# TARGET
# BUSINESS CASE STUDY

```
-->> 1) Data type of columns in a table:
SELECT TABLE_NAME, COLUMN_NAME, DATA_TYPE FROM target.INFORMATION_SCHEMA.COLUMNS;
```

Query results

JOB INFORMATION | RESULTS | JSON | EXECUTION DETAILS

| Row | TABLE_NAME | COLUMN_NAME | DATA_TYPE |
|---|---|---|---|
| 1 | order_items | order_id | STRING |
| 2 | order_items | order_item_id | INT64 |
| 3 | order_items | product_id | STRING |
| 4 | order_items | seller_id | STRING |
| 5 | order_items | shipping_limit_date | TIMESTAMP |
| 6 | order_items | price | FLOAT64 |

```
-->> 2)TimePeriod for which the data is given
select min(order_purchase_timestamp) as start_time,
       max(order_estimated_delivery_date) as end_time
from `target.orders`;
```

Query results

JOB INFORMATION | RESULTS | JSON | EXECUTION DETAILS

| Row | start_time | end_time |
|---|---|---|
| 1 | 2016-09-04 21:15:19 UTC | 2018-11-12 00:00:00 UTC |

```
-->> 3)Cities and States covered in the dataset

select distinct customer_state , customer_city
from `target.customers`
group by 1 ,2
order by 1,2
```

## Query results

JOB INFORMATION | RESULTS | JSON | EXECUTION DET

| Row | customer_state | customer_city |
|---|---|---|
| 1 | AC | brasileia |
| 2 | AC | cruzeiro do sul |
| 3 | AC | epitaciolandia |
| 4 | AC | manoel urbano |
| 5 | AC | porto acre |
| 6 | AC | rio branco |

```
-->> 4) Is there a growing trend in e-commerce in Brazil? How can we describe a complete
scenario? Can we see some seasonality with peaks at specific months?
-- formula is ((current_order_count - prev_order_count)/prev_order_count)*100
--- for specfic month extract year , month from order table

SELECT *, ROUND(((orders_count - prev_order_count) / prev_order_count) * 100, 2) AS
order_growth_rate_percent
FROM (SELECT *, LAG(orders_count) OVER(ORDER BY YEAR, MONTH) AS prev_order_count FROM (SELECT
EXTRACT(YEAR FROM order_purchase_timestamp) AS YEAR, EXTRACT(MONTH FROM
order_purchase_timestamp) AS MONTH, COUNT(*) AS orders_count
FROM `target.orders` WHERE order_status = 'delivered' GROUP BY 1, 2 ORDER BY 1, 2) AS BASE1
ORDER BY YEAR, MONTH) AS BASE2;
```

## Query results

SAVE RESU

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

| ow | YEAR | MONTH | orders_count | prev_order_count | order_growth_rate_pe |
|---|---|---|---|---|---|
| 1 | 2016 | 9 | 1 | null | null |
| 2 | 2016 | 10 | 265 | 1 | 26400.0 |
| 3 | 2016 | 12 | 1 | 265 | -99.62 |
| 4 | 2017 | 1 | 750 | 1 | 74900.0 |
| 5 | 2017 | 2 | 1653 | 750 | 120.4 |
| 6 | 2017 | 3 | 2546 | 1653 | 54.02 |

```
-->> 5) What time do Brazilian customers tend to buy (Dawn, Morning, Afternoon or Night)?
-- from order table column = order_purchase_timestamp
```

```sql
select  extract(time from order_purchase_timestamp) as timing
from `target.orders`

SELECT COUNTIF((TIME(order_purchase_timestamp) >= '05:00:00' AND
TIME(order_purchase_timestamp) < '06:00:00')) AS dawn_orders_count_5am_6am,
      COUNTIF((TIME(order_purchase_timestamp) >= '06:00:00' AND TIME(order_purchase_timestamp)
< '12:00:00')) AS morning_orders_count_6am_12pm,
      COUNTIF((TIME(order_purchase_timestamp) >= '12:00:00' AND TIME(order_purchase_timestamp)
< '18:00:00')) AS afternoon_orders_count_12pm_6pm,
      COUNTIF((TIME(order_purchase_timestamp) >= '18:00:00' AND TIME(order_purchase_timestamp)
<= '23:59:59')  OR (TIME(order_purchase_timestamp) >= '00:00:00' AND
TIME(order_purchase_timestamp) < '05:00:00')) AS night_orders_count_12am_5am FROM
`target.orders`;
```

Query results

JOB INFORMATION | RESULTS | JSON | EXECUTION DETAILS | EXE

| Row | dawn_orders_count_ | morning_orders_cou | afternoon_orders_co | night_orders_count_1 |
|---|---|---|---|---|
| 1 | 188 | 22240 | 38361 | 38652 |

```sql
-->> 6)Get month-on-month orders by region, states

with cte2 as
           (SELECT *, LAG(orders_count) OVER(PARTITION BY customer_state, customer_city ORDER
BY YEAR, MONTH) AS prev_orders_count FROM (SELECT C.customer_state, C.customer_city,
EXTRACT(MONTH FROM order_purchase_timestamp) AS MONTH, EXTRACT(YEAR FROM
order_purchase_timestamp) AS YEAR, COUNT(*) AS orders_count FROM `target.customers` AS C
           JOIN
           (SELECT *, EXTRACT(MONTH FROM order_purchase_timestamp) AS MONTH, EXTRACT(YEAR
FROM order_purchase_timestamp) AS YEAR FROM `target.orders` WHERE order_status = 'delivered')
AS J2 ON C.customer_id = J2.customer_id GROUP BY 1, 2, 3, 4))

 SELECT *, ROUND(((orders_count - prev_orders_count) / prev_orders_count) * 100, 2) AS
orders_count_growth_rate
 FROM cte2
 order by YEAR DESC ,orders_count_growth_rate DESC ;
```

Query results

SAVE RESULTS | EXPLORE DATA

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

| ow | customer_state | customer_city | MONTH | YEAR | orders_count | prev_orders_count | orders_count_growth |
|---|---|---|---|---|---|---|---|
| 1 | RS | novo hamburgo | 1 | 2018 | 13 | 1 | 1200.0 |
| 2 | RS | viamao | 1 | 2018 | 13 | 1 | 1200.0 |
| 3 | PR | pinhais | 1 | 2018 | 10 | 1 | 900.0 |
| 4 | SP | ferraz de vasconcelos | 7 | 2018 | 10 | 1 | 900.0 |
| 5 | BA | itabuna | 7 | 2018 | 10 | 1 | 900.0 |
| 6 | BA | ilheus | 7 | 2018 | 9 | 1 | 800.0 |

```sql
--6.2) Distribution of customers across the states in Brazil

SELECT customer_state, customer_city ,
      COUNT(DISTINCT customer_id) AS count_customer_id,
      COUNT(DISTINCT customer_unique_id) AS count_customer_unique_id
FROM `target.customers`
GROUP BY 1, 2
ORDER BY 1, 2;
```

Query results

SAVE RESULT

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

| Row | customer_state | customer_city | count_customer_id | count_customer_unic |
|---|---|---|---|---|
| 1 | AC | brasileia | 1 | 1 |
| 2 | AC | cruzeiro do sul | 3 | 3 |
| 3 | AC | epitaciolandia | 1 | 1 |
| 4 | AC | manoel urbano | 1 | 1 |
| 5 | AC | porto acre | 1 | 1 |
| 6 | AC | rio branco | 70 | 66 |

```sql
--->>7) Get % increase in cost of orders from 2017 to 2018 (include months between Jan to Aug only)

WITH QUERY1 AS
(SELECT ROUND(SUM(price + freight_value), 2) AS total_cost_2017
FROM
      (SELECT O.*, OI.* FROM `target.orders` AS O JOIN `target.order_items` AS OI ON O.order_id = OI.order_id
      WHERE O.order_status = 'delivered' AND (EXTRACT(YEAR FROM O.order_purchase_timestamp) = 2017) AND EXTRACT(MONTH FROM O.order_purchase_timestamp) BETWEEN 1 AND 8)),
QUERY2 AS (SELECT ROUND(SUM(price + freight_value),2) AS total_cost_2018
FROM
      (SELECT O.*, OI.* FROM `target.orders` AS O JOIN `target.order_items` AS OI ON O.order_id = OI.order_id
      WHERE O.order_status = 'delivered' AND (EXTRACT(YEAR FROM O.order_purchase_timestamp) = 2018) AND EXTRACT(MONTH FROM O.order_purchase_timestamp) BETWEEN 1 AND 8))
SELECT Q1.total_cost_2017, Q2.total_cost_2018, ROUND(((Q2.total_cost_2018 - Q1.total_cost_2017) / Q1.total_cost_2017) * 100, 2) AS cost_growth_rate
FROM QUERY1 AS Q1 CROSS JOIN QUERY2 AS Q2;
```

JOB INFORMATION | RESULTS | JSON | EXECUTION DETAILS

| Row | total_cost_2017 | total_cost_2018 | cost_growth_rate |
|---|---|---|---|
| 1 | 3472898.25 | 8451584.77 | 143.36 |

```
-->> 8)Mean & Sum of price and freight value by customer state

SELECT C.customer_state, ROUND(AVG(OI.price + OI.freight_value), 2) AS avg_cost,
      ROUND(SUM(price + freight_value), 2) AS sum_cost
FROM `target.customers` AS C JOIN `target.orders` AS O ON C.customer_id = O.customer_id
                             JOIN `target.order_items` AS OI ON O.order_id = OI.order_id
WHERE O.order_status = 'delivered' GROUP BY 1;
```

Query results

JOB INFORMATION | RESULTS | JSON | EXECUTION DETAILS

| Row | customer_state | avg_cost | sum_cost |
|---|---|---|---|
| 1 | GO | 146.78 | 334212.35 |
| 2 | SP | 124.22 | 5769703.15 |
| 3 | RS | 140.44 | 861472.79 |
| 4 | BA | 160.5 | 591137.81 |
| 5 | MG | 140.82 | 1818891.67 |

```
-->>9) Calculate days between purchasing, delivering, and estimated delivery QUERY:

SELECT order_id, order_purchase_timestamp, order_delivered_customer_date,
order_estimated_delivery_date,
(TIMESTAMP_DIFF(order_delivered_customer_date,order_purchase_timestamp, DAY)) AS
time_to_delivery,
(TIMESTAMP_DIFF(order_estimated_delivery_date, order_purchase_timestamp, DAY)) AS
diff_estimated_delivery, (TIMESTAMP_DIFF(order_estimated_delivery_date,
order_delivered_customer_date, DAY)) AS diff_estim_date
FROM `target.orders`
WHERE order_status = 'delivered';
```

| time_to_delivery | diff_estimated_delive | diff_estim_date |
|---|---|---|
| 20 | 61 | 40 |
| 10 | 58 | 48 |
| 28 | 57 | 29 |
| 9 | 44 | 35 |
| 10 | 52 | 41 |

```
--10) Create columns: ♣ time_to_delivery = order_purchase_timestamp-
order_delivered_customer_date ♣                    diff_estimated_delivery =
order_estimated_delivery_date-order_delivered_customer_date
```

```sql
SELECT order_id, order_purchase_timestamp, order_delivered_customer_date,
order_estimated_delivery_date,
(TIMESTAMP_DIFF(order_delivered_customer_date,order_purchase_timestamp, DAY)) AS
time_to_delivery,
(TIMESTAMP_DIFF(order_estimated_delivery_date, order_delivered_customer_date, DAY)) AS
diff_estimated_delivery
FROM `target.orders`
WHERE order_status = 'delivered';

-->>11) Group data by state, take mean of freight_value, time_to_delivery,
diff_estimated_delivery

SELECT C.customer_state, AVG(OI.freight_value) AS avg_freight_value,
        AVG(Q1.time_to_delivery) AS avg_time_to_delivery,
        AVG(Q1.diff_estimated_delivery) AS  avg_diff_estimated_delivery
FROM `target.customers` AS C
JOIN
    (SELECT *, TIMESTAMP_DIFF(order_delivered_customer_date, order_purchase_timestamp, DAY) AS
time_to_delivery,
     TIMESTAMP_DIFF(order_estimated_delivery_date, order_purchase_timestamp, DAY) AS
diff_estimated_delivery,
    FROM `target.orders` WHERE order_status = 'delivered') AS Q1 ON C.customer_id =
Q1.customer_id
    JOIN `target.order_items` AS OI ON Q1.order_id = OI.order_id GROUP BY C.customer_state;
```

| | customer_state | avg_freight_value | avg_time_to_delivery | avg_diff_estimated_c |
|---|---|---|---|---|
| 1 | GO | 22.56 | 14.95 | 26.63 |
| 2 | SP | 15.12 | 8.26 | 18.87 |
| 3 | RS | 21.61 | 14.71 | 28.27 |
| 4 | BA | 26.49 | 18.77 | 29.18 |
| 5 | MG | 20.63 | 11.51 | 24.26 |
| 6 | MT | 28.0 | 17.51 | 31.48 |

```sql
-->>12) Top 5 states with highest avg_freight_value:


SELECT C.customer_state, ROUND(AVG(OI.freight_value),2) avg_freight_value , FROM
`target.customers` AS C
JOIN `target.orders` AS O ON C.customer_id = O.customer_id
JOIN `target.order_items` AS OI ON O.order_id = OI.order_id
WHERE O.order_status = 'delivered'
GROUP BY C.customer_state
ORDER BY ROUND(AVG(OI.freight_value),2) DESC LIMIT 5
```

| | customer_state | avg_freight_value |
|---|---|---|
| 1 | RR | 43.09 |
| 2 | PB | 43.09 |
| 3 | RO | 41.33 |
| 4 | AC | 40.05 |
| 5 | PI | 39.12 |

-->>13)Top 5 states with lowest avg_freight_value:

```
SELECT C.customer_state, ROUND(AVG(OI.freight_value),2) avg_freight_value , FROM
`target.customers` AS C
JOIN `target.orders` AS O ON C.customer_id = O.customer_id
JOIN `target.order_items` AS OI ON O.order_id = OI.order_id
WHERE O.order_status = 'delivered'
GROUP BY C.customer_state
ORDER BY ROUND(AVG(OI.freight_value),2) LIMIT 5
```

Query results

JOB INFORMATION | RESULTS | JSON | EXE

| Row | customer_state | avg_freight_value |
|---|---|---|
| 1 | SP | 15.12 |
| 2 | PR | 20.47 |
| 3 | MG | 20.63 |
| 4 | RJ | 20.91 |
| 5 | DF | 21.07 |

-->>14) top 5 states with the highest average time to delivery:

```
SELECT C.customer_state, ROUND(AVG(B.time_to_delivery), 2) AS avg_time_to_delivery
FROM `target.customers` AS C
JOIN
      (SELECT *, TIMESTAMP_DIFF(order_delivered_customer_date, order_purchase_timestamp, DAY)
AS time_to_delivery,
       FROM `target.orders` WHERE order_status = 'delivered') AS B ON C.customer_id =
B.customer_id
GROUP BY C.customer_state ORDER BY AVG(B.time_to_delivery) DESC limit 5;
```

Query results

JOB INFORMATION | RESULTS | JSON | EXEC

| Row | customer_state | avg_time_to_delivery |
|---|---|---|
| 1 | RR | 28.98 |
| 2 | AP | 26.73 |
| 3 | AM | 25.99 |
| 4 | AL | 24.04 |
| 5 | PA | 23.32 |

```
-->>15) top 5 states with the lowest average time to delivery:

SELECT C.customer_state, ROUND(AVG(B.time_to_delivery), 2) AS avg_time_to_delivery
FROM `target.customers` AS C
JOIN
    (SELECT *, TIMESTAMP_DIFF(order_delivered_customer_date, order_purchase_timestamp, DAY) AS
time_to_delivery
    FROM `target.orders` WHERE order_status = 'delivered') AS B ON C.customer_id =
B.customer_id
GROUP BY C.customer_state ORDER BY AVG(B.time_to_delivery) ASC LIMIT 5;
```

Query results

JOB INFORMATION | RESULTS | JSON | EXECU

| ow | customer_state | avg_time_to_delivery |
|---|---|---|
| 1 | SP | 8.3 |
| 2 | PR | 11.53 |
| 3 | MG | 11.54 |
| 4 | DF | 12.51 |
| 5 | SC | 14.48 |

```
-->> 16)top 5 states where delivery is really fast compared to the estimated date
```

```sql
SELECT C.customer_state, ROUND(AVG(B.diff_estim_date), 2) AS avg_diff_estim_date
FROM `target.customers` AS C
JOIN
(SELECT *, TIMESTAMP_DIFF(order_estimated_delivery_date, order_delivered_customer_date, DAY)
AS diff_estim_date FROM `target.orders`
WHERE order_status = 'delivered') AS B ON C.customer_id = B.customer_id
GROUP BY C.customer_state ORDER BY AVG(B.diff_estim_date) DESC LIMIT 5;
```

Query results

JOB INFORMATION | RESULTS | JSON | EXEC

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

```sql
-->>17)top 5 states where delivery is NOT SO FAST compared to the estimated date
```

```sql
SELECT C.customer_state, ROUND(AVG(B.diff_estim_date), 2) AS avg_diff_estim_date
FROM `target.customers` AS C
JOIN
(SELECT *, TIMESTAMP_DIFF(order_estimated_delivery_date, order_delivered_customer_date, DAY)
AS diff_estim_date FROM `target.orders`
WHERE order_status = 'delivered') AS B ON C.customer_id = B.customer_id
GROUP BY C.customer_state ORDER BY AVG(B.diff_estim_date) ASC LIMIT 5;
```

Query results

JOB INFORMATION | RESULTS | JSON | EXE

| Row | customer_state | avg_diff_estim_date |
|---|---|---|
| 1 | AL | 7.95 |
| 2 | MA | 8.77 |
| 3 | SE | 9.17 |
| 4 | ES | 9.62 |
| 5 | BA | 9.93 |

-->>18)Month over Month count of orders for different payment types

```
SELECT T2.*, ROUND(((T2.payment_type_count - T2.prev_count) / T2.prev_count) * 100, 2) AS
count_growth_rate_percent FROM (SELECT payment_type, YEAR, MONTH, payment_type_count,
LAG(payment_type_count) OVER(PARTITION BY payment_type ORDER BY YEAR, MONTH) AS prev_count
FROM (SELECT DISTINCT P.payment_type, O.YEAR, O.MONTH, COUNT(*) OVER (PARTITION BY
P.payment_type, O.YEAR, O.MONTH ORDER BY O.YEAR, O.MONTH) AS payment_type_count FROM
`target.payments` AS P JOIN (SELECT order_id, EXTRACT(MONTH FROM order_purchase_timestamp) AS
MONTH, EXTRACT(YEAR FROM order_purchase_timestamp) AS YEAR, FROM `target.orders` WHERE
order_status = 'delivered') AS O ON P.order_id = O.order_id) AS T1) AS T2
```

Query results

SAVE RESULTS | EXPLORE DATA

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

| Row | payment_type | YEAR | MONTH | payment_type_count | prev_count | count_growth_rate_p |
|---|---|---|---|---|---|---|
| 1 | debit_card | 2016 | 10 | 2 | null | null |
| 2 | debit_card | 2017 | 1 | 9 | 2 | 350.0 |
| 3 | debit_card | 2017 | 2 | 13 | 9 | 44.44 |
| 4 | debit_card | 2017 | 3 | 30 | 13 | 130.77 |
| 5 | debit_card | 2017 | 4 | 25 | 30 | -16.67 |
| 6 | debit_card | 2017 | 5 | 29 | 25 | 16.0 |

-->>19)Count of orders based on the no. of payment and payment type installments

```
SELECT P.payment_installments, COUNT(*) AS orders_count
FROM `target.payments` AS P
JOIN
(SELECT * FROM `target.orders` WHERE order_status = 'delivered') AS O ON P.order_id =
O.order_id
GROUP BY 1;
```

Query results

JOB INFORMATION | RESULTS | JSON | EXECUTION DETAILS

| Row | payment_installment | payment_type | orders_count |
|---|---|---|---|
| 1 | 0 | credit_card | 2 |
| 2 | 1 | voucher | 5493 |
| 3 | 1 | credit_card | 24759 |
| 4 | 1 | debit_card | 1486 |
| 5 | 1 | UPI | 19191 |
| 6 | 2 | credit_card | 12075 |