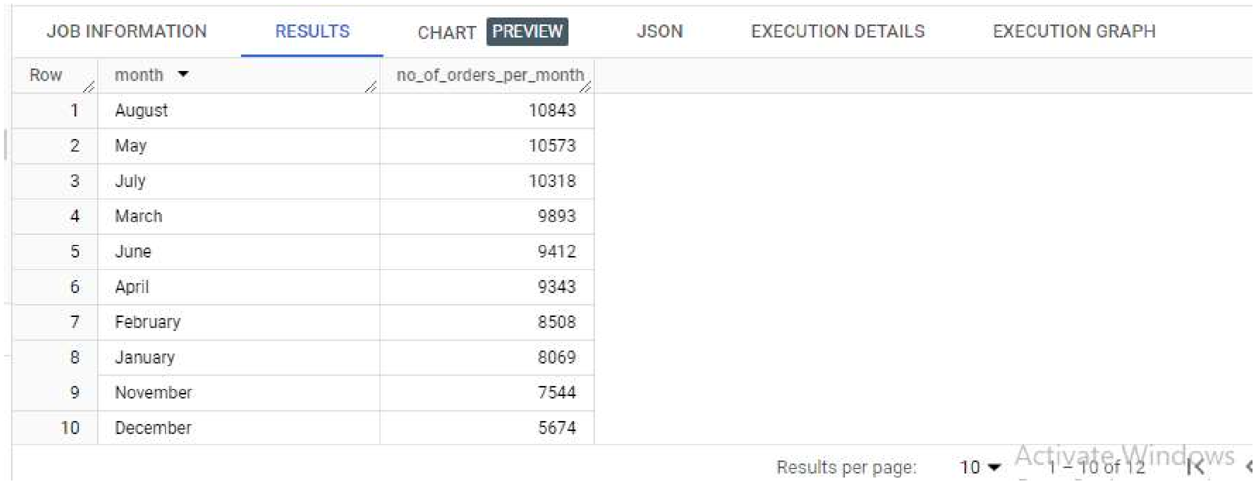
```
GROUP BY
  month
ORDER BY 2 DESC
```



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

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

ANS:
```
SELECT
  FORMAT_DATE('%B',order_purchase_timestamp)AS month,
  COUNT(order_id)AS no_of_orders_per_month
FROM
  `target_sql.orders`
GROUP BY
  month
ORDER BY
  2 desc
```



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

| Row | month | no_of_orders_per_month |
|---|---|---|
| 1 | August | 10843 |
| 2 | May | 10573 |
| 3 | July | 10318 |
| 4 | March | 9893 |
| 5 | June | 9412 |
| 6 | April | 9343 |
| 7 | February | 8508 |
| 8 | January | 8069 |
| 9 | November | 7544 |
| 10 | December | 5674 |

Results per page: 10 1 – 10 of 12

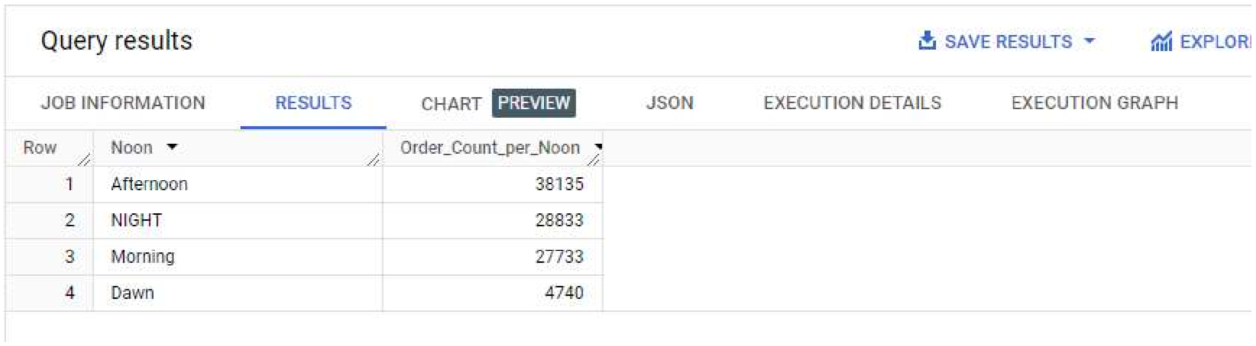
**Insights:** Monthly Seasonality in no. of orders is in months of August, May and July is more.

**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 Time < '06' THEN 'Dawn'`

```
       WHEN Time BETWEEN '07' AND '12' THEN 'Morning'
       WHEN Time BETWEEN '13' AND '18' THEN 'Afternoon'
       ELSE 'NIGHT' end AS Noon,
       SUM(no_of_orders_per_hour)AS Order_Count_per_Noon
FROM
(SELECT
  FORMAT_TIMESTAMP('%H',order_purchase_timestamp)AS Time,
  COUNT(order_id)AS no_of_orders_per_hour
FROM
  `target_sql.orders`
GROUP BY Time
ORDER BY 1)
GROUP BY Noon
ORDER BY 2 desc
```


Query results

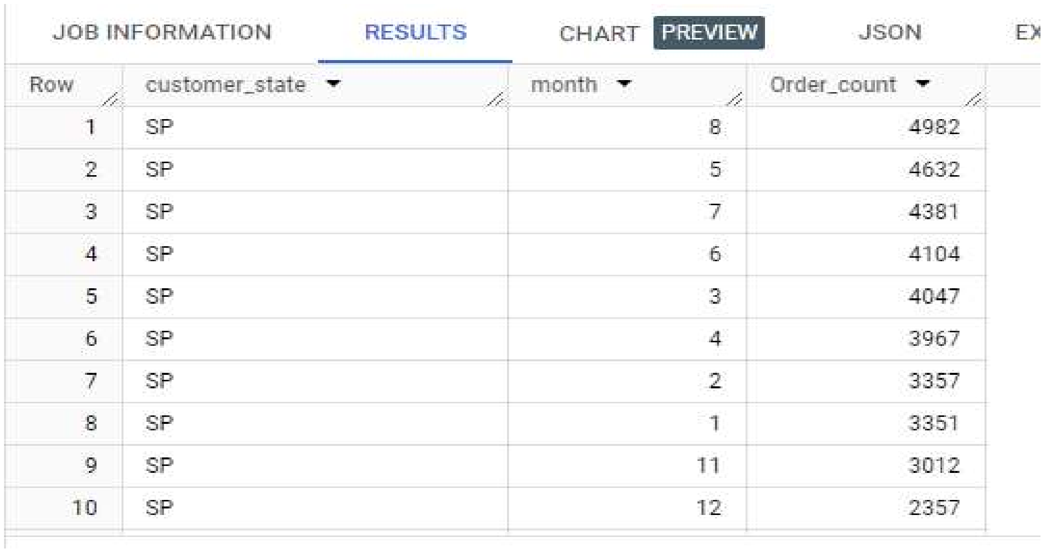
| Row | Noon | Order_Count_per_Noon |
|---|---|---|
| 1 | Afternoon | 38135 |
| 2 | NIGHT | 28833 |
| 3 | Morning | 27733 |
| 4 | Dawn | 4740 |

**Insights:** The Order s are mostly being placed in Afternoon and Less in Dawn.

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

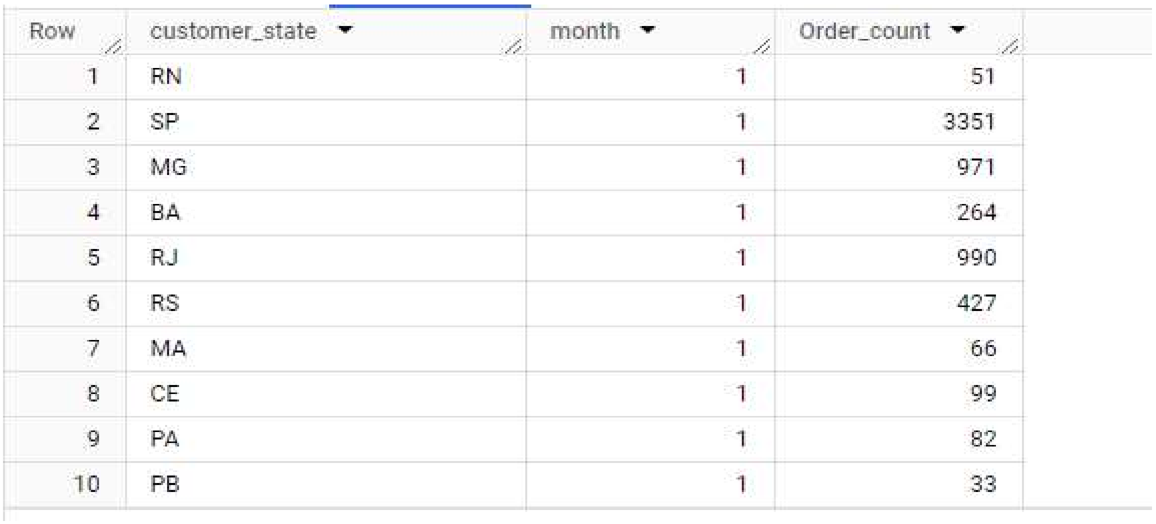
**A.Get the month on month no. of orders placed in each state.**

ANS:
```
SELECT c.customer_state,

       EXTRACT(month FROM o.order_purchase_timestamp)AS month,
       COUNT(o.order_id)AS Order_count
FROM
  `target_sql.customers`c
JOIN `target_sql.orders`o
ON c.customer_id=o.customer_id
GROUP BY 1,2
ORDER BY 3 desc
```


JOB INFORMATION RESULTS CHART PREVIEW JSON EX

| Row | customer_state | month | Order_count |
|---|---|---|---|
| 1 | SP | 8 | 4982 |
| 2 | SP | 5 | 4632 |
| 3 | SP | 7 | 4381 |
| 4 | SP | 6 | 4104 |
| 5 | SP | 3 | 4047 |
| 6 | SP | 4 | 3967 |
| 7 | SP | 2 | 3357 |
| 8 | SP | 1 | 3351 |
| 9 | SP | 11 | 3012 |
| 10 | SP | 12 | 2357 |

ANS 2:
```
SELECT c.customer_state,

       EXTRACT(month FROM o.order_purchase_timestamp)AS month,
       COUNT(o.order_id)AS Order_count
FROM
  `target_sql.customers`c
JOIN `target_sql.orders`o
ON c.customer_id=o.customer_id
GROUP BY 1,2
ORDER BY 2
```



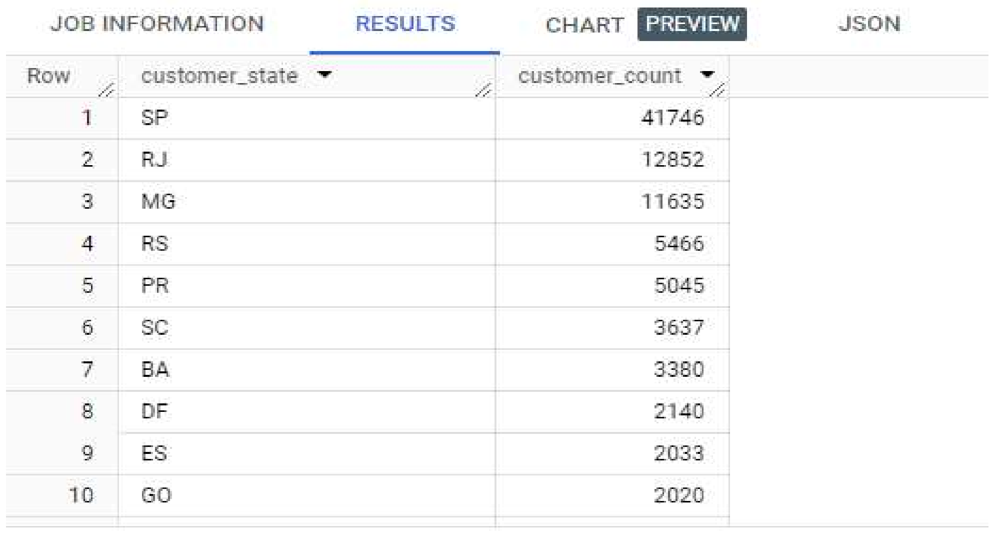
| Row | customer_state | month | Order_count |
|---|---|---|---|
| 1 | RN | 1 | 51 |
| 2 | SP | 1 | 3351 |
| 3 | MG | 1 | 971 |
| 4 | BA | 1 | 264 |
| 5 | RJ | 1 | 990 |
| 6 | RS | 1 | 427 |
| 7 | MA | 1 | 66 |
| 8 | CE | 1 | 99 |
| 9 | PA | 1 | 82 |
| 10 | PB | 1 | 33 |

<u>**Insights:**</u> Here we have given the no. of orders placed by each state on each month.

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

ANS:
```
SELECT distinct(c.customer_state),

       COUNT(c.customer_id)AS customer_count
FROM
```

```
  `target_sql.customers`c
GROUP BY 1
ORDER BY 2 desc
```

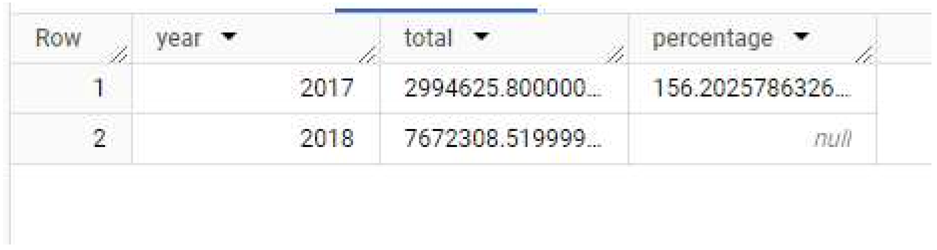

JOB INFORMATION RESULTS CHART PREVIEW JSON

| Row | customer_state | customer_count |
|---|---|---|
| 1 | SP | 41746 |
| 2 | RJ | 12852 |
| 3 | MG | 11635 |
| 4 | RS | 5466 |
| 5 | PR | 5045 |
| 6 | SC | 3637 |
| 7 | BA | 3380 |
| 8 | DF | 2140 |
| 9 | ES | 2033 |
| 10 | GO | 2020 |

**Insights:** More no. of customers are present in the state of SP in Brazil. Remaining states customers are distributed.

**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.**
ANS: `WITH final AS`

```
(SELECT  Extract(year FROM o.order_purchase_timestamp)AS year,
        Sum(p.payment_value)AS total
FROM `target_sql.payments`p
JOIN `target_sql.orders`o
ON o.order_id=p.order_id
WHERE o.order_purchase_timestamp between '2017-01-01' AND '2017-08-01'
      OR o.order_purchase_timestamp between '2018-01-01' AND '2018-08-
01'
GROUP BY 1)
SELECT *,
       (100*((LEAD(total)over(ORDER BY year)-total)/total))AS
percentage
FROM final
Order by year
```

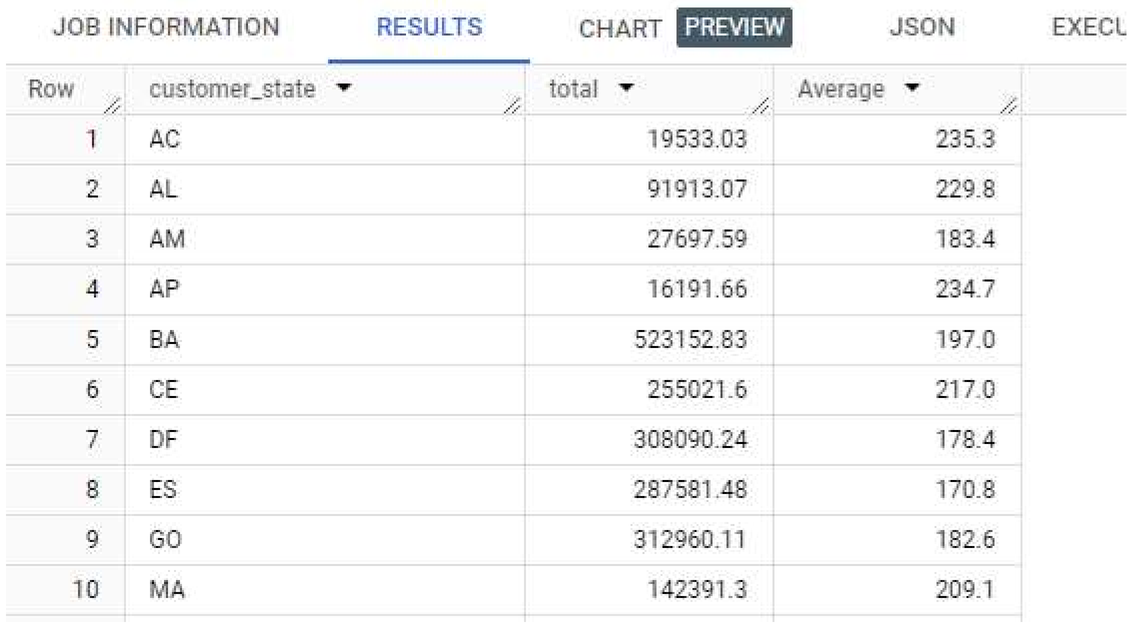


| Row | year | total | percentage |
|---|---|---|---|
| 1 | 2017 | 2994625.800000... | 156.2025786326... |
| 2 | 2018 | 7672308.519999... | null |

**Insights:** The percentage increase of cost of orders in 2018 is 156.20 as compared with the past yeari.e 2017.

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

ANS:
```
SELECT  DISTINCT(c.customer_state),

        Sum(p.payment_value)over(partition by c.customer_state)AS
total,
        Round(AVG(p.payment_value)over(partition by
c.customer_state),1) AS Average
FROM `target_sql.payments`p
JOIN `target_sql.orders`o
ON o.order_id=p.order_id
JOIN `target_sql.customers`c
ON c.customer_id=o.customer_id
GROUP BY 1,p.payment_value
ORDER BY 1
```

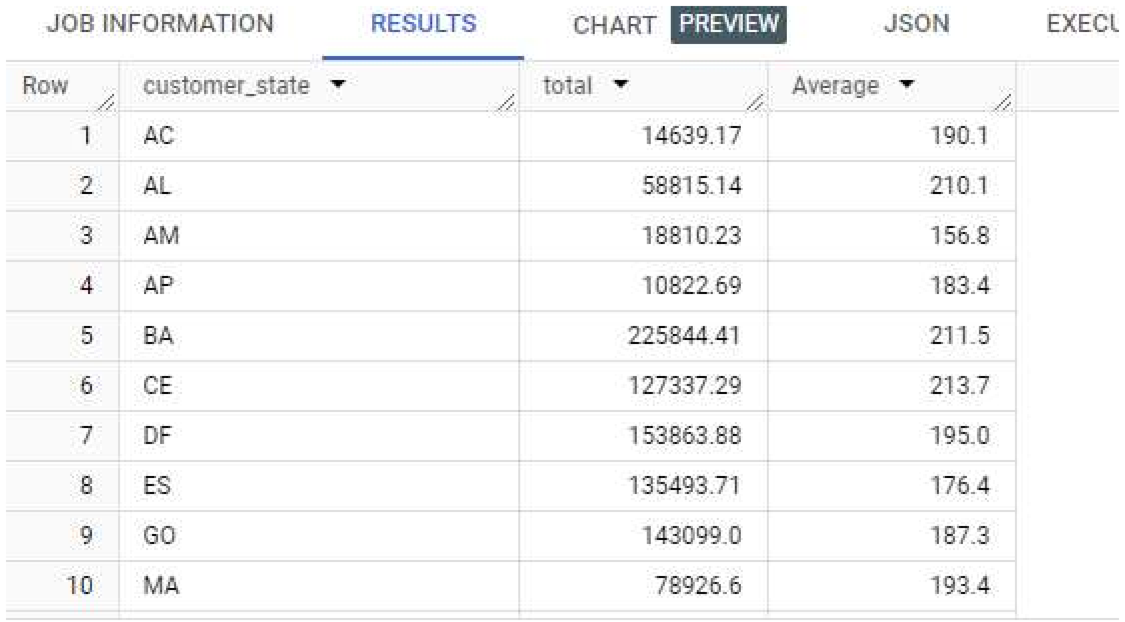


JOB INFORMATION RESULTS CHART PREVIEW JSON EXECU

| Row | customer_state | total | Average |
|---|---|---|---|
| 1 | AC | 19533.03 | 235.3 |
| 2 | AL | 91913.07 | 229.8 |
| 3 | AM | 27697.59 | 183.4 |
| 4 | AP | 16191.66 | 234.7 |
| 5 | BA | 523152.83 | 197.0 |
| 6 | CE | 255021.6 | 217.0 |
| 7 | DF | 308090.24 | 178.4 |
| 8 | ES | 287581.48 | 170.8 |
| 9 | GO | 312960.11 | 182.6 |
| 10 | MA | 142391.3 | 209.1 |

ANS:
```
SELECT  DISTINCT(c.customer_state),

        Sum(oi.price)over(partition by c.customer_state)AS total,
        Round(AVG(oi.price)over(partition by c.customer_state),1) AS
Average
FROM `target_sql.order_items`oi
JOIN `target_sql.orders`o
```

```
ON o.order_id=oi.order_id
JOIN `target_sql.customers`c
ON c.customer_id=o.customer_id
GROUP BY 1,oi.price
ORDER BY 1
```

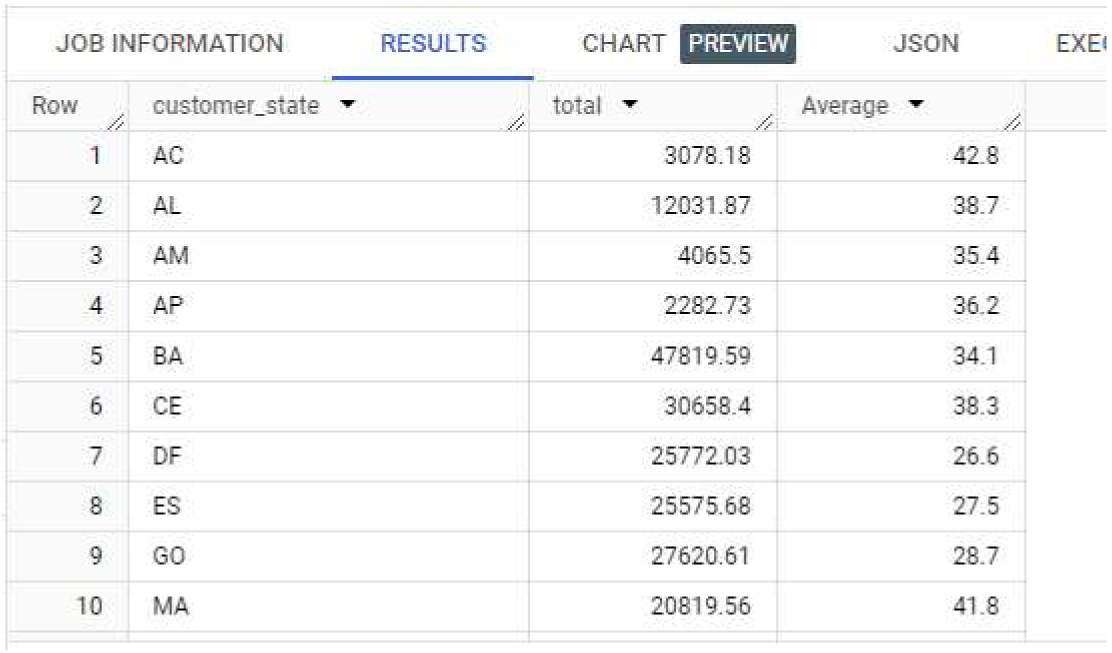


| Row | customer_state | total | Average |
|---|---|---|---|
| 1 | AC | 14639.17 | 190.1 |
| 2 | AL | 58815.14 | 210.1 |
| 3 | AM | 18810.23 | 156.8 |
| 4 | AP | 10822.69 | 183.4 |
| 5 | BA | 225844.41 | 211.5 |
| 6 | CE | 127337.29 | 213.7 |
| 7 | DF | 153863.88 | 195.0 |
| 8 | ES | 135493.71 | 176.4 |
| 9 | GO | 143099.0 | 187.3 |
| 10 | MA | 78926.6 | 193.4 |

**Insights:** The Total And Average value of each state is mentioned above.

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

ANS:
```
SELECT  DISTINCT(c.customer_state),

        Sum(oi.freight_value)over(partition by c.customer_state)AS
total,
        Round(AVG(oi.freight_value)over(partition by
c.customer_state),1) AS Average
FROM `target_sql.order_items`oi
JOIN `target_sql.orders`o
ON o.order_id=oi.order_id
JOIN `target_sql.customers`c
ON c.customer_id=o.customer_id
GROUP BY 1,oi.freight_value
ORDER BY 1
```



| Row | customer_state | total | Average |
|---|---|---|---|
| 1 | AC | 3078.18 | 42.8 |
| 2 | AL | 12031.87 | 38.7 |
| 3 | AM | 4065.5 | 35.4 |
| 4 | AP | 2282.73 | 36.2 |
| 5 | BA | 47819.59 | 34.1 |
| 6 | CE | 30658.4 | 38.3 |
| 7 | DF | 25772.03 | 26.6 |
| 8 | ES | 25575.68 | 27.5 |
| 9 | GO | 27620.61 | 28.7 |
| 10 | MA | 20819.56 | 41.8 |

**Insights:** The Total And Average value of freight value per each state is mentioned above.

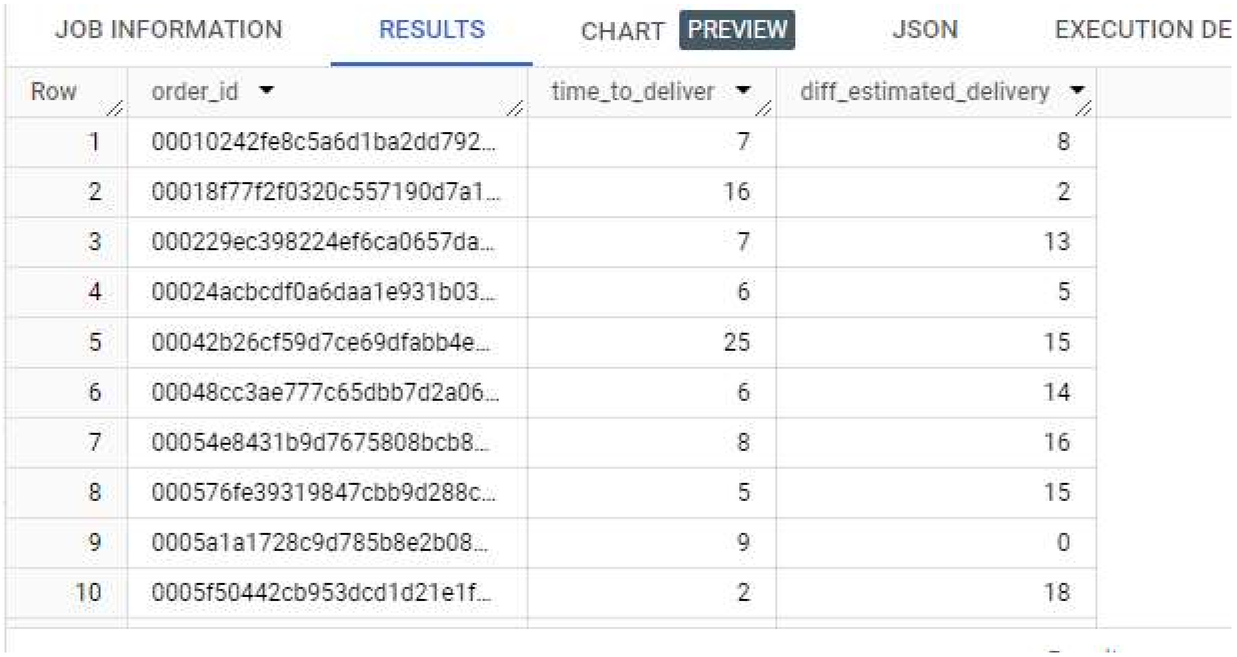
**5.Analysis based on sales, freight and delivery time.**

**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:**

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

**ANS:**
```
SELECT order_id,

 TIMESTAMP_DIFF(order_delivered_customer_date,order_purchase_timestamp
,day)AS time_to_deliver,

 TIMESTAMP_DIFF(order_estimated_delivery_date,order_delivered_customer
_date,day)AS diff_estimated_delivery
FROM `target_sql.orders`
order by order_id
```

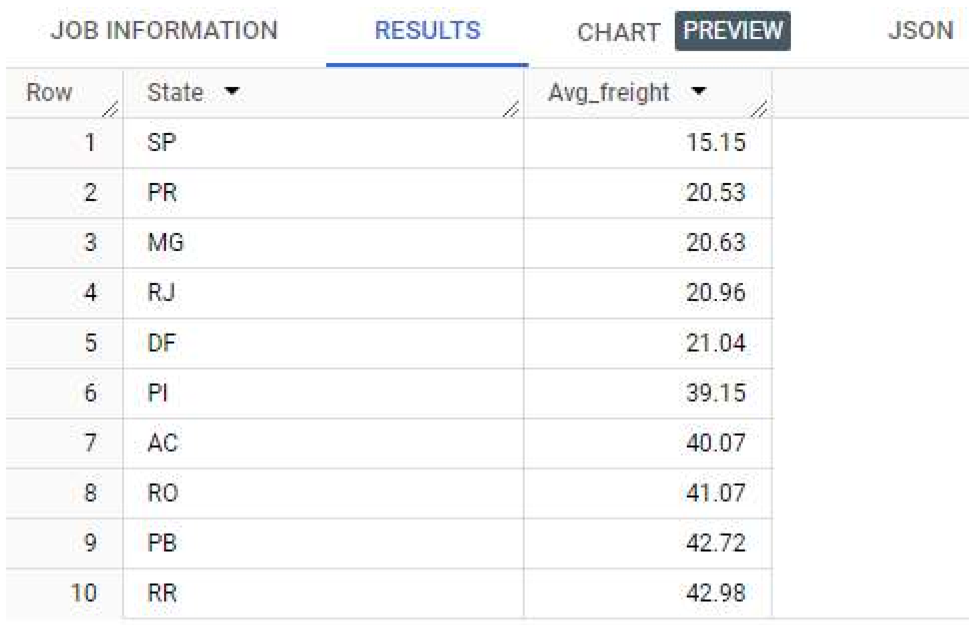


| Row | order_id | time_to_deliver | diff_estimated_delivery |
|---|---|---|---|
| 1 | 00010242fe8c5a6d1ba2dd792... | 7 | 8 |
| 2 | 00018f77f2f0320c557190d7a1... | 16 | 2 |
| 3 | 000229ec398224ef6ca0657da... | 7 | 13 |
| 4 | 00024acbcdf0a6daa1e931b03... | 6 | 5 |
| 5 | 00042b26cf59d7ce69dfabb4e... |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

**Insights:** The time to deliver and difference of estimated delivery is given by order id here we use the orders table.

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

ANS: 
```
SELECT State,Avg_freight

FROM(SELECT Distinct(c.customer_state) AS State,
       ROUND(Avg(oi.freight_value),2) AS Avg_freight,
       ROW_NUMBER() OVER (ORDER BY Avg(oi.freight_value)DESC) AS
rank_high,
       ROW_NUMBER() OVER (ORDER BY Avg(oi.freight_value)ASC) AS
rank_low
FROM `target_sql.customers`c
JOIN `target_sql.orders`o
ON o.customer_id=c.customer_id
JOIN `target_sql.order_items`oi
ON oi.order_id=o.order_id
WHERE freight_value is not null
GROUP BY 1)
where rank_high <=5 or rank_low <=5
order by 2
```

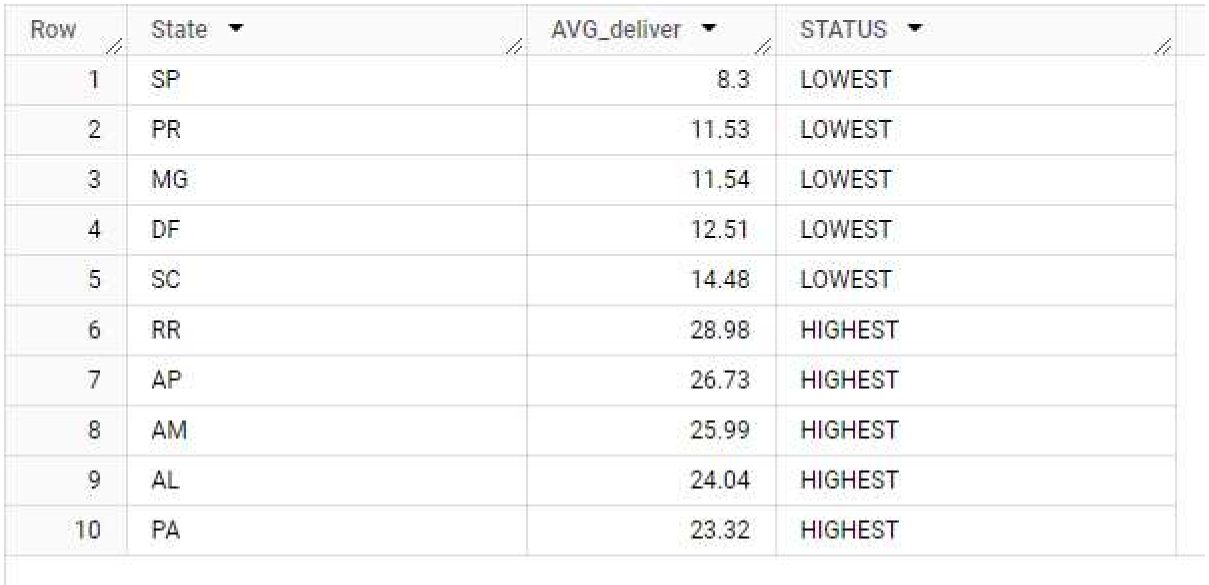


| Row | State | Avg_freight |
|---|---|---|
| 1 | SP | 15.15 |
| 2 | PR | 20.53 |
| 3 | MG | 20.63 |
| 4 | RJ | 20.96 |
| 5 | DF | 21.04 |
| 6 | PI | 39.15 |
| 7 | AC | 40.07 |
| 8 | RO | 41.07 |
| 9 | PB | 42.72 |
| 10 | RR | 42.98 |

**Insights:** Here we have the top 5 state and least 5 states having Average Freight values in Brazil.

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

ANS:
```
(SELECT Distinct(c.customer_state) AS State,

 ROUND(AVG(TIMESTAMP_DIFF(o.order_delivered_customer_date,o.order_purc
hase_timestamp,day)),2)AS AVG_deliver,
       'HIGHEST' AS STATUS
FROM `target_sql.customers`c
JOIN `target_sql.orders`o
ON o.customer_id=c.customer_id
GROUP BY 1
Order by AVG_deliver desc
LIMIT 5)
UNION DISTINCT
(SELECT Distinct(c.customer_state) AS State,

 ROUND(AVG(TIMESTAMP_DIFF(o.order_delivered_customer_date,o.order_purc
hase_timestamp,day)),2)AS AVG_deliver,
       'LOWEST' AS STATUS
FROM `target_sql.customers`c
JOIN `target_sql.orders`o
ON o.customer_id=c.customer_id
GROUP BY 1
Order by AVG_deliver asc
LIMIT 5)
```

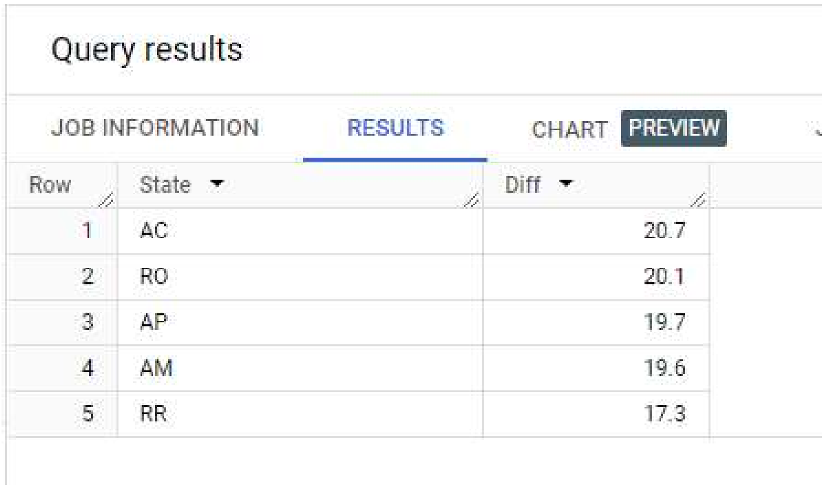


| Row | State | AVG_deliver | STATUS |
|---|---|---|---|
| 1 | SP | 8.3 | LOWEST |
| 2 | PR | 11.53 | LOWEST |
| 3 | MG | 11.54 | LOWEST |
| 4 | DF | 12.51 | LOWEST |
| 5 | SC | 14.48 | LOWEST |
| 6 | RR | 28.98 | HIGHEST |
| 7 | AP | 26.73 | HIGHEST |
| 8 | AM | 25.99 | HIGHEST |
| 9 | AL | 24.04 | HIGHEST |
| 10 | PA | 23.32 | HIGHEST |

**Insights:** In this there are 5 highest and 5 lowest states having 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.**

```
ANS:WITH AVG_table AS
(SELECT c.customer_state AS State,
       DATE_DIFF((EXTRACT(DATE FROM
o.order_delivered_customer_date)),EXTRACT(DATE FROM
o.order_purchase_timestamp),day)AS time_to_deliver,
       DATE_DIFF(EXTRACT(DATE FROM
o.order_estimated_delivery_date),EXTRACT(DATE FROM
o.order_purchase_timestamp),day)AS diff_estimated_delivery
FROM `target_sql.orders`o
JOIN `target_sql.customers`c
ON o.customer_id=c.customer_id
WHERE order_status= 'delivered')
SELECT State,ROUND(AVG(diff_estimated_delivery - time_to_deliver),1)
AS Diff
FROM AVG_table
GROUP BY 1
ORDER BY Diff desc
LIMIT 5
```



Query results

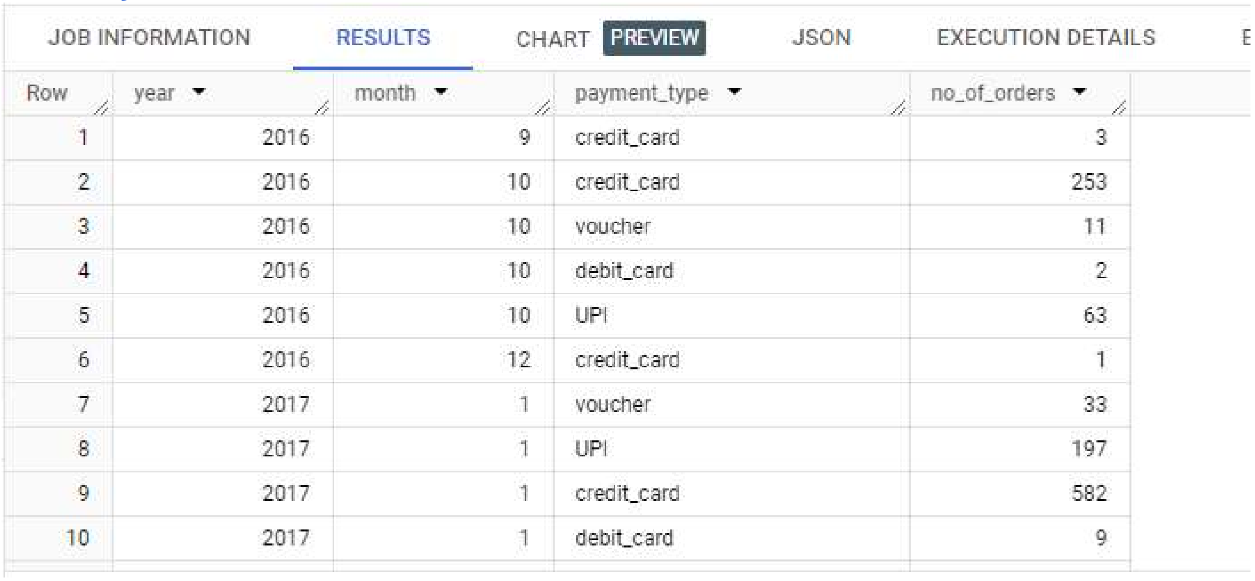
JOB INFORMATION RESULTS CHART PREVIEW

| Row | State | Diff |
|---|---|---|
| 1 | AC | 20.7 |
| 2 | RO | 20.1 |
| 3 | AP | 19.7 |
| 4 | AM | 19.6 |
| 5 | RR | 17.3 |

**Insights:** In this query, we seen that the Fastest delivery in Highest 5 states.

6. Analysis based on the payments:
A. Find the month on month no. of orders placed using different payment types.

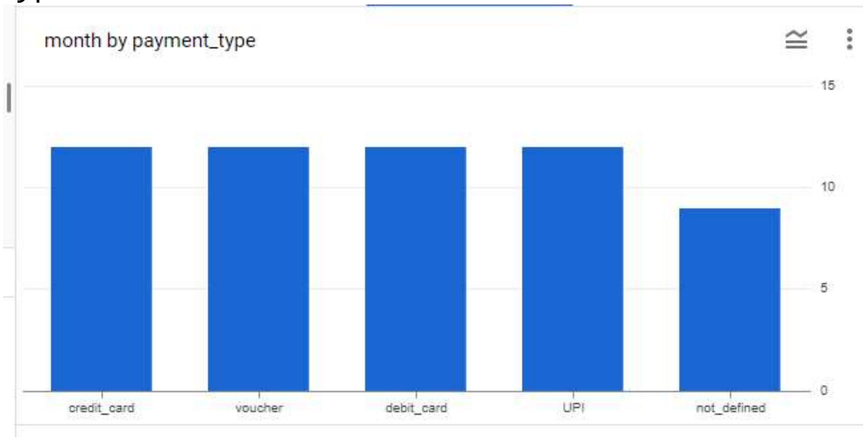
```
ANS: SELECT

       EXTRACT(year FROM o.order_purchase_timestamp)AS year,
       EXTRACT(month FROM o.order_purchase_timestamp)AS
month,p.payment_type,
       count(DISTINCT(p.order_id))AS no_of_orders
FROM `target_sql.payments`p
JOIN `target_sql.orders`o
ON o.order_id=p.order_id
GROUP BY 3,2,1
order by 1,2
```



JOB INFORMATION RESULTS CHART PREVIEW JSON EXECUTION DETAILS

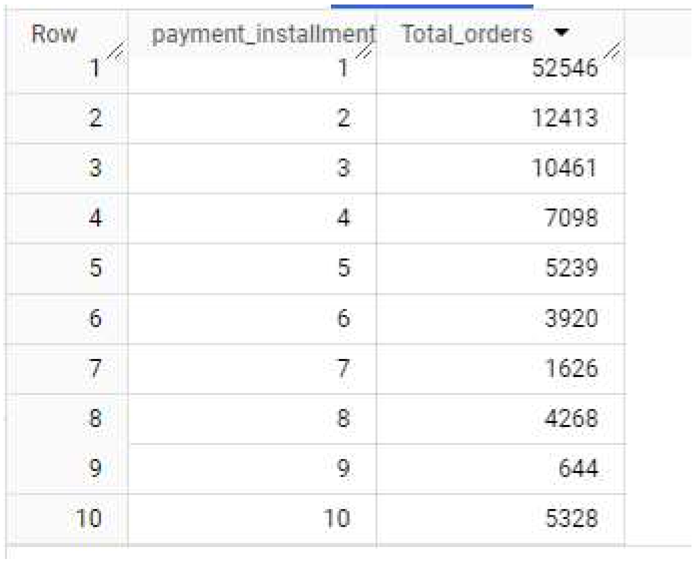
| Row | year | month | payment_type | no_of_orders |
|---|---|---|---|---|
| 1 | 2016 | 9 | credit_card | 3 |
| 2 | 2016 | 10 | credit_card | 253 |
| 3 | 2016 | 10 | voucher | 11 |
| 4 | 2016 | 10 | debit_card | 2 |
| 5 | 2016 | 10 | UPI | 63 |
| 6 | 2016 | 12 | credit_card | 1 |
| 7 | 2017 | 1 | voucher | 33 |
| 8 | 2017 | 1 | UPI | 197 |
| 9 | 2017 | 1 | credit_card | 582 |
| 10 | 2017 | 1 | debit_card | 9 |

**Insights:** Here we understand the payments made by the customers with respect to month, year and no. of orders were payment paid by payment type.



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

**ANS:**
```
SELECT payment_installments,count(order_id)AS Total_orders

FROM `target_sql.payments`
WHERE payment_installments > 0
GROUP BY 1
ORDER BY 1
```



| Row | payment_installment | Total_orders |
|---|---|---|
| 1 | 1 | 52546 |
| 2 | 2 | 12413 |
| 3 | 3 | 10461 |
| 4 | 4 | 7098 |
| 5 | 5 | 5239 |
| 6 | 6 | 3920 |
| 7 | 7 | 1626 |
| 8 | 8 | 4268 |
| 9 | 9 | 644 |
| 10 | 10 | 5328 |

**Insights:** In this we have a look on the Total orders placed by making payment by installments in brazil.