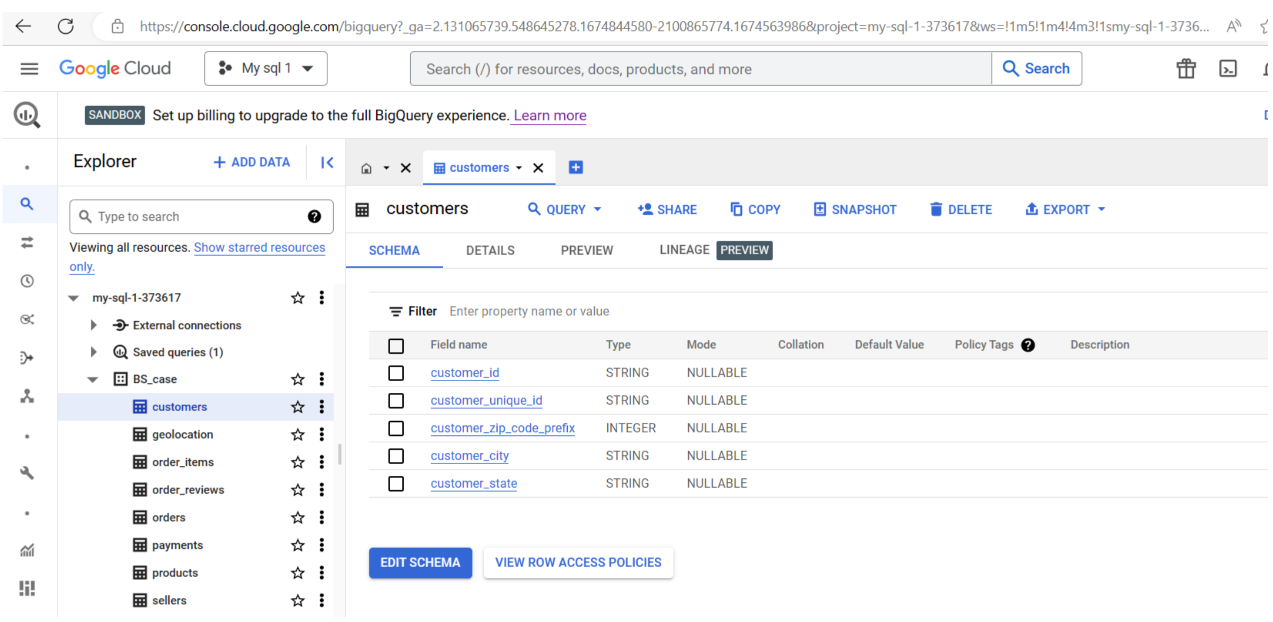
Business Case: Target SQL

1.Checking the structure & characteristics of the dataset

1. *)Data type of columns in a table

Query – Desc BS_case.customers



https://console.cloud.google.com/bigquery?_ga=2.131065739.548645278.1674844580-2100865774.1674563986&project=my-sql-1-373617&ws=!1m5!1m4!4m3!1smy-sql-1-3736...

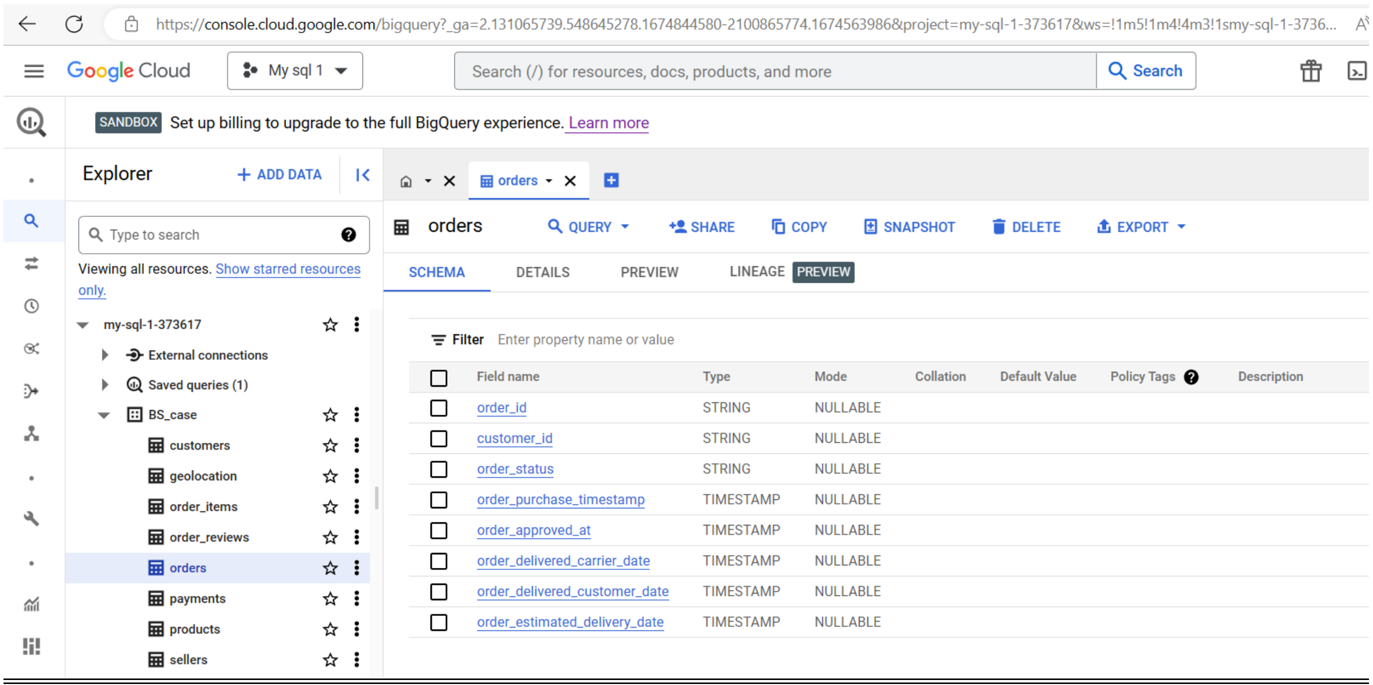
SANDBOX Set up billing to upgrade to the full BigQuery experience. Learn more

Explorer: my-sql-1-373617 > External connections; Saved queries (1); BS_case > customers, geolocation, order_items, order_reviews, orders, payments, products, sellers

customers — SCHEMA

Field name	Type	Mode	Collation	Default Value	Policy Tags	Description
customer_id	STRING	NULLABLE				
customer_unique_id	STRING	NULLABLE				
customer_zip_code_prefix	INTEGER	NULLABLE				
customer_city	STRING	NULLABLE				
customer_state	STRING	NULLABLE				

Query - BS_case.orders;



https://console.cloud.google.com/bigquery?_ga=2.131065739.548645278.1674844580-2100865774.1674563986&project=my-sql-1-373617&ws=!1m5!1m4!4m3!1smy-sql-1-3736...

SANDBOX Set up billing to upgrade to the full BigQuery experience. Learn more

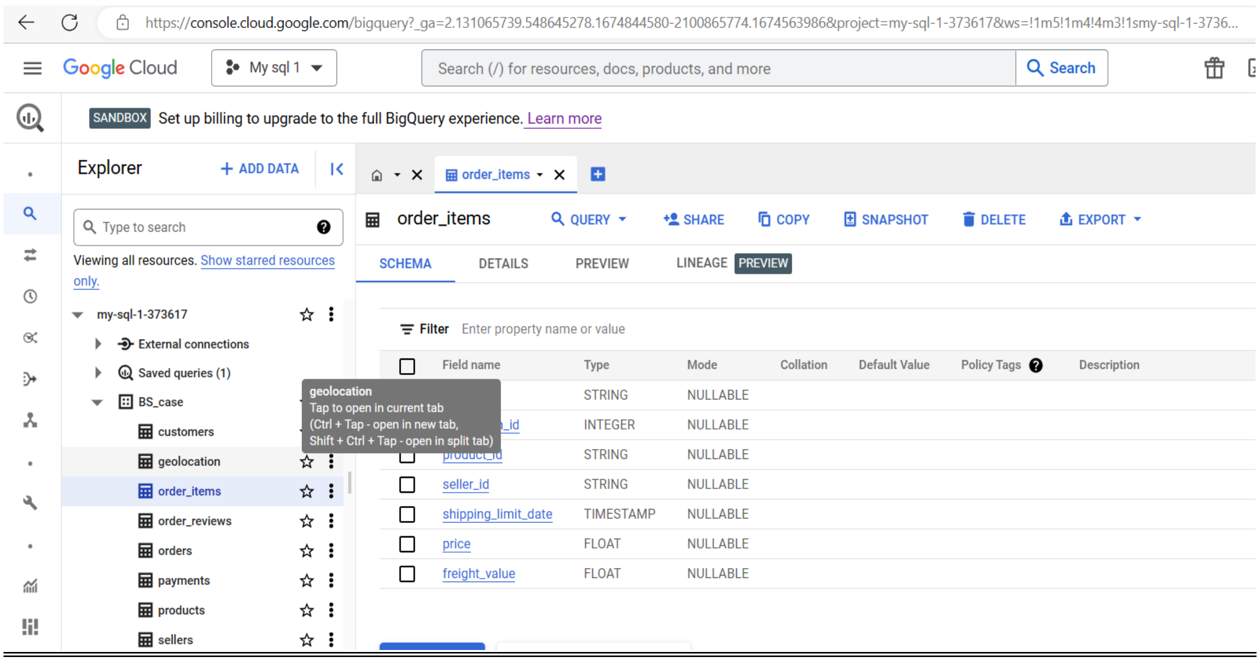
Explorer: my-sql-1-373617 > External connections; Saved queries (1); BS_case > customers, geolocation, order_items, order_reviews, orders, payments, products, sellers

orders — SCHEMA

Field name	Type	Mode	Collation	Default Value	Policy Tags	Description
order_id	STRING	NULLABLE				
customer_id	STRING	NULLABLE				
order_status	STRING	NULLABLE				
order_purchase_timestamp	TIMESTAMP	NULLABLE				
order_approved_at	TIMESTAMP	NULLABLE				
order_delivered_carrier_date	TIMESTAMP	NULLABLE				
order_delivered_customer_date	TIMESTAMP	NULLABLE				
order_estimated_delivery_date	TIMESTAMP	NULLABLE				

<u>Query –</u>

BS_case.order_items;



https://console.cloud.google.com/bigquery?_ga=2.131065739.548645278.1674844580-2100865774.1674563986&project=my-sql-1-373617&ws=!1m5!1m4!4m3!1smy-sql-1-3736...

SANDBOX Set up billing to upgrade to the full BigQuery experience. Learn more

Explorer

Viewing all resources. Show starred resources only.

- my-sql-1-373617
  - External connections
  - Saved queries (1)
  - BS_case
    - customers
    - geolocation
    - order_items
    - order_reviews
    - orders
    - payments
    - products
    - sellers

geolocation
Tap to open in current tab
(Ctrl + Tap - open in new tab,
Shift + Ctrl + Tap - open in split tab)

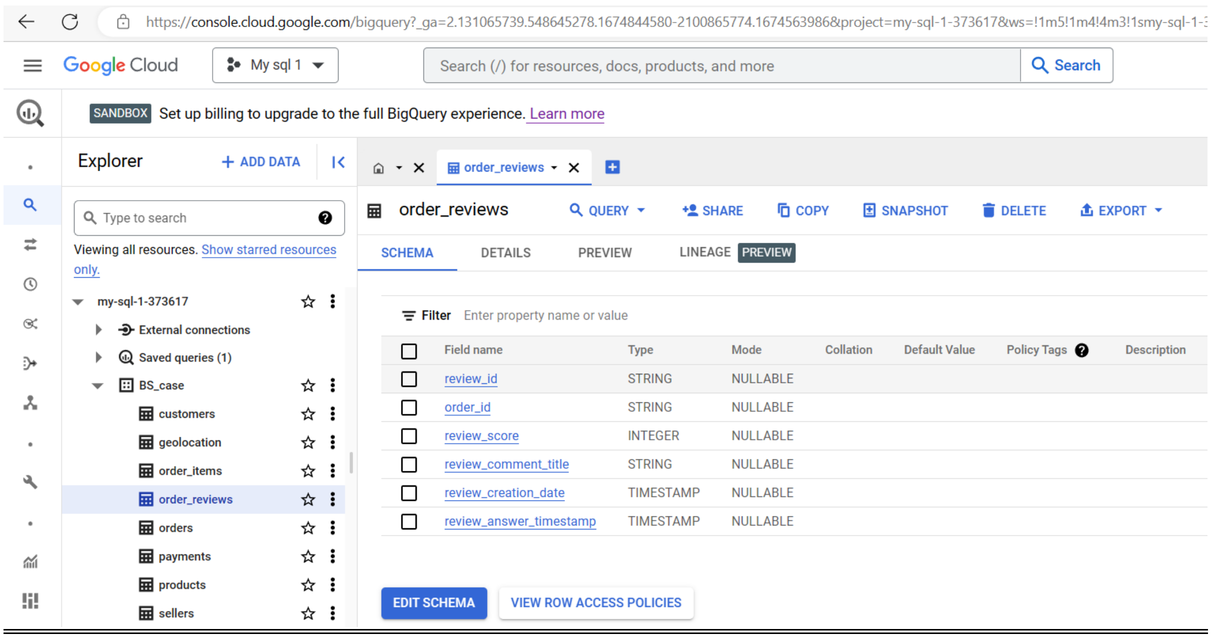
order_items

SCHEMA DETAILS PREVIEW LINEAGE PREVIEW

Field name	Type	Mode	Collation	Default Value	Policy Tags	Description
	STRING	NULLABLE				
_id	INTEGER	NULLABLE				
product_id	STRING	NULLABLE				
seller_id	STRING	NULLABLE				
shipping_limit_date	TIMESTAMP	NULLABLE				
price	FLOAT	NULLABLE				
freight_value	FLOAT	NULLABLE				

<u>Query –</u>

BS_case.order_reviews;



https://console.cloud.google.com/bigquery?_ga=2.131065739.548645278.1674844580-2100865774.1674563986&project=my-sql-1-373617&ws=!1m5!1m4!4m3!1smy-sql-1-3

SANDBOX Set up billing to upgrade to the full BigQuery experience. Learn more

Explorer

Viewing all resources. Show starred resources only.

- my-sql-1-373617
  - External connections
  - Saved queries (1)
  - BS_case
    - customers
    - geolocation
    - order_items
    - order_reviews
    - orders
    - payments
    - products
    - sellers

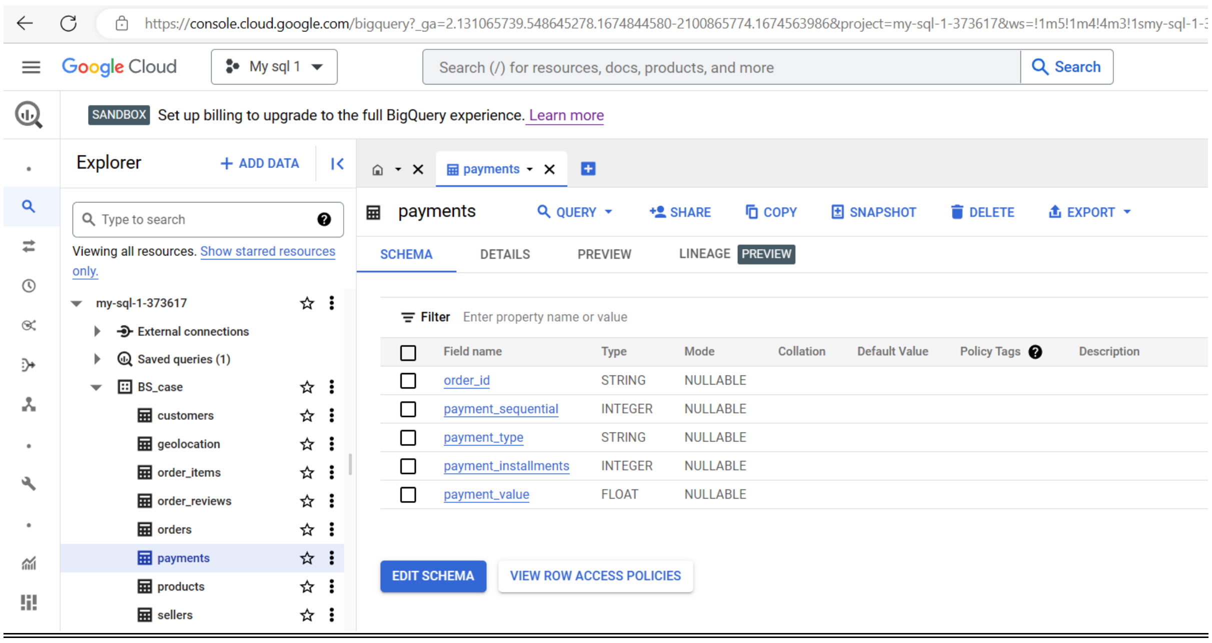
order_reviews

SCHEMA DETAILS PREVIEW LINEAGE PREVIEW

Field name	Type	Mode	Collation	Default Value	Policy Tags	Description
review_id	STRING	NULLABLE				
order_id	STRING	NULLABLE				
review_score	INTEGER	NULLABLE				
review_comment_title	STRING	NULLABLE				
review_creation_date	TIMESTAMP	NULLABLE				
review_answer_timestamp	TIMESTAMP	NULLABLE				

EDIT SCHEMA VIEW ROW ACCESS POLICIES

BS_case.payments;



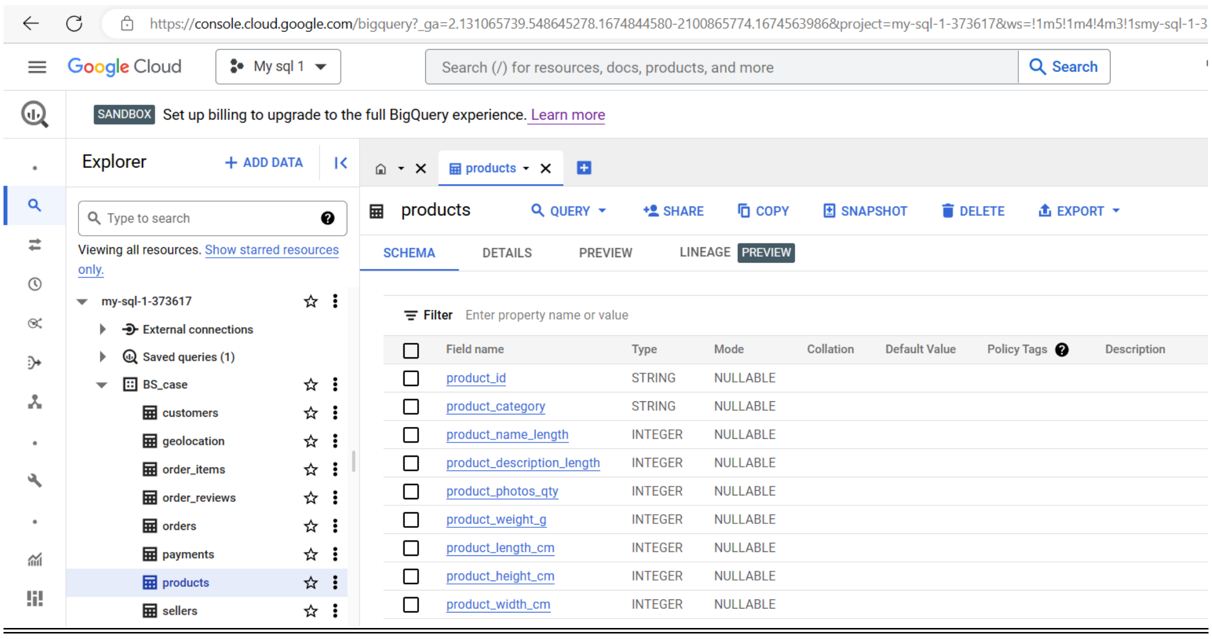
https://console.cloud.google.com/bigquery?_ga=2.131065739.548645278.1674844580-2100865774.1674563986&project=my-sql-1-373617&ws=!1m5!1m4!4m3!1smy-sql-1-3

Set up billing to upgrade to the full BigQuery experience. Learn more

Viewing all resources. Show starred resources only.

Field name	Type	Mode	Collation	Default Value	Policy Tags	Description
order_id	STRING	NULLABLE				
payment_sequential	INTEGER	NULLABLE				
payment_type	STRING	NULLABLE				
payment_installments	INTEGER	NULLABLE				
payment_value	FLOAT	NULLABLE				

BS_case.products;



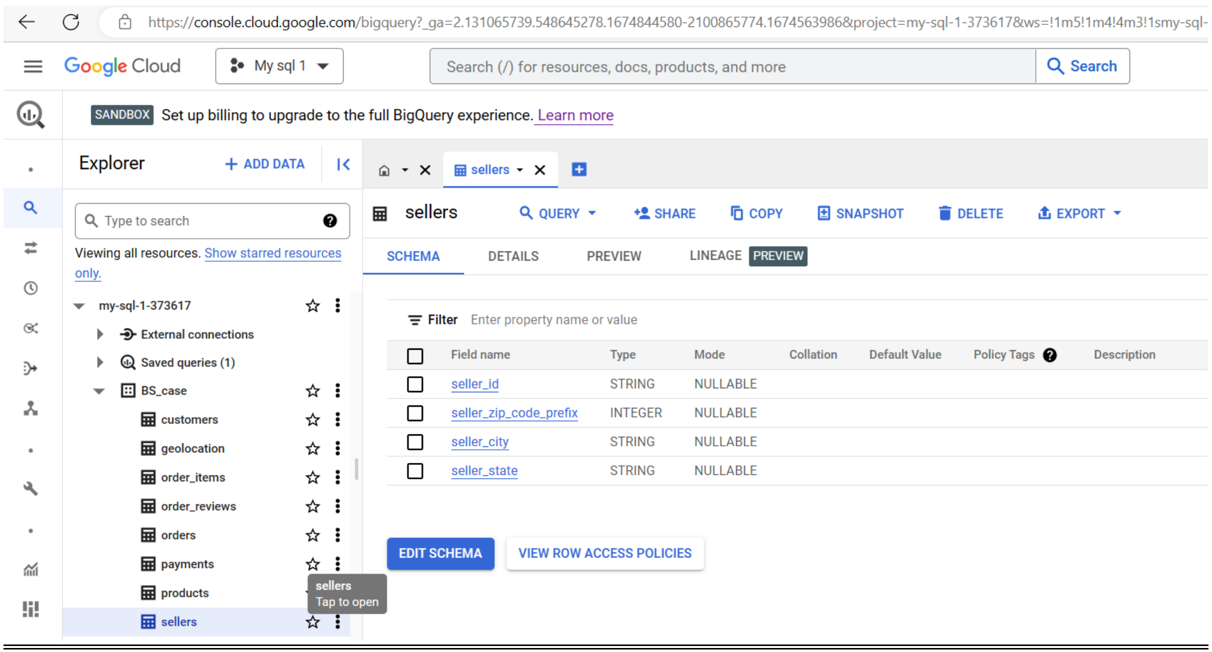
https://console.cloud.google.com/bigquery?_ga=2.131065739.548645278.1674844580-2100865774.1674563986&project=my-sql-1-373617&ws=!1m5!1m4!4m3!1smy-sql-1-3

Set up billing to upgrade to the full BigQuery experience. Learn more

Viewing all resources. Show starred resources only.

Field name	Type	Mode	Collation	Default Value	Policy Tags	Description
product_id	STRING	NULLABLE				
product_category	STRING	NULLABLE				
product_name_length	INTEGER	NULLABLE				
product_description_length	INTEGER	NULLABLE				
product_photos_qty	INTEGER	NULLABLE				
product_weight_g	INTEGER	NULLABLE				
product_length_cm	INTEGER	NULLABLE				
product_height_cm	INTEGER	NULLABLE				
product_width_cm	INTEGER	NULLABLE				

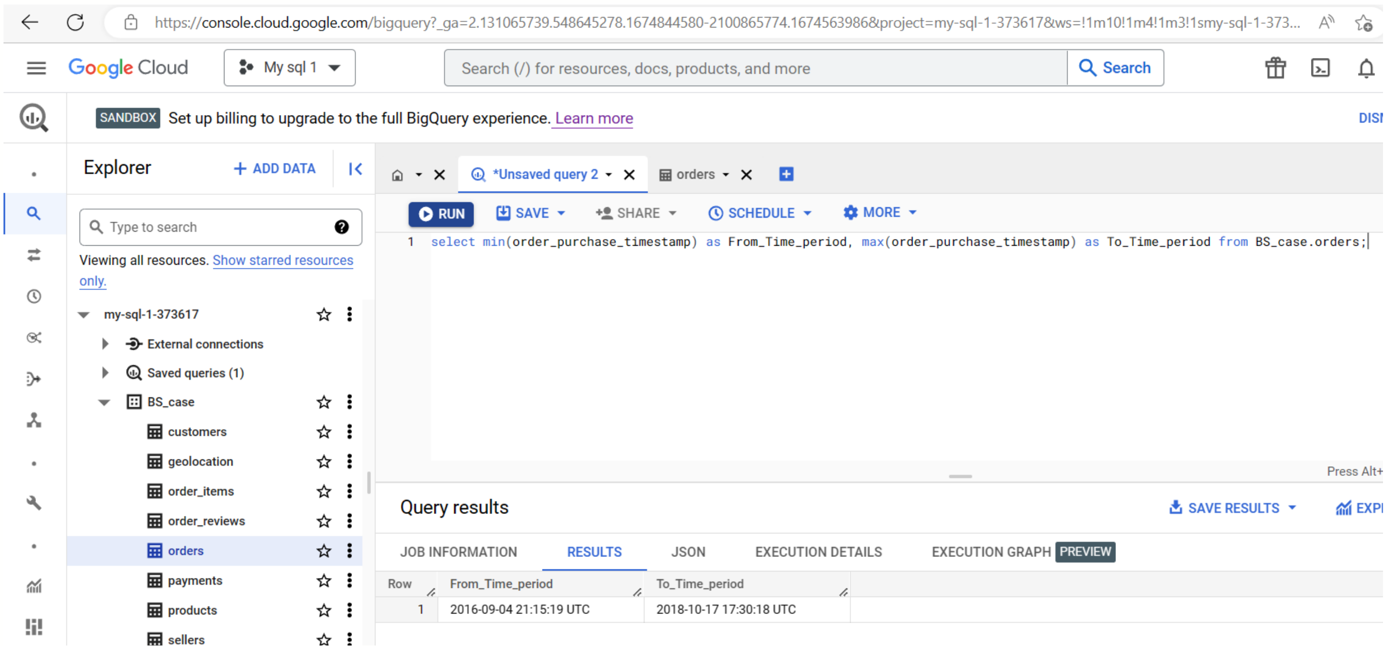
BS_case.sellers;



Field name	Type	Mode	Collation	Default Value	Policy Tags	Description
seller_id	STRING	NULLABLE				
seller_zip_code_prefix	INTEGER	NULLABLE				
seller_city	STRING	NULLABLE				
seller_state	STRING	NULLABLE				

Time period for which the data is given

select min(order_purchase_timestamp) as From_Time_period, max(order_purchase_timestamp) as To_Time_period from BS_case.orders;



```
select min(order_purchase_timestamp) as From_Time_period, max(order_purchase_timestamp) as To_Time_period from BS_case.orders;
```

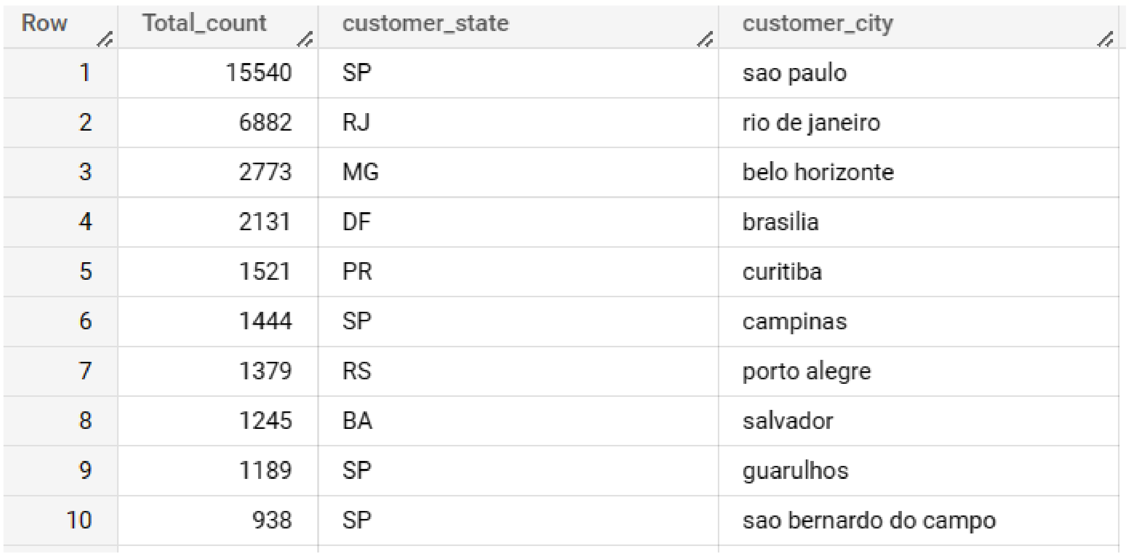
Row	From_Time_period	To_Time_period
1	2016-09-04 21:15:19 UTC	2018-10-17 17:30:18 UTC

Cities and States of customers ordered during the given period

Query -

select distinct count(O.order_id) over(partition by C.customer_state,C.customer_city) as Total_count,C.customer_state,C.customer_city from BS_case.customers as C

inner join BS_case.orders as O
on C.customer_id=O.customer_id
order by Total_count desc;



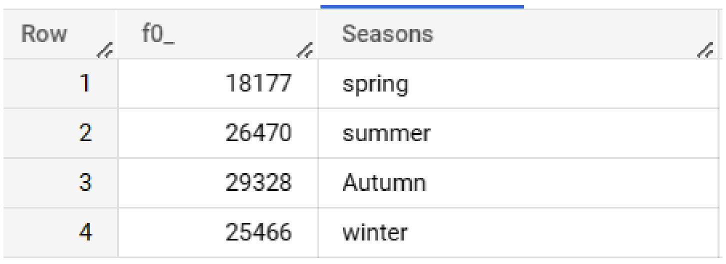
Row	Total_count	customer_state	customer_city
1	15540	SP	sao paulo
2	6882	RJ	rio de janeiro
3	2773	MG	belo horizonte
4	2131	DF	brasilia
5	1521	PR	curitiba
6	1444	SP	campinas
7	1379	RS	porto alegre
8	1245	BA	salvador
9	1189	SP	guarulhos
10	938	SP	sao bernardo do campo

2.In-depth Exploration

Can we see some seasonality with peaks of customers orders at specific months?

Query - select count(order_id),

```
case
  when extract (month from order_purchase_timestamp ) between 1 and 3 then "summer"
  when extract (month from order_purchase_timestamp ) between 4 and 6 then "Autumn"
  when extract (month from order_purchase_timestamp ) between 7 and 9 then "winter"
  when extract (month from order_purchase_timestamp ) between 10 and 12 then "spring"
  else "Unknown"
END as Seasons
from BS_case.orders
group by seasons;
```



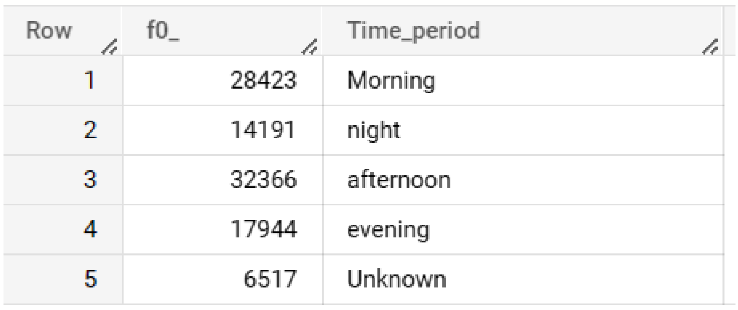
Row	f0_	Seasons
1	18177	spring
2	26470	summer
3	29328	Autumn
4	25466	winter

1. What time do Brazilian customers tend to buy (Dawn, Morning, Afternoon or Night)?

Query)

select count(order_id),

case
when extract (hour from order_purchase_timestamp) between 5 and 12 then "Morning"
when extract (hour from order_purchase_timestamp) between 12 and 17 then "afternoon"
when extract (hour from order_purchase_timestamp) between 17 and 20 then "evening"
when extract (hour from order_purchase_timestamp) between 20 and 22 or
extract (hour from order_purchase_timestamp) between 1 and 4 then "night"
else "Unknown"
end as Time_period
from BS_case.orders
group by time_period;



Row	f0_	Time_period
1	28423	Morning
2	14191	night
3	32366	afternoon
4	17944	evening
5	6517	Unknown

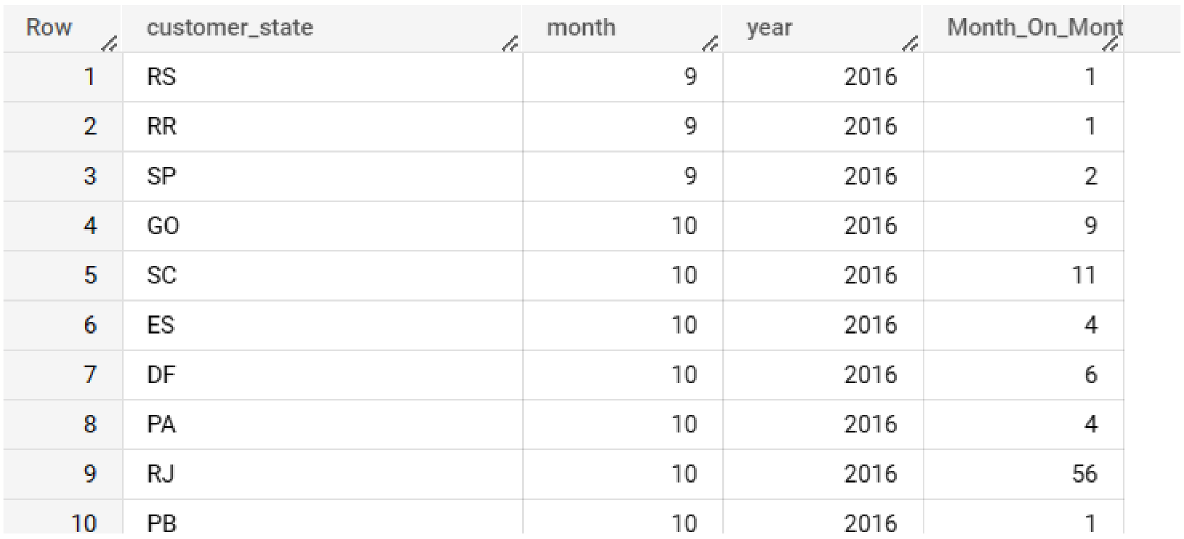
3)Evolution of E-commerce orders in the Brazil region:

Get month on month orders by states

Query- select distinct C. customer_state,
extract(month from O.order_purchase_timestamp) as month,extract(year from O.order_purchase_timestamp) as year,

count(O.order_id) over(partition by extract(month from O.order_purchase_timestamp),extract(year from O.order_purchase_timestamp),C.customer_state) as Month_On_Month,
from BS_case.orders as O

```
inner join BS_case.customers as C
on O.customer_id = C.customer_id
order by Year,month;
```

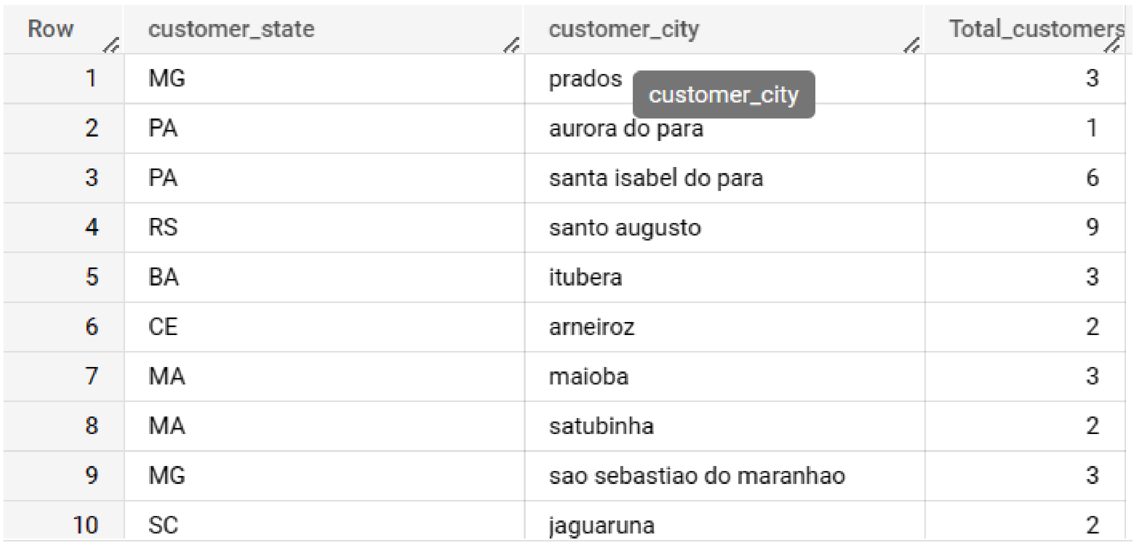


Row	customer_state	month	year	Month_On_Mont
1	RS	9	2016	1
2	RR	9	2016	1
3	SP	9	2016	2
4	GO	10	2016	9
5	SC	10	2016	11
6	ES	10	2016	4
7	DF	10	2016	6
8	PA	10	2016	4
9	RJ	10	2016	56
10	PB	10	2016	1

Distribution of customers across the states in Brazil

Query - select distinct customer_state,customer_city,count(customer_id)

over(partition by customer_state,customer_city) as Total_customers from BS_case.custom
ers;

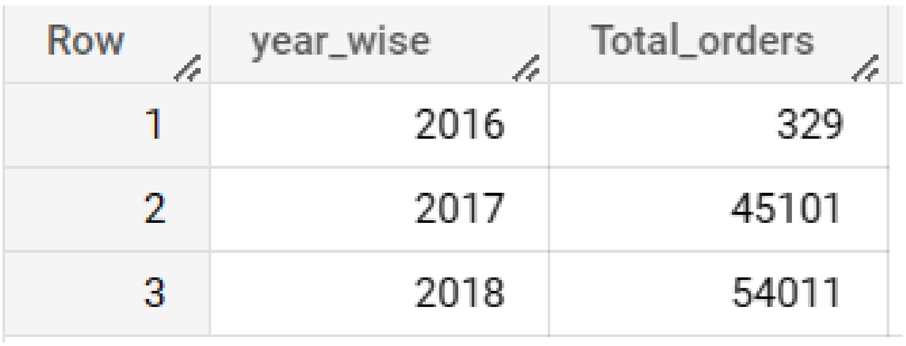

customer_city

Row	customer_state	customer_city	Total_customers
1	MG	prados	3
2	PA	aurora do para	1
3	PA	santa isabel do para	6
4	RS	santo augusto	9
5	BA	itubera	3
6	CE	arneiroz	2
7	MA	maioba	3
8	MA	satubinha	2
9	MG	sao sebastiao do maranhao	3
10	SC	jaguaruna	2

DATA ACCORDING TO YEAR WISE

```
select distinct year_wise,
count(No_Of_Orders) over (partition by year_wise) as Total_orders
from
(
select extract (year from order_purchase_timestamp) as year_wise,

count(order_id) over(partition by extract (year from order_purchase_timestamp) order by o
rder_purchase_timestamp) as No_Of_Orders,
from BS_case.orders)
order by year_wise
;
```



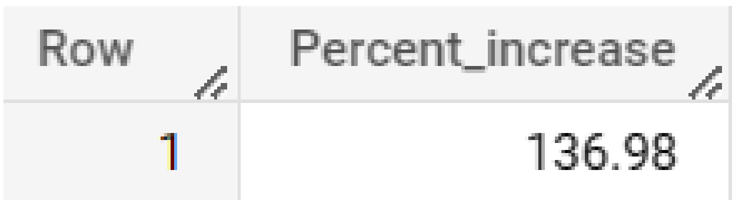
Row	year_wise	Total_orders
1	2016	329
2	2017	45101
3	2018	54011

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

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

sQuery –

```
select round(t1.percent_increase_in_cost_of_orders,2) as Percent_increase
from
(
select
((t.Cost_of_orders/lag(t.Cost_of_orders,1) over (order by year) )-
1)*100 as percent_increase_in_cost_of_orders
from (
select Sum(p.payment_value) as Cost_of_orders,
extract(year from o.order_purchase_timestamp) as year
from BS_case.orders as o
inner join BS_case.payments as p
on o.order_id = p.order_id
where  extract(month from o.order_purchase_timestamp) in (1,2,3,4,5,6,7,8) and extract(ye
ar from o.order_purchase_timestamp) in ( 2017 ,2018)
group by year
order by 2
) as t) as t1
where t1.percent_increase_in_cost_of_orders is not null
;
```

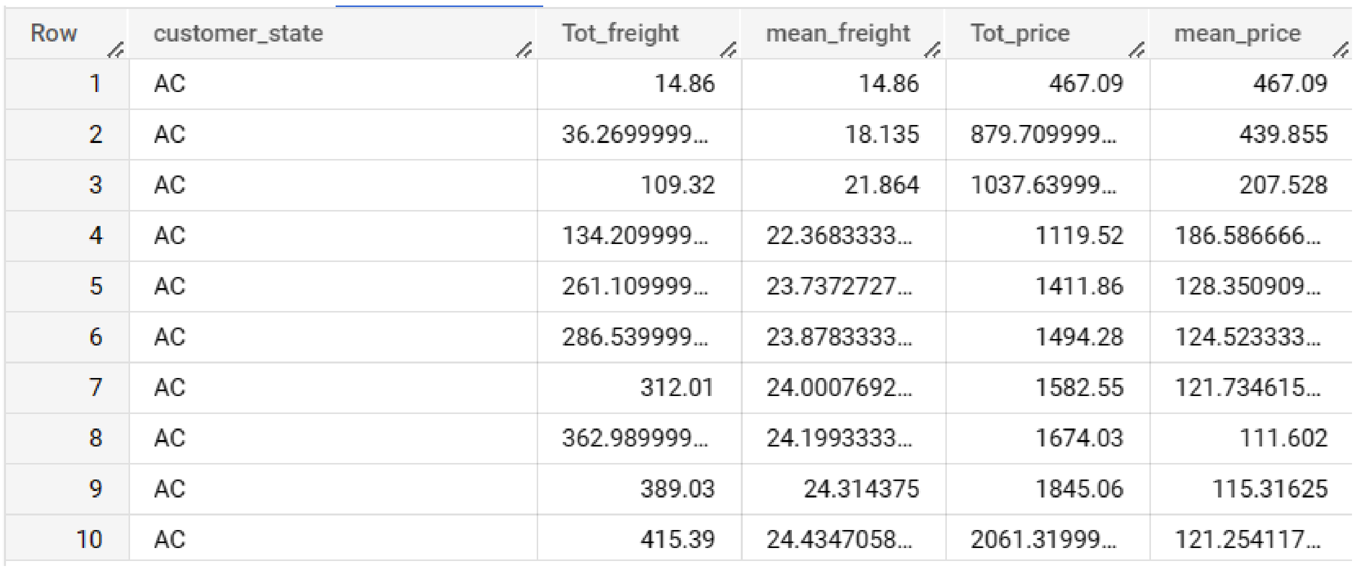


Row	Percent_increase
1	136.98

Mean & Sum of price and freight value by customer state

```
select distinct C.customer_state,
sum(OT.freight_value) over(partition by C.customer_state order by OT.freight_value
) as Tot_freight,
AVG(OT.freight_value) over(partition by C.customer_state order by OT.freight_value
) as mean_freight,
sum(P.payment_value) over(partition by C.customer_state order by OT.freight_value )
 as Tot_price,
Avg(P.payment_value) over(partition by C.customer_state order by OT.freight_value )
 as mean_price
from BS_case.customers as C
inner join BS_case.orders as O
```

```
on C.customer_id=O.customer_id
inner join BS_case.order_items  as OT
on O.order_id=OT.order_id
inner join BS_case.payments as P
on O.order_id=P.order_id
group by C.customer_state,OT.freight_value,P.payment_value
order by C.customer_state;
```

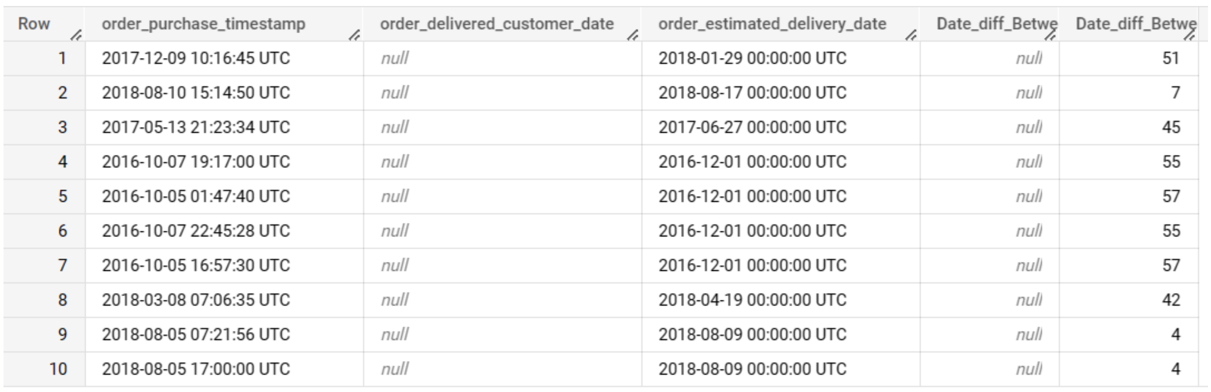


Row	customer_state	Tot_freight	mean_freight	Tot_price	mean_price
1	AC	14.86	14.86	467.09	467.09
2	AC	36.2699999...	18.135	879.709999...	439.855
3	AC	109.32	21.864	1037.63999...	207.528
4	AC	134.209999...	22.3683333...	1119.52	186.586666...
5	AC	261.109999...	23.7372727...	1411.86	128.350909...
6	AC	286.539999...	23.8783333...	1494.28	124.523333...
7	AC	312.01	24.0007692...	1582.55	121.734615...
8	AC	362.989999...	24.1993333...	1674.03	111.602
9	AC	389.03	24.314375	1845.06	115.31625
10	AC	415.39	24.4347058...	2061.31999...	121.254117...

5. Analysis on sales, freight and delivery time

Calculate days between purchasing, delivering and estimated delivery

```
  SELECT Avg(OT.freight_value),C.customer_state,DATE_DIFF(date(order_delivered_customer_date),d
ate( order_purchase_timestamp),day) as time_to_delivery,
  DATE_DIFF(date(order_estimated_delivery_date),date( order_purchase_timestamp),day) as diff_esti
mated_delivery from  BS_case.orders as O
inner join BS_case.order_items as OT  on O.order_id=OT.order_id inner join BS_case.customers as C
  on O.customer_id=C.customer_id
  Group by C.customer_state,time_to_delivery, diff_estimated_delivery
  order;
```

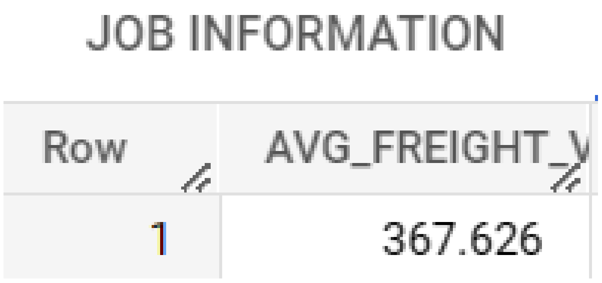


Row	order_purchase_timestamp	order_delivered_customer_date	order_estimated_delivery_date	Date_diff_Betwe	Date_diff_Betwe
1	2017-12-09 10:16:45 UTC	null	2018-01-29 00:00:00 UTC	null	51
2	2018-08-10 15:14:50 UTC	null	2018-08-17 00:00:00 UTC	null	7
3	2017-05-13 21:23:34 UTC	null	2017-06-27 00:00:00 UTC	null	45
4	2016-10-07 19:17:00 UTC	null	2016-12-01 00:00:00 UTC	null	55
5	2016-10-05 01:47:40 UTC	null	2016-12-01 00:00:00 UTC	null	57
6	2016-10-07 22:45:28 UTC	null	2016-12-01 00:00:00 UTC	null	55
7	2016-10-05 16:57:30 UTC	null	2016-12-01 00:00:00 UTC	null	57
8	2018-03-08 07:06:35 UTC	null	2018-04-19 00:00:00 UTC	null	42
9	2018-08-05 07:21:56 UTC	null	2018-08-09 00:00:00 UTC	null	4
10	2018-08-05 17:00:00 UTC	null	2018-08-09 00:00:00 UTC	null	4

Top 5 states with highest/lowest average freight value - sort in desc/asc limit 5

Query) **SELECT AVG(freight_value) as AVG_FREIGHT_VALUE**

FROM (SELECT freight_value FROM BS_case.order_items ORDER BY freight_value DESC LIMIT 5);

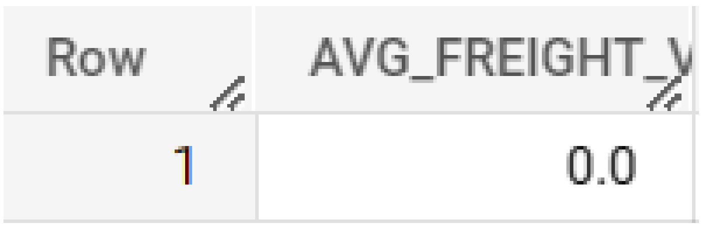


JOB INFORMATION

Row	AVG_FREIGHT_V
1	367.626

Query) **SELECT AVG(freight_value) as AVG_FREIGHT_VALUE**

FROM (SELECT freight_value FROM BS_case.order_items ORDER BY freight_value LIMIT 5);

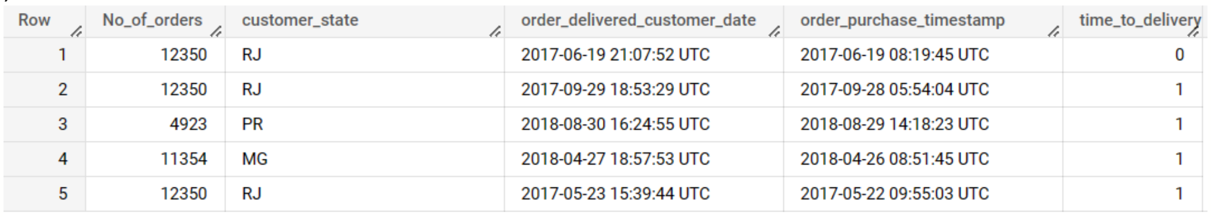


Row	AVG_FREIGHT_V
1	0.0

Top 5 states with highest/lowest average time to delivery

Query – lowest

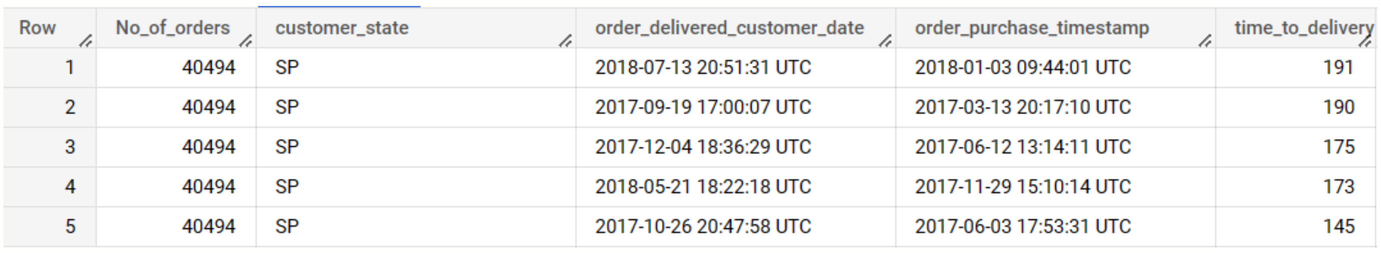
```
select distinct Count(O.order_id) over(partition by C.customer_state) No_of_orders,C.custo
mer_state,O.order_delivered_customer_date,O.order_purchase_timestamp, DATE_DIFF(da
te(O.order_delivered_customer_date),date( O.order_purchase_timestamp),day) as time_to
_delivery
from BS_case.orders as O inner join BS_case.customers as C on O.customer_id=C.customer_
id
where order_status in ("shipped","delivered","approved","invoiced","processing") and O.or
der_delivered_customer_date is not null
 order by No_of_orders ,time_to_delivery  limit 5;
;
```



Row	No_of_orders	customer_state	order_delivered_customer_date	order_purchase_timestamp	time_to_delivery
1	12350	RJ	2017-06-19 21:07:52 UTC	2017-06-19 08:19:45 UTC	0
2	12350	RJ	2017-09-29 18:53:29 UTC	2017-09-28 05:54:04 UTC	1
3	4923	PR	2018-08-30 16:24:55 UTC	2018-08-29 14:18:23 UTC	1
4	11354	MG	2018-04-27 18:57:53 UTC	2018-04-26 08:51:45 UTC	1
5	12350	RJ	2017-05-23 15:39:44 UTC	2017-05-22 09:55:03 UTC	1

Highest

```
select distinct Count(O.order_id) over(partition by C.customer_state) No_of_orders,C.customer_stat
e,O.order_delivered_customer_date,O.order_purchase_timestamp, DATE_DIFF(date(O.order_delive
red_customer_date),date( O.order_purchase_timestamp),day) as time_to_delivery
from BS_case.orders as O inner join BS_case.customers as C on O.customer_id=C.customer_id
where order_status in ("shipped","delivered","approved","invoiced","processing") and O.order_deli
vered_customer_date is not null
 order by No_of_orders desc,time_to_delivery desc  limit 5;
```



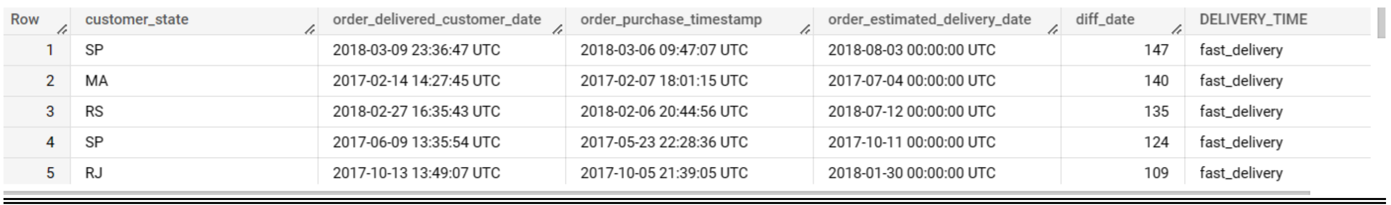
Row	No_of_orders	customer_state	order_delivered_customer_date	order_purchase_timestamp	time_to_delivery
1	40494	SP	2018-07-13 20:51:31 UTC	2018-01-03 09:44:01 UTC	191
2	40494	SP	2017-09-19 17:00:07 UTC	2017-03-13 20:17:10 UTC	190
3	40494	SP	2017-12-04 18:36:29 UTC	2017-06-12 13:14:11 UTC	175
4	40494	SP	2018-05-21 18:22:18 UTC	2017-11-29 15:10:14 UTC	173
5	40494	SP	2017-10-26 20:47:58 UTC	2017-06-03 17:53:31 UTC	145

Top 5 states where delivery is really fast/ not so fast compared to estimated date

Query -

```
SELECT C.customer_state,O.order_delivered_customer_date,O.order_purchase_timestamp,O.order
_estimated_delivery_date,

Date_diff(date(O.order_estimated_delivery_date),date(O.order_delivered_customer_date),day) as
diff_date,
CASE
When Extract (DATE FROM O.order_estimated_delivery_date) > Extract (date from O.order_delivere
d_customer_date) THEN "fast_delivery"
when  Extract (date from O.order_estimated_delivery_date) = Extract (date from O.order_delivered_
customer_date) THEN"normal_delivery"
when  Extract (date from O.order_estimated_delivery_date) < Extract (date from O.order_delivered_
customer_date) THEN "Late_delivery"
END as DELIVERY_TIME
from BS_case.orders  as O
inner join BS_case.customers as C
on O.customer_id=C.customer_id
where O.order_status in ("shipped","delivered","approved","invoiced","processing") and O.order_d
elivered_customer_date is not null
 order by diff_date desc;
```



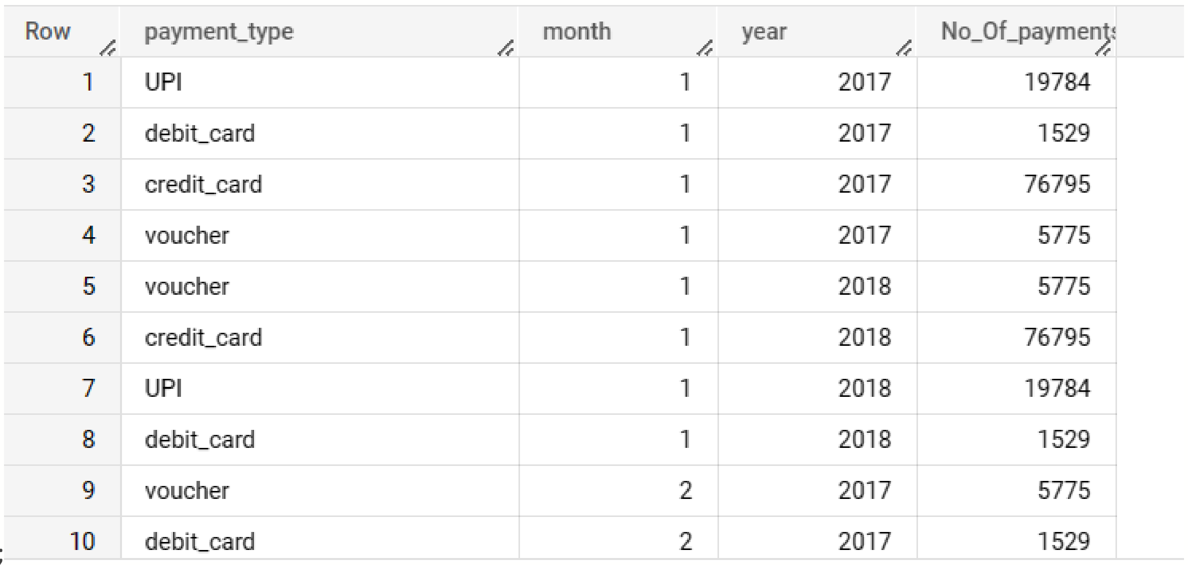
Row	customer_state	order_delivered_customer_date	order_purchase_timestamp	order_estimated_delivery_date	diff_date	DELIVERY_TIME
1	SP	2018-03-09 23:36:47 UTC	2018-03-06 09:47:07 UTC	2018-08-03 00:00:00 UTC	147	fast_delivery
2	MA	2017-02-14 14:27:45 UTC	2017-02-07 18:01:15 UTC	2017-07-04 00:00:00 UTC	140	fast_delivery
3	RS	2018-02-27 16:35:43 UTC	2018-02-06 20:44:56 UTC	2018-07-12 00:00:00 UTC	135	fast_delivery
4	SP	2017-06-09 13:35:54 UTC	2017-05-23 22:28:36 UTC	2017-10-11 00:00:00 UTC	124	fast_delivery
5	RJ	2017-10-13 13:49:07 UTC	2017-10-05 21:39:05 UTC	2018-01-30 00:00:00 UTC	109	fast_delivery

<u>6.PAYMENT TYPE ANALYSIS</u> :

Month over Month count of orders for different payment Types

Query)

```
SELECT distinct p.payment_type,extract (month from O.order_purchase_timestamp) as month,extra
ct(year from O.order_purchase_timestamp) as year,
count(O.order_id) over(partition by p.payment_type) No_Of_payments
 from BS_case.payments as P
inner join BS_case.orders as O
on O.order_id=P.order_id
inner join BS_case.customers as C
on C.customer_id=O.customer_id
order by month,year;
```



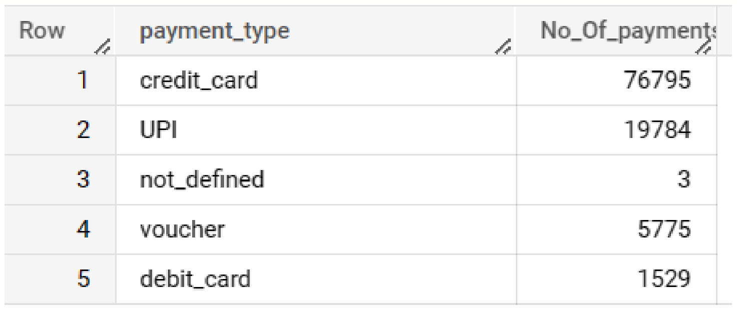
Row	payment_type	month	year	No_Of_payments
1	UPI	1	2017	19784
2	debit_card	1	2017	1529
3	credit_card	1	2017	76795
4	voucher	1	2017	5775
5	voucher	1	2018	5775
6	credit_card	1	2018	76795
7	UPI	1	2018	19784
8	debit_card	1	2018	1529
9	voucher	2	2017	5775
10	debit_card	2	2017	1529

;

Count of orders based on the no. of payment installments

Query)

SELECT distinct payment_type,
count(payment_type) over(partition by payment_type) No_Of_payments
from BS_case.payments ;



Row	payment_type	No_Of_payments
1	credit_card	76795
2	UPI	19784
3	not_defined	3
4	voucher	5775
5	debit_card	1529