

# Business Case: Target SQL

This business case has data of 100k orders from 2016 to 2018 made at Target, Brazil. It is Americas leading retailer business chain.

Data is available in 8 tables, which gives information about orders from different dimensions like status of order, payment details, location and time of the order , customer who made the purchase, items in the order, product details, seller information of the products, order reviews etc.

## Analysis

### 1. Initial exploration of dataset


1.1 Show all the tables and all the columns present in each table along with its data type.

Query:

```
SELECT
    table_schema,
    table_name,
    column_name,
    data_type
FROM
    `target`.INFORMATION_SCHEMA.COLUMNS;
```

Result:

Query results

 SAVE

JOB INFORMATIONRESULTSJSONEXECUTION DETAILSEXECUTION GRAPHPREVIEW

Row	table_schema	table_name	column_name	data_type
1	target	order_items	order_id	STRING
2	target	order_items	order_item_id	INT64
3	target	order_items	product_id	STRING
4	target	order_items	seller_id	STRING
5	target	order_items	shipping_limit_date	TIMESTAMP
6	target	order_items	price	FLOAT64
7	target	order_items	freight_value	FLOAT64
8	target	sellers	seller_id	STRING
9	target	sellers	seller_zip_code_prefix	INT64
10	target	sellers	seller_city	STRING

## 1.2 For which time period the data is given.

Query:

```
SELECT
  MIN(DATE(order_purchase_timestamp)) AS first_date,
  MAX(DATE(order_purchase_timestamp)) AS last_date
FROM
  `target.orders`;
```

Result:

Query results			
JOB INFORMATION		RESULTS	JSON
Row	first_date ▼	last_date ▼	
1	2016-09-04	2018-10-17	

## 1.3 From which Cities and States , orders were placed during the given period.

Query:

```
SELECT
  DISTINCT c.customer_state , c.customer_city
FROM
  `target.customers` c
RIGHT JOIN
  `target.orders` o
USING
  (customer_id)
ORDER BY
  c.customer_state , c.customer_city;
```

Result:

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
7	AC	senador guiomard
8	AC	xapuri
9	AL	agua branca
10	AL	anadia

#### 1.4 What is distribution of total orders as per their status?

Query:

```
SELECT
  order_status,
  COUNT(*) AS orders_count
FROM
  `target.orders`
GROUP BY
  order_status
ORDER BY
  orders_count DESC;
```

Result:

##### Query results

JOB INFORMATION		RESULTS	JSON	EXI
Row	order_status ▼	orders_count ▼		
1	delivered	96478		
2	shipped	1107		
3	canceled	625		
4	unavailable	609		
5	invoiced	314		
6	processing	301		
7	created	5		
8	approved	2		

#### 1.5 How is the order value of each order is calculated in payment table, Is it addition of price and freight for each item in the order.

*(Used order items and payment table)*

```
SELECT
  order_id, total_price, total_freight,
  total_price + total_freight AS total_order_value
FROM ( SELECT
  order_id,
  SUM(price) AS total_price,
  SUM(freight_value) AS total_freight,
  COUNT(*) AS total_items
FROM
  `target.order_items`
```

```

WHERE
  order_id = "95d6357ffe41aa6d2998852a710c70a0"
GROUP BY
  order_id) t1;

```

Result:

Query results					
JOB INFORMATION		RESULTS	JSON	EXECUTION DETAILS	EXECUTION GRAPH <span>PREVIEW</span>
Row	order_id	total_price	total_freight	total_order_value	
1	95d6357ffe41aa6d2998852a710c70a0	17.5	91.15	108.65	

```

SELECT
  order_id, payment_value
FROM
  `target.payments`
WHERE
  order_id = "95d6357ffe41aa6d2998852a710c70a0";

```

Query results

JOB INFORMATION		RESULTS	JSON	EXECUTION DET
Row	order_id	payment_value		
1	95d6357ffe41aa6d2998852a710c70a0	108.65		

## 2. In-depth exploration of dataset

2.1 Is there a growing trend on e-commerce in Brazil? How can we describe a complete scenario?

Query:

```

SELECT
  time_period,
  order_count,

```

```

ROUND((((order_count - LAG(order_count) OVER(ORDER BY t1.YEAR, t1.month)) / LAG(order_count)
OVER(ORDER BY t1.YEAR, t1.month))* 100), 2) AS growth_percent
FROM (
SELECT
EXTRACT(YEAR
FROM
order_purchase_timestamp) AS year,
EXTRACT(MONTH
FROM
order_purchase_timestamp) AS month,
FORMAT_DATE('%b %Y', DATE(ORDER_PURCHASE_TIMESTAMP)) AS time_period,
COUNT(order_id) AS order_count
FROM
`target.orders`
WHERE
order_status = "delivered"
GROUP BY
month, year, time_period) t1
ORDER BY
year, month;

```

Result:

Query results				
JOB INFORMATION		RESULTS	JSON	EXECUTION DETAILS
Row	time_period	order_count	growth_percent	
1	Sep 2016	1	null	
2	Oct 2016	265	26400.0	
3	Dec 2016	1	-99.62	
4	Jan 2017	750	74900.0	
5	Feb 2017	1653	120.4	
6	Mar 2017	2546	54.02	
7	Apr 2017	2303	-9.54	
8	May 2017	3546	53.97	
9	Jun 2017	3135	-11.59	
10	Jul 2017	3872	23.51	

Can we see some seasonality with peaks at specific months?

No, as for some months data is showing orders of only 1 or 2 records and also there is huge spike in orders is seen in different months so we can't comment on seasonality.  
Overall business is in uptrend and sharp spike in orders is seen MoM basis.

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

Query:

```
SELECT
  temp.purchase_time,
  COUNT(*) AS total_orders
FROM (
  SELECT
    order_id,
    CASE
      WHEN TIME(order_purchase_timestamp) BETWEEN "00:00:00" AND "07:00:00" THEN "Dawn"
      WHEN TIME(order_purchase_timestamp) BETWEEN "07:00:01" AND "12:00:00" THEN "Morning"
      WHEN TIME(order_purchase_timestamp) BETWEEN "12:00:01" AND "18:00:00" THEN "Afternoon"
      WHEN TIME(order_purchase_timestamp) BETWEEN "18:00:01" AND "23:59:59" THEN "Night"
    END
    AS purchase_time
  FROM
    `target.orders`) TEMP
GROUP BY
  temp.purchase_time
ORDER BY
  total_orders DESC;
```

Result:

### Query results

JOB INFORMATION		RESULTS	JSON	EXECUTION
Row	purchase_time	total_orders		
1	Afternoon	38365		
2	Night	34096		
3	Morning	21738		
4	Dawn	5242		

**Brazilian customers usually tend to buy in afternoon and night.**

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

#### 3.1 Get month on month orders by states

Query:

```
SELECT
  state , time_period , total_orders,
  LAG(total_orders) OVER(PARTITION BY state ORDER BY year, month ) AS prev_month_orders_count,
  ROUND(((total_orders - LAG(total_orders) OVER(PARTITION BY state ORDER BY year, month )) /
LAG(total_orders) OVER(PARTITION BY state ORDER BY year, month))* 100,2) AS MoM_percent_growth
FROM (
  SELECT
    state,
    time_period,
    year,
    month,
    COUNT(*) AS total_orders
  FROM (
    SELECT
      o.order_id, o.order_purchase_timestamp,
      EXTRACT(YEAR FROM order_purchase_timestamp) AS year,
      EXTRACT(Month FROM order_purchase_timestamp) AS month,
      FORMAT_DATE('%b %Y', DATE(ORDER_PURCHASE_TIMESTAMP)) AS time_period,
      c.customer_state AS state
    FROM
      `target.orders` o
    JOIN
      `target.customers` c
    USING
      (customer_id)
    ORDER BY
      year, month) t1
  GROUP BY
    state, time_period,year, month) t2;
```

Query results



JOB INFORMATION		RESULTS	JSON	EXECUTION DETAILS	EXECUTION GRAPH	PREVIEW
Row	state	time_period	total_orders	prev_month_orders	MoM_percent_growth	
1	AL	Oct 2016	2	null	null	
2	AL	Jan 2017	2	2	0.0	
3	AL	Feb 2017	12	2	500.0	
4	AL	Mar 2017	10	12	-16.67	
5	AL	Apr 2017	23	10	130.0	
6	AL	May 2017	27	23	17.39	
7	AL	Jun 2017	10	27	-62.96	
8	AL	Jul 2017	17	10	70.0	
9	AL	Aug 2017	18	17	5.88	
10	AL	Sep 2017	20	18	11.11	

### 3.2 Distribution of customers across the states in Brazil

Query:

```
SELECT
  customer_state AS state,
  COUNT(*) AS total_customers
FROM
  `target.customers`
GROUP BY
  customer_state
ORDER BY
  total_customers DESC;
```

Result:

#### Query results

JOB INFORMATION		RESULTS	JSON	EXE
Row	state	total_customers		
1	SP	41746		
2	RJ	12852		
3	MG	11635		
4	RS	5466		
5	PR	5045		
6	SC	3637		
7	BA	3380		
8	DF	2140		
9	ES	2033		
10	GO	2020		

## 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

Query:

```
SELECT
  *,
  COALESCE((ROUND(((total_orders_value - LAG(total_orders_value) OVER(ORDER BY
year)))/LAG(total_orders_value) OVER(ORDER BY year))* 100, 2)), 0) AS percent_increase_YOY
FROM (
```



```

SELECT
  year,
  ROUND(SUM(payment_value), 2) AS total_orders_value
FROM (
  SELECT
    o.order_id,
    o.order_purchase_timestamp,
    EXTRACT(MONTH FROM o.order_purchase_timestamp) AS month,
    EXTRACT(YEAR FROM o.order_purchase_timestamp) AS year,
    p.payment_value
  FROM
    `target.orders` o
  JOIN
    `target.payments` p
  USING
    (order_id)
  WHERE
    o.order_status = "delivered")t1
WHERE
  month BETWEEN 1
  AND 8
GROUP BY
  year) t1
ORDER BY
  Year;

```

## Query results

JOB INFORMATION		RESULTS	JSON	EXECUTION DETAILS	
Row	year ▼	total_orders_value ▼	percent_increase_YOY ▼		
1	2017	3473862.76	0.0		
2	2018	8452975.2	143.33		

Comparison on Monthly total order value:

Query:

```

SELECT
  t2.Month_n_Year,
  t2.total_orders_value,
  ROUND(((t2.total_orders_value - LAG(total_orders_value) OVER(PARTITION BY month ORDER BY
year))/LAG(total_orders_value) OVER(PARTITION BY month ORDER BY year))*100, 2) AS
percent_increase
FROM (
  SELECT
    month,
    year,

```

```

Month_n_Year,
ROUND(SUM(payment_value)) AS total_orders_value
FROM (
SELECT
o.order_id,
o.order_purchase_timestamp,
EXTRACT(month FROM o.order_purchase_timestamp ) AS month,
EXTRACT(year FROM o.order_purchase_timestamp ) AS year,
FORMAT_DATE('%b %Y', DATE(ORDER_PURCHASE_TIMESTAMP)) AS Month_n_Year,
p.payment_value
FROM
`target.orders` o
JOIN
`target.payments` p
USING
(order_id)) t1
WHERE
month BETWEEN 1
AND 8
GROUP BY
month, year, Month_n_Year
ORDER BY
month, year) t2
ORDER BY
month;

```

## Query results

JOB INFORMATION		RESULTS	JSON	EXECUTION DETAILS	
Row	Month_n_Year	total_orders_value	percent_increase		
1	Jan 2017	138488.0	<i>null</i>		
2	Jan 2018	1115004.0	705.13		
3	Feb 2017	291908.0	<i>null</i>		
4	Feb 2018	992463.0	239.99		
5	Mar 2017	449864.0	<i>null</i>		
6	Mar 2018	1159652.0	157.78		
7	Apr 2017	417788.0	<i>null</i>		
8	Apr 2018	1160785.0	177.84		
9	May 2017	592919.0	<i>null</i>		
10	May 2018	1153982.0	94.63		

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

Query:

```
SELECT
  c.customer_state,
  ROUND(SUM(oi.price)) total_price,
  ROUND(AVG(oi.price)) avg_price,
  ROUND(SUM(oi.freight_value)) total_freight,
  ROUND(AVG(oi.freight_value)) avg_freight
FROM
  `target.order_items` oi
JOIN
  `target.orders` o
USING
  (order_id)
JOIN
  `target.customers` c
ON
  c.customer_id = o.customer_id
GROUP BY
  c.customer_state;
```

### Query results

JOB INFORMATION		RESULTS	JSON	EXECUTION DETAILS		EXECUTION GRAPH	PREVIEW
Row	customer_state	total_price	avg_price	total_freight	avg_freight		
1	MT	156454.0	148.0	29715.0	28.0		
2	MA	119648.0	145.0	31524.0	38.0		
3	AL	80315.0	181.0	15915.0	36.0		
4	SP	5202955.0	110.0	718723.0	15.0		
5	MG	1585308.0	121.0	270853.0	21.0		
6	PE	262788.0	146.0	59450.0	33.0		
7	RJ	1824093.0	125.0	305589.0	21.0		
8	DF	302604.0	126.0	50625.0	21.0		
9	RS	750304.0	120.0	135523.0	22.0		
10	SE	58921.0	153.0	14111.0	37.0		

## 5. Analysis on sales, freight and delivery time

### 5.1 Calculate days between purchasing, delivering and estimated delivery

Query:

```
SELECT
  order_id,
  TIMESTAMP_DIFF(order_delivered_customer_date, order_purchase_timestamp, day) AS
  actual_delivery_time_in_days ,
  TIMESTAMP_DIFF(order_estimated_delivery_date, order_purchase_timestamp, day) AS
  estimated_delivery_time_in_days
FROM
  `target.orders`
WHERE
  order_status = "delivered";
```

#### Query results

JOB INFORMATION		RESULTS	JSON	EXECUTION DETAILS	EXECUTION GRAPH	PREVIEW
Row	order_id	actual_delivery_time_in_days	estimated_delivery_time_in_days			
1	635c894d068ac37e6e03dc54e...	30	32			
2	3b97562c3aee8bdedcb5c2e45...	32	33			
3	68f47f50f04c4cb6774570cfde...	29	31			
4	276e9ec344d3bf029ff83a161c...	43	39			
5	54e1a3c2b97fb0809da548a59...	40	36			
6	fd04fa4105ee8045f6a0139ca5...	37	35			
7	302bb8109d097a9fc6e9cefc5...	33	28			
8	66057d37308e787052a32828...	38	32			
9	19135c945c554eebfd7576c73...	36	33			
10	4493e45e7ca1084efcd38ddeb...	34	33			

### 5.2 Find time\_to\_delivery & diff\_estimated\_delivery. Formula for the same given below:

time\_to\_delivery = order\_delivered\_customer\_date-order\_purchase\_timestamp

diff\_estimated\_delivery = order\_estimated\_delivery\_date-order\_delivered\_customer\_date

```
SELECT
```

```

    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";

```

## Query results

JOB INFORMATION		RESULTS	JSON	EXECUT
Row	time_to_delivery	diff_estimated_delivery		
1	30	1		
2	32	0		
3	29	1		
4	43	-4		
5	40	-4		
6	37	-1		
7	33	-5		
8	38	-6		
9	36	-2		
10	34	0		

### 5.3 Group data by state, take mean of freight\_value, time\_to\_delivery, diff\_estimated\_delivery

```

SELECT
    c.customer_state,
    ROUND(AVG(oi.freight_value),2) AS avg_freight_value,
    ROUND(AVG(o.time_to_delivery),2) AS avg_time_to_delivery,
    ROUND(AVG(o.diff_estimated_delivery),2) AS avg_diff_estimated_delivery
FROM
    `target.order_items` oi JOIN
    ( SELECT
        order_id, customer_id,
        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") o
USING
    (order_id)
JOIN
    `target.customers` c
USING
    (customer_id)
GROUP BY
    customer_state;

```

Query results

JOB INFORMATION		RESULTS	JSON	EXECUTION DETAILS	EXECUTION GRAPH	PREVIEW
Row	customer_state	avg_freight_value	avg_time_to_delivery	avg_diff_estimated_delivery		
1	SP	15.12	8.26	10.26		
2	RJ	20.91	14.69	11.14		
3	PR	20.47	11.48	12.53		
4	SC	21.51	14.52	10.66		
5	DF	21.07	12.5	11.27		
6	MG	20.63	11.51	12.4		
7	PA	35.63	23.3	13.37		
8	BA	26.49	18.77	10.12		
9	GO	22.56	14.95	11.37		
10	RS	21.61	14.71	13.2		

5.4 Sort the data to get the following: Top 5 states with highest/lowest average freight value - sort in desc/asc limit 5

States with Highest Avg freight value

```

SELECT
    customer_state
FROM (
    SELECT
        customer_state,
        ROUND(AVG(freight_value),2) AS avg_freight
    FROM
        `target.order_items` oi
    JOIN
        `target.orders` o
    USING
        (order_id)
    JOIN
        `target.customers` c
    USING
        (customer_id)
    GROUP BY

```

```

customer_state
ORDER BY
    avg_freight DESC) t1
LIMIT 5;

```

JOB INFORMATION		RESULTS
Row	customer_state	
1	RR	
2	PB	
3	RO	
4	AC	
5	PI	

States with lowest Avg freight value

```

SELECT
    customer_state
FROM (
    SELECT
        customer_state,
        ROUND(AVG(freight_value),2) AS avg_freight
    FROM
        `target.order_items` oi
    JOIN
        `target.orders` o
    USING
        (order_id)
    JOIN
        `target.customers` c
    USING
        (customer_id)
    GROUP BY
        customer_state
    ORDER BY
        avg_freight ) t1
LIMIT 5;

```

## Query results

JOB INFORMATION		RESULTS
Row	customer_state	
1	SP	
2	PR	
3	MG	
4	RJ	
5	DF	

### 5.5 Top 5 states with highest/lowest average time to delivery

States with highest average time to delivery

```
SELECT
  customer_state
FROM (
  SELECT
    customer_state,
    ROUND(AVG(time_to_delivery),2) AS avg_delivery_time
  FROM (
    SELECT
      customer_state,
      TIMESTAMP_DIFF(order_delivered_customer_date, order_purchase_timestamp, day) AS
time_to_delivery,
    FROM
      `target.orders` o
    JOIN
      `target.customers` c
    USING
      (customer_id)
    WHERE
      order_status = "delivered") t1
  GROUP BY
    customer_state
  ORDER BY
    avg_delivery_time DESC) t2
LIMIT 5;
```



## Query results

JOB INFORMATION		RESULTS
Row	customer_state	
1	RR	
2	AP	
3	AM	
4	AL	
5	PA	

States with lowest average time to delivery

```
SELECT
  customer_state
FROM (
  SELECT
    customer_state,
    ROUND(AVG(time_to_delivery),2) AS avg_delivery_time
  FROM (
    SELECT
      customer_state,
      TIMESTAMP_DIFF(order_delivered_customer_date, order_purchase_timestamp, day) AS
time_to_delivery,
    FROM
      `target.orders` o
    JOIN
      `target.customers` c
    USING
      (customer_id)
    WHERE
      order_status = "delivered") t1
  GROUP BY
    customer_state
  ORDER BY
    avg_delivery_time ) t2
LIMIT 5;
```

## Query results

JOB INFORMATION		RESULTS
Row	customer_state	
1	SP	
2	PR	
3	MG	
4	DF	
5	SC	

### 5.6 Top 5 states with really fast delivery compared to estimated date

```
SELECT
  customer_state,
  ROUND(AVG(estimated_delivery_time - actual_delivery_time), 2) AS delivery_time_difference
FROM (
  SELECT
    o.order_id,
    c.customer_state,
    TIMESTAMP_DIFF(order_delivered_customer_date, order_purchase_timestamp, day) AS
actual_delivery_time,
    TIMESTAMP_DIFF(order_estimated_delivery_date, order_purchase_timestamp, day) AS
estimated_delivery_time
  FROM
    `target.orders` o
  JOIN
    `target.customers` c
  USING
    (customer_id)
  WHERE
    order_status = "delivered") t1
GROUP BY
  customer_state
ORDER BY
  delivery_time_difference DESC
LIMIT 5;
```

## Query results

JOB INFORMATION		RESULTS	JSON	EXECUTION I
Row	customer_state	delivery_time_difference		
1	AC	20.09		
2	RO	19.47		
3	AP	19.13		
4	AM	18.94		
5	RR	16.66		

states with not so fast delivery compared to estimated date

```
SELECT
  customer_state,
  ROUND(AVG(estimated_delivery_time - actual_delivery_time), 2) AS delivery_time_difference
FROM (
  SELECT
    o.order_id,
    c.customer_state,
    TIMESTAMP_DIFF(order_delivered_customer_date, order_purchase_timestamp, day) AS
actual_delivery_time,
    TIMESTAMP_DIFF(order_estimated_delivery_date, order_purchase_timestamp, day) AS
estimated_delivery_time
  FROM
    `target.orders` o
  JOIN
    `target.customers` c
  USING
    (customer_id)
  WHERE
    order_status = "delivered") t1
GROUP BY
  customer_state
ORDER BY
  delivery_time_difference
LIMIT 5;
```

# Query results

JOB INFORMATION		RESULTS	JSON	EXECUTION D
Row	customer_state ▼	delivery_time_difference ▼		
1	AL	8.17		
2	MA	8.97		
3	SE	9.45		
4	ES	9.89		
5	CE	10.19		

## 6. Payment type analysis:

### 6.1 Month over Month count of orders for different payment types

```
SELECT
  time_period,
  payment_type,
  COUNT(*) AS total_orders
FROM (
  SELECT
    p.order_id,
    p.payment_type,
    EXTRACT(YEAR FROM order_purchase_timestamp) AS year,
    EXTRACT(Month FROM order_purchase_timestamp) AS month,
    FORMAT_DATE('%b %Y', DATE(ORDER_PURCHASE_TIMESTAMP)) AS time_period
  FROM
    `target.payments` p
  JOIN
    `target.orders` o
  USING
    (order_id)) t1
GROUP BY
  time_period, payment_type, t1.YEAR, t1.month
ORDER BY
  t1.YEAR, t1.month ;
```

### Query results

JOB INFORMATION		RESULTS	JSON	EXECUTION DETAILS
Row	time_period	payment_type	total_orders	
1	Sep 2016	credit_card	3	
2	Oct 2016	credit_card	254	
3	Oct 2016	voucher	23	
4	Oct 2016	debit_card	2	
5	Oct 2016	UPI	63	
6	Dec 2016	credit_card	1	
7	Jan 2017	voucher	61	
8	Jan 2017	UPI	197	
9	Jan 2017	credit_card	583	
10	Jan 2017	debit_card	9	

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

```
SELECT
  payment_installments,
  COUNT(*) AS total_orders
FROM
  `target.payments`
GROUP BY
  payment_installments;
```

Query results

JOB INFORMATION		RESULTS	JSON
Row	payment_installments	total_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	

## 7. Actionable Insights

- Total 609 orders were unavailable and 625 orders were cancelled during the given time period, which makes it to be around 1.2 % of total orders.  
We can reduce this number by studying the reasons behind order cancellation and items unavailability.

### Query results

JOB INFORMATION		RESULTS	JSON	EXECUTION DETAILS	EXECU
Row	order_status	orders_count	percent_of_total_orders		
1	delivered	96478	97.02		
2	shipped	1107	1.11		
3	canceled	625	0.63		
4	unavailable	609	0.61		
5	invoiced	314	0.32		
6	processing	301	0.3		
7	created	5	0.01		
8	approved	2	0.0		

- We can see how the orders trajectory is showing very abrupt increase in orders volume with in very short time. Looking at overall trend, it is seen that business is picking up very fast in brazil so company has to be ready with extra workforce. To avoid high risk, it can consider hiring contractual employees.

Query results			
JOB INFORMATION		RESULTS	JSON
Row	time_period	order_count	growth_percent
1	Sep 2016	1	null
2	Oct 2016	265	26400.0
3	Dec 2016	1	-99.62
4	Jan 2017	750	74900.0
5	Feb 2017	1653	120.4
6	Mar 2017	2546	54.02
7	Apr 2017	2303	-9.54
8	May 2017	3546	53.97
9	Jun 2017	3135	-11.59
10	Jul 2017	3872	23.51

- Company received low rating for maximum orders in highlighted states; need to study further about the reasons for customer dissatisfaction to such great extent in these states.

This is the query for counting the number of rating in each state.

```
SELECT
*
FROM (
SELECT
c.customer_state,
orv.review_score
FROM
`target.order_reviews` orv
JOIN
`target.orders` o
USING
(order_id)
JOIN
`target.customers` c
USING
(customer_id)) PIVOT(COUNT(*) FOR review_score IN (1, 2, 3, 4, 5));
```



## Query results

JOB INFORMATION		RESULTS	JSON	EXECUTION DETAILS		EXECUTION GRAPH	PREVIEW
Row	customer_state	_1	_2	_3	_4	_5	
1	RS	560	172	449	1098	3204	
2	RJ	2183	464	1050	2137	6931	
3	PR	473	156	381	1009	3019	
4	SC	413	119	321	712	2058	
5	SP	4054	1211	3299	7991	25135	
6	BA	504	130	337	744	1642	
7	GO	233	67	191	423	1110	
8	MG	1207	339	969	2259	6851	
9	PE	221	53	131	322	919	
10	RO	24	15	27	44	142	
11	RN	54	14	39	95	280	
12	SE	63	13	29	67	177	
13	MA	131	31	74	157	353	
14	PA	155	35	96	197	485	
15	CE	210	54	131	263	671	
16	ES	243	57	182	425	1109	

## 8. Recommendations

- As Brazilian customers usually tend to buy in afternoon and night, we can increase staff in during this time frame in order to manage the customers' requests, and services better during this time by reducing workforce of morning and dawn.
- We can see, only 3 state contribute for maximum volume, and rest of the state need to be focused for improving the business.

### Query results

JOB INFORMATION		RESULTS	JSON	EXECU
Row	customer_state	orders_count		
1	SP	41746		
2	RJ	12852		
3	MG	11635		
4	RS	5466		
5	PR	5045		
6	SC	3637		
7	BA	3380		
8	DF	2140		
9	ES	2033		
10	GO	2020		
11	PE	1652		
12	CE	1336		
13	PA	975		
14	MT	807		

- Avg delivery time is quite high for most of those states from where company is receiving quite less volume of orders, detailed study is needed further for checking the other reasons behind such low volume of orders from majority of states. Huge delivery time can be the one of the reason and need to work on it.  
States with highest average delivery time -

## Query results

JOB INFORMATION		RESULTS	JSON	EXE
Row	customer_state ▼	avg_delivery_time ▼		
1	RR	28.98		
2	AP	26.73		
3	AM	25.99		
4	AL	24.04		
5	PA	23.32		
6	MA	21.12		
7	SE	21.03		
8	CE	20.82		
9	AC	20.64		
10	PB	19.95		
11	PI	18.99		
12	RO	18.91		
13	BA	18.87		
14	RN	18.82		
15	PE	17.97		
16	MT	17.59		
17	TO	17.23		