

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

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

The screenshot shows the Google Cloud BigQuery interface. On the left, the Explorer sidebar lists projects and datasets, with 'Business_case_I' selected. Under 'Business_case_I', the 'customers' table is highlighted. The main pane displays the schema of the 'customers' table, which includes columns: customer_id (STRING, NULLABLE), customer_unique_id (STRING, NULLABLE), customer_zip_code_prefix (INTEGER, NULLABLE), customer_city (STRING, NULLABLE), and customer_state (STRING, NULLABLE). Below the schema, there are buttons for 'EDIT SCHEMA' and 'VIEW ROW ACCESS POLICIES'. At the bottom, there are tabs for 'PERSONAL HISTORY' and 'PROJECT HISTORY'.

Field name	Type	Mode	Key	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					

Inference - Data types of customers table are as follows

Customer_id - String

Customer_unique_id- String

Customer_zip_code_prefix-Integer

Customer_city- String

customer_state- String

2. Get the time range between which the orders were

```
select min(order_purchase_timestamp) as min ,  
max(order_purchase_timestamp) as max
```

```
from `Business_case_I.orders`
```

```

1 select min(order.purchase_timestamp) as min_
2 max(order.purchase_timestamp) as max_
3
4 from `Business_case_I.orders`

```

The screenshot shows the Google Cloud BigQuery interface. The sidebar on the left lists workspace resources under 'Business_case_I', including 'customers', 'geolocation', 'order_items', 'order_reviews', 'orders', 'payments', 'products', 'sellers', and 'farmers_market'. The main area displays a query titled 'Untitled' with the following SQL code:

```

1 select min(order.purchase_timestamp) as min_
2 max(order.purchase_timestamp) as max_
3
4 from `Business_case_I.orders`

```

The 'Query results' section shows the output of the query:

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

Inference- orders when started getting placed are from 2016-09-04 21:15:19 UTC to 2018-10-17 17:30:18 UTC

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

```

select count(distinct(c.customer_city)) as city ,count(distinct(c.customer_state)) as state
from `Business_case_I.customers` as c
inner join `Business_case_I.orders` as o
on c.customer_id=o.customer_id

```

```

4 from `Business_case_I.orders`
5
6 #Count the Cities & States of customers who ordered during the given period
7 select * from `Business_case_I.customers`
8
9 select count(distinct(c.customer_city)) as city ,count(distinct(c.customer_state)) as state
10 from `Business_case_I.orders` as o
11 inner join `Business_case_I.orders` as o
12 on c.customer_id=o.customer_id

```

The screenshot shows the Google Cloud BigQuery interface. The sidebar on the left lists workspace resources under 'Business_case_I', including 'customers', 'geolocation', 'order_items', 'order_reviews', 'orders', 'payments', 'products', 'sellers', and 'farmers_market'. The main area displays a query titled 'Untitled' with the following SQL code:

```

4 from `Business_case_I.orders`
5
6 #Count the Cities & States of customers who ordered during the given period
7 select * from `Business_case_I.customers`
8
9 select count(distinct(c.customer_city)) as city ,count(distinct(c.customer_state)) as state
10 from `Business_case_I.orders` as o
11 inner join `Business_case_I.orders` as o
12 on c.customer_id=o.customer_id

```

The 'Query results' section shows the output of the query:

Row	city	state
1	4119	27

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

```
select extract(year from order_purchase_timestamp) as year,  
count(order_id) as total_orders  
from `Business_case_I.orders`  
group by 1  
order by 1
```

The screenshot shows the Google Cloud BigQuery interface. On the left, the 'Explorer' sidebar lists datasets like 'Business_case_I' containing tables such as 'customers', 'geolocation', 'order_items', etc. In the center, an 'Untitled' query tab is open with the following SQL code:

```
16  
17 select extract(year from order_purchase_timestamp) as year,  
18 count(order_id) as total_orders  
19 from `Business_case_I.orders`  
20 group by 1  
21 order by 1  
22
```

The 'Query results' section displays the output of the query:

Row	year	total_orders
1	2016	329
2	2017	45101
3	2018	54011

The interface includes various navigation and configuration buttons at the top and bottom.

Inference- As you can observe that is there is growing trend in the no. of orders from 2016 to 2018

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

```
select extract(month from order_purchase_timestamp) as month,  
count(order_id) as total_orders  
from `Business_case_I.orders`  
group by 1  
order by 1
```

The screenshot shows the Google Cloud BigQuery interface. On the left, the 'Explorer' sidebar lists datasets like 'Business_case_I' and its tables such as 'customers', 'geolocation', 'order_items', etc. The main area displays the results of an 'Untitled' query:

```
20 group by 1  
21 order by 1  
^
```

Query results

Row	month	total_orders
1	1	8069
2	2	8508
3	3	9893
4	4	9343
5	5	10573
6	6	9412
7	7	10318
8	8	10843
9	9	4305
10	10	4959
11	11	7544

Results per page: 50 | 1 – 12 of 12 | Refresh

Inference- according to monthly seasonality no. of orders being placed in the month of may,july and august are maximum

2.In-depth Exploration:

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

0-6 hrs : Dawn

7-12 hrs : Mornings

13-18 hrs : Afternoon

19-23 hrs : Night

```
select  
case
```

```

when extract(HOUR FROM order_purchase_timestamp) between 0 and 6 THEN 'Dawn'
when extract(HOUR FROM order_purchase_timestamp) between 7 and 12 THEN 'Morning'
when extract(HOUR FROM order_purchase_timestamp) between 13 and 18 THEN
'Afternoon'
when extract(HOUR FROM order_purchase_timestamp) between 19 and 23 THEN 'Night'
end as hours,
COUNT(order_id) AS total_order_count
from
`Business_case_I.orders`
group by hours
order by total_order_count desc

```

The screenshot shows the Google Cloud BigQuery interface. On the left, the Explorer sidebar lists datasets like 'Business_case_I' and its tables: 'customers', 'geolocation', 'order_items', etc. In the center, a query editor window titled 'Untitled' contains the provided SQL code. The code includes a complex CASE statement for determining the time of day based on the hour of purchase. The 'RUN' button is visible at the top of the editor. Below the editor, the 'Query results' section displays a table with four rows, each representing a time period and its total order count. The table is as follows:

hours	total_order_count
Afternoon	38135
Night	28331
Morning	27733
Dawn	5242

Inference-maximum orders were placed in afternoon followed by night,morning and dawn

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

```
select extract(month from o.order_purchase_timestamp) as months,
count(o.order_id) as total_order,
(c.customer_state) as state
from `Business_case_I.orders` as o
inner join `Business_case_I.customers` as c
on c.customer_id= o.customer_id
group by c.customer_state,months
order by total_order desc
```

The screenshot shows the Google Cloud BigQuery interface. On the left, the Explorer sidebar displays workspace resources, including a folder named 'Business_case_I' containing tables like 'customers', 'geolocation', 'order_items', etc. In the center, an 'Untitled' query tab is open with the following SQL code:

```
45 from
46 `Business_case_I.orders`
47 group by hours
```

The 'RESULTS' tab is selected, showing the query results in a table:

Row	months	total_order	state
1	8	4982	SP
2	5	4632	SP
3	7	4381	SP
4	6	4104	SP
5	3	4047	SP
6	4	3967	SP
7	2	3357	SP
8	1	3351	SP
9	11	3012	SP
10	12	2357	SP
11	10	1908	SP

At the bottom of the interface, there are various system icons and a status bar indicating '29°C Mostly cloudy' and the date '8/20/2023'.

Inference-As we can observed SP state has highest no. of orders in the month of august

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

```
select count(customer_id) as customer,customer_state  
from `Business_case_I.customers`  
group by 2  
order by customer desc
```

The screenshot shows the Google Cloud BigQuery interface. On the left, the 'Explorer' sidebar lists datasets like 'Business_case_I' and 'farmers_market'. In the center, a query editor window titled 'Untitled' contains the following SQL code:

```
1 select count(customer_id) as customer,customer_state  
2 from `Business_case_I.customers`  
3 group by 2  
4 order by customer desc
```

The 'RESULTS' tab is selected, displaying the query results in a table:

customer	customer_state
41746	SP
12852	RJ
11635	MG
5466	RS
5045	PR
3637	SC
3380	BA
2140	DF
2033	ES
2020	GO
1652	PE

At the bottom of the interface, there are various status icons and a system tray showing the date and time.

Inference- data show us that SP state has highest no. of customers followed by RJ,MG, RS etc

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

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

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

```
select sum(oi.price) as total, avg(oi.price) as average, c.customer_state
from `Business_case_I.order_items` as oi
inner join `Business_case_I.orders` as o
on oi.order_id=o.order_id
inner join `Business_case_I.customers` as c
on o.customer_id=c.customer_id
group by 3
```

The screenshot shows the Google Cloud BigQuery interface. The left sidebar displays the 'Explorer' section with a tree view of datasets and tables, including 'Business_case_I' with tables like 'customers', 'geolocation', 'order_items', etc. The main area shows a query editor with the following code:

```
order by 2 desc
#Calculate the Total & Average value of order price for each state
select sum(oi.price) as total, avg(oi.price) as average, c.customer_state
from `Business_case_I.order_items` as oi
inner join `Business_case_I.orders` as o
on oi.order_id=o.order_id
inner join `Business_case_I.customers` as c
on o.customer_id=c.customer_id
group by 3
```

The results pane shows a table titled 'Query results' with the following data:

Row	total	average	customer_state
1	5202955.05000...	109.6536291597...	SP
2	1824092.669999...	125.1178180945...	RJ
3	683083.760000...	119.0041393728...	PR
4	520553.340000...	124.6535775862...	SC
5	302603.939999...	125.7705486284...	DF
6	1585308.029999...	120.7485741488...	MG
7	178947.809999...	165.6924166666...	PA
8	511349.990000...	134.6012082126...	BA
9	294591.949999...	126.2717316759...	GO
10	750304.020000...	120.3374530874...	RS

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

```
select round (sum(oi.freight_value),2 )as total,round (avg(oi.freight_value),2 ) as
average,c.customer_state
from `Business_case_I.order_items` as oi
inner join `Business_case_I.orders` as o
on oi.order_id=o.order_id
inner join `Business_case_I.customers` as c
on o.customer_id=c.customer_id
group by 3
order by total desc
```

The screenshot shows the Google Cloud BigQuery interface. On the left, the Explorer sidebar displays workspace resources under 'Business_case_1' and 'farmers_market'. The main area shows a query titled 'Untitled' with the following SQL code:

```
94 #Calculate the Total & Average value of order freight for each state
95 select round(sum(oi.freight_value),2) as total,round(avg(oi.freight_value),2) as average,c.customer_state
96 from `Business_case_1.order_items` as oi
```

The results table has three columns: Row, total, and average, with an additional header customer_state. The data is as follows:

Row	total	average	customer_state
1	718723.07	15.15	SP
2	305589.31	20.96	RJ
3	270853.46	20.63	MG
4	135522.74	21.74	RS
5	117851.68	20.53	PR
6	100156.68	26.36	BA
7	89660.26	21.47	SC
8	59449.66	32.92	PE
9	53114.98	22.77	GO
10	50625.5	21.04	DF

At the bottom, there are navigation links for PERSONAL HISTORY and PROJECT HISTORY, and system status indicators like weather (29°C Mostly cloudy), search bar, and system tray.

Inference - from output we can understand that state SP has highest no. of total freight value

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

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.

Do this in a single query.

You can calculate the delivery time and the difference between the estimated & actual delivery date using the given formula:

```
time_to_deliver = order_delivered_customer_date - order_purchase_timestamp
```

```
diff_estimated_delivery = order_estimated_delivery_date -  
order_delivered_customer_date
```

```
select customer_id,  
DATE_DIFF(order_delivered_customer_date, order_purchase_timestamp, DAY) AS  
delivery_time,  
DATE_DIFF(order_estimated_delivery_date, order_delivered_customer_date, DAY) AS  
difference_estimated_delivery  
from `Business_case_I.orders`  
order by delivery_time desc
```

The screenshot shows the Google BigQuery interface. At the top, there are several tabs and a search bar. Below the tabs, a sidebar displays workspace resources like 'customers', 'geolocation', 'order_items', etc. The main area shows a query editor with the following code:

```
101 group by 3  
102 order by total desc  
103  
104 Analysis based on sales, freight and delivery time.  
105 End of analysis of days taken to deliver each order from the order's purchase date as delivery time.
```

The 'RESULTS' tab is selected, showing a table with the following data:

customer_id	delivery_time	difference_estimated_delivery
7568392331068e2d281b11a...	209	-181
d306426abe5fc15e54b454d...	208	-188
7815125148cfa1e8c7feff79...	195	-165
1a8a4a30dc296976717f44e78...	194	-161
9cf2c2fa2632ee748e1a59c9a...	194	-166
217906bc11a32c1e470eb7e08...	194	-155
cb2caaaead400c97350c37af...	191	-175
65b14237885b3972ebec28c0f...	189	-167
8199345f576cd1be9701f924...	188	-159
9b39de85d94d55a21991e70b...	187	-144

At the bottom, there are navigation buttons for personal and project history, and a status bar showing '28°C Near record' and the date '8/20/2023'.

Inference - highest delivery time required to customer was 209 days and we need to improve in our delivery time to customers

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

```
select round (avg(oi.freight_value),2) as maxaverage,c.customer_state
from `Business_case_I.order_items` as oi
inner join `Business_case_I.orders` as o
on oi.order_id=o.order_id
inner join `Business_case_I.customers` as c
on o.customer_id=c.customer_id
group by 2
order by 1 desc
```

The screenshot shows the Google Cloud BigQuery interface. On the left, the Explorer sidebar displays datasets like Business_case_I and farmers_market, with tables such as customers, geolocation, order_items, order_reviews, orders, payments, products, and sellers. The main workspace shows a query titled "Untitled" with the following code:

```
118 #Find out the top 5 states with the highest & lowest average freight value.
120 select round (avg(oi.freight_value),2) as maxaverage,c.customer_state
--
```

The results tab shows the output of the query:

Row	maxaverage	customer_state
1	42.98	RR
2	42.72	PB
3	41.07	RO
4	40.07	AC
5	39.15	PI
6	38.26	MA
7	37.25	TO
8	36.65	SE
9	35.84	AL
10	35.83	PA
11	35.65	RN

At the bottom, the status bar indicates "27°C Mostly cloudy" and the system clock shows "8/20/2023 8:17 PM".

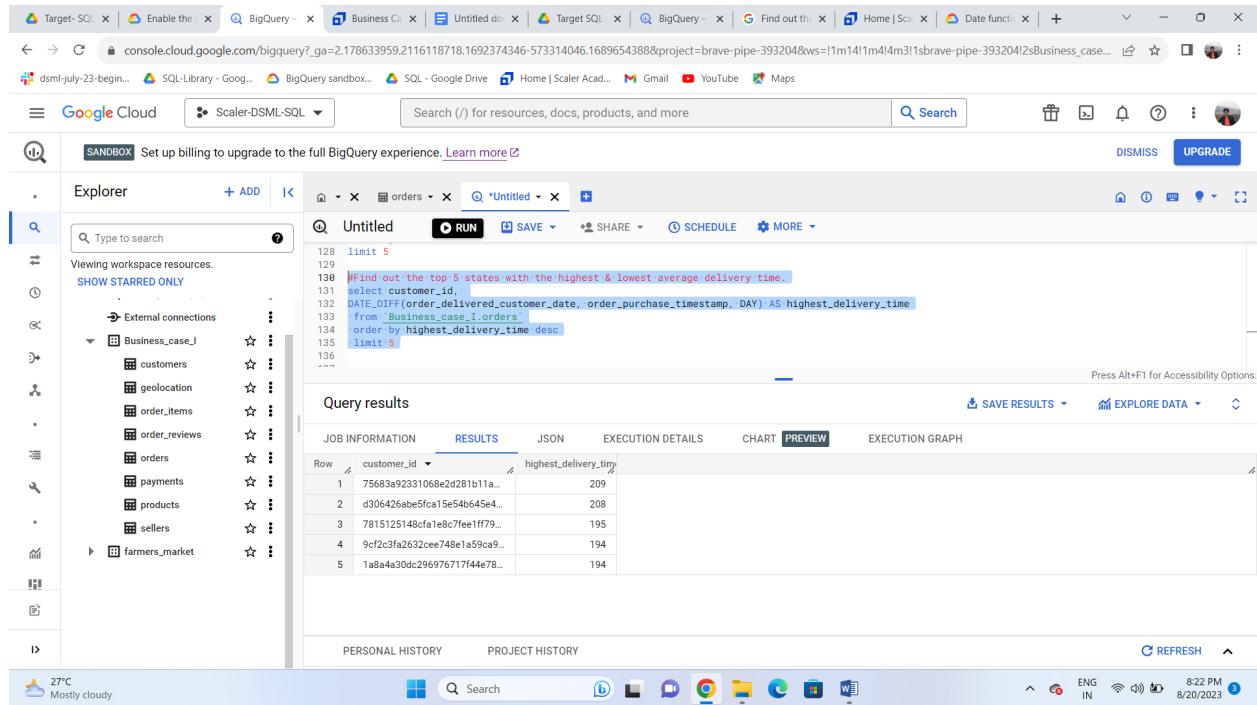
2. Find out the top 5 states with the highest & lowest average freight value.

```
select round (avg(oi.freight_value),2) as minaverage,c.customer_state
  from `Business_case_I.order_items` as oi
inner join `Business_case_I.orders` as o
on oi.order_id=o.order_id
inner join `Business_case_I.customers` as c
on o.customer_id=c.customer_id
group by 2
order by 1
limit 5
```

The screenshot shows the Google Cloud BigQuery interface. On the left, the 'Explorer' sidebar lists datasets like 'Business_case_I' and 'farmers_market'. In the center, a query editor window titled 'Untitled' contains the SQL code. Below it, the 'Query results' section displays a table with five rows of data. The table has columns 'Row', 'minaverage', and 'customer_state'. The data is as follows:

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

```
#Find out the top 5 states with the highest & lowest average delivery time.
select customer_id,
DATE_DIFF(order_delivered_customer_date, order_purchase_timestamp, DAY) AS
highest_delivery_time
from `Business_case_I.orders`
order by highest_delivery_time desc
limit 5
```



The screenshot shows the Google Cloud BigQuery interface. On the left, the Explorer sidebar displays workspace resources under the 'Sandbox' project. In the center, a query editor window titled 'Untitled' contains the SQL code provided above. The 'RESULTS' tab is selected, showing the following query results:

Row	customer_id	highest_delivery_time
1	75683a92331068e2d281b11a...	209
2	d306426abe5fc15e54b645e4...	208
3	7815125148cfa1e8c7feef79...	195
4	9cf2c3fa2632cee748e1a59ca9...	194
5	1a8a4a30dc29697671744e78...	194

```
#Find out the top 5 states with the highest & lowest average delivery time.
select customer_id,
DATE_DIFF(order_delivered_customer_date, order_purchase_timestamp, DAY) AS
lowest_delivery_time
from `Business_case_I.orders`
order by lowest_delivery_time
limit 5
```

The screenshot shows the Google Cloud BigQuery interface. On the left, the Explorer sidebar lists datasets like 'Business_case_I' containing tables such as 'customers', 'geolocation', 'order_items', etc. The main area displays a query editor with the following code:

```
136 #Find out the top 5 states with the highest & lowest average delivery time.
137 select customer_id,
138 DATE_DIFF(order_delivered_customer_date, order_purchase_timestamp, DAY) AS
139 lowest_delivery_time
140 from `Business_case_I.orders`
141 order by lowest_delivery_time
142 limit 5
143
144
```

Below the code, the 'Query results' section shows the output:

Row	customer_id	lowest_delivery_time
1	725e9c75605414b21fd8c8d5a...	null
2	4ee64fbfc542546f422da0ebe...	null
3	438449d4af89900d07bf04571...	null
4	964a6df3d9bd60fe3e7b8bb69...	null
5	7d61b9f4f216052ba664f22e9c...	null

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

You can use the difference between the averages of actual & estimated delivery date to figure out how fast the delivery was for each state.

```
select c.customer_id,
DATE_DIFF( (order_estimated_delivery_date), (order_delivered_customer_date), DAY) AS
difference_estimated_delivery,
c.customer_state
from `Business_case_I.orders` as o
inner join `Business_case_I.customers` as c
on o.customer_id=c.customer_id
order by difference_estimated_delivery

limit 5
```

The screenshot shows the Google Cloud BigQuery interface. The query has been run and the results are displayed in a table. The table has columns: Row, customer_id, difference_estimated_delivery, and customer_state. The results are as follows:

Row	customer_id	difference_estimated_delivery	customer_state
1	725e9c75605414b21f8c8d5a...	null	RJ
2	4ee64fbfc542546f422d00eb...	null	RS
3	438449d4af8990d107bf04571...	null	SP
4	964a6df3d9bdf60fe3e7b8bb69...	null	DF
5	7d61b9f4f216052ba664f22e9c...	null	PR

Inference

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

```
select (o.order_id), (extract (month from order_purchase_timestamp ) )as months,
payment_type
from `Business_case_I.orders` as o
inner join `Business_case_I.payments`as p
on o.order_id=p.order_id
order by extract (month from order_purchase_timestamp )
```

The screenshot shows the Google Cloud BigQuery interface. On the left, the 'Explorer' sidebar lists datasets like 'Business_case_I' containing tables such as 'customers', 'geolocation', 'order_items', etc. The main area displays a query editor with the following content:

```
171
172 #Find the month on month no. of orders placed using different payment types.
173
```

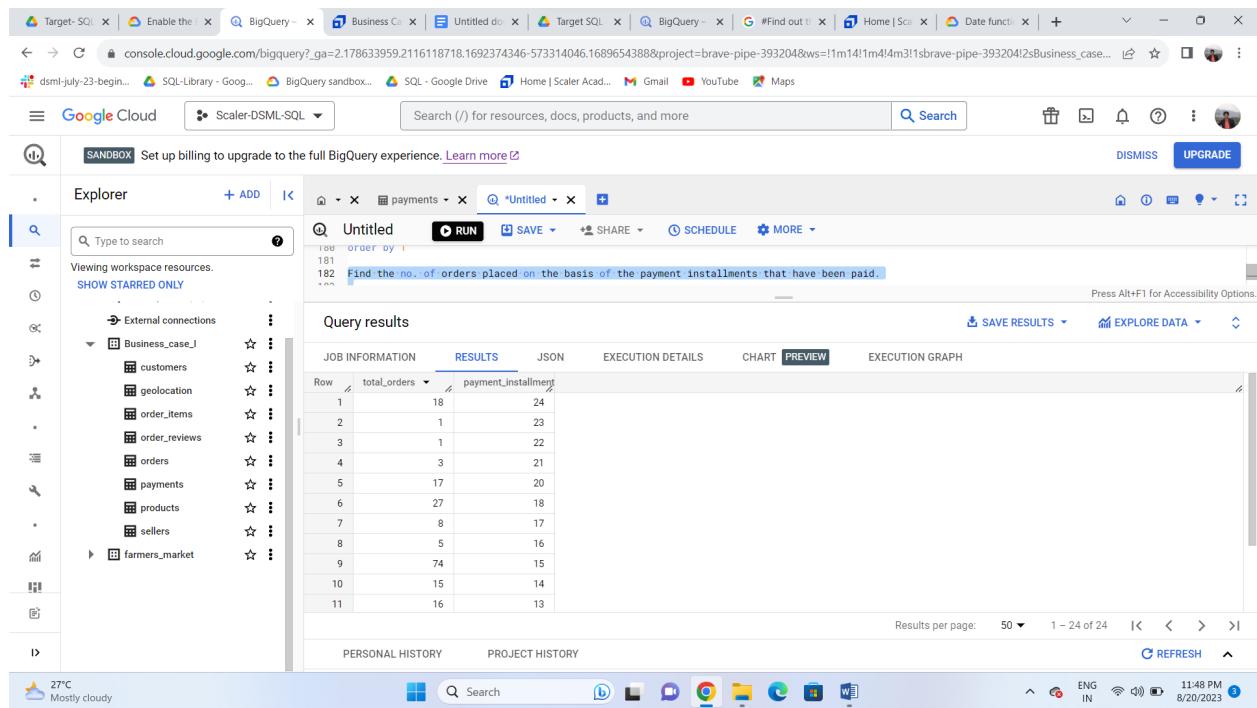
The 'RESULTS' tab is selected, showing the query results in a table format:

Row	order_id	months	payment_type
1	a34db9935aad747acfecadb...	1	credit_card
2	2d1fc5cc5ed32322159b08b86ff...	1	credit_card
3	7b9906348a4044ff73ce3034e...	1	credit_card
4	5dfcc26420fd2e45ee13e8dfb...	1	credit_card
5	f247dbea2a3d9e8bb68b00f...	1	credit_card
6	cb7911a7f5f016586ddcf6ea52...	1	credit_card
7	a0123fb74e1c68a8f3f4d14a...	1	credit_card
8	ef68f7ac518a9309ceb3a3290...	1	credit_card
9	9f53528dd92b28e67517b8128...	1	credit_card
10	251f7157dc5e425718ae076743...	1	credit_card
11	8a027244dea2edca1449d9150...	1	credit_card

Below the table, it says 'Results per page: 50 ▾ 1 – 50 of 103886'. The bottom status bar shows the date and time as 8/20/2023 11:31 PM.

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

```
select * from `Business_case_I.payments`  
  
select count(order_id) total_orders, payment_installments  
from `Business_case_I.payments`  
group by payment_installments  
order by 2 desc
```



The screenshot shows the Google Cloud BigQuery interface. The left sidebar displays the 'Explorer' section with a tree view of workspace resources, including 'Business_case_I' which contains tables like 'customers', 'payments', and 'orders'. The main area shows a query editor with the following code:

```
181  
182 Find the no. of orders placed on the basis of the payment installments that have been paid.  
183
```

The results pane displays the query results as a table:

Row	total_orders	payment_installment
1	18	24
2	1	23
3	1	22
4	3	21
5	17	20
6	27	18
7	8	17
8	5	16
9	74	15
10	15	14
11	16	13

At the bottom of the results pane, there are buttons for 'SAVE RESULTS' and 'EXPLORE DATA'. The status bar at the bottom right shows the date and time as 8/20/2023 11:48 PM.