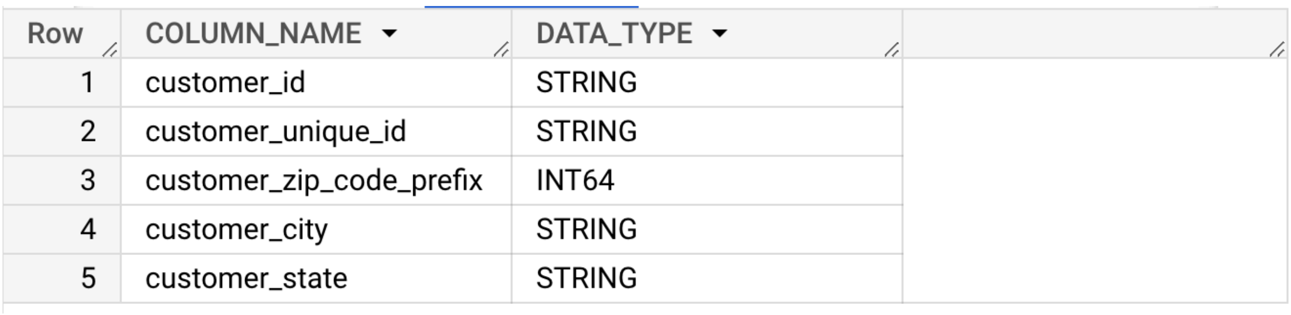
Q1. Data type of all columns in the "customers" table.

A1.

```
SELECT COLUMN_NAME, DATA_TYPE
FROM Target.INFORMATION_SCHEMA.COLUMNS
WHERE TABLE_NAME = ‘customers';
```



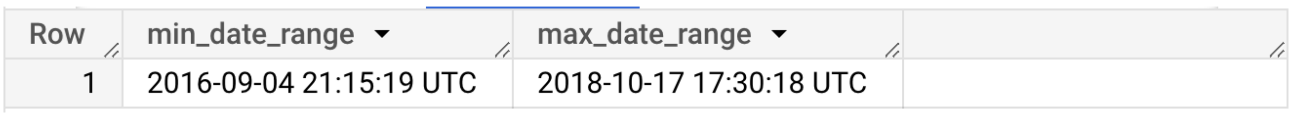
Row	COLUMN_NAME	DATA_TYPE
1	customer_id	STRING
2	customer_unique_id	STRING
3	customer_zip_code_prefix	INT64
4	customer_city	STRING
5	customer_state	STRING

INSIGHTS:
1. Where as ‘customer_zip_code_prefix’ has a data type = INT64, rest of the columns have a data type = STRING
2. Its a very useful tool when data types of columns are to be retrieved in table form

Q2. Get the time range between which the orders were placed.

A2.

```
SELECT
min(order_purchase_timestamp) as min_date_range,
max(order_purchase_timestamp) as max_date_range
FROM `profound-outlet-393214.Target.orders`;
```



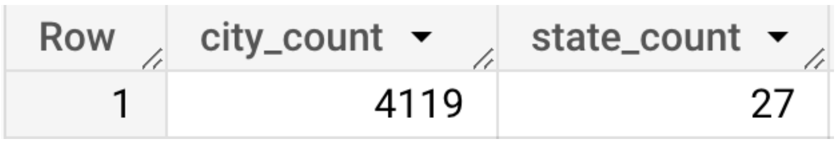
Row	min_date_range	max_date_range
1	2016-09-04 21:15:19 UTC	2018-10-17 17:30:18 UTC

INSIGHTS:
1. The time range of orders placed starts from 2016 and ends in 2018, as the case study description explains, hence the result is in accordance with the question requirement.

Q3. Count the Cities & States of customers who ordered during the given period.

A3.

```
select
count(distinct(c.customer_city)) as city_count,
count(distinct(c.customer_state)) as state_count
from Target.customers c left join `Target.orders` o on
c.customer_id = o.customer_id
where order_purchase_timestamp between '2016-09-04 21:15:19 UTC'
and '2018-10-17 17:30:18 UTC';
```



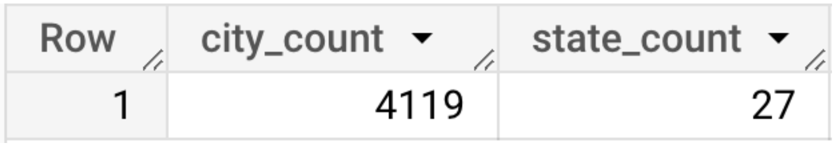
Row	city_count	state_count
1	4119	27

INSIGHTS:

1. The same result can also be achieved via:

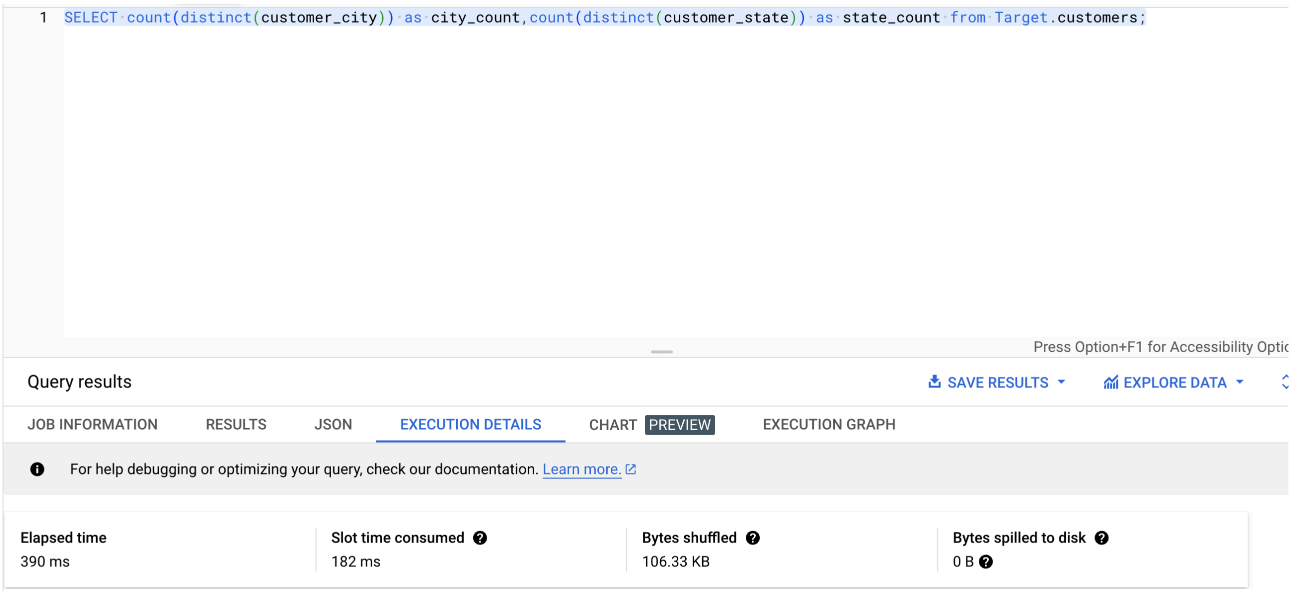
```
SELECT count(distinct(customer_city)) as city_count,
count(distinct(customer_state)) as state_count
from Target.customers;
```

Which gives the output:



Row	city_count	state_count
1	4119	27

This is not always the case, but in this case the customer_id is unique for each customer, hence there's no impact of a join or filtration. The entire data set is the answer and hence the count of cities and states.
Also a pre analysis like this can give us a chance to optimise our query, which can be seen as follows:

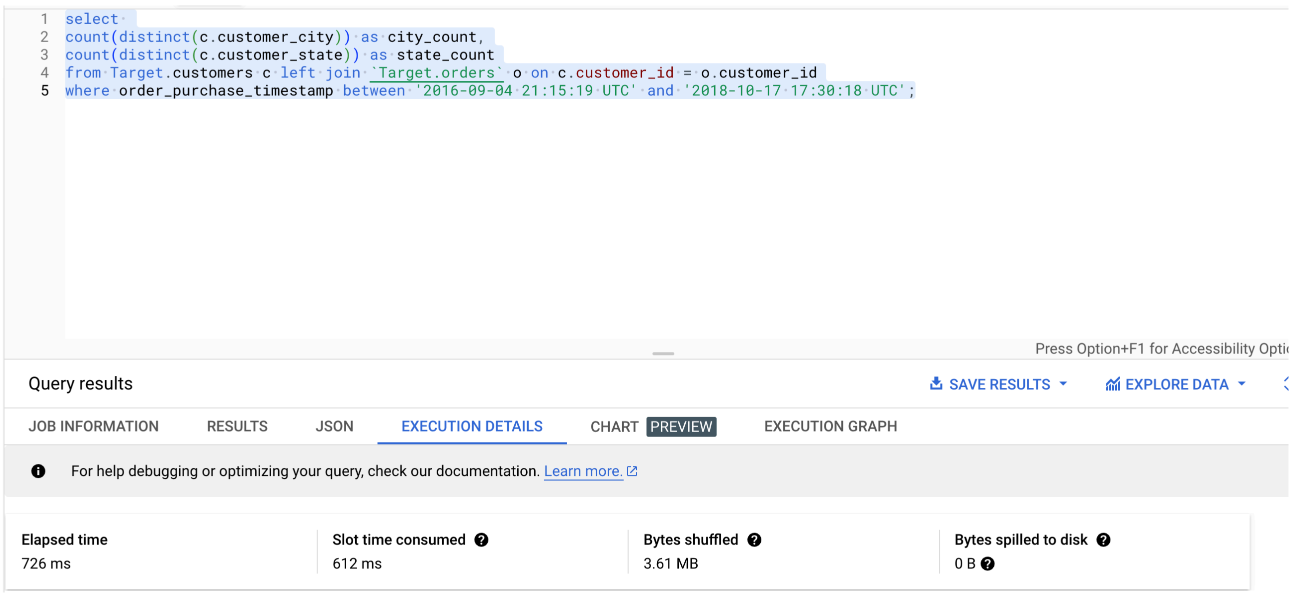


```
SELECT count(distinct(customer_city)) as city_count,count(distinct(customer_state)) as state_count from Target.customers;
```

For help debugging or optimizing your query, check our documentation. Learn more.

Elapsed time	Slot time consumed	Bytes shuffled	Bytes spilled to disk
390 ms	182 ms	106.33 KB	0 B

VS



```sql
select
count(distinct(c.customer_city)) as city_count,
count(distinct(c.customer_state)) as state_count
from Target.customers c left join `Target.orders` o on c.customer_id = o.customer_id
where order_purchase_timestamp between '2016-09-04 21:15:19 UTC' and '2018-10-17 17:30:18 UTC';
```

Press Option+F1 for Accessibility Opti

Query results

SAVE RESULTS EXPLORE DATA

JOB INFORMATION RESULTS JSON EXECUTION DETAILS CHART PREVIEW EXECUTION GRAPH

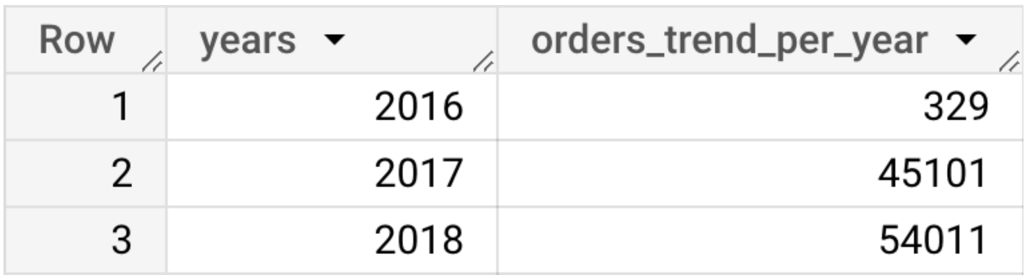
For help debugging or optimizing your query, check our documentation. Learn more.

Elapsed time	Slot time consumed	Bytes shuffled	Bytes spilled to disk
726 ms	612 ms	3.61 MB	0 B

2. In-depth Exploration:

Q1. Is there a growing trend in the no. of orders placed over the past years?

A1.
```
select years, count(*) as orders_trend_per_year
from
(SELECT
extract(year from order_purchase_timestamp) as years
FROM `profound-outlet-393214.Target.orders`) o
group by years
order by years;
```

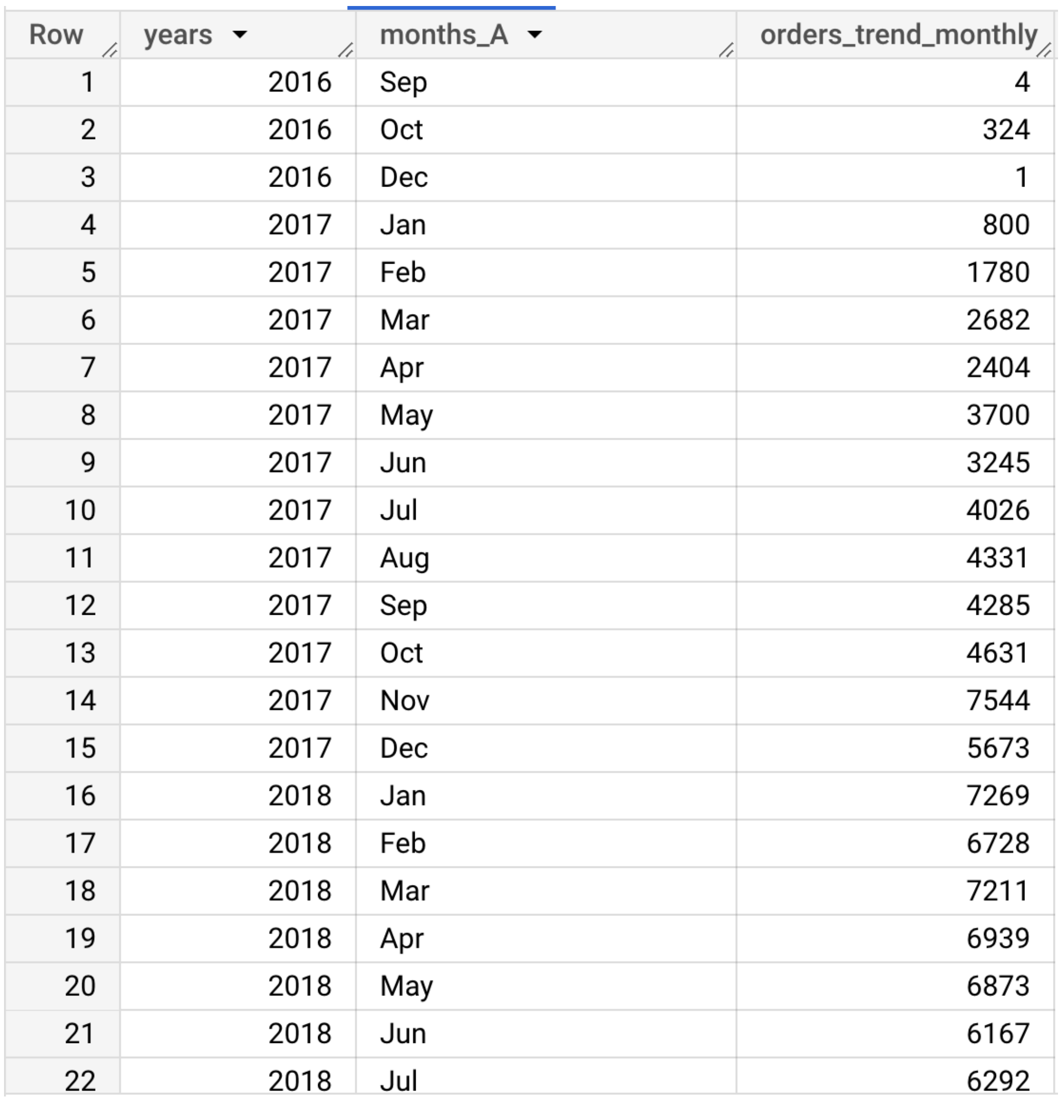


Row	years	orders_trend_per_year
1	2016	329
2	2017	45101
3	2018	54011

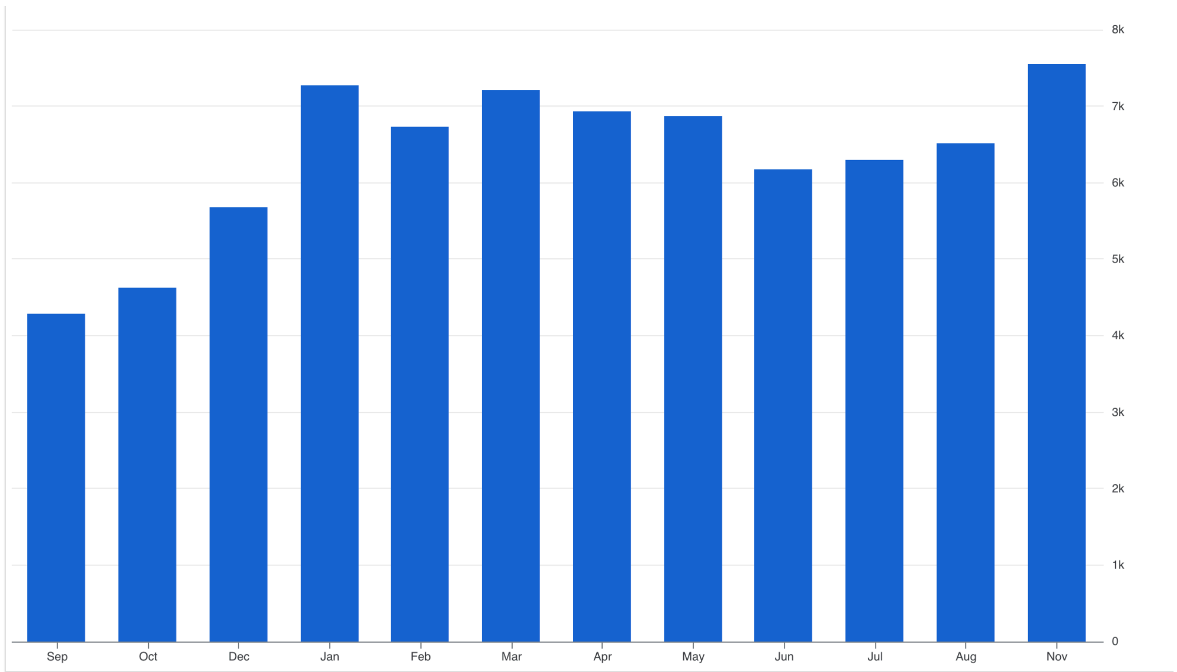
INSIGHTS:
There's Indeed an upward trend of orders ordered each year, in 2016 the number of orders are very grim, where as in 2017 and 2018, the company picked up the pace and no. of orders are quite high.

Q2. Can we see some kind of monthly seasonality in terms of the no. of orders being placed?

A2. YES
```
select
years,
months_A,
count(*) as orders_trend_monthly
from
(SELECT
extract(year from order_purchase_timestamp) as years,
extract(month from order_purchase_timestamp) as
months,
FORMAT_DATETIME("%b", order_purchase_timestamp ) as
months_A
FROM `profound-outlet-393214.Target.orders`) o
group by years,months,months_A
order by years,months;
```



Row	years	months_A	orders_trend_monthly
1	2016	Sep	4
2	2016	Oct	324
3	2016	Dec	1
4	2017	Jan	800
5	2017	Feb	1780
6	2017	Mar	2682
7	2017	Apr	2404
8	2017	May	3700
9	2017	Jun	3245
10	2017	Jul	4026
11	2017	Aug	4331
12	2017	Sep	4285
13	2017	Oct	4631
14	2017	Nov	7544
15	2017	Dec	5673
16	2018	Jan	7269
17	2018	Feb	6728
18	2018	Mar	7211
19	2018	Apr	6939
20	2018	May	6873
21	2018	Jun	6167
22	2018	Jul	6292

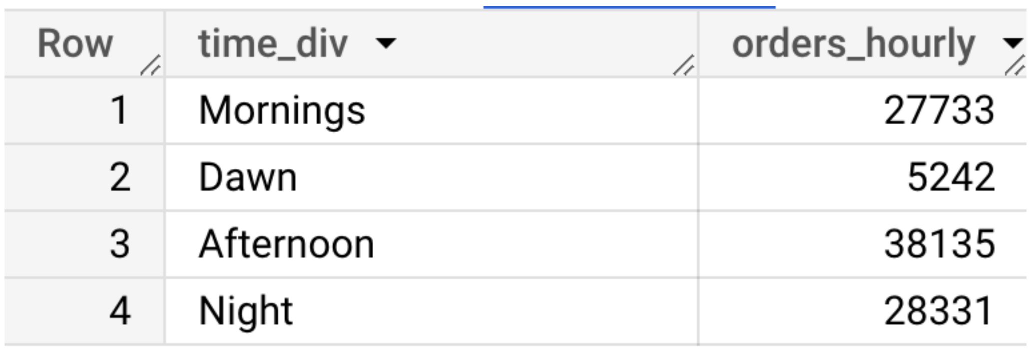


INSIGHTS:
The number of orders dip during the month of September, October and December and its maximum January and November

Q3. During what time of the day, do the Brazilian customers mostly place their orders? (Dawn, Morning, Afternoon or Night)

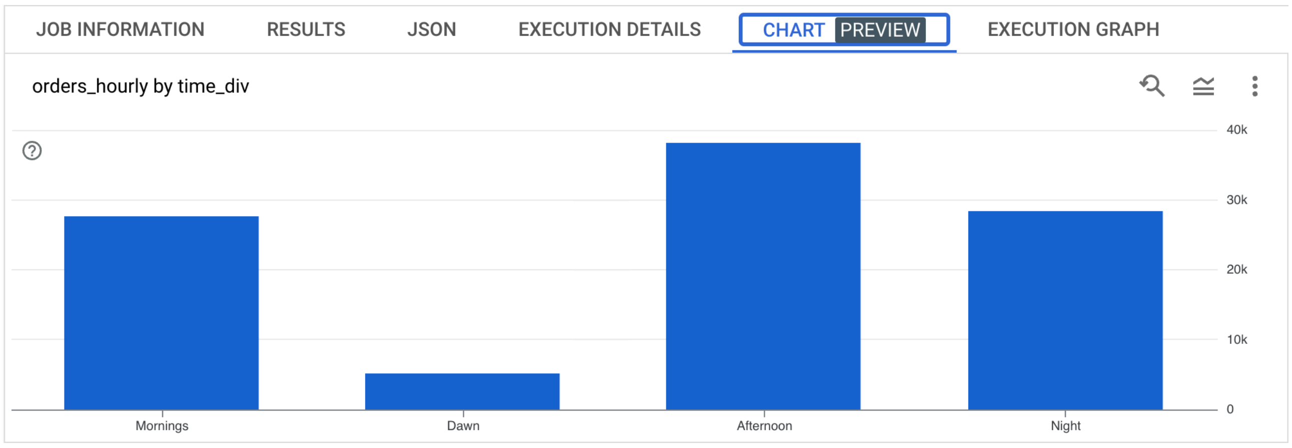
- 0-6 hrs : Dawn
- 7-12 hrs : Mornings
- 13-18 hrs : Afternoon
- 19-23 hrs : Night

A3.
```
select time_div, count(*) as orders_hourly from
(select
*,
case
  when times between '00:00:00' and '06:59:59' then 'Dawn'
  when times between '07:00:00' and '12:59:59' then 'Mornings'
  when times between '13:00:00' and '18:59:59' then 'Afternoon'
  when times between '19:00:00' and '23:59:59' then 'Night'
end as time_div
from
(SELECT
extract(year from order_purchase_timestamp) as years,
extract(month from order_purchase_timestamp) as months,
extract(time from order_purchase_timestamp) as times,
FORMAT_DATETIME("%b", order_purchase_timestamp ) as months_A
FROM `profound-outlet-393214.Target.orders`) o) o2
group by time_div;
```



Row	time_div	orders_hourly
1	Mornings	27733
2	Dawn	5242
3	Afternoon	38135
4	Night	28331

INSIGHTS:
As clearly seen in the Bar Graph below, Afternoon is the peak time when people of Brazil would place their orders and during Dawn, it's the minimum, which is indeed very interesting.

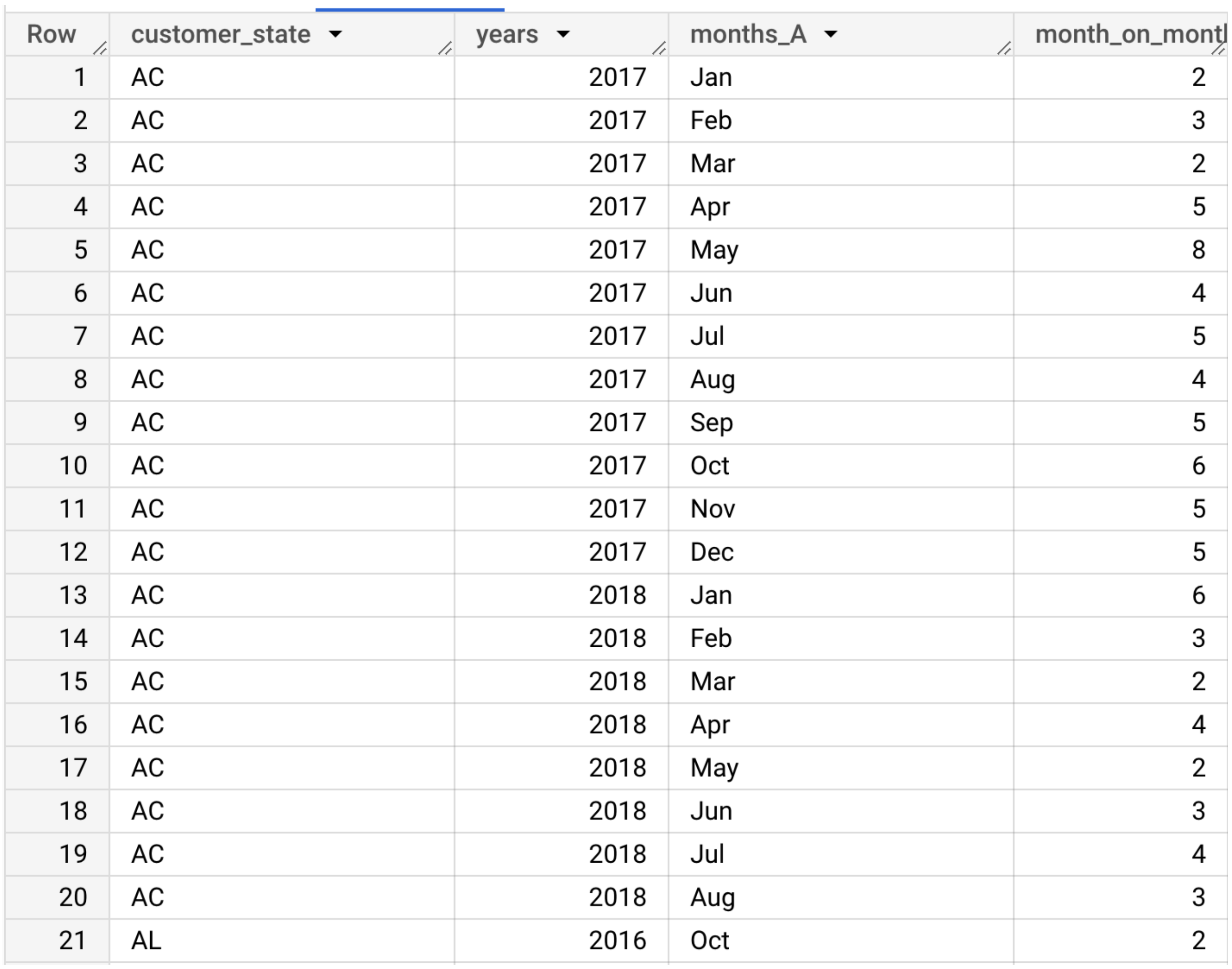


Evolution of E-commerce orders in the Brazil region:
Q1. Get the month on month no. of orders placed in each state.

A2.

```
with mom as
(select customer_state,
years,
months,
months_A,
count(*) as month_on_month
from
(select c.customer_state,
extract(year from o.order_purchase_timestamp) as years,
extract(month from o.order_purchase_timestamp) as months,
FORMAT_DATETIME("%b", o.order_purchase_timestamp ) as months_A
from Target.customers c left join Target.orders o
on c.customer_id = o.customer_id
order by customer_state, years, months) oc
group by customer_state,years,months,months_A
order by customer_state,years,months)

select customer_state,
years,
months_A,
month_on_month
from mom;
```



Row	customer_state	years	months_A	month_on_montl
1	AC	2017	Jan	2
2	AC	2017	Feb	3
3	AC	2017	Mar	2
4	AC	2017	Apr	5
5	AC	2017	May	8
6	AC	2017	Jun	4
7	AC	2017	Jul	5
8	AC	2017	Aug	4
9	AC	2017	Sep	5
10	AC	2017	Oct	6
11	AC	2017	Nov	5
12	AC	2017	Dec	5
13	AC	2018	Jan	6
14	AC	2018	Feb	3
15	AC	2018	Mar	2
16	AC	2018	Apr	4
17	AC	2018	May	2
18	AC	2018	Jun	3
19	AC	2018	Jul	4
20	AC	2018	Aug	3
21	AL	2016	Oct	2

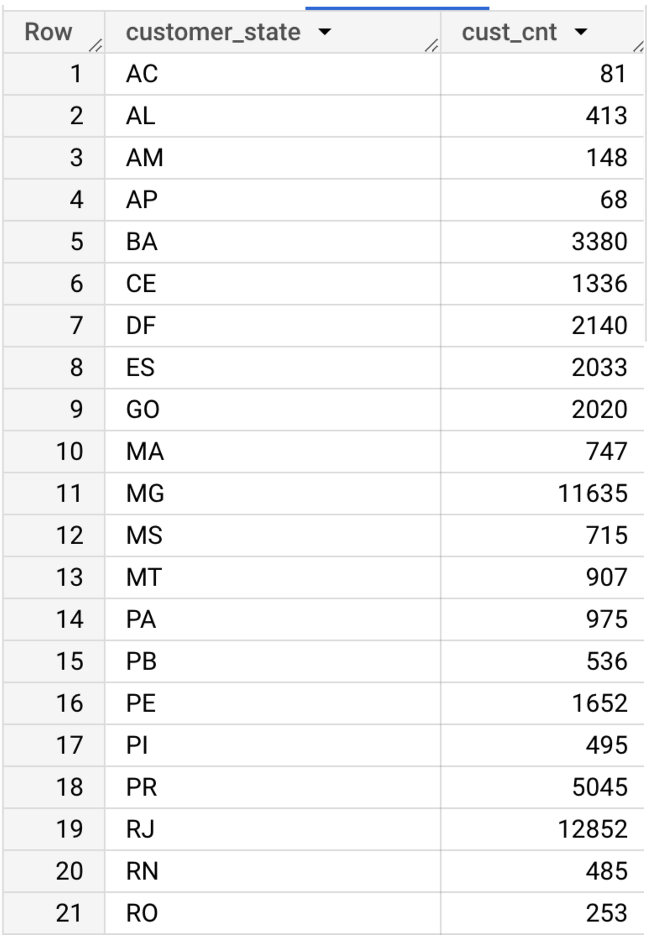
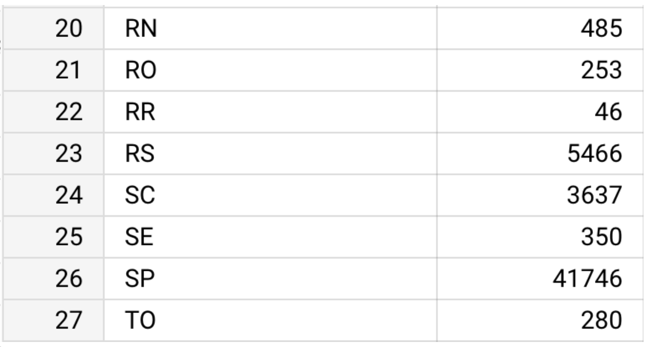
```
INSIGHTS:
The state from which there are least number of orders is AC and
the state from which there are maximum number of orders is SP.
Talking about month on month division state wise, in State AC,
there are hardly any orders all through 2017 and 2018, order
number ranging from 2-8,
Where as for the state SP, the orders range from 2-3253 from
2016-2018.
```

Q2.How are the customers distributed across all the states?

A2.

```
select
customer_state,
count(*) as cust_cnt
from Target.customers
group by customer_state
order by customer_state;
```

INSIGHTS:
Where as this could have been ordered by the number of Counts of customers in asc or desc order,
I present my answer like this, alphabetically ordered by state names, which tells us,
The maximum number of customers are from state SP, ie, 41746, which also coincides with maximum number of orders from the same state and minimum number of customers are from the state RR, ie, 46, which is actually quite low. Company needs better penetration there, and other states like, AC and AP.

Row	customer_state	cust_cnt
1	AC	81
2	AL	413
3	AM	148
4	AP	68
5	BA	3380
6	CE	1336
7	DF	2140
8	ES	2033
9	GO	2020
10	MA	747
11	MG	11635
12	MS	715
13	MT	907
14	PA	975
15	PB	536
16	PE	1652
17	PI	495
18	PR	5045
19	RJ	12852
20	RN	485
21	RO	253
22	RR	46
23	RS	5466
24	SC	3637
25	SE	350
26	SP	41746
27	TO	280

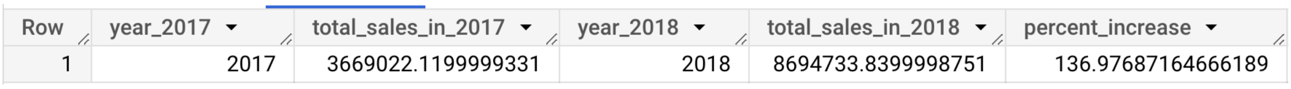
Impact on Economy: Analyze the money movement by e-commerce by looking at order prices, freight and others.

Q1.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.

A1.

```
with p_inc as
(select o.order_id,
o.order_purchase_timestamp,
extract(year from o.order_purchase_timestamp) as yr,
extract(month from o.order_purchase_timestamp) as m,
p.p_value
from `Target.orders` o inner join
(SELECT
order_id,
sum(payment_value) as p_value
FROM `profound-outlet-393214.Target.payments`
group by order_id) p
on o.order_id = p.order_id)

select
f.yr as year_2017,
f.p_val as total_sales_in_2017,
b.yr as year_2018,
b.p_val as total_sales_in_2018,
(((b.p_val-f.p_val)/f.p_val)*100)  as percent_increase
from
(select yr,sum(p_value) as p_val from p_inc  where yr = 2017 and m
between 1 and 8 group by yr) f,
(select yr,sum(p_value) as p_val from p_inc where yr = 2018 and m
between 1 and 8 group by yr) b;
```



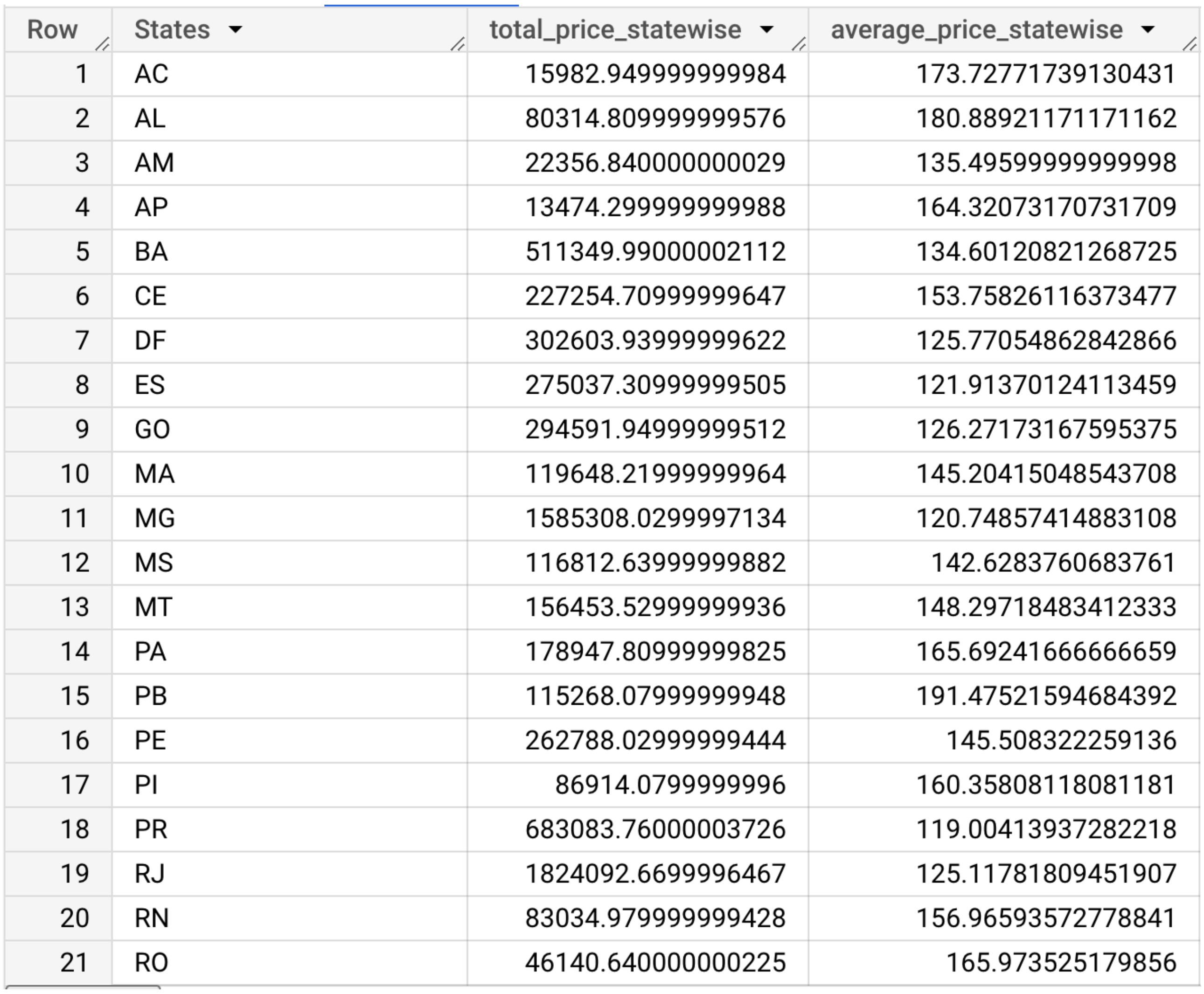
Row	year_2017	total_sales_in_2017	year_2018	total_sales_in_2018	percent_increase
1	2017	3669022.1199999331	2018	8694733.8399998751	136.97687164666189

```
INSIGHTS:
The percent increase from the year 2017 to 2018 in sales
concerning just the months January to August is a whooping 136%.
Which is tremendous.
```

Q2. Calculate the Total & Average value of order price for each state.

A2.

```sql
SELECT c.customer_state as States,
sum(oi.price) as total_price_statewise,
avg(oi.price) as average_price_statewise
FROM `profound-outlet-393214.Target.order_items` oi
left join `profound-outlet-393214.Target.orders` o
on oi.order_id = o.order_id
left join `profound-outlet-393214.Target.customers` c
on o.customer_id = c.customer_id
group by c.customer_state
order by c.customer_state;
```



Row	States	total_price_statewise	average_price_statewise
1	AC	15982.949999999984	173.72771739130431
2	AL	80314.809999999576	180.88921171171162
3	AM	22356.840000000029	135.49599999999998
4	AP	13474.299999999988	164.32073170731709
5	BA	511349.99000002112	134.60120821268725
6	CE	227254.70999999647	153.75826116373477
7	DF	302603.93999999622	125.77054862842866
8	ES	275037.30999999505	121.91370124113459
9	GO	294591.94999999512	126.27173167595375
10	MA	119648.21999999964	145.20415048543708
11	MG	1585308.0299997134	120.74857414883108
12	MS	116812.63999999882	142.6283760683761
13	MT	156453.52999999936	148.29718483412333
14	PA	178947.80999999825	165.69241666666659
15	PB	115268.07999999948	191.47521594684392
16	PE	262788.02999999444	145.508322259136
17	PI	86914.0799999996	160.35808118081181
18	PR	683083.76000003726	119.00413937282218
19	RJ	1824092.6699996467	125.11781809451907
20	RN	83034.979999999428	156.96593572778841
21	RO	46140.640000000225	165.973525179856

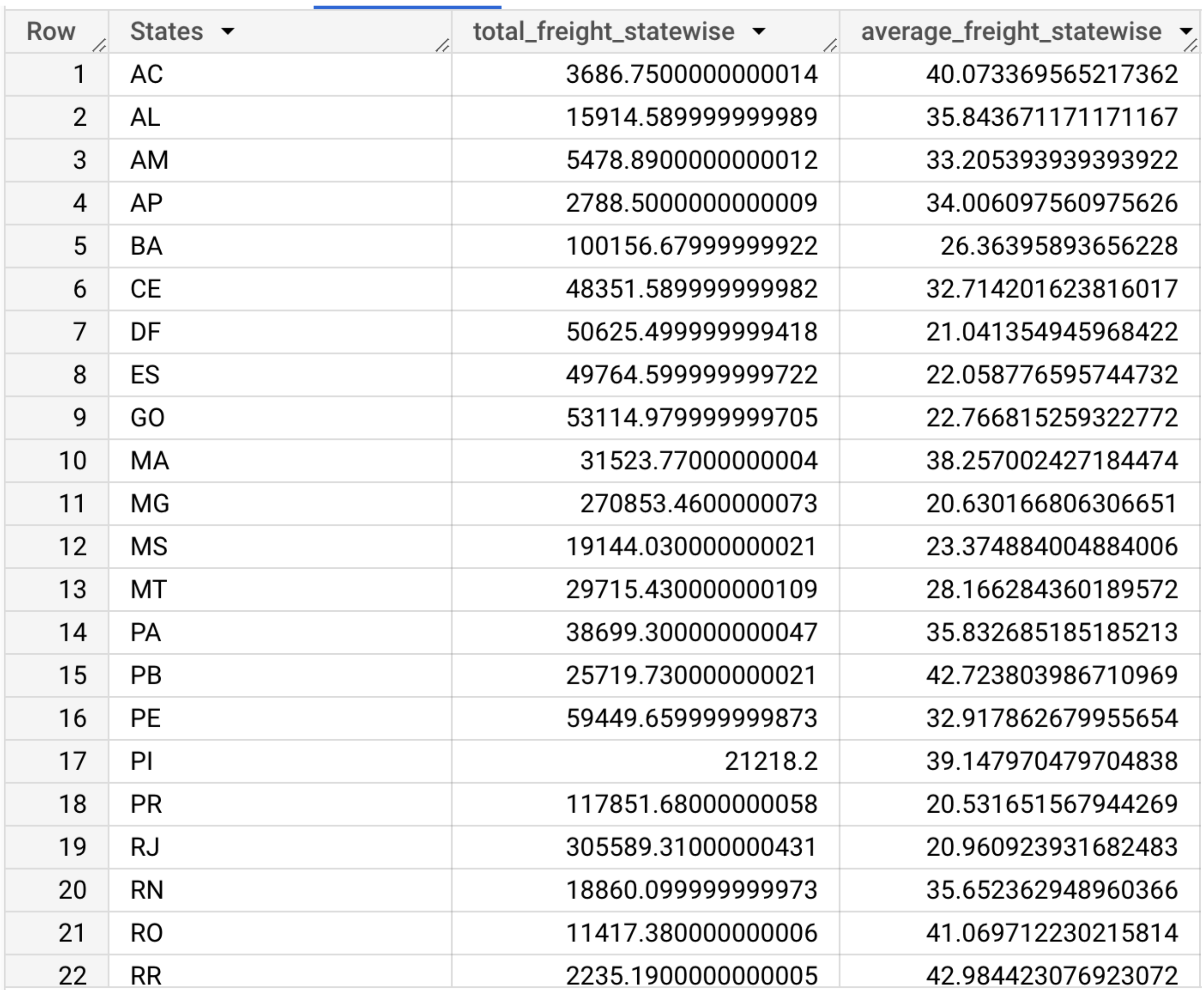
INSIGHTS:
Here is the total price and average price state wise, which can be shown with numbers unto two decimal places as well, but as it has been not mentioned so im assuming the manager shall work with the

data in this form otherwise, I could ceil it, floor it or round it off.

Q3. Calculate the Total & Average value of order freight for each state.

A3.

```
SELECT c.customer_state as States,
sum(oi.freight_value) as total_freight_statewise,
avg(oi.freight_value) as average_freight_statewise
FROM `profound-outlet-393214.Target.order_items` oi
left join `profound-outlet-393214.Target.orders` o
on oi.order_id = o.order_id
left join `profound-outlet-393214.Target.customers` c
on o.customer_id = c.customer_id
group by c.customer_state
order by c.customer_state;
```

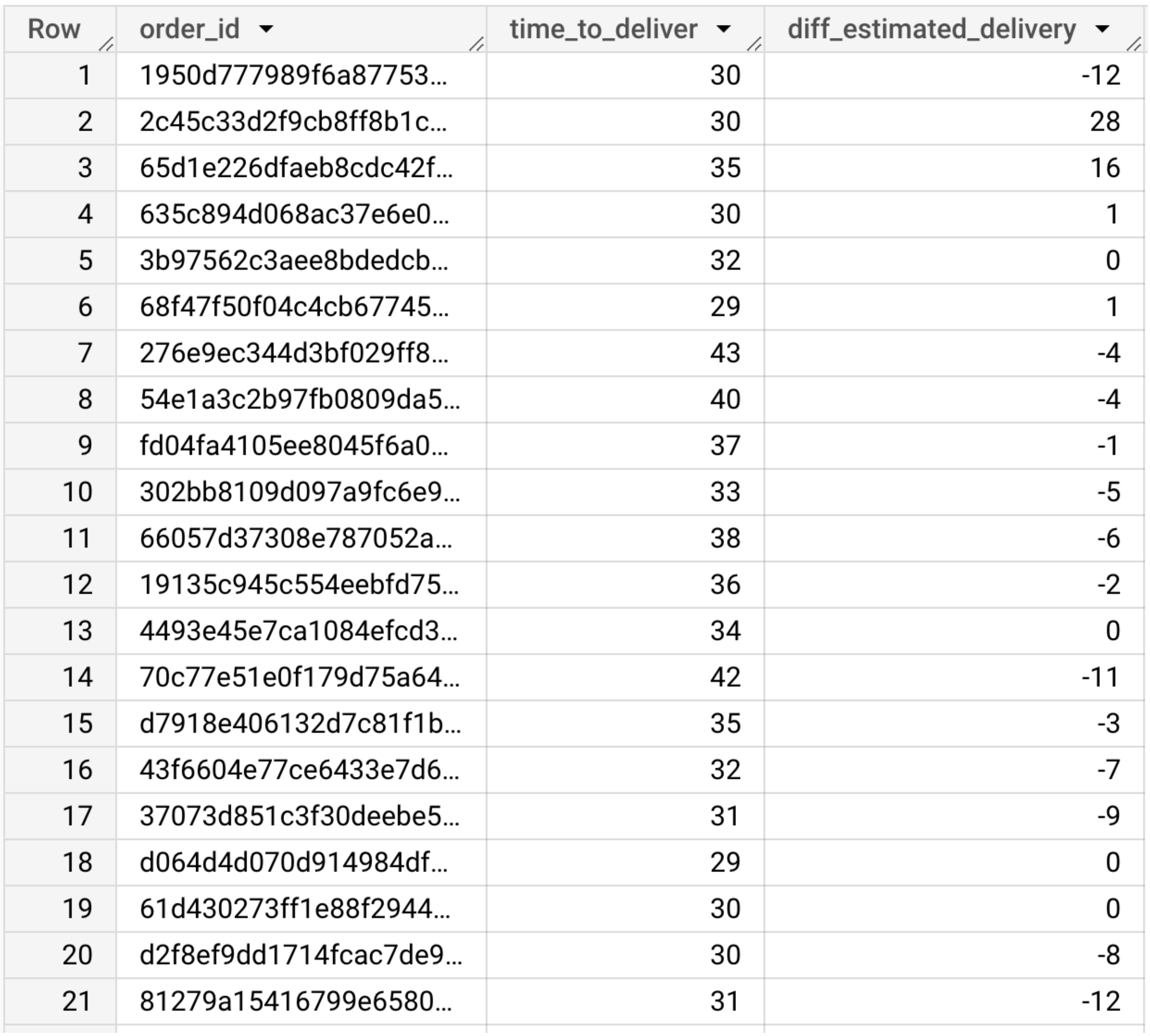


Row	States	total_freight_statewise	average_freight_statewise
1	AC	3686.7500000000014	40.073369565217362
2	AL	15914.589999999989	35.843671171171167
3	AM	5478.8900000000012	33.205393939393922
4	AP	2788.5000000000009	34.006097560975626
5	BA	100156.67999999922	26.36395893656228
6	CE	48351.589999999982	32.714201623816017
7	DF	50625.499999999418	21.041354945968422
8	ES	49764.599999999722	22.058776595744732
9	GO	53114.979999999705	22.766815259322772
10	MA	31523.77000000004	38.257002427184474
11	MG	270853.4600000073	20.630166806306651
12	MS	19144.030000000021	23.374884004884006
13	MT	29715.430000000109	28.166284360189572
14	PA	38699.300000000047	35.832685185185213
15	PB	25719.730000000021	42.723803986710969
16	PE	59449.659999999873	32.917862679955654
17	PI	21218.2	39.147970479704838
18	PR	117851.68000000058	20.531651567944269
19	RJ	305589.31000000431	20.960923931682483
20	RN	18860.099999999973	35.652362948960366
21	RO	11417.380000000006	41.069712230215814
22	RR	2235.1900000000005	42.984423076923072

Analysis based on sales, freight and delivery time.

Q1. 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.

A1.
```
SELECT order_id,
date_diff(order_delivered_customer_date,order_purchase_timestamp,d
ay) as time_to_deliver,
date_diff(order_estimated_delivery_date,order_delivered_customer_d
ate,day) as diff_estimated_delivery
 FROM `profound-outlet-393214.Target.orders`
```



Row	order_id	time_to_deliver	diff_estimated_delivery
1	1950d777989f6a87753...	30	-12
2	2c45c33d2f9cb8ff8b1c...	30	28
3	65d1e226dfaeb8cdc42f...	35	16
4	635c894d068ac37e6e0...	30	1
5	3b97562c3aee8bdedcb...	32	0
6	68f47f50f04c4cb67745...	29	1
7	276e9ec344d3bf029ff8...	43	-4
8	54e1a3c2b97fb0809da5...	40	-4
9	fd04fa4105ee8045f6a0...	37	-1
10	302bb8109d097a9fc6e9...	33	-5
11	66057d37308e787052a...	38	-6
12	19135c945c554eebfd75...	36	-2
13	4493e45e7ca1084efcd3...	34	0
14	70c77e51e0f179d75a64...	42	-11
15	d7918e406132d7c81f1b...	35	-3
16	43f6604e77ce6433e7d6...	32	-7
17	37073d851c3f30deebe5...	31	-9
18	d064d4d070d914984df...	29	0
19	61d430273ff1e88f2944...	30	0
20	d2f8ef9dd1714fcac7de9...	30	-8
21	81279a15416799e6580...	31	-12

INSIGHTS:
It takes approx 1 month to deliver a product by Target,
The negative sign in diff_estimated_delivery depicts that the actual delivery was delayed by respective number of days from the estimated delivery.

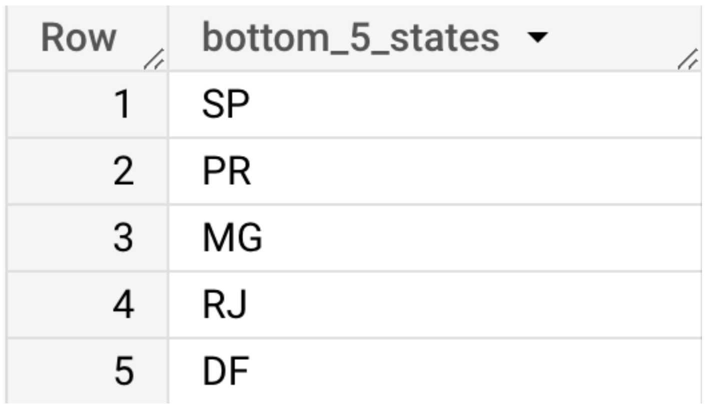
Q2. Find out the top 5 states with the highest & lowest average freight value.

A2.

For bottom 5 States:

```
with fr as
(SELECT c.customer_state as States,
sum(oi.freight_value) as total_freight_statewise,
avg(oi.freight_value) as average_freight_statewise
FROM `profound-outlet-393214.Target.order_items` oi
left join `profound-outlet-393214.Target.orders` o
on oi.order_id = o.order_id
left join `profound-outlet-393214.Target.customers` c
on o.customer_id = c.customer_id
group by c.customer_state
order by c.customer_state)

select States as bottom_5_states from fr
order by fr.average_freight_statewise
limit 5;
```

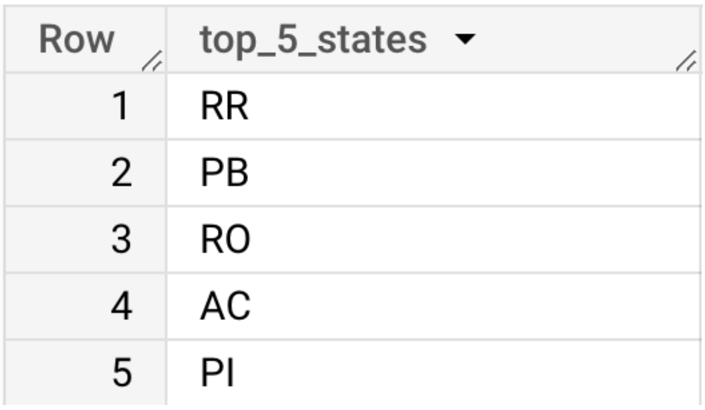


Row	bottom_5_states
1	SP
2	PR
3	MG
4	RJ
5	DF

For top 5 States:

```
with fr as
(SELECT c.customer_state as States,
sum(oi.freight_value) as total_freight_statewise,
avg(oi.freight_value) as average_freight_statewise
FROM `profound-outlet-393214.Target.order_items` oi
left join `profound-outlet-393214.Target.orders` o
on oi.order_id = o.order_id
left join `profound-outlet-393214.Target.customers` c
on o.customer_id = c.customer_id
group by c.customer_state
order by c.customer_state)

select States as top_5_states from fr
order by fr.average_freight_statewise desc
limit 5;
```



Row	top_5_states
1	RR
2	PB
3	RO
4	AC
5	PI

INSIGHT: The states which highest and lowest average freight value are listed above

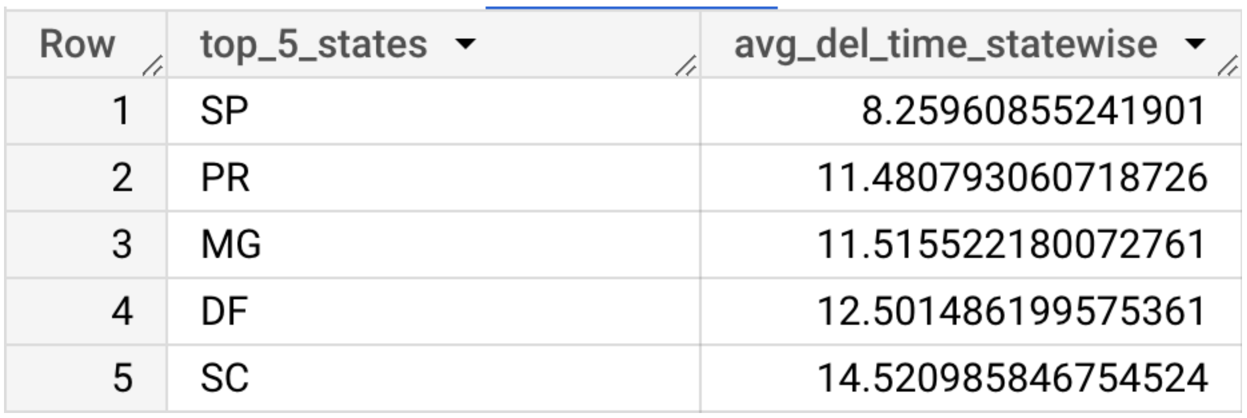
Q3. Find out the top 5 states with the highest & lowest average delivery time.

A3.

<u>For top 5 States:</u>

```
with fr as
(SELECT c.customer_state as States,
date_diff(o.order_delivered_customer_date,o.order_purchase_timesta
mp,day) as time_to_deliver
FROM `profound-outlet-393214.Target.order_items` oi
left join `profound-outlet-393214.Target.orders` o
on oi.order_id = o.order_id
left join `profound-outlet-393214.Target.customers` c
on o.customer_id = c.customer_id
order by c.customer_state)

select States as top_5_states,
avg(time_to_deliver) as avg_del_time_statewise from fr
group by States
order by avg_del_time_statewise
limit 5;
```

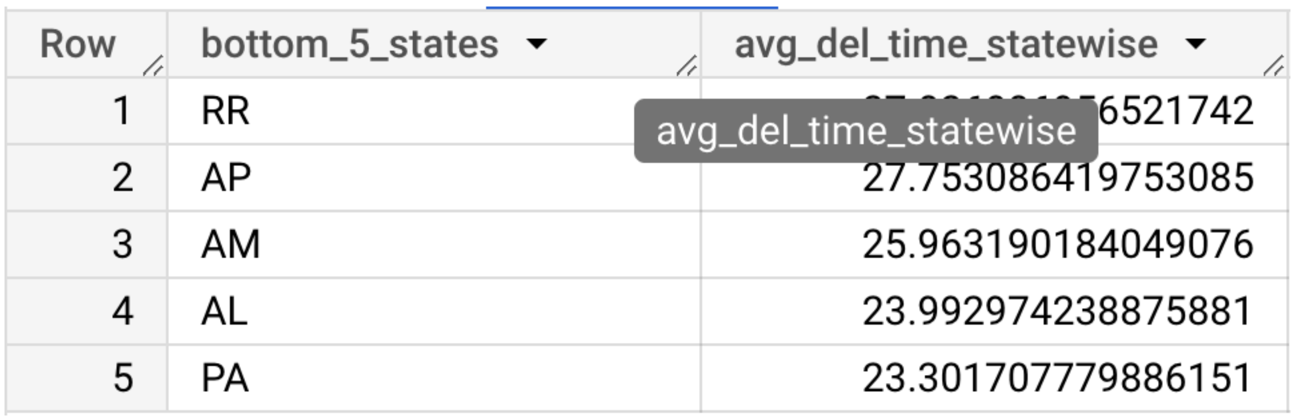


Row	top_5_states	avg_del_time_statewise
1	SP	8.25960855241901
2	PR	11.480793060718726
3	MG	11.515522180072761
4	DF	12.501486199575361
5	SC	14.520985846754524

For bottom 5 States:

```
with fr as
(SELECT c.customer_state as States,
date_diff(o.order_delivered_customer_date,o.order_purchase_timesta
mp,day) as time_to_deliver
FROM `profound-outlet-393214.Target.order_items` oi
left join `profound-outlet-393214.Target.orders` o
on oi.order_id = o.order_id
left join `profound-outlet-393214.Target.customers` c
on o.customer_id = c.customer_id
order by c.customer_state)

select States as bottom_5_states,
avg(time_to_deliver) as avg_del_time_statewise from fr
group by States
order by avg_del_time_statewise desc
limit 5;
```


avg_del_time_statewise

Row	bottom_5_states	avg_del_time_statewise
1	RR	[illegible]6521742
2	AP	27.753086419753085
3	AM	25.963190184049076
4	AL	23.992974238875881
5	PA	23.301707779886151

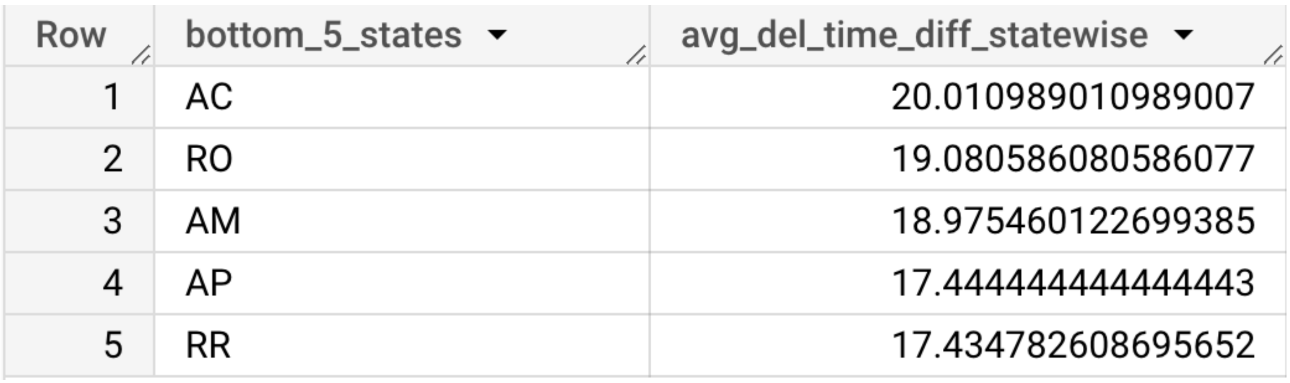
INSIGHT: The states which highest and lowest average delivery time are listed above, what is interesting though is that, the state having the lowest avg. delivery time is at the top and vice versa.

Q4. Find out the top 5 states where the order delivery is really fast as compared to the estimated date of delivery.

A4.

```
with fr as
(SELECT c.customer_state as States,
date_diff(o.order_estimated_delivery_date,o.order_delivered_custom
er_date,day) as diff_estimated_delivery
FROM `profound-outlet-393214.Target.order_items` oi
left join `profound-outlet-393214.Target.orders` o
on oi.order_id = o.order_id
left join `profound-outlet-393214.Target.customers` c
on o.customer_id = c.customer_id
order by c.customer_state)

select States as bottom_5_states,
avg(diff_estimated_delivery) as avg_del_time_diff_statewise from
fr
group by States
order by avg_del_time_diff_statewise desc
limit 5;
```



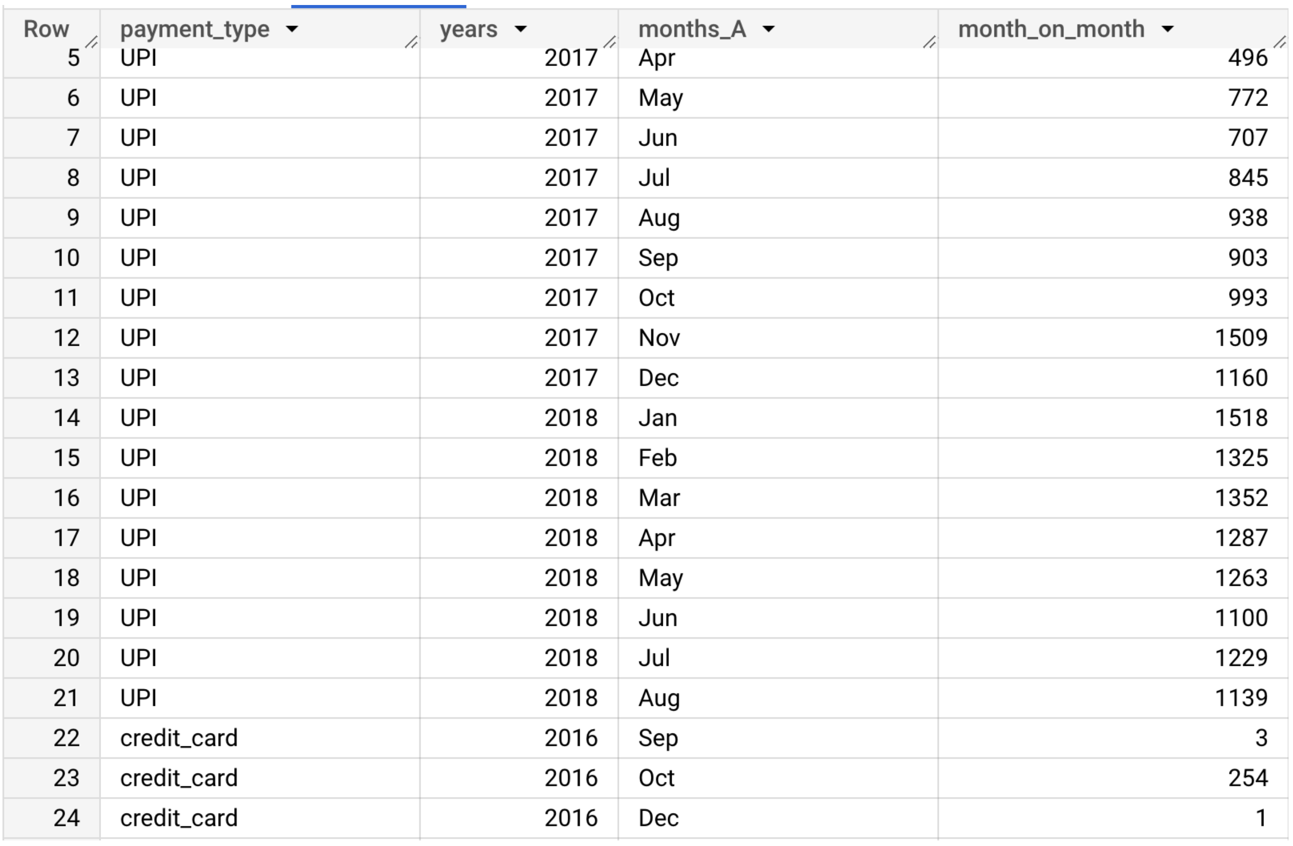
Row	bottom_5_states	avg_del_time_diff_statewise
1	AC	20.010989010989007
2	RO	19.080586080586077
3	AM	18.975460122699385
4	AP	17.444444444444443
5	RR	17.434782608695652

INSIGHTS: As deliveries in state AC are approx 20 days early than the estimated deliveries hence its at the top.

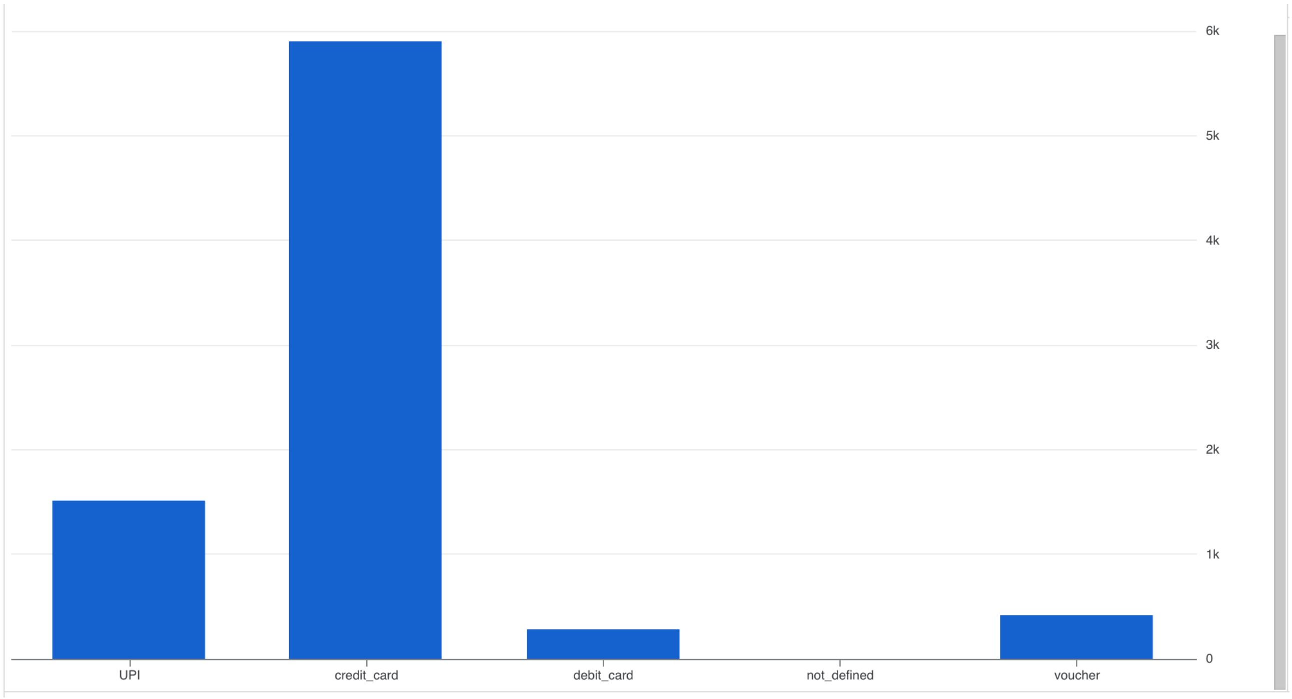
Analysis based on the payments:
Q1. Find the month on month no. of orders placed using different payment types.

```
A1.
with mom as
(select payment_type,
years,
months,
months_A,
count(*) as month_on_month
from
(select p.payment_type,
extract(year from o.order_purchase_timestamp) as years,
extract(month from o.order_purchase_timestamp) as months,
FORMAT_DATETIME("%b", o.order_purchase_timestamp ) as months_A
from Target.payments p inner join Target.orders o
on p.order_id = o.order_id) op
group by payment_type,years,months,months_A
order by payment_type,years,months)

select payment_type,
years,
months_A,
month_on_month
from mom;
```



Row	payment_type	years	months_A	month_on_month
5	UPI	2017	Apr	496
6	UPI	2017	May	772
7	UPI	2017	Jun	707
8	UPI	2017	Jul	845
9	UPI	2017	Aug	938
10	UPI	2017	Sep	903
11	UPI	2017	Oct	993
12	UPI	2017	Nov	1509
13	UPI	2017	Dec	1160
14	UPI	2018	Jan	1518
15	UPI	2018	Feb	1325
16	UPI	2018	Mar	1352
17	UPI	2018	Apr	1287
18	UPI	2018	May	1263
19	UPI	2018	Jun	1100
20	UPI	2018	Jul	1229
21	UPI	2018	Aug	1139
22	credit_card	2016	Sep	3
23	credit_card	2016	Oct	254
24	credit_card	2016	Dec	1



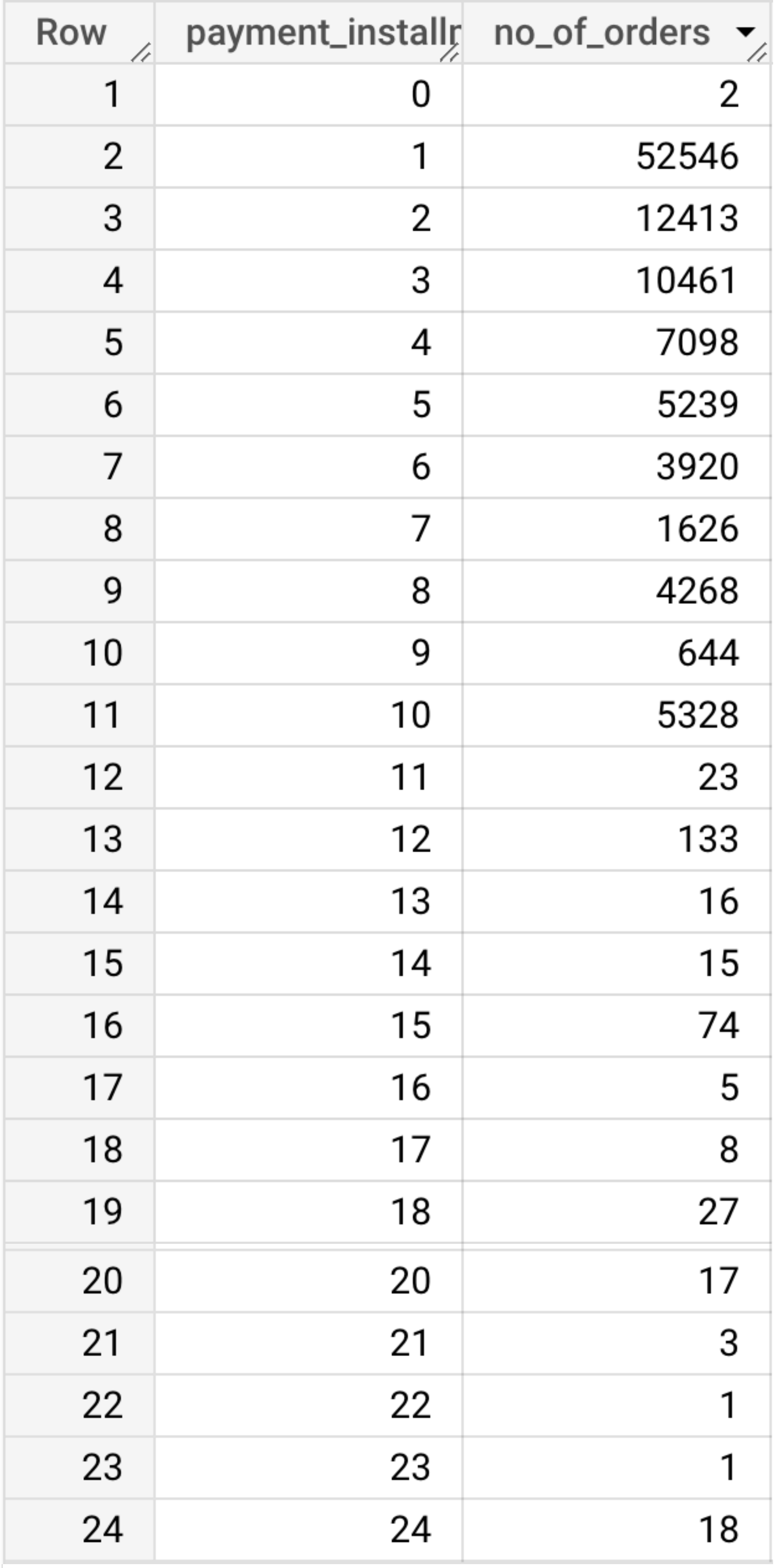
INSIGHTS:
As the bar graph is depicting, the payment type in which there are least number of orders paid is Debit card and the payment type in which there are maximum number of orders made is Credit card.
Talking about month on month division payment type wise, in Debit card method, there are very meagre number of orders all through 2016 and 2018, order number ranging from 2-277,
Where as for the payment method Credit card, the orders range from 1 in Dec 2016, to 5897 in Nov 2017, to ultimately 4985 orders in aug 2018.

Q2. Find the no. of orders placed on the basis of the payment installments that have been paid.

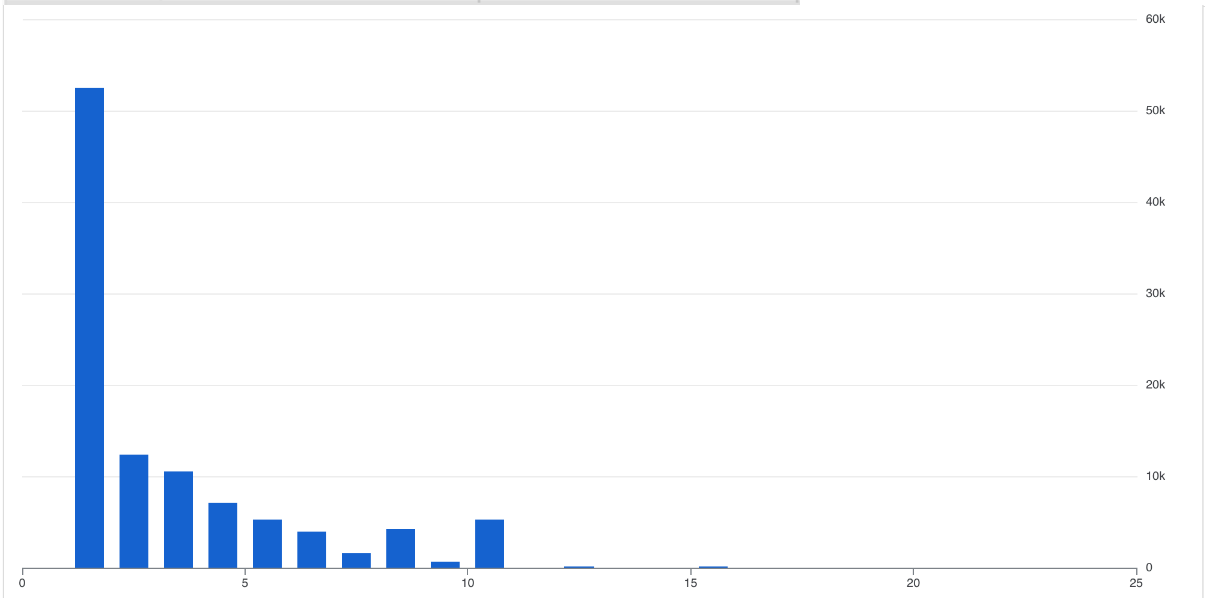
A2.

```
SELECT payment_installments,
count(*) as no_of_orders
FROM `profound-outlet-393214.Target.payments`
group by payment_installments
```

INSIGHTS:
As the output and the bar Graph are depicting, based on payment instalments, the number of orders are minimum with zero instalments, ie, just 2 orders, otherwise, maximum no of orders are being processed with 1 instalment as option, ie, 52,546 orders. The trend is almost swift decrease in number of orders as the number of instalments increase, but something interesting also happens when it comes to 10 instalments, suddenly the numer. Of orders increases to 5328, from 644 which corresponds to 9 instalments.



Row	payment_installr	no_of_orders
1	0	2
2	1	52546
3	2	12413
4	3	10461
5	4	7098
6	5	5239
7	6	3920
8	7	1626
9	8	4268
10	9	644
11	10	5328
12	11	23
13	12	133
14	13	16
15	14	15
16	15	74
17	16	5
18	17	8
19	18	27
20	20	17
21	21	3
22	22	1
23	23	1
24	24	18


60k
50k
40k
30k
20k
10k
0
0
5
10
15
20
25