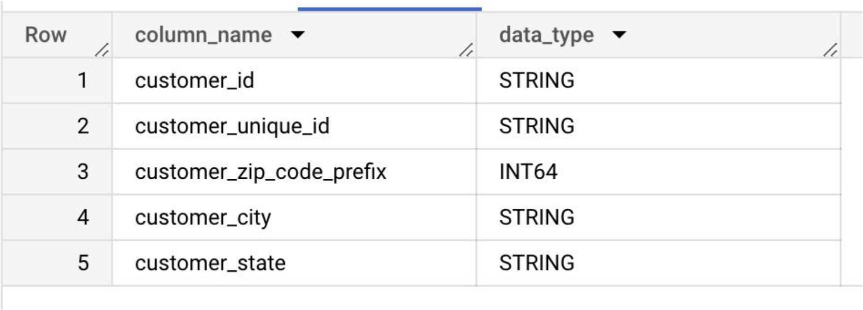
1. Data type of all columns in the "customers" table.

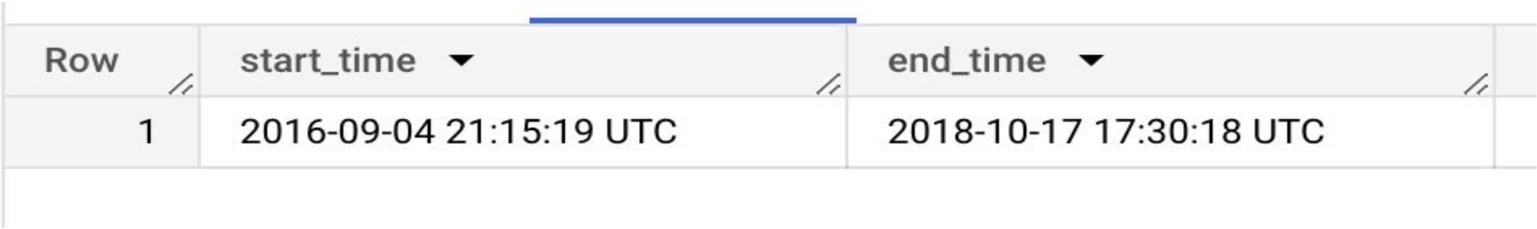
```
SELECT
 column_name
 ,data_type
FROM
 dsml-may-23-beg.Target_Case_Study.INFORMATION_SCHEMA.COLUMNS
WHERE
 table_name = 'customers'
```



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

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

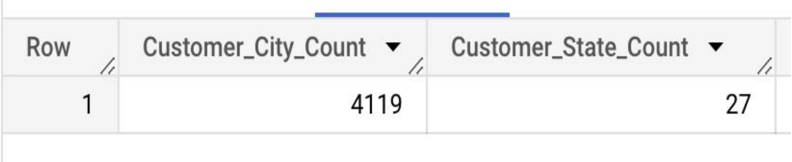
```
SELECT
  MIN(order_purchase_timestamp) as start_time
 ,MAX(order_purchase_timestamp) AS end_time
FROM
  `Target_Case_Study.orders`
```



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

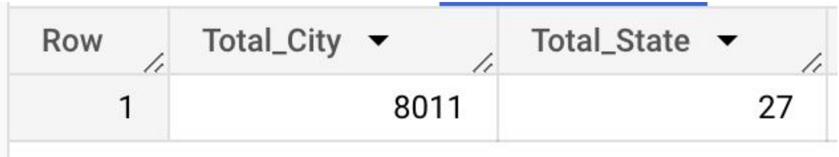
1. Count the number of Cities and States in our dataset.

```
SELECT
  COUNT(DISTINCT customer_city) AS Customer_City_Count
 ,COUNT(DISTINCT customer_state) AS Customer_State_Count
FROM `Target_Case_Study.customers`
```



Row	Customer_City_Count	Customer_State_Count
1	4119	27

```
SELECT
  COUNT(DISTINCT geolocation_city)   AS Total_City
 ,COUNT(DISTINCT geolocation_state)  AS Total_State
FROM `Target_Case_Study.geolocation`
```



Row	Total_City	Total_State
1	8011	27

Analysis:

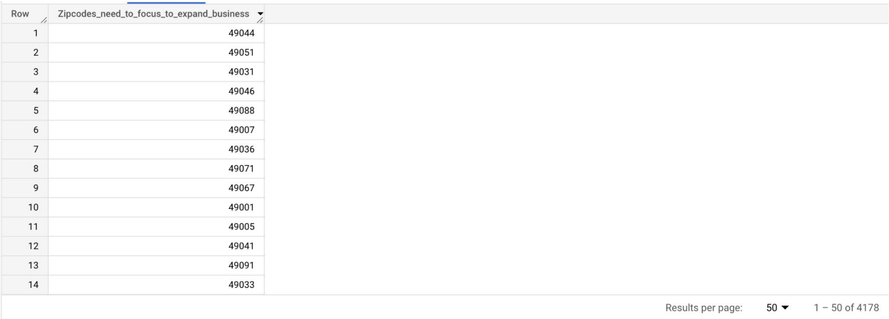
There are total **27** states customers are present in all **27** states.

There are total **8011** cities but customers are present in only **4119**

Zip Codes where Customers are not present

```
SELECT
  Distinct geolocation_zip_code_prefix AS
Zipcodes_need_to_focus_to_expand_business
FROM `Target_Case_Study.geolocation` Geo
LEFT JOIN `Target_Case_Study.customers` C ON
C.customer_zip_code_prefix = Geo.geolocation_zip_code_prefix
where customer_zip_code_prefix is null
```

There are total 4178 zip codes where customers are not present

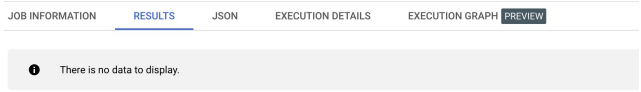


Row	Zipcodes_need_to_focus_to_expand_business
1	49044
2	49051
3	49031
4	49046
5	49088
6	49007
7	49036
8	49071
9	49067
10	49001
11	49005
12	49041
13	49091
14	49033

Results per page: 50 ▾ 1 – 50 of 4178

```
select distinct geolocation_zip_code_prefix
FROM `Target_Case_Study.geolocation`
where geolocation_zip_code_prefix IN ((Select
distinct customer_zip_code_prefix
FROM `Target_Case_Study.customers`
where customer_zip_code_prefix NOT IN (SELECT
Distinct geolocation_zip_code_prefix
FROM `Target_Case_Study.geolocation` Geo
LEFT JOIN `Target_Case_Study.customers` C ON
C.customer_zip_code_prefix =
Geo.geolocation_zip_code_prefix
where customer_zip_code_prefix is not null))
)
```

157 ZipCodes are not present in Geolocation table But present in customers table.So,157 zipcodes missing in Geolocation

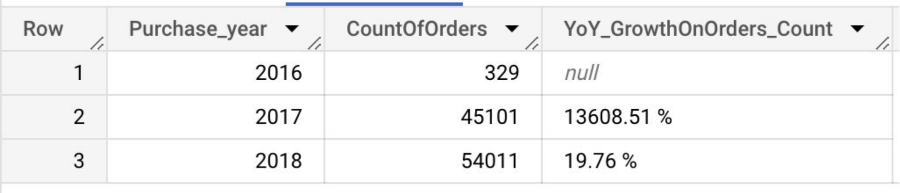


There is no data to display.

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

```
SELECT
 Purchase_year
 ,Count(order_id) AS CountOfOrders
 ,CONCAT(ROUND((Count(order_id)/(LAG(Count(order_id))OVER(ORDER BY Count(order_id) ASC))-1)*100,2)," %") as
YoY_GrowthOnOrders_Count
FROM
(select
  DISTINCT O.order_id
 ,order_purchase_timestamp
 ,EXTRACT(YEAR FROM order_purchase_timestamp) as Purchase_year
FROM `Target_Case_Study.orders` O
LEFT JOIN `Target_Case_Study.order_items` OI ON OI.order_id =O.order_id)
GROUP BY Purchase_year
ORDER BY Purchase_year ASC
```

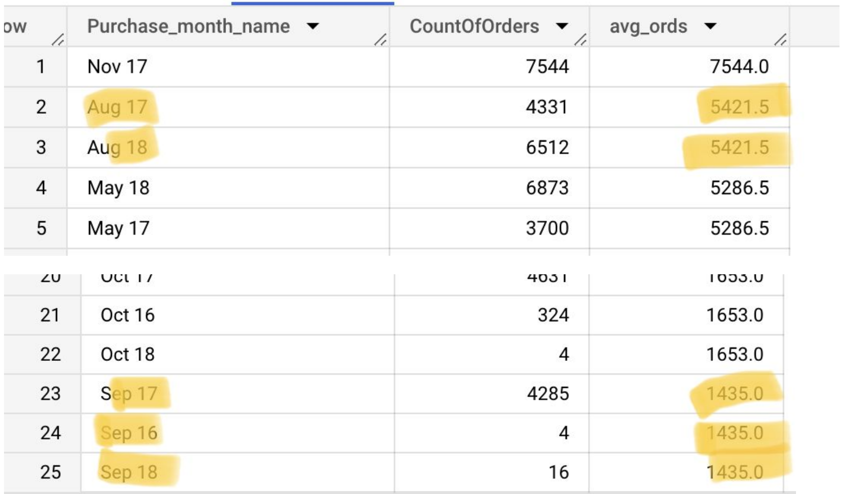
Yes,there is growing trend in the no. of orders placed over past years.



Row	Purchase_year	CountOfOrders	YoY_GrowthOnOrders_Count
1	2016	329	*null*
2	2017	45101	13608.51 %
3	2018	54011	19.76 %

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

```
SELECT Purchase_month_name,CountOfOrders,avg_ords
FROM
(SELECT
  Purchase_month_name
 ,Count(order_id) AS CountOfOrders
 ,AVG(Count(order_id))OVER(PARTITION BY Purchase_month ) as avg_ords
FROM
(select
  DISTINCT O.order_id
 ,order_purchase_timestamp
 ,EXTRACT(YEAR FROM order_purchase_timestamp) as Purchase_year
 ,format_datetime("%b %y",order_purchase_timestamp) as Purchase_month_name
 ,EXTRACT(MONTH FROM order_purchase_timestamp) as Purchase_month
FROM `Target_Case_Study.orders` O
LEFT JOIN `Target_Case_Study.order_items` OI ON OI.order_id =O.order_id)
GROUP BY Purchase_month_name,Purchase_month,Purchase_year
ORDER BY Purchase_year,Purchase_month)
ORDER BY avg_ords DESC
```



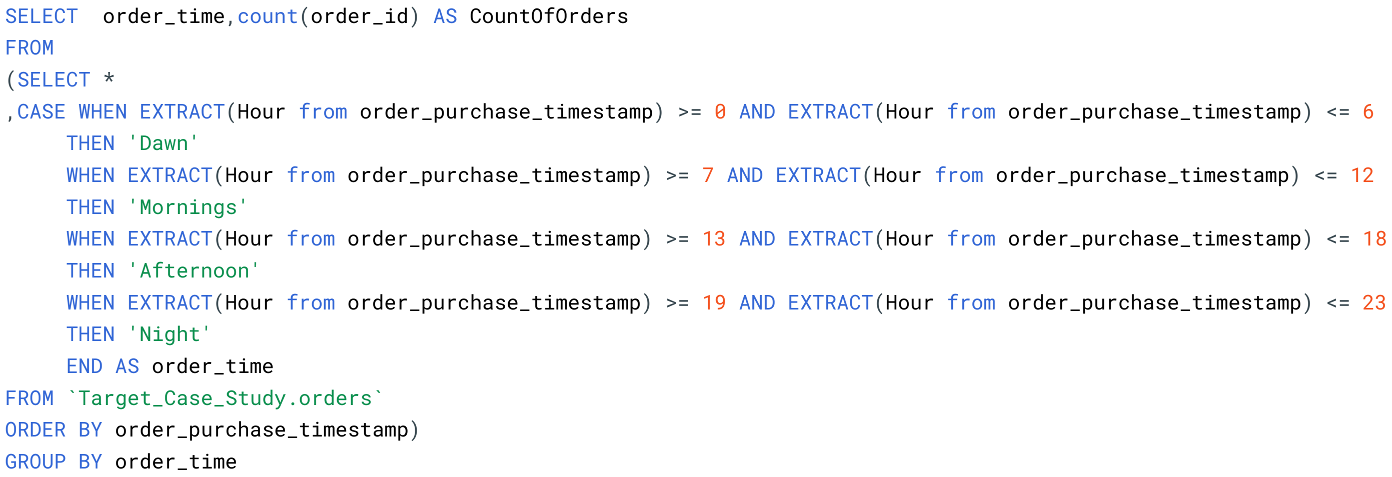
ow	Purchase_month_name	CountOfOrders	avg_ords
1	Nov 17	7544	7544.0
2	Aug 17	4331	5421.5
3	Aug 18	6512	5421.5
4	May 18	6873	5286.5
5	May 17	3700	5286.5
20	Oct 17	4631	1653.0
21	Oct 16	324	1653.0
22	Oct 18	4	1653.0
23	Sep 17	4285	1435.0
24	Sep 16	4	1435.0
25	Sep 18	16	1435.0

Yes, there is monthly seasonality in terms of no. of orders placed.

- Maximum Orders : August
- Minimum Orders : September

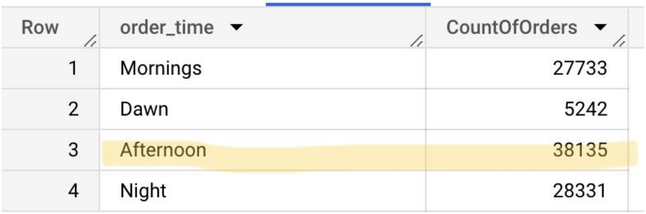
Note: The highest orders were placed in month Nov-2017 but there were no orders in Nov-2016 and Nov-2018

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



```
SELECT  order_time,count(order_id) AS CountOfOrders
FROM
(SELECT *
,CASE WHEN EXTRACT(Hour from order_purchase_timestamp) >= 0 AND EXTRACT(Hour from order_purchase_timestamp) <= 6
     THEN 'Dawn'
     WHEN EXTRACT(Hour from order_purchase_timestamp) >= 7 AND EXTRACT(Hour from order_purchase_timestamp) <= 12
     THEN 'Mornings'
     WHEN EXTRACT(Hour from order_purchase_timestamp) >= 13 AND EXTRACT(Hour from order_purchase_timestamp) <= 18
     THEN 'Afternoon'
     WHEN EXTRACT(Hour from order_purchase_timestamp) >= 19 AND EXTRACT(Hour from order_purchase_timestamp) <= 23
     THEN 'Night'
     END AS order_time
FROM `Target_Case_Study.orders`
ORDER BY order_purchase_timestamp)
GROUP BY order_time
```

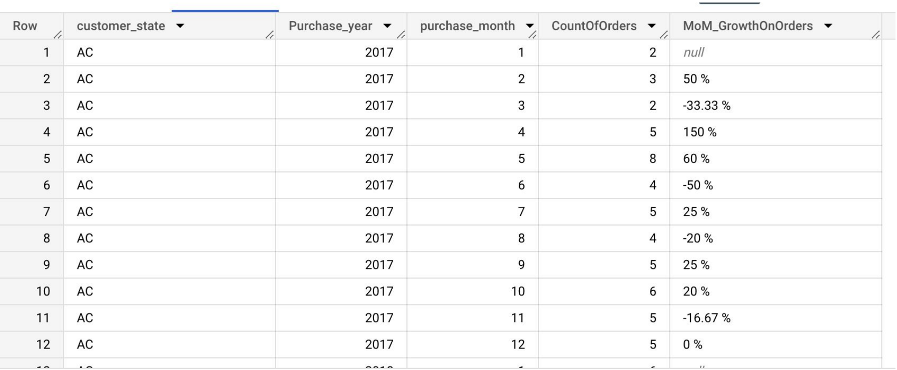
- **Brazilian customers mostly place orders at the time of Afternoon**



Row	order_time	CountOfOrders
1	Mornings	27733
2	Dawn	5242
3	Afternoon	38135
4	Night	28331

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

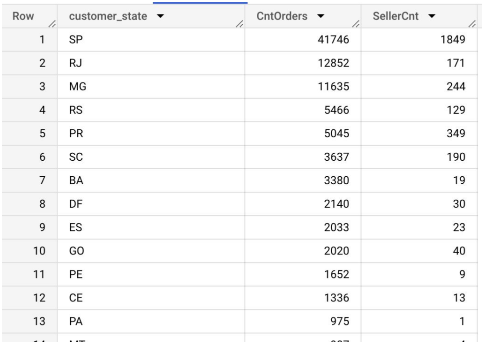
```
SELECT
 customer_state
 ,Purchase_year
,purchase_month
,Count(order_id) AS CountOfOrders
,CONCAT(ROUND((Count(order_id)/(LAG(Count(order_id))OVER(PARTITION BY customer_state,Purchase_year ORDER BY Purchase_year,purchase_month ASC))-1)*100,2),"
%") as MoM_GrowthOnOrders
FROM
(select
 DISTINCT O.order_id
,order_purchase_timestamp
,EXTRACT(YEAR FROM order_purchase_timestamp) as Purchase_year
,EXTRACT(Month FROM order_purchase_timestamp) as Purchase_month
,customer_state
FROM `Target_Case_Study.orders` O
LEFT JOIN `Target_Case_Study.order_items` OI ON OI.order_id =O.order_id
LEFT JOIN `Target_Case_Study.customers` C ON C.customer_id = O.customer_id)
GROUP BY customer_state,purchase_month,purchase_year
ORDER BY customer_state,purchase_year,purchase_month
```



Row	customer_state	Purchase_year	purchase_month	CountOfOrders	MoM_GrowthOnOrders
1	AC	2017	1	2	*null*
2	AC	2017	2	3	50 %
3	AC	2017	3	2	-33.33 %
4	AC	2017	4	5	150 %
5	AC	2017	5	8	60 %
6	AC	2017	6	4	-50 %
7	AC	2017	7	5	25 %
8	AC	2017	8	4	-20 %
9	AC	2017	9	5	25 %
10	AC	2017	10	6	20 %
11	AC	2017	11	5	-16.67 %
12	AC	2017	12	5	0 %

State wise Seller and Order count

```sql
SELECT customer_state,CntOrders,IF(sellerCnt is null,0,sellerCnt) as SellerCnt
FROM
(select customer_state,count(order_id) as CntOrders
from `Target_Case_Study.orders` O
left join `Target_Case_Study.customers` C ON C.customer_id = O.customer_id
group by customer_state) C
LEFT JOIN (SELECT seller_state,count(seller_id) as sellerCnt
FROM `Target_Case_Study.sellers`
group by seller_state) SS ON SS.seller_state = C.customer_state
order by CntOrders DESC
```

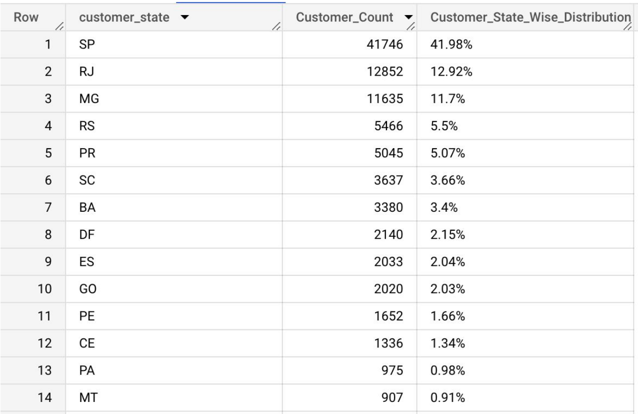


Row	customer_state	CntOrders	SellerCnt
1	SP	41746	1849
2	RJ	12852	171
3	MG	11635	244
4	RS	5466	129
5	PR	5045	349
6	SC	3637	190
7	BA	3380	19
8	DF	2140	30
9	ES	2033	23
10	GO	2020	40
11	PE	1652	9
12	CE	1336	13
13	PA	975	1
14	MT	907	4

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

State Wise Customer Distribution

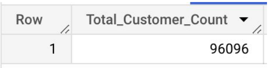
```
SELECT
customer_state
,count(customer_id) as Customer_Count
,CONCAT(ROUND((count(customer_id)/(select count(Distinct customer_id) FROM `Target_Case_Study.customers`))*100,2),"%") AS Customer_State_Wise_Distribution
FROM `Target_Case_Study.customers` C
GROUP BY customer_state
ORDER BY Customer_Count DESC
```



Row	customer_state	Customer_Count	Customer_State_Wise_Distribution
1	SP	41746	41.98%
2	RJ	12852	12.92%
3	MG	11635	11.7%
4	RS	5466	5.5%
5	PR	5045	5.07%
6	SC	3637	3.66%
7	BA	3380	3.4%
8	DF	2140	2.15%
9	ES	2033	2.04%
10	GO	2020	2.03%
11	PE	1652	1.66%
12	CE	1336	1.34%
13	PA	975	0.98%
14	MT	907	0.91%

Total Customers

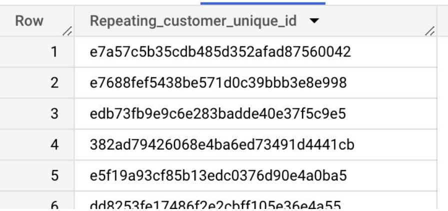
```
SELECT COUNT(Distinct customer_unique_id)
FROM `Target_Case_Study.customers`
```



Row	Total_Customer_Count
1	96096

Repeating Customers

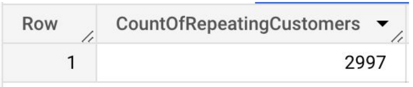
```
SELECT  customer_unique_id AS Repeating_customer_unique_id
FROM `Target_Case_Study.customers`
GROUP BY customer_unique_id
HAVING COUNT(customer_id) > 1
```



Row	Repeating_customer_unique_id
1	e7a57c5b35cdb485d352afad87560042
2	e7688fef5438be571d0c39bbb3e8e998
3	edb73fb9e9c6e283badde40e37f5c9e5
4	382ad79426068e4ba6ed73491d4441cb
5	e5f19a93cf85b13edc0376d90e4a0ba5
6	dd8253fe17486f2e2cbff105e36e4a55

Count of repeating customers

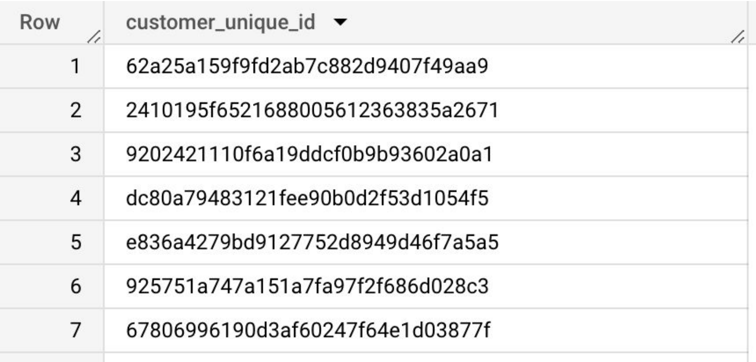
```
SELECT COUNT(Repeating_customer_unique_id) AS CountOfRepeatingCustomers
FROM
(SELECT  customer_unique_id AS Repeating_customer_unique_id
FROM `Target_Case_Study.customers`
GROUP BY customer_unique_id
HAVING COUNT(customer_id) > 1)
```



Row	CountOfRepeatingCustomers
1	2997

Customers ordered from more than one state.

```
SELECT customer_unique_id AS customer_unique_id
FROM
(select distinct customer_unique_id,customer_state
from `Target_Case_Study.customers`)
group by customer_unique_id
having count(customer_unique_id)>1
```



Row	customer_unique_id
1	62a25a159f9fd2ab7c882d9407f49aa9
2	2410195f6521688005612363835a2671
3	9202421110f6a19ddcf0b9b93602a0a1
4	dc80a79483121fee90b0d2f53d1054f5
5	e836a4279bd9127752d8949d46f7a5a5
6	925751a747a151a7fa97f2f686d028c3
7	67806996190d3af60247f64e1d03877f

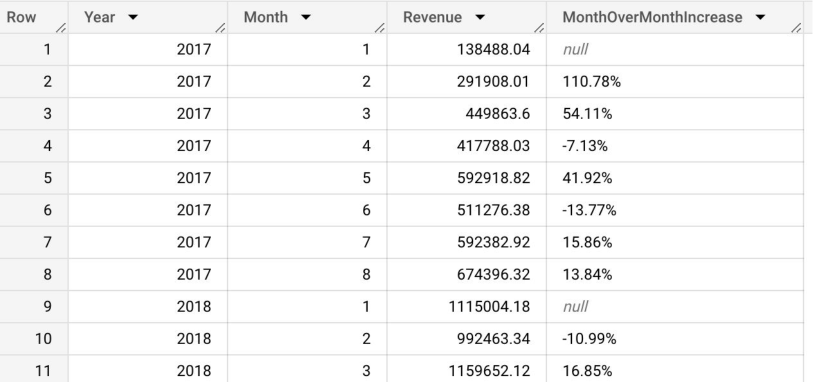
There are total 96096 customers.

There are 2997 Repeating Customers.

There are 39 Customers has ordered from more than one states.

Get the % increase in the cost of orders from year 2017 to 2018 (months between Jan to Aug only).

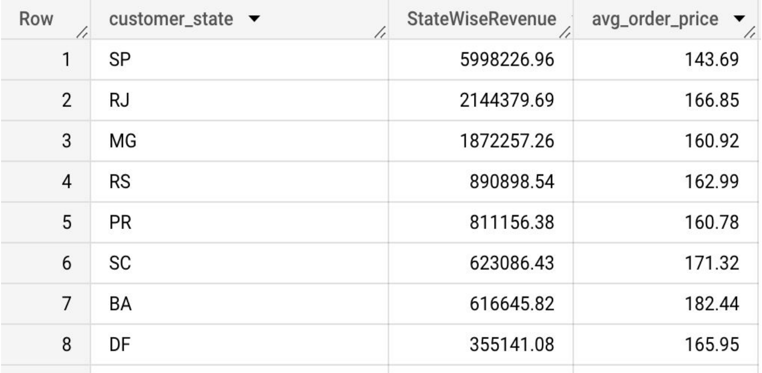
```
SELECT
Year
,Month
,((ROUND(SUM(CostOfOrder),2))) as Revenue
,CONCAT((ROUND((((ROUND(SUM(CostOfOrder),2))- (LAG(ROUND(SUM(CostOfOrder),2)) OVER(PARTITION BY Year ORDER BY
Month)))/LAG(ROUND(SUM(CostOfOrder),2)) OVER(PARTITION BY Year ORDER BY Month) * 100,2)),"%") AS MonthOverMonthIncrease
FROM
(SELECT
order_purchase_timestamp
,EXTRACT(Year FROM order_purchase_timestamp) As Year
,EXTRACT(Month FROM order_purchase_timestamp) AS Month
,O.order_id,SUM(payment_value) AS CostOfOrder
FROM `Target_Case_Study.orders` O
LEFT JOIN `Target_Case_Study.payments` P ON O.order_id = P.order_id
WHERE EXTRACT(DATE FROM order_purchase_timestamp) BETWEEN "2017-01-01" AND "2017-08-31"
OR EXTRACT(DATE FROM order_purchase_timestamp) BETWEEN "2018-01-01" AND "2018-08-31"
GROUP BY order_purchase_timestamp,O.order_id
ORDER BY order_purchase_timestamp)
GROUP BY Year,Month
ORDER BY Year,Month
```



Row	Year	Month	Revenue	MonthOverMonthIncrease
1	2017	1	138488.04	*null*
2	2017	2	291908.01	110.78%
3	2017	3	449863.6	54.11%
4	2017	4	417788.03	-7.13%
5	2017	5	592918.82	41.92%
6	2017	6	511276.38	-13.77%
7	2017	7	592382.92	15.86%
8	2017	8	674396.32	13.84%
9	2018	1	1115004.18	*null*
10	2018	2	992463.34	-10.99%
11	2018	3	1159652.12	16.85%

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

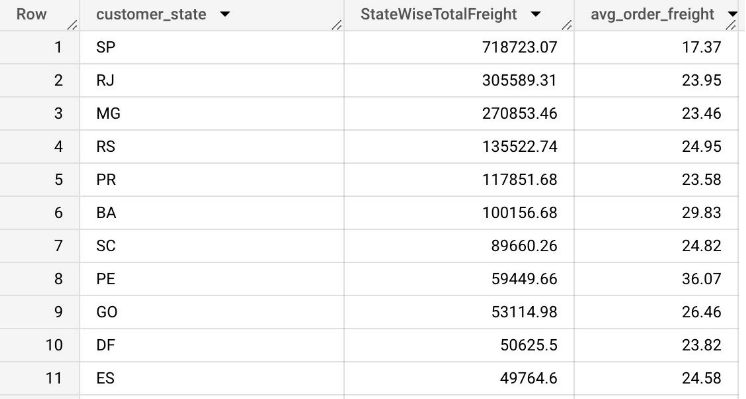
```sql
SELECT Distinct customer_state
,SUM(SUM(O.CostOfOrder))OVER(PARTITION BY customer_state) AS StateWiseRevenue
,ROUND(AVG(ROUND(SUM(O.CostOfOrder),2))OVER(PARTITION BY customer_state),2) AS avg_order_price
FROM
(SELECT
O.order_id
,O.customer_id
,ROUND(SUM(payment_value),2) AS CostOfOrder
FROM `Target_Case_Study.orders` O
LEFT JOIN `Target_Case_Study.payments` P ON O.order_id = P.order_id
GROUP BY O.order_id,O.customer_id
ORDER BY O.order_id,O.customer_id) O
LEFT JOIN `Target_Case_Study.customers` C ON C.customer_id = O.customer_id
GROUP BY customer_state,O.order_id
ORDER BY StateWiseRevenue DESC
```



Row	customer_state	StateWiseRevenue	avg_order_price
1	SP	5998226.96	143.69
2	RJ	2144379.69	166.85
3	MG	1872257.26	160.92
4	RS	890898.54	162.99
5	PR	811156.38	160.78
6	SC	623086.43	171.32
7	BA	616645.82	182.44
8	DF	355141.08	165.95

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

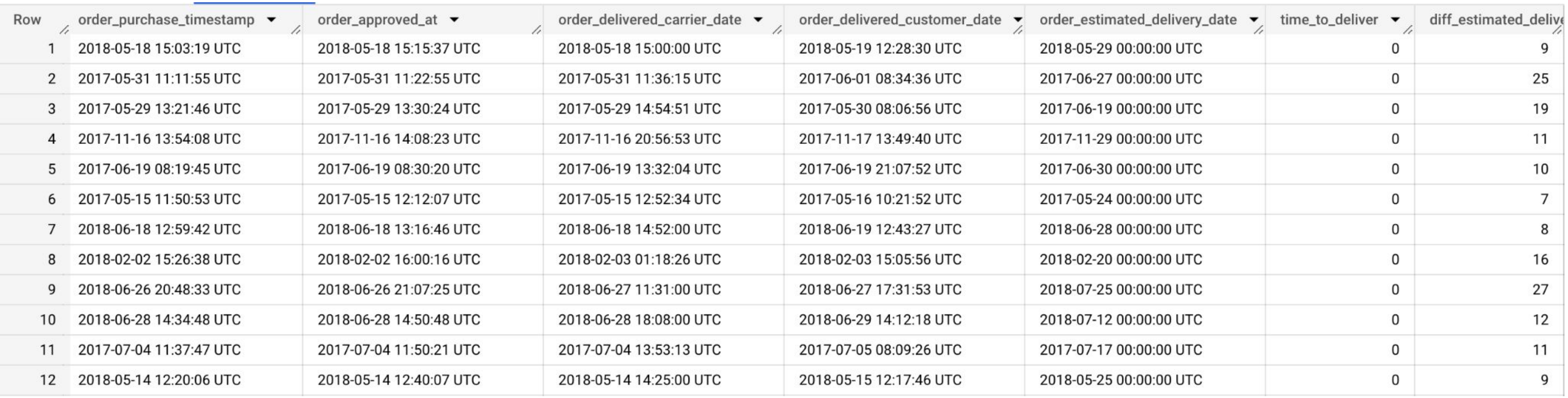
```
SELECT Distinct customer_state
,ROUND(SUM(SUM(O.freight))OVER(PARTITION BY customer_state),2) AS StateWiseTotalFreight
,ROUND(AVG(ROUND(SUM(O.freight),2))OVER(PARTITION BY customer_state),2) AS avg_order_freight
FROM
(SELECT
O.order_id
,O.customer_id
,ROUND(SUM(freight_value),2) AS freight
FROM `Target_Case_Study.orders` O
LEFT JOIN `Target_Case_Study.order_items` P ON O.order_id = P.order_id
GROUP BY O.order_id,O.customer_id
ORDER BY O.order_id,O.customer_id) O
LEFT JOIN `Target_Case_Study.customers` C ON C.customer_id = O.customer_id
GROUP BY customer_state,O.order_id
ORDER BY StateWiseTotalFreight DESC
```



Row	customer_state	StateWiseTotalFreight	avg_order_freight
1	SP	718723.07	17.37
2	RJ	305589.31	23.95
3	MG	270853.46	23.46
4	RS	135522.74	24.95
5	PR	117851.68	23.58
6	BA	100156.68	29.83
7	SC	89660.26	24.82
8	PE	59449.66	36.07
9	GO	53114.98	26.46
10	DF	50625.5	23.82
11	ES	49764.6	24.58

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.

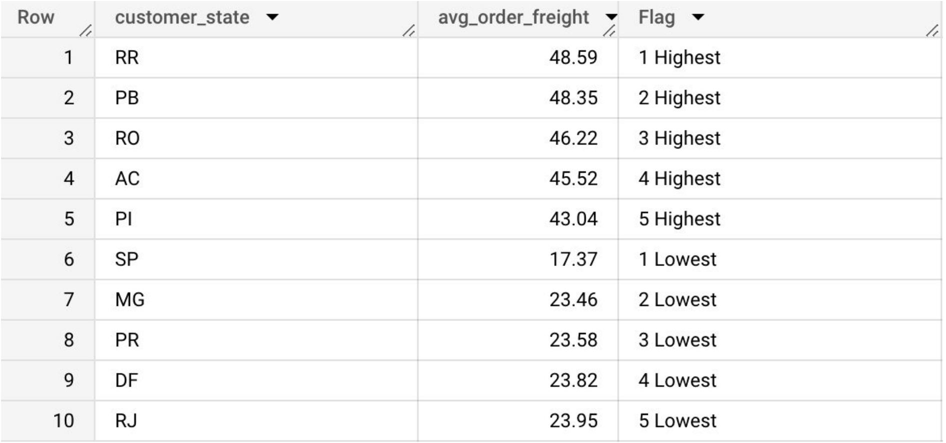
```
SELECT *
,TIMESTAMP_DIFF(order_delivered_customer_date,order_purchase_timestamp,DAY) AS time_to_deliver
,TIMESTAMP_DIFF(order_estimated_delivery_date,order_delivered_customer_date,DAY) AS diff_estimated_delivery
FROM `Target_Case_Study.orders`
WHERE order_status IN ('delivered')
AND order_delivered_customer_date IS NOT NULL
ORDER BY time_to_deliver
```



Row	order_purchase_timestamp	order_approved_at	order_delivered_carrier_date	order_delivered_customer_date	order_estimated_delivery_date	time_to_deliver	diff_estimated_delive
1	2018-05-18 15:03:19 UTC	2018-05-18 15:15:37 UTC	2018-05-18 15:00:00 UTC	2018-05-19 12:28:30 UTC	2018-05-29 00:00:00 UTC	0	9
2	2017-05-31 11:11:55 UTC	2017-05-31 11:22:55 UTC	2017-05-31 11:36:15 UTC	2017-06-01 08:34:36 UTC	2017-06-27 00:00:00 UTC	0	25
3	2017-05-29 13:21:46 UTC	2017-05-29 13:30:24 UTC	2017-05-29 14:54:51 UTC	2017-05-30 08:06:56 UTC	2017-06-19 00:00:00 UTC	0	19
4	2017-11-16 13:54:08 UTC	2017-11-16 14:08:23 UTC	2017-11-16 20:56:53 UTC	2017-11-17 13:49:40 UTC	2017-11-29 00:00:00 UTC	0	11
5	2017-06-19 08:19:45 UTC	2017-06-19 08:30:20 UTC	2017-06-19 13:32:04 UTC	2017-06-19 21:07:52 UTC	2017-06-30 00:00:00 UTC	0	10
6	2017-05-15 11:50:53 UTC	2017-05-15 12:12:07 UTC	2017-05-15 12:52:34 UTC	2017-05-16 10:21:52 UTC	2017-05-24 00:00:00 UTC	0	7
7	2018-06-18 12:59:42 UTC	2018-06-18 13:16:46 UTC	2018-06-18 14:52:00 UTC	2018-06-19 12:43:27 UTC	2018-06-28 00:00:00 UTC	0	8
8	2018-02-02 15:26:38 UTC	2018-02-02 16:00:16 UTC	2018-02-03 01:18:26 UTC	2018-02-03 15:05:56 UTC	2018-02-20 00:00:00 UTC	0	16
9	2018-06-26 20:48:33 UTC	2018-06-26 21:07:25 UTC	2018-06-27 11:31:00 UTC	2018-06-27 17:31:53 UTC	2018-07-25 00:00:00 UTC	0	27
10	2018-06-28 14:34:48 UTC	2018-06-28 14:50:48 UTC	2018-06-28 18:08:00 UTC	2018-06-29 14:12:18 UTC	2018-07-12 00:00:00 UTC	0	12
11	2017-07-04 11:37:47 UTC	2017-07-04 11:50:21 UTC	2017-07-04 13:53:13 UTC	2017-07-05 08:09:26 UTC	2017-07-17 00:00:00 UTC	0	11
12	2018-05-14 12:20:06 UTC	2018-05-14 12:40:07 UTC	2018-05-14 14:25:00 UTC	2018-05-15 12:17:46 UTC	2018-05-25 00:00:00 UTC	0	9

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

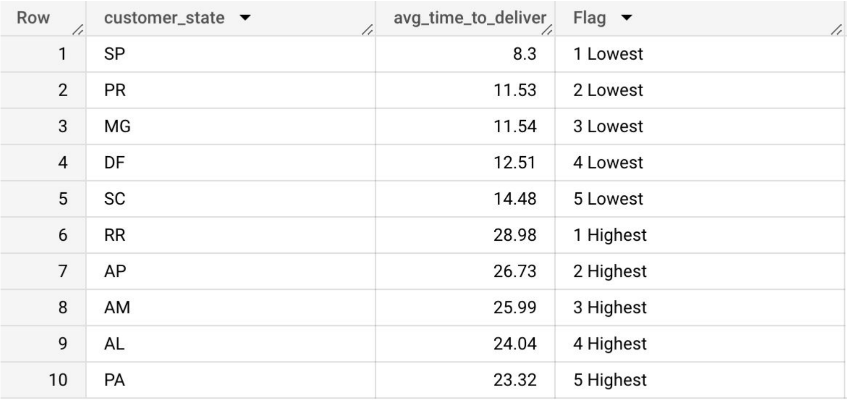
```
WITH AvgFreightData AS(SELECT *
FROM (SELECT Distinct customer_state
,ROUND(AVG(ROUND(SUM(O.freight),2))OVER(PARTITION BY customer_state),2) AS avg_order_freight
FROM
(SELECT
O.order_id
,O.customer_id
,ROUND(SUM(freight_value),2) AS freight
FROM `Target_Case_Study.orders` O
LEFT JOIN `Target_Case_Study.order_items` P ON O.order_id = P.order_id
GROUP BY O.order_id,O.customer_id
ORDER BY O.order_id,O.customer_id) O
LEFT JOIN `Target_Case_Study.customers` C ON C.customer_id = O.customer_id
GROUP BY customer_state,O.order_id)
ORDER BY avg_order_freight )

(Select *
,CONCAT(DENSE_RANK()OVER(ORDER BY avg_order_freight DESC)," Highest") AS Flag
FROM AvgFreightData
ORDER BY avg_order_freight DESC LIMIT 5)
UNION ALL
(Select *
,CONCAT(DENSE_RANK()OVER(ORDER BY avg_order_freight ASC)," Lowest") AS Flag
FROM AvgFreightData
ORDER BY avg_order_freight ASC LIMIT 5)
```



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

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

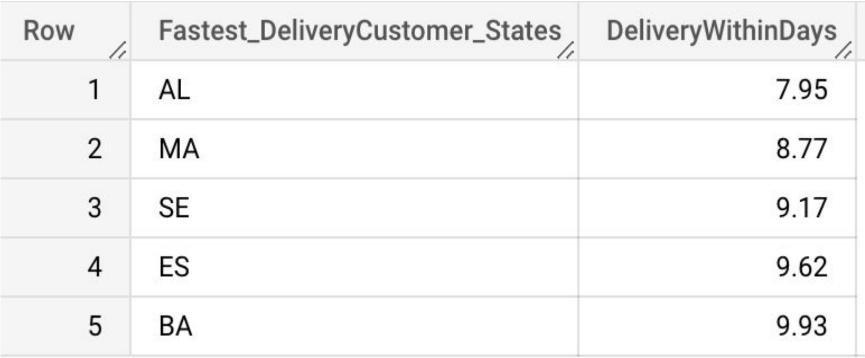
```
WITH orders AS(SELECT C.customer_state,O.order_id,O.customer_id,O.order_status,order_purchase_timestamp,order_delivered_customer_date,order_estimated_delivery_date
,TIMESTAMP_DIFF(order_delivered_customer_date,order_purchase_timestamp,DAY) AS time_to_deliver
,TIMESTAMP_DIFF(order_estimated_delivery_date,order_delivered_customer_date,DAY) AS diff_estimated_delivery
FROM `Target_Case_Study.orders` O
LEFT JOIN `Target_Case_Study.customers` C ON O.customer_id = C.customer_id
WHERE order_status IN ('delivered')
AND order_delivered_customer_date IS NOT NULL
ORDER BY time_to_deliver)

(SELECT customer_state
,ROUND(SUM(time_to_deliver)/COUNT(order_id),2) AS avg_time_to_deliver
,CONCAT(ROW_NUMBER()OVER(ORDER BY (SUM(time_to_deliver)/COUNT(order_id)) DESC)," Highest") as Flag
FROM orders
GROUP BY customer_state
ORDER BY avg_time_to_deliver DESC LIMIT 5)
UNION ALL
(SELECT customer_state
,ROUND(SUM(time_to_deliver)/COUNT(order_id),2) AS avg_time_to_deliver
,CONCAT(ROW_NUMBER()OVER(ORDER BY (SUM(time_to_deliver)/COUNT(order_id)))," Lowest") as Flag
FROM orders
GROUP BY customer_state
ORDER BY avg_time_to_deliver LIMIT 5)
```



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

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

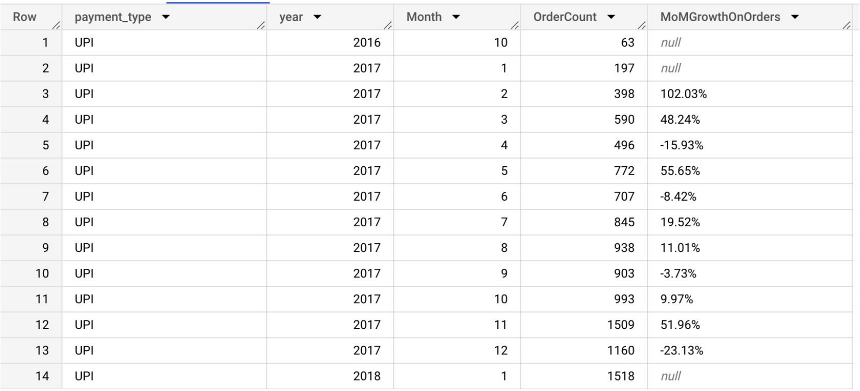
```
WITH orders AS(SELECT
C.customer_state,O.order_id,O.customer_id,O.order_status,order_purchase_timestamp,order_delivered_customer_date,order_estimated_delivery_date
,TIMESTAMP_DIFF(order_delivered_customer_date,order_purchase_timestamp,DAY) AS time_to_deliver
,TIMESTAMP_DIFF(order_estimated_delivery_date,order_delivered_customer_date,DAY) AS diff_estimated_delivery
FROM `Target_Case_Study.orders` O
LEFT JOIN `Target_Case_Study.customers` C ON O.customer_id = C.customer_id
WHERE order_status IN ('delivered')
AND order_delivered_customer_date IS NOT NULL
ORDER BY diff_estimated_delivery DESC)


(SELECT customer_state AS Fastest_DeliveryCustomer_States
,ROUND(SUM(diff_estimated_delivery)/COUNT(order_id),2) AS DeliveryWithinDays
FROM orders
GROUP BY customer_state
ORDER BY DeliveryWithinDays ASC LIMIT 5)
```



Row	Fastest_DeliveryCustomer_States	DeliveryWithinDays
1	AL	7.95
2	MA	8.77
3	SE	9.17
4	ES	9.62
5	BA	9.93

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

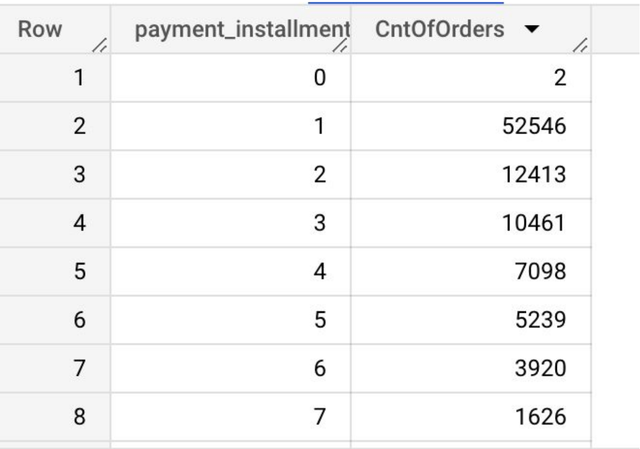
```
SELECT payment_type,year,Month,SUM(CntOfOrders) AS OrderCount
,CONCAT(ROUND((((SUM(CntOfOrders))-LAG(SUM(CntOfOrders))OVER(PARTITION BY payment_type,year ORDER BY
year,Month))/LAG(SUM(CntOfOrders))OVER(PARTITION BY payment_type,year ORDER BY year,Month))*100,2),"%") AS MoMGrowthOnOrders
FROM
(SELECT payment_type
,EXTRACT(Year FROM O.order_purchase_timestamp) AS Year
,EXTRACT(Month FROM O.order_purchase_timestamp) AS Month
,(COUNT(Distinct O.order_id)) AS CntOfOrders
FROM `Target_Case_Study.orders` O
left join `Target_Case_Study.payments`  P ON P.order_id = O.order_id
where payment_type is not null
Group by payment_type,year,month
Order by payment_type,year,month)
GROUP BY payment_type,year,Month
ORDER BY payment_type,year,Month
```



Row	payment_type	year	Month	OrderCount	MoMGrowthOnOrders
1	UPI	2016	10	63	*null*
2	UPI	2017	1	197	*null*
3	UPI	2017	2	398	102.03%
4	UPI	2017	3	590	48.24%
5	UPI	2017	4	496	-15.93%
6	UPI	2017	5	772	55.65%
7	UPI	2017	6	707	-8.42%
8	UPI	2017	7	845	19.52%
9	UPI	2017	8	938	11.01%
10	UPI	2017	9	903	-3.73%
11	UPI	2017	10	993	9.97%
12	UPI	2017	11	1509	51.96%
13	UPI	2017	12	1160	-23.13%
14	UPI	2018	1	1518	*null*

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

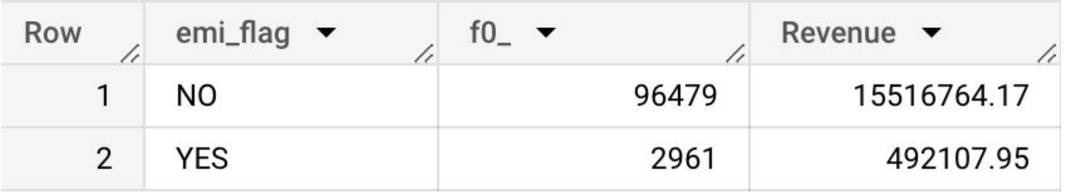
```
SELECT payment_installments,count(order_id) as CntOfOrders
FROM `Target_Case_Study.payments`
GROUP BY payment_installments
ORDER BY payment_installments
```



Row	payment_installment	CntOfOrders
1	0	2
2	1	52546
3	2	12413
4	3	10461
5	4	7098
6	5	5239
7	6	3920
8	7	1626

Orders with EMI’s and Non-EMI’s

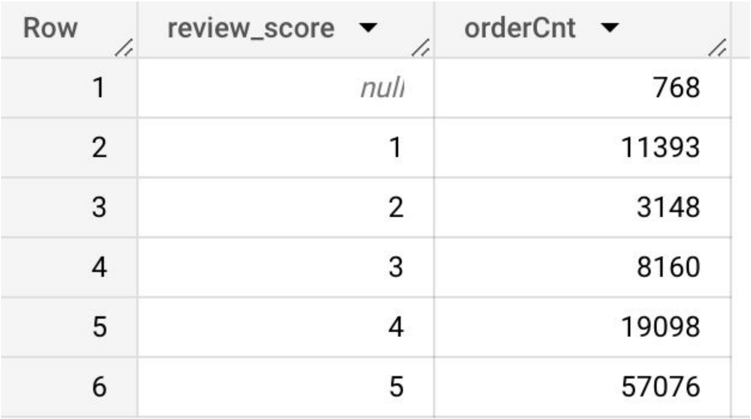
```
WITH payments AS(SELECT P.order_id,payment_sequential,payment_type,payment_installments,payment_value
,case when P1.order_id is null
     THEN 'NO'
     ELSE 'YES' END AS EMI_Flag
FROM `Target_Case_Study.payments`  P
LEFT JOIN (select distinct order_id from `Target_Case_Study.payments`
where order_id IN (select order_id
from `Target_Case_Study.payments`
group by order_id
Having count(order_id)>1)) P1 ON P.order_id = P1.order_id)

SELECT emi_flag,COUNT(Distinct order_id),ROUND(SUM(payment_value),2) AS Revenue
from payments
group by emi_flag
```



Row	emi_flag	f0_	Revenue
1	NO	96479	15516764.17
2	YES	2961	492107.95

Count of order based on review

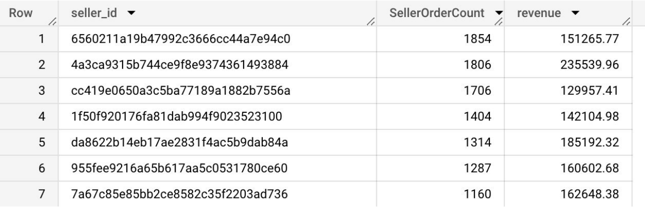
```
select review_score,COUNT(Distinct O.order_id) as orderCnt
FROM `Target_Case_Study.orders` O
LEFT JOIN `Target_Case_Study.order_reviews` R ON R.order_id = O.order_id
Group by review_score
Order by review_score
```



Row	review_score ▼	orderCnt ▼
1	*null*	768
2	1	11393
3	2	3148
4	3	8160
5	4	19098
6	5	57076

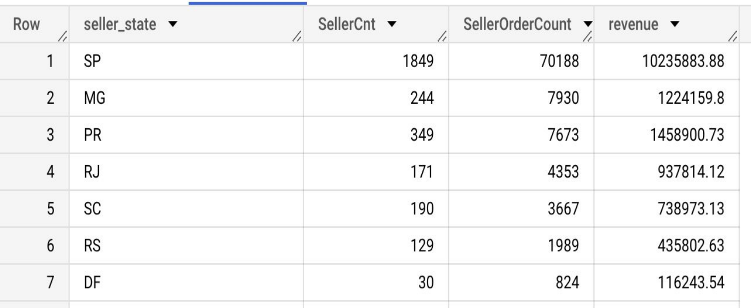
Count of orders and revenue of each sellers

```sql
select seller_id ,count(Distinct O.order_id) as SellerOrderCount,ROUND(SUM(price + freight_value),2) AS revenue
from `Target_Case_Study.orders` O
LEFT JOIN `Target_Case_Study.order_items` OI ON OI.order_id = O.order_id
GROUP BY seller_id
ORDER BY SellerOrderCount DESC
```



Row	seller_id	SellerOrderCount	revenue
1	6560211a19b47992c3666cc44a7e94c0	1854	151265.77
2	4a3ca9315b744ce9f8e9374361493884	1806	235539.96
3	cc419e0650a3c5ba77189a1882b7556a	1706	129957.41
4	1f50f920176fa81dab994f9023523100	1404	142104.98
5	da8622b14eb17ae2831f4ac5b9dab84a	1314	185192.32
6	955fee9216a65b617aa5c0531780ce60	1287	160602.68
7	7a67c85e85bb2ce8582c35f2203ad736	1160	162648.38

```sql
State Wise Seller Order Count and Revenue
select S.seller_state
,COUNT(DISTINCT S.seller_id) AS SellerCnt
,count(Distinct O.order_id) as SellerOrderCount
,ROUND(SUM(price + freight_value),2) AS revenue
from `Target_Case_Study.orders` O
LEFT JOIN `Target_Case_Study.order_items` OI ON OI.order_id = O.order_id
LEFT JOIN `Target_Case_Study.sellers` S ON S.seller_id = OI.seller_id
GROUP BY S.seller_state
ORDER BY SellerOrderCount DESC
```

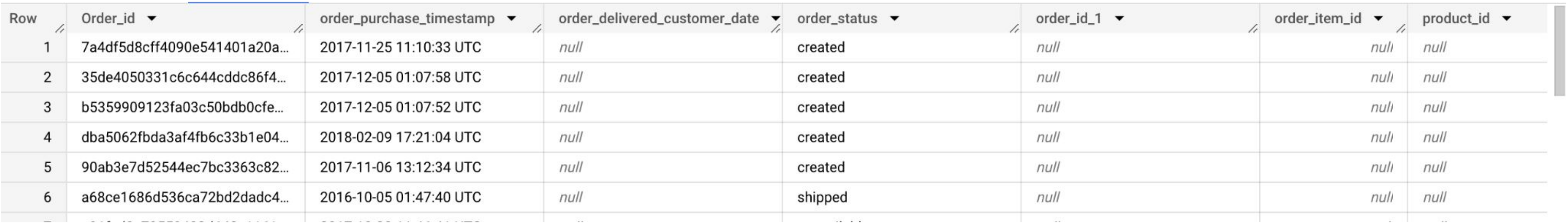


Row	seller_state	SellerCnt	SellerOrderCount	revenue
1	SP	1849	70188	10235883.88
2	MG	244	7930	1224159.8
3	PR	349	7673	1458900.73
4	RJ	171	4353	937814.12
5	SC	190	3667	738973.13
6	RS	129	1989	435802.63
7	DF	30	824	116243.54

Orders which are placed but not flowing in order_items table

```
select Distinct O.Order_id,O.order_purchase_timestamp,O.order_delivered_customer_date,O.order_status,OI.*,P.*
from `Target_Case_Study.orders` O
LEFT JOIN `Target_Case_Study.order_items` OI ON OI.order_id = O.order_id
LEFT JOIN `Target_Case_Study.payments` P ON P.order_id = O.order_id
where OI.order_id is null
and order_status IN ('created','shipped','unavailable')
```

Observation : For some of the order_id’s the data was not present in orders_items.
Customers are ordering the products which are not available.
For products that are not available we should show OUT OF STOCK.



Row	Order_id	order_purchase_timestamp	order_delivered_customer_date	order_status	order_id_1	order_item_id	product_id
1	7a4df5d8cff4090e541401a20a...	2017-11-25 11:10:33 UTC	*null*	created	*null*	*null*	*null*
2	35de4050331c6c644cddc86f4...	2017-12-05 01:07:58 UTC	*null*	created	*null*	*null*	*null*
3	b5359909123fa03c50bdb0cfe...	2017-12-05 01:07:52 UTC	*null*	created	*null*	*null*	*null*
4	dba5062fbda3af4fb6c33b1e04...	2018-02-09 17:21:04 UTC	*null*	created	*null*	*null*	*null*
5	90ab3e7d52544ec7bc3363c82...	2017-11-06 13:12:34 UTC	*null*	created	*null*	*null*	*null*
6	a68ce1686d536ca72bd2dadc4...	2016-10-05 01:47:40 UTC	*null*	shipped	*null*	*null*	*null*

Detailed Observations:

There are total 27 states customers are present in all 27 states.

There are total 8011 cities but customers are present in only 4119

There are total 4178 zip codes where customers are not present

157 ZipCodes are not present in Geolocation table, but present in customers table.So,157 zipcodes missing in Geolocation

There is growing trend in the no. of orders placed over past years.

Yes, there is monthly seasonality in terms of no. of orders placed.

- Maximum Orders : August (10,843 orders placed which is 10.9% of entire orders)
- Minimum Orders : September (4,305 orders placed which is 4.3% of entire orders)

Brazilian customers mostly place orders at the time of Afternoon

- Afternoon : 38,135 orders placed
- Dawn : 5,242 orders placed

State SP has highest customers were as RR has lowest customers

- SP : 41746
- RR : 46

There are total 96096 customers and 2997 are repeating customers

One customer is ordered from different states.

- 39 customers orders from different states

SP has lowest avg cost per order and PB has highest avg cost per order

- SP : 143.69
- PP : 264.08

SP has lowest freight and RR has highest freight

- SP : 17.37
- RR : 48.59

Delivery time

- SP : 8 days which is fast from rest all states.
- RR : 29 days which is very late.

Fastest delivery as compared to estimated delivery time

- AL,SE,ME,ES,BA (Orders deliver 7-10 days faster than estimated date)

Max Orders placed using Credit Card and least with Debit Card

Customers purchased orders with 1 installment approx. 52k

There are more orders where sellers are more and vice versa.

Recommendations :

Pay attention to the cities were customers are not present.We have business from only 51.4% of cities in brazil.

Need detailed analysis on the this cities.

Fix geolocation data, some customers zip codes are missing.

In the month of September bring some SALE/DISCOUNTS or give offers to the customers to boost sales in september.

There are very less repeating customers only 3%.

Give some coupons on purchase so that they will use that coupon and order for next tym, which will increase the revenue.

Increase sellers in each state to increase orders.

Mostly focus on states where sellers are very less.

There should be at least 50+ sellers in each state.

Delivery speed should be increased, most of the orders delivered within 5 days of delivery time.Some orders are taking more than 100 days to deliver.

Highest orders are placed in month of august.Hence, the store operating times,delivery time,operating staff should be taken care to provide customer satisfaction.

The estimated delivery time is more than the actual delivery time for all of the orders.The estimated delivery time should be precisely mentioned near to the actual delivery time.So, the customer get the better idea about when the product will be delivered.

Show stock availability for customers for the products they want to order.It will decrease cost for refund.

NOTE : If sellers increased most of the problems will be solved and it will boost the sale and customer satisfaction.