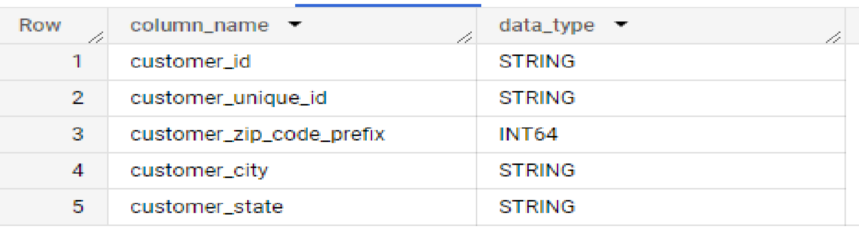
TARGET – SQL Business Case study

1) **Import the dataset and do usual exploratory analysis steps like checking the structure & characteristics of the dataset**

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

- SELECT column_name , data_type
- FROM Customer_details.INFORMATION_SCHEMA.COLUMNS
- WHERE table_name='customers';

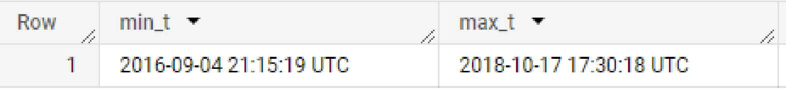


Row	column_name	data_type
1	customer_id	STRING
2	customer_unique_id	STRING
3	customer_zip_code_prefix	INT64
4	customer_city	STRING
5	customer_state	STRING

- There are 5 columns in the “**customers**” table.
- 4 columns have **String** datatype and 1 column has **integer** datatype.

Q.2 Get the time range between which the orders were placed.

- SELECT
- MIN(order_purchase_timestamp) AS min_t,
- MAX(order_purchase_timestamp) AS max_t,
- FROM `Customer_details.orders`



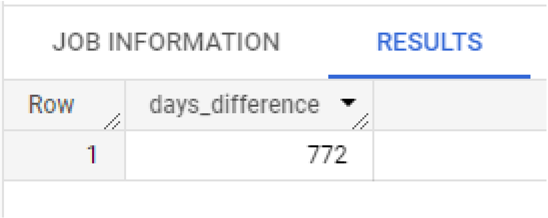
Row	min_t	max_t
1	2016-09-04 21:15:19 UTC	2018-10-17 17:30:18 UTC

- The first order was placed on 4th Sept , 2016 ,while the last order was placed on 17th Oct , 2018.

Additional Inference:

Difference in number of days from the first order till the last?

- SELECT
- TIMESTAMP_DIFF(max_t,min_t,day) AS days_difference
- FROM (SELECT
- MIN(order_purchase_timestamp) AS min_t,
- MAX(order_purchase_timestamp) AS max_t,
- FROM `Customer_details.orders`)



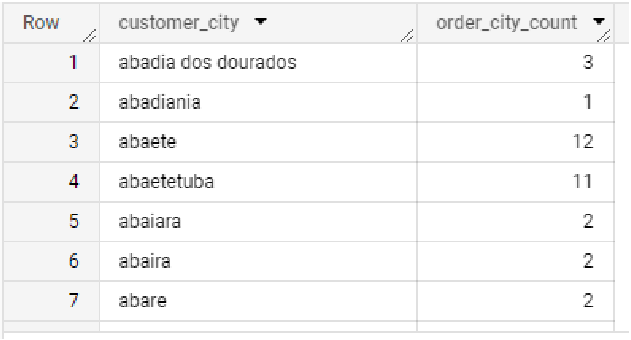
Row	days_difference
1	772

➢ There was a difference of 772 days from first order to the last.

Q.3 Count the Cities & States of customers who ordered during the given period.

CITIES

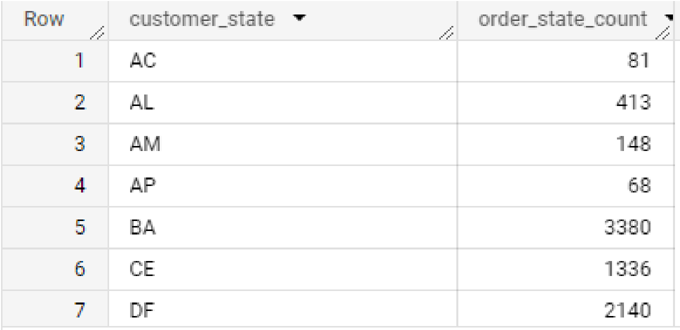
- SELECT
- c1.customer_city,
- COUNT(c1.customer_city) as order_city_count
- FROM `Customer_details.customers` AS c1 INNER JOIN `Customer_details.orders` AS o1 on c1.customer_id = o1.customer_id
- GROUP BY c1.customer_city
- ORDER BY c1.customer_city



Row	customer_city	order_city_count
1	abadia dos dourados	3
2	abadiania	1
3	abaete	12
4	abaetetuba	11
5	abaiara	2
6	abaira	2
7	abare	2

STATE

- SELECT
- c1.customer_state,
- COUNT(c1.customer_state) as order_state_count
- FROM `Customer_details.customers` AS c1 INNER JOIN `Customer_details.orders` AS o1 on c1.customer_id = o1.customer_id
- GROUP BY c1.customer_state
- ORDER BY c1.customer_state



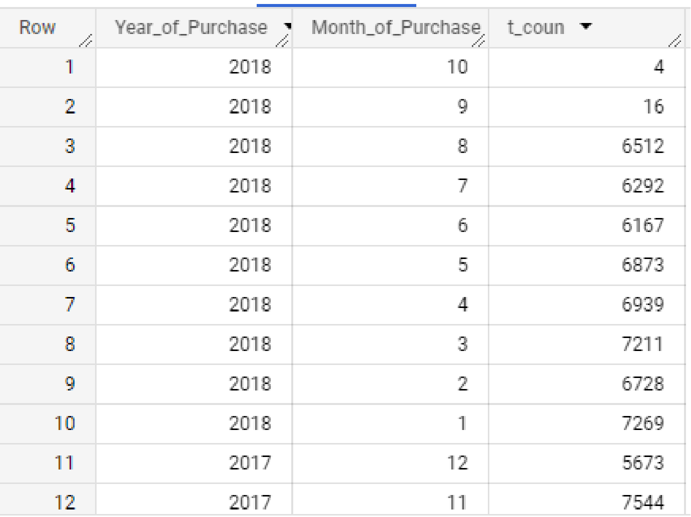
Row	customer_state	order_state_count
1	AC	81
2	AL	413
3	AM	148
4	AP	68
5	BA	3380
6	CE	1336
7	DF	2140

➢ Counted the number of orders per City , &
➢ Counted the number of orders per State

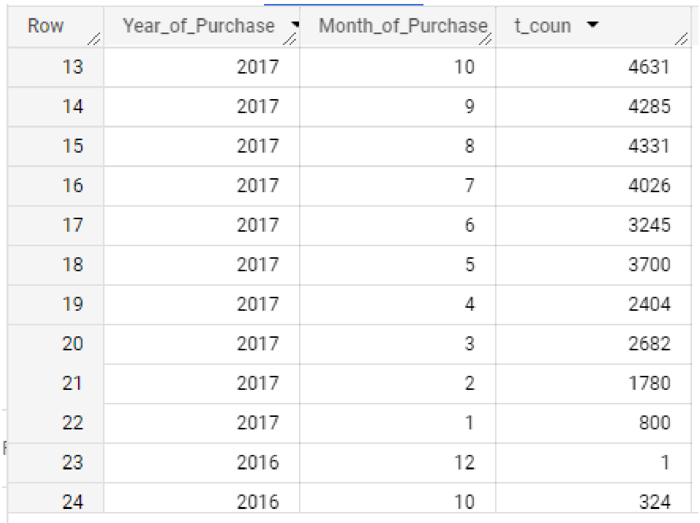
2) In-depth Exploration:

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

- SELECT
- Year_of_Purchase,
- Month_of_Purchase,
- COUNT(Month_of_Purchase) AS t_coun
- FROM (SELECT
- EXTRACT(YEAR FROM order_purchase_timestamp) AS Year_of_Purchase,
- EXTRACT(MONTH FROM order_purchase_timestamp) AS Month_of_Purchase,
- FROM `Customer_details.orders`)
- GROUP BY Year_of_Purchase , Month_of_Purchase
- ORDER BY Year_of_Purchase DESC ,Month_of_Purchase DESC



Row	Year_of_Purchase	Month_of_Purchase	t_coun
1	2018	10	4
2	2018	9	16
3	2018	8	6512
4	2018	7	6292
5	2018	6	6167
6	2018	5	6873
7	2018	4	6939
8	2018	3	7211
9	2018	2	6728
10	2018	1	7269
11	2017	12	5673
12	2017	11	7544

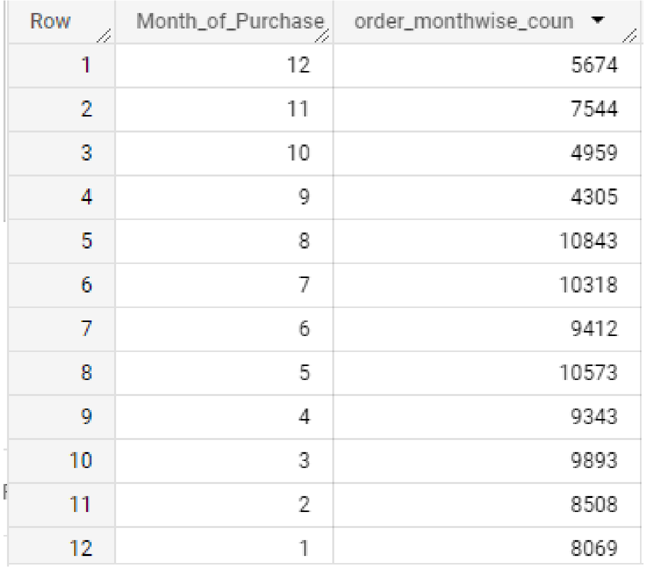


Row	Year_of_Purchase	Month_of_Purchase	t_coun
13	2017	10	4631
14	2017	9	4285
15	2017	8	4331
16	2017	7	4026
17	2017	6	3245
18	2017	5	3700
19	2017	4	2404
20	2017	3	2682
21	2017	2	1780
22	2017	1	800
23	2016	12	1
24	2016	10	324

- ➢ There is a growing trend in the number of orders placed over the past years
- ➢ However, in the latest months i.e last couple of months of 2018 there has been a sharp decline in orders purchased
- ➢ But if we compare the orders purchased in 2018 from that of 2017 and 2016 , then there is a growing trend if exception months of 2018 are left aside.

Q.2 Can we see some kind of monthly seasonality in terms of the no. of orders being placed?

- SELECT
- Month_of_Purchase,
- COUNT(Month_of_Purchase) AS order_monthwise_coun
- FROM
- (SELECT
- EXTRACT(MONTH FROM order_purchase_timestamp) AS Month_of_Purchase
- FROM `Customer_details.orders`)
- GROUP BY Month_of_Purchase
- ORDER BY Month_of_Purchase DESC

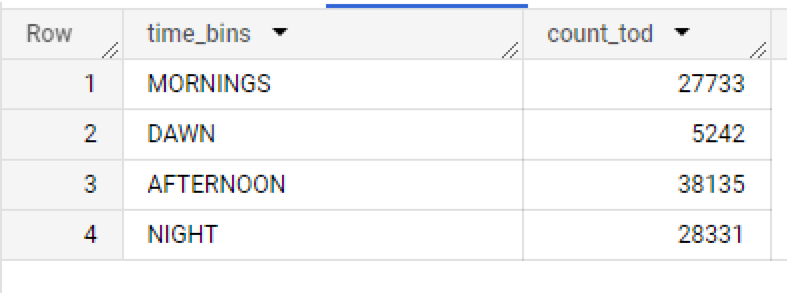


Row	Month_of_Purchase	order_monthwise_coun
1	12	5674
2	11	7544
3	10	4959
4	9	4305
5	8	10843
6	7	10318
7	6	9412
8	5	10573
9	4	9343
10	3	9893
11	2	8508
12	1	8069

- ➢ We can't see any significant difference in terms of numbers of orders being placed when it comes to monthly seasonality.
- ➢ However , there is a slight up-trend in orders in mid-months(5th to 8th) and a downtrend in the end-months(9th to 12th).

Q.3 During what time of the day, do the Brazilian customers mostly place their orders? (Dawn, Morning, Afternoon or Night) ?

- SELECT
- time_bins,
- COUNT(time_bins) AS count_tod
- FROM (SELECT
- order_purchase_timestamp,
- extracted_hour,
- CASE
- WHEN extracted_hour BETWEEN 0 AND 6
- THEN "DAWN"
- WHEN extracted_hour BETWEEN 7 AND 12
- THEN "MORNINGS"
- WHEN extracted_hour BETWEEN 13 AND 18
- THEN "AFTERNOON"
- WHEN extracted_hour BETWEEN 19 AND 23
- THEN "NIGHT"
- END AS time_bins
- FROM (SELECT
- order_purchase_timestamp,
- EXTRACT(HOUR FROM order_purchase_timestamp) AS extracted_hour,
- FROM `Customer_details.orders`))
- GROUP BY time_bins



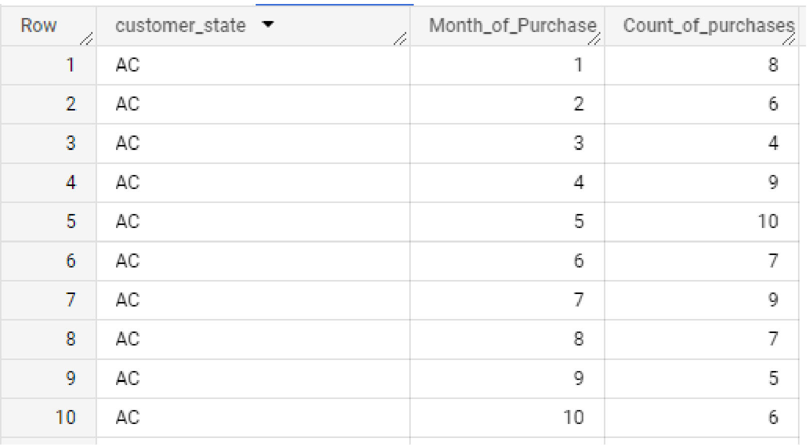
Row	time_bins	count_tod
1	MORNINGS	27733
2	DAWN	5242
3	AFTERNOON	38135
4	NIGHT	28331

➢ Brazilian customers mostly place their orders in ‘Afternoon’ as per the data.

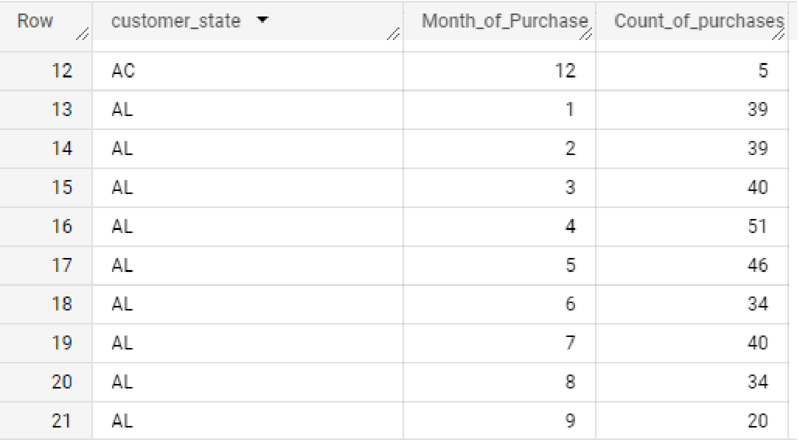
3) Evolution of E-commerce orders in the Brazil region:

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

- SELECT
- customer_state,
- Month_of_Purchase,
- COUNT(Month_of_Purchase) AS Count_of_purchases
- FROM (SELECT
- c1.customer_state,
- EXTRACT(MONTH FROM o1.order_purchase_timestamp) AS Month_of_Purchase
- FROM `Customer_details.customers` AS c1 INNER JOIN `Customer_details.orders` AS o1 on c1.customer_id = o1.customer_id)
- GROUP BY customer_state ,Month_of_Purchase
- ORDER BY customer_state , Month_of_Purchase ;



Row	customer_state ▾	Month_of_Purchase	Count_of_purchases
1	AC	1	8
2	AC	2	6
3	AC	3	4
4	AC	4	9
5	AC	5	10
6	AC	6	7
7	AC	7	9
8	AC	8	7
9	AC	9	5
10	AC	10	6

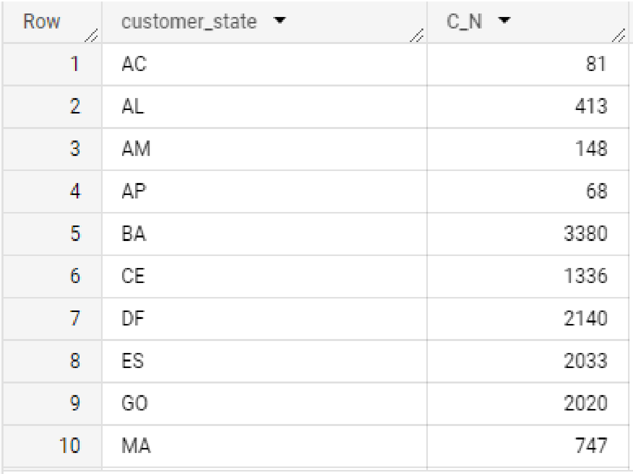


Row	customer_state ▾	Month_of_Purchase	Count_of_purchases
12	AC	12	5
13	AL	1	39
14	AL	2	39
15	AL	3	40
16	AL	4	51
17	AL	5	46
18	AL	6	34
19	AL	7	40
20	AL	8	34
21	AL	9	20

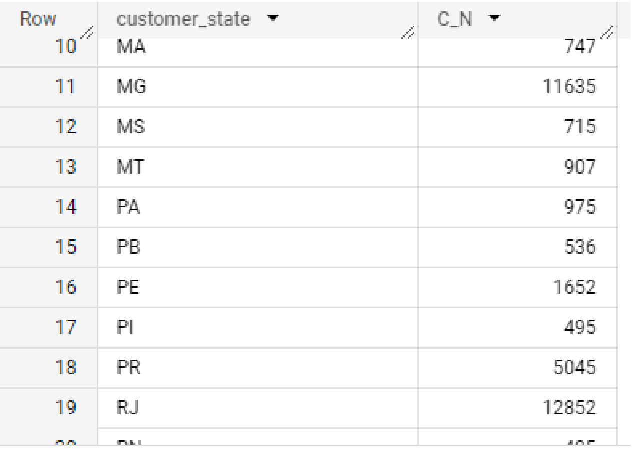
➢ Given above is the month-on-month no. of orders placed in each state.

Q.2 How are the customers distributed across all the states?

- SELECT
- customer_state,
- COUNT(customer_id) AS C_N
- FROM `Customer_details.customers`
- GROUP BY customer_state
- ORDER BY customer_state



Row	customer_state	C_N
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



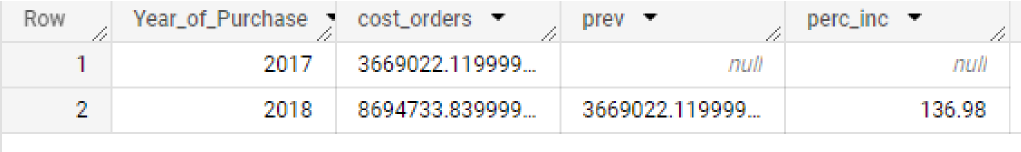
Row	customer_state	C_N
10	MA	747
11	MG	11635
12	MS	715
13	MT	907
14	PA	975
15	PB	536
16	PE	1652
17	PI	495
18	PR	5045
19	RJ	12852

➢ Above is the distribution of customers state wise.

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

Q.1 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.

- SELECT
- Year_of_Purchase,
- cost_orders,
- prev,
- ROUND(((cost_orders - prev)/prev) * 100 , 2) AS perc_inc
- FROM (
- SELECT
- Year_of_Purchase,
- cost_orders,
- LAG(cost_orders,1,NULL)OVER(ORDER BY Year_of_Purchase) AS prev
- FROM (SELECT
- Year_of_Purchase,
- SUM(payment_value) AS cost_orders
- FROM (SELECT
- o1.order_id,
- EXTRACT(MONTH FROM o1.order_purchase_timestamp) AS Month_of_Purchase,
- EXTRACT(YEAR FROM o1.order_purchase_timestamp) AS Year_of_Purchase,
- p1.payment_value
- FROM `Customer_details.orders` as o1 INNER JOIN `Customer_details.payments` as p1 ON o1.order_id = p1.order_id)
- WHERE Year_of_Purchase IN (2017,2018) AND Month_of_Purchase BETWEEN 1 AND 8
- GROUP BY Year_of_Purchase))

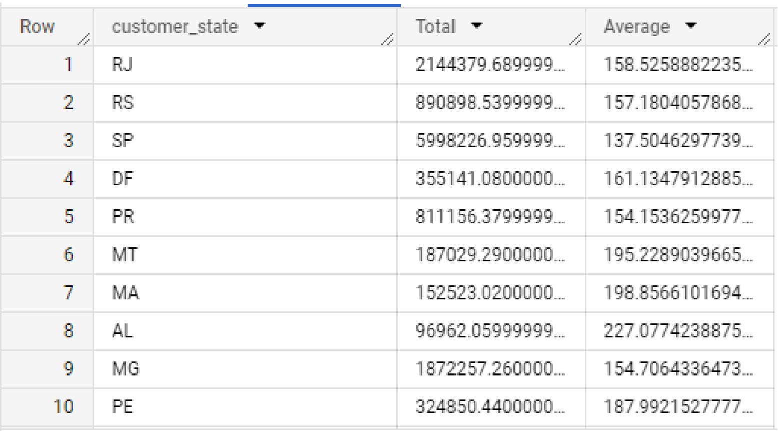


Row	Year_of_Purchase	cost_orders	prev	perc_inc
1	2017	3669022.119999…	*null*	*null*
2	2018	8694733.839999…	3669022.119999…	136.98

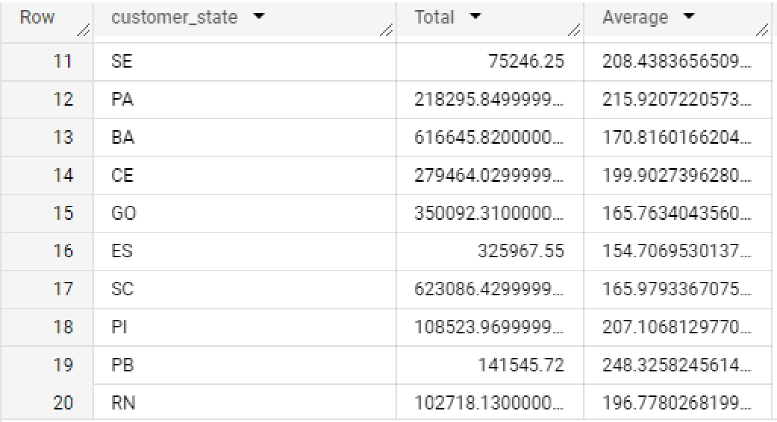
➢ Hence, we can infer that there was a 136% increase in the cost of orders from year 2017 to 2018.

Q.2 Calculate the Total & Average value of order price for each state.

- SELECT
- c1.customer_state,
- SUM(p1.payment_value) AS Total,
- AVG(p1.payment_value) AS Average
- FROM `Customer_details.customers` AS c1 INNER JOIN `Customer_details.orders` AS o1 ON c1.customer_id = o1.customer_id INNER JOIN `Customer_details.payments` as p1 ON o1.order_id = p1.order_id
- GROUP BY c1.customer_state



Row	customer_state	Total	Average
1	RJ	2144379.689999...	158.5258882235...
2	RS	890898.5399999...	157.1804057868...
3	SP	5998226.959999...	137.5046297739...
4	DF	355141.0800000...	161.1347912885...
5	PR	811156.3799999...	154.1536259977...
6	MT	187029.2900000...	195.2289039665...
7	MA	152523.0200000...	198.8566101694...
8	AL	96962.05999999...	227.0774238875...
9	MG	1872257.260000...	154.7064336473...
10	PE	324850.4400000...	187.9921527777...

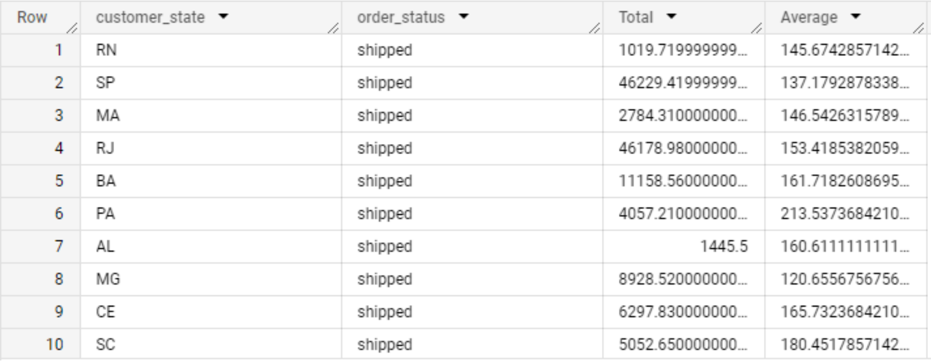


Row	customer_state	Total	Average
11	SE	75246.25	208.4383656509...
12	PA	218295.8499999...	215.9207220573...
13	BA	616645.8200000...	170.8160166204...
14	CE	279464.0299999...	199.9027396280...
15	GO	350092.3100000...	165.7634043560...
16	ES	325967.55	154.7069530137...
17	SC	623086.4299999...	165.9793367075...
18	PI	108523.9699999...	207.1068129770...
19	PB	141545.72	248.3258245614...
20	RN	102718.1300000...	196.7780268199...

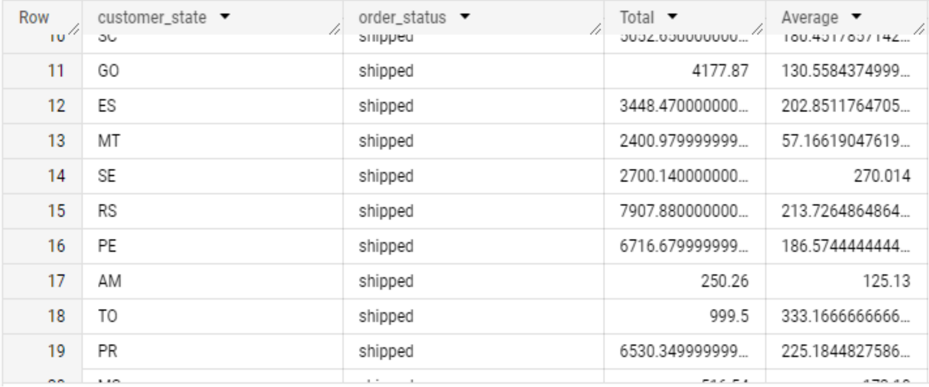
➢ Henceforth , the above image gives the result of Total & Average value of Order price for each state.

Q.3 Calculate the Total & Average value of order freight for each state

- SELECT
- c1.customer_state,
- o1.order_status,
- SUM(p1.payment_value) AS Total,
- AVG(p1.payment_value) AS Average
- FROM `Customer_details.customers` AS c1 INNER JOIN `Customer_details.orders` AS o1 ON c1.customer_id = o1.customer_id INNER JOIN `Customer_details.payments` as p1 ON o1.order_id = p1.order_id
- WHERE o1.order_status = 'shipped'
- GROUP BY c1.customer_state,o1.order_status



Row	customer_state	order_status	Total	Average
1	RN	shipped	1019.719999999...	145.6742857142...
2	SP	shipped	46229.41999999...	137.1792878338...
3	MA	shipped	2784.310000000...	146.5426315789...
4	RJ	shipped	46178.98000000...	153.4185382059...
5	BA	shipped	11158.56000000...	161.7182608695...
6	PA	shipped	4057.210000000...	213.5373684210...
7	AL	shipped	1445.5	160.6111111111...
8	MG	shipped	8928.520000000...	120.6556756756...
9	CE	shipped	6297.830000000...	165.7323684210...
10	SC	shipped	5052.650000000...	180.4517857142...



Row	customer_state	order_status	Total	Average
11	GO	shipped	4177.87	130.5584374999...
12	ES	shipped	3448.470000000...	202.8511764705...
13	MT	shipped	2400.979999999...	57.16619047619...
14	SE	shipped	2700.140000000...	270.014
15	RS	shipped	7907.880000000...	213.7264864864...
16	PE	shipped	6716.679999999...	186.5744444444...
17	AM	shipped	250.26	125.13
18	TO	shipped	999.5	333.1666666666...
19	PR	shipped	6530.349999999...	225.1844827586...

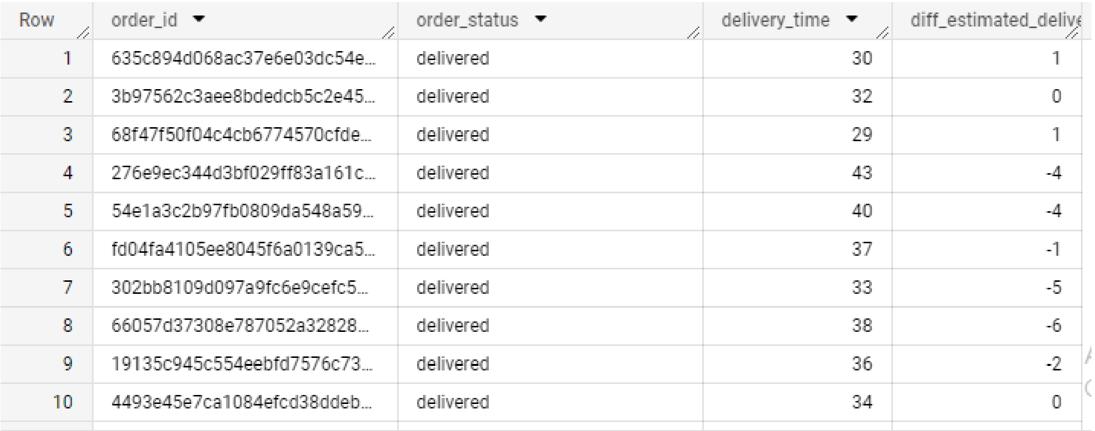
➢ Henceforth , the above image gives the result of Total & Average value of Order freight for each state.

5) Analysis based on sales, freight and delivery time

Q.1 **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`
- `order_id,`
- `order_status,`
- `TIMESTAMP_DIFF(order_delivered_customer_date,order_purchase_timestamp, DAY) AS delivery_time,`
- `TIMESTAMP_DIFF(order_estimated_delivery_date,order_delivered_customer_date,DAY) AS diff_estimated_delivery_time`
- ``FROM `Customer_details.orders` ``
- `WHERE order_status = 'delivered'`



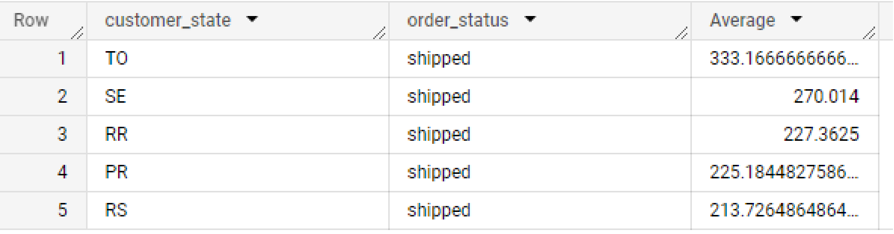
Row	order_id	order_status	delivery_time	diff_estimated_delive
1	635c894d068ac37e6e03dc54e…	delivered	30	1
2	3b97562c3aee8bdedcb5c2e45…	delivered	32	0
3	68f47f50f04c4cb6774570cfde…	delivered	29	1
4	276e9ec344d3bf029ff83a161c…	delivered	43	-4
5	54e1a3c2b97fb0809da548a59…	delivered	40	-4
6	fd04fa4105ee8045f6a0139ca5…	delivered	37	-1
7	302bb8109d097a9fc6e9cefc5…	delivered	33	-5
8	66057d37308e787052a32828…	delivered	38	-6
9	19135c945c554eebfd7576c73…	delivered	36	-2
10	4493e45e7ca1084efcd38ddeb…	delivered	34	0

- In the above results , 'delivery_time' refers to the actual time taken to deliver the order
- 'diff_estimated_delivery_time' refers to the difference in estimated time of order delivery and actual time taken to deliver the order.

Q.2 Find out the top 5 states with the highest & lowest average freight value

- SELECT
- c1.customer_state,

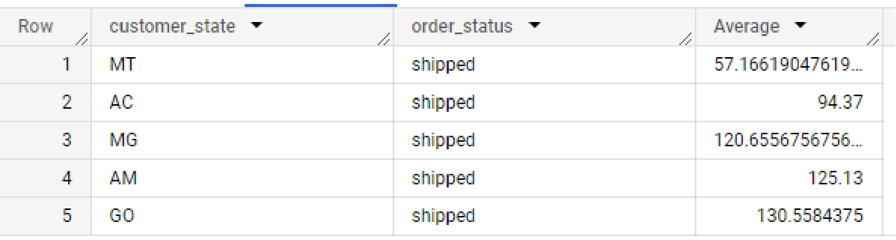
- o1.order_status,
- AVG(p1.payment_value) AS Average
- FROM `Customer_details.customers` AS c1 INNER JOIN `Customer_details.orders` AS o1 ON c1.customer_id = o1.customer_id INNER JOIN `Customer_details.payments` as p1 ON o1.order_id = p1.order_id
- WHERE o1.order_status = 'shipped'
- GROUP BY c1.customer_state,o1.order_status
- ORDER BY average DESC
- LIMIT 5



Row	customer_state	order_status	Average
1	TO	shipped	333.1666666666...
2	SE	shipped	270.014
3	RR	shipped	227.3625
4	PR	shipped	225.1844827586...
5	RS	shipped	213.7264864864...

- ➢ Top 5 states with highest freight cost

- SELECT
- c1.customer_state,
- o1.order_status,
- AVG(p1.payment_value) AS Average
- FROM `Customer_details.customers` AS c1 INNER JOIN `Customer_details.orders` AS o1 ON c1.customer_id = o1.customer_id INNER JOIN `Customer_details.payments` as p1 ON o1.order_id = p1.order_id
- WHERE o1.order_status = 'shipped'
- GROUP BY c1.customer_state,o1.order_status
- ORDER BY average
- LIMIT 5

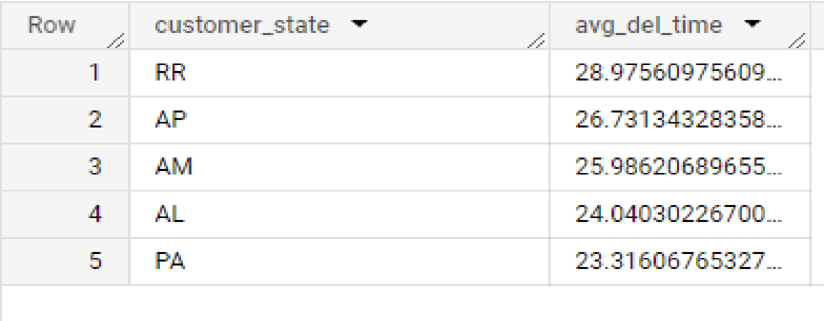


Row	customer_state	order_status	Average
1	MT	shipped	57.16619047619...
2	AC	shipped	94.37
3	MG	shipped	120.6556756756...
4	AM	shipped	125.13
5	GO	shipped	130.5584375

- ➢ Top 5 states with lowest freight cost

Q.3 Find out the top 5 states with the highest & lowest average delivery time.

- SELECT
- customer_state,
- AVG(delivery_time) AS avg_del_time
- FROM (SELECT
- c1.customer_state,
- o1.order_id,
- o1.order_status,
- TIMESTAMP_DIFF(o1.order_delivered_customer_date,o1.order_purchase_timestamp,DAY) AS delivery_time,
- FROM `Customer_details.orders` AS o1 INNER JOIN `Customer_details.customers` AS c1 ON o1.customer_id = c1.customer_id
- WHERE o1.order_status = 'delivered')
- GROUP BY customer_state
- ORDER BY avg_del_time DESC
- LIMIT 5

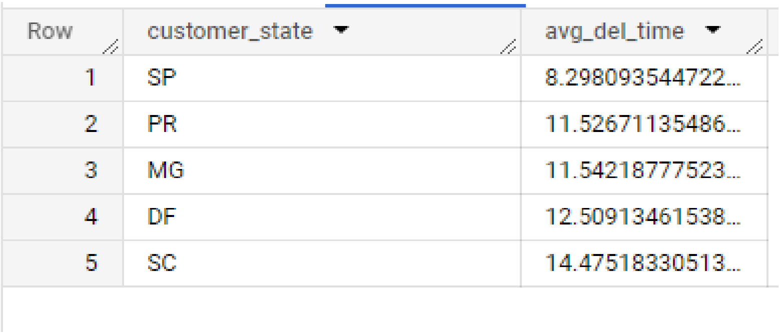


Row	customer_state	avg_del_time
1	RR	28.97560975609...
2	AP	26.73134328358...
3	AM	25.98620689655...
4	AL	24.04030226700...
5	PA	23.31606765327...

- ➢ top 5 states with the highest average delivery time.

- SELECT
- customer_state,
- AVG(delivery_time) AS avg_del_time
- FROM (SELECT
- c1.customer_state,
- o1.order_id,
- o1.order_status,
- TIMESTAMP_DIFF(o1.order_delivered_customer_date,o1.order_purchase_timestamp,DAY) AS delivery_time,
- FROM `Customer_details.orders` AS o1 INNER JOIN `Customer_details.customers` AS c1 ON o1.customer_id = c1.customer_id
- WHERE o1.order_status = 'delivered')
- GROUP BY customer_state
- ORDER BY avg_del_time

- LIMIT 5

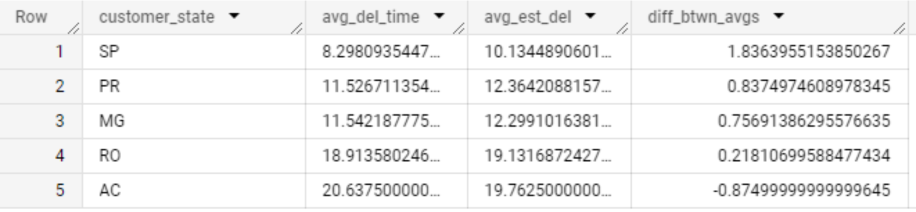


Row	customer_state	avg_del_time
1	SP	8.298093544722…
2	PR	11.52671135486…
3	MG	11.54218777523…
4	DF	12.50913461538…
5	SC	14.47518330513…

➢ top 5 states with the lowest average delivery time.

Q.4 Find out the top 5 states where the order delivery is really fast as compared to the estimated date of delivery.

- SELECT
- customer_state,
- avg_del_time,
- avg_est_del,
- (avg_est_del - avg_del_time) AS diff_btwn_avgs
- FROM (SELECT
- customer_state,
- AVG(delivery_time) AS avg_del_time,
- AVG(diff_estimated_delivery_time) AS avg_est_del,
- FROM (SELECT
- c1.customer_state,
- o1.order_id,
- o1.order_status,
- TIMESTAMP_DIFF(o1.order_delivered_customer_date,o1.order_purchase_timestamp,DAY) AS delivery_time,
- TIMESTAMP_DIFF(o1.order_estimated_delivery_date,o1.order_delivered_customer_date,DAY) AS diff_estimated_delivery_time
- FROM `Customer_details.orders` AS o1 INNER JOIN `Customer_details.customers` AS c1 ON o1.customer_id = c1.customer_id
- WHERE o1.order_status = 'delivered')
- GROUP BY customer_state
- ORDER BY avg_del_time)
- ORDER BY diff_btwn_avgs DESC
- LIMIT 5



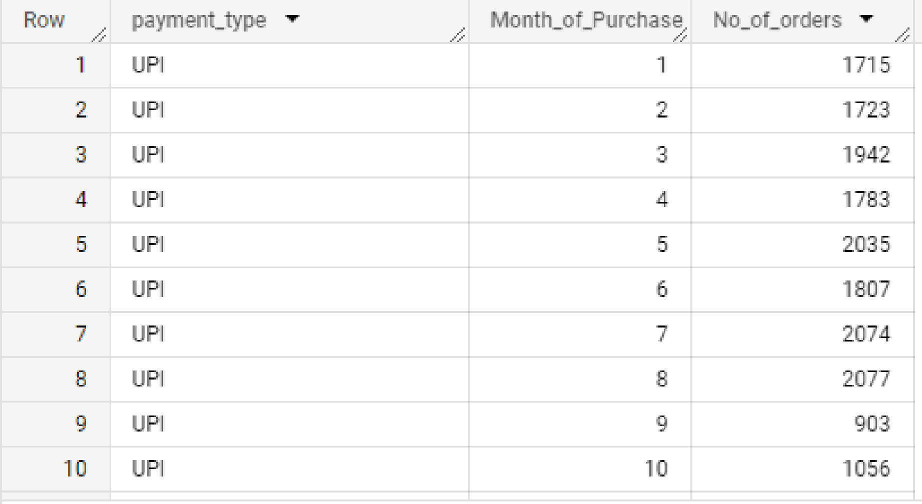
Row	customer_state	avg_del_time	avg_est_del	diff_btwn_avgs
1	SP	8.2980935447…	10.1344890601…	1.8363955153850267
2	PR	11.526711354…	12.3642088157…	0.8374974608978345
3	MG	11.542187775…	12.2991016381…	0.75691386295576635
4	RO	18.913580246…	19.1316872427…	0.21810699588477434
5	AC	20.637500000…	19.7625000000…	-0.87499999999999645

- ➢ top 5 states where the order delivery is really fast as compared to the estimated date of delivery

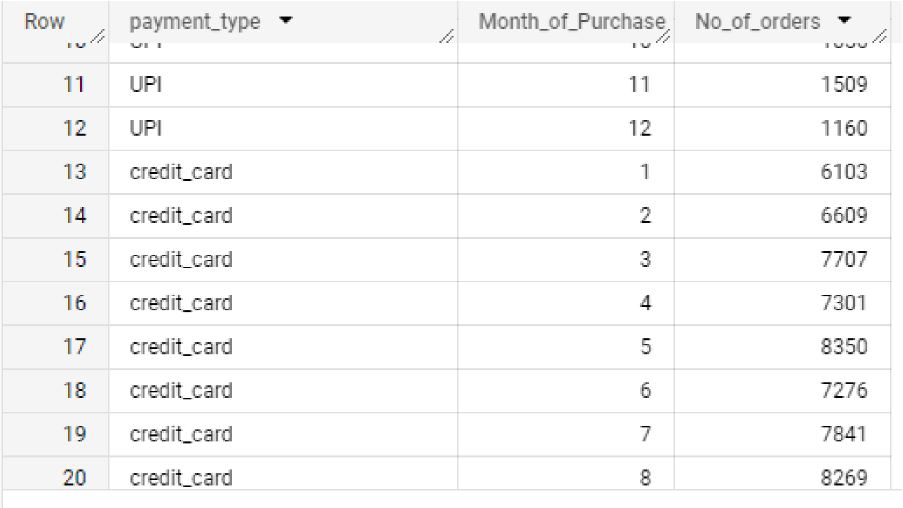
6) Analysis based on the payments:

Q.1 Find the month on month no. of orders placed using different payment types.

- SELECT
- payment_type,
- Month_of_Purchase,
- COUNT(order_id) AS No_of_orders
- FROM (SELECT
- o1.order_id,
- p1.payment_type,
- EXTRACT(MONTH FROM o1.order_purchase_timestamp) AS Month_of_Purchase
- FROM `Customer_details.payments` AS p1 INNER JOIN `Customer_details.orders` AS o1 on p1.order_id = o1.order_id)
- GROUP BY Payment_type,Month_of_Purchase
- ORDER BY Payment_type,Month_of_Purchase ;



Row	payment_type	Month_of_Purchase	No_of_orders
1	UPI	1	1715
2	UPI	2	1723
3	UPI	3	1942
4	UPI	4	1783
5	UPI	5	2035
6	UPI	6	1807
7	UPI	7	2074
8	UPI	8	2077
9	UPI	9	903
10	UPI	10	1056

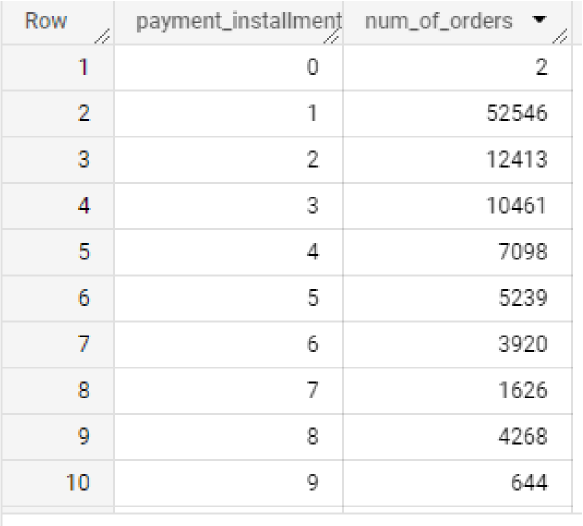


Row	payment_type	Month_of_Purchase	No_of_orders
11	UPI	11	1509
12	UPI	12	1160
13	credit_card	1	6103
14	credit_card	2	6609
15	credit_card	3	7707
16	credit_card	4	7301
17	credit_card	5	8350
18	credit_card	6	7276
19	credit_card	7	7841
20	credit_card	8	8269

- ➢ Above shows Month on Month no. of orders placed using different payment types

Q.2 Find the no. of orders placed on the basis of the payment installments that have been paid.

- SELECT
- p1.payment_installments,
- COUNT(o1.order_id) as num_of_orders
- FROM `Customer_details.payments` AS p1 INNER JOIN `Customer_details.orders` AS o1 on p1.order_id = o1.order_id
- GROUP BY p1.payment_installments



Row	payment_installment	num_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

- ➢ The above gives the no. of orders placed on the basis of the payment installments.