




























PROJECT- TARGET SQL

1.1Data type of columns in a table

	customers	 QUERY ▾	 SHARE	 COPY	 SNAPSHOT	 DELETE	 EXPORT ▾
<div>SCHEMADETAILSPREVIEW</div>							
<div> Filter Enter property name or value</div>							
<input type="checkbox"/>	Field name	Type	Mode	Collation	Default Value	Policy Tags 	Description
<input type="checkbox"/>	customer_id	STRING	NULLABLE				
<input type="checkbox"/>	customer_unique_id	STRING	NULLABLE				
<input type="checkbox"/>	customer_zip_code_prefix	INTEGER	NULLABLE				
<input type="checkbox"/>	customer_city	STRING	NULLABLE				
<input type="checkbox"/>	customer_state	STRING	NULLABLE				

	geolocation	 QUERY ▾	 SHARE	 COPY	 SNAPSHOT	 DELETE	 EXPORT ▾
<div>SCHEMADETAILSPREVIEW</div>							
<div> Filter Enter property name or value</div>							
<input type="checkbox"/>	Field name	Type	Mode	Collation	Default Value	Policy Tags 	Description
<input type="checkbox"/>	geolocation_zip_code_prefix	INTEGER	NULLABLE				
<input type="checkbox"/>	geolocation_lat	FLOAT	NULLABLE				
<input type="checkbox"/>	geolocation_lng	FLOAT	NULLABLE				
<input type="checkbox"/>	geolocation_city	STRING	NULLABLE				
<input type="checkbox"/>	geolocation_state	STRING	NULLABLE				

	order_items	 QUERY ▾	 SHARE	 COPY	 SNAPSHOT	 DELETE	 EXPORT ▾
<div>SCHEMADETAILSPREVIEW</div>							
<div> Filter Enter property name or value</div>							
<input type="checkbox"/>	Field name	Type	Mode	Collation	Default Value	Policy Tags 	Description
<input type="checkbox"/>	order_id	STRING	NULLABLE				
<input type="checkbox"/>	order_item_id	INTEGER	NULLABLE				
<input type="checkbox"/>	product_id	STRING	NULLABLE				
<input type="checkbox"/>	seller_id	STRING	NULLABLE				
<input type="checkbox"/>	shipping_limit_date	TIMESTAMP	NULLABLE				
<input type="checkbox"/>	price	FLOAT	NULLABLE				
<input type="checkbox"/>	freight_value	FLOAT	NULLABLE				

SCHEMA

DETAILS

PREVIEW

Filter

Enter property name or value

<input type="checkbox"/>	Field name	Type	Mode	Collation	Default Value	Policy Tags	Description
<input type="checkbox"/>	review_id	STRING	NULLABLE				
<input type="checkbox"/>	order_id	STRING	NULLABLE				
<input type="checkbox"/>	review_score	INTEGER	NULLABLE				
<input type="checkbox"/>	review_comment_title	STRING	NULLABLE				
<input type="checkbox"/>	review_creation_date	TIMESTAMP	NULLABLE				
<input type="checkbox"/>	review_answer_timestamp	TIMESTAMP	NULLABLE				

SCHEMA

DETAILS

PREVIEW

Filter

Enter property name or value

<input type="checkbox"/>	Field name	Type	Mode	Collation	Default Value	Policy Tags	Description
<input type="checkbox"/>	order_id	STRING	NULLABLE				
<input type="checkbox"/>	customer_id	STRING	NULLABLE				
<input type="checkbox"/>	order_status	STRING	NULLABLE				
<input type="checkbox"/>	order_purchase_timestamp	TIMESTAMP	NULLABLE				
<input type="checkbox"/>	order_approved_at	TIMESTAMP	NULLABLE				
<input type="checkbox"/>	order_delivered_carrier_date	TIMESTAMP	NULLABLE				
<input type="checkbox"/>	order_delivered_customer_date	TIMESTAMP	NULLABLE				
<input type="checkbox"/>	order_estimated_delivery_date	TIMESTAMP	NULLABLE				

SCHEMA

DETAILS

PREVIEW

Filter

Enter property name or value

<input type="checkbox"/>	Field name	Type	Mode	Collation	Default Value	Policy Tags	Description
<input type="checkbox"/>	order_id	STRING	NULLABLE				
<input type="checkbox"/>	payment_sequential	INTEGER	NULLABLE				
<input type="checkbox"/>	payment_type	STRING	NULLABLE				
<input type="checkbox"/>	payment_installments	INTEGER	NULLABLE				
<input type="checkbox"/>	payment_value	FLOAT	NULLABLE				

Filter Enter property name or value

<input type="checkbox"/>	Field name	Type	Mode	Collation	Default Value	Policy Tags ?	Description
<input type="checkbox"/>	product_id	STRING	NULLABLE				
<input type="checkbox"/>	product_category	STRING	NULLABLE				
<input type="checkbox"/>	product_name_length	INTEGER	NULLABLE				
<input type="checkbox"/>	product_description_length	INTEGER	NULLABLE				
<input type="checkbox"/>	product_photos_qty	INTEGER	NULLABLE				
<input type="checkbox"/>	product_weight_g	INTEGER	NULLABLE				
<input type="checkbox"/>	product_length_cm	INTEGER	NULLABLE				
<input type="checkbox"/>	product_height_cm	INTEGER	NULLABLE				
<input type="checkbox"/>	product_width_cm	INTEGER	NULLABLE				

Filter Enter property name or value

<input type="checkbox"/>	Field name	Type	Mode	Collation	Default Value	Policy Tags ?	Description
<input type="checkbox"/>	seller_id	STRING	NULLABLE				
<input type="checkbox"/>	seller_zip_code_prefix	INTEGER	NULLABLE				
<input type="checkbox"/>	seller_city	STRING	NULLABLE				
<input type="checkbox"/>	seller_state	STRING	NULLABLE				

1.1 Time period for which the data is given

Code-

```
select max(order_purchase_timestamp) as latest_order, min(order_purchase_timestamp) as .earliest_order from `Target_Case_Study.orders`;
```

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

Conclusion - The data related to orders placed between “2016-09-04” to “2018-10-17”

1.2Number of States and Cities of the Customers

Code-

```
select count(distinct customer_state) as StatesCount, count(distinct customer_city) as CitiesCount from `Target_Case_Study.customers`;
```

Row	StatesCount	CitiesCount
1	27	4119

Conclusion - This include data of 27 States and 4119 Cities

1.3.2 List of Distinct States and Cities

Code-

```
select distinct customer_state as State, customer_city as City from `Target_Case_Study.customers`;
```

Row	State	City
1	RN	acu
2	CE	ico
3	RS	ipe
4	CE	ipu
5	SC	ita
6	SP	itu
7	SP	jau
8	MG	luz

2.1.1 Orders and Change in Number of Orders every year

Code-

```
select year, Orders, Orders - lag(Orders) over(order by year) as ChangeInOrders from
(select Extract(year from order_purchase_timestamp) as year, count(order_id) as Orders, from Target_Case_Stu
dy.orders group by year order by year) a order by year;
```

Row	year	Orders	ChangeInOr...
1	2016	329	null
2	2017	45101	44772
3	2018	54011	8910

Here The **Orders** column represents the Number of Orders Placed, and **ChangeInOrder** represents the increase in number of orders compared to Previous Year

Conclusion— The sales increased from 2016 to 2018. It has a clear increasing Trend.

2.1.2 Number of Orders and Change in Number of Orders every Year- Month

Code-

```
select Year, Month, No_Orders, No_Orders -
lag(No_Orders) over(order by Year, Month) as Change_NoOrders from
(select EXTRACT(year FROM order_purchase_timestamp) as Year, EXTRACT(month FROM order_purchase_timestamp) as
Month,
count(order_id) as No_Orders
from Target_Case_Study.orders group by year, month order by year, month) a
```

order by Year, Month;

Row	Year	Month	No_Orders	Change_No...
1	2016	9	4	null
2	2016	10	324	320
3	2016	12	1	-323
4	2017	1	800	799
5	2017	2	1780	980
6	2017	3	2682	902
7	2017	4	2404	-278
8	2017	5	3700	1296
9	2017	6	3245	-455
10	2017	7	4026	781
11	2017	8	4331	305
12	2017	9	4285	-46
13	2017	10	4631	346
14	2017	11	7544	2913
15	2017	12	5673	-1871
16	2018	1	7269	1596

Here the **No_Orders** column represents the number of Orders placed in that Month and Year. And **Change_NoOrders** column represents the change in Number of Orders when compared to previous month. (The above Screenshot shows only the First 16 Rows)

Conclusion – The Purchases kept increasing and reached the peak in 11th month of 2017.

The same month witnessed the maximum jump in purchases when compared to the previous year.
The increased sales in the month of November could be because of the festive seasons

2.2.1 Purchases and Change in Purchases every Hour

Code-

```
select Hour, Orders, Orders - lag(Orders) over(order by Hour) as Change_Orders from
(select EXTRACT (hour FROM order_purchase_timestamp) as Hour, count(order_id) as Orders
from Target_Case_Study.orders group by hour) a
order by Orders desc;
```

Row	Hour	Orders	Change_Ord...
1	16	6675	221
2	11	6578	401
3	14	6569	51
4	13	6518	523
5	15	6454	-115
6	21	6217	24
7	20	6193	211
8	10	6177	1392
9	17	6150	-525
10	12	5995	-583

Here **Orders** column represents the number of Orders placed in that Hour. And the **Change_Orders** column represents the change in number of Orders when compared to the number of Orders placed in the last Hour.

The Busiest hour is 16th. Second Busiest Hour is 11th followed by 14

2.2.2 Purchases and Change in Purchases every Grouped Hour

The Hours were grouped based on the following Criteria -

- * 4-11 Hours = Morning
- * 12-15 Hours = Afternoon
- * 16-20 Hours = Evening
- * 21-03 Hours = Night

Code-

```
select b.Period, round(b.Average_Purchases, 2) as AveragePurchases,
concat(round((b.Average_Purchases/sum(b.Average_Purchases) over(rows between unbounded preceding and unbound
ed following))*100, 2), "%") as Percentage from
```

```
(Select Period, AVG(Purchases) as Average_Purchases from
(Select case when Hour between 4 and 11 then 'Morning'
when Hour between 12 and 15 then 'Afternoon'
when Hour between 16 and 20 then 'Evening'
when Hour between 0 and 3 or Hour between 21 and 23 then 'Night'
end as Period, Purchases
from
(select EXTRACT(hour FROM order_purchase_timestamp) as Hour, count(order_id) as Purchases
from Target_Case_Study.orders group by hour) a)
group by Period
order by Average_Purchases desc) b
order by Percentage desc;
```

Row	Period	AveragePur...	Percentage
1	Afternoon	6384.0	34.89%
2	Evening	6153.8	33.63%
3	Night	2928.86	16.01%
4	Morning	2829.25	15.46%

Here **Average_Purchases** Column represents the average number of purchases made during the particular Period. **Percentage** Column represents the percentage of Customers who placed order in that particular period.

Conclusion – Customers make maximum purchases during the Afternoon period

3.1.1 Month on Month Break down of City and State (Number of Orders placed)

Code-

```
select EXTRACT(Year FROM order_purchase_timestamp) as year,
EXTRACT(Month FROM order_purchase_timestamp) as Month,
customer_state, customer_city, count(order_id) as Orders
from Target_Case_Study.orders o join Target_Case_Study.customers c on o.customer_id = c.customer_id
group by year, month, customer_state, customer_city
order by Orders desc;
```

Row	year	Month	customer_state	customer_city	Orders
1	2018	8	SP	sao paulo	1308
2	2018	5	SP	sao paulo	1222
3	2018	4	SP	sao paulo	1166
4	2018	3	SP	sao paulo	1151
5	2017	11	SP	sao paulo	1118
6	2018	1	SP	sao paulo	1098
7	2018	7	SP	sao paulo	1084
8	2018	6	SP	sao paulo	1054

Conclusion - The above table is about the number of orders placed each year, each month, and in each State and City respectively.

The highest number of orders where in the 8th month of 2018 in the City – Sao Paulo, the number of Orders where = 1308.

3.1.2 Number of Orders placed from difference State and Cities in the year 2016, 2017 and 2018, along with the growth percentage(from 2017 to 2018)

Code-

```
with years as
(select EXTRACT(Year FROM o.order_purchase_timestamp) as y, customer_state, customer_city, count(order_id) as Orders
from Target_Case_Study.orders o join Target_Case_Study.customers c on o.customer_id = c.customer_id
group by y, customer_state, customer_city
order by Orders desc)

select y6.customer_state, y6.customer_city, y6.orders as orders_2016, y7.orders as orders_2017,
y8.orders as orders_2018, concat( round(((y8.orders -
y7.orders)/y7.orders)*100, 2), "%") as growth_percent from years y6
join years y7 on y6.customer_city = y7.customer_city
join years y8 on y8.customer_city = y7.customer_city
where y6.y = 2016 and y7.y = 2017 and y8.y = 2018
order by orders_2018 desc, orders_2017 desc, orders_2016 desc;
```

Row	customer_state	customer_city	orders_2016	orders_2017	orders_2018	growth_percent
1	SP	sao paulo	36	6381	9123	42.97%
2	RJ	rio de janeiro	38	3341	3503	4.85%
3	MG	belo horizonte	12	1206	1555	28.94%
4	DF	brasilia	6	912	1213	33%
5	PR	curitiba	6	651	864	32.72%
6	SP	campinas	7	618	819	32.52%
7	SP	guarulhos	1	493	695	40.97%
8	RS	porto alegre	7	691	681	-1.45%
9	SP	sao bernardo do campo	3	397	538	35.52%
10	RJ	niteroi	1	352	496	40.91%

Here the columns **orders_2016**, **orders_2017**, **orders_2018** represent the number of orders placed in the year 2016, 2017 and 2018 respectively. The column **growth_percent** represents the **percentage increase in orders from 2017 to 2018**.

Conclusion – Maximum number of orders in the year 2018 and 2017 was from Sao Paulo City. In the year 2016 it was from Rio De Janeiro.

But if we consider the Percentage growth over the years 2017 to 2018, Praia Grande (18000%) and Mesquita (4500%)

have best figures.

Cities with declining growth are Santo Andre (-99.71%) and Sao Carlos (-99.05%).

3.1.3 Top performing State Overall (Considering number of Orders placed)

Code-

```
select a.customer_state, a.Orders,
concat(round(((a.Orders/sum(a.Orders) over(rows between unbounded preceding and unbounded following))*100),
2), "%") as Percentge from
(select customer_state, count(order_id) as Orders
from Target_Case_Study.orders o join Target_Case_Study.customers c on o.customer_id = c.customer_id
group by customer_state
order by Orders desc)a
order by a.orders desc;
```

Row	customer_state	Orders	Percentge
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%

Conclusion - The maximum number of orders are from the SP state, which accounts for 41.98% of total orders. They are followed by RJ and MG states which have 13% and 12% of total Orders.

3.1.4 Top performing City Overall (Considering number of Orders placed)

Code-

```
select a.customer_city, a.Orders,
concat(round((a.Orders/sum(a.Orders) over(rows between unbounded preceding and unbounded following))*100, 2)
, "%") as Percentge from
(select customer_city, count(order_id) as Orders
from Target_Case_Study.orders o join Target_Case_Study.customers c on o.customer_id = c.customer_id
group by customer_city)a
order by a.orders desc;
```

Row	customer_city	Orders	Percentge
1	sao paulo	15540	15.63%
2	rio de janeiro	6882	6.92%
3	belo horizonte	2773	2.79%
4	brasilia	2131	2.14%
5	curitiba	1521	1.53%
6	campinas	1444	1.45%
7	porto alegre	1379	1.39%
8	salvador	1245	1.25%
9	guarulhos	1189	1.2%
10	sao bernardo do campo	938	0.94%
11	niteroi	849	0.85%
12	santo andre	797	0.8%
13	osasco	746	0.75%
14	santos	713	0.72%

Conclusion - When we consider the cities, most of the orders are from Sao Paulo, which accounts for 15.63% of total number of orders. It is followed by Rio De Janeiro with 6.92% of total number of Orders.

3.2 Spread of Customers (Considering the customer_id)

Code-

```
select a.customer_state, a.customer_city, a.Customers,
       round((a.Customers / sum(a.Customers) over(rows between unbounded preceding and unbounded following))*100,
2) as percentage from
(select customer_state, customer_city, count(customer_id) as Customers
from Target_Case_Study.customers c
group by customer_state, customer_city
order by Customers desc) a
order by a.Customers desc;
```

Row	customer_state	customer_city	Customers	percentage
1	SP	sao paulo	15540	15.63
2	RJ	rio de janeiro	6882	6.92
3	MG	belo horizonte	2773	2.79
4	DF	brasilia	2131	2.14
5	PR	curitiba	1521	1.53
6	SP	campinas	1444	1.45
7	RS	porto alegre	1379	1.39

Conclusion – Most of the customers are from Sao Paulo City, they account for 15.63% of the customers. Next is Rio De Janeiro, which accounts for 6.92% of the customers.

Extract year, month, group by year, month, average price, average freight, where year = 2018 to 2018

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

Code-

```
with items as
(select extract(year FROM o.order_purchase_timestamp) as year,
       oi.order_id,
       count(distinct order_item_id) as no_items,
       sum(freight_value) as total_freight,
       sum(price) as total_price
from `Target_Case_Study.order_items` oi join `Target_Case_Study.orders` o on oi.order_id =
o.order_id
where (o.order_purchase_timestamp >= '2017-01-01 00:00:00 UTC' and
o.order_purchase_timestamp <= '2017-08-31 00:00:00 UTC') or
(o.order_purchase_timestamp >= '2018-01-01 00:00:00 UTC' and
o.order_purchase_timestamp <= '2018-08-31 00:00:00 UTC')
group by extract(year FROM o.order_purchase_timestamp), order_id)

select i7.no_items, round(avg(i7.total_freight), 2) as avgFreight17,
round(avg(i8.total_freight), 2) as avgFreight18,
```

```
concat(round((avg(i8.total_freight) -
  avg(i7.total_freight))/avg(i7.total_freight)*100, 2), '%') as IncFreight,
round(avg(i7.total_price), 2) as avgPrice17,
round(avg(i8.total_price), 2) as avgPrice18,
concat (round((avg(i8.total_price)- avg(i7.total_price))/avg(i7.total_price)*100, 2), '%') as IncPrice
from items i7
join items i8 on i7.no_items = i8.no_items
where i7.year = 2017 and i8.year = 2018
group by i7.no_items
order by i7.no_items;
```

Row	no_items	avgFreight17	avgFreight18	IncFreight	avgPrice17	avgPrice18	IncPrice
1	1	19.58	20.91	6.8%	131.01	129.85	-0.88%
2	2	36.93	37.58	1.76%	167.08	177.19	6.05%
3	3	57.55	56.02	-2.66%	247.43	240.89	-2.64%
4	4	74.03	74.6	0.76%	360.9	296.88	-17.74%
5	5	74.52	95.01	27.5%	362.37	405.04	11.78%
6	6	118.4	123.51	4.32%	300.58	429.52	42.9%
7	7	96.56	139.09	44.05%	438.5	470.4	7.28%
8	20	288.8	202.4	-29.92%	1974.0	2000.0	1.32%

The table shows number of items in an order (**no_items**), and its average Freight Value in the year 2017 (**avgFreight17**), average Freight Value in the year 2018 (**avgFreight18**), Percentage increase in Freight (**IncFreight**), average Price in the year 2017 (**avgPrice17**), average Price in the year 2018 (**avgPrice17**) and Percentage increase in Price (**IncPrice**). Even though this value also depends on the products purchased as well, this table gives us an understanding of relationship between the cost and number of items.

Conclusion – The average freight value for the order with number of items as 7 had the maximum increase (44.05%). Where as the average Freight value for orders with number of items as 20 had the maximum decrease (-29.92%) The Average price for the orders with number of items as 6 had the maximum increase (42.9%) And the Average price for the orders with number of items as 4 had the maximum decrease (-17.74%)

4.2.1 Mean & Sum of price value by customer state

Code-

```
with state as (
  select extract(year FROM o.order_purchase_timestamp) as year, c.customer_state as state,
  sum(oi.price) as priceSum, avg(oi.price) as priceAvg
  from `Target_Case_Study.customers` c join `Target_Case_Study.orders` o on c.customer_id = o.customer_id
  join `Target_Case_Study.order_items` oi on o.order_id = oi.order_id
  group by extract(year FROM o.order_purchase_timestamp), c.customer_state
)
select y7.state, round(y7.priceSum, 2) as priceSum17, round(y8.priceSum, 2) as priceSum18,
round(y7.priceAvg, 2) as priceAvg17, round(y8.priceAvg, 2) as priceAvg18,
concat(round((y8.priceSum - y7.priceSum)/y7.priceSum, 2), "%") as IncSum,
concat(round((y8.priceAvg - y7.priceAvg)/y7.priceAvg, 2), "%") as IncAvg
from state y7 join state y8 on y7.state = y8.state
where y7.year = 2017 and y8.year = 2018
order by priceAvg18 desc, priceAvg17 desc;
```

Row	state	priceSum17	priceSum18	priceAvg17	priceAvg18	IncSum	IncAvg
1	RR	1404.76	6312.08	73.93	210.4	3.49%	1.85%
2	PB	51903.37	63314.81	178.36	204.24	0.22%	0.15%
3	RO	24577.45	21563.19	153.61	182.74	-0.12%	0.19%
4	RN	35459.84	46846.45	136.91	176.78	0.32%	0.29%
5	AP	6046.64	7427.66	155.04	172.74	0.23%	0.11%
6	PI	36484.88	50219.2	147.71	170.81	0.38%	0.16%
7	AL	43805.64	36426.68	196.44	166.33	-0.17%	-0.15%
8	AC	10667.95	5315.0	177.8	166.09	-0.5%	-0.07%
9	PA	91004.93	86855.28	165.16	166.07	-0.05%	0.01%
10	SE	31387.3	27248.1	147.36	161.23	-0.13%	0.09%
11	TO	23930.62	25691.12	159.54	155.7	0.07%	-0.02%
12	CE	112865.61	112699.72	156.32	150.47	-0%	-0.04%
13	MA	60232.82	58711.14	143.41	148.64	-0.03%	0.04%
14	PE	123588.05	137830.88	144.21	146.32	0.12%	0.01%
15	MT	77650.22	78475.52	158.79	139.39	0.01%	-0.12%

In the above table **priceSum17** and **priceSum18** represents the Sum of all the prices in the year 2017 and 2018 for their respective states. Similarly, **priceAvg17** and **priceAvg18** represent the average prices in the year 2017 and 2018. **IncSum** and **IncAvg** values represent the percentage increase in Sum and Average.

Conclusion – RR is the state with maximum Average price (210.4) in the year 2018.
AL is the state with maximum Average price (196.44) in the year 2017.

4.2.2 Mean & Sum of freight value by customer state

Code-

```
with state as (
  select extract(year FROM o.order_purchase_timestamp) as year, c.customer_state as state,
  sum(oi.freight_value) as freightSum, avg(oi.freight_value) as freightAvg
  from `Target_Case_Study.customers` c join `Target_Case_Study.orders` o on c.customer_id = o.customer_id
  join `Target_Case_Study.order_items` oi on o.order_id = oi.order_id
  group by extract(year FROM o.order_purchase_timestamp), c.customer_state
)
select y7.state, round(y7.freightSum, 2) as freightSum17, round(y8.freightSum, 2) as freightSum18,
round(y7.freightAvg, 2) as freightAvg17, round(y8.freightAvg, 2) as freightAvg18,
concat(round((y8.freightSum - y7.freightSum)/y7.freightSum, 2), "%") as IncSum,
concat(round((y8.freightAvg - y7.freightAvg)/y7.freightAvg, 2), "%") as IncAvg
from state y7 join state y8 on y7.state = y8.state
where y7.year = 2017 and y8.year = 2018
order by freightAvg18, freightAvg17;
```

Row	state	freightSum17	freightSum18	freightAvg17	freightAvg18	IncSum	IncAvg
1	RR	601.82	1540.71	31.67	51.36	1.56%	0.62%
2	PB	10966.83	14728.06	37.69	47.51	0.34%	0.26%
3	RO	6326.44	5090.94	39.54	43.14	-0.2%	0.09%
4	PI	8580.62	12601.49	34.74	42.86	0.47%	0.23%
5	AC	2318.65	1368.1	38.64	42.75	-0.41%	0.11%
6	TO	4766.41	6966.27	31.78	42.22	0.46%	0.33%
7	MA	14973.19	16255.99	35.65	41.15	0.09%	0.15%
8	RN	8435.54	10271.91	32.57	38.76	0.22%	0.19%
9	SE	7577.05	6473.95	35.57	38.31	-0.15%	0.08%
10	PA	18640.94	19862.87	33.83	37.98	0.07%	0.12%
11	AL	7756.37	8110.81	34.78	37.04	0.05%	0.06%
12	AM	2416.63	3062.26	30.59	35.61	0.27%	0.16%
13	AP	1284.89	1503.61	32.95	34.97	0.17%	0.06%
14	CE	22400.53	25690.66	31.03	34.3	0.15%	0.11%
15	PE	27107.88	32022.39	31.63	33.99	0.18%	0.07%

In the above table **freightSum17** and **freightSum18** represents the Sum of all the prices in the year 2017 and 2018 for their respective states. Similarly, **freightAvg17** and **freightAvg18** represent the average prices in the year 2017 and 2018.

IncSum and **IncAvg** values represent the percentage increase in Sum and Average.

Conclusion – RR is the state with maximum Average price (51.36) in the year 2018.
RO is the state with maximum Average price (39.54) in the year 2017.

4.2.3 Increase in cost of Freight values for different categories based on weight

Code-

```
with weiCat as
(select extract(year FROM o.order_purchase_timestamp) as year,
       Ntile(4) over(order by product_weight_g) as weightCat,
       freight_value
 from `Target_Case_Study.products` p join `Target_Case_Study.order_items` oi
  on p.product_id = oi.product_id
 join `Target_Case_Study.orders` o
  on o.order_id = oi.order_id)

select w1.weightCat, round(avg(w1.freight_value), 2) as freightV17,
       round(avg(w2.freight_value),2) as freightV18,
       concat(round(((avg(w2.freight_value) -
       avg(w1.freight_value))/avg(w1.freight_value))*100, 2), "%") as perIncrease
from weiCat w1 join weiCat w2 on w1.weightCat = w2.weightCat
where w1.year = 2017 and w2.year = 2018
group by weightCat
order by weightCat;
```

Row	weightCat	freightV17	freightV18	perIncrease
1	1	14.55	14.9	2.36%
2	2	15.57	16.63	6.78%
3	3	17.41	18.76	7.75%
4	4	29.39	32.32	9.97%

Here all the weights of orders are categorized into 4 categories (Based on quantiles, such that the heavier object falls into bigger quantile). The column **weightCat** represents the category, **freightv17** and **freightv18** represent the average freight value for the categories in the year 2017 and 2018.
perIncrease represents the percentage increase in the cost of the freight values of the respective categories.

Conclusion – Maximum increase was seen in the 4th category of orders followed by 3rd category.

4.2.4 Increase in cost of Freight values for different categories based on Volume

Code-

```
with volCat as
(select extract(year FROM o.order_purchase_timestamp) as year,
       Ntile(4) over(order by product_length_cm*product_height_cm*product_width_cm) as volumeCat,
       freight_value
 from `Target_Case_Study.products` p join `Target_Case_Study.order_items` oi
  on p.product_id = oi.product_id
 join `Target_Case_Study.orders` o
  on o.order_id = oi.order_id)

select w1.volumeCat, round(avg(w1.freight_value), 2) as freightV17,
       round(avg(w2.freight_value),2) as freightV18,
       concat(round(((avg(w2.freight_value) -
       avg(w1.freight_value))/avg(w1.freight_value))*100, 2), "%") as perIncrease
from volCat w1 join volCat w2 on w1.volumeCat = w2.volumeCat
where w1.year = 2017 and w2.year = 2018
group by volumeCat
```

order by volumeCat;

Row	volumeCat	freightV17	freightV18	perIncrease
1	1	15.09	15.21	0.75%
2	2	15.92	17.26	8.4%
3	3	18.54	19.63	5.86%
4	4	26.91	31.05	15.41%

Here all the Volumes of orders are categorized into 4 categories (Based on quantiles, such that the object with bigger volume falls into bigger quantile). The column **volumeCat** represents the category, **freightv17** and **freightv18** represent the average freight value for the categories in the year 2017 and 2018.

perIncrease represents the percentage increase in the cost of the freight values of the respective categories.

Conclusion - Maximum increase was seen in the 4th category of orders followed by 2nd category.

5. Analysis on sales, freight, and delivery time

5.1. Calculate days between purchasing, delivering and estimated delivery

Code-

```
select order_id,
date_diff(order_delivered_customer_date , order_purchase_timestamp, day) as time_to_delivery,
date_diff(order_estimated_delivery_date, order_delivered_customer_date, day) as diff_estimated_delivery,
from `Target_Case_Study.orders` as o
where order_delivered_customer_date is not null and order_purchase_timestamp is not null and
order_estimated_delivery_date is not null;
```

Row	order_id	time_to_deli...	diff_estimat...
1	1950d777989f6a877539f5379...	30	-12
2	2c45c33d2f9cb8ff8b1c86cc28...	30	28
3	65d1e226dfaeb8cdc42f66542...	35	16
4	635c894d068ac37e6e03dc54e...	30	1
5	3b97562c3aee8bdedcb5c2e45...	32	0
6	68f47f50f04c4cb6774570cfde...	29	1
7	276e9ec344d3bf029ff83a161c...	43	-4
8	54e1a3c2b97fb0809da548a59...	40	-4
9	fd04fa4105ee8045f6a0139ca5...	37	-1
10	302bb8109d097a9fc6e9cefc5...	33	-5
11	66057d37308e787052a32828...	38	-6
12	19135c945c554eebfd7576c73...	36	-2
13	4493e45e7ca1084efcd38ddeb...	34	0
14	70c77e51e0f179d75a64a6141...	42	-11
15	d7918e406132d7c81f1b84527...	35	-3
16	43f6604e77ce6433e7d68dd86...	32	-7

5.2 Grouping the Data by state and Average value of Time to deliver, Estimated time and Extra time taken to deliver

Code-

```
select customer_state,
round(avg(date_diff(order_delivered_customer_date , order_purchase_timestamp, day)), 2) as AvgActualTime,
round(avg(date_diff(order_estimated_delivery_date, order_purchase_timestamp, day)), 2) as AvgEstimatedTime,

round(avg(date_diff(order_delivered_customer_date, order_estimated_delivery_date, day)), 2) as AvgExtraTimeTaken
from `Target_Case_Study.orders` as o
join `Target_Case_Study.customers` as c on o.customer_id = c.customer_id
where order_delivered_customer_date is not null and order_purchase_timestamp is not null and
order_estimated_delivery_date is not null
group by customer_state
order by AvgExtraTimeTaken;
```

Row	customer_state	AvgActualTi...	AvgEstimat...	AvgExtraTi...
1	AC	20.64	40.73	-19.76
2	RO	18.91	38.39	-19.13
3	AP	26.73	45.87	-18.73
4	AM	25.99	44.92	-18.61
5	RR	28.98	45.63	-16.41
6	MT	17.59	31.37	-13.43
7	PA	23.32	36.79	-13.19
8	RS	14.82	28.16	-12.98
9	RN	18.82	31.87	-12.76
10	PE	17.97	30.69	-12.4
11	PB	19.95	32.65	-12.37
12	PR	11.53	24.25	-12.36
13	MG	11.54	24.19	-12.3
14	GO	15.15	26.72	-11.27
15	TO	17.23	28.73	-11.26

- The column AvgActualTime denotes the Average value of difference between Date on which the Order was delivered and the Date on which the Purchase was made.
- The column AvgEstimatedTime denotes the Average value of difference between Estimated delivery date and Date on which the Purchase was made.
- The Column AvgExtraTimeTaken denotes the Average value of difference between Date on which Order was Delivered and Estimated delivery date.

Conclusion – States with Quickest Average delivery time – AC, RO, AP, AM, RR, MT (Based on extra time)
States with Slowest Average delivery time – AL, MA, SE, ES

5.3 States with highest number of Delayed orders

Code-

```
select customer_state,
sum(case when date_diff(order_estimated_delivery_date,order_delivered_customer_date, day)<0
then 1
else 0
end) as delayed_orders
from `Target_Case_Study.orders` as o
join `Target_Case_Study.customers` as c on o.customer_id = c.customer_id
where order_delivered_customer_date is not null and
order_estimated_delivery_date is not null
group by customer_state
order by delayed_orders desc;
```

Row	customer_state	delayed_ord...
1	SP	1820
2	RJ	1495
3	MG	520
4	BA	396
5	RS	325
6	SC	291
7	ES	214
8	PR	199
9	CE	176
10	PE	153
11	GO	128
12	MA	125
13	DF	118
14	PA	106
15	AL	85
16	MS	68

Conclusion – State MP has the highest number of delayed orders (1820) and followed by RJ (1495)

5.4.1.1 States with least Freight Value

Code-

```
select c.customer_state, round(avg(oi.freight_value), 2) as AvgFreight
from `Target_Case_Study.customers` c join `Target_Case_Study.orders` o on c.customer_id = o.customer_id
join `Target_Case_Study.order_items` oi on oi.order_id = o.order_id
group by c.customer_state
order by AvgFreight
limit 5;
```

Row	customer_state	AvgFreight
1	SP	15.15
2	PR	20.53
3	MG	20.63
4	RJ	20.96
5	DF	21.04

5.4.1.2 States with highest Freight values

Code-

```
select c.customer_state, round(avg(oi.freight_value), 2) as AvgFreight
from `Target_Case_Study.customers` c join `Target_Case_Study.orders` o on c.customer_id = o.customer_id
join `Target_Case_Study.order_items` oi on oi.order_id = o.order_id
group by c.customer_state
order by AvgFreight desc
limit 5;
```

Row	customer_state	AvgFreight
1	RR	42.98
2	PB	42.72
3	RO	41.07
4	AC	40.07
5	PI	39.15

5.4.2.1 States with highest Average Time to delivery

Code-

```
select c.customer_state, round(avg(date_diff(order_delivered_customer_date , order_purchase_timestamp, day))
, 2) as Avg_time_to_delivery
from `Target_Case_Study.customers` c join `Target_Case_Study.orders` o on c.customer_id = o.customer_id
group by c.customer_state
order by Avg_time_to_delivery desc
limit 5;
```

Row	customer_state	Avg_time_to...
1	RR	28.98
2	AP	26.73
3	AM	25.99
4	AL	24.04
5	PA	23.32

5.4.2.2 States with Lowest Average Time to delivery

Code-

```
select c.customer_state, round(avg(date_diff(order_delivered_customer_date , order_purchase_timestamp, day))
, 2) as Avg_time_to_delivery
from `Target_Case_Study.customers` c join `Target_Case_Study.orders` o on c.customer_id = o.customer_id
group by c.customer_state
order by Avg_time_to_delivery
limit 5;
```

Row	customer_state	Avg_time_to...
1	SP	8.3
2	PR	11.53
3	MG	11.54
4	DF	12.51
5	SC	14.48

5.4.3.1 States with Quickest delivery (before Delivery date)

Code-

```
select c.customer_state, round(avg(date_diff(order_estimated_delivery_date, order_delivered_customer_date, day)), 2) as Diff_estimated_delivery
from `Target_Case_Study.customers` c join `Target_Case_Study.orders` o on c.customer_id = o.customer_id
group by c.customer_state
order by Diff_estimated_delivery desc
limit 5;
```


Row	customer_state	Diff_estimat...
1	AC	19.76
2	RO	19.13
3	AP	18.73
4	AM	18.61
5	RR	16.41

5.4.3.2 States with Not so quick delivery (before Delivery date, bottom 5)

Code-

```
select c.customer_state, round(avg(date_diff(order_estimated_delivery_date, order_delivered_customer_date, day)), 2) as Diff_estimated_delivery
from `Target_Case_Study.customers` c join `Target_Case_Study.orders` o on c.customer_id = o.customer_id
group by c.customer_state
order by Diff_estimated_delivery
limit 5;
```

Row	customer_state	Diff_estimat...
1	AL	7.95
2	MA	8.77
3	SE	9.17
4	ES	9.62
5	BA	9.93

6.1 Month over Month count of orders for different payment types

Code-

```
with pt as (select extract(year FROM o.order_purchase_timestamp) as year,
extract(month FROM o.order_purchase_timestamp) as month,
payment_type,
count(payment_type) as type_count
from `Target_Case_Study.orders` o join `Target_Case_Study.payments` p
on o.order_id = p.order_id
group by year, month, payment_type
order by year, month, payment_type)

select pt1.year, pt1.month, pt1.type_count as CreditCard,
concat(round((pt1.type_count/(pt2.type_count + pt3.type_count+pt4.type_count+pt1.type_count))*100, 2), "%")
as CCShare,
pt2.type_count as UPI,
concat(round((pt2.type_count/(pt2.type_count + pt3.type_count+pt4.type_count+pt1.type_count))*100, 2), "%")
as UPIShare,
pt3.type_count as DebitCard,
concat(round((pt3.type_count/(pt2.type_count + pt3.type_count+pt4.type_count+pt1.type_count))*100, 2), "%")
as DCShare,
pt4.type_count as Voucher,
concat(round((pt4.type_count/(pt2.type_count + pt3.type_count+pt4.type_count+pt1.type_count))*100, 2), "%")
as VShare
from pt pt1 join pt pt2 on pt1.year = pt2.year and pt1.month = pt2.month
join pt pt3 on pt1.year = pt3.year and pt1.month = pt3.month
join pt pt4 on pt1.year = pt4.year and pt1.month = pt4.month
where pt1.payment_type = 'credit_card' and pt2.payment_type = 'UPI'
and pt3.payment_type = 'debit_card' and pt4.payment_type = 'voucher'
```

order by year, month;

Row		month	CreditCard	CCShare	UPI	UPIShare	DebitCard	DCShare	Voucher	VShare
1	2016	10	254	74.27%	63	18.42%	2	0.58%	23	6.73%
2	2017	1	583	68.59%	197	23.18%	9	1.06%	61	7.18%
3	2017	2	1356	71.9%	398	21.1%	13	0.69%	119	6.31%
4	2017	3	2016	71.06%	590	20.8%	31	1.09%	200	7.05%
5	2017	4	1846	71.8%	496	19.29%	27	1.05%	202	7.86%
6	2017	5	2853	72.34%	772	19.57%	30	0.76%	289	7.33%
7	2017	6	2463	71.68%	707	20.58%	27	0.79%	239	6.96%
8	2017	7	3086	71.48%	845	19.57%	22	0.51%	364	8.43%
9	2017	8	3284	72.18%	938	20.62%	34	0.75%	294	6.46%
10	2017	9	3283	72.7%	903	20%	43	0.95%	287	6.36%
11	2017	10	3524	72.51%	993	20.43%	52	1.07%	291	5.99%
12	2017	11	5897	75%	1509	19.19%	70	0.89%	387	4.92%
13	2017	12	4377	74.25%	1160	19.68%	64	1.09%	294	4.99%
14	2018	1	5520	72.99%	1518	20.07%	109	1.44%	416	5.5%

6.2.1 Distribution of payment instalments and count of orders

Code-

```
select payment_installments, count(order_id) NoOrders FROM `Target_Case_Study.payments`  
group by payment_installments order by count(order_id) desc;
```

Row	payment_in...	NoOrders
1	1	52546
2	2	12413
3	3	10461
4	4	7098
5	10	5328
6	5	5239
7	8	4268
8	6	3920
9	7	1626
10	9	644
11	12	133
12	15	74

Code-

```
select payment_installments, count(order_id) NoOrders FROM `Target_Case_Study.payments`  
group by payment_installments order by count(order_id);
```

Row	payment_in...	NoOrders
1	22	1
2	23	1
3	0	2
4	21	3
5	16	5
6	17	8
7	14	15
8	13	16
9	20	17
10	24	18
11	11	23
12	18	27