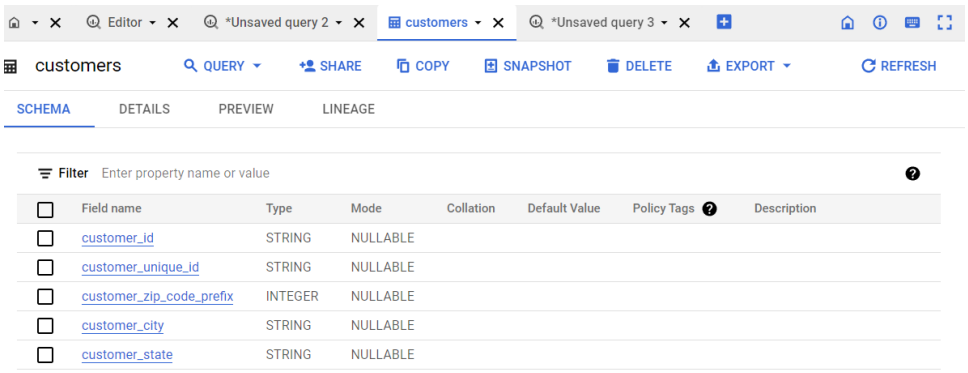
Business Case: Target SQL

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

(a) Data type of columns in a table



(b) Time period for which the data is given

```
SELECT
max(date(order_purchase_timestamp)) as First_order_timestamp,
min(date(order_purchase_timestamp)) as Last_order_timestamp,


max(date(order_delivered_customer_date)) as First_delivered_timestamp,
min(date(order_delivered_customer_date)) as Last_delivered_timestamp,


date_diff(max(date(order_purchase_timestamp)), min(date(order_purchase_timestamp)),
day) as order_recorded_span_in_days


 FROM `my-new-project-scaler-dsml.Scaler_project1.orders`
```

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

Row	First_order_timestamp	Last_order_timestamp	First_delivered_timestamp	Last_delivered_timestamp	order_recorded_span_in_days
1	2018-10-17	2016-09-04	2018-10-17	2016-10-11	773

(c) Cities and States of customers ordered during the given period

```
select distinct c.customer_city, c.customer_state
FROM `my-new-project-scaler-dsml.Scaler_project1.customers` c join
`my-new-project-scaler-dsml.Scaler_project1.geolocation` g
on c.customer_zip_code_prefix=g.geolocation_zip_code_prefix limit 100;
```

(2) In-depth Exploration:

(a) 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?

```
select extract(month from o.order_purchase_timestamp) as Month,
count(o.order_id) as no_of_orders
FROM `my-new-project-scaler-dsml.Scaler_project1.orders` o
group by Month order by no_of_orders desc ;
```

```
select extract(year from o.order_purchase_timestamp) as year,
count(o.order_id) as no_of_orders
FROM `my-new-project-scaler-dsml.Scaler_project1.orders` o
group by year order by no_of_orders desc ;
```

We have data from 4th September, 2016 till 17th Oct, 2018.

So excluding 2016 as we don't have much data on it, there is a significant amount of growth in order in 2018 (till oct) compared to whole year of 2017.

Also overall the month of August has the maximum under of orders.

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

```
select
CASE
when extract(hour from o.order_purchase_timestamp) in (4,5,6,7,8)
then "Dawn"
when extract(hour from o.order_purchase_timestamp) in (9,10,11,12,13)
then "Morning"
when extract(hour from o.order_purchase_timestamp) in (14,15,16,17,18)
then "Afternoon"
else "Night"
end as purchase_time,
count(o.order_id) as no_of_orders
FROM
`my-new-project-scaler-dsml.Scaler_project1.orders` o group by purchase_time order by
no_of_orders desc ;
```

JOB INFORMATION | RESULTS | JSON

Row	purchase_time	no_of_orders
1	Night	32677
2	Afternoon	31617
3	Morning	30053
4	Dawn	5094

Approximately 65% of the orders are bought between 2pm till 3am. Purchases are mostly done at night time.

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

(a) Get month on month orders by states

```
select extract(month from o.order_purchase_timestamp) as Month,
c.customer_state,
count(o.order_id) as number_of_orders
FROM `my-new-project-scaler-dsml.Scaler_project1.orders` o
join `my-new-project-scaler-dsml.Scaler_project1.customers` c on
o.customer_id=c.customer_id
group by Month,customer_state order by Month, number_of_orders desc ;
```

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

Row	Month	customer_state	number_of_orde
23	1	TO	19
24	1	AM	12
25	1	AP	11
26	1	AC	8
27	1	RR	2
28	2	SP	3357
29	2	RJ	1176
30	2	MG	1063
31	2	RS	473
32	2	PR	460
33	2	SC	316
34	2	BA	273

Results per page: 50 1 – 50 of 322

```
select extract(month from o.order_purchase_timestamp) as Month,
 c.customer_city,
count(o.order_id) as number_of_orders
FROM `my-new-project-scaler-dsml.Scaler_project1.orders` o
join `my-new-project-scaler-dsml.Scaler_project1.customers` c on
o.customer_id=c.customer_id
group by Month, customer_city order by Month, number_of_orders desc ;
```

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

Row	Month	customer_city	number_of_orders
1481	1	ampere	1
1482	1	sao joao do itaperiu	1
1483	1	sao joao evangelista	1
1484	1	teotonio vilela	1
1485	1	pirapetinga	1
1486	1	aracruz	1
1487	2	sao paulo	1272
1488	2	rio de janeiro	656
1489	2	belo horizonte	218
1490	2	brasilia	196
1491	2	curitiba	137
1492	2	campinas	131
1493	2	porto alegre	112
1494	2	salvador	90
1495	2	sao bernardo do campo	85

Results per page: 200 1401 – 1600 of 17385

Month wise aggregation of total number of orders at city and state level.

(b) Distribution of customers across the states in Brazil

```
select
customer_city,
count(distinct customer_id) as total_no_of_customers
from `my-new-project-scaler-dsml.Scaler_project1.customers`
group by 1 order by total_no_of_customers desc;
```

JOB INFORMATION | RESULTS | JSON | EXE

Row	customer_city	total_no_of_cust
1	sao paulo	15540
2	rio de janeiro	6882
3	belo horizonte	2773
4	brasilia	2131
5	curitiba	1521
6	campinas	1444
7	porto alegre	1379
8	salvador	1245
9	guarulhos	1189
10	sao bernardo do campo	938

```sql
select
customer_state,
count(distinct customer_id) as total_no_of_customers
from `my-new-project-scaler-dsml.Scaler_project1.customers`
group by 1 order by total_no_of_customers desc,customer_state;
```

Row	customer_state	total_no_of_cust
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
11	PE	1652

State and city wise customer distribution.

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

(a) Get % increase in cost of orders from 2017 to 2018 (include months between Jan to Aug only) - You can use “payment_value” column in payments table

```sql
WITH Table_with_total_order_value as (
select sum(payment_value) as total_cost_of_order,
extract(Year from o.order_purchase_timestamp) as Year
from `my-new-project-scaler-dsml.Scaler_project1.orders` o
```

```
inner join `my-new-project-scaler-dsml.Scaler_project1.payments` p on o.order_id =
p.order_id

where Extract(year from o.order_purchase_timestamp) IN (2017,2018)

and Extract(month from o.order_purchase_timestamp) NOT IN (9,10,11,12)

group by Year)
```

```
SELECT

((select total_cost_of_order from Table_with_total_order_value where Year = 2018)

/(select total_cost_of_order from Table_with_total_order_value where Year = 2017)) *
100 as Percentage_increment
```

JOB INFORMATION | RESULTS | JSON | EXECUTION DETAILS

Row	Percentage_increment
1	236.97687164666226

Cost of orders has been increased by 236% from 2017 to 2018 (including months between Jan to Aug only)

(b) Mean & Sum of price and freight value by customer state

```
select x.customer_state,

sum(item.freight_value) as total_freight_value,

avg(item.freight_value) as avg_freight_value,

sum(item.price) as total_price,

avg(item.price) as avg_price

from

(

select o.order_id,c.customer_state

from `my-new-project-scaler-dsml.Scaler_project1.orders` o

join `my-new-project-scaler-dsml.Scaler_project1.customers` c on

o.customer_id=c.customer_id
```

```
) x join
`my-new-project-scaler-dsml.Scaler_project1.order_items` item on
x.order_id=item.order_id
group by customer_state
order by total_freight_value, total_price DESC
```

9	RN	18860.100000000...	35.65236294896...	83034.98000000001	156.96593572...
10	MS	19144.030000000...	23.37488400488...	116812.63999999974	142.62837606...

(5) Analysis on sales, freight and delivery time

(a)Calculate days between purchasing, delivering and estimated delivery

```
SELECT
date_diff(
date(order_delivered_customer_date),date(order_purchase_timestamp),day)
as diff_between_ordered_n_delivered,
date_diff(
date(order_estimated_delivery_date),date(order_delivered_customer_date),day)
as diff_between_estimated_n_actual_delivered,
date_diff(
date(order_estimated_delivery_date),date(order_purchase_timestamp),day)
as diff_between_order_n_estimated_delivery

FROM `my-new-project-scaler-dsml.Scaler_project1.orders`
```

10	34	-5	29

(b) 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(
date(order_purchase_timestamp),date(order_delivered_customer_date),day)
as time_to_delivery,
date_diff(
date(order_estimated_delivery_date),date(order_delivered_customer_date),day)
as diff_estimated_delivery
FROM `my-new-project-scaler-dsml.Scaler_project1.orders`
```

JOB INFORMATION | RESULTS | JSON | EXECUTION D

Row	time_to_delivery	diff_estimated_delivery
1	-30	-12
2	-31	29
3	-36	17
4	-31	2
5	-33	1
6	-30	2
7	-44	-4
8	-41	-4
9	-37	-1
10	-34	-5

(c)Group data by state, take mean of freight_value, time_to_delivery, diff_estimated_delivery

```
with temp_table as (SELECT
date_diff(date(order_delivered_customer_date),date(order_purchase_timestamp),day) as
time_to_delivery,
date_diff(date(order_estimated_delivery_date),date(order_delivered_customer_date),day)
as diff_estimated_delivery,
item.freight_value,
o.customer_id
FROM `my-new-project-scaler-dsml.Scaler_project1.orders` o join
```

```
`my-new-project-scaler-dsml.Scaler_project1.order_items` item on item.order_id=
o.order_id)


select c.customer_state,
avg(temp_table.diff_estimated_delivery) as mean_estimated_delivery,
avg(temp_table.time_to_delivery) as mean_time_delivery,
avg(temp_table.freight_value) as mean_freight,
from temp_table join
`my-new-project-scaler-dsml.Scaler_project1.customers` c on
c.customer_id=temp_table.customer_id
group by c.customer_state;
```

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

Row	customer_state	mean_estimated_del	mean_time_del	mean_freight
1	MT	14.571841851494709	17.907425265188...	28.1662843601896
2	MA	9.9062499999999929	21.589999999999...	38.25700242718446
3	AL	8.73536299765808	24.447306791569...	35.843671171171...
4	SP	11.207910772344571	8.66225265379071	15.147275390419...
5	MG	13.342649221955588	11.920724626461...	20.630166806306...
6	PE	13.450171821305863	18.224513172966...	32.917862679955...
7	RJ	12.014774494556768	15.074791460483...	20.96092393168248
8	DF	12.200424628450103	12.893842887473...	21.041354945968...
9	RS	14.1341920756563	15.134518180335...	21.735804330392...
10	SE	10.002666666666677	21.418666666666...	36.653168831168...

(d) Sort the data to get the following:

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

```
with temp_table as (SELECT
date_diff(date(order_delivered_customer_date),date(order_purchase_timestamp),day) as
time_to_delivery,
date_diff(date(order_estimated_delivery_date),date(order_delivered_customer_date),day)
as diff_estimated_delivery,
item.freight_value,
o.customer_id
FROM `my-new-project-scaler-dsml.Scaler_project1.orders` o join
`my-new-project-scaler-dsml.Scaler_project1.order_items` item on item.order_id=
o.order_id)
```

```sql
select c.customer_state,
avg(temp_table.freight_value) as avg_freight,
from temp_table join
`my-new-project-scaler-dsml.Scaler_project1.customers` c on
c.customer_id=temp_table.customer_id
group by c.customer_state order by avg_freight desc LIMIT 5;
```

JOB INFORMATION | RESULTS | JSON | E

Row	customer_state	avg_freight
1	RR	42.9844230...
2	PB	42.7238039...
3	RO	41.0697122...
4	AC	40.0733695...
5	PI	39.1479704...

Top 5 states with highest/lowest average time to delivery

```sql
Select
avg(date_diff(date(order_delivered_customer_date),date(order_purchase_timestamp),day))
as avg_time_to_delivery_in_days,
customer_state
from `my-new-project-scaler-dsml.Scaler_project1.orders` o
join `my-new-project-scaler-dsml.Scaler_project1.customers` c
on o.customer_id = c.customer_id
group by 2 order by avg_time_to_delivery_in_days desc limit 5
```

JOB INFORMATION | RESULTS | JSON | EXECUTION DETAILS

Row	avg_time_to_delivery_in_days	customer_state
1	29.341463414634148	RR
2	27.179104477611947	AP
3	26.358620689655169	AM
4	24.501259445843843	AL
5	23.725158562367902	PA

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

```
Select

avg(date_diff(date(order_delivered_customer_date),date(order_estimated_delivery_date),
day)) delivered_time_diff_to_estimated_date,

customer_state

from `my-new-project-scaler-dsml.Scaler_project1.orders` o

join `my-new-project-scaler-dsml.Scaler_project1.customers` c

on o.customer_id = c.customer_id

group by 2 order by

delivered_time_diff_to_estimated_date asc

limit 5
```

JOB INFORMATION | RESULTS | JSON | EXECUTION DETAILS

Row	delivered_time_diff_to_estimated_date	customer_state
1	-20.724999999999998	AC
2	-20.102880658436224	RO
3	-19.686567164179106	AP
4	-19.5655172413793	AM
5	-17.292682926829272	RR

Bigger negative value indicates order is delivered much before estimated delivery date

(6) Payment type analysis:

(a) Month over Month count of orders for different payment types

```
select
extract(month from o.order_purchase_timestamp) as Month,
p.payment_type,
count(o.order_id) as total_no_of_orders
FROM `my-new-project-scaler-dsml.Scaler_project1.orders` o
join `my-new-project-scaler-dsml.Scaler_project1.payments` p on o.order_id=
p.order_id
group by 1,2 order by Month,total_no_of_orders desc ;
```

JOB INFORMATION | RESULTS | JSON | EXECUTION DETAILS | EXE

Row	Month	payment_type	total_no_of_orders
1	1	credit_card	6103
2	1	UPI	1715
3	1	voucher	477
4	1	debit_card	118
5	2	credit_card	6609
6	2	UPI	1723
7	2	voucher	424
8	2	debit_card	82
9	3	credit_card	7707
10	3	UPI	1942
11	3	voucher	591

(b) Count of orders based on the no. of payment installments

```
select
p.payment_installments,
count(p.order_id) as total_no_of_orders
```

```
from `my-new-project-scaler-dsml.Scaler_project1.payments` p
group by 1 order by 1, total_no_of_orders desc ;
```

Row	payment_installments	total_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
11	10	5328

Actionable insights:

(1) 41% sales is happening between may, june, july, aug

approx 30% sales in nov, dec, jan, feb holiday

(2) Approximately 65% of the orders are bought between 2pm till 3am. Purchases are mostly done at night time.

(3) City of Sao paolo and Rio accuries 23% of total customers

(4) State of Sao paolo, Rio and Minas Gerais has 67% of total customers

(5) State of Sao paolo, Rio and Minas Gerais has 63% of total order values

(6) Roraima, Amapá, Acre has lowest order values

(7) State of Sao paolo, Paraná and Minas Gerais are 3 fastest delivery service state in that order, Rio comes 6th, though Rio has 2nd highest order value

(8) State of Amapá, Roraima and Amazonas are 3 slowest delivery service state in that order

(9) Debit card payment has lowest frequency

Recommendations:

(1) Give discount offers on gifting items during holiday season nov-feb to increase sales

(2) Stock up more daily goods and give buy 1 get 1 type offer in daily goods during may, june, july

(3) Share push notifications with offers valid for 2/3 hours during day time.

(4) Open multiple target store in Rio as the delivery service is poor compared to order values we are getting from Rio area.

__or decrease delivery partner in Acre and increase in Rio

(5) Give pay later type option in debit card