Business Case: Target

PART 1 -

Time period for which the data is given-

```
select min(data_period) as first_day,max(data_period) as last_day
from
(select distinct extract(date from order_purchase_timestamp) as data_period from `jan23-scalersql.Target.orders`
order by data_period);
```

Query results

JOB INFORMATION | RESULTS | JSON | EXECUTION DET

Row	first_day	last_day
1	2016-09-04	2018-10-17

Cities and States of customers ordered during the given period-

```
select c.customer_id,c.customer_city,c.customer_state,extract(date from o.order_purchase_timestamp) as order
_date
from `jan23-scalersql.Target.customers` as c
join `jan23-scalersql.Target.orders` as o
on c.customer_id=o.customer_id
order by order_date;
```

Query results

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

Row	customer_id	customer_city	customer_state	order_date
1	08c5351a6aca1c1589a38f244...	boa vista	RR	2016-09-04
2	683c54fc24d40ee9f8a6fc179f...	passo fundo	RS	2016-09-05
3	622e13439d6b5a0b486c4356...	sao jose dos campos	SP	2016-09-13
4	86dc2ffce2dfff336de2f386a78...	sao joaquim da barra	SP	2016-09-15
5	b106b360fe2ef8849fbbd056f7...	sao paulo	SP	2016-10-02
6	7ec40b22510fdbea1b08921dd...	panambi	RS	2016-10-03
7	7812fcebfc5e8065d31e1bb5f0...	taubate	SP	2016-10-03
8	dc607dc98d6a11d5d04d9f2a7...	ipatinga	MG	2016-10-03
9	355077684019f7f60a031656b...	sao paulo	SP	2016-10-03
10	b8cf418e97ae795672d326288...	hortolandia	SP	2016-10-03

PART 2 –

Is there a growing trend on e-commerce in Brazil?

```
select count(order_id) number_of_orders_per_year,purchase_year from
(select order_id, extract(year from order_purchase_timestamp) as purchase_year from `jan23-
scalersql.Target.orders`
where order_status = "delivered")
group by purchase_year
order by purchase_year;
```

Query results

JOB INFORMATION | RESULTS | JSON

Row	number_of_orders_per_year	purchase_year
1	267	2016
2	43428	2017
3	52783	2018

Yes, there is a growing trend on e-commerce in Brazil. On doing year by year analysis of 'orders successfully delivered' it can be seen that there is a significant growth of orders from 2016 into 2017 and while maintaining the 2017 orders there is again growth in 2018. Yes, some seasonality peak is also being noticed between months November to March, mainly in Winter season.

Can we see some seasonality with peaks at specific months?

```
select count(order_id) number_of_orders_per_month,purchase_months,purchase_year from
(select order_id, extract(MONTH from order_purchase_timestamp) as purchase_months, extract(year from order
_purchase_timestamp) as purchase_year from `jan23-scalersql.Target.orders`
where order_status = "delivered")
group by purchase_months,purchase_year
order by purchase_year,purchase_months;
```

Query results

JOB INFORMATION | RESULTS | JSON | EXECUTIO

Row	number_of_orders	purchase_months	purchase_year
11	4193	8	2017
12	4150	9	2017
13	4478	10	2017
14	7289	11	2017
15	5513	12	2017
16	7069	1	2018
17	6555	2	2018
18	7003	3	2018
19	6798	4	2018
20	6749	5	2018

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

```
select count(order_id) number_of_orders,purchase_hour
from
(select order_id, extract(hour from order_purchase_timestamp) as purchase_hour from `jan23-scalersql.Target.orders`)
group by purchase_hour
order by purchase_hour;
```

Query results

JOB INFORMATION | RESULTS | J

Row	number_of_orders	purchase_hour
8	1231	7
9	2967	8
10	4785	9
11	6177	10
12	6578	11
13	5995	12
14	6518	13
15	6569	14
16	6454	15
17	6675	16
18	6150	17
19	5769	18

Query results

JOB INFORMATION | RESULTS | J

Row	number_of_orders	purchase_hour
13	5995	12
14	6518	13
15	6569	14
16	6454	15
17	6675	16
18	6150	17
19	5769	18
20	5982	19
21	6193	20
22	6217	21
23	5816	22
24	4123	23

There is not a lot to separate from the number of orders placed in the working hours (i.e 0900 hrs to 2200 hrs). Although there is a big rise in orders from around 0800-0900 hrs in the morning till 1600 hrs in the afternoon then there is a little decline but it again picks up at 1900 hrs and continues

growing till 2200 hrs in the night. Therefore, dawn and mid night are the times when Brazilians' buy least.

PART 3 –

Get month on month orders by states

```
select count(t1.order_id) month_on_month_orders,t1.purchase_month,t1.customer_state
from
(select o.order_id,o.customer_id,c.customer_state, extract(month from o.order_purchase_timestamp) as purchas
e_month
from Target.orders as o
join Target.customers as c
on o.customer_id=c.customer_id
order by purchase_month) as t1
group by t1.purchase_month,t1.customer_state
order by t1.purchase_month;
```

Query results

JOB INFORMATION | RESULTS | JSON | EXECUTION DET

Row	month_on_mont	purchase_month	customer_state
1	8	1	AC
2	39	1	AL
3	12	1	AM
4	11	1	AP
5	264	1	BA
6	99	1	CE
7	151	1	DF
8	159	1	ES
9	164	1	GO
10	66	1	MA

Query results

JOB INFORMATION | RESULTS | JSON | EXECUTION DET

Row	month_on_mont	purchase_month	customer_state
26	3351	1	SP
27	19	1	TO
28	6	2	AC
29	39	2	AL
30	16	2	AM
31	4	2	AP
32	273	2	BA
33	101	2	CE
34	196	2	DF
35	186	2	ES

Distribution of customers across the states in Brazil

```
select count(customer_id) as customer_per_state,customer_state
from Target.customers
group by customer_state
order by customer_state;
```

Query results

JOB INFORMATION | RESULTS | JSON | EXE

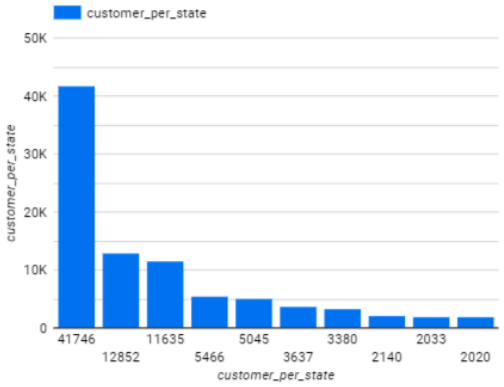
Row	customer_per_state	customer_state
1	81	AC
2	413	AL
3	148	AM
4	68	AP
5	3380	BA
6	1336	CE
7	2140	DF
8	2033	ES
9	2020	GO
10	747	MA

The distribution of customers across Brazil states-wise is pretty uneven with top 3 states being SP, RJ & MG with 41746,12852 & 11635 number of customers respectively and lowest 3 states being RR, AP & AC with 46,61 & 81 no of customers respectively.

Customers per state

	customer_state	customer_per_state
1.	SP	41,746
2.	RJ	12,852
3.	MG	11,635
4.	RS	5,466
5.	PR	5,045
6.	SC	3,637
7.	BA	3,380
8.	DF	2,140
9.	ES	2,033
10.	GO	2,020
11.	PE	1,652

1 - 27 / 27



PART 4 –

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

```
select *, round(((cost_2018-cost_2017)/(cost_2017))*100) as percent_increase
from
(select round(sum(case when purchase_year=2018 and purchase_month between 1 and 8 then payment_value
end)) as cost_2018,
round(sum(case when purchase_year=2017 and purchase_month between 1 and 8 then payment_value end)) a
s cost_2017
from
(select extract(month from o.order_purchase_timestamp) as purchase_month,extract(year from order_purchase_
timestamp) as purchase_year,p.payment_value
from `jan23-scalersql.Target.orders` as o
join Target.payments as p
on o.order_id = p.order_id
order by purchase_year,purchase_month) as t1);
```

Query results

JOB INFORMATION | RESULTS | JSON | EXE

Row	cost_2018	cost_2017	percent_increase
1	8694734.0	3669022.0	137.0

Money movement through e-commerce has increased significantly and precisely by 137% from 2017 in 2018.

Mean & Sum of price and freight value by customer state

```
select round((sum(price)/count(price))) as price_mean,
round((sum(freight_value)/count(freight_value))) as freight_mean,round(sum(price)) as price_sum,round(sum(freight_value)) as freight_sum,customer_state
from
 (select oi.price,oi.freight_value,c.customer_state
 from `jan23-scalersql.Target.order_items`  as oi
 left join Target.orders as o
 on oi.order_id = o.order_id
 left join `jan23-scalersql.Target.customers` as c
 on o.customer_id = c.customer_id
 order by c.customer_state) t1
group by t1.customer_state
order by t1.customer_state;
```

Query results

JOB INFORMATION | RESULTS | JSON | EXECUTION DETAILS | EXECU

Row	price_mean	freight_mean	price_sum	freight_sum	customer_state
1	174.0	40.0	15983.0	3687.0	AC
2	181.0	36.0	80315.0	15915.0	AL
3	135.0	33.0	22357.0	5479.0	AM
4	164.0	34.0	13474.0	2789.0	AP
5	135.0	26.0	511350.0	100157.0	BA
6	154.0	33.0	227255.0	48352.0	CE
7	126.0	21.0	302604.0	50625.0	DF
8	122.0	22.0	275037.0	49765.0	ES
9	126.0	23.0	294592.0	53115.0	GO
10	145.0	38.0	119648.0	31524.0	MA

The avg. price of product across different states ranges from 110 to 191 while the avg. freight value varies from 15 to 43.

PART 5 –

Calculate days between purchasing, delivering and estimated delivery

```
SELECT
  t1.order_id,
  DATE_DIFF(delivery_date,purchase_date,day) AS actual_delivery_days,
  DATE_DIFF(estimated_delivery_date,purchase_date,day) AS estimated_delivery_days,
  DATE_DIFF(estimated_delivery_date,delivery_date,day) AS days_betw_actual_estimated
FROM (
  SELECT
    order_id,
    EXTRACT(date
    FROM
```

```
    order_purchase_timestamp) AS purchase_date,
  EXTRACT(date
  FROM
    order_delivered_customer_date) AS delivery_date,
  EXTRACT(date
  FROM
    order_estimated_delivery_date) AS estimated_delivery_date
FROM
  Target.orders
ORDER BY
  purchase_date) t1
WHERE
  purchase_date IS NOT NULL
  AND delivery_date IS NOT NULL
  AND estimated_delivery_date IS NOT NULL;
```

Query results

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

Row	order_id	actual_delivery_days	estimated_delivery_days	days_betw_actual_estimated
1	bfbd0f9bdef84302105ad712db...	55	19	-36
2	65d1e226dfaeb8cdc42f66542...	36	53	17
3	be5bc2f0da14d8071e2d45451...	24	35	11
4	ae8a60e4b03c5a4ba9ca0672c...	31	59	28
5	cd3b8574c82b42fc8129f6d50...	11	51	40
6	d207cc272675637bfed0062ed...	28	51	23
7	a41c8759fbe7aab36ea07e038...	31	57	26
8	ef1b29b591d31d57c0d733746...	29	53	24
9	3b697a20d9e427646d925679...	23	24	1
10	35d3a51724a47ef1d0b89911e...	22	77	55

Usually, all the orders are delivered before the estimated delivery date. There are negligible number of times when delivery has happened after the estimated date.

Find time_to_delivery & diff_estimated_delivery. Formula for the same given below:

time_to_delivery = order_purchase_timestamp-order_delivered_customer_date

diff_estimated_delivery = order_estimated_delivery_date-order_delivered_customer_date

```
SELECT
  DATE_DIFF(delivery_date,purchase_date,day) AS time_to_delivery,
  DATE_DIFF(estimated_delivery_date,delivery_date,day) AS diff_estimated_delivery
FROM (
  SELECT
    EXTRACT(date
    FROM
      order_purchase_timestamp) AS purchase_date,
    EXTRACT(date
    FROM
      order_delivered_customer_date) AS delivery_date,
    EXTRACT(date
    FROM
      order_estimated_delivery_date) AS estimated_delivery_date
```

```
  FROM
    Target.orders
  ORDER BY
    purchase_date)
WHERE
  purchase_date IS NOT NULL
  AND delivery_date IS NOT NULL
  AND estimated_delivery_date IS NOT NULL;
```

Query results

JOB INFORMATION | RESULTS | JSON

Row	time_to_delivery	diff_estimated_delivery
1	55	-36
2	36	17
3	24	11
4	31	28
5	11	40
6	28	23
7	31	26
8	29	24
9	23	1
10	22	55

Group data by state, take mean of freight_value, time_to_delivery, diff_estimated_delivery

```
SELECT
  t1.customer_state,
  ROUND(AVG(freight_value)) AS mean_freight,
  ROUND(AVG(DATE_DIFF(t1.delivery_date,purchase_date,day))) AS avg_time_to_delivery,
  ROUND(AVG(DATE_DIFF(estimated_delivery_date,delivery_date,day))) AS avg_diff_estimated_delivery
FROM (
  SELECT
    customer_state,
    freight_value,
    EXTRACT(date
    FROM
      order_purchase_timestamp) AS purchase_date,
    EXTRACT(date
    FROM
      order_delivered_customer_date) AS delivery_date,
    EXTRACT(date
    FROM
      order_estimated_delivery_date) AS estimated_delivery_date
  FROM
    Target.order_items AS oi
  JOIN
    Target.orders AS o
  ON
    oi.order_id = o.order_id
  JOIN
    Target.customers AS c
```

```
  ON
    o.customer_id=c.customer_id) AS t1
GROUP BY t1.customer_state;
```

Query results

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

Row	customer_state	mean_freight	avg_time_to_delivery	avg_diff_estimated_delivery
1	MT	28.0	18.0	15.0
2	MA	38.0	22.0	10.0
3	AL	36.0	24.0	9.0
4	SP	15.0	9.0	11.0
5	MG	21.0	12.0	13.0
6	PE	33.0	18.0	13.0
7	RJ	21.0	15.0	12.0
8	DF	21.0	13.0	12.0
9	RS	22.0	15.0	14.0
10	SE	37.0	21.0	10.0

Average time taken for the delivery of orders ranges from 9 to 28 days across different states in the country. And as observed in the 1st part of this question most orders are delivered before the estimated delivery date, the average difference between estimated and actual delivery date is positive and ranges from 9 to 21 days which means the orders are delivered between 9 to 21 days before the estimated date across states.

Sort the data to get the following:

Making a view named state_freights of the above code to perform sorting and limit.

```
create view Target.state_freights as
SELECT
  t1.customer_state,
  ROUND(AVG(freight_value)) AS mean_freight,
  ROUND(AVG(DATE_DIFF(t1.delivery_date,purchase_date,day))) AS avg_time_to_delivery,
  ROUND(AVG(DATE_DIFF(estimated_delivery_date,delivery_date,day))) AS avg_diff_estimated_delivery
FROM (
  SELECT
    customer_state,
    freight_value,
    EXTRACT(date FROM order_purchase_timestamp) AS purchase_date,
    EXTRACT(date FROM order_delivered_customer_date) AS delivery_date,
    EXTRACT(date FROM order_estimated_delivery_date) AS estimated_delivery_date
  FROM
    Target.order_items AS oi
  JOIN
    Target.orders AS o
  ON oi.order_id = o.order_id

  JOIN
    Target.customers AS c
  ON
    o.customer_id=c.customer_id) AS t1
```

```
GROUP BY
  t1.customer_state;
```

Now performing the required sorting.

- Top 5 states with highest/lowest average freight value - sort in desc/asc limit 5

```
select * from Target.state_freights
order by mean_freight
limit 5;
```

Query results

JOB INFORMATION | RESULTS | JSON | EXECUTION DETAILS | EXECUTIO

Row	customer_state	mean_freight	avg_time_to_delivery	avg_diff_estimated_delivery
1	SP	15.0	9.0	11.0
2	DF	21.0	13.0	12.0
3	RJ	21.0	15.0	12.0
4	SC	21.0	15.0	12.0
5	PR	21.0	12.0	13.0

Top 5 states with lowest average freight value.

```
select * from Target.state_freights
order by mean_freight desc
limit 5;
```

Query results

JOB INFORMATION | RESULTS | JSON | EXECUTION DETAILS | EXE

Row	customer_state	mean_freight	avg_time_to_deli	avg_diff_estimat
1	PB	43.0	21.0	13.0
2	RR	43.0	28.0	18.0
3	RO	41.0	20.0	20.0
4	AC	40.0	21.0	21.0
5	PI	39.0	19.0	12.0

Top 5 states with highest average freight value.

- Top 5 states with highest/lowest average time to delivery.

```
select * from Target.state_freights
order by avg_time_to_delivery asc
limit 5;
```

Query results

JOB INFORMATION | RESULTS | JSON | EXECUTION DETAILS | EXECUTIO

Row	customer_state	mean_freight	avg_time_to_delivery	avg_diff_estimated_delivery
1	SP	15.0	9.0	11.0
2	MG	21.0	12.0	13.0
3	PR	21.0	12.0	13.0
4	DF	21.0	13.0	12.0
5	SC	21.0	15.0	12.0

Top 5 states with lowest average time to delivery

```
select * from Target.state_freights
order by avg_time_to_delivery desc
limit 5;
```

Query results

JOB INFORMATION | RESULTS | JSON | EXECUTION DETAILS | EXECUTIC

Row	customer_state	mean_freight	avg_time_to_delivery	avg_diff_estimated_delivery
1	AP	34.0	28.0	18.0
2	RR	43.0	28.0	18.0
3	AM	33.0	26.0	20.0
4	AL	36.0	24.0	9.0
5	PA	36.0	24.0	14.0

Top 5 states with highest average time to delivery.

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

```
select * from Target.state_freights
order by avg_diff_estimated_delivery desc
limit 5;
```

Query results

JOB INFORMATION | RESULTS | JSON | EXECUTION DETAILS | EXECUTIO

Row	customer_state	mean_freight	avg_time_to_delivery	avg_diff_estimated_delivery
1	AC	40.0	21.0	21.0
2	AM	33.0	26.0	20.0
3	RO	41.0	20.0	20.0
4	RR	43.0	28.0	18.0
5	AP	34.0	28.0	18.0

Top 5 states with fastest delivery.

```
select * from Target.state_freights
order by avg_diff_estimated_delivery
limit 5;
```

Query results

JOB INFORMATION | RESULTS | JSON | EXECUTION DETAILS | EXECUTION GR

Row	customer_state	mean_freight	avg_time_to_delivery	avg_diff_estimated_delivery
1	AL	36.0	24.0	9.0
2	SE	37.0	21.0	10.0
3	MA	38.0	22.0	10.0
4	SP	15.0	9.0	11.0
5	BA	26.0	19.0	11.0

Top 5 states with slowest delivery

PART 6 –

Month over Month count of orders for different payment types

```
select Months,count(payment_type) no_of_orders,payment_type
from
(select extract(month from order_purchase_timestamp) as Months, payment_type
from Target.orders as o
join Target.payments as p on o.order_id=p.order_id)
group by Months,payment_type
order by Months;
```

Query results

JOB INFORMATION | RESULTS | JSON | EXECUTION DET

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

The most used payment method is credit card then UPIs then vouchers and the least used is debit cards.

Count of orders based on the no. of payment installments

```
select count(order_id) as no_of_orders,payment_installments
from `jan23-scalersql.Target.payments`
group by payment_installments;
```

Query results

JOB INFORMATION | RESULTS | JS

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