

1.

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

```
## Data type of all columns in the "customers" table.
```

The screenshot shows a data schema editor interface. At the top, there are navigation links: 'customers' (highlighted), 'QUERY', 'SHARE', 'COPY', 'SNAPSHOT', 'DELETE', and 'EXPORT'. Below these are tabs: 'SCHEMA' (selected), 'DETAILS', 'PREVIEW', 'LINEAGE', 'DATA PROFILE', and 'DATA QUALITY'. A search bar labeled 'Filter' with the placeholder 'Enter property name or value' is present. A table lists the schema for the 'customers' table:

<input type="checkbox"/>	Field name	Type	Mode	Key	Collation	Default Value	Policy Tags	Description
<input type="checkbox"/>	customer_id	STRING	NULABLE					
<input type="checkbox"/>	customer_unique_id	STRING	NULABLE					
<input type="checkbox"/>	customer_zip_code_prefix	INTEGER	NULABLE					
<input type="checkbox"/>	customer_city	STRING	NULABLE					
<input type="checkbox"/>	customer_state	STRING	NULABLE					

At the bottom left are buttons for 'EDIT SCHEMA' (blue) and 'VIEW ROW ACCESS POLICIES' (white).

```
## Get the time range between which the orders were placed.
```

```
SELECT *,  
       latest_time - earliest_time AS range_time  
FROM  
(  
    SELECT MIN(EXTRACT(TIME FROM order_purchase_timestamp)) AS earliest_time,  
           MAX(EXTRACT(TIME FROM order_purchase_timestamp)) AS latest_time,  
           FROM `heroic-calculus-396815.Target_case_study.orders`  
) t;
```

[ACCESSIBILITY](#) | [FOR ACCESSIBILITY OPTIONS.](#)

[← Query results](#)

[SAVE RESULTS](#) [EXPLORE DATA](#) [▼](#)

JOB INFORMATION	RESULTS	CHART	PREVIEW	JSON	EXECUTION DETAILS	EXECUTION GRAPH
Row	earliest_time	latest_time	range_time			
1	00:00:00	23:59:59	0.0 0 23:59:59			

[PERSONAL HISTORY](#) [PROJECT HISTORY](#) [REFRESH](#) [^](#)

```
## Count the Cities & States of customers who ordered during the given period.
SELECT COUNT(DISTINCT geolocation_state) AS no_of_states,
       COUNT(DISTINCT geolocation_city) AS no_of_cities
FROM `heroic-calculus-396815.Target_case_study.geolocation`
WHERE geolocation_zip_code_prefix IN
(
  SELECT customer_zip_code_prefix
  FROM `heroic-calculus-396815.Target_case_study.customers`
  WHERE customer_id IN
  (
    SELECT customer_id
    FROM `heroic-calculus-396815.Target_case_study.orders`
  )
);

```

[← Query results](#)

[SAVE RESULTS](#) [EXPLORE DATA](#) [▼](#)

JOB INFORMATION	RESULTS	CHART	PREVIEW	JSON	EXECUTION DETAILS	EXECUTION GRAPH
Row	no_of_states	no_of_cities				
1	27	5812				

[PERSONAL HISTORY](#) [PROJECT HISTORY](#) [REFRESH](#) [^](#)

2.

#In-depth Exploration:

```
##Is there a growing trend in the no. of orders placed over the past years?
SELECT *,
       ROUND(((ttt.no_of_orders - previous_year_orders) / ttt.previous_year_orders) * 100,
0) AS growth_percentage
```

```

FROM
(
    SELECT year,
        tt.no_of_orders,
        LEAD(tt.no_of_orders, 1) OVER(ORDER BY year DESC) AS previous_year_orders
    FROM
    (
        SELECT COUNT(t.order_id) AS no_of_orders,
            t.year
        FROM
        (
            SELECT *,
                EXTRACT(YEAR FROM order_purchase_timestamp) AS year
            FROM `heroic-calculus-396815.Target_case_study.orders`
        ) t
        GROUP BY t.year
    ) tt
) ttt
ORDER BY year DESC;

```

← Query results

SAVE RESULTS EXPLORE DATA ▾

JOB INFORMATION		RESULTS	CHART	PREVIEW	JSON	EXECUTION DETAILS	EXECUTION GRAPH
Row	year	no_of_orders	previous_year_orders	growth_percentage			
1	2018	54011	45101	20.0			
2	2017	45101	329	13609.0			
3	2016	329	null	null			

PERSONAL HISTORY PROJECT HISTORY REFRESH ^

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

```

SELECT *
FROM
(
    (SELECT 1 AS month,
        8069 AS no_of_orders,
        7544 AS previous_month_order,
        ROUND(((8069 - 7544) / 7544) * 100, 0) AS growth_percentage)
    UNION ALL
    (SELECT *,

```

```

    ROUND(((ttt.no_of_orders - previous_month_orders) / ttt.previous_month_orders) *
100, 0) AS growth_percentage
FROM
(
SELECT month,
       tt.no_of_orders,
       LEAD(tt.no_of_orders, 1) OVER(ORDER BY month DESC) AS previous_month_orders
FROM
(
SELECT COUNT(t.order_id) AS no_of_orders,
       t.month
FROM
(
SELECT *,
       EXTRACT(MONTH FROM order_purchase_timestamp) AS Month
FROM `heroic-calculus-396815.Target_case_study.orders`
) t
GROUP BY t.month
) tt
) ttt
ORDER BY month ASC)
)
WHERE growth_percentage IS NOT NULL
ORDER BY month;

```

← Query results SAVE RESULTS ▾ EXPLORE DATA ▾

JOB INFORMATION		RESULTS		CHART	PREVIEW	JSON	EXECUTION DETAILS	EXECUTION GRAPH
Row	month	no_of_orders	previous_month_order	growth_percentage				
1	1	8069	7544	7.0				
2	2	8508	8069	5.0				
3	3	9893	8508	16.0				
4	4	9343	9893	-6.0				
5	5	10573	9343	13.0				
6	6	9412	10573	-11.0				
7	7	10318	9412	10.0				
8	8	10843	10318	5.0				
9	9	4305	10843	-60.0				
10	10	4959	4305	15.0				
11	11	7544	4959	52.0				
12	12	5674	7544	-25.0				

Results per page: 50 ▾ 1 – 12 of 12 | < < > >|

PERSONAL HISTORY PROJECT HISTORY REFRESH ^

3.

#Evolution of E-commerce orders in the Brazil region:

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

```

SELECT customer_state,
       month,
       COUNT(order_id) AS no_of_orders
FROM
(
    SELECT *, EXTRACT(MONTH FROM orders.order_purchase_timestamp) AS month
    FROM
        `heroic-calculus-396815.Target_case_study.orders` orders
        INNER JOIN `heroic-calculus-396815.Target_case_study.customers` customers
        ON orders.customer_id = customers.customer_id
) t
GROUP BY customer_state, t.MONTH
ORDER BY 1, 2;

```

← Query results

SAVE RESULTS ▾ EXPLORE DATA ▾

JOB INFORMATION		RESULTS	CHART	PREVIEW	JSON	EXECUTION DETAILS	EXECUTION GRAPH
Row	customer_state		month		no_of_orders		
1	AC		1		8		
2	AC		2		6		
3	AC		3		4		
4	AC		4		9		
5	AC		5		10		
6	AC		6		7		
7	AC		7		9		
8	AC		8		7		
9	AC		9		5		
10	AC		10		6		

Results per page: 50 ▾ 1 – 50 of 322 | < < > > |

PERSONAL HISTORY PROJECT HISTORY REFRESH ^

##How are the customers distributed across all the states?

```

SELECT customer_state,
       COUNT(customer_id) AS no_of_customers
FROM `heroic-calculus-396815.Target_case_study.customers`
GROUP BY customer_state
ORDER BY no_of_customers DESC;

```

Query results

JOB INFORMATION **RESULTS** **CHART** **PREVIEW** **JSON** **EXECUTION DETAILS** **EXECUTION GRAPH**

The screenshot shows a data preview interface with a table titled 'customer_state' containing 10 rows. The columns are 'Row', 'customer_state', and 'no_of_customers'. The data is as follows:

Row	customer_state	no_of_customers
1	SP	41746
2	RJ	12852
3	MG	11635
4	RS	5466
5	PR	5045
6	SC	3637
7	BA	3380
8	DF	2140
9	ES	2033
10	GO	2020

Results per page: 50 ▾ 1 – 27 of 27 |< < > >|

PERSONAL HISTORY PROJECT HISTORY **REFRESH** ^

4.

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

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

```
SELECT ROUND(((total_cost - previous_year_cost) / previous_year_cost) * 100, 2) AS growth_2017_to_2018
FROM
(
  SELECT *, LEAD(t.total_cost) OVER(ORDER BY year DESC) AS previous_year_cost
  FROM
  (
    SELECT o.year, SUM(p.payment_value) AS total_cost
    FROM
    (
      SELECT order_id,
        EXTRACT(YEAR FROM order_purchase_timestamp) AS year,
        EXTRACT(MONTH FROM order_purchase_timestamp) AS month
      FROM `heroic-calculus-396815.Target_case_study.orders`
    ) o
    LEFT JOIN
    `heroic-calculus-396815.Target_case_study.payments` p
    ON o.order_id = p.order_id
  )
```

```

        WHERE o.month BETWEEN 1 AND 8
        GROUP BY o.year
    ) t
) tt
WHERE previous_year_cost IS NOT NULL;

```

← Query results SAVE RESULTS ▾ EXPLORE DATA ▾

JOB INFORMATION		RESULTS	CHART	PREVIEW	JSON	EXECUTION DETAILS	EXECUTION GRAPH
Row	growth_2017_to_2018						
1		136.98					

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

```

SELECT c.customer_state,
    ROUND(SUM(p.payment_value), 2) AS Total_value,
    ROUND(AVG(p.payment_value), 2) AS Avg_value
FROM `heroic-calculus-396815.Target_case_study.orders` o
    INNER JOIN `heroic-calculus-396815.Target_case_study.customers` c
    ON o.customer_id = c.customer_id
    INNER JOIN `heroic-calculus-396815.Target_case_study.payments` p
    ON o.order_id = p.order_id
GROUP BY c.customer_state
ORDER BY 2 DESC, 3 DESC;

```

← Query results SAVE RESULTS ▾ EXPLORE DATA ▾

JOB INFORMATION		RESULTS	CHART	PREVIEW	JSON	EXECUTION DETAILS	EXECUTION GRAPH
Row	customer_state	Total_value	Avg_value				
1	SP	5998226.96	137.5				
2	RJ	2144379.69	158.53				
3	MG	1872257.26	154.71				
4	RS	890898.54	157.18				
5	PR	811156.38	154.15				
6	SC	623086.43	165.98				
7	BA	616645.82	170.82				
8	DF	355141.08	161.13				
9	GO	350092.31	165.76				
10	ES	325967.55	154.71				

Results per page: 50 ▾ 1 - 27 of 27 |< < > >|

[PERSONAL HISTORY](#) [PROJECT HISTORY](#) REFRESH ▾

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

```

SELECT c.customer_state,
    ROUND(SUM(oi.freight_value), 2) AS Total_frieght_value,
    ROUND(AVG(oi.freight_value), 2) AS AVG_frieght_value
FROM `heroic-calculus-396815.Target_case_study.order_items` oi
    INNER JOIN `heroic-calculus-396815.Target_case_study.orders` o
    ON oi.order_id = o.order_id

```

```

    INNER JOIN `heroic-calculus-396815.Target_case_study.customers` c
    ON o.customer_id = c.customer_id
GROUP BY c.customer_state
ORDER BY 3 DESC, 2 DESC;

```

← Query results SAVE RESULTS EXPLORE DATA

JOB INFORMATION		RESULTS	CHART	PREVIEW	JSON	EXECUTION DETAILS	EXECUTION GRAPH
Row	customer_state		Total_frieght_value	Avg_frieght_value			
1	RR		2235.19	42.98			
2	PB		25719.73	42.72			
3	RO		11417.38	41.07			
4	AC		3686.75	40.07			
5	PI		21218.2	39.15			
6	MA		31523.77	38.26			
7	TO		11732.68	37.25			
8	SE		14111.47	36.65			
9	AL		15914.59	35.84			
10	PA		38699.3	35.83			
Load more							

Results per page: 50 ▾ 1 – 27 of 27 |◀◀▶▶| [REFRESH](#)

5.

#Analysis based on sales, freight and delivery timeTABLE

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

```

SELECT DATETIME_DIFF(order_delivered_customer_date, order_purchase_timestamp, DAY) AS
days_to_deliver,
DATETIME_DIFF(order_estimated_delivery_date, order_delivered_customer_date, DAY) AS
day_diff_estimate
FROM `heroic-calculus-396815.Target_case_study.orders`;

```

← Query results SAVE RESULTS EXPLORE DATA

JOB INFORMATION		RESULTS	CHART	PREVIEW	JSON	EXECUTION DETAILS	EXECUTION GRAPH
Row	days_to_deliver		day_diff_estimate				
1	30		-12				
2	30		28				
3	35		16				
4	30		1				
5	32		0				
6	29		1				
7	43		-4				
8	40		-4				
9	37		-1				
10	33		-5				
Results per page: 50 ▾ 1 – 50 of 99441 ◀◀▶▶ REFRESH							
PERSONAL HISTORY PROJECT HISTORY							

```

##Find out the top 5 states with the highest & lowest average freight value.
--highest avg_frieght
WITH STATE_FREIGHT AS
(
  SELECT customers.customer_state, ROUND(AVG(order_items.freight_value), 2) AS
avg_freight_value
  FROM `heroic-calculus-396815.Target_case_study.order_items` order_items
  INNER JOIN `heroic-calculus-396815.Target_case_study.orders` orders
  ON order_items.order_id = orders.order_id
  INNER JOIN `heroic-calculus-396815.Target_case_study.customers` customers
  ON customers.customer_id = orders.customer_id
  GROUP BY customers.customer_state
)
SELECT *
FROM
(
  SELECT *,  

    RANK() OVER(ORDER BY avg_freight_value DESC) AS rank
  FROM state_freight
)
WHERE rank <= 5
ORDER BY rank;

```

← Query results

SAVE RESULTS ▾ EXPLORE DATA ▾ ⌂

JOB INFORMATION		RESULTS	CHART	PREVIEW	JSON	EXECUTION DETAILS	EXECUTION GRAPH
Row	customer_state			avg_freight_value	rank		
1	RR			42.98	1		
2	PB			42.72	2		
3	RO			41.07	3		
4	AC			40.07	4		
5	PI			39.15	5		

PERSONAL HISTORY PROJECT HISTORY REFRESH ⌂

```
--lowest avg_frieght
```

```

WITH STATE_FREIGHT AS
(

```

```

    SELECT customers.customer_state, ROUND(AVG(order_items.freight_value), 2) AS
avg_freight_value
    FROM `heroic-calculus-396815.Target_case_study.order_items` order_items
    INNER JOIN `heroic-calculus-396815.Target_case_study.orders` orders
    ON order_items.order_id = orders.order_id
    INNER JOIN `heroic-calculus-396815.Target_case_study.customers` customers
    ON customers.customer_id = orders.customer_id
    GROUP BY customers.customer_state
)

SELECT *
FROM
(
    SELECT *,
        RANK() OVER(ORDER BY avg_freight_value ASC) AS rank
    FROM state_freight
)
WHERE rank <= 5
ORDER BY rank;

```

← Query results

SAVE RESULTS ▾ EXPLORE DATA ▾

JOB INFORMATION		RESULTS	CHART	PREVIEW	JSON	EXECUTION DETAILS	EXECUTION GRAPH
Row	customer_state			avg_freight_value	rank		
1	SP			15.15	1		
2	PR			20.53	2		
3	MG			20.63	3		
4	RJ			20.96	4		
5	DF			21.04	5		

PERSONAL HISTORY PROJECT HISTORY

REFRESH ^

```

##Find out the top 5 states with the highest & lowest average delivery time.
--LOWEST
WITH new_table AS
(
    SELECT customers.customer_state,
        ROUND(AVG(DATETIME_DIFF(order_delivered_customer_date, order_purchase_timestamp,
DAY)), 2) AS delivery_time,
    FROM `heroic-calculus-396815.Target_case_study.orders` orders
    INNER JOIN `heroic-calculus-396815.Target_case_study.customers` customers
    ON orders.customer_id = customers.customer_id

```

```

        GROUP BY customers.customer_state
    )

SELECT customer_state AS state,
       delivery_time
FROM
(
    SELECT *,
           RANK() OVER(ORDER BY new_table.delivery_time ASC) AS rank
    FROM new_table
) t
WHERE rank <= 5
ORDER BY rank;

```

← Query results

SAVE RESULTS EXPLORE DATA ▾

JOB INFORMATION	RESULTS	CHART	PREVIEW	JSON	EXECUTION DETAILS	EXECUTION GRAPH
Row	state		delivery_time			
1	SP		8.3			
2	PR		11.53			
3	MG		11.54			
4	DF		12.51			
5	SC		14.48			

PERSONAL HISTORY PROJECT HISTORY REFRESH ▾

```

--HIGHEST
WITH new_table AS
(
    SELECT customers.customer_state,
           ROUND(AVG(DATETIME_DIFF(order_delivered_customer_date, order_purchase_timestamp,
DAY)), 2) AS delivery_time,
    FROM `heroic-calculus-396815.Target_case_study.orders` orders
    INNER JOIN `heroic-calculus-396815.Target_case_study.customers` customers
    ON orders.customer_id = customers.customer_id
    GROUP BY customers.customer_state
)

SELECT customer_state AS state,
       delivery_time

```

```

FROM
(
  SELECT *,
    RANK() OVER(ORDER BY new_table.delivery_time DESC) AS rank
  FROM new_table
) t
WHERE rank <= 5
ORDER BY rank;

```

← Query results

SAVE RESULTS EXPLORE DATA

Row	state	delivery_time
1	RR	28.98
2	AP	26.73
3	AM	25.99
4	AL	24.04
5	PA	23.32

PERSONAL HISTORY PROJECT HISTORY

REFRESH

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

```

SELECT customers.customer_state,
  ROUND(AVG(DATETIME_DIFF(order_estimated_delivery_date,
order_delivered_customer_date, DAY)), 2) AS day_diff_estimate
FROM `heroic-calculus-396815.Target_case_study.orders` orders
  INNER JOIN
`heroic-calculus-396815.Target_case_study.customers` customers
  ON orders.customer_id = customers.customer_id
GROUP BY customers.customer_state
ORDER BY 2 DESC
LIMIT 5;

```

[← Query results](#)[SAVE RESULTS](#) ▾[EXPLORE DATA](#) ▾

JOB INFORMATION	RESULTS	CHART	PREVIEW	JSON	EXECUTION DETAILS	EXECUTION GRAPH
Row	customer_state	day_diff_estimate				
1	AC		19.76			
2	RO		19.13			
3	AP		18.73			
4	AM		18.61			
5	RR		16.41			

PERSONAL HISTORY

PROJECT HISTORY

[REFRESH](#)

6.

#Analysis based on the payments

```
##Find the month on month no. of orders placed using different payment types.  
SELECT orders.month,  
       payments.payment_type,  
       COUNT(orders.order_id) AS no_of_orders  
FROM  
(  
    SELECT *, EXTRACT(MONTH FROM order_purchase_timestamp) AS month  
    FROM `heroic-calculus-396815.Target_case_study.orders`  
) orders  
INNER JOIN  
`heroic-calculus-396815.Target_case_study.payments` payments  
ON orders.order_id = payments.order_id  
GROUP BY orders.month, payments.payment_type  
ORDER BY orders.month, payments.payment_type;
```

[← Query results](#)

[SAVE RESULTS](#) [EXPLORE DATA](#) [▼](#)

JOB INFORMATION [RESULTS](#) CHART **PREVIEW** JSON EXECUTION DETAILS EXECUTION GRAPH

Row	month	payment_type	no_of_orders
1	1	UPI	1715
2	1	credit_card	6103
3	1	debit_card	118
4	1	voucher	477
5	2	UPI	1723
6	2	credit_card	6609
7	2	debit_card	82
8	2	voucher	424
9	3	UPI	1942
10	3	credit_card	7707

Results per page: 50 ▾ 1 – 50 of 50 | < < > >|

[PERSONAL HISTORY](#) [PROJECT HISTORY](#) [REFRESH](#) [^](#)

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

```
SELECT payment_installments,
       COUNT(order_id) AS no_of_orders
  FROM `heroic-calculus-396815.Target_case_study.payments`
 GROUP BY payment_installments;
```

[← Query results](#)

[SAVE RESULTS](#) [EXPLORE DATA](#) [▼](#)

JOB INFORMATION [RESULTS](#) CHART **PREVIEW** JSON EXECUTION DETAILS EXECUTION GRAPH

Row	payment_installment	no_of_orders
1	0	2
2	1	52546
3	2	12413
4	3	10461
5	4	7098
6	5	5239
7	6	3920
8	7	1626
9	8	4268
10	9	644

Results per page: 50 ▾ 1 – 24 of 24 | < < > >|

[PERSONAL HISTORY](#) [PROJECT HISTORY](#) [REFRESH](#) [^](#)