Target Case Study

Q1.

a. Query-
```
SELECT * EXCEPT(table_catalog, ordinal_position, is_stored, is_updatable, is_system_defined, clustering_ordinal_position) FROM `dsml-396802.target_case_studies`.INFORMATION_SCHEMA.COLUMNS WHERE table_name = 'customer';
```

Query results

SAVE RESULTS | EXPLORE DATA

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

Row	table_schema	table_name	column_name	is_nullable	data_type	is_generated
1	target_case_studies	customer	customer_id	YES	STRING	NEVER
2	target_case_studies	customer	customer_unique_id	YES	STRING	NEVER
3	target_case_studies	customer	customer_zip_code_prefix	YES	INT64	NEVER
4	target_case_studies	customer	customer_city	YES	STRING	NEVER
5	target_case_studies	customer	customer_state	YES	STRING	NEVER

Insights- customer table has usual columns but also a customer_unique_id column, which is a unique identifier number to each customer. It is diff from customer_id which is sequential identifier which is generated each time customer places an order.

b. Query-
```
WITH rangee AS(SELECT MIN(EXTRACT(TIME FROM order_purchase_timestamp))AS orderdate_min,MAX(EXTRACT(TIME FROM order_purchase_timestamp))AS orderdate_max FROM `dsml-396802.target_case_studies.orders`)SELECT r.orderdate_min,r.orderdate_max,TIME_DIFF(TIME (r.orderdate_max), TIME (r.orderdate_min), HOUR) AS time_range FROM rangee r;
```

Query results

SAVE RESULTS | EXPLORE DATA

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

Row	orderdate_min	orderdate_max	time_range
1	00:00:00	23:59:59	23

Insights- Items are ordered as early as 12:00 AM morning and as late as 11:59 PM evening And range between them is 23 Hour.

Assumptions- I considered that the que is asking for time difference between minimum and maximum time from order table.

c. Query-
```
SELECT g.geolocation_city,g.geolocation_state,COUNT(DISTINCT g.geolocation_city) OVER() AS city_count,COUNT(DISTINCT g.geolocation_state) OVER() AS state_count FROM `dsml-396802.target_case_studies.geolocation` g JOIN
```

```
`dsml-396802.target_case_studies.customer` c ON g.geolocation_zip_code_prefix =
c.customer_zip_code_prefix LIMIT 10;
```

Insights- Customers have ordered from 5812 Distinct Cities Spanning across 27 States.

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

Row	geolocation_city	geolocation_state	city_count	state_count
1	aracaju	SE	5812	27
2	aracaju	SE	5812	27
3	aracaju	SE	5812	27
4	aracaju	SE	5812	27
5	aracaju	SE	5812	27
6	aracaju	SE	5812	27
7	aracaju	SE	5812	27
8	aracaju	SE	5812	27
9	aracaju	SE	5812	27

Assumptions- Time period was assumed to be between 2016-2018

----------xxxxxxxxx----------
-

Q2.

a. Query-
```
WITH simp AS(SELECT EXTRACT(YEAR FROM order_purchase_timestamp) AS
Year,COUNT(order_purchase_timestamp) AS Number_Of_Orders FROM
`dsml-396802.target_case_studies.orders` WHERE EXTRACT(YEAR FROM
order_purchase_timestamp) = 2016 GROUP BY EXTRACT(YEAR FROM
order_purchase_timestamp)

UNION ALL

SELECT EXTRACT(YEAR FROM order_purchase_timestamp) AS year,
COUNT(order_purchase_timestamp) AS no_of_orders FROM
`dsml-396802.target_case_studies.orders` WHERE EXTRACT(YEAR FROM
order_purchase_timestamp) = 2017 GROUP BY EXTRACT(YEAR FROM
order_purchase_timestamp)

UNION ALL

SELECT EXTRACT(YEAR FROM order_purchase_timestamp) AS count_2018,
COUNT(order_purchase_timestamp) AS no_of_orders FROM
`dsml-396802.target_case_studies.orders` WHERE EXTRACT(YEAR FROM
order_purchase_timestamp) = 2018 GROUP BY EXTRACT(YEAR FROM
order_purchase_timestamp))
```

```
SELECT *,ROUND((Number_Of_Orders-LAG(Number_Of_Orders,1) OVER(ORDER BY
Year))/LAG(Number_Of_Orders,1) OVER(ORDER BY Year) * 100,0) AS Percentage FROM
simp s ORDER BY s.Year;
```

Query results

JOB INFORMATION | RESULTS | JSON | EXECUTION DETAILS

Row	Year	Number_Of_Orders	Percentage
1	2016	329	*null*
2	2017	45101	13609.0
3	2018	54011	20.0

Insights- Their has been a sudden change in the number of order booked from 2016 to 2017, From 2016 to 2017 their has been 13609% change in number of orders and subsequently 20% in next Year.

Assumption- This query was written assuming that que is asking for order booked rather than order confirmed.

b. Query-
```
WITH c1 AS(SELECT FORMAT_DATETIME('%B', order_purchase_timestamp) AS
monthly, COUNT(order_id) AS order_count FROM
`dsml-396802.target_case_studies.orders` WHERE EXTRACT(YEAR FROM
order_purchase_timestamp) = 2017 GROUP BY FORMAT_DATETIME('%B',
order_purchase_timestamp))

SELECT * FROM c1 ORDER BY c1.order_count DESC
```

Query results

JOB INFORMATION | RESULTS | JSON | EXECUTION

Row	monthly	order_count
1	November	7544
2	December	5673
3	October	4631
4	August	4331
5	September	4285
6	July	4026
7	May	3700
8	June	3245
9	March	2682
10	April	2404

Insights- Observation clearly states that number of orders booked slowly increases through SUMMER and Peaks during WINTER i.e. NOVEMBER. After that it slowly tapers off.

Assumption- Seasonality trend data is taken only from 2017 as similar trend would be seen for 2018 as well.

c. Query-

```
WITH tbl AS (SELECT CASE
   WHEN EXTRACT(HOUR FROM order_purchase_timestamp) BETWEEN 00 AND 06 THEN
'Dawn'
   WHEN EXTRACT(HOUR FROM order_purchase_timestamp) BETWEEN 07 AND 12 THEN
'Morning'
   WHEN EXTRACT(HOUR FROM order_purchase_timestamp) BETWEEN 13 AND 18 THEN
'Afternoon'
   ELSE 'Night'
END AS Time_of_Day,o.order_id FROM `dsml-396802.target_case_studies.customer` c
JOIN `dsml-396802.target_case_studies.orders` o ON c.customer_id=o.customer_id)
```

```
SELECT Time_of_Day, COUNT(order_id) AS Num_of_Orders FROM tbl GROUP BY
Time_of_Day
```

Row	Time_of_Day	Num_of_Orders
1	Morning	27733
2	Dawn	5242
3	Afternoon	38135
4	Night	28331

Insights- Brazilian Customers mostly orders during 'Afternoon' followed by Night and Morning with Dawn being last.

----------xxxxxxxxx----------

Q3.

a. Query-
```
WITH tbl AS(SELECT c.customer_state,EXTRACT(MONTH FROM
o.order_purchase_timestamp) AS
month,FORMAT_DATETIME('%B',o.order_purchase_timestamp)AS
monthly,COUNT(o.order_id) AS num_order FROM
`dsml-396802.target_case_studies.orders` o JOIN
`dsml-396802.target_case_studies.customer` c ON o.customer_id=c.customer_id
GROUP BY c.customer_state,month,monthly ORDER BY c.customer_state,month)

SELECT customer_state,monthly,num_order,ROUND((num_order-LAG(num_order,1)
OVER(PARTITION BY customer_state ORDER BY month))/LAG(num_order,1) OVER(PARTITION BY
customer_state ORDER BY month)*100,0) AS Month_on_Month_Change FROM tbl ORDER BY
customer_state,month LIMIT 10
```

Row	customer_state	monthly	num_order	Month_on_Month_Ch
1	AC	January	8	*null*
2	AC	February	6	-25.0
3	AC	March	4	-33.0
4	AC	April	9	125.0
5	AC	May	10	11.0
6	AC	June	7	-30.0
7	AC	July	9	29.0
8	AC	August	7	-22.0
9	AC	September	5	-29.0
10	AC	October	6	20.0

b. Query- SELECT customer_state, COUNT(customer_unique_id) AS Customer_Count FROM `dsml-396802.target_case_studies.customer` GROUP BY customer_state ORDER BY Customer_Count DESC LIMIT 10

Row	customer_state	Customer_Count
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

Insights- Majority of customers are staying in States like 'SP','RJ','MG'.

---xxxxxxxxx---

Q4.

a. Query-

```
WITH mom_sales AS (

SELECT

EXTRACT(MONTH FROM o.order_purchase_timestamp) AS monthly,

FORMAT_DATETIME('%B', o.order_purchase_timestamp) AS Month,

ROUND(SUM(CASE WHEN EXTRACT(YEAR FROM o.order_purchase_timestamp) = 2017 THEN
p.payment_value ELSE 0 END ),0) AS payment_2017,

ROUND(SUM(CASE WHEN EXTRACT(YEAR FROM o.order_purchase_timestamp) = 2018 THEN
p.payment_value ELSE 0 END ),0) AS payment_2018 FROM
`dsml-396802.target_case_studies.payments` p JOIN
`dsml-396802.target_case_studies.orders` o ON p.order_id = o.order_id WHERE
```

```
    EXTRACT(MONTH FROM o.order_purchase_timestamp) BETWEEN 1 AND 8 GROUP BY
    monthly, Month ORDER BY monthly)

SELECT Month,payment_2017,payment_2018,ROUND((payment_2018-payment_2017)/payment_2017
* 100,2) AS per_change FROM mom_sales
```

Row	Month	payment_2017	payment_2018	per_change
1	January	138488.0	1115004.0	705.13
2	February	291908.0	992463.0	239.99
3	March	449864.0	1159652.0	157.78
4	April	417788.0	1160785.0	177.84
5	May	592919.0	1153982.0	94.63
6	June	511276.0	1023880.0	100.26
7	July	592383.0	1066541.0	80.04
8	August	674396.0	1022425.0	51.61

Insights- Sales comparison we can see that sales are higher compared to 2017. Sales are in higher 7 Digit Mark in 2018.

b. Query-

```
SELECT

 c.customer_state,

 ROUND(SUM(p.payment_value),2) AS Total_Value,

 ROUND(AVG(p.payment_value),2) AS Avg_Value

FROM `dsml-396802.target_case_studies.customer`c JOIN
`dsml-396802.target_case_studies.orders` o

ON c.customer_id = o.customer_id

JOIN `dsml-396802.target_case_studies.payments` p

ON o.order_id = p.order_id

GROUP BY c.customer_state

ORDER BY Total_Value DESC

LIMIT 10;
```

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

Assumptions- Total & Avg value are considered and added for all the years.

c. Query-

```
SELECT
 c.customer_state,
 ROUND(SUM(p.freight_value),2) AS Total_Value,
 ROUND(AVG(p.freight_value),2) AS Avg_Value
FROM `dsml-396802.target_case_studies.customer`c JOIN
`dsml-396802.target_case_studies.orders` o
ON c.customer_id = o.customer_id
JOIN `dsml-396802.target_case_studies.order_item` p
ON o.order_id = p.order_id
GROUP BY c.customer_state
ORDER BY Total_Value DESC
LIMIT 10
```

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

Insights- Where the avg freight is less than avg freight value of other states, that state ships more orders. As can be seen in total freight value. Although other factors such as size of population might also be a reason for this.

---xxxxxxxxx---

Q5.

a. Query:

```
SELECT

 order_id,

 DATE_DIFF(order_delivered_customer_date,order_purchase_timestamp,DAY) AS
Delivery_Time,

 DATE_DIFF(order_delivered_customer_date,order_estimated_delivery_date,DAY) AS
Estimated_Time_Diff

FROM `dsml-396802.target_case_studies.orders`
```

```
WHERE order_status = 'delivered'

LIMIT 10
```

Row	order_id	Delivery_Time	Estimated_Time_Diff
1	c158e9806f85a33877bdfd4f60...	23	-9
2	b60b53ad0bb7dacacf2989fe2...	12	5
3	c830f223aae08493ebecb52f2...	12	-12
4	a8aa2cd070eeac7e4368cae3d...	7	-1
5	813c55ce9b6baa8f879e064fbf...	12	-9
6	44558a1547e448b41c48c4087...	1	-5
7	036b791897847cdb8e39df794...	6	0
8	1aba60c04110bdd421b250ea3...	21	-7
9	0312ecf90786def87f98aa19e0...	7	0
10	635c894d068ac37e6e03dc54e...	30	-1

Insights- Majority Of Orders are Reaching Early as concluded from further analysis by comparing the number of orders reaching before and after. About 6500 Orders reached late and more than 85000 orders reached on time.

b. Query:

```
SELECT

 q1.geolocation_state,

 q1.AVG_Value

FROM(SELECT

 g.geolocation_state,

 ROUND(AVG(p.freight_value),2) AS AVG_Value,

FROM `dsml-396802.target_case_studies.sellers` s JOIN
`dsml-396802.target_case_studies.order_item` p

ON s.seller_id = p.seller_id  JOIN
`dsml-396802.target_case_studies.geolocation` g

ON s.seller_zip_code_prefix = g.geolocation_zip_code_prefix
```

```
GROUP BY g.geolocation_state

ORDER BY AVG_Value DESC

LIMIT 5) q1

UNION ALL

SELECT

 q2.geolocation_state,

 q2.AVG_Value

FROM(SELECT

 g.geolocation_state,

 ROUND(AVG(p.freight_value),2) AS AVG_Value

FROM `dsml-396802.target_case_studies.sellers` s JOIN
`dsml-396802.target_case_studies.order_item` p

ON s.seller_id = p.seller_id  JOIN
`dsml-396802.target_case_studies.geolocation` g

ON s.seller_zip_code_prefix = g.geolocation_zip_code_prefix

GROUP BY g.geolocation_state

ORDER BY AVG_Value ASC

LIMIT 5) q2
```

Row	geolocation_state	AVG_Value
1	CE	54.44
2	RO	50.32
3	PI	36.94
4	PB	34.69
5	AC	32.84
6	RN	15.93
7	SP	18.44
8	RJ	18.93
9	DF	18.99
10	PR	22.11

Ps. Top 5 are states with highest Avg freight value and Bottom 5 are states with lowest freight value.

c. Query:

```
SELECT customer_state,Avg_Delivery_Time

FROM(SELECT

  c.customer_state,


ROUND(AVG(DATE_DIFF(DATE(o.order_delivered_customer_date),DATE(o.order_purchase
_timestamp), DAY)),2) AS Avg_Delivery_Time

FROM `dsml-396802.target_case_studies.orders` o JOIN
`dsml-396802.target_case_studies.customer` c

ON o.customer_id = c.customer_id

WHERE o.order_status = 'delivered' AND EXTRACT(YEAR FROM
o.order_purchase_timestamp) = 2018

GROUP BY c.customer_state

ORDER BY Avg_Delivery_Time DESC

LIMIT 5) T1

UNION ALL

SELECT customer_state,Avg_Delivery_Time

FROM(SELECT

  c.customer_state,


ROUND(AVG(DATE_DIFF(DATE(o.order_delivered_customer_date),DATE(o.order_purchase
_timestamp), DAY)),2) AS Avg_Delivery_Time

FROM `dsml-396802.target_case_studies.orders` o JOIN
`dsml-396802.target_case_studies.customer` c

ON o.customer_id = c.customer_id

WHERE o.order_status = 'delivered' AND EXTRACT(YEAR FROM
o.order_purchase_timestamp) = 2018
```

```
GROUP BY c.customer_state

ORDER BY Avg_Delivery_Time ASC

LIMIT 5) T2
```

Row	customer_state	Avg_Delivery_Time
1	SP	8.22
2	PR	11.53
3	MG	11.79
4	DF	12.37
5	SC	14.51
6	RR	28.05
7	AM	27.13
8	AP	25.97
9	PA	25.41
10	AL	23.59

P.S.- Top 5 are states with lowest delivery time and Bottom 5 are states with highest delivery time.

Insights- 'SP' The state with the highest freight and payment value is also the state with lowest delivery time.

Assumption- Data of 2018 is considered for this query as data of a year should give more accurate data, instead of aggregation of avg_delivery_time of all years.

d. Query-

```
WITH T1 AS(SELECT

  c.customer_state,


ROUND(AVG(DATE_DIFF(DATE(o.order_delivered_customer_date),DATE(o.order_purchase
_timestamp), DAY)),2) AS Avg_Delivery_Time,


ROUND(AVG(DATE_DIFF(DATE(o.order_estimated_delivery_date),DATE(o.order_purchase
_timestamp), DAY)),2) AS Avg_Est_Time
```

```
FROM `dsml-396802.target_case_studies.orders` o JOIN
`dsml-396802.target_case_studies.customer` c

ON o.customer_id = c.customer_id

WHERE o.order_status = 'delivered' AND EXTRACT(YEAR FROM
o.order_purchase_timestamp) = 2018

GROUP BY c.customer_state

ORDER BY Avg_Delivery_Time DESC)

SELECT customer_state

FROM T1

WHERE T1.Avg_Delivery_Time < T1.Avg_Est_Time

LIMIT 5
```

Row	customer_state
1	RR
2	AM
3	AP
4	PA
5	AL

Insights- 'RR' the state with highest Delivery Time, Coincidently is also the state where delivery time is less when compared to est. delivery time.

Assumptions- Here also we are taking data of the year 2018, as values are accurate and doesn't get mix with values of previous years.