# SCALER DSML ASSIGNMENT

STUDENT NAME MANOJ BHAT
BUSINESS CASE - TARGET SQL
BATCH - DSML JULY 2023

---

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

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

```
SELECT
 column_name,
 data_type
FROM
 Target.INFORMATION_SCHEMA.COLUMNS
WHERE
 table_name = 'customers';
```

Query results

SAVE RESULTS EXPLORE DATA

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

| 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 |

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

```
SELECT
 MIN(order_purchase_timestamp) AS First_Order,
 MAX(order_purchase_timestamp) AS Last_Order
FROM
 `Target.orders`;
```

Query results

SAVE RESULTS EXPLORE DATA

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

| Row | First_Order | Last_Order |
|---|---|---|
| 1 | 2016-09-04 21:15:19 UTC | 2018-10-17 17:30:18 UTC |

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

```
SELECT
 COUNT (DISTINCT geolocation_city) AS Number_of_Distinct_Cities,
 COUNT (DISTINCT geolocation_state) AS Number_of_Distinct_States
FROM
 `Target.geolocation`;
```

Query results

SAVE RESULTS EXPLORE DATA

JOB INFORMATION RESULTS JSON EXECUTION DETAILS CHART PREVIEW EXECUTION GRAPH

| Row | Number_of_Distinct_ | Number_of_Distinct_ |
|---|---|---|
| 1 | 8011 | 27 |

## 2. In-depth Exploration:

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

```
SELECT
 COUNT(order_id) Number_Of_Orders,
 EXTRACT(Month FROM order_purchase_timestamp) AS Month,
 EXTRACT(YEAR FROM order_purchase_timestamp) AS Year
 FROM
 `Target.orders`
GROUP BY
 Year,
 Month
ORDER BY
 Year,
 Month;
```

# SCALER DSML ASSIGNMENT

STUDENT NAME MANOJ BHAT
BUSINESS CASE - TARGET SQL
BATCH - DSML JULY 2023

---

Query results

SAVE RESULTS EXPLORE DATA

JOB INFORMATION RESULTS JSON EXECUTION DETAILS CHART PREVIEW EXECUTION GRAPH

| Row | Number_Of_Orders | Month | Year |
|---|---|---|---|
| 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 |

Results per page: 50 1 – 25 of 25

PERSONAL HISTORY PROJECT HISTORY REFRESH

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

```
SELECT
 COUNT(order_id) Number_Of_Orders,
 EXTRACT(Month FROM order_purchase_timestamp) AS Month,
 EXTRACT(YEAR FROM order_purchase_timestamp) AS Year
 FROM
 `Target.orders`
GROUP BY
 Year,
 Month
ORDER BY
 Year,
 Month;
```

***Insight*** - With the given data we can derive from the query results that there is no such seasonality every year. However for year 2017 it was revealed that orders from month 11 is at Peak

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

```sql
SELECT
 COUNT(order_id) Number_Of_Orders,
 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'
   WHEN EXTRACT(HOUR FROM order_purchase_timestamp) BETWEEN 19 AND 23 THEN 'Night'
END
 AS Time_Of_the_Day
FROM
 `Target.orders`
GROUP BY
 Time_Of_the_Day
ORDER BY
 Number_Of_Orders;
```

Query results

SAVE RESULTS EXPLORE DATA

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

| Row | Number_Of_Orders | Time_Of_the_Day |
|---|---|---|
| 1 | 5242 | Dawn |
| 2 | 27733 | Mornings |
| 3 | 28331 | Night |
| 4 | 38135 | Afternoon |

***Insight*** - The highest orders are placed in the afternoon.

3. Evolution of E-commerce orders in the Brazil region:
    a. Get the month on month no. of orders placed in each state. by our customers.

```sql
SELECT
 COUNT(o.order_id) Number_Of_Orders,
 c.customer_state AS State,
 EXTRACT(Month FROM o.order_purchase_timestamp ) AS Month,
 EXTRACT(Year FROM o.order_purchase_timestamp ) AS Year
FROM
 `Target.orders` o
LEFT JOIN
 `Target.customers` c
ON
 c.customer_id = o.customer_id
GROUP BY
 Month,
 Year,
 c.customer_state
ORDER BY
 c.customer_state,
 Year,
 Month;
```

Query results

SAVE RESULTS EXPLORE DATA

JOB INFORMATION RESULTS JSON EXECUTION DETAILS CHART PREVIEW EXECUTION GRAPH

| Row | Number_Of_Orders | State | Month | Year |
|---|---|---|---|---|
| 1 | 2 | AC | 1 | 2017 |
| 2 | 3 | AC | 2 | 2017 |
| 3 | 2 | AC | 3 | 2017 |
| 4 | 5 | AC | 4 | 2017 |
| 5 | 8 | AC | 5 | 2017 |
| 6 | 4 | AC | 6 | 2017 |
| 7 | 5 | AC | 7 | 2017 |
| 8 | 4 | AC | 8 | 2017 |
| 9 | 5 | AC | 9 | 2017 |
| 10 | 6 | AC | 10 | 2017 |
| 11 | 5 | AC | 11 | 2017 |

Results per page: 50 1 – 50 of 565

PERSONAL HISTORY PROJECT HISTORY REFRESH

***Insight*** - There is a considerable and significant growth in the no.of placed orders from 2017 onwards. This may be due to major marketing activities and campaigns however we need more data to conclude.

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

```
SELECT
 COUNT (DISTINCT(customer_unique_id)) AS Number_of_Customers,
 customer_state AS States
FROM
 `Target.customers`
GROUP BY
 Customer_state;
```

Query results

SAVE RESULTS EXPLORE DATA

JOB INFORMATION RESULTS JSON EXECUTION DETAILS CHART PREVIEW EXECUTION GRAPH

| Row | Number_of_Custome | States |
|---|---|---|
| 1 | 45 | RR |
| 2 | 67 | AP |
| 3 | 77 | AC |
| 4 | 143 | AM |
| 5 | 240 | RO |
| 6 | 273 | TO |
| 7 | 342 | SE |
| 8 | 401 | AL |
| 9 | 474 | RN |
| 10 | 482 | PI |
| 11 | 519 | PB |

Results per page: 50 1 – 27 of 27

PERSONAL HISTORY PROJECT HISTORY REFRESH

***Insight*** - The state RJ has the the most no.of Customers in the list and state RR has the least no.of Customers

4. Impact on Economy: Analyze the money movement by e-commerce by looking at order prices, freight and others
    a. Get the % increase in the cost of orders from year 2017 to 2018 (include months between Jan to Aug only).

```sql
SELECT
 Total_Value_of_Orders,
 Year,
 ROUND((((Total_Value_of_Orders - LAG(Total_Value_of_Orders) OVER(ORDER BY
Year))/LAG(Total_Value_of_Orders) OVER(ORDER BY Year)) *100),2) AS
Percentage_Growth
FROM (
 SELECT
   ROUND(SUM(payment_value),2) AS Total_Value_of_Orders,
   EXTRACT(Year
   FROM
     o.order_purchase_timestamp) AS Year,
 FROM
   `Target.orders` o
 LEFT JOIN
   `Target.payments` p
 ON
   p.order_id = o.order_id
 WHERE
   EXTRACT(MONTH FROM o.order_purchase_timestamp ) BETWEEN 1 AND 8
 GROUP BY
   Year
 ORDER BY
   Year )
ORDER BY
 Year;
```

# SCALER DSML ASSIGNMENT

STUDENT NAME MANOJ BHAT
BUSINESS CASE - TARGET SQL
BATCH - DSML JULY 2023

---

Query results

SAVE RESULTS EXPLORE DATA

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

| Row | Total_Value_of_Orde | Year | Percentage_Growth |
|---|---|---|---|
| 1 | 3669022.12 | 2017 | null |
| 2 | 8694733.84 | 2018 | 136.98 |

PERSONAL HISTORY PROJECT HISTORY REFRESH

***Insight*** - There is a considerable and significant growth in the no.of placed orders from 2017 onwards. This maybe due to major marketing activities and campaigns however we need more data to conclude

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

```
SELECT
 AVG(payment_value) AS Average_Value_of_Orders,
 SUM(payment_value) AS Total_Value_of_Orders,
 c.customer_state AS State
FROM
 `Target.payments` p
LEFT JOIN
 `Target.orders` o
ON
 o.order_id = p.order_id
LEFT JOIN
 `Target.customers` c
ON
 c.customer_id = o.customer_id
GROUP BY
 State
ORDER BY
 Total_Value_of_Orders,
 Average_Value_of_Orders;
```

# SCALER DSML ASSIGNMENT

STUDENT NAME MANOJ BHAT

BUSINESS CASE - TARGET SQL

BATCH - DSML JULY 2023

---

Query results

SAVE RESULTS EXPLORE DATA

JOB INFORMATION RESULTS JSON EXECUTION DETAILS CHART PREVIEW EXECUTION GRAPH

| Row | Average_Value_of_O | Total_Value_of_Orde | State |
|---|---|---|---|
| 1 | 218.7960869565... | 10064.61999999... | RR |
| 2 | 232.3257142857... | 16262.80000000... | AP |
| 3 | 234.2930952380... | 19680.62000000... | AC |
| 4 | 181.6034415584... | 27966.93 | AM |
| 5 | 233.2038314176... | 60866.19999999... | RO |
| 6 | 204.2701993355... | 61485.33000000... | TO |
| 7 | 208.4383656509... | 75246.25000000... | SE |
| 8 | 227.0774238875... | 96962.06 | AL |
| 9 | 196.7780268199... | 102718.1299999... | RN |
| 10 | 207.1068129770... | 108523.9699999... | PI |
| 11 | 186.8679891304 | 137534.84 | MS |

Results per page: 50 1 – 27 of 27

PERSONAL HISTORY PROJECT HISTORY REFRESH

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

```sql
SELECT
 AVG(freight_value) AS Avg_Freight_Value_of_Orders,
 SUM(freight_value) AS Total_Freight_Value_of_Orders,
 c.customer_state AS State
FROM
 `Target.order_items` oi
LEFT JOIN
 `Target.orders` o
ON
 o.order_id = oi.order_id
LEFT JOIN
 `Target.customers` c
ON
 c.customer_id = o.customer_id
GROUP BY
 State;
```

# SCALER DSML ASSIGNMENT

STUDENT NAME MANOJ BHAT
BUSINESS CASE - TARGET SQL
BATCH - DSML JULY 2023

---

Press Alt+F1 for Accessibility Options.

Query results

SAVE RESULTS | EXPLORE DATA

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

| Row | Avg_Freight_Value_o | Total_Freight_Value_ | State |
|---|---|---|---|
| 1 | 15.14727539041... | 718723.0699999... | SP |
| 2 | 20.96092393168... | 305589.3100000... | RJ |
| 3 | 20.53165156794... | 117851.6800000... | PR |
| 4 | 21.47036877394... | 89660.26000000... | SC |
| 5 | 21.04135494596... | 50625.49999999... | DF |
| 6 | 20.63016680630... | 270853.4600000... | MG |
| 7 | 35.83268518518... | 38699.30000000... | PA |
| 8 | 26.36395893656... | 100156.6799999... | BA |
| 9 | 22.76681525932... | 53114.97999999... | GO |
| 10 | 21.73580433039... | 135522.7400000... | RS |
| 11 | 37.24660317460... | 11732.67999999... | TO |

Results per page: 50 | 1 – 27 of 27

PERSONAL HISTORY | PROJECT HISTORY | REFRESH

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.
      ●time_to_deliver = order_delivered_customer_date - order_purchase_timestamp
      ●diff_estimated_delivery = order_estimated_delivery_date - order_delivered_customer_date

```
SELECT
 order_id,
 DATE_DIFF(order_delivered_carrier_date, order_purchase_timestamp, day) AS
time_to_deliver,
 DATE_DIFF(order_estimated_delivery_date, order_delivered_customer_date, day) AS
diff_estimated_delivery
FROM
 `Target.orders`;
```

# SCALER DSML ASSIGNMENT

STUDENT NAME MANOJ BHAT

BUSINESS CASE - TARGET SQL

BATCH - DSML JULY 2023

---

Query results

SAVE RESULTS EXPLORE DATA

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

| Row | order_id | time_to_deliver | diff_estimated_deliv |
|---|---|---|---|
| 1 | 770d331c84e5b214bd9dc70a1... | 4 | 45 |
| 2 | 1950d777989f6a877539f5379... | 1 | -12 |
| 3 | 2c45c33d2f9cb8ff8b1c86cc28... | 4 | 28 |
| 4 | dabf2b0e35b423f94618bf965f... | 4 | 44 |
| 5 | 8beb59392e21af5eb9547ae1a... | 6 | 41 |
| 6 | b60b53ad0bb7dacacf2989fe2... | 1 | -5 |
| 7 | 276e9ec344d3bf029ff83a161c... | 16 | -4 |
| 8 | 1a0b31f08d0d7e87935b819ed... | 2 | 29 |
| 9 | cec8f5f7a13e5ab934a486ec9e... | 17 | 40 |
| 10 | 2d846c03073b1a424c1be1a77... | 4 | -7 |
| 11 | 54e1a3c2b97fb0809da548a59... | 0 | -4 |

Results per page: 50 1 – 50 of 99441

PERSONAL HISTORY PROJECT HISTORY REFRESH

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

```
WITH T1 AS(
SELECT
 ROW_NUMBER() OVER (ORDER BY AVG(freight_value)) as RN,
 AVG(freight_value) AS Average_Value_of_Orders,
 c.customer_state AS State
FROM
 `Target.order_items` oi
LEFT JOIN
 `Target.orders` o
ON
 o.order_id = oi.order_id
LEFT JOIN
 `Target.customers` c
ON
 c.customer_id = o.customer_id
GROUP BY
 State
ORDER BY
 Average_value_of_Orders)

SELECT T.*,
CASE
```

```
    WHEN RN BETWEEN 1 AND 5 THEN 'BOTTOM'
    ELSE 'TOP'
  END AS TopOrBottom
FROM T1 AS T
WHERE RN IN (1,2,3,4,5) OR RN>(SELECT COUNT(*) - 5 FROM T1);
```

Query results

SAVE RESULTS EXPLORE DATA

JOB INFORMATION RESULTS JSON EXECUTION DETAILS CHART PREVIEW EXECUTION GRAPH

| Row | RN | Average_Value_of_O | State | TopOrBottom |
|---|---|---|---|---|
| 1 | 1 | 15.14727539041... | SP | BOTTOM |
| 2 | 2 | 20.53165156794... | PR | BOTTOM |
| 3 | 3 | 20.63016680630... | MG | BOTTOM |
| 4 | 4 | 20.96092393168... | RJ | BOTTOM |
| 5 | 5 | 21.04135494596... | DF | BOTTOM |
| 6 | 23 | 39.14797047970... | PI | TOP |
| 7 | 24 | 40.07336956521... | AC | TOP |
| 8 | 25 | 41.06971223021... | RO | TOP |
| 9 | 26 | 42.72380398671... | PB | TOP |
| 10 | 27 | 42.98442307692... | RR | TOP |

PERSONAL HISTORY PROJECT HISTORY REFRESH

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

```
WITH T2 AS
(
SELECT
 ROW_NUMBER() OVER (ORDER BY ROUND (AVG(DATE_DIFF(o.order_delivered_carrier_date,
o.order_purchase_timestamp, day)), 2)) as RN,
 c.customer_state AS State,
 ROUND (AVG(DATE_DIFF(o.order_delivered_carrier_date, o.order_purchase_timestamp,
day)), 2) AS Avg_Delivery_Time
FROM
 `Target.orders` o
LEFT JOIN
 `Target.customers` c
ON
 c.customer_id = o.customer_id
GROUP BY
 State
ORDER BY
 Avg_Delivery_Time
)
```

# SCALER DSML ASSIGNMENT

STUDENT NAME MANOJ BHAT
BUSINESS CASE - TARGET SQL
BATCH - DSML JULY 2023

---

```sql
SELECT T.*,
CASE
   WHEN RN BETWEEN 1 AND 5 THEN 'LOW'
   ELSE 'HIGH'
 END AS TopOrBottom
FROM T2 AS T
WHERE RN IN (1,2,3,4,5) OR RN>(SELECT COUNT(*) - 5 FROM T2);
```

Query results

SAVE RESULTS | EXPLORE DATA

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

| Row | RN | State | Avg_Delivery_Time | TopOrBottom |
|---|---|---|---|---|
| 1 | 1 | AM | 2.31 | LOW |
| 2 | 2 | RO | 2.33 | LOW |
| 3 | 3 | SP | 2.68 | LOW |
| 4 | 4 | MT | 2.68 | LOW |
| 5 | 5 | GO | 2.68 | LOW |
| 6 | 23 | AP | 3.04 | HIGH |
| 7 | 24 | RN | 3.11 | HIGH |
| 8 | 25 | MA | 3.13 | HIGH |
| 9 | 26 | SE | 3.15 | HIGH |
| 10 | 27 | RR | 4.02 | HIGH |

PERSONAL HISTORY | PROJECT HISTORY | REFRESH

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

```sql
SELECT
 c.customer_state AS State,
 ROUND(AVG(DATE_DIFF(o.order_estimated_delivery_date,
          o.order_delivered_customer_date, day)), 2) AS Avg_Order_Delivery_Time
FROM
 `Target.orders` o
LEFT JOIN
 `Target.customers` c
ON
 c.customer_id = o.customer_id
WHERE o.order_status = 'delivered'
GROUP BY
```

```
 State
ORDER BY
 Avg_Order_Delivery_Time
LIMIT 5;
```

Query results

SAVE RESULTS EXPLORE DATA

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

| Row | State | Avg_Order_Delivery_` |
|---|---|---|
| 1 | AL | 7.95 |
| 2 | MA | 8.77 |
| 3 | SE | 9.17 |
| 4 | ES | 9.62 |
| 5 | BA | 9.93 |

PERSONAL HISTORY PROJECT HISTORY REFRESH

## 6. Analysis based on the payments:

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

```
SELECT
 COUNT(o.order_id) Number_Of_Orders,
 p.payment_type AS Payment_type,
 EXTRACT(Month
 FROM
   o.order_purchase_timestamp ) AS Month,
 EXTRACT(Year
 FROM
   o.order_purchase_timestamp ) AS Year
FROM
 `Target.orders` o
LEFT JOIN
 `Target.payments` p
ON
 p.order_id = o.order_id
GROUP BY
 Month,
 Year,
```

```
 Payment_type
ORDER BY
 Year,
 Month;
```

Query results

SAVE RESULTS EXPLORE DATA

JOB INFORMATION RESULTS JSON EXECUTION DETAILS CHART PREVIEW EXECUTION GRAPH

| Row | Number_Of_Orders | Payment_type | Month | Year |
|---|---|---|---|---|
| 1 | 3 | credit_card | 9 | 2016 |
| 2 | 1 | null | 9 | 2016 |
| 3 | 254 | credit_card | 10 | 2016 |
| 4 | 63 | UPI | 10 | 2016 |
| 5 | 23 | voucher | 10 | 2016 |
| 6 | 2 | debit_card | 10 | 2016 |
| 7 | 1 | credit_card | 12 | 2016 |
| 8 | 583 | credit_card | 1 | 2017 |
| 9 | 197 | UPI | 1 | 2017 |
| 10 | 61 | voucher | 1 | 2017 |
| 11 | 9 | debit_card | 1 | 2017 |

Results per page: 50 1 – 50 of 91

PERSONAL HISTORY PROJECT HISTORY REFRESH

***Insight*** - Most of the orders are placed via Credit cards

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

```
SELECT
 COUNT(o.order_id) Number_Of_Orders,
 p.payment_installments AS Payment_installments,
 EXTRACT(Month
 FROM
   o.order_purchase_timestamp ) AS Month,
 EXTRACT(Year
 FROM
   o.order_purchase_timestamp ) AS Year
FROM
 `Target.orders` o
LEFT JOIN
 `Target.payments` p
ON
 p.order_id = o.order_id
WHERE
 p.payment_installments <> 0
GROUP BY
```

```
  Month,
  Year,
  Payment_installments
ORDER BY
  Year,
  Month;
```

Query results

SAVE RESULTS EXPLORE DATA

JOB INFORMATION RESULTS JSON EXECUTION DETAILS CHART PREVIEW EXECUTION GRAPH

| Row | Number_Of_Orders | Payment_installment | Month | Year |
|---|---|---|---|---|
| 1 | 1 | 1 | 9 | 2016 |
| 2 | 1 | 3 | 9 | 2016 |
| 3 | 1 | 2 | 9 | 2016 |
| 4 | 144 | 1 | 10 | 2016 |
| 5 | 42 | 10 | 10 | 2016 |
| 6 | 30 | 2 | 10 | 2016 |
| 7 | 43 | 3 | 10 | 2016 |
| 8 | 26 | 4 | 10 | 2016 |
| 9 | 20 | 5 | 10 | 2016 |
| 10 | 18 | 6 | 10 | 2016 |
| 11 | 13 | 7 | 10 | 2016 |

Results per page: 50 1 – 50 of 322

PERSONAL HISTORY PROJECT HISTORY REFRESH