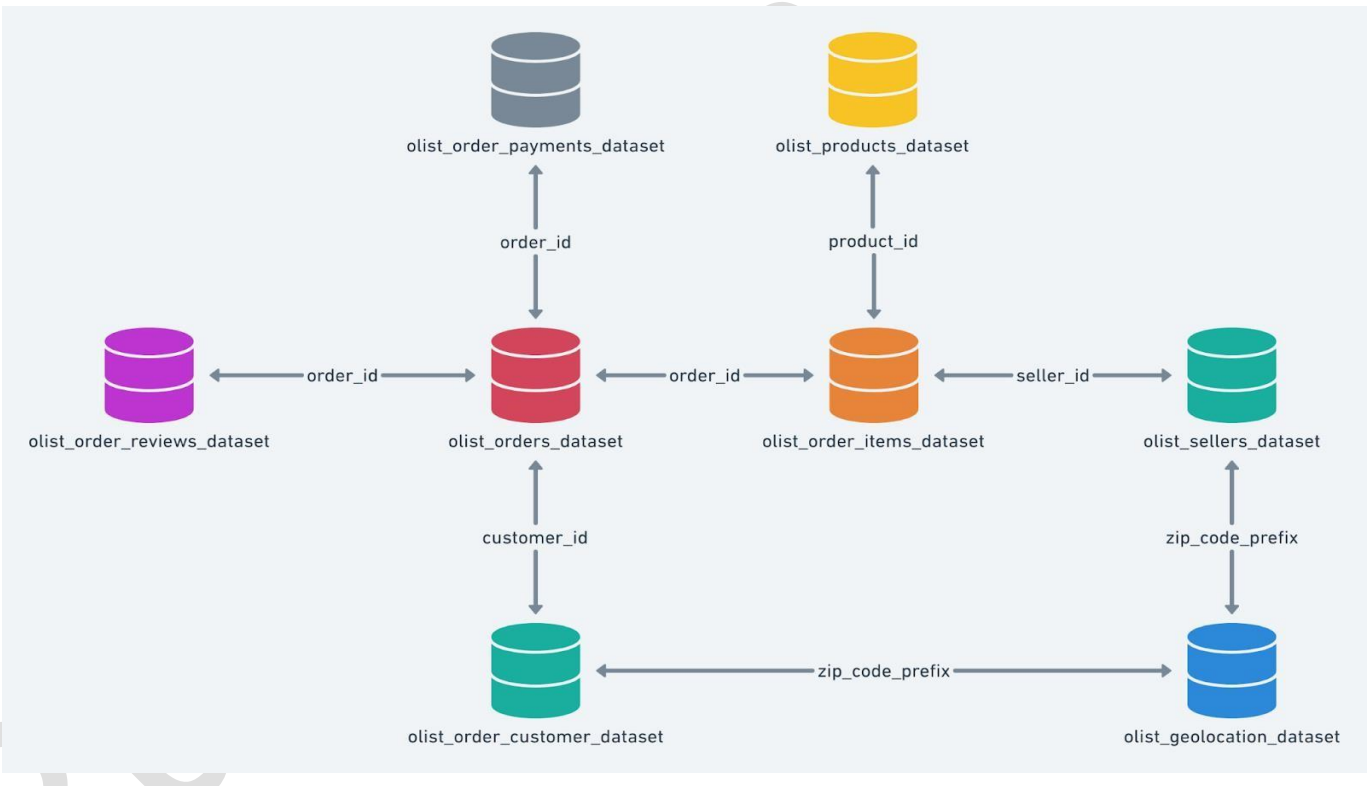
Target – case study using SQL

Context

Target is one of the world's most recognized brands and one of America's leading retailers. Target makes itself a preferred shopping destination by offering outstanding value, inspiration, innovation and an exceptional guest experience that no other retailer can deliver.

This business case has information of 100k orders from 2016 to 2018 made at Target in Brazil. Its features allows viewing an order from multiple dimensions: from order status, price, payment and freight performance to customer location, product attributes and finally reviews written by customers.

High level overview of relationship between datasets:



1: Import the dataset and do usual exploratory analysis steps like checking the structure & characteristic

A: Data type of all columns in the "customers" table.
Hint: We want you to display the data type of each column present in the “customers” table.

Ans :

```
select column_name, data_type
from `target_SQL.INFORMATION_SCHEMA.COLUMNS`
WHERE table_name = 'customers';
```

Row	column_name	data_type
1	customer_id	STRING
2	customer_unique_id	STRING
3	customer_zip_code_prefix	INT64
4	customer_city	STRING
5	customer_state	STRING

Insight : Here we have more string data type in customers table.

B: Get the time range between which the orders were placed.
Hint: We want you to get the date & time when the first and last orders in our dataset were placed

Ans :

```
SELECT Min(order_purchase_timestamp) as First_order_placed
 Max(order_purchase_timestamp) as Last_order_placed
 FROM `target_SQL.orders`;
```

Row	First_order_placed	Last_order_placed
1	2016-09-04 21:15:19 UTC	2018-10-17 17:30:18 UTC

Insight : This market was opened between 04-09-2016 and 17-10-2018 were all orders get placed.

C: Count the Cities & States of customers who ordered during the given period.
Hint: We want you to count the number of unique cities and states present in our dataset.

Ans :

```
SELECT  count(distinct customer_city)as No_of_city, count(distinct customer_state) as No_of_state
FROM `target_SQL.customers` as c
join `target_SQL.orders` as o
on c.customer_id = o.customer_id;
```

Row	No_of_city	No_of_state
1	4119	27

Insight : There are 4119 city and 27 states who ordered during given period.

2. In-depth Exploration:

A. Is there a growing trend in the no. of orders placed over the past years?
Hint: We want you to find out if no. of orders placed has increased gradually in each month, over the past years.

Ans :

```
select *
from (SELECT extract (year from order_purchase_timestamp) as years,
count(order_id) as no_of_orders,
FROM `target_SQL.orders`
group by extract (year from order_purchase_timestamp)) as x
order by years;
```

Row	years	no_of_orders
1	2016	329
2	2017	45101
3	2018	54011

Insight : As we can see the number of orders are increasing year by year. In the year of 2016 the orders are much less as compar to the 2017 and 2018.

B. Can we see some kind of monthly seasonality in terms of the no. of orders being placed?

Ans :

```
select count(order_id) as no_of_orders,extract( month from order_purchase_timestamp) as month,
extract( year from order_purchase_timestamp) as years
FROM `target_SQL.orders`
group by month,years
order by years asc, month asc;
```

Row	no_of_orders	month	years
1	4	9	2016
2	324	10	2016
3	1	12	2016
4	800	1	2017
5	1780	2	2017
6	2682	3	2017
7	2404	4	2017
8	3700	5	2017
9	3245	6	2017
10	4026	7	2017
11	4331	8	2017
12	4285	9	2017
13	4631	10	2017

Insight : In year of 2016 with respect to the month the orders get increase but suddenly get decrease at the end of the year Also happening in the other year also. In 2018 the order get increases in some month and get decreases also.

C. During what time of the day, do the Brazilian customers mostly place their orders? (Dawn, Morning, Afternoon or Night)

- 0-6 hrs : Dawn
- 7-12 hrs : Mornings
- 13-18 hrs : Afternoon
- 19-23 hrs : Night

Ans:

```
select count(order_id) as count_of_order,
 case
when  x.hour between 0 and 6 then 'dawn'
when  x.hour between 7 and 12 then 'morning'
when  x.hour between 13 and 18 then 'afternoon'
when  x.hour between 19 and 23 then 'night'
end as Time_of_day
from(select order_id,order_purchase_timestamp , extract (hour from order_purchase_timestamp) as hour
FROM `target_SQL.orders`) as x
group by Time_of_day
order by count_of_order;
```

Row	count_of_order	Time_of_day
1	5242	dawn
2	27733	morning
3	28331	night
4	38135	afternoon

Insight : From the result table it is clearly shown that Brazilian customer place there orders mostly in the afternoon (13-19 hrs).

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

A : Get the month on month no. of orders placed in each state.
Hint: We want you to get the no. of orders placed in each state, in each month by our customers.

Ans :

```
SELECT  count(o.order_id) as no_of_orders,c.customer_state,
extract( month from o.order_purchase_timestamp) as month,
extract( year from o.order_purchase_timestamp) as years
FROM `target_SQL.customers` as c
join `target_SQL.orders` as o
on c.customer_id = o.customer_id
group bmonth,years,c.customer_state
order by years asc, month asc;
```

.

Row	no_of_orders	customer_state	month	years
1	1	RR	9	2016
2	1	RS	9	2016
3	2	SP	9	2016
4	113	SP	10	2016
5	24	RS	10	2016
6	4	BA	10	2016
7	19	PR	10	2016
8	56	RJ	10	2016
9	4	RN	10	2016
10	3	MT	10	2016

Insight : Here we are getting the numbers of orders placed in each of the state and in each month of the years. In 2016 the SP state placed more order as compare to other state.

B: How are the customers distributed across all the states?
Hint: We want you to get the no. of unique customers present in each state.

Ans :

```
SELECT COUNT(customer_unique_id) as No_of_unique_customer,customer_state
FROM `target_SQL.customers`
Group by customer_state;
```

Row	No_of_unique_customer	customer_state
1	485	RN
2	1336	CE
3	5466	RS
4	3637	SC
5	41746	SP
6	11635	MG
7	3380	BA
8	12852	RJ
9	2020	GO
10	747	MA

Insight: All are the unique customers which are present in state. We have state SP which have highest no unique customers.

4: Impact on Economy: Analyze the money movement by e-commerce by looking at order prices, freig

A: Get the % increase in the cost of orders from year 2017 to 2018 (include months between Jan to Aug only).
Hint: You can use the payment_value column in the payments table to get the cost of orders.

Ans :

```
 with cte
 as
(select sum(p.payment_value) as cost, o.year,
from (SELECT order_id, extract(year from order_purchase_timestamp ) as Year
 FROM `target_SQL.orders`
where extract(year from order_purchase_timestamp ) in (2017, 2018)
order by year) as o
join `target_SQL.payments` as p
on o.order_id = p.order_id
group by o.year)

select year,cost,
round((cost - LAG (cost) OVER (ORDER BY year ASC))/LAG (cost) OVER (ORDER BY year ASC)*100,2) AS
pct_increase_inCost
from cte
```

Row	year	cost	pct_increase_inCost
1	2017	7249746.729999...	null
2	2018	8699763.049999...	20.0

Insight: As per the result table there is hike of 20% in the year of 2018 as compare with the last year

B: Calculate the Total & Average value of order price for each state.
Hint: We want you to fetch the total price and the average price of orders for each state.

Ans :

```
 with cte as
 (
 SELECT *
 FROM (select customer_id,customer_state from `target_SQL.customers`) as c
join (select order_id,customer_id from `target_SQL.orders`) as o
on c.customer_id = o.customer_id
join (select order_id,price from `target_SQL.order_items`) as ot
on o.order_id = ot.order_id )

 select customer_state, count(price) as total_price, round(avg(price),2) as avg_price
 from cte
 group by customer_state
 order by total_price
```

Row	customer_state	total_price	avg_price
1	RR	52	150.57
2	AP	82	164.32
3	AC	92	173.73
4	AM	165	135.5
5	RO	278	165.97
6	TO	315	157.53
7	SE	385	153.04
8	AL	444	180.89
9	RN	529	156.97
10	PI	542	160.36

Insight: Total price and average price of the orders with respect to each sate.

C: Calculate the Total & Average value of order freight for each state.
Hint: We want you to fetch the total freight value and the average freight value of orders for each state.

Ans :

```
 with cte as
(
SELECT *
FROM (select customer_id,customer_state from `target_SQL.customers`) as c
join (select order_id,customer_id from `target_SQL.orders`) as o
on c.customer_id = o.customer_id
join (select order_id, freight_value from `target_SQL.order_items`) as ot
 on o.order_id = ot.order_id )

select customer_state, count(freight_value) as total_freight_value,
round(avg(freight_value),2) as  avg_freight_value
from cte
group by customer_state
order by total_freight_value
```

Row	customer_state	total_freight_value	avg_freight_value
1	RR	52	42.98
2	AP	82	34.01
3	AC	92	40.07
4	AM	165	33.21
5	RO	278	41.07
6	TO	315	37.25
7	SE	385	36.65
8	AL	444	35.84
9	RN	529	35.65
10	PI	542	39.15

Insight: Total price and average value of the freight with respect to each state.

5: Analysis based on sales, freight and delivery time.

A: Find the no. of days taken to deliver each order from the order's purchase date as delivery time.
Also, calculate the difference (in days) between the estimated & actual delivery date of an order. Do this in a single query.

Ans:

```
 SELECT order_purchase_timestamp, order_delivered_customer_date,
datetime_diff(order_delivered_customer_date,order_purchase_timestamp,Day) as Time_to_delivery,
order_estimated_delivery_date, order_delivered_customer_date,
datetime_diff(order_estimated_delivery_date,order_delivered_customer_date, Day) as Diff_estimate_delivery
```

```
FROM `target_SQL.orders`
```

Row	order_purchase_timestamp	order_delivered_customer_date	Time_to_delivery	order_estimated_delivery_date	order_delivered_customer_date_1	Diff_estimate_delivery
1	2018-02-19 19:48:52 UTC	2018-03-21 22:03:51 UTC	30	2018-03-09 00:00:00 UTC	2018-03-21 22:03:51 UTC	-12
2	2016-10-09 15:39:56 UTC	2016-11-09 14:53:50 UTC	30	2016-12-08 00:00:00 UTC	2016-11-09 14:53:50 UTC	28
3	2016-10-03 21:01:41 UTC	2016-11-08 10:58:34 UTC	35	2016-11-25 00:00:00 UTC	2016-11-08 10:58:34 UTC	16
4	2017-04-15 15:37:38 UTC	2017-05-16 14:49:55 UTC	30	2017-05-18 00:00:00 UTC	2017-05-16 14:49:55 UTC	1
5	2017-04-14 22:21:54 UTC	2017-05-17 10:52:15 UTC	32	2017-05-18 00:00:00 UTC	2017-05-17 10:52:15 UTC	0
6	2017-04-16 14:56:13 UTC	2017-05-16 09:07:47 UTC	29	2017-05-18 00:00:00 UTC	2017-05-16 09:07:47 UTC	1
7	2017-04-08 21:20:24 UTC	2017-05-22 14:11:31 UTC	43	2017-05-18 00:00:00 UTC	2017-05-22 14:11:31 UTC	-4
8	2017-04-11 19:49:45 UTC	2017-05-22 16:18:42 UTC	40	2017-05-18 00:00:00 UTC	2017-05-22 16:18:42 UTC	-4
9	2017-04-12 12:17:08 UTC	2017-05-19 13:44:52 UTC	37	2017-05-18 00:00:00 UTC	2017-05-19 13:44:52 UTC	-1
10	2017-04-19 22:52:59 UTC	2017-05-23 14:19:48 UTC	33	2017-05-18 00:00:00 UTC	2017-05-23 14:19:48 UTC	-5

Insight: The coloum Time_to_delivery shows the number of days that orders has taken from date which it order to the date on which customer receive the order. Also we have calculated difference between estimated and actual delivery date of an order.

B: Find out the top 5 states with the highest & lowest average freight value.
Hint: We want you to find the top 5 & the bottom 5 states arranged in increasing order of the average freight value.

Ans :

```
with cte as
(
SELECT c.customer_state, round(avg(ot.freight_value),2) as avg_freight_value,
FROM (select customer_id,customer_state from `target_SQL.customers`) as c
join (select order_id,customer_id from `target_SQL.orders`) as o
on c.customer_id = o.customer_id
join (select order_id, freight_value from `target_SQL.order_items`) as ot
on o.order_id = ot.order_id        group by customer_state ),

cte2 as
(select *,RANK() OVER (ORDER BY avg_freight_value DESC) AS highest_rank,
RANK() OVER (ORDER BY avg_freight_value ASC) AS lowest_rank from cte)


 select * from cte2
 WHERE
 highest_rank <= 5 OR lowest_rank <= 5
 ORDER BY
 highest_rank ASC, lowest_rank ASC;
```

Row	customer_state	avg_freight_value	highest_rank	lowest_rank
1	RR	42.98	1	27
2	PB	42.72	2	26
3	RO	41.07	3	25
4	AC	40.07	4	24
5	PI	39.15	5	23
6	DF	21.04	23	5
7	RJ	20.96	24	4
8	MG	20.63	25	3
9	PR	20.53	26	2
10	SP	15.15	27	1

Insight: Here we have find top 5 state with the highest avg_freight_value numbered from 1-5 in highest rank column and lowest top 5 state with lowest avg_freight_value in the lowest rank Coolum.

C: Find out the top 5 states with the highest & lowest average delivery time.
Hint: We want you to find the top 5 & the bottom 5 states arranged in increasing order of the average delivery time.

Ans :

```
with cte as
(
  SELECT c.customer_state, o.order_purchase_timestamp, o.order_delivered_customer_date,
  datetime_diff(o.order_delivered_customer_date,o.order_purchase_timestamp,Day) as Time_to_delivery,
  FROM (select customer_id,customer_state from `target_SQL.customers`) as c
  join (select order_id,customer_id,order_purchase_timestamp,order_delivered_customer_date
  from `target_SQL.orders`) as o
  on c.customer_id = o.customer_id ),
 cte2 as
(
   select customer_state,avg(Time_to_delivery) as avg_delivery_time
   from cte
    group by customer_state),

cte3 as
(
   select customer_state, avg_delivery_time, RANK() OVER (ORDER BY avg_delivery_time ASC) AS ascending_rank,
   RANK() OVER (ORDER BY avg_delivery_time DESC) AS descending_rank
   from cte2)

select * from cte3
where ascending_rank <= 5 or descending_rank <= 5
order by ascending_rank asc,descending_rank asc
```

Row	customer_state	avg_delivery_time	ascending_rank	descending_rank
1	SP	8.298061489072...	1	27
2	PR	11.52671135486...	2	26
3	MG	11.54381329810...	3	25
4	DF	12.50913461538...	4	24
5	SC	14.47956019171...	5	23
6	PA	23.31606765327...	23	5
7	AL	24.04030226700...	24	4
8	AM	25.98620689655...	25	3
9	AP	26.73134328358...	26	2
10	RR	28.97560975609...	27	1

Insight: here we find the top 5 & the bottom 5 states arranged in increasing order of the average delivery time

D : Find out the top 5 states where the order delivery is really fast as compared to the estimated date of delivery.
You can use the difference between the averages of actual & estimated delivery date to figure out how fast the delivery was for each state.

Ans :

```
with cte as
(
SELECT c.customer_state,o.Diff_estimate_delivery
FROM (select customer_id,customer_state from `target_SQL.customers`) as c
join (select order_estimated_delivery_date, order_delivered_customer_date,
datetime_diff(order_estimated_delivery_date,order_delivered_customer_date, Day) as Diff_estimate_delivery,
order_id,customer_id
from `target_SQL.orders`) as o
on c.customer_id = o.customer_id)
select customer_state,avg(Diff_estimate_delivery) as avgDiff_estimate_delivery
from cte
group by customer_state
order by avgDiff_estimate_delivery
limit 5;
```

Row	customer_state	avgDiff_estimate_delivery
1	AL	7.9471032745592
2	MA	8.7684797768479665
3	SE	9.1731343283582127
4	ES	9.618546365914785
5	BA	9.93488943488944

Insight: here we have find the top 5 states where the order delivery is really fast as compared to the estimated date of Delivery.

Analysis based on the payments:

A: Find the month on month no. of orders placed using different payment types.
Hint: We want you to count the no. of orders placed using different payment methods in each month over the past years.

Ans:

```
SELECT  c.customer_state,count(o.order_id) as no_of_orders,p.payment_type,
extract( month from o.order_purchase_timestamp) as month,
extract( year from o.order_purchase_timestamp) as years,
FROM `target_SQL.customers` as c
join `target_SQL.orders` as o
on c.customer_id = o.customer_id
join `target_SQL.payments` as p
on o.order_id = p.order_id
group by month,years,c.customer_state,payment_type
order by years asc, month asc;
```

Row	customer_state	no_of_orders	payment_type	month	years
1	RR	1	credit_card	9	2016
2	RS	1	credit_card	9	2016
3	SP	1	credit_card	9	2016
4	SP	92	credit_card	10	2016
5	RS	10	UPI	10	2016
6	BA	3	credit_card	10	2016
7	SP	18	UPI	10	2016
8	SP	2	debit_card	10	2016
9	PR	3	UPI	10	2016
10	RJ	42	credit_card	10	2016

Insight: All the details of the order with month on month order placed by different payment modes as above. In the year 2016 most of the order get placed with the credit card as payment mode.

B: Find the no. of orders placed on the basis of the payment instalments that have been paid.
Hint: We want you to count the no. of orders placed based on the no. of payment instalments where at least one installment has been successfully paid.

Ans :

```
SELECT  c.customer_state,count(o.order_id) as no_of_orders,payment_installments,
extract( month from o.order_purchase_timestamp) as month,
extract( year from o.order_purchase_timestamp) as years,
FROM `target_SQL.customers` as c
join `target_SQL.orders` as o
on c.customer_id = o.customer_id
join `target_SQL.payments` as p
on o.order_id = p.order_id
where payment_installments >= 1
group by month,years,c.customer_state,payment_installments
order by years asc, month asc;
```

Row	customer_state	no_of_orders	payment_installments	month	years
1	RR	1	1	9	2016
2	RS	1	3	9	2016
3	SP	1	2	9	2016
4	SP	51	1	10	2016
5	RS	13	1	10	2016
6	RS	2	10	10	2016
7	RJ	8	2	10	2016
8	MT	3	10	10	2016
9	GO	3	1	10	2016
10	MG	13	1	10	2016

customer_state

Insight: we have find number of order placed on the basis of instalments with respect to the each month over the past years. In oct 2016 more order get placed on the instalment basis.