SQL PROJECT

HARIUN NISHA S

I 1.Data type of all columns in the “customers table”

RESULT:

Reference

customer Preview

sql-project-2025-471915.proj_sql_2025.customer

99.44K rows • 8.76 MB • Last modified Sep 13, 2025, 9:02 AM

Filter Enter property name or value

Field name	Type
customer_id	STRING
customer_unique_id	STRING
customer_zip_code_prefix	INTEGER
customer_city	STRING
customer_state	STRING

INTERPERTATION :

- This is customer data table
- It shows all details about the customer
- It is combine orders and orderitem table

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

QUERY:

```
Select min(order_purchase_timestamp) as start_date,
max(order_purchase_timestamp) as end_date
from `sql-project-2025-471915.proj_sql_2025.orders`;
```

RESULT :

Query results

Save results Open in

Job information | Results | Visualization | JSON | Execution details | Execution graph

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

INTERPRETATION:

- This shows the time period of orders
- It shows order and transaction
- It shows the ordertime which customer place first and last

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

QUERY :

```
select count(customer_city) as totcity ,
count(customer_state) as totstate
from `sql-project-2025-471915.proj_sql_2025.customer` c
join `sql-project-2025-471915.proj_sql_2025.orders` o
on c.customer_id=o.customer_id
where order_purchase_timestamp between '2015-01-01'and'2017-12-01'
```

RESULT :

Query results

Save results Open in

Job information | Results | Visualization | JSON | Execution details | Execution graph

Row	totcity	totstate
1	39757	39757

INTERPRETATION :

- It shows the unqiue customer with total city and total state
- It shows all customer city and state

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

QUERY :

```
select
orderyear,ordermonth,totalorder,lag(totalorder)o
ver(order by orderyear,ordermonth)as
pervi_sales,
  (totalorder)-lag(totalorder)over(order by
orderyear,ordermonth) as diff_sales from
  (select extract(year from
order_purchase_timestamp) as  orderyear,
extract(month from order_purchase_timestamp)
as ordermonth,
count(order_purchase_timestamp) as totalorder
from `sql-project-2025-
471915.proj_sql_2025.orders`
group by orderyear,ordermonth
order by orderyear,ordermonth)
```

RESULT :

Query results

Save results Open in

Job information | Results | Visualization | JSON | Execution details | Execution graph

Row	orderyear	ordermonth	totalorder	pervi_sales	diff_sales
1	2017	6	3245	3700	-455
2	2016	12	1	324	-323
3	2018	4	6939	7211	-272
4	2017	2	1780	800	980
5	2017	12	5673	7544	-1871
6	2017	1	800	1	799
7	2017	5	3700	2404	1296
8	2018	5	6873	6939	-66
9	2018	9	16	6512	-6496
10	2018	1	7269	5673	1596

INTERPRETATION :

- **It shows the order place month of month**
- **It shows the month and yearwise growth**
- **No of orders were placed and what is perivous year sales**

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

QUERY :

```
select extract (month from order_purchase_timestamp ) as monthlyorder, count(order_id) as totalorder
 from`sql-project-2025-471915.proj_sql_2025.orders`
group by monthlyorder
```

RESULT :

Query results

Save results Open in

Job information | Results | Visualization | JSON | Execution details | Execution graph

Row	customer_state	totalorder	monthlyorder
1	PI	1	2
2	RO	1	4
3	SE	1	11
4	AL	1	12
5	PI	1	2
6	TO	1	2
7	RR	1	1
8	PI	1	3
9	RO	1	6
10	RO	1	8

interpertation:

- This query shows the total order monthlywise and give the growth of orders in seasonality where customers purchased

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

QUERY:

```
select case
when extract(hour from
o.order_purchase_timestamp) between 0 and 6
then'Dawn'
when extract(hour from
o.order_purchase_timestamp) between 7 AND 12
then 'Morning'
when extract(hour from
o.order_purchase_timestamp) BETWEEN 13 AND
18 then'Afternoon'
else
'Night’
end as time_order,
count(o.order_id) as totorder,
c.customer_state
from `sql-project-2025-
471915.proj_sql_2025.orders`o
join `sql-project-2025-
471915.proj_sql_2025.customer`c
on o.customer_id=c.customer_id
group by time_order,customer_state
order by totorder desc;
```

RESULT :

Query results

Job information | Results | Visualization | JSON | Execution details | Execution graph

Row	time_order	totorder	customer_state
1	Afternoon	18482	SP
2	Morning	13315	SP
3	Night	13083	SP
4	Afternoon	5634	RJ
5	Afternoon	5014	MG
6	Night	4074	RJ
7	Morning	3987	RJ
8	Morning	3812	MG
9	Night	3779	MG
10	Dawn	2569	SP

INTERPRETATION:

- **This query shows the brazil customer placed their orders in dawn, morning,afernoon, night**

- **It shows the no of orders where placed in particular time period which time was placed more orders**

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

QUERY:

```
select c.customer_state, count(o.order_id) as totalorder,extract(month from o.order_purchase_timestamp) as monthlyorder from `sql-project-2025-471915.proj_sql_2025.customer` c join `sql-project-2025-471915.proj_sql_2025.orders`o on c.customer_id=o.customer_id group by c.customer_state,o.order_id,monthlyorder;
```

RESULT :

Query results

Save results Open in

Job information | Results | Visualization | JSON | Execution details | Execution graph

Row	customer_state	totalorder	monthlyorder
1	PI	1	2
2	RO	1	4
3	SE	1	11
4	AL	1	12
5	PI	1	2
6	TO	1	2
7	RR	1	1
8	PI	1	3
9	RO	1	6
10	RO	1	8

INTERPRETATION:

- This query shows the no of orders placed by monthlywise with the state of customer in unique

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

QUERY :

```
select count(customer_id) as
totcustomer,customer_state
from
 `sql-project-2025-471915.proj_sql_2025.customer`
group by customer_state
order by totcustomer desc;
```

RESULT :

Query results

Save results Open in

Job information | Results | Visualization | JSON | Execution details | Execution graph

Row	orderyear	ordermonth	totalorder	pervi_sales	diff_sales
1	2017	6	3245	3700	-455
2	2016	12	1	324	-323
3	2018	4	6939	7211	-272
4	2017	2	1780	800	980
5	2017	12	5673	7544	-1871
6	2017	1	800	1	799
7	2017	5	3700	2404	1296
8	2018	5	6873	6939	-66
9	2018	9	16	6512	-6496
10	2018	1	7269	5673	1596

INTERPRETATION:

- This query shows total customer who placed order with their state and calculate the customer's total in unique

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

QUERY :

```
select extract(year from
o.order_purchase_timestamp) as totyear,
sum(oi.price) as totcost,
sum(oi.price) * 100.0 / sum(sum(oi.price)) over() as
precen_tot
from `sql-project-2025-
471915.proj_sql_2025.order_items` oi
join `sql-project-2025-471915.proj_sql_2025.orders`
o
on o.order_id = oi.order_id
where extract(year from
o.order_purchase_timestamp) in (2017, 2018)
and extract(month from
o.order_purchase_timestamp) between 1 and 8
group by totyear
order by totyear desc;
```

RESULT :

Query results

Job information | Results | Visualization | JSON | Execution details

Row	totyear	totcost	precen_tot
1	2018	7385905.800003...	70.34928892193...
2	2017	3113000.319999...	29.65071107806...

INTERPRETATION :

- **For this query, we have extract the month and year**

- **Also it is showing total number of price, percentage and months only from jan to aug**

- **It combining the table customer and order**

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

QUREY :

```
select sum(oi.price)as totalvalue,count(o.order_id)
as totalorder,
avg(oi.price)as avgvalue ,c.customer_state
from`sql-project-2025-
471915.proj_sql_2025.order_items`oi
join `sql-project-2025-
471915.proj_sql_2025.orders`o
on oi.order_id=o.order_id
join `sql-project-2025-
471915.proj_sql_2025.customer`c
on o.customer_id=c.customer_id
group by customer_state
order by totalorder;
```

RESULT :

Query results

Save results Open in

Job information | Results | Visualization | JSON | Execution details | Execution graph

Row	totalvalue	totalorder	avgvalue	customer_state
1	7829.429999999...	52	150.5659615384...	RR
2	13474.29999999...	82	164.3207317073...	AP
3	15982.94999999...	92	173.7277173913...	AC
4	22356.84000000...	165	135.4960000000...	AM
5	46140.64000000...	278	165.9735251798...	RO
6	49621.74000000...	315	157.5293333333...	TO
7	58920.8500000001	385	153.0411688311...	SE
8	80314.80999999...	444	180.8892117117...	AL
9	83034.97999999...	529	156.9659357277...	RN
10	86914.07999999...	542	160.3580811808...	PI

INTERPRETATION :

- **This query is the combination of table customer and order**

- **It shows the totalprice and average of price which was purchased by customer by their states**

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

QUERY :

```
select c.customer_state,
avg(oi.freight_value) as avg_freight,
sum(oi.freight_value) as tot_freight
from `sql-project-2025-471915.proj_sql_2025.order_items`oi
join `sql-project-2025-471915.proj_sql_2025.orders`o
on oi.order_id=o.order_id
join `sql-project-2025-471915.proj_sql_2025.customer`c
on o.customer_id=c.customer_id
group by customer_state
order by avg_freight,tot_freight
```

RESULT :

Query results Save results Open in

Job information | Results | Visualization | JSON | Execution details

Row	customer_state	avg_freight	tot_freight
1	SP	15.14727539041...	718723.0700000...
2	PR	20.53165156794...	117851.6799999...
3	MG	20.63016680630...	270853.4600000...
4	RJ	20.96092393168...	305589.3100000...
5	DF	21.04135494596...	50625.50000000...
6	SC	21.47036877394...	89660.25999999...
7	RS	21.73580433039...	135522.7399999...
8	ES	22.05877659574...	49764.59999999...
9	GO	22.76681525932...	53114.98000000...
10	MS	23.37488400488...	19144.03000000...

INTERPRETATION :

- **This query is the combination of table customer,orders,order item**

- **It shows the total freight value and average freight value of customer by their states order result shows the total freight cost**

V. Analysis based on sales, freight and delivery time.

A. Find the no. of days taken to deliver each order from the order's purchase date
as delivery time.

B. Also, calculate the difference (in days) between the estimated & actual delivery
date of an order.
Do this in a single query.

QUERY :

```
select   order_id,
date_diff(order_delivered_customer_date,
order_purchase_timestamp, day) as
delivery_day,

 date_diff(order_delivered_customer_date,
order_estimated_delivery_date, day) as
estimate_dif_day

from `sql-project-2025-
471915.proj_sql_2025.orders`
where order_delivered_customer_date is not
null;
```

RESULT :

Job information | Results | Visualization | JSON | Execution details | Execution graph

Row	order_id	delivery_day	estimate_dif_day
1	65d1e226dfaeb8cdc42f665422...	35	-16
2	2c45c33d2f9cb8ff8b1c86cc28...	30	-28
3	1950d777989f6a877539f53795...	30	12
4	bfbd0f9bdef84302105ad712db...	54	36
5	98974b076b01553d49ee64679...	43	-6
6	c4b41c36dd589e901f6879f25a...	36	-14
7	d2292ff2201e74c5db154d1b7a...	29	-20
8	95e01270fcbae986342340010...	30	-19
9	ed8c7b1b3eb256c70ce0c7423...	44	-5
10	5cc475c7c03290048eb2e742c...	68	18

INTERPRETATION :

- **In this query deliveryday tells the deliverydate of order which customer purchased**
- **Estimated day tells delivery of the order before or after from orderdate**
- **It is positive the product was delivery before delivery date**
- **It is negative the product was delivery after delivery date**

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

Hint: We want you to find the top 5 & the bottom 5 states arranged in
increasing order of the average freight value.

QUERY :

```
with avg_freight_value  as (

select
 c.customer_state,
avg(oi.freight_value) as avg_freight,
dense_rank() over(order by avg(oi.freight_value)
desc) as high_freight,
dense_rank() over (order by avg (oi.freight_value)as
as  low_freight
from `sql-project-2025-
471915.proj_sql_2025.order_items` oi
join `sql-project-2025-471915.proj_sql_2025.orders`
on oi.order_id = o.order_id
join `sql-project-2025-
471915.proj_sql_2025.customer` c
on o.customer_id = c.customer_id
group by c.customer_state
)
```

```
select
  customer_state,
  avg_freight,
  high_freight,
  low_freight
from avg_freight_value
where high_freight <= 5 or low_freight <= 5
order by avg_freight desc;
```

RESULT :

Query results

Job information | Results | Visualization | JSON | Execution details | Execution graph

Row	customer_state	avg_freight	high_freight	low_freight
1	RR	42.98442307692...	1	27
2	PB	42.72380398671...	2	26
3	RO	41.06971223021...	3	25
4	AC	40.07336956521...	4	24
5	PI	39.14797047970...	5	23
6	DF	21.04135494596...	23	5
7	RJ	20.96092393168...	24	4
8	MG	20.63016680630...	25	3
9	PR	20.53165156794...	26	2
10	SP	15.14727539041...	27	1

INTERPRETATION :

- **In this query have find the average of freight value**
- **We shoud make the rank for top 5 state with high freight value**
- **And have to show the top 5 state with low freight**

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

Query :

```
with avg_delivery_per_state as (
   select
c.customer_state,avg(date_diff(o.order_delivered_
customer_date, o.order_purchase_timestamp,
day)) as avg_delivery,
   rank()over(order by
avg(date_diff(o.order_delivered_customer_date,
o.order_purchase_timestamp, day))asc) as
lowrank,
   rank() over(order by
avg(date_diff(o.order_delivered_customer_date,
o.order_purchase_timestamp,day))desc) as
highrank
   from `sql-project-2025-
471915.proj_sql_2025.orders` o
 join `sql-project-2025-
471915.proj_sql_2025.customer` c
 on o.customer_id = c.customer_id
```

```
where
o.order_delivered_customer_date is not
null
 and  o.order_purchase_timestamp is
not null
group by c.customer_state
)
select customer_state,
avg_delivery,lowrank,highrank
from avg_delivery_per_state
where lowrank <= 5 or highrank <= 5
order by avg_delivery asc;
```

RESULT :

Query results

Job information | Results | Visualization | JSON | Execution details | Execution graph

ow	customer_state	avg_delivery	lowrank	highrank
1	SP	8.298061489072...	1	27
2	PR	11.52671135486...	2	26
3	MG	11.54381329810...	3	25
4	DF	12.50913461538...	4	24
5	SC	14.47956019171...	5	23
6	PA	23.31606765327...	23	5
7	AL	24.04030226700...	24	4
8	AM	25.98620689655...	25	3
9	AP	26.73134328358...	26	2
10	RR	28.97560975609...	27	1

INTERPRETATION:

- **It showing the avgerage delivery time by customer state**
- **It based on the distance**
- **Highest delivery time may be near warehouse or based on shipping**
- **Lower delivery time may be based on rural areas**

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

QUERY :

```
select c.customer_state,
date_diff(o.order_estimated_delivery_date,
o.order_delivered_customer_date,day) as
fast_delivery
from `sql-project-2025-
471915.proj_sql_2025.orders`o
join `sql-project-2025-
471915.proj_sql_2025.customer`c
on o.customer_id=c.customer_id
where (o.order_delivered_customer_date) is not
null
```

RESULT :

Query results

Save results

Job information | Results | Visualization | JSON | Execution details | Execution graph

Row	customer_state	fast_delivery
1	RJ	45
2	SP	44
3	RJ	41
4	MG	-12
5	SP	-36
6	SP	39
7	RJ	40
8	SP	34
9	SP	41
10	SP	33

INTERPRETATION :

- **Here we get the information of fastest delivery state by comparing the delivery date and estimated delivery date**

VI . Analysis based on the payments:

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

QUERY :

```
select count(o.order_id) as totalorder,
extract(month from
o.order_purchase_timestamp) as month_order,
extract(year from o.order_purchase_timestamp)
as year_order,
p.payment_type
from `sql-project-2025-
471915.proj_sql_2025.orders`o
join `sql-project-2025-
471915.proj_sql_2025.payments`p
on o.order_id=p.order_id
group by
month_order,p.payment_type,year_order
order by totalorder, month_order,year_order
desc;
```

RESULT :

Query results

Save results Open in

Job information | Results | Visualization | JSON | Execution details | Execution graph

Row	totalorder	month_order	year_order	payment_type
1	1	9	2018	not_defined
2	1	12	2016	credit_card
3	2	8	2018	not_defined
4	2	10	2016	debit_card
5	3	9	2016	credit_card
6	4	10	2018	voucher
7	9	1	2017	debit_card
8	13	2	2017	debit_card
9	15	9	2018	voucher
10	22	7	2017	debit_card

INTERPRETATION :

- **It tells about the totalorder of customer by year and month and using the payment mode of customer**

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

QUERY :

```
select p.payment_installments, count(o.order_id)
as totalorder,
from `sql-project-2025-
471915.proj_sql_2025.orders` o
join  `sql-project-2025-
471915.proj_sql_2025.payments`p
on o.order_id=p.order_id
group by p.payment_installments
order by totalorder desc
```

RESULT :

Query results

Job information | Results | Visualization | JSON | Execut

Row	payment_installm...	totalorder
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

INTERPRETATION :

- **In this query we getting no of orders with full payments by customer paid mode**

THANK YOU