

Business Case: Target SQL

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.

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

1) Data type of columns in a table :

```
SELECT column_name, data_type
FROM target_project.INFORMATION_SCHEMA.COLUMNS
```

The screenshot shows the Google Cloud BigQuery console interface. On the left, the 'Explorer' pane displays the project hierarchy: 'target-project-386401' > 'target_project' > 'customers'. The main area shows a query titled 'Untitled 4' with the following SQL code:

```
1 SELECT column_name, data_type
2 FROM target_project.INFORMATION_SCHEMA.COLUMNS
```

The query has been executed, and the results are displayed in a table. The table has two columns: 'column_name' and 'data_type'. The results are as follows:

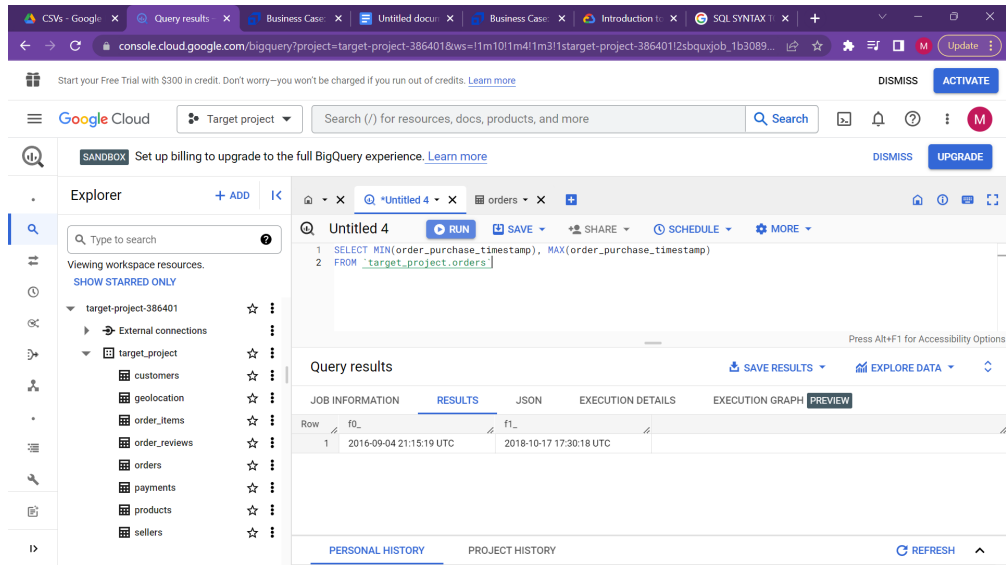
Row	column_name	data_type
1	order_id	STRING
2	order_item_id	INT64
3	product_id	STRING
4	seller_id	STRING
5	shipping_limit_date	TIMESTAMP
6	price	FLOAT64
7	freight_value	FLOAT64

At the bottom of the console, there are tabs for 'PERSONAL HISTORY' and 'PROJECT HISTORY', and a 'REFRESH' button.

INSIGHTS : shows all column names and data types.

2) Time period for which the data is given :

```
SELECT MIN(order_purchase_timestamp), MAX(order_purchase_timestamp)
FROM `target_project.orders`
```



The screenshot shows the Google Cloud BigQuery console interface. On the left, the Explorer pane displays the project hierarchy for 'target-project-386401', including datasets like 'customers', 'geolocation', 'order_items', 'order_reviews', 'orders', 'payments', 'products', and 'sellers'. The main editor area shows a query named 'Untitled 4' with the following SQL:

```
1 SELECT MIN(order_purchase_timestamp), MAX(order_purchase_timestamp)
2 FROM `target_project.orders`
```

The 'Query results' pane at the bottom displays the output of the query. It includes a table with two columns: 'f0_' and 'f1_'. The first row shows the minimum and maximum order purchase timestamps.

Row	f0_	f1_
1	2016-09-04 21:15:19 UTC	2018-10-17 17:30:18 UTC

INSIGHTS : Time frame for the given dataset is between 04/09/2016 - 17/10/2018

3) Cities and States of customers ordered during the given period:

```
select c.customer_city, c.customer_state
FROM `target_project.customers` AS c LEFT JOIN `target_project.orders` as o
ON c.customer_id = o.customer_id
where o.order_purchase_timestamp BETWEEN "2016-09-04" AND "2018-10-17"
group by c.customer_city, c.customer_state
```

The screenshot shows the Google Cloud BigQuery console. The query editor displays the following SQL query:

```

2 FROM `target-project.customers` AS c LEFT JOIN `target-project.orders` AS o
3 ON c.customer_id = o.customer_id
4 WHERE o.order_purchase_timestamp BETWEEN "2016-09-04" AND "2018-10-17"
5 GROUP BY c.customer_city, c.customer_state

```

The query results are displayed in a table with the following columns: Row, customer_city, and customer_state. The results show 6 rows of data.

Row	customer_city	customer_state
1	acu	RN
2	ico	CE
3	ipe	RS
4	ipu	CE
5	ita	SC
6	itu	SP

INSIGHTS : Represents city and state names from where customers have ordered.

In-depth Exploration:

- 1) Is there a growing trend on e-commerce in Brazil? How can we describe a complete scenario? Can we see some seasonality with peaks at specific months?

```

SELECT o.order_id, p.payment_value, extract (date from
o.order_purchase_timestamp) as date
FROM `target-project.orders` as o join `target-project.payments` as p
on o.order_id = p.order_id
order by date

```

The screenshot shows the Google Cloud BigQuery console. The query editor displays the following SQL query:

```

1 SELECT o.order_id, p.payment_value, extract (date from o.order_purchase_timestamp) as date
2 FROM `target-project.orders` as o join `target-project.payments` as p
3 on o.order_id = p.order_id
4 order by o.order_purchase_timestamp

```

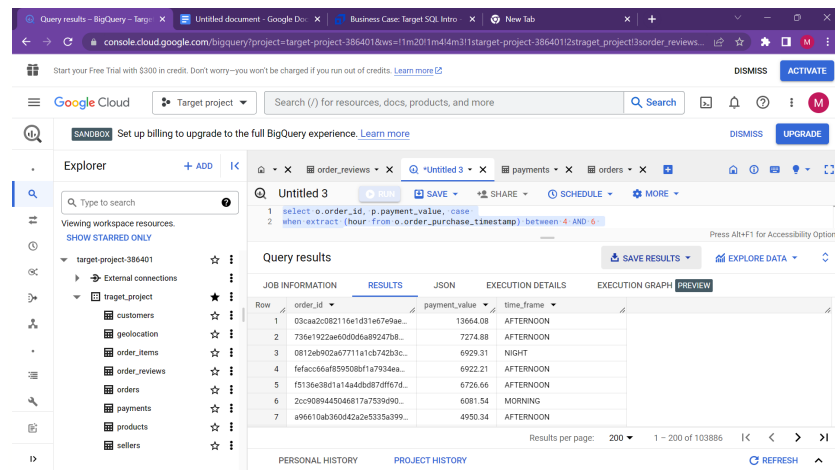
The query results are displayed in a table with the following columns: Row, order_id, payment_value, and date. The results show 6 rows of data.

Row	order_id	payment_value	date
1	03caa2c082116e1d31e67e9ae...	13664.08	2017-09-29
2	730e1922ae60d0da8924708...	7274.88	2018-07-15
3	0812e902a67711a1cd742b3c...	6929.31	2017-02-12
4	feacc6a6f859508bf1a7934ea...	6922.21	2018-02-25
5	f5136e38f1a14a40b97df67d...	6726.66	2017-05-24
6	3c790844d50a817a7539a9d...	6081.54	2017-11-24

INSIGHTS : Growing trend on ecommerce in Brazil was normal. Between August to November for both 2017 and 2018 the payment value was huge when compared to other months.

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

```
select o.order_id, p.payment_value, case
when extract (hour from o.order_purchase_timestamp) between 4 AND 6
THEN "DAWN"
when extract (hour from o.order_purchase_timestamp) between 6 AND 12
THEN "MORNING"
when extract (hour from o.order_purchase_timestamp) between 12 AND 18
THEN "AFTERNOON"
else "NIGHT"
END AS time_frame
from `traget_project.orders` as o join `traget_project.payments` as p
on o.order_id =p.order_id
order by p.payment_value desc
```



The screenshot shows the Google Cloud BigQuery interface. The query results are displayed in a table with the following data:

Row	order_id	payment_value	time_frame
1	03caa208211e1d31e67e9ae...	13664.08	AFTERNOON
2	738e1922ae60d0d6a89247b8...	7274.88	AFTERNOON
3	0812ae602a67711a1cb742b3c...	6929.31	NIGHT
4	1efacc66af859508b1a7934ea...	6922.21	AFTERNOON
5	15136e38d1a14a4dbd87df67d...	6726.66	AFTERNOON
6	2cc9089445046817a7539a990...	6081.54	MORNING
7	a96610ab36042a2e5335a399...	4950.34	AFTERNOON

INSIGHTS : From the results, most of the high value payments happened during the "AFTERNOON" time frame.

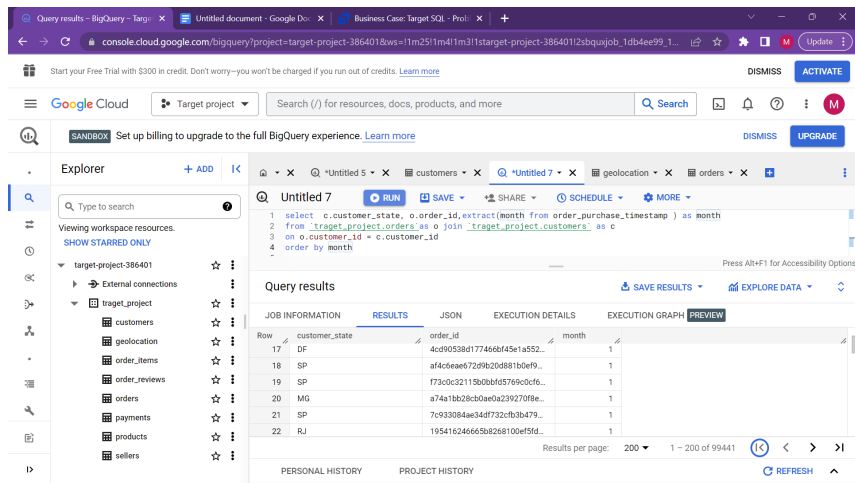
Evolution of E-commerce orders in the Brazil region:

1) Get month on month orders by states :

```

select c.customer_state, o.order_id, extract(month from order_purchase_timestamp ) as
month
from `traget_project.orders` as o join `traget_project.customers` as c
on o.customer_id = c.customer_id
order by month

```



The screenshot shows the Google Cloud BigQuery interface. On the left is the Explorer pane showing the project hierarchy. The main area displays a query titled 'Untitled 7' with the following SQL:

```

1 select c.customer_state, o.order_id, extract(month from order_purchase_timestamp ) as month
2 from `traget_project.orders` as o join `traget_project.customers` as c
3 on o.customer_id = c.customer_id
4 order by month

```

Below the query, the 'Query results' section shows a table with 6 rows and 3 columns: customer_state, order_id, and month. The results are as follows:

Row	customer_state	order_id	month
17	DF	4cd90538d1774668f45e1a552...	1
18	SP	a4c6eae672d9b20d881b0e9...	1
19	SP	f73dc0c32115b0bb65769dc0f6...	1
20	MG	a74a1bb28cb0ae0a239270f9e...	1
21	SP	7c933084ae34df732c3b3b479...	1
22	RJ	19541624665b8268100ef5fd...	1

At the bottom, it indicates 'Results per page: 200' and '1 - 200 of 99441'.

2) Distribution of customers across the states in Brazil:

```

select distinct c.customer_state, c.customer_id, o.order_id, extract(date from
o.order_purchase_timestamp) as purchase_date , o.order_status
from traget_project.customers as c join `traget_project.orders` as o
on c.customer_id = o.customer_id
order by purchase_date

```

The screenshot displays the Google Cloud BigQuery interface. On the left, the 'Explorer' pane shows the project structure with tables like 'customers', 'orders', and 'payments'. The main area shows a query titled 'Untitled 5' with the following SQL code:

```

1 # Distribution of customers across the states in Brazil
2
3 select distinct c.customer_state, c.customer_id, o.order_id, extract(date from o.order_purchase_timestamp) as
4 purchase_date, o.order_status
5 from `traget-project.customers` as c join `traget-project.orders` as o
6 on c.customer_id = o.customer_id
7 order by purchase_date

```

Below the query, the 'Query results' section shows a table with the following data:

Row	customer_state	customer_id	order_id	purchase_date	order_status
1	RR	08c5351adaca1c1589a38f244...	2e7a8482f6b09756ca50c10d...	2016-09-04	shipped
2	RS	683c54fc24d40ee9f8a6fc179f...	e5fa5a72109417d56d0208e4...	2016-09-05	cancelled
3	SP	622e13439d6b5a0b486c4356...	809a282bd5dbcab6f2724fc...	2016-09-13	cancelled

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

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

```

select (payment_value - previous)/ previous * 100 as percent_change
from (select extract (date from o.order_purchase_timestamp) as date_year, o.order_id,
p.payment_value, lag (p.payment_value) over (partition by o.order_id order by
order_purchase_timestamp ) as previous
from `traget-project.orders` as o join `traget-project.payments` as p
on o.order_id = p.order_id
where o.order_purchase_timestamp between "2016-12-31" and "2017-09-01"
order by date_year)

```

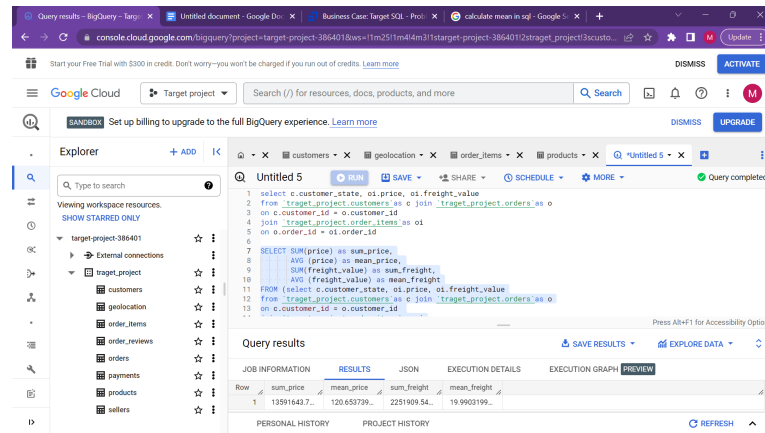
2) Mean & Sum of price and freight value by customer state

```

SELECT SUM(price) as sum_price,
AVG (price) as mean_price,
SUM(freight_value) as sum_freight,
AVG (freight_value) as mean_freight

```

```
FROM (select c.customer_state, oi.price, oi.freight_value
from `traget_project.customers`as c join `traget_project.orders`as o
on c.customer_id = o.customer_id
join `traget_project.order_items`as oi
on o.order_id = oi.order_id)
```



Analysis on sales, freight and delivery time

1) Calculate days between purchasing, delivering and estimated delivery

```
select order_id,extract (day from (estimated_delivery_date - purchased_date))
as day_diff_purchase_estimated_delivery,extract (day from (delivered_date -
purchased_date)) as day_diff_purchase_delivery, extract (day from
(estimated_delivery_date - delivered_date )) as
day_diff_delivery_estimated_delivery,
from (select order_id,extract (date from order_purchase_timestamp) as
purchased_date, extract (date from order_delivered_carrier_date) as
delivered_date, extract (date from order_estimated_delivery_date) as
estimated_delivery_date
from `traget_project.orders`)
```

The screenshot shows the Google Cloud BigQuery console. The query editor contains the following SQL:

```

1 select order_id,extract(day from (estimated_delivery_date - purchased_date)) as
   day_diff_purchase_estimated_delivery,extract(day from (delivered_date - purchased_date)) as
   day_diff_purchase_delivery,extract(day from (estimated_delivery_date - delivered_date)) as
   day_diff_delivery_estimated_delivery
2 from (select order_id,extract(date from order_purchase_timestamp) as purchased_date,extract(date from

```

The query results are displayed in a table with the following columns: `order_id`, `day_diff_purchase_estimated_delivery`, `day_diff_purchase_delivery`, and `day_diff_delivery_estimated_delivery`. The results show data for 7 rows.

Row	order_id	day_diff_purchase_estimated_delivery	day_diff_purchase_delivery	day_diff_delivery_estimated_delivery
3	49db7943d86b68053a41f547...	45	7	38
4	063b573b88fc80e516aba87df...	55	23	32
5	a68ce1686d536ca72bd2dadcd...	57	33	24
6	45973912e49086800c0aeab8f...	55	19	36
7	ndfa87339ba7ah71f673d4ae1...	57	40	17

Insights : This results shows the day's difference between purchased date, delivered date and estimated delivery date.

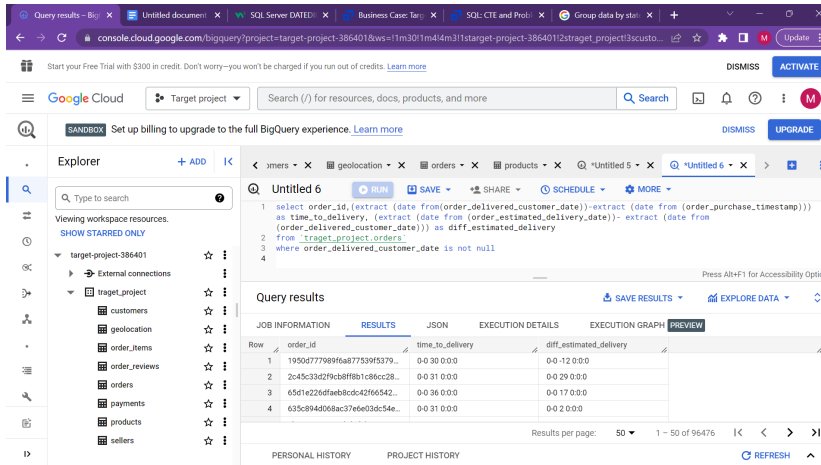
2) Find time_to_delivery & diff_estimated_delivery. Formula for the same given below:

- time_to_delivery =
order_delivered_customer_date-order_purchase_timestamp
- diff_estimated_delivery =
order_estimated_delivery_date-order_delivered_customer_date

```

select order_id,(extract (date from(order_delivered_customer_date))-extract
(date from (order_purchase_timestamp))) as time_to_delivery, (extract (date from
(order_estimated_delivery_date))- extract (date from (order_delivered_customer_date)))
as diff_estimated_delivery
from `traget_project.orders`
where order_delivered_customer_date is not null

```

3) Group data by state, take mean of freight_value, time_to_delivery, diff_estimated_delivery

```
select avg(count(freight_value) as Avg_freight_value, avg(time_to_delivery) as
Avg_ttd, avg (diff_estimated_delivery) as avg_ded
from(select oi.freight_value, (count(o.order_delivered_customer_date)-count
(o.order_purchase_timestamp)) as time_to_delivery,
(count(o.order_estimated_delivery_date)- count
(o.order_delivered_customer_date)) as diff_estimated_delivery
from `traget_project.order_items` as oi join `traget_project.orders` as o
on oi.order_id = o.order_id
join `traget_project.customers` as c
on o.customer_id = c.customer_id)
group by time_to_delivery, c.customer_state, oi.freight_value
```

4) Sort the data to get the following:

- Top 5 states with highest/lowest average freight value - sort in desc/asc limit 5

Highest Freight value :

```
select cus.customer_state, round(avg(oi.freight_value),2) as Avg_freight_value
from `traget_project.order_items` as oi join `traget_project.orders` as o
on oi.order_id = o.order_id
```

```

join `traget_project.customers` as cus
on o.customer_id = cus.customer_id
group by cus.customer_state
order by avg(oi.freight_value) desc
limit 5

```

The screenshot shows the Google Cloud BigQuery console interface. On the left is the Explorer pane showing the project structure with tables like customers, geolocation, order_items, order_reviews, orders, payments, products, and sellers. The main area displays a query titled 'Untitled 3' with the following SQL code:

```

1 #Top 5 states with highest/lowest average freight value - sort in desc/asc limit 5
2
3 select cus.customer_state, round(avg(oi.freight_value),2) as Avg_freight_value
4 from `traget_project.order_items` as oi join `traget_project.orders` as o

```

Below the query editor, the 'Query results' section shows a table with 5 rows of data:

Row	customer_state	Avg_freight_value
1	RR	42.98
2	PB	42.72
3	RO	41.07
4	AC	40.07
5	PI	39.15

Lowest Freight value :

```

select cus.customer_state, round(avg(oi.freight_value),2) as Avg_freight_value
from `traget_project.order_items` as oi join `traget_project.orders` as o
on oi.order_id = o.order_id
join `traget_project.customers` as cus
on o.customer_id = cus.customer_id
group by cus.customer_state
order by avg(oi.freight_value) asc
limit 5

```

The screenshot shows the Google Cloud BigQuery console. On the left, the Explorer pane lists workspace resources for 'target-project-386401', including tables like 'customers', 'geolocation', 'order_items', 'order_reviews', 'orders', 'payments', 'products', and 'sellers'. The main editor shows a query titled 'Untitled 3' with the following SQL:

```
1 #Top 5 states with highest/lowest average freight value - sort in desc/asc limit 5
2
3 select cus.customer_state, round(avg(o1.freight_value),2) as Avg_freight_value
4 from `traget_project.order_items` as o1 join `traget_project.orders` as o
5 on o1.order_id = o.order_id
```

The 'Query results' pane shows a table with 5 rows and 2 columns: 'customer_state' and 'Avg_freight_value'.

Row	customer_state	Avg_freight_value
1	SP	15.15
2	PR	20.53
3	MG	20.63
4	RJ	20.96
5	DF	21.04

- Top 5 states with highest/lowest average time to delivery

Highest average time to delivery:

```
select cus.customer_state,
avg(order_delivered_customer_date-order_purchase_timestamp)as time_to_delivery
from `traget_project.orders` as o join `traget_project.customers` as cus
on o.customer_id = cus.customer_id
group by cus.customer_state
order by time_to_delivery desc
limit 5
```

The screenshot shows the Google Cloud BigQuery console with the same Explorer pane. The main editor shows a query titled 'Untitled 3' with the following SQL:

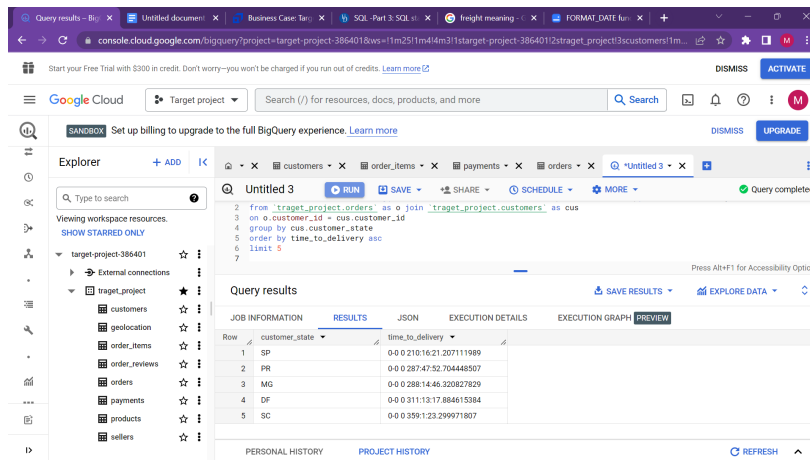
```
1
2 select cus.customer_state, avg(order_delivered_customer_date-order_purchase_timestamp)as time_to_delivery
3 from `traget_project.orders` as o join `traget_project.customers` as cus
4 on o.customer_id = cus.customer_id
5 group by cus.customer_state
6 order by time_to_delivery desc
```

The 'Query results' pane shows a table with 5 rows and 2 columns: 'customer_state' and 'time_to_delivery'.

Row	customer_state	time_to_delivery
1	RR	0.0705183.975609756
2	AP	0.04522629.850746268
3	AM	0.04341325.613793103
4	AL	0.058939.103274559
5	PA	0.0570333.021141649

Lowest average time to delivery:

```
select cus.customer_state,  
avg(order_delivered_customer_date-order_purchase_timestamp)as time_to_delivery  
from `traget_project.orders` as o join `traget_project.customers` as cus  
on o.customer_id = cus.customer_id  
group by cus.customer_state  
order by time_to_delivery asc  
limit 5
```



The screenshot shows the Google Cloud BigQuery console interface. The query editor on the left contains the SQL query from the previous block. The 'Query results' tab is active, displaying a table with 5 rows. The columns are 'customer_state' and 'time_to_delivery'. The results are ordered by 'time_to_delivery' in ascending order.

Row	customer_state	time_to_delivery
1	SP	0-0 0 210:16:21.207111989
2	PR	0-0 0 287:47:52.704448507
3	MG	0-0 0 288:14:46.320827829
4	DF	0-0 0 311:13:17.884615384
5	SC	0-0 0 359:123.299971807

Payment type analysis:

1. Month over Month count of orders for different payment types :

```
select format_date("%m-%Y", o.order_purchase_timestamp) as month_year, p.payment_type,  
count (o.order_id) as orders_count,  
from `traget_project.orders` as o join `traget_project.payments` as p  
on o.order_id = p.order_id  
group by p.payment_type, month_year  
order by p.payment_type, month_year, orders_count
```

Query results

Row	month_year	payment_type	orders_count
1	01-2017	UPI	197
2	01-2018	UPI	1518
3	02-2017	UPI	298
4	02-2018	UPI	1325
5	03-2017	UPI	590
6	03-2018	UPI	1352

2. Count of orders based on the no. of payment installments :

```
select format_date("%m-%Y",o.order_purchase_timestamp) as
month_year,p.payment_installments, count (o.order_id) as orders_count,
from `traget-project.orders` as o join `traget-project.payments` as p
on o.order_id = p.order_id
group by p.payment_installments, month_year
order by p.payment_installments, month_year,orders_count
```

Query results

Row	month_year	payment_installment	orders_count
1	04-2018	0	1
2	05-2018	0	1
3	01-2017	1	469
4	01-2018	1	4076
5	02-2017	1	1044
6	02-2018	1	3697
7	03-2017	1	1490

Results per page: 50 1 - 50 of 324