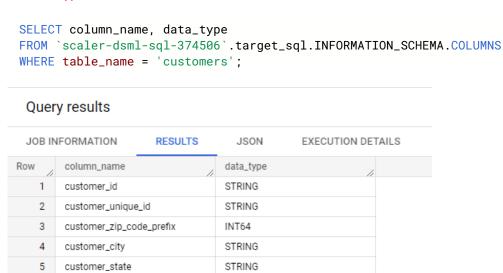
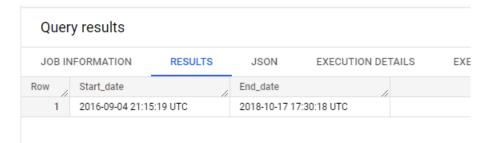
Query 1. Import the dataset and do usual exploratory analysis steps like checking the structure & characteristics of the dataset

1.1 Data type of columns in a table



1.2 Time period for which the data is given

```
select min(order_purchase_timestamp) as Start_date,
max(order_purchase_timestamp) as End_date,
from `target_sql.orders`;
```



Query 2. In-depth Exploration:

2.1 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?

```
with cte as
(
select * from
(
select *, lag(Amount) over(order by Year) as Pre_1 from
(
select
extract(Year from o.order_purchase_timestamp) as Year,
count(*) as count_of_orders,
round(sum(oi.price),2) as Amount,
from `target_sql.orders` o
inner join `target_sql.order_items` oi on o.order_id = oi.order_id
group by Year
order by Year
)
order by Year
)
select cte.Year, cte.count_of_orders,cte.Amount,
round(((cte.Amount - cte.Pre_1) / cte.Amount * 100),2) as sales_percentage
from cte
```

Query results

JOB IN	IFORMATION	RESULTS	JSON EXEC	CUTION DETAILS	EXECUTION
Row	Year	count_of_orders	Amount	sales_percentage	
1	2016	370	49785.92	null	
2	2017	50864	6155806.98	99.19	
3	2018	61416	7386050.8	16.66	

We can definitely see a growing trend in e-commerce in Brazil. In 2017, sales increased by 99%, and in 2018, close to 17%.

Can we see some seasonality with peaks at specific months?

```
with cte as
(
select *,
```

```
lag(Amount) over(order by Year, Month) as Pre_1
from
(
select
extract(Year from o.order_purchase_timestamp) as Year,
extract(Month from o.order_purchase_timestamp) as Month,
count(*) as count_of_orders,
round(sum(oi.price),2) as Amount,
from `target_sql.orders` o
inner join `target_sql.order_items` oi on o.order_id = oi.order_id
group by Year, Month
order by Year, Month
)
)
select cte.Year, cte.Month,cte.count_of_orders,cte.Amount,
round(((cte.Amount - cte.Pre_1) / cte.Amount * 100),2) as sales_percentage
from cte
order by Year,Month
```

Query results

JOB IN	IFORMATION	RESULTS	JSON E	XECUTION DETAIL	LS EXECUTION GF
Row	Year	Month	count_of_orders	Amount	sales_percentage
1	2016	9	6	267.36	null
2	2016	10	363	49507.66	99.46
3	2016	12	1	10.9	-454098.72
4	2017	1	955	120312.87	99.99
5	2017	2	1951	247303.02	51.35
6	2017	3	3000	374344.3	33.94
7	2017	4	2684	359927.23	-4.01
8	2017	5	4136	506071.14	28.88
9	2017	6	3583	433038.6	-16.87
10	2017	7	4519	498031.48	13.05
11	2017	8	4910	573971.68	13.23
10	2017	0	4021	604401.60	0.00

Yes. Every year, sales increase before Christmas and in January as well.

In the summer, sales were consistent, but dropped in the fall.

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

```
select distinct
case
when hour >= 0 and hour < 6 then 'Dawn'
when hour >= 6 and hour <12 then 'Morning'
when hour >= 12 and hour <18 then 'Afternoon'</pre>
```

```
when hour >= 18 and hour <= 23 then 'Night'
end as Timings,
count(*) as count_of_orders
from
select
EXTRACT(HOUR from order_purchase_timestamp) as hour,
from `target_sql.orders`
group by Timings;
 Query results
 JOB INFORMATION
                     RESULTS
                                 JSON
                                          EXECUT
Row
                               count_of_orders
       Timings
   1
      Morning
                                    22240
```

3

4

Afternoon

Night

Customers tend to make purchases in the afternoon, followed by the night and morning. There are less orders in Dawn.

4740

38361

34100

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

3.1 Get month on month orders by states

```
select *,
sum(count_of_orders) over(rows between unbounded preceding and current row) as cumulat
ive_count
from
(
select
c.customer_state,
extract(Year from order_purchase_timestamp) as Year,
extract(Month from order_purchase_timestamp) as Month,
count(*) as count_of_orders
from `target_sql.customers` c
inner join `target_sql.orders` o on c.customer_id = o.customer_id
group by Year, Month, c.customer_state
order by c.customer_state, Year, Month, count_of_orders
)
order by customer_state, Year, Month, count_of_orders
```

JOB IN	FORMATION	RESULTS	JSON	EXECUTION DET	TAILS EXECU	JTION GRAPH PR	REVIEW
Row	customer_state	//	Year //	Month	count_of_orders	cumulative_cour	
1	AC		2017	1	2	2	
2	AC		2017	2	3	5	
3	AC		2017	3	2	7	
4	AC		2017	4	5	12	
5	AC		2017	5	8	20	
6	AC		2017	6	4	24	
7	AC		2017	7	5	29	
8	AC		2017	8	4	33	
9	AC		2017	9	5	38	
10	AC		2017	10	6	44	

3.2 Distribution of customers across the states in Brazil

```
select distinct
customer_state,
count(*) over(partition by customer_state) as customer_count
from `target_sql.customers`
order by customer_count;
```

JOB II	NFORMATION RESULTS	JSON	EXECUTION DETAILS	EXECUTION GRAPH PR
Row	customer_state	customer_count		
1	RR	46		
2	AP	68		
3	AC	81		
4	AM	148		
5	RO	253		
6	ТО	280		
7	SE	350		
8	AL	413		
9	RN	485		
10	PI	495		
11	PB	536		
12	MS	715		
13	MA	747		
14	MT	907		
15	PA	975		

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

4.1 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

```
select
round(((payment_value_2018-
payment_value_2017) / payment_value_2017 * 100 ),2) as pct_incr
from
(
select
extract(Month from order_purchase_timestamp) as Month,
round(sum(case when extract(Year from order_purchase_timestamp) = 2017 then p.payment_
value ELSE 0 END),2) as payment_value_2017,
round(sum(case when extract(Year from order_purchase_timestamp) = 2018 then p.payment_
value ELSE 0 END),2) as payment_value_2018
from
`target_sql.orders` o
inner join `target_sql.payments` p on o.order_id = p.order_id
group by month
order by Month
where Month in(1,2,3,4,5,6,7,8)
```

Query results

JOB IN	IFORMATION	RESULTS JSC	ON EXECUTION D	DETAILS I	EXECUTION GRA
Row	Month	payment_value_2017	payment_value_2018	pct_incr	
1	1	138488.04	1115004.18	705.13	
2	2	291908.01	992463.34	239.99	
3	3	449863.6	1159652.12	157.78	
4	4	417788.03	1160785.48	177.84	
5	5	592918.82	1153982.15	94.63	
6	6	511276.38	1023880.5	100.26	
7	7	592382.92	1066540.75	80.04	
8	8	674396.32	1022425.32	51.61	

The first quarter of 2018 showed a huge increase over 2017.

4.2 Mean & Sum of price and freight value by customer state

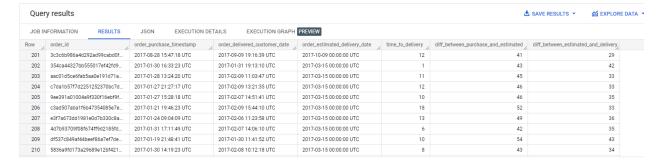
```
select distinct
c.customer_state,
round(avg(oi.price) over(partition by c.customer_state),2) as mean_of_price_value,
round(sum(oi.price) over(partition by c.customer_state),2) as sum_of_price_value,
round(avg(oi.freight_value) over(partition by c.customer_state),2) as mean_of_freight_
value,
round(sum(oi.freight_value) over(partition by c.customer_state),2) as sum_of_freight_v
alue
from `target_sql.customers` c
inner join `target_sql.orders` o on c.customer_id = o.customer_id
inner join `target_sql.order_items` oi on o.order_id = oi.order_id
order by c.customer_state;
```

Quer	y results				
JOB IN	FORMATION	RESULTS JSON	EXECUTION DET	AILS EXECUTION	GRAPH PREVIEW
Row	customer_state	mean_of_price_value	sum_of_price_value	mean_of_freight_value	sum_of_freight_value
1	AC	173.73	15982.95	40.07	3686.75
2	AL	180.89	80314.81	35.84	15914.59
3	AM	135.5	22356.84	33.21	5478.89
4	AP	164.32	13474.3	34.01	2788.5
5	BA	134.6	511349.99	26.36	100156.68
6	CE	153.76	227254.71	32.71	48351.59
7	DF	125.77	302603.94	21.04	50625.5
8	ES	121.91	275037.31	22.06	49764.6
9	GO	126.27	294591.95	22.77	53114.98
10	MA	145.2	119648.22	38.26	31523.77
11	MG	120.75	1585308.03	20.63	270853.46
12	MS	142.63	116812.64	23.37	19144.03
13	MT	148.3	156453.53	28.17	29715.43
14	ΡΔ	165.69	178947 81	35.83	38699.3

Query 5. Analysis on sales, freight and delivery time

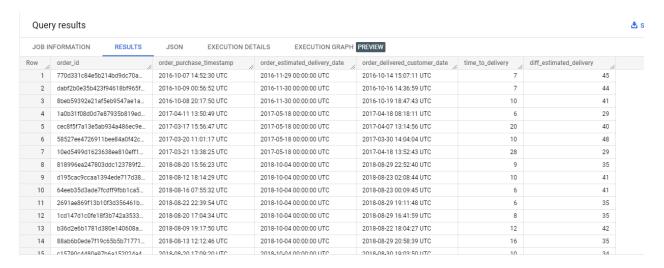
5.1 Calculate days between purchasing, delivering and estimated delivery

```
select
order_id,
order_purchase_timestamp,
order_delivered_customer_date,
order_estimated_delivery_date,
abs(date_diff(order_delivered_customer_date,order_purchase_timestamp, DAY)) as time_to
_delivery,
abs(date_diff(order_estimated_delivery_date,order_purchase_timestamp, DAY)) as diff_be
tween_purchase_and_estimated,
abs(date_diff(order_estimated_delivery_date, order_delivered_customer_date, DAY)) as d
iff_between_estimated_and_delivery
from `target_sql.orders`
```



5.2 Find time_to_delivery & diff_estimated_delivery.

```
select
order_id,
order_purchase_timestamp,
order_estimated_delivery_date,
order_delivered_customer_date,
abs(date_diff(order_delivered_customer_date, order_purchase_timestamp, DAY)) as time_to
_delivery,
abs(date_diff(order_estimated_delivery_date, order_delivered_customer_date, DAY)) as d
iff_estimated_delivery
from `target_sql.orders`
```



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

```
select distinct
c.customer_state,
round(avg(oi.freight_value) over(partition by c.customer_state),2) as mean_of_freight_
value,
round(avg(date_diff(order_delivered_customer_date,order_purchase_timestamp, DAY)) over
(partition by c.customer_state),2) as mean_time_to_delivery,
round(avg(date_diff(order_estimated_delivery_date, order_delivered_customer_date, DAY))
) over(partition by c.customer_state),2) as mean_diff_estimated_delivery
from `target_sql.customers` c
inner join `target_sql.orders` o on c.customer_id = o.customer_id
inner join `target_sql.order_items` oi on o.order_id = oi.order_id
order by c.customer_state;
```

JOB IN	IFORMATION	RESULTS	JSON EXECUT	ION DETAILS EXECU	TION GRAPH PREVIEW
Row	customer_state	//	mean_of_freight_value	mean_time_to_delivery	mean_diff_estimated_delivery
1	AC		40.07	20.33	20.01
2	AL		35.84	23.99	7.98
3	AM		33.21	25.96	18.98
4	AP		34.01	27.75	17.44
5	BA		26.36	18.77	10.12
6	CE		32.71	20.54	10.26
7	DF		21.04	12.5	11.27
8	ES		22.06	15.19	9.77
9	GO		22.77	14.95	11.37

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

```
select distinct
c.customer_state,
round(avg(oi.freight_value) over(partition by c.customer_state),2) as mean_of_freight_
value,
from `target_sql.customers` c
inner join `target_sql.orders` o on c.customer_id = o.customer_id
inner join `target_sql.order_items` oi on o.order_id = oi.order_id
order by mean_of_freight_value desc limit 5;
```

Quer	y results			
JOB IN	IFORMATION	RESULTS	JSON	EXECUTION DETAILS
Row	customer_state	//	mean_of_freigh	t_value
1	RR			42.98
2	PB			42.72
3	RO			41.07
4	AC			40.07
5	PI			39.15

select distinct

```
c.customer_state,
round(avg(oi.freight_value) over(partition by c.customer_state),2) as mean_of_freight_
value,
from `target_sql.customers` c
inner join `target_sql.orders` o on c.customer_id = o.customer_id
inner join `target_sql.order_items` oi on o.order_id = oi.order_id
order by mean_of_freight_value limit 5;
```

JOB	INFORMATION	RESULTS	JSON EXECUT	ION DETAILS
Row	customer_state	//	mean_of_freight_value	
1	SP		15.15	
2	PR		20.53	
3	MG		20.63	
4	RJ		20.96	
5	DF		21.04	

5.6 Top 5 states with highest/lowest average time to delivery

```
select distinct
c.customer_state,
round(avg(date_diff(order_delivered_customer_date,order_purchase_timestamp, DAY)) over
(partition by c.customer_state),2) as mean_time_to_delivery,
from `target_sql.customers` c
inner join `target_sql.orders` o on c.customer_id = o.customer_id
inner join `target_sql.order_items` oi on o.order_id = oi.order_id
order by mean_time_to_delivery desc limit 5;
```

Quer	y results			
JOB IN	IFORMATION	RESULTS	JSON	EXECUTION DETA
Row	customer_state	//	mean_time_to	o_delivery
1	RR			27.83
2	AP			27.75
3	AM			25.96
4	AL			23.99
5	PA			23.3

```
select distinct
c.customer_state,
round(avg(date_diff(order_delivered_customer_date,order_purchase_timestamp, DAY)) over
(partition by c.customer_state),2) as mean_time_to_delivery,
from `target_sql.customers` c
inner join `target_sql.orders` o on c.customer_id = o.customer_id
inner join `target_sql.order_items` oi on o.order_id = oi.order_id
order by mean_time_to_delivery limit 5;
```

JOB IN	IFORMATION	RESULTS	JSON	EXECUTION DET	AILS
Row	customer_state	//	mean_time_to	_delivery	
1	SP			8.26	
2	PR			11.48	
3	MG			11.52	
4	DF			12.5	
5	SC			14.52	

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

```
select distinct
c.customer_state,
round(avg(date_diff(order_estimated_delivery_date, order_delivered_customer_date, DAY)
) over(partition by c.customer_state),2) as mean_diff_estimated_delivery
from `target_sql.customers` c
inner join `target_sql.orders` o on c.customer_id = o.customer_id
inner join `target_sql.order_items` oi on o.order_id = oi.order_id
order by mean_diff_estimated_delivery limit 5;
```

Quer	y results			
JOB IN	IFORMATION	RESULTS	JSON	EXECUTION DETAILS
Row	customer_state	//	mean_diff_es	timated_delivery
1	AL			7.98
2	MA			9.11
3	SE			9.17
4	ES			9.77
5	BA			10.12

Query 6- Payment type analysis:

6.1 Month over Month count of orders for different payment type

```
select *,
sum(count_of_orders) over(rows between unbounded preceding and current row) as cumulat
ive_count
from
(
select
extract(Year from order_purchase_timestamp) as Year,
extract(Month from order_purchase_timestamp) as Month,
p.payment_type,
count(*) as count_of_orders
from `target_sql.payments` p
right join `target_sql.orders` o on p.order_id = o.order_id
group by Year, Month, payment_type
order by Year, Month, payment_type
)
order by Year, Month, payment_type;
```

Quer	y results					
JOB INFORMATION		RESULTS	JSON	EXECUTION DETAILS EXECUTION GRAPH		
Row	Year	Month	payment_type	//	count_of_orders	cumulative_cour
1	2016	9	null		1	1
2	2016	9	credit_card		3	4
3	2016	10	UPI		63	67
4	2016	10	credit_card		254	321
5	2016	10	debit_card		2	323
6	2016	10	voucher		23	346
7	2016	12	credit_card		1	347
8	2017	1	UPI		197	544
9	2017	1	credit_card		583	1127
10	2017	1	debit_card		9	1136
11	2017	1	voucher		61	1197

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

```
select payment_installments,
count(*) as count_of_orders
from `target_sql.payments`
group by payment_installments;
```

Quer	y results			
JOB IN	FORMATION	RESULT	S JSON	EXECL
Row	payment_installments	,	count_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	

Actionable Insights:

- ==> We can see a growing trend in e-commerce in Brazil. In 2017, sales increased by 99%, and in 2018, close to 17%.
- ==> As per data in 10-2016 has more orders, 11-2017 and followed by 01-2018.
- ==> Every year, sales increase before Christmas and in January as well. In the summer, sales were consistent, but dropped in the fall.
- ==> Customers tend to make purchases in the afternoon, followed by the night and morning. There are less orders in Dawn.
- ==> Compared to other states, RR, AP, and AC have fewer customers.
- ==> Sales in the first quarter of 2018 showed a huge increase over 2017.
- ==> There is high delivery cost in RR, PB, RO, AC and PI states.
- ==> There is low delivery cost in SP, PR, MG, RJ and DF states
- ==> Delivery of orders in RR, AP, AM, AL and PA states is taking longer than expected.
- ==> Whereas in SP, PR, MG, DF and SC states, orders are delivered quickly.
- ==> More orders are placed through UPI and credit card than through other methods.
- ==> There are more payments done through single payments than installments.

Recommendations:

- ==> Sales drop during the fall season. This can be addressed by offering discounts on seasonal clothing and essentials.
- ==> Customers make more purchases in the afternoon and night. In the morning hours, less resources are needed to manage work.
- ==> Usually customers shop during the afternoon, so we need to provide some offers We can provide some benefits during the day to increase sales during Dawn.
- ==> SP, PR, MG and DF have low delivery costs and orders are delivered quickly, which is why there are more customers in those states.
- ==> RR, AC and AP states have less customers. More time to deliver and higher delivery costs could be the reason for this. We can solve this problem by offering seasonal offers and a discount on delivery or by temporarily eliminating delivery costs.
- ==> More payments are done through UPI and credit cards. Possibly because of fast and easy payments. It is essential to ensure there are fewer transaction failures and no third-party fraud payments.

.

	Target SQL Case				
==> Customers are making more purchases through single payments than installments which is a positive sign. No cost EMI option may be beneficial for bulk orders.					