NAME:- Supratick Dey

ANSWERS:

Q1. Import the dataset and do usual exploratory analysis steps like checking the structure & characteristics of the dataset.

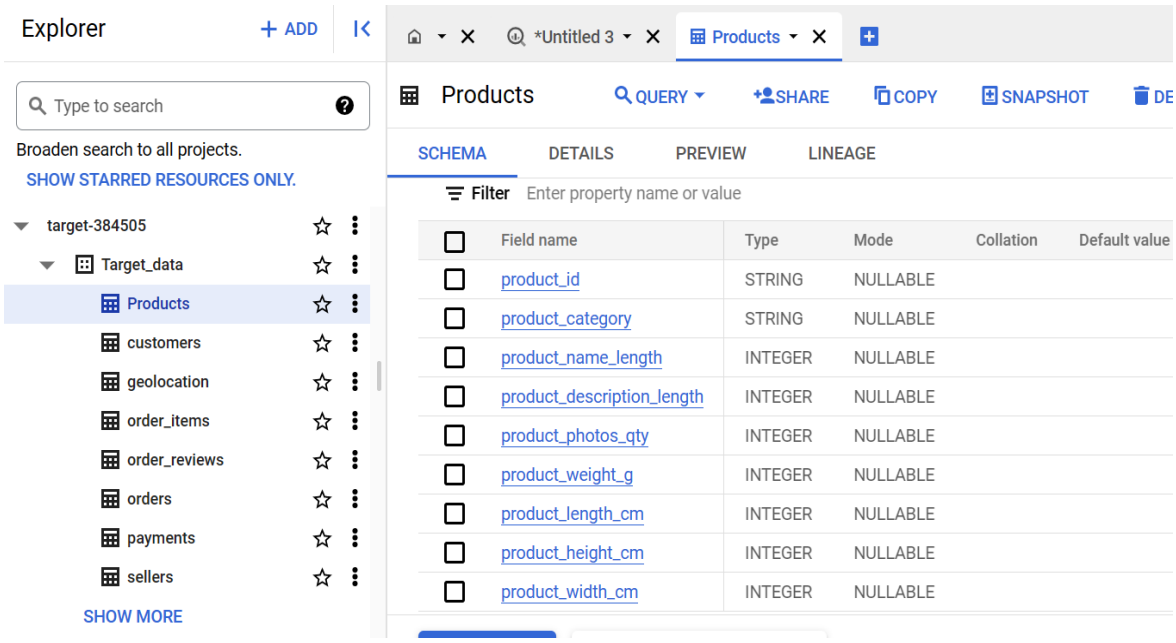
1. Data type of columns in a table
2. Time period for which the data is given
3. Cities and States of customers ordered during the given period

A1.

1. How to find the Data type of columns in a table

There are two ways of doing it .

a. By clicking the table name under the data set in google big query.



b. Second method is by providing query as below:-

```
select column_name,data_type
FROM `target-384505.Target_data.INFORMATION_SCHEMA.COLUMN_FIELD_PATHS`
WHERE table_name = 'Products'
```

```
select column_name,data_type
FROM `target-384505.Target_data.INFORMATION_SCHEMA.COLUMN_FIELD_PATHS`
WHERE table_name = 'Products'
```

Query results

SA

JOB INFORMATION | RESULTS | JSON | EXECUTION DETAILS | EX

Row	column_name	data_type
1	product_id	STRING
2	product_category	STRING
3	product_name_length	INT64
4	product_description_length	INT64
5	product_photos_qty	INT64
6	product_weight_g	INT64
7	product_length_cm	INT64
8	product_height_cm	INT64

Above both examples are for Products table.

Here are the schema for other tables:-

Customers table:-

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

Geolocation Table:-

Filter Enter property name or value

Field name	Type	Mode	Collat
geolocation_zip_code_prefix	INTEGER	NULLABLE	
geolocation_lat	FLOAT	NULLABLE	
geolocation_lng	FLOAT	NULLABLE	
geolocation_city	STRING	NULLABLE	
geolocation_state	STRING	NULLABLE	

Order_items table:-

Filter Enter property name or value

Field name	Type	Mode	Collati
order_id	STRING	NULLABLE	
order_item_id	INTEGER	NULLABLE	
product_id	STRING	NULLABLE	
seller_id	STRING	NULLABLE	
shipping_limit_date	TIMESTAMP	NULLABLE	
price	FLOAT	NULLABLE	
freight_value	FLOAT	NULLABLE	

Order_review table:-

Filter Enter property name or value

Field name	Type	Mode	Collation
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	

Orders table:-

Field name	Type	Mode	Collation	De
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		

Payments table:-

ilter Enter property name or value

Field name	Type	Mode	Collation	D
order_id	STRING	NULLABLE		
payment_sequential	INTEGER	NULLABLE		
payment_type	STRING	NULLABLE		
payment_installments	INTEGER	NULLABLE		
payment_value	FLOAT	NULLABLE		

Sellers table:-

Field name	Type	Mode	Collation
seller_id	STRING	NULLABLE	
seller_zip_code_prefix	INTEGER	NULLABLE	
seller_city	STRING	NULLABLE	
seller_state	STRING	NULLABLE	

2. Time period for which data was given is 2016 -2018

For month wise the minimum order purchase date was 2016- sept and max date was 2018-oct.

```
SELECT CONCAT(EXTRACT(year from MIN(order_purchase_timestamp)),'-
',FORMAT_DATETIME('%b',MIN(order_purchase_timestamp))) as min_date FROM `target-
384505.Target_data.orders;

SELECT CONCAT(EXTRACT(year from MAX(order_purchase_timestamp)),'-
',FORMAT_DATETIME('%b',MAX(order_purchase_timestamp))) as max_date FROM `target-
384505.Target_data.orders` ;
```

```
SELECT CONCAT(EXTRACT(year from MIN
(order_purchase_timestamp)),'-',FORMAT_DATETIME('%b',
MIN(order_purchase_timestamp))) as min_date FROM
`target-384505.Target_data.orders` ;

SELECT CONCAT(EXTRACT(year from MAX
(order_purchase_timestamp)),'-',FORMAT_DATETIME('%b',
MAX(order_purchase_timestamp))) as min_date FROM
`target-384505.Target_data.orders` ;
```

Press Alt+F1 for accessibility optio

ery results

JOB INFORMATION | RESULTS | JSON | EX

min_date
2016-Sep

```
SELECT CONCAT(EXTRACT(year from MAX
(order_purchase_timestamp)),'-',FORMAT_DATETIME('%b',
MAX(order_purchase_timestamp))) as max_date FROM
`target-384505.Target_data.orders` ;
```

Press Alt+F1 for accessibility options.

ıery results

JOB INFORMATION | RESULTS | JSON | EX

	max_date
1	2018-Oct

For month wise minimum order delivery date is 2016 – Oct and maximum order delivery date is 2018-oct.

```
SELECT CONCAT(EXTRACT(year from MIN(order_delivered_customer_date)),'-
',FORMAT_DATETIME('%b',MIN(order_delivered_customer_date))) as min_date FROM `target-
384505.Target_data.orders`;

SELECT CONCAT(EXTRACT(year from MAX(order_delivered_customer_date)),'-
',FORMAT_DATETIME('%b',MAX(order_delivered_customer_date))) as max_date FROM `target-
384505.Target_data.orders` ;
```

```
5 SELECT CONCAT(EXTRACT(year from MIN
  (order_delivered_customer_date)),'-',FORMAT_DATETIME
  ('%b',MIN(order_delivered_customer_date))) as
  min_date FROM `target-384505.Target_data.orders`;
6
7 SELECT CONCAT(EXTRACT(year from MAX
  (order_delivered_customer_date)),'-',FORMAT_DATETIME
  ('%b',MAX(order_delivered_customer_date))) as
  max_date FROM `target-384505.Target_data.orders` ;
```

Press Alt+F1 for accessibility options.

Query results

JOB INFORMATION | RESULTS | JSON | EX

ow	min_date
1	2016-Oct

```
7 SELECT CONCAT(EXTRACT(year from MAX
  (order_delivered_customer_date)),'-',FORMAT_DATETIME
  ('%b',MAX(order_delivered_customer_date))) as
  max_date FROM `target-384505.Target_data.orders` ;
```

Press Alt+F1 for accessibility options.

Query results

JOB INFORMATION | RESULTS | JSON | EX

w	max_date
1	2018-Oct

3. City and state for the given time period :-

```
SELECT distinct geolocation_city as city,geolocation_state as state FROM `targe
t-384505.Target_data.geolocation` g inner join `target-
384505.Target_data.customers` c ON g.geolocation_zip_code_prefix = c.customer_z
ip_code_prefix
WHERE EXISTS (SELECT 1 FROM `Target_data.orders` o where o.customer_id = c.cust
omer_id );
```

```
1  SELECT distinct geolocation_city as city,geolocation_state as state FROM `target-384505.Target_data.geolocation` g
   inner join `target-384505.Target_data.customers` c ON g.geolocation_zip_code_prefix = c.customer_zip_code_prefix
2  WHERE EXISTS (SELECT 1 FROM `Target_data.orders` o where o.customer_id = c.customer_id );
```

Press Alt+F1 for accessibility options

Query results

SAVE RESULTS EXPLORE DATA

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

Row	city	state
1	aracaju	SE
2	riachuelo	SE
3	nossa senhora do socorro	SE
4	barra dos coqueiros	SE
5	itaporanga d'ajuda	SE
6	sao cristovao	SE
7	são cristóvão	SE
8	santo amaro das brotas	SE
9	pirambu	SE
10	umbauba	SE
11	estancia	SE
12	itabaianinha	SE

2. In-depth Exploration:

    1. Is there a growing trend on e-commerce in Brazil? How can we describe a complete scenario? Can we see some seasonality with peaks at specific months?

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

A2.

1.
```
select EXTRACT(year from order_delivered_carrier_date) as order_date,Count(o
rder_id) as order_count from `Target_data.orders`  where EXTRACT(year from o
rder_delivered_carrier_date) IS NOT NULL group by order_date order by order_
date;
```

Row	order_date	order_count
1	2016	281
2	2017	43328
3	2018	54049

As we can see, with each consecutive year the order count has only increased, which indicates a positive growth trend for e-commerce in Brazil.

```
SELECT b.order_date,b.order_month,b.order_count
FROM(select a.order_date,a.order_month,a.order_count,dense_rank() over(partitio
n by a.order_date order by a.order_count desc) as ranking
FROM(select EXTRACT(year from order_delivered_carrier_date) as order_date,FORMA
T_DATETIME('%b',order_delivered_carrier_date) as order_month,Count(order_id) as
 order_count
from `Target_data.orders`  where EXTRACT(year from order_delivered_carrier_date
) IS NOT NULL group by order_date,order_month order by order_date) a
order by a.order_date) b
WHERE b.ranking = 1;
```

JOB INFORMATION | RESULTS | JSON | EXECUTION DETAILS

Row	order_date	order_month	order_count
1	2016	Oct	247
2	2017	Nov	6637
3	2018	Mar	7178

From the above analysis we can conclude that the peak month in every year is different.
For 2016 ,Oct had the highest order count of 247. For 2017 ,Nov had the highest order count of 6637 and In 2018 March it had all time highest with the order count of 7178.

2.

```
select case when EXTRACT(HOUR from order_purchase_timestamp) >= 0 and EXTRAC
T(HOUR from order_purchase_timestamp)<=4 then 'Dawn' when EXTRACT(HOUR from
order_purchase_timestamp) >= 5 and EXTRACT(HOUR from order_purchase_timestam
p)<=11 then 'Morning' when EXTRACT(HOUR from order_purchase_timestamp)>=12 a
nd EXTRACT(HOUR from order_purchase_timestamp)<=16 then 'Afternoon' when EXT
RACT(HOUR from order_purchase_timestamp)>=17 and EXTRACT(HOUR from order_pur
chase_timestamp)<=21 then 'Evening' else 'Night' end As timestmp, Count(orde
r_id) as order_placed
FROM `Target_data.orders`
Group By timestmp;
```

Row	timestmp	order_placed
1	Morning	22428
2	Dawn	4552
3	Evening	30311
4	Afternoon	32211
5	Night	9939

As from the above picture we can see mostly customers like to buy in Evening or Afternoon. With the most purchases made during Afternoon we can conclude customers like to order/purchase mostly at afternoon.

But Normally customers would make purchase at any time of the day but Morning ,Evening and Afternoon being the most customer's preferred purchasing time.

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

    1. Get month on month orders by states

    2. Distribution of customers across the states in Brazil

A3.

1.
```
select c.customer_state AS state, COUNT(distinct CASE WHEN FORMAT_DATETIME('%b'
    , order_purchase_timestamp) = 'Jan' THEN Order_id END) As January,
COUNT(distinct CASE WHEN FORMAT_DATETIME('%b', order_purchase_timestamp) = 'Feb'THEN
Order_id END) As February,
COUNT(distinct CASE WHEN FORMAT_DATETIME('%b', order_purchase_timestamp) = 'Mar' THEN
Order_id END) As March,
COUNT(distinct CASE WHEN FORMAT_DATETIME('%b', order_purchase_timestamp) = 'Apr' THEN
Order_id END) As April,
COUNT(distinct CASE WHEN FORMAT_DATETIME('%b', order_purchase_timestamp) = 'May' THEN
Order_id END) As May,
COUNT(distinct CASE WHEN FORMAT_DATETIME('%b', order_purchase_timestamp) = 'Jun' THEN
Order_id END) As June,
COUNT(distinct CASE WHEN FORMAT_DATETIME('%b', order_purchase_timestamp) = 'Jul' THEN
Order_id END) As July,
COUNT(distinct CASE WHEN FORMAT_DATETIME('%b', order_purchase_timestamp) = 'Aug' THEN
Order_id END) As August,
COUNT(distinct CASE WHEN FORMAT_DATETIME('%b', order_purchase_timestamp) = 'Sep' THEN
Order_id END) As September,
COUNT(distinct CASE WHEN FORMAT_DATETIME('%b', order_purchase_timestamp) = 'Oct' THEN
Order_id END) As October,
COUNT(distinct CASE WHEN FORMAT_DATETIME('%b', order_purchase_timestamp) = 'Nov' THEN
Order_id END) As November,
COUNT(distinct CASE WHEN FORMAT_DATETIME('%b', order_purchase_timestamp) = 'Dec' THEN
Order_id END) As December
FROM `Target_data.orders` o
INNER JOIN `Target_data.customers` c ON o.customer_id = c.customer_id
GROUP BY c.customer_state;
```

```
select c.customer_state AS state, COUNT(distinct CASE WHEN FORMAT_DATETIME('%b', order_purchase_timestamp) = 'Jan'
THEN Order_id END) As January,
COUNT(distinct CASE WHEN FORMAT_DATETIME('%b', order_purchase_timestamp) = 'Feb'THEN  Order_id END) As February,
```

Press Alt+F1 for accessibility options.

Query results

SAVE RESULTS EXPLORE DATA

JOB INFORMATION RESULTS JSON EXECUTION DETAILS EXECUTION GRAPH PREVIEW

w	state	January	February	March	April	May	June	July
1	RJ	990	1176	1302	1172	1321	1128	
2	RS	427	473	569	488	559	526	
3	SP	3351	3357	4047	3967	4632	4104	
4	DF	151	196	207	183	208	220	
5	PR	443	460	504	500	524	478	
6	MT	96	84	71	92	104	83	
7	MA	66	67	77	73	65	59	
8	AL	39	39	40	51	46	34	

	state	January	February	March	April	May	June	July	August	September	O
8	AL	39	39	40	51	46	34	40	34	20	
9	MG	971	1063	1237	1061	1190	1080	1111	1177	511	
10	PE	113	146	153	154	174	140	210	170	76	
11	SE	24	27	43	27	19	37	42	43	16	
12	PA	82	83	109	107	75	92	96	104	41	
13	BA	264	273	340	318	368	307	405	323	170	
14	CE	99	101	126	143	136	121	140	130	77	

1.
```
SELECT c.customer_state , COUNT(distinct c.customer_id) as customer_count FR
OM `Target_data.customers` c GROUP BY c.customer_state
```

```
select g.geolocation_state AS state, COUNT(distinct c.customer_id) as customer_count FROM `Target_
INNER JOIN `Target_data.geolocation` g on c.customer_zip_code_prefix = g.geolocation_zip_code_pref
GROUP BY g.geolocation_state;
```

Press Al

Query results

SAVE RESULTS EXPLORE

JOB INFORMATION RESULTS JSON EXECUTION DETAILS EXECUTION GRAPH PREVIEW

Row	state	customer_count
1	SE	349
2	AL	412
3	PI	492
4	AP	68
5	AM	148
6	RR	46
7	AC	120
8	RO	256

8	RJ	12852
9	GO	2020
10	MA	747
11	PE	1652
12	PB	536
13	ES	2033
14	PR	5045

A4.

1.
```
SELECT a.Year , ROUND((a.payments/SUM(a.payments) OVER())*100,2) as Percentage_
Diff FROM (select EXTRACT(year from o.order_purchase_timestamp) AS Year, SUM(p.
payment_value) AS payments
FROM `Target_data.payments` p INNER JOIN `Target_data.orders` o ON p.order_id =
 o.order_id
WHERE FORMAT_DATETIME('%b',o.order_purchase_timestamp) IN ('Jan','Feb','Mar','A
pr','May','Jun','Jul','Aug')
AND EXTRACT(year from o.order_purchase_timestamp) IN (2017,2018)
GROUP BY Year
order by Year) a
Order BY a.Year;
```

Row	Year	Percentage_Diff
1	2017	29.68
2	2018	70.32

Percentage Increase in value from 2017 to 2018 is (70.32-29.68) a 40.64 positive growth

2.
```
SELECT c.customer_state,ROUND(AVG(price),2) as Avg_price,ROUND(SUM(price),2) as
 sum_price,ROUND(AVG(freight_value),2) AS avg_fright,ROUND(SUM(freight_value),2
)as sum_fright
FROM `Target_data.order_items` oi
INNER JOIN `Target_data.orders` o
ON o.order_id = oi.order_id
INNER JOIN `Target_data.customers` c
ON c.customer_id = o.customer_id
GROUP BY c.customer_state;
```

```
SELECT c.customer_state,ROUND(AVG(price),2) as Avg_price,ROUND(SUM(price),2) as sum_price,ROUND(AVG(freight_value),
2) AS avg_fright,ROUND(SUM(freight_value),2) as sum_fright
FROM `Target_data.order_items` oi
INNER JOIN `Target_data.orders` o
ON o.order_id = oi.order_id
INNER JOIN `Target_data.customers` c
ON c.customer_id = o.customer_id
```

Press Alt+F1 for accessibility options.

Query results

SAVE RESULTS EXPLORE DATA

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

Row	customer_state	Avg_price	sum_price	avg_fright	sum_fright
1	SP	109.65	5202955.05	15.15	718723.07
2	RJ	125.12	1824092.67	20.96	305589.31
3	PR	119.0	683083.76	20.53	117851.68
4	SC	124.65	520553.34	21.47	89660.26
5	DF	125.77	302603.94	21.04	50625.5
5	DF	125.77	302603.94	21.04	50625.5
6	MG	120.75	1585308.03	20.63	270853.46
7	PA	165.69	178947.81	35.83	38699.3
8	BA	134.6	511349.99	26.36	100156.68
9	GO	126.27	294591.95	22.77	53114.98
10	RS	120.34	750304.02	21.74	135522.74
11	TO	157.53	49621.74	37.25	11732.68

A5.

1.

```
SELECT IFNULL(ABS(DATE_DIFF(o.order_purchase_timestamp,o.order_estimated_
delivery_date,DAY)),0) as Difference_between_purchase_and_EstimatedDelivery,

IFNULL(ABS(DATE_DIFF(o.order_estimated_delivery_date,o.order_delivered_customer
_date,DAY)),0) as
Difference_between_EstimatedDelivery_and_delivered
FROM `Target_data.orders` o
```

```
SELECT IFNULL(ABS(DATE_DIFF(o.order_purchase_timestamp,o.order_estimated_delivery_date,DAY)),0) as
Difference_between_purchase_and_EstimatedDelivery,
IFNULL(ABS(DATE_DIFF(o.order_estimated_delivery_date,o.order_delivered_customer_date,DAY)),0) as
Difference_between_EstimatedDelivery_and_delivered
FROM `Target_data.orders` o
```

Press Alt+I

Query results

SAVE RESULTS EXPLORE D

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

Row	Difference_between_purchase_and_EstimatedDelivery	Difference_between_EstimatedDelivery_and_delivered
1	50	0
2	6	0
3	44	0
4	54	0
5	56	0
6	54	0
7	56	0
8	41	0
9	3	0
10	3	0
11	47	0
12	44	0
13	43	0

2.

```
SELECT IFNULL(DATE_DIFF(o.order_delivered_customer_date,o.order_purchase_timestamp,DAY
),0) as Time_to_delivery,
 IFNULL(DATE_DIFF(o.order_delivered_customer_date,o.order_estimated_delivery_date,DAY)
,0) as diff_estimated_delivery
FROM `Target_data.orders` o
```

```
SELECT IFNULL(DATE_DIFF(o.order_delivered_customer_date,o.order_purchase_timestamp,DAY),0) as Time_to_delivery,
 IFNULL(DATE_DIFF(o.order_delivered_customer_date,o.order_estimated_delivery_date,DAY),0) as diff_estimated_delivery
FROM `Target_data.orders` o
```

Press Alt+F1 for accessibility optio

Query results

SAVE RESULTS EXPLORE DATA

JOB INFORMATION RESULTS JSON EXECUTION DETAILS EXECUTION GRAPH PREVIEW

Row	Time_to_delivery	diff_estimated_delivery
1	30	12
2	30	-28
3	35	-16
4	30	-1
5	32	0
6	29	-1

Row	Time_to_delivery	diff_estimated_delivery
6	29	-1
7	43	4
8	40	4
9	37	1
10	33	5
11	38	6
12	36	2
13	34	0

3.

```
SELECT c.customer_state,ROUND(AVG(freight_value),2) as Mean_of_frieghtValue,ROUND(AVG(
DATE_DIFF(o.order_delivered_customer_date,o.order_purchase_timestamp,DAY)),2) as Time_
to_delivery,
ROUND(AVG(DATE_DIFF(o.order_delivered_customer_date,o.order_estimated_delivery_date,DA
Y)),2) as diff_estimated_delivery
FROM `Target_data.order_items` oi
INNER JOIN `Target_data.orders` o
ON o.order_id = oi.order_id
INNER JOIN `Target_data.customers` c
ON c.customer_id = o.customer_id
GROUP BY c.customer_state;
```

```
SELECT c.customer_state,ROUND(AVG(freight_value),2) as Mean_of_frieghtValue,ROUND(AVG(DATE_DIFF(o.order_delivered_customer_date,o.order_purchase_timestamp,DAY)),2) as Time_to_delivery,
ROUND(AVG(DATE_DIFF(o.order_delivered_customer_date,o.order_estimated_delivery_date,DAY)),2) as diff_estimated_delivery
FROM `Target_data.order_items` oi
INNER JOIN `Target_data.orders` o
ON o.order_id = oi.order_id
INNER JOIN `Target_data.customers` c
```

Press Alt+F

Query results

SAVE RESULTS EXPLORE DA

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

Row	customer_state	Mean_of_frieght	Time_to_delivery	diff_estimated_d
1	MT	28.17	17.51	-13.64
2	MA	38.26	21.2	-9.11
3	AL	35.84	23.99	-7.98
4	SP	15.15	8.26	-10.27
5	MG	20.63	11.52	-12.4
6	DF	22.02	17.70	-12.55

Row	customer_state	Mean_of_frieght	Time_to_delivery	diff_estimated_d
5	DF	21.04	12.5	-11.27
6	MG	20.63	11.52	-12.4
7	PA	35.83	23.3	-13.37
8	BA	26.36	18.77	-10.12
9	GO	22.77	14.95	-11.37
10	RS	21.74	14.71	-13.2
11	TO	37.25	17.0	-11.46
12	AM	33.21	25.96	-18.98

4.

1. Highest average freight value by state

```
SELECT c.customer_state,ROUND(AVG(freight_value),2) as Mean_of_frieghtValue

FROM `Target_data.order_items` oi

INNER JOIN `Target_data.orders` o
ON o.order_id = oi.order_id
INNER JOIN `Target_data.customers` c
ON c.customer_id = o.customer_id
GROUP BY c.customer_state
Order By Mean_of_frieghtValue Desc Limit 5;
```

```
SELECT c.customer_state,ROUND(AVG(freight_value),2) as Mean_of_frieghtValue
FROM `Target_data.order_items` oi
INNER JOIN `Target_data.orders` o
ON o.order_id = oi.order_id
INNER JOIN `Target_data.customers` c
ON c.customer_id = o.customer_id
GROUP BY c.customer_state
```

Query results

SAVE RESU

JOB INFORMATION | RESULTS | JSON | EXECUTION DETAILS | EXECUTION

ow	customer_state	Mean_of_frieght
1	RR	42.98
2	PB	42.72
3	RO	41.07
4	AC	40.07
5	PI	39.15

Top 5 lowest average freight value by state:-

```
SELECT c.customer_state,ROUND(AVG(freight_value),2) as Mean_of_frieghtValue
FROM `Target_data.order_items` oi
INNER JOIN `Target_data.orders` o
ON o.order_id = oi.order_id
INNER JOIN `Target_data.customers` c
ON c.customer_id = o.customer_id
GROUP BY c.customer_state
Order By Mean_of_frieghtValue ASC Limit 5;
```

2.

Top 5 states with highest/lowest average time to delivery

Top 5 states with highest delivery:-

```
SELECT c.customer_state,ROUND(AVG(DATE_DIFF(o.order_delivered_customer_date,o.o
rder_purchase_timestamp, day)),2) as AVG_time_to_delivery
FROM `Target_data.orders` o
INNER JOIN `Target_data.customers` c
ON c.customer_id = o.customer_id
GROUP BY c.customer_state
Order By AVG_time_to_delivery DESC Limit 5;
```

```
AVG_time_to_delivery
FROM `Target_data.orders` o
INNER JOIN `Target_data.customers` c
ON c.customer_id = o.customer_id
GROUP BY c.customer_state
Order By AVG_time_to_delivery DESC Limit 5;
```

Query results

JOB INFORMATION | RESULTS | JSON | EXECUTION DETAILS

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

Top 5 lowest average time to delivery:-

```
SELECT c.customer_state,ROUND(AVG(DATE_DIFF(o.order_delivered_customer_date,o.o
rder_purchase_timestamp, day)),2) as AVG_time_to_delivery
FROM `Target_data.orders` o
INNER JOIN `Target_data.customers` c
ON c.customer_id = o.customer_id
GROUP BY c.customer_state
Order By AVG_time_to_delivery ASC Limit 5;
```

JOB INFORMATION | RESULTS | JSON | EXECUTION DETAILS | EXECUTION

ow	customer_state	AVG_time_to_de
1	SP	8.3
2	PR	11.53
3	MG	11.54
4	DF	12.51
5	SC	14.48

PERSONAL HISTORY | PROJECT HISTORY

3.

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

Top 5 states where delivery is really fast compared to estimated date.

```
SELECT c.customer_state,ROUND(AVG(DATE_DIFF(o.order_delivered_customer_date,o.o
rder_estimated_delivery_date, day)),2) as AVG_time_to_delivery
FROM `Target_data.orders` o
INNER JOIN `Target_data.customers` c
ON c.customer_id = o.customer_id
GROUP BY c.customer_state
Order By AVG_time_to_delivery ASC Limit 5;
```

JOB INFORMATION | RESULTS | JSON | EXECUTION DETA

Row	customer_state	AVG_time_to_de
1	AC	-19.76
2	RO	-19.13
3	AP	-18.73
4	AM	-18.61
5	RR	-16.41

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

```
SELECT c.customer_state,ROUND(AVG(DATE_DIFF(o.order_delivered_customer_date,o.o
rder_estimated_delivery_date, day)),2) as AVG_time_to_delivery
FROM `Target_data.orders` o
INNER JOIN `Target_data.customers` c
ON c.customer_id = o.customer_id
```

```
GROUP BY c.customer_state
Order By AVG_time_to_delivery DESC Limit 5;
```

Row	customer_state	AVG_time_to_de
1	AL	-7.95
2	MA	-8.77
3	SE	-9.17
4	ES	-9.62
5	BA	-9.93

A6.

1. Month over Month count of orders for different payment types

```
select p.payment_type AS payment_type, COUNT(distinct CASE WHEN FORMAT_DATETIME
('%b', order_purchase_timestamp) = 'Jan' THEN p.Order_id END) As January,

COUNT(distinct CASE WHEN FORMAT_DATETIME('%b', order_purchase_timestamp) = 'Feb
'THEN  p.Order_id END) As February,
COUNT(distinct CASE WHEN FORMAT_DATETIME('%b', order_purchase_timestamp) = 'Mar
' THEN p.Order_id END) As March,
COUNT(distinct CASE WHEN FORMAT_DATETIME('%b', order_purchase_timestamp) = 'Apr
' THEN p.Order_id END) As April,
COUNT(distinct CASE WHEN FORMAT_DATETIME('%b', order_purchase_timestamp) = 'May
' THEN p.Order_id END) As May,
COUNT(distinct CASE WHEN FORMAT_DATETIME('%b', order_purchase_timestamp) = 'Jun
' THEN p.Order_id END) As June,
COUNT(distinct CASE WHEN FORMAT_DATETIME('%b', order_purchase_timestamp) = 'Jul
' THEN p.Order_id END) As July,
COUNT(distinct CASE WHEN FORMAT_DATETIME('%b', order_purchase_timestamp) = 'Aug
' THEN p.Order_id END) As August,
COUNT(distinct CASE WHEN FORMAT_DATETIME('%b', order_purchase_timestamp) = 'Sep
' THEN p.Order_id END) As September,
COUNT(distinct CASE WHEN FORMAT_DATETIME('%b', order_purchase_timestamp) = 'Oct
' THEN p.Order_id END) As October,
COUNT(distinct CASE WHEN FORMAT_DATETIME('%b', order_purchase_timestamp) = 'Nov
' THEN p.Order_id END) As November,
COUNT(distinct CASE WHEN FORMAT_DATETIME('%b', order_purchase_timestamp) = 'Dec
' THEN p.Order_id END) As December
FROM `Target_data.payments` p
INNER JOIN `Target_data.orders` o
ON p.order_id = o.order_id
INNER JOIN `Target_data.customers` c
ON o.customer_id = c.customer_id
GROUP BY p.payment_type;
```

Row	payment_type	January	February	March	April	May	June
1	credit_card	6093	6582	7682	7276	8308	7248
2	voucher	337	288	395	353	374	373
3	not_defined	0	0	0	0	0	0
4	debit_card	118	82	109	124	81	208
5	UPI	1715	1723	1942	1783	2035	1807

2. Count of orders based on the no. of payment installments

```
select p.payment_installments AS payment_installments,COUNT(distinct p.order_i)
AS Order_count

FROM `Target_data.payments` p
INNER JOIN `Target_data.orders` o
ON p.order_id = o.order_id
INNER JOIN `Target_data.customers` c
ON o.customer_id = c.customer_id
GROUP BY p.payment_installments ORDER BY p.payment_installments;
```

Row	payment_installr	Order_count
1	0	2
2	1	49060
3	2	12389
4	3	10443
5	4	7088
6	5	5234
7	6	3916
8	7	1623
9	8	4253
10	9	644
11	10	5315
12	11	23
13	12	133