NOTE- DATASET NAME TARGET_SQL

Assumption- Difference in Time zones is taken in account in the dataset itself

Question 1) part-2) Time period for which the data is given

2016-09-04 21:15:19 UTC TO 2018-10-17 17:30:18 UTC

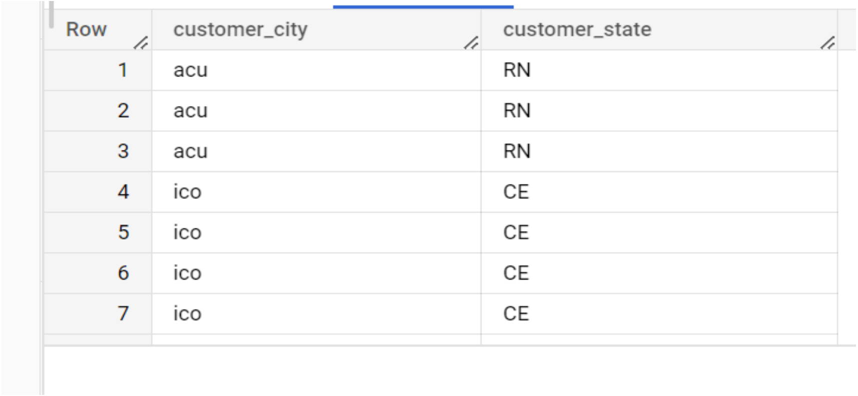
4 september 2016 to 17 october 2018

Question 1) part-3 Cities and States of customers ordered during the given period FOR TABLE customers

QUERY-

```
SELECT c.customer_city, c.customer_state FROM `TARGET_SQL.customers`AS c
JOIN `TARGET_SQL.orders`AS o ON c.customer_id=o.customer_id
WHERE (EXTRACT(DATE FROM o.order_purchase_timestamp)) BETWEEN '2016-09-04' and '2018-
10-17'
```

OUTPUT-



Row	customer_city	customer_state
1	acu	RN
2	acu	RN
3	acu	RN
4	ico	CE
5	ico	CE
6	ico	CE
7	ico	CE

Question 2 Seasonal variation in orders

Asuumption- 1)Money Value of all orders is almost same

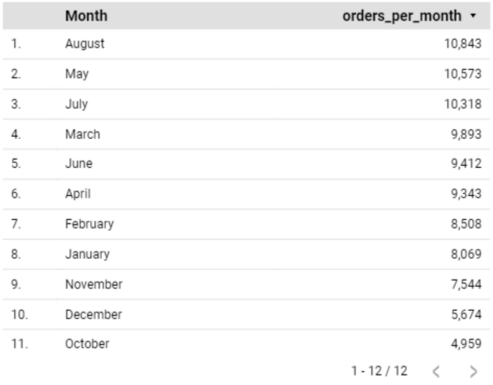
2)Order_id of all orders is different

3)Cancelled and undelivered orders are also counted

QUERY FOR MONTHLY DISTRIBUTION OF ORDERS

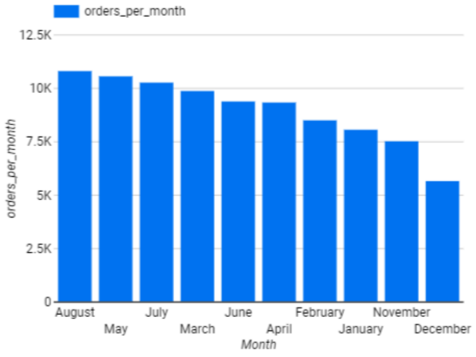
```
SELECT COUNT(order_id), EXTRACT (Year FROM order_purchase_timestamp)  AS YEAR  FROM `T
ARGET_SQL.orders` GROUP BY EXTRACT (Year FROM order_purchase_timestamp)
```

BigQuery Custom SQL



	Month	orders_per_month ▾
1.	August	10,843
2.	May	10,573
3.	July	10,318
4.	March	9,893
5.	June	9,412
6.	April	9,343
7.	February	8,508
8.	January	8,069
9.	November	7,544
10.	December	5,674
11.	October	4,959

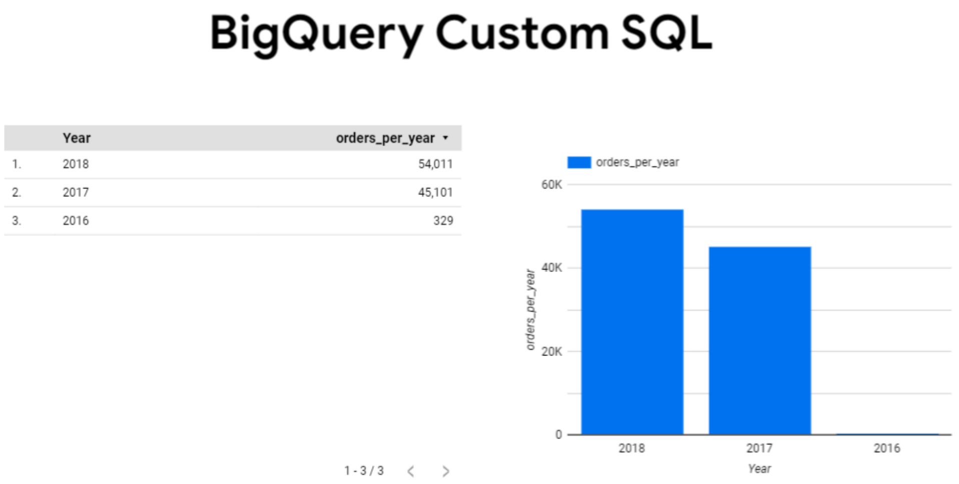
1 - 12 / 12 < >



Yearly distribution of orders

Query

```
SELECT COUNT(order_id), EXTRACT (Year FROM order_purchase_timestamp)  AS YEAR  FROM `T
ARGET_SQL.orders` GROUP BY EXTRACT (Year FROM order_purchase_timestamp)
```



BigQuery Custom SQL

	Year	orders_per_year ▾
1.	2018	54,011
2.	2017	45,101
3.	2016	329

Insights- 1)It can be observed that there is small percentage increase in the numbers of orders placed year on year basis

2)There is no absolute trend in the number of orders on monthly basis but the maximum number of orders are placed in the months of August, May and July.

3)Least number of orders placed in the months of October, December and November

Reccomendation-

1)The reason for drop in sales needs to be examined and corrective actions like Discount Offers, Sale should be organised in these months.

2)Fresh products can be introduced in the months of August,July and May as in these months the market sentiment is positive and sales are generally good.

DISTRIBUTION OF ORDERS TIME PERIOD WISE(DAWN,MORNING,AFTERNOON AND NIGHT)

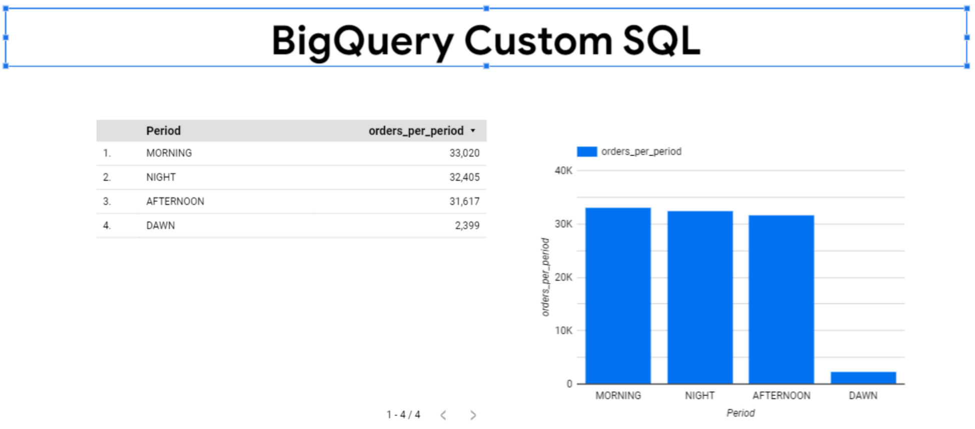
ASSUMPTION -1)Dawn(3-7),Morning(8-13),Afternoon(14-18),Night(19-24,0-2)

2)It is assumed that the time in UTC in database was adjusted for timezone in Brazil

QUERY

```
SELECT O.Period , COUNT(order_id) AS orders_per_period FROM

(SELECT order_id,
EXTRACT (Month FROM order_purchase_timestamp) AS Month,
EXTRACT (Hour FROM order_purchase_timestamp)  AS Hour,
EXTRACT (Year FROM order_purchase_timestamp)  AS YEAR,
CASE WHEN
EXTRACT (Hour FROM order_purchase_timestamp) BETWEEN 3 AND 7
THEN 'DAWN'
WHEN EXTRACT (Hour FROM order_purchase_timestamp) BETWEEN 8 AND 13
THEN 'MORNING'
WHEN EXTRACT (Hour FROM order_purchase_timestamp) BETWEEN 14 AND 18
THEN 'AFTERNOON'
WHEN EXTRACT (Hour FROM order_purchase_timestamp) BETWEEN 19 AND 24 OR
EXTRACT (Hour FROM order_purchase_timestamp) BETWEEN 0 AND 2
THEN 'NIGHT'
END AS Period FROM `TARGET_SQL.orders`) as O
GROUP BY O.Period
```

GRAPHICAL REPRESENTATION



BigQuery Custom SQL

	Period	orders_per_period ▾
1.	MORNING	33,020
2.	NIGHT	32,405
3.	AFTERNOON	31,617
4.	DAWN	2,399

1 - 4 / 4

Insights -There is no big difference in the number of orders placed in Morning, Night and Afternoon. However maximum number of orders were placed in Morning then in Night then in Afternoon. Very less number of orders were placed during the Dawn.

QUESTION 3

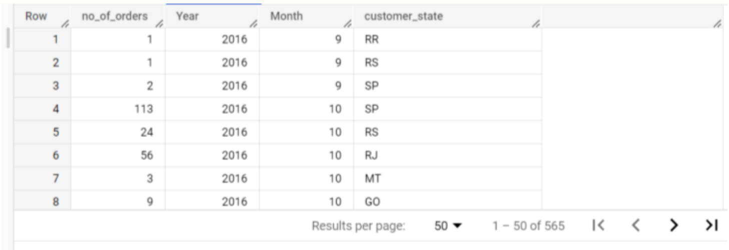
PART1) Month on month orders by state

QUERY

```
with tem as(
  select customer_id, order_id,date(order_purchase_timestamp) date_detail from `TARGET
_SQL.orders`
),
tem2 as (
  select t.order_id, c.customer_state, date_detail from tem t inner join `TARGET_SQL.c
ustomers` c
  on t.customer_id = c.customer_id
),
tem3 as(
  select count(order_id) no_of_orders, extract(year from date_detail) Year,
  extract(month from date_detail) Month, customer_state from tem2
  group by extract(month from date_detail),extract(year from date_detail), customer_st
ate
)
```

```
select * from tem3 order by Year, Month
```

OUTPUT



Row	no_of_orders	Year	Month	customer_state
1	1	2016	9	RR
2	1	2016	9	RS
3	2	2016	9	SP
4	113	2016	10	SP
5	24	2016	10	RS
6	56	2016	10	RJ
7	3	2016	10	MT
8	9	2016	10	GO

Results per page: 50 ▾ 1 – 50 of 565

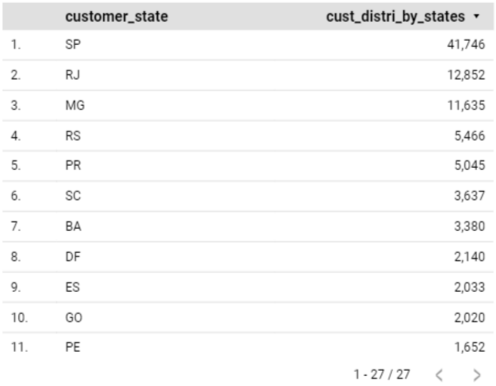
PART 2) Distribution of customers across the states in Brazi

ASSUMPTION – It is assumed that all the customers have registered in the customer table

QUERY

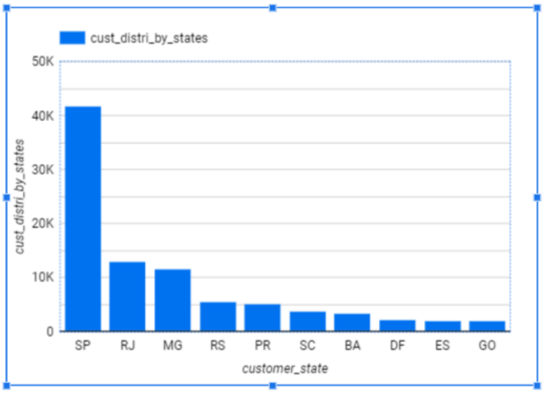
```
SELECT * FROM
(SELECT  customer_state, COUNT(customer_id) AS cust_distri_by_states  FROM
`TARGET_SQL.customers`
GROUP BY customer_state) AS x
ORDER BY x.cust_distri_by_states DESC
```

GRAPHICAL DISTRIBUTION



	customer_state	cust_distri_by_states ▾
1.	SP	41,746
2.	RJ	12,852
3.	MG	11,635
4.	RS	5,466
5.	PR	5,045
6.	SC	3,637
7.	BA	3,380
8.	DF	2,140
9.	ES	2,033
10.	GO	2,020
11.	PE	1,652

1 - 27 / 27



Insight-

1) A major chunk of around 40% of customers are from SP state. Other States like RJ, MG also have high number of customers.
2) Very less number of orders were placed from the following states PE,AN,AL,SE,TO,RO,AM,AC,AP,AR.

Reccomendations-

1)It can be due to various reasons like relatively less number of outlets in these states,lesser population density, low per capita income etc.

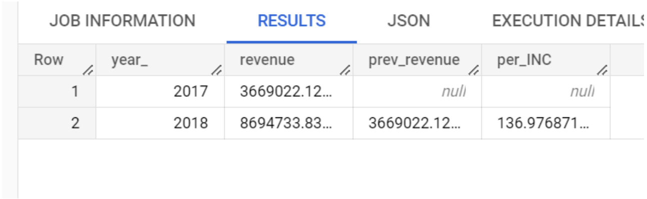
2) Proper attention should be paid towards increasing presence in these states and optimising products and giving suitable discount offers etc.

QUESTION 4

1. Get % increase in cost of orders from 2017 to 2018 (include months between Jan to Aug only) - You can use “payment_value” column in payments table

```
WITH
  base AS (
  SELECT
    EXTRACT(year
    FROM
      orders.order_purchase_timestamp) AS year_,
    SUM(payments.payment_value) AS revenue
  FROM
    `TARGET_SQL.orders` AS orders
  INNER JOIN
    `TARGET_SQL.payments` AS payments
  ON
    orders.order_id = payments.order_id
  WHERE
    EXTRACT(month
    FROM
      orders.order_purchase_timestamp) BETWEEN 0
    AND 8
  GROUP BY
    year_),
  base2 AS(
  SELECT
    *,
    LAG(revenue) OVER(ORDER BY year_ ASC) AS prev_revenue
  FROM
    base)
SELECT *,
  (revenue-prev_revenue)*100/prev_revenue AS per_INC
FROM
  base2
```

RESULT



Row	year_	revenue	prev_revenue	per_INC
1	2017	3669022.12...	null	null
2	2018	8694733.83...	3669022.12...	136.976871...

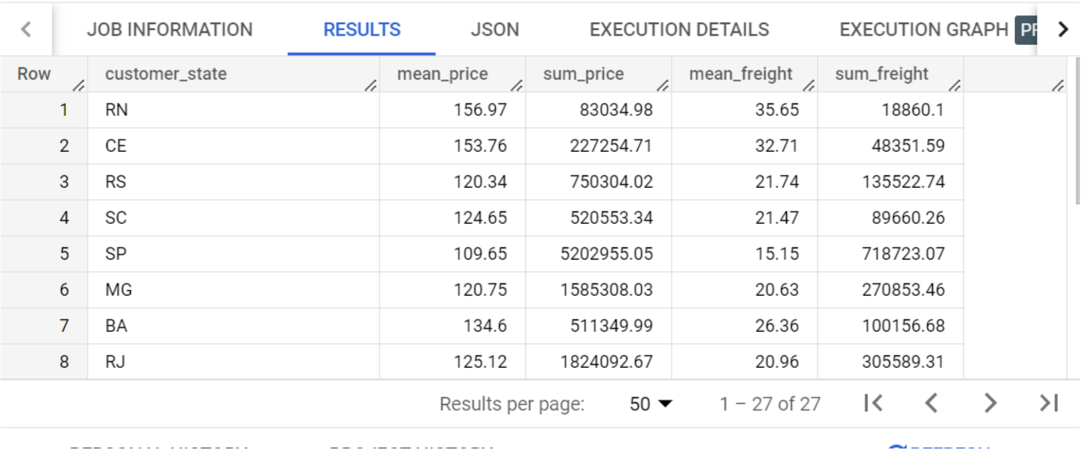
Insight- Revenue increased significantly by 136.97% in 2018 as compared to 2017.

2) Mean & Sum of price and freight value by customer state

QUERY

```
SELECT c.customer_state,

ROUND(AVG(oi.price),2) AS mean_price,

ROUND(SUM(oi.price),2) AS sum_price,

ROUND(AVG(oi.freight_value),2) AS mean_freight,

ROUND(SUM(oi.freight_value),2) AS sum_freight

FROM `TARGET_SQL.customers` AS c

JOIN `TARGET_SQL.orders`AS o

ON c.customer_id = o.customer_id

JOIN `TARGET_SQL.order_items`AS oi

ON o.order_id = oi.order_id

GROUP BY c.customer_state
```

Result



Row	customer_state	mean_price	sum_price	mean_freight	sum_freight
1	RN	156.97	83034.98	35.65	18860.1
2	CE	153.76	227254.71	32.71	48351.59
3	RS	120.34	750304.02	21.74	135522.74
4	SC	124.65	520553.34	21.47	89660.26
5	SP	109.65	5202955.05	15.15	718723.07
6	MG	120.75	1585308.03	20.63	270853.46
7	BA	134.6	511349.99	26.36	100156.68
8	RJ	125.12	1824092.67	20.96	305589.31

Results per page: 50 1 – 27 of 27

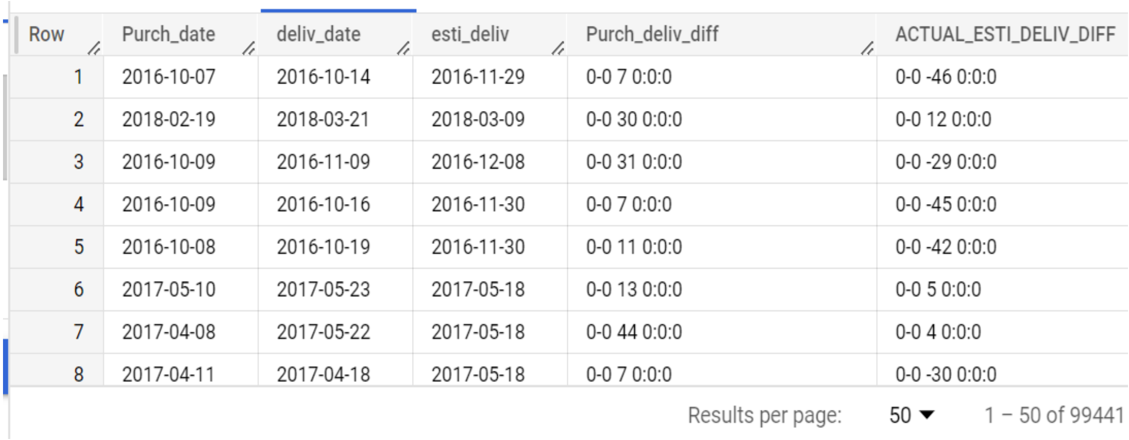
5

1) Calculate days between purchasing, delivering and estimated delivery

QUERY

```
SELECT *, (x.deliv_date-x.Purch_date) AS Purch_deliv_diff,
(x.deliv_date-x.esti_deliv) AS ACTUAL_ESTI_DELIV_DIFF
 FROM
(SELECT
EXTRACT(DATE FROM order_purchase_timestamp) AS Purch_date,
EXTRACT(DATE FROM order_delivered_customer_date) AS deliv_date,
EXTRACT(DATE FROM order_estimated_delivery_date) AS esti_deliv,
FROM `TARGET_SQL.orders`) AS x
```

Result



Row	Purch_date	deliv_date	esti_deliv	Purch_deliv_diff	ACTUAL_ESTI_DELIV_DIFF
1	2016-10-07	2016-10-14	2016-11-29	0-0 7 0:0:0	0-0 -46 0:0:0
2	2018-02-19	2018-03-21	2018-03-09	0-0 30 0:0:0	0-0 12 0:0:0
3	2016-10-09	2016-11-09	2016-12-08	0-0 31 0:0:0	0-0 -29 0:0:0
4	2016-10-09	2016-10-16	2016-11-30	0-0 7 0:0:0	0-0 -45 0:0:0
5	2016-10-08	2016-10-19	2016-11-30	0-0 11 0:0:0	0-0 -42 0:0:0
6	2017-05-10	2017-05-23	2017-05-18	0-0 13 0:0:0	0-0 5 0:0:0
7	2017-04-08	2017-05-22	2017-05-18	0-0 44 0:0:0	0-0 4 0:0:0
8	2017-04-11	2017-04-18	2017-05-18	0-0 7 0:0:0	0-0 -30 0:0:0

Results per page: 50 1 – 50 of 99441

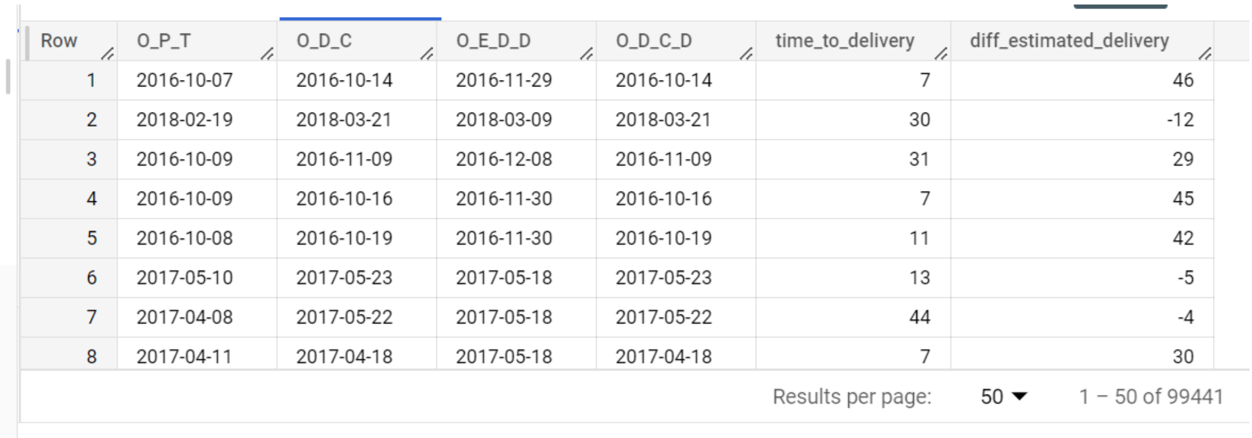
2)Find time_to_delivery & diff_estimated_delivery.

QUERY

```
SELECT *, date_diff(x.O_D_C,x.O_P_T,day) as time_to_delivery,
date_diff(x.O_E_D_D,x.O_D_C_D,day) AS diff_estimated_delivery
```

```
FROM
(SELECT
EXTRACT(DATE FROM order_purchase_timestamp) AS O_P_T,
EXTRACT(DATE FROM order_delivered_customer_date) AS O_D_C,
EXTRACT(DATE FROM order_estimated_delivery_date) AS O_E_D_D,
EXTRACT(DATE FROM order_delivered_customer_date) AS O_D_C_D
FROM `TARGET_SQL.orders`) AS x
```

RESULT



Row	O_P_T	O_D_C	O_E_D_D	O_D_C_D	time_to_delivery	diff_estimated_delivery
1	2016-10-07	2016-10-14	2016-11-29	2016-10-14	7	46
2	2018-02-19	2018-03-21	2018-03-09	2018-03-21	30	-12
3	2016-10-09	2016-11-09	2016-12-08	2016-11-09	31	29
4	2016-10-09	2016-10-16	2016-11-30	2016-10-16	7	45
5	2016-10-08	2016-10-19	2016-11-30	2016-10-19	11	42
6	2017-05-10	2017-05-23	2017-05-18	2017-05-23	13	-5
7	2017-04-08	2017-05-22	2017-05-18	2017-05-22	44	-4
8	2017-04-11	2017-04-18	2017-05-18	2017-04-18	7	30

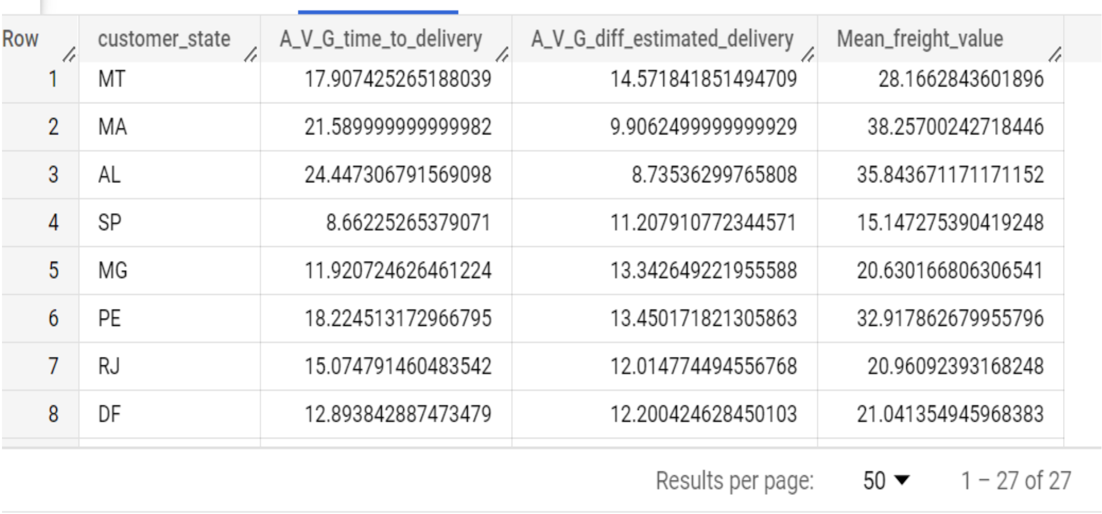
Here- order_purchase_timestamp is O_P_T, order_delivered_customer_date is O_D_C, order_estimated_delivery_date is O_E_D_D and order_delivered_customer_date is O_D_C_D)

3.)Group data by state, take mean of freight_value, time_to_delivery, diff_estimated_delivery

Query

```
SELECT x.customer_state,
  AVG(DATE_DIFF(x.O_D_C,x.O_P_T,day)) AS A_V_G_time_to_delivery,
  AVG(DATE_DIFF(x.O_E_D_D,x.O_D_C_D,day)) AS A_V_G_diff_estimated_delivery,
  AVG(x.freight_value) AS Mean_freight_value
FROM (
  SELECT
    c.customer_state,oi.freight_value,
    EXTRACT(DATE
    FROM
      O.order_purchase_timestamp) AS O_P_T,
    EXTRACT(DATE
    FROM
      O.order_delivered_customer_date) AS O_D_C,
    EXTRACT(DATE
    FROM
      O.order_estimated_delivery_date) AS O_E_D_D,
    EXTRACT(DATE
    FROM
      O.order_delivered_customer_date) AS O_D_C_D,
  FROM
    `TARGET_SQL.orders`AS O
  JOIN
    `TARGET_SQL.customers` AS c
  ON
    O.customer_id=c.customer_id
    JOIN `TARGET_SQL.order_items`AS oi
    ON O.order_id=oi.order_id) AS x
GROUP BY
  customer_state
```

OUTPUT



Row	customer_state	A_V_G_time_to_delivery	A_V_G_diff_estimated_delivery	Mean_freight_value
1	MT	17.907425265188039	14.571841851494709	28.1662843601896
2	MA	21.589999999999982	9.9062499999999929	38.25700242718446
3	AL	24.447306791569098	8.73536299765808	35.843671171171152
4	SP	8.66225265379071	11.207910772344571	15.147275390419248
5	MG	11.920724626461224	13.342649221955588	20.630166806306541
6	PE	18.224513172966795	13.450171821305863	32.917862679955796
7	RJ	15.074791460483542	12.014774494556768	20.96092393168248
8	DF	12.893842887473479	12.200424628450103	21.041354945968383

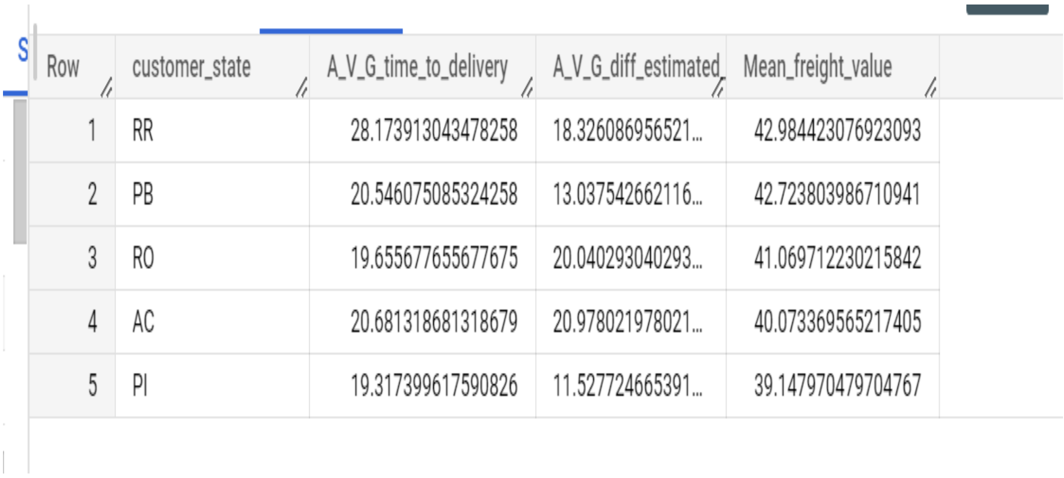
4.) Sort the data to get the following:

5.)(a)Top 5 states with highest average freight value - sort in desc limit 5

QUERY

```
SELECT
  x.customer_state,
  AVG(DATE_DIFF(x.O_D_C,x.O_P_T,day)) AS A_V_G_time_to_delivery,
  AVG(DATE_DIFF(x.O_E_D_D,x.O_D_C_D,day)) AS A_V_G_diff_estimated_delivery,
  AVG(x.freight_value) AS Mean_freight_value
FROM (
  SELECT
    c.customer_state,
    oi.freight_value,
    EXTRACT(DATE
    FROM
      O.order_purchase_timestamp) AS O_P_T,
    EXTRACT(DATE
    FROM
      O.order_delivered_customer_date) AS O_D_C,
    EXTRACT(DATE
    FROM
      O.order_estimated_delivery_date) AS O_E_D_D,
    EXTRACT(DATE
    FROM
      O.order_delivered_customer_date) AS O_D_C_D,
  FROM
    `TARGET_SQL.orders`AS O
  JOIN
    `TARGET_SQL.customers` AS c
  ON
    O.customer_id=c.customer_id
  JOIN
    `TARGET_SQL.order_items`AS oi
  ON
    O.order_id=oi.order_id) AS x
GROUP BY
  customer_state
ORDER BY
  Mean_freight_value DESC
  LIMIT 5
```

OutPut



Row	customer_state	A_V_G_time_to_delivery	A_V_G_diff_estimated_	Mean_freight_value
1	RR	28.173913043478258	18.326086956521...	42.984423076923093
2	PB	20.546075085324258	13.037542662116...	42.723803986710941
3	RO	19.655677655677675	20.040293040293...	41.069712230215842
4	AC	20.681318681318679	20.978021978021...	40.073369565217405
5	PI	19.317399617590826	11.527724665391...	39.147970479704767

Recommendations –

1)Efficient transport systems should be used for these states and bigger vehicles can be used to reduce per order freight value.

2)Comparatively bigger lot of orders can be selected for single delivery operation. This may increase delivery time for some orders but the overll freight cost will be reduced significantly.

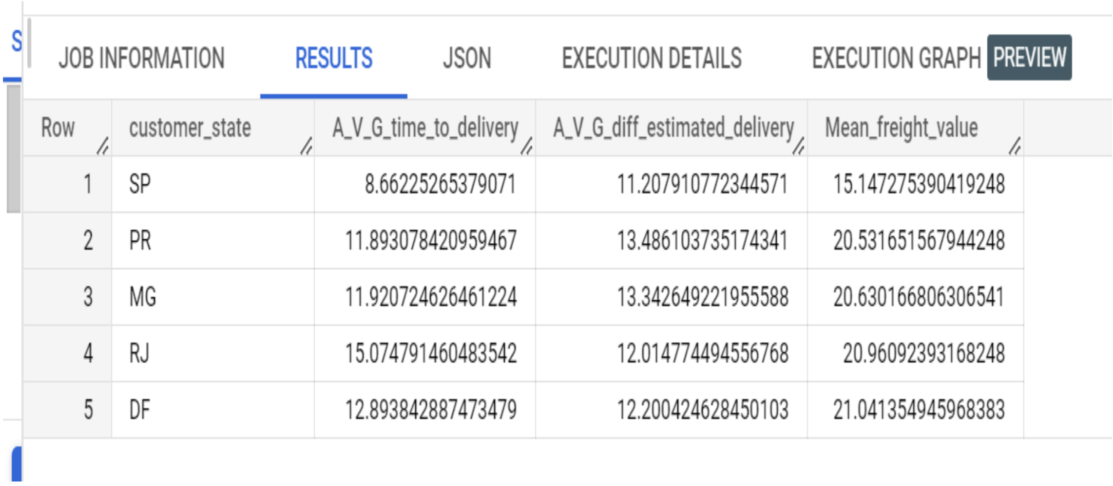
5(b) Top 5 states with lowest average freight value - sort in asc limit

Query

```
SELECT
  x.customer_state,
  AVG(DATE_DIFF(x.O_D_C,x.O_P_T,day)) AS A_V_G_time_to_delivery,
  AVG(DATE_DIFF(x.O_E_D_D,x.O_D_C_D,day)) AS A_V_G_diff_estimated_delivery,
  AVG(x.freight_value) AS Mean_freight_value
FROM (
  SELECT
```

```
    c.customer_state,
    oi.freight_value,
    EXTRACT(DATE
    FROM
      O.order_purchase_timestamp) AS O_P_T,
    EXTRACT(DATE
    FROM
      O.order_delivered_customer_date) AS O_D_C,
    EXTRACT(DATE
    FROM
      O.order_estimated_delivery_date) AS O_E_D_D,
    EXTRACT(DATE
    FROM
      O.order_delivered_customer_date) AS O_D_C_D,
  FROM
    `TARGET_SQL.orders`AS O
  JOIN
    `TARGET_SQL.customers` AS c
  ON
    O.customer_id=c.customer_id
  JOIN
    `TARGET_SQL.order_items`AS oi
  ON
    O.order_id=oi.order_id) AS x
GROUP BY
  customer_state
ORDER BY
  Mean_freight_value ASC
  LIMIT 5
```

OutPut

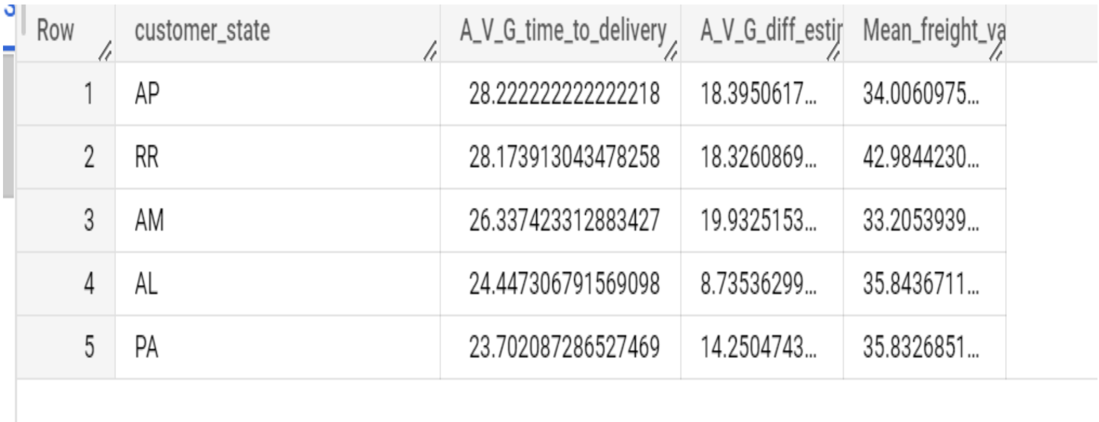


Row	customer_state	A_V_G_time_to_delivery	A_V_G_diff_estimated_delivery	Mean_freight_value
1	SP	8.66225265379071	11.207910772344571	15.147275390419248
2	PR	11.893078420959467	13.486103735174341	20.531651567944248
3	MG	11.920724626461224	13.342649221955588	20.630166806306541
4	RJ	15.074791460483542	12.014774494556768	20.96092393168248
5	DF	12.893842887473479	12.200424628450103	21.041354945968383

6)a) Top 5 states with highest average time to delivery

```
SELECT
  x.customer_state,
  AVG(DATE_DIFF(x.O_D_C,x.O_P_T,day)) AS A_V_G_time_to_delivery,
  AVG(DATE_DIFF(x.O_E_D_D,x.O_D_C_D,day)) AS A_V_G_diff_estimated_delivery,
  AVG(x.freight_value) AS Mean_freight_value
FROM (
  SELECT
    c.customer_state,
    oi.freight_value,
    EXTRACT(DATE
    FROM
      O.order_purchase_timestamp) AS O_P_T,
    EXTRACT(DATE
    FROM
      O.order_delivered_customer_date) AS O_D_C,
    EXTRACT(DATE
    FROM
      O.order_estimated_delivery_date) AS O_E_D_D,
    EXTRACT(DATE
    FROM
      O.order_delivered_customer_date) AS O_D_C_D,
  FROM
    `TARGET_SQL.orders`AS O
  JOIN
    `TARGET_SQL.customers` AS c
  ON
    O.customer_id=c.customer_id
  JOIN
    `TARGET_SQL.order_items`AS oi
  ON
    O.order_id=oi.order_id) AS x
GROUP BY
  customer_state
ORDER BY
   A_V_G_time_to_delivery DESC
  LIMIT 5
```

OUTPUT



Row	customer_state	A_V_G_time_to_delivery	A_V_G_diff_estir	Mean_freight_va
1	AP	28.222222222222218	18.3950617...	34.0060975...
2	RR	28.173913043478258	18.3260869...	42.9844230...
3	AM	26.337423312883427	19.9325153...	33.2053939...
4	AL	24.447306791569098	8.73536299...	35.8436711...
5	PA	23.702087286527469	14.2504743...	35.8326851...

Reccomendations-

_1)Appropriate steps should be taken to reduce the delivery time.

2)This can be done by optimal inventory management and thus optimising the delivery time and freight costs.

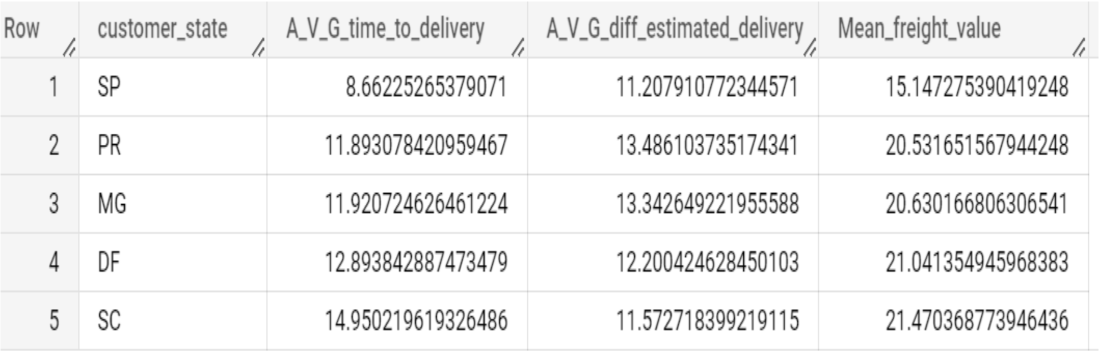
3)Faster vehicles should be used and less crowded routes should be taken by the drivers.

1. 6)b) Top 5 states with lowest average time to delivery

Query

```
SELECT
  x.customer_state,
  AVG(DATE_DIFF(x.O_D_C,x.O_P_T,day)) AS A_V_G_time_to_delivery,
  AVG(DATE_DIFF(x.O_E_D_D,x.O_D_C_D,day)) AS A_V_G_diff_estimated_delivery,
  AVG(x.freight_value) AS Mean_freight_value
FROM (
  SELECT
    c.customer_state,
    oi.freight_value,
    EXTRACT(DATE
    FROM
      O.order_purchase_timestamp) AS O_P_T,
    EXTRACT(DATE
    FROM
      O.order_delivered_customer_date) AS O_D_C,
    EXTRACT(DATE
    FROM
      O.order_estimated_delivery_date) AS O_E_D_D,
    EXTRACT(DATE
    FROM
      O.order_delivered_customer_date) AS O_D_C_D,
  FROM
    `TARGET_SQL.orders`AS O
  JOIN
    `TARGET_SQL.customers` AS c
  ON
    O.customer_id=c.customer_id
  JOIN
    `TARGET_SQL.order_items`AS oi
  ON
    O.order_id=oi.order_id) AS x
GROUP BY
  customer_state
ORDER BY
   A_V_G_time_to_delivery ASC
  LIMIT 5
```

OUTPUT



Row	customer_state	A_V_G_time_to_delivery	A_V_G_diff_estimated_delivery	Mean_freight_value
1	SP	8.66225265379071	11.207910772344571	15.147275390419248
2	PR	11.893078420959467	13.486103735174341	20.531651567944248
3	MG	11.920724626461224	13.342649221955588	20.630166806306541
4	DF	12.893842887473479	12.200424628450103	21.041354945968383
5	SC	14.950219619326486	11.572718399219115	21.470368773946436

Reccomendation-

1)As the delivery time is less for these states so less space can be used as inventory as the orders can be delivered quickly here.

2)Thus cost on inventory can be reduced here

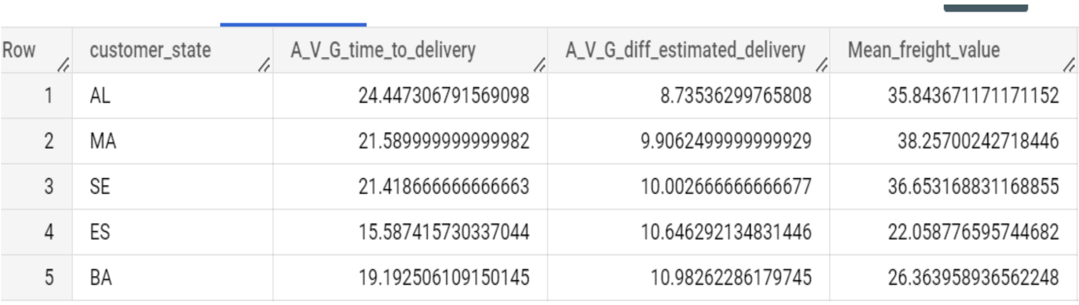
1. **7)a)** Top 5 states where delivery is really fast compared to estimated date

Query

```
SELECT
  x.customer_state,
  AVG(DATE_DIFF(x.O_D_C,x.O_P_T,day)) AS A_V_G_time_to_delivery,
  AVG(DATE_DIFF(x.O_E_D_D,x.O_D_C_D,day)) AS A_V_G_diff_estimated_delivery,
  AVG(x.freight_value) AS Mean_freight_value
FROM (
  SELECT
    c.customer_state,
    oi.freight_value,
    EXTRACT(DATE
    FROM
      O.order_purchase_timestamp) AS O_P_T,
    EXTRACT(DATE
    FROM
      O.order_delivered_customer_date) AS O_D_C,
    EXTRACT(DATE
    FROM
      O.order_estimated_delivery_date) AS O_E_D_D,
    EXTRACT(DATE
    FROM
      O.order_delivered_customer_date) AS O_D_C_D,
  FROM
    `TARGET_SQL.orders`AS O
  JOIN
    `TARGET_SQL.customers` AS c
  ON
    O.customer_id=c.customer_id
  JOIN
    `TARGET_SQL.order_items`AS oi
  ON
```

```
    O.order_id=oi.order_id) AS x
GROUP BY
  customer_state
ORDER BY
A_V_G_diff_estimated_delivery
    ASC
  LIMIT 5
```

OutPut



Row	customer_state	A_V_G_time_to_delivery	A_V_G_diff_estimated_delivery	Mean_freight_value
1	AL	24.447306791569098	8.73536299765808	35.843671171171152
2	MA	21.589999999999982	9.9062499999999929	38.25700242718446
3	SE	21.418666666666663	10.002666666666677	36.653168831168855
4	ES	15.587415730337044	10.646292134831446	22.058776595744682
5	BA	19.192506109150145	10.98262286179745	26.363958936562248

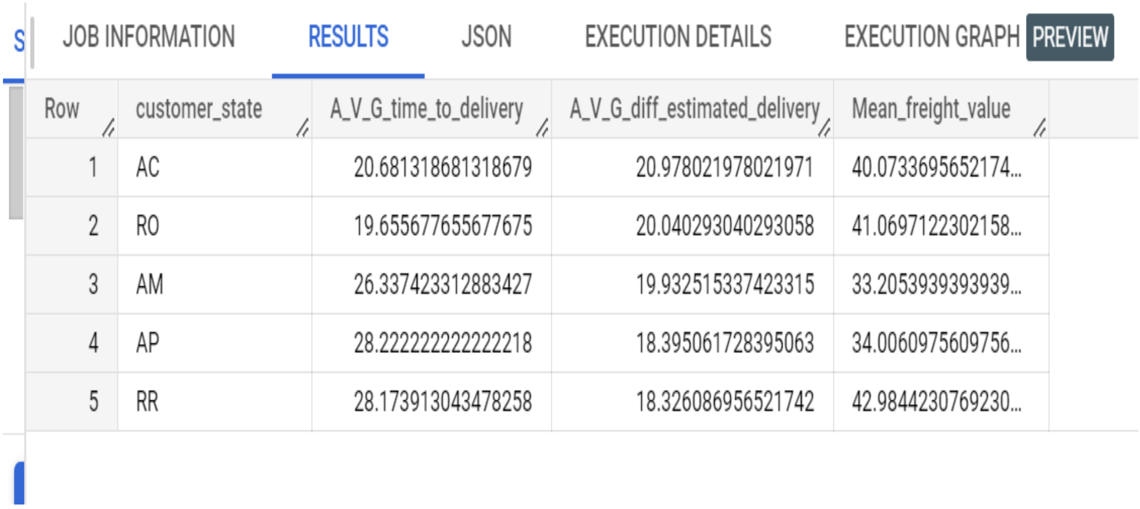
Insight- The estimated and actual delivery time is close for these states

7)b Top 5 states where delivery is not so fast compared to estimated date.

```
SELECT
  x.customer_state,
  AVG(DATE_DIFF(x.O_D_C,x.O_P_T,day)) AS A_V_G_time_to_delivery,
  AVG(DATE_DIFF(x.O_E_D_D,x.O_D_C_D,day)) AS A_V_G_diff_estimated_delivery,
  AVG(x.freight_value) AS Mean_freight_value
FROM (
  SELECT
    c.customer_state,
    oi.freight_value,
    EXTRACT(DATE
    FROM
      O.order_purchase_timestamp) AS O_P_T,
    EXTRACT(DATE
    FROM
      O.order_delivered_customer_date) AS O_D_C,
    EXTRACT(DATE
    FROM
      O.order_estimated_delivery_date) AS O_E_D_D,
    EXTRACT(DATE
    FROM
      O.order_delivered_customer_date) AS O_D_C_D,
  FROM
    `TARGET_SQL.orders`AS O
  JOIN
    `TARGET_SQL.customers` AS c
  ON
    O.customer_id=c.customer_id
  JOIN
    `TARGET_SQL.order_items`AS oi
  ON
```

```
    O.order_id=oi.order_id) AS x
GROUP BY
  customer_state
ORDER BY
A_V_G_diff_estimated_delivery
    DESC
  LIMIT 5
```

OutPut



Row	customer_state	A_V_G_time_to_delivery	A_V_G_diff_estimated_delivery	Mean_freight_value
1	AC	20.681318681318679	20.978021978021971	40.0733695652174...
2	RO	19.655677655677675	20.040293040293058	41.0697122302158...
3	AM	26.337423312883427	19.932515337423315	33.2053939393939...
4	AP	28.222222222222218	18.395061728395063	34.0060975609756...
5	RR	28.173913043478258	18.326086956521742	42.9844230769230...

Recommendations

1)Better programs and parameters should be used to calculate the actual customer delivery time.

2)Better delivery vehicles should be used and number of inventories can be increased for these states in order to reduce the actual customer delivery time period

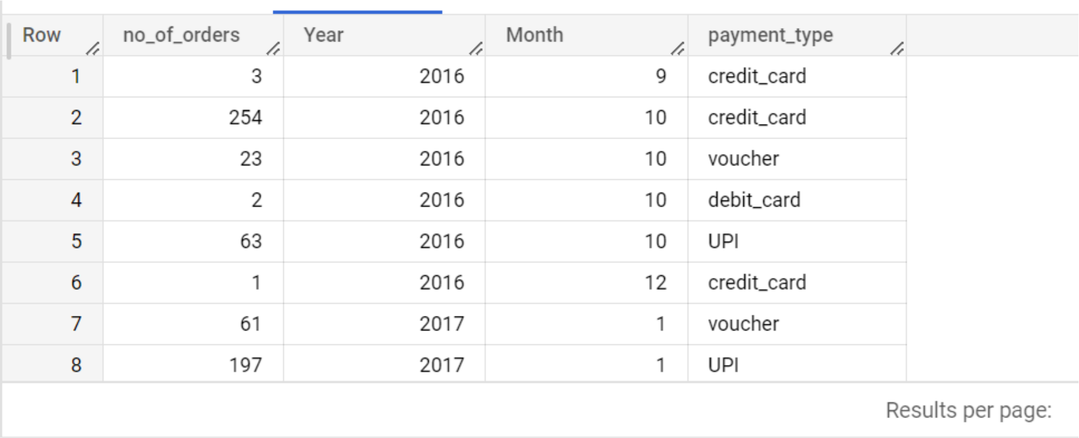
Question 6. Payment type analysis

1. Month over Month count of orders for different payment types

Query

```
with tem as(
  select order_id,date(order_purchase_timestamp) date_detail from `TARGET_SQL.orders`
),
tem2 as (
  select t.order_id,P.payment_type,date_detail from tem t inner join `TARGET_SQL.payme
nts`AS P
  on t.order_id = P.order_id),
tem3 as(
  select count(order_id) no_of_orders, extract(year from date_detail) Year,
  extract(month from date_detail) Month, payment_type from tem2
  group by extract(month from date_detail),extract(year from date_detail), payment_typ
e
)
select * from tem3 order by Year, Month
```

OutPut



Row	no_of_orders	Year	Month	payment_type
1	3	2016	9	credit_card
2	254	2016	10	credit_card
3	23	2016	10	voucher
4	2	2016	10	debit_card
5	63	2016	10	UPI
6	1	2016	12	credit_card
7	61	2017	1	voucher
8	197	2017	1	UPI

INSIGHT- The maximum number of orders are placed using credit card as payment method it is followed by UPI payment and then using Vouchers.

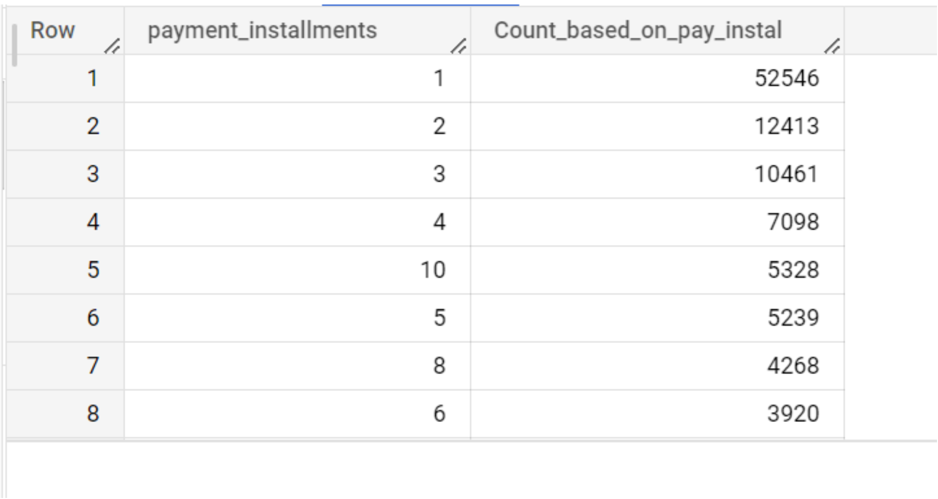
Reccomendations- It seems that majority of people use credit cards, so number of payment points should be increased so as to facilitate hassle free payment.

2)Count of orders based on the no. of payment installments

Query

```
SELECT payment_installments, COUNT(order_id) AS Count_based_on_pay_instal FROM `TARGET
_SQL.payments`
GROUP BY payment_installments
ORDER BY Count_based_on_pay_instal DESC
```

OutPut



Row	payment_installments	Count_based_on_pay_instal
1	1	52546
2	2	12413
3	3	10461
4	4	7098
5	10	5328
6	5	5239
7	8	4268
8	6	3920

Insight- It can be seen that as number of payment installment increase the count of orders based on payment installment decreases