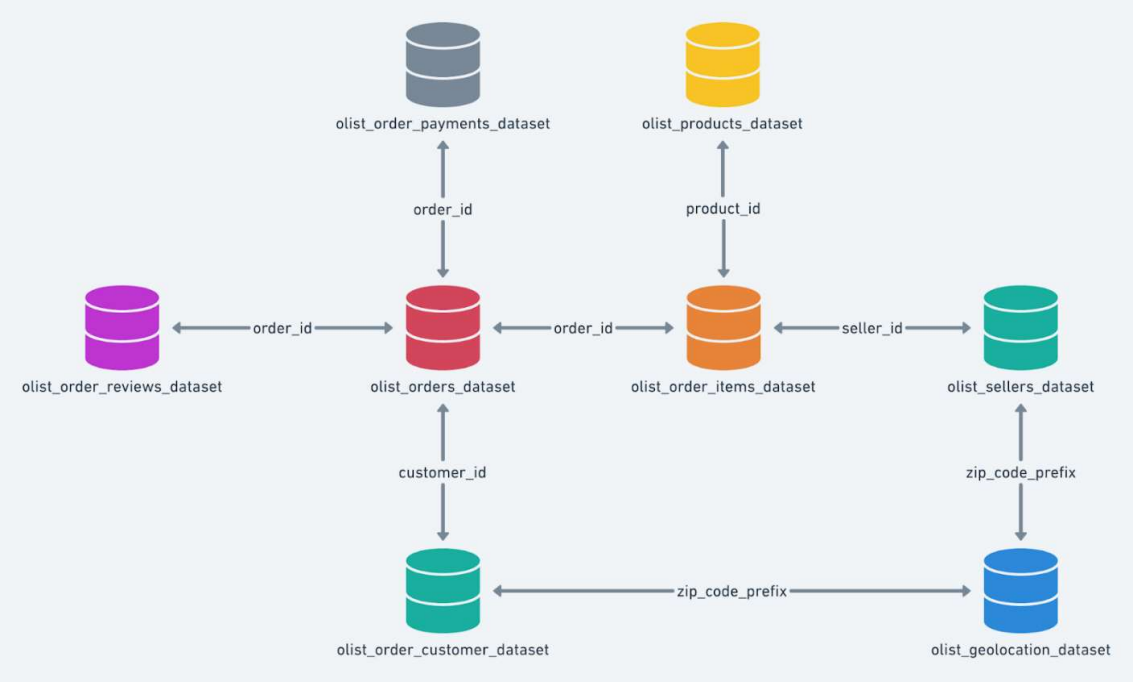
Business Case: Target SQL

Context:

Target is a globally renowned brand and a prominent retailer in the United States. 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 particular business case focuses on the operations of Target in Brazil and provides insightful information about 100,000 orders placed between 2016 and 2018. The dataset offers a comprehensive view of various dimensions including the order status, price, payment and freight performance, customer location, product attributes, and customer reviews.

By analyzing this extensive dataset, it becomes possible to gain valuable insights into Target's operations in Brazil. The information can shed light on various aspects of the business, such as order processing, pricing strategies, payment and shipping efficiency, customer demographics, product characteristics, and customer satisfaction levels.

Data Schema:



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

1. Data type of all columns in the "customers" table.

SQL Query:

```
SELECT
  column_name,
  data_type
FROM
  `may23-386911.target_sql.INFORMATION_SCHEMA.COLUMNS`
WHERE
  table_name = 'customers'
```

Query results

JOB INFORMATION | RESULTS | JSON | EXECUTION DETAILS

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

Insights:
Table named “customers” include information regarding customer id, customer unique id, customer zip codes, customer state and city.

2. Get the time range between which the orders were placed.

SQL Query:

```
SELECT
MIN(order_purchase_timestamp) AS min_time_range,
MAX(order_purchase_timestamp) AS max_time_range
FROM `target_sql.orders`
```

Query results

JOB INFORMATION | RESULTS | JSON | EXECUTION DETAILS

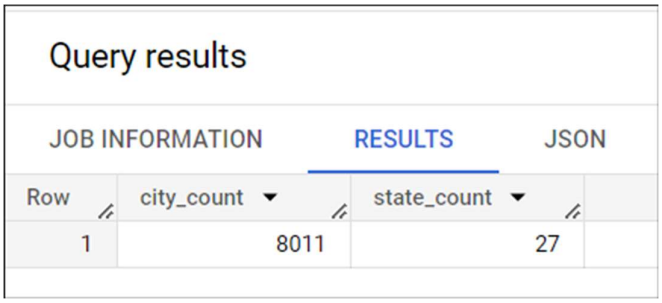
Row	min_time_range	max_time_range
1	2016-09-04 21:15:19 UTC	2018-10-17 17:30:18 UTC

Insights: NA

3. Count the number of Cities and States in our dataset.

SQL Query:

```
SELECT COUNT(DISTINCT geolocation_city) AS city_count,
COUNT(DISTINCT geolocation_state) AS state_count
FROM `target_sql.geolocation`
```



I have geolocation table to get the count of cities and states in all of dataset.

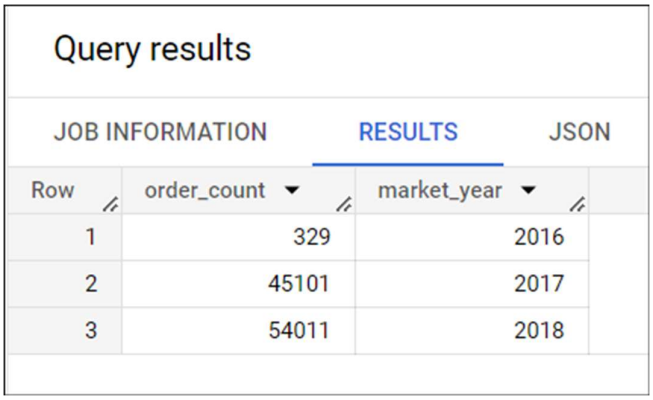
Insights:
There are a total of 8011 cities and 27 states.

Q.2: In-depth Exploration:

1. Is there a growing trend in the no. of orders placed over the past years?

SQL Query:

```
SELECT COUNT(order_id) AS order_count,
EXTRACT(year from order_purchase_timestamp) AS market_year,
FROM `target_sql.orders`
GROUP BY market_year
ORDER BY market_year ASC
```



Insights:

In 2016 the no. of orders were less but later on in 2017 and 2018 Target grew its order count and became successful in catering to customer's needs.

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

SQL Query:

```
SELECT COUNT(*) AS order_count,
EXTRACT(month FROM order_purchase_timestamp) AS month
FROM `target_sql.orders`
GROUP BY month
ORDER BY month
```

Query results

JOB INFORMATION | RESULTS | JSON

Row	order_count	month
1	8069	1
2	8508	2
3	9893	3
4	9343	4
5	10573	5
6	9412	6
7	10318	7
8	10843	8
9	4305	9
10	4959	10
11	7544	11
12	5674	12

Insights

Target made most of its orders in the months of May, July & August.
Months September, October & December were low on orders in comparison.

Recommendation:

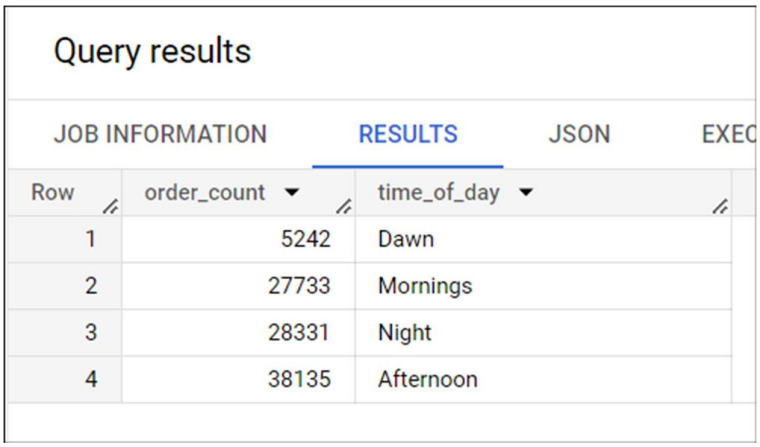
Target can add discounts offers and flash sale in the months having low order count to boost customer purchases in order to increase the order count.

3. During what time of the day, do the Brazilian customers mostly place their orders? (Dawn, Morning, Afternoon or Night)
    i. 0-6 hrs : Dawn
    ii. 7-12 hrs : Mornings
    iii. 13-18 hrs : Afternoon
    iv. 19-23 hrs : Night

SQL Query:

```sql
WITH table1 AS
(SELECT order_id,
EXTRACT(hour FROM order_purchase_timestamp) AS hour_cal,
CASE WHEN EXTRACT(hour FROM order_purchase_timestamp) BETWEEN 0 AND 6 THEN
"Dawn"
WHEN EXTRACT(hour FROM order_purchase_timestamp) BETWEEN 7 AND 12 THEN
"Mornings"
WHEN EXTRACT(hour FROM order_purchase_timestamp) BETWEEN 13 AND 18 THEN
"Afternoon"
ELSE "Night" END AS time_of_day
FROM `target_sql.orders`)

SELECT COUNT(distinct order_id) AS order_count, time_of_day
FROM table1
GROUP BY time_of_day
ORDER BY order_count ASC
```



Row	order_count	time_of_day
1	5242	Dawn
2	27733	Mornings
3	28331	Night
4	38135	Afternoon

Insights
Maximum orders are in Afternoon and minimum orders are in Dawn as its sleeping period in Brazil.

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

1. Get the month on month no. of orders placed in each state.

SQL Query:

```
SELECT DISTINCT c.customer_state,
COUNT(o.order_id) AS order_count,
EXTRACT(month FROM o.order_purchase_timestamp) as month
FROM `target_sql.orders` AS o JOIN `target_sql.customers` AS c
ON o.customer_id = c.customer_id
GROUP BY month, c.customer_state
ORDER BY c.customer_state ASC, month ASC
```

Query results

JOB INFORMATION | RESULTS | JSON | EXECUTION DETAILS

Row	customer_state	order_count	month
1	AC	8	1
2	AC	6	2
3	AC	4	3
4	AC	9	4
5	AC	10	5
6	AC	7	6
7	AC	9	7
8	AC	7	8
9	AC	5	9
10	AC	6	10
11	AC	5	11

Insights
Some states show low order count on monthly basis which shows the customers are not buying Most of their goods from Target
Recommendation
Target needs to increase their promotion campaign, sales pitch & offers in the states having low order count to boost customer acquisition.

2. How are the customers distributed across all the states?

SQL Query:

```
SELECT DISTINCT customer_state,
COUNT (DISTINCT customer_id) AS customer_count
FROM `target_sql.customers`
GROUP BY customer_state
ORDER BY customer_state ASC
```

Query results

JOB INFORMATION | RESULTS | JSON | EXEC

Row	customer_state	customer_count
1	AC	81
2	AL	413
3	AM	148
4	AP	68
5	BA	3380
6	CE	1336
7	DF	2140
8	ES	2033
9	GO	2020
10	MA	747
11	MG	11635

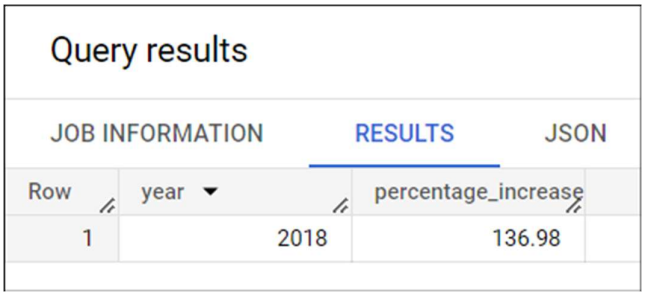
Insights
RR state has highest customer count and SP state has lowest customer count.
Recommendation
Target can increase its service and discount on products and come up with month ending sales to boost customer acquisition in the states having low customer count.

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

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

SQL Query:

```
SELECT
table2.year,
ROUND(((amount / lead(table2.amount) OVER(ORDER BY table2.year DESC))-
1)*100,2) AS percentage_increase
FROM
(SELECT
table1.year AS year,
SUM(table1.total_amount) AS amount
FROM
(SELECT
EXTRACT(year FROM o.order_purchase_timestamp) AS year,
EXTRACT(month FROM o.order_purchase_timestamp) AS month,
SUM(payment_value) AS total_amount
FROM `target_sql.orders` AS o JOIN `target_sql.payments` AS p
ON o.order_id=p.order_id
GROUP BY year, month
HAVING year IN (2017, 2018) AND (month BETWEEN 1 AND 8)) AS table1
GROUP BY year) AS table2
ORDER BY year DESC
LIMIT 1
```



Insights
The growth rate for Target looks good and promising.

2. Calculate the Total & Average value of order price for each state.

SQL Query:

```
SELECT s.seller_state,
ROUND(SUM(o.price),2) AS total_price,
ROUND(AVG(o.price),2) AS avg_price,
FROM `target_sql.order_items` AS o JOIN `target_sql.sellers` AS s
ON o.seller_id = s.seller_id
GROUP BY s.seller_state
```

Query results

JOB INFORMATION | RESULTS | JSON | EXECUTION DETAILS

Row	seller_state	total_price	avg_price
1	SP	8753396.21	108.95
2	MG	1011564.74	114.6
3	PR	1261887.21	145.53
4	SC	632426.07	155.2
5	RS	378559.54	172.15
6	DF	97749.48	108.73
7	ES	47689.61	128.2
8	RJ	843984.22	175.17
9	GO	66399.21	127.69
10	PA	1238.0	154.75

Insights
State wise price bifurcation can be seen by total and average price

3. Calculate the Total & Average value of order freight for each state.

SQL Query:

```
SELECT s.seller_state,
ROUND(SUM(o.freight_value),2) AS total_freight_price,
ROUND(AVG(o.freight_value),2) AS avg_freight_price,
FROM `target_sql.order_items` AS o JOIN `target_sql.sellers` AS s
ON o.seller_id = s.seller_id
GROUP BY s.seller_state
```

Query results

JOB INFORMATION | RESULTS | JSON | EXECUTION DETAILS

Row	seller_state	total_freight_price	avg_freight_price
1	SP	1482487.67	18.45
2	MG	212595.06	24.08
3	PR	197013.52	22.72
4	SC	106547.06	26.15
5	RS	57243.09	26.03
6	DF	18494.06	20.57
7	ES	12171.13	32.72
8	RJ	93829.9	19.47
9	GO	12565.5	24.16
10	PA	155.11	19.39

Insights
Freight charges can be displayed state-wise
Recommendation
Company should think about reducing the cost of freight charges in the states where the prices are high currently. They can negotiate with their delivery partners to reduce the cost.

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

1. 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.

   You can calculate the delivery time and the difference between the estimated & actual delivery date using the given formula:

- **time_to_deliver** = order_delivered_customer_date - order_purchase_timestamp
- **diff_estimated_delivery** = order_estimated_delivery_date - order_delivered_customer_date

SQL Query:

```
SELECT order_id,
TIMESTAMP_DIFF(order_delivered_customer_date,order_purchase_timestamp,day)
AS time_to_deliver,
TIMESTAMP_DIFF(order_estimated_delivery_date,order_delivered_customer_date,d
ay) AS diff_estimated_delivery,
FROM `target_sql.orders`
```

JOB INFORMATION | RESULTS | JSON | EXECUTION DETAILS

Row	order_id	time_to_deliver	diff_estimated_delive
1	1950d777989f6a877539f5379...	30	-12
2	2c45c33d2f9cb8ff8b1c86cc28...	30	28
3	65d1e226dfaeb8cdc42f66542...	35	16
4	635c894d068ac37e6e03dc54e...	30	1
5	3b97562c3aee8bdedcb5c2e45...	32	0
6	68f47f50f04c4cb6774570cfde...	29	1
7	276e9ec344d3bf029ff83a161c...	43	-4
8	54e1a3c2b97fb0809da548a59...	40	-4
9	fd04fa4105ee8045f6a0139ca5...	37	-1
10	302bb8109d097a9fc6e9cefc5...	33	-5

Insights
Negative integer indicates the number of days taken after the actual delivery date to deliver the product to customers.

Recommendation
Target should work with their supply chain and with their delivery partners to improve the delivery time in order to retain customers.

2. Find out the top 5 states with the highest & lowest average freight value.

SQL Query:

```
(SELECT s.seller_state,
ROUND(AVG(freight_value)) as avg_value, 'high_avg_value' as category,
FROM `target_sql.order_items` as o JOIN `target_sql.sellers` as s
ON o.seller_id = s.seller_id
GROUP BY s.seller_state
ORDER BY avg_value DESC
LIMIT 5)
UNION ALL
(SELECT s.seller_state,
ROUND(AVG(freight_value)) as avg_value, 'low_avg_value' as category
FROM `target_sql.order_items` as o JOIN `target_sql.sellers` as s
ON o.seller_id = s.seller_id
GROUP BY s.seller_state
ORDER BY avg_value ASC
LIMIT 5)
```

Query results

JOB INFORMATION | RESULTS | JSON | EXECUTION DETAILS | EXEC

Row	seller_state	avg_value	category
1	RO	51.0	high_avg_value
2	CE	46.0	high_avg_value
3	PB	39.0	high_avg_value
4	PI	37.0	high_avg_value
5	ES	33.0	high_avg_value
6	SP	18.0	low_avg_value
7	PA	19.0	low_avg_value
8	RJ	19.0	low_avg_value
9	DF	21.0	low_avg_value
10	PR	23.0	low_avg_value

Insights
States displaying highest & lowest average freight charges.

Recommendation
Company needs to reduce the freight difference between highest freight vale and lowest freight value.
They need to gather more data to analyse why the freight charges are high in RO, CE, PB, PI & ES states and how they can reduce them to cost benefit the company in long run.

3. Find out the top 5 states with the highest & lowest average delivery time.

SQL Query:

```
(SELECT c.customer_state,
ROUND(AVG(TIMESTAMP_DIFF(order_delivered_customer_date,order_purchase_timest
amp, day)), 2) avg_time_to_deliver, 'highest_avg_time' AS category,
FROM `target_sql.orders` AS o JOIN `target_sql.customers` AS c
ON o.customer_id = c.customer_id
GROUP BY c.customer_state
ORDER BY avg_time_to_deliver DESC
LIMIT 5)
UNION ALL
(SELECT c.customer_state,
ROUND(AVG(TIMESTAMP_DIFF(order_delivered_customer_date,order_purchase_timest
amp, day)), 2) avg_time_to_deliver, 'lowest_avg_time' AS category,
FROM `target_sql.orders` AS o JOIN `target_sql.customers` AS c
ON o.customer_id = c.customer_id
GROUP BY c.customer_state
ORDER BY avg_time_to_deliver ASC
LIMIT 5)
```

Query results

JOB INFORMATION | RESULTS | JSON | EXECUTION DETAILS | EXECUT

Row	customer_state	avg_time_to_deliver	category
1	RR	28.98	highest_avg_time
2	AP	26.73	highest_avg_time
3	AM	25.99	highest_avg_time
4	AL	24.04	highest_avg_time
5	PA	23.32	highest_avg_time
6	SP	8.3	lowest_avg_time
7	PR	11.53	lowest_avg_time
8	MG	11.54	lowest_avg_time
9	DF	12.51	lowest_avg_time
10	SC	14.48	lowest_avg_time

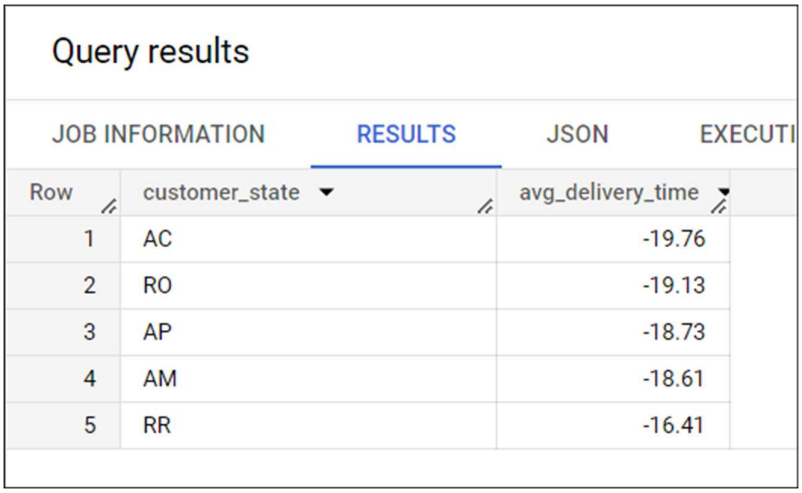
Insights
Highest average delivery time is approx. more than 22 days

Recommendation
Company should work with delivery partners to reduce the delivery time for the states showing high average time to deliver.

4. 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.

SQL Query:

```
SELECT c.customer_state,
ROUND(AVG(TIMESTAMP_DIFF(order_delivered_customer_date,order_estimated_deliver
y_date,day)),2) AS avg_delivery_time
FROM `target_sql.orders` AS o JOIN `target_sql.customers` AS c
ON o.customer_id = c.customer_id
GROUP BY c.customer_state
ORDER BY avg_delivery_time ASC
LIMIT 5
```



Query results

JOB INFORMATION | RESULTS | JSON | EXECUTI

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

Insights
Top 5 states having lowest delivery time.

Q.6: Analysis based on the payments:

1. Find the month on month no. of orders placed using different payment types.

SQL Query:

```
SELECT p.payment_type,
EXTRACT(month FROM o.order_purchase_timestamp) AS months,
COUNT(p.order_id) AS order_count
FROM `target_sql.payments` AS p JOIN `target_sql.orders` AS o
ON p.order_id = o.order_id
GROUP BY p.payment_type, months
ORDER BY p.payment_type, months
```

Query results

JOB INFORMATION | RESULTS | JSON | EXECUTION DETAILS

Row	payment_type	months	order_count
1	UPI	1	1715
2	UPI	2	1723
3	UPI	3	1942
4	UPI	4	1783
5	UPI	5	2035
6	UPI	6	1807
7	UPI	7	2074
8	UPI	8	2077
9	UPI	9	903
10	UPI	10	1056
11	UPI	11	1509
12	UPI	12	1160

2. Find the no. of orders placed on the basis of the payment installments that have been paid.

SQL Query:

```
WITH table1 AS
(SELECT
payment_installments, payment_sequential,
COUNT (order_id) AS order_count
FROM `target_sql.payments`
GROUP BY payment_installments, payment_sequential)
SELECT SUM(order_count) AS paid_orders
FROM table1
WHERE payment_installments = payment_sequential
```

Query results

JOB INFORMATION | RESULTS

Row	paid_orders
1	48290