

## 1.1 Data type of columns in a table

```
all tables x *Unsaved query 3 x *Unsaved query 8 x +
RUN SAVE SHARE SCHEDULE MORE Query completed
1 SELECT table_name, column_name, DATA_TYPE
2 FROM Target_SQL_prproject.INFORMATION_SCHEMA.COLUMNS
3 where table_schema = 'Target_SQL_prproject' and table_name in ('customers','geolocation','order_items',
4 'order_reviews',
'orders','payments','products','sellers')
```

Query results

SAVE RESULTS

EXPLORE DATA

JOB INFORMATIONRESULTSJSONEXECUTION DETAILSEXECUTION GRAPHPREVIEW

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	

## 1.2 Time period for which the data is given

```
all tables x *Unsaved query 3 x orders x *Unsaved query 6 x +
RUN SAVE SHARE SCHEDULE MORE
1 SELECT min(order_purchase_timestamp) as start_date,
2 max(order_purchase_timestamp) as end_date
3 FROM 'first-target-sql-project.Target_SQL_prproject.orders'
```

Query results					SAVE RESULTS	
JOB INFORMATION		RESULTS	JSON	EXECUTION DETAILS	EXECUTION GRAPH	PREVIEW
Row	start_date	end_date				
1	2016-09-04 21:15:19 UTC	2018-10-17 17:30:18 UTC				

### 1.3 Cities and States covered in the dataset

all tables ×

\*Unsaved query 3 ×

Editor 4 ×

geolocation ×

+

RUN

SAVE

SHARE

SCHEDULE

MORE

```
1 SELECT distinct geolocation_city, geolocation_state
2 FROM `first-target-sql-project.Target_SQL_prproject.geolocation`
```

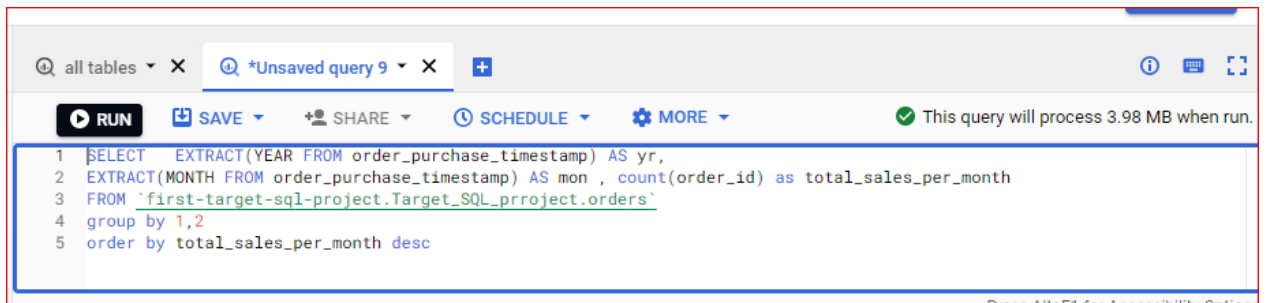
Query results

SAVE RESULTSEXPLORE DATA

JOB INFORMATION		RESULTS	JSON	EXECUTION DETAILS	EXECUTION GRAPH	PREVIEW
Row	geolocation_city	geolocation_state				
1	aracaju	SE				
2	riachuelo	SE				
3	nossa senhora do socorro	SE				
4	barra dos coqueiros	SE				
5	itaporanga d'ajuda	SE				
6	sao cristovao	SE				

Results per page: 501 – 50 of 8463< < >

2.1 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?



The screenshot shows a SQL query editor with a toolbar at the top containing buttons for RUN, SAVE, SHARE, SCHEDULE, and MORE. A status message on the right indicates the query will process 3.98 MB. The query text is as follows:

```
1 SELECT EXTRACT(YEAR FROM order_purchase_timestamp) AS yr,
2 EXTRACT(MONTH FROM order_purchase_timestamp) AS mon , count(order_id) as total_sales_per_month
3 FROM `first-target-sql-project.Target_SQL_prroject.orders`
4 group by 1,2
5 order by total_sales_per_month desc
```

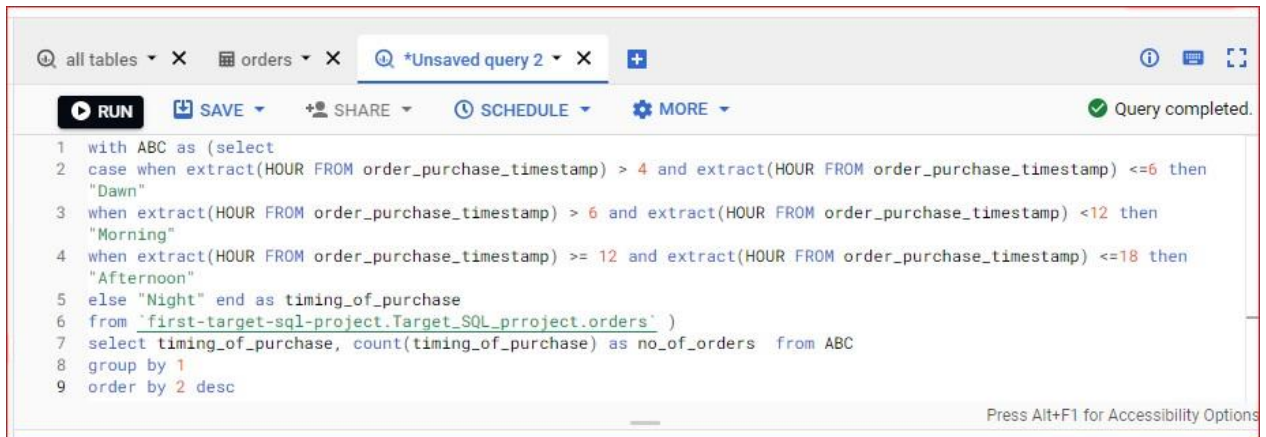
Query results

[SAVE RESULTS](#) [EXPLORE DATA](#)

JOB INFORMATION		RESULTS	JSON	EXECUTION DETAILS	EXECUTION GRAPH	PREVIEW
Row	yr	mon	total_sales_per_month			
1	2017	11	7544			
2	2018	1	7269			
3	2018	3	7211			

Results per page: 50 1 - 25 of 25

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





The screenshot shows a SQL query editor with a toolbar at the top containing buttons for RUN, SAVE, SHARE, SCHEDULE, and MORE. The query text is as follows:


```
1 with ABC as (select
2 case when extract(HOUR FROM order_purchase_timestamp) > 4 and extract(HOUR FROM order_purchase_timestamp) <=6 then
3 "Dawn"
4 when extract(HOUR FROM order_purchase_timestamp) > 6 and extract(HOUR FROM order_purchase_timestamp) <12 then
5 "Morning"
6 when extract(HOUR FROM order_purchase_timestamp) >= 12 and extract(HOUR FROM order_purchase_timestamp) <=18 then
7 "Afternoon"
8 else "Night" end as timing_of_purchase
9 from `first-target-sql-project.Target_SQL_project.orders` )
10 select timing_of_purchase, count(timing_of_purchase) as no_of_orders from ABC
11 group by 1
12 order by 2 desc
```

At the bottom right of the editor, it says "Press Alt+F1 for Accessibility Options". A status message at the top right indicates "Query completed."

Query results

 SAVE RESULTS

 EXPLORE DATA



JOB INFORMATIONRESULTSJSONEXECUTION DETAILSEXECUTION GRAPHPREVIEW

Row	timing_of_purchase	no_of_orders
1	Afternoon	44130
2	Night	32883
3	Morning	21738
4	Dawn	690

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

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

```
1 select B.customer_state, EXTRACT(MONTH from order_purchase_timestamp) as month,
2 EXTRACT(YEAR from order_purchase_timestamp) as year,
3 count(A.order_id)
4 from first-target-sql-project.Target_SQL_prproject.orders A join first-target-sql-project.Target_SQL_prproject.
5 customers B
6 on A.customer_id = B.customer_id
7 group by 1,2,3
8 order by 1
```

Query results

[SAVE RESULTS](#) [EXPLORE DATA](#)

JOB INFORMATION		RESULTS	JSON	EXECUTION DETAILS	EXECUTION GRAPH	PREVIEW
Row	customer_state	month	year	f0_		
17	AC	5	2018	2		
18	AC	3	2017	2		
19	AC	6	2018	3		
20	AC	4	2017	5		
21	AL	7	2017	17		
22	AL	3	2018	30		

Results per page: 50 1 - 50 of 565

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

Query results

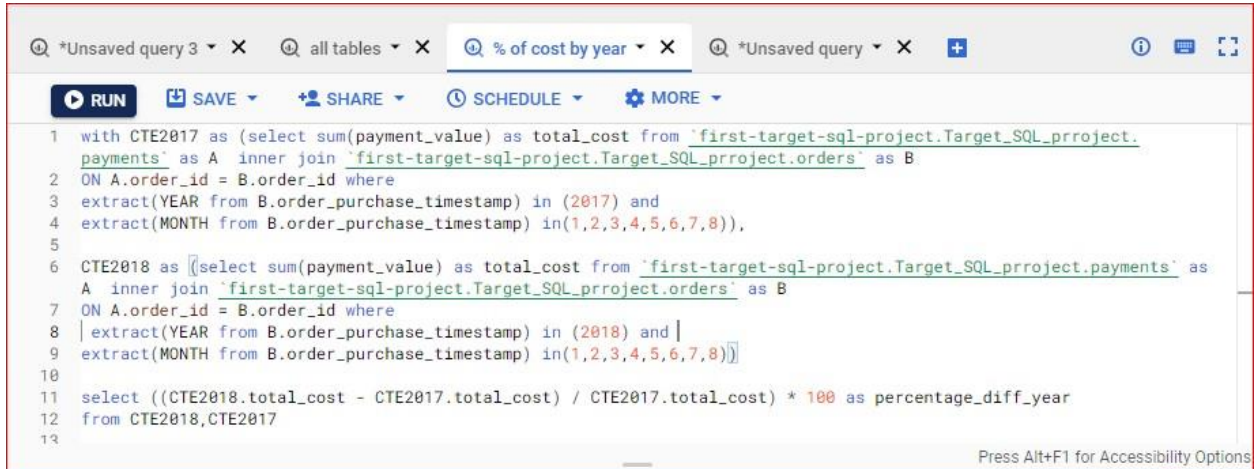
[SAVE RESULTS](#) [EXPLORE DATA](#)

JOB INFORMATION		RESULTS	JSON	EXECUTION DETAILS	EXECUTION GRAPH	PREVIEW
Row	customer_state	Distribution_of_customers				
1	RN	485				
2	CE	1336				
3	RS	5466				
4	SC	3637				
5	SP	41746				

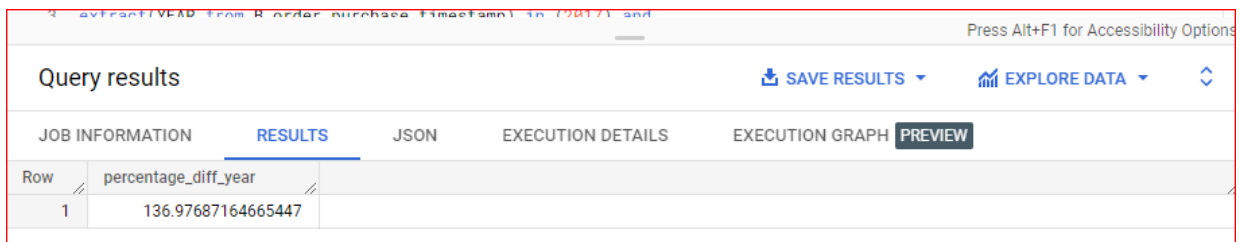
Results per page: 50 1 - 27 of 27

## 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



```
1 with CTE2017 as (select sum(payment_value) as total_cost from `first-target-sql-project.Target_SQL_prproject.payments` as A inner join `first-target-sql-project.Target_SQL_prproject.orders` as B
2 ON A.order_id = B.order_id where
3 extract(YEAR from B.order_purchase_timestamp) in (2017) and
4 extract(MONTH from B.order_purchase_timestamp) in(1,2,3,4,5,6,7,8)),
5
6 CTE2018 as (select sum(payment_value) as total_cost from `first-target-sql-project.Target_SQL_prproject.payments` as
7 A inner join `first-target-sql-project.Target_SQL_prproject.orders` as B
8 ON A.order_id = B.order_id where
9 | extract(YEAR from B.order_purchase_timestamp) in (2018) and |
10 | extract(MONTH from B.order_purchase_timestamp) in(1,2,3,4,5,6,7,8)|)
11 select ((CTE2018.total_cost - CTE2017.total_cost) / CTE2017.total_cost) * 100 as percentage_diff_year
12 from CTE2018,CTE2017
13
```



Query results		SAVE RESULTS	EXPLORE DATA
JOB INFORMATION		RESULTS	JSON
EXECUTION DETAILS		EXECUTION GRAPH	PREVIEW
Row	percentage_diff_year		
1	136.97687164665447		

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

```
1 select A.customer_state, sum(C.payment_value) as SUM_OF_PMTALUE, avg(C.payment_value) as AVG_OF_PMTALUE
2 from `first-target-sql-project.Target_SQL_prproject.customers` as A
3 inner join `first-target-sql-project.Target_SQL_prproject.orders` as B
4 ON A.customer_id = B.customer_id join `first-target-sql-project.Target_SQL_prproject.payments` as C
5 ON B.order_id = C.order_id
6 group by A.customer_state
7 order by A.customer_state
8
```

Query results

[SAVE RESULTS](#) [EXPLORE DATA](#)

JOB INFORMATION		RESULTS	JSON	EXECUTION DETAILS	EXECUTION GRAPH	PREVIEW
Row	customer_state	SUM_OF_PMTAL	AVG_OF_PMTAL			
1	AC	19680.6199...	234.293095...			
2	AL	96962.0599...	227.077423...			
3	AM	27966.9299...	181.603441...			
4	AP	16262.7999...	232.325714...			
5	BA	616645.820...	170.816016...			
6	CF	279464.029	199.902739			

Results per page: 50 1 - 27 of 27

PERSONAL HISTORY PROJECT HISTORY [REFRESH](#)

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

### 1. Calculate days between purchasing,delivering and estimated delivery

```
1 SELECT order_purchase_timestamp, order_estimated_delivery_date, order_delivered_customer_date,
2 abs(DATE_DIFF (order_estimated_delivery_date, order_purchase_timestamp, DAY)) as tot_estmtd_dys_to_delivey,
3 abs(DATE_DIFF (order_delivered_customer_date, order_purchase_timestamp, DAY)) as tot_days_to_delivey,
4 abs(DATE_DIFF (order_estimated_delivery_date, order_delivered_customer_date, DAY)) as diff_estmtd_and_delivey
5 FROM `first-target-sql-project.Target_SQL_project.orders`
6 WHERE order_delivered_customer_date is not null
7
```

Press Alt+F1 for Accessibility Options

Query results

SAVE RESULTS

EXPLORE DATA

JOB INFORMATION

RESULTS

JSON

EXECUTION DETAILS

EXECUTION GRAPH

PREVIEW

Row	order_purchase_timestamp	order_estimated_delivery_date	order_delivered_customer_d	tot_estmtd_dys	tot_days_to_delj	diff_estmtd_and_deliv
1	2016-10-07 14:52:30 UTC	2016-11-29 00:00:00 UTC	2016-10-14 15:07:11 UTC	52	7	45
2	2016-10-09 00:56:52 UTC	2016-11-30 00:00:00 UTC	2016-10-16 14:36:59 UTC	51	7	44
3	2016-10-08 20:17:50 UTC	2016-11-30 00:00:00 UTC	2016-10-19 18:47:43 UTC	52	10	41
4	2017-04-11 13:50:49 UTC	2017-05-18 00:00:00 UTC	2017-04-18 08:18:11 UTC	36	6	29
5	2017-03-17 15:56:47 UTC	2017-05-18 00:00:00 UTC	2017-04-07 13:14:56 UTC	61	20	40
6	2017-03-20 11:01:17 UTC	2017-05-18 00:00:00 UTC	2017-03-30 14:04:04 UTC	58	10	48

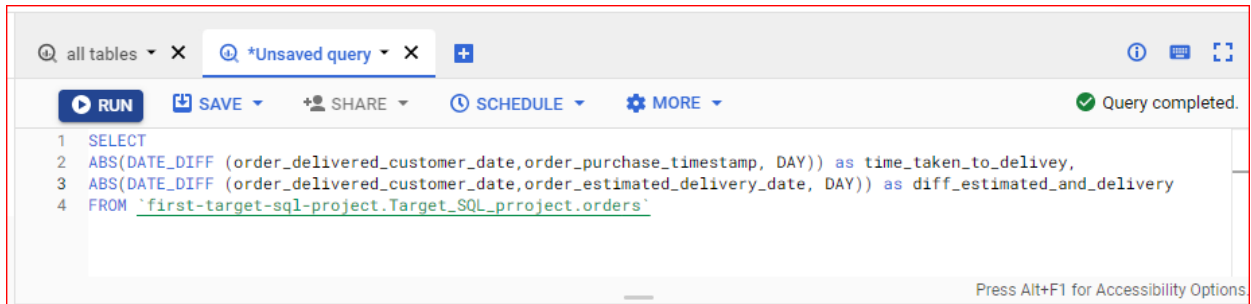
Results per page: 501 - 50 of 96476



## 5.2

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

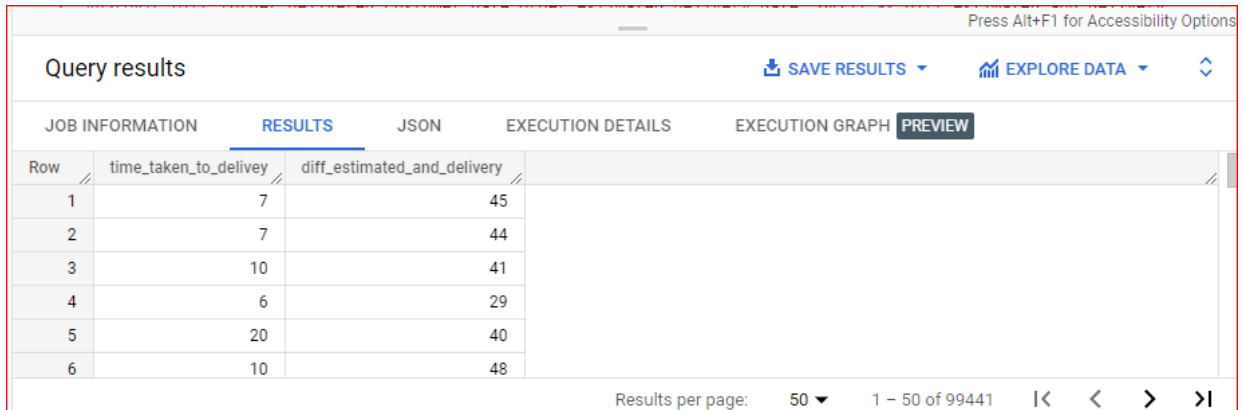
- time\_to\_delivery =  
order\_purchase\_timestamp-order\_delivered\_customer\_date
- diff\_estimated\_delivery =  
order\_estimated\_delivery\_date-order\_delivered\_customer\_date



The screenshot shows a SQL query editor with a toolbar at the top containing buttons for RUN, SAVE, SHARE, SCHEDULE, and MORE. The query text is as follows:

```
1 SELECT
2 ABS(DATE_DIFF (order_delivered_customer_date,order_purchase_timestamp, DAY)) as time_taken_to_delivey,
3 ABS(DATE_DIFF (order_delivered_customer_date,order_estimated_delivery_date, DAY)) as diff_estimated_and_delivery
4 FROM `first-target-sql-project.Target_SQL_prroject.orders`
```

A status message at the bottom right indicates "Query completed." and a hint "Press Alt+F1 for Accessibility Options" is visible.



The screenshot displays the "Query results" section with tabs for JOB INFORMATION, RESULTS, JSON, EXECUTION DETAILS, EXECUTION GRAPH, and PREVIEW. The RESULTS tab is active, showing a table with 6 rows of data. The table has columns for Row, time\_taken\_to\_delivey, and diff\_estimated\_and\_delivery. The data is as follows:

Row	time_taken_to_delivey	diff_estimated_and_delivery
1	7	45
2	7	44
3	10	41
4	6	29
5	20	40
6	10	48

At the bottom, the interface shows "Results per page: 50" and "1 - 50 of 99441" with navigation arrows.

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

```
1 WITH ABC as (select C.customer_state, CAST(B.freight_value as INT) as freight_value,
2 ABS(DATE_DIFF (A.order_delivered_customer_date, A.order_purchase_timestamp, DAY)) as tot_time_taken_delivery,
3 ABS(DATE_DIFF (A.order_delivered_customer_date, A.order_estimated_delivery_date, DAY)) as estimated_delivery_time
4 FROM
5 'first-target-sql-project.Target_SQL_prproject.orders' as A inner join
6 'first-target-sql-project.Target_SQL_prproject.order_items' as B ON A.order_id = B.order_id inner join
7 'first-target-sql-project.Target_SQL_prproject.customers' as C ON A.customer_id = C.customer_id)
8 SELECT customer_state, avg(freight_value) as avg_freight_value, avg(tot_time_taken_delivery) as avg_time_to_delivery,
9 avg(estimated_delivery_time) as avg_estimated_delivery_time
10 from ABC
11 group by 1;
```

Query results

SAVE RESULTS
EXPLORE DATA

JOB INFORMATION		RESULTS	JSON	EXECUTION DETAILS	EXECUTION GRAPH	PREVIEW
Row	customer_state	avg_freight_value	avg_time_to_deliver	avg_estimated_delivery_time		
1	SP	15.153196063141419	8.25960855241901	10.98557371401499		
2	RJ	20.94937924411834	14.68938215750037	14.225505443234859		
3	PR	20.513414634146304	11.480793060718726	13.161621525933793		
4	SC	21.474137931034445	14.520985846754524	12.117862371888725		
5	DF	21.01537822111391	12.501486199575361	12.219957537154977		
6	MG	20.617259501866197	11.515522180072761	13.134783618487242		

Results per page: 50
1 – 27 of 27

5.4 Sort the data to get the following:

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

Highest / DESC

all tables × \*5.5 - highest freig...lue × AVG time taken f... ery × \*5.7 delivery fast ornot × +

RUN

SAVE

SHARE

SCHEDULE

MORE

Query completed.

```
1 SELECT C.customer_state ,SUM(A.freight_value) as total_freight_value, AVG(A.freight_value) as
2 avg_freight_value
3 FROM `first-target-sql-project.Target_SQL_prproject.order_items` as A inner join
4 `first-target-sql-project.Target_SQL_prproject.orders` as B ON A.order_id = B.order_id inner join
5 `first-target-sql-project.Target_SQL_prproject.customers` as C ON B.customer_id = C.customer_id
6 group by 1
7 order by AVG(A.freight_value) desc
8 limit 5;
```

Query results

SAVE RESULTS

EXPLORE DATA

Query results				
Press Alt+F1 for Accessibility Options				
JOB INFORMATION RESULTS JSON EXECUTION DETAILS EXECUTION GRAPH PREVIEW				
Row	customer_state	total_freight_value	avg_freight_value	
1	RR	2235.19	42.984423076923093	
2	PB	25719.7300000000029	42.723803986710941	
3	RO	11417.379999999996	41.069712230215842	
4	AC	3686.7499999999991	40.073369565217405	
5	PI	21218.2000000000033	39.147970479704767	

Lowest / ASC

Query results				
Press Alt+F1 for Accessibility Options				
JOB INFORMATION RESULTS JSON EXECUTION DETAILS EXECUTION GRAPH PREVIEW				
Row	customer_state	total_freight_value	avg_freight_value	
1	SP	718723.069999999378	15.147275390419132	
2	PR	117851.680000000058	20.531651567944269	
3	MG	270853.46000000073	20.630166806306651	
4	RJ	305589.31000000431	20.960923931682483	
5	DF	50625.499999999418	21.041354945968422	

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

highest

all tables × \*5.5 - highest freig...lue × **AVG time taken f... ery** × \*5.7 delivery fast ornot × + ⋮

RUN

SAVE

SHARE

SCHEDULE




MORE

Query completed.

```
1 SELECT B.customer_state,
2 AVG(DATE_DIFF (A.order_delivered_customer_date, A.order_purchase_timestamp, DAY)) as
   avg_time_taken_for_delivery
3 from `first-target-sql-project.Target_SQL_prproject.orders` as A inner join
4 `first-target-sql-project.Target_SQL_prproject.customers` as B ON
5 A.customer_id = B.customer_id
6 group by B.customer_state
7 order by AVG(ABS(DATE_DIFF (A.order_delivered_customer_date, A.order_purchase_timestamp, DAY))) desc
8 limit 5;
9
```




Press Alt+F1 for Accessibility Options

Press Alt+F1 for Accessibility Options

Query results			 SAVE RESULTS ▾	 EXPLORE DATA ▾		
JOB INFORMATION		RESULTS	JSON	EXECUTION DETAILS	EXECUTION GRAPH	PREVIEW
Row	customer_state	avg_time_taken_for_delivery				
1	RR	28.975609756097562				
2	AP	26.731343283582085				
3	AM	25.986206896551728				
4	AL	24.040302267002513				
5	PA	23.316067653276981				

Lowest

Press Alt+F1 for Accessibility Options

Query results			 SAVE RESULTS ▾	 EXPLORE DATA ▾	
JOB INFORMATION		RESULTS	JSON	EXECUTION DETAILS	EXECUTION GRAPH
		<div>PREVIEW</div>			
Row	customer_state	avg_time_taken_for_delivery			
1	SP	8.2980614890725874			
2	PR	11.526711354864908			
3	MG	11.543813298106569			
4	DF	12.509134615384616			
5	SC	14.479560191711331			

## 5.7 Top 5 states where delivery is really fast/ not so fast compared to estimated date

### a. ASC

all tables X \*5.5 - highest freig...lue X \*AVG time taken f... ery X \*5.7 delivery fast ornot X

RUN SAVE SHARE SCHEDULE MORE Query completed.

```
1 SELECT B.customer_state,
2 ABS(AVG(DATE_DIFF (A.order_delivered_customer_date, A.order_purchase_timestamp, DAY))) as
   total_time_of_delivery,
3 ABS(AVG(DATE_DIFF (A.order_estimated_delivery_date, A.order_purchase_timestamp, DAY))) as
   estimated_delivery_time,
4 AVG(DATE_DIFF (A.order_estimated_delivery_date, A.order_delivered_customer_date, DAY)) as
   diff_estmtd_and_delivey
5 FROM `first-target-sql-project.Target_SQL_prproject.orders` as A inner join
6 `first-target-sql-project.Target_SQL_prproject.customers` as B ON
7 A.customer_id = B.customer_id
8 group by B.customer_state
9 order by AVG(DATE_DIFF (A.order_estimated_delivery_date, A.order_delivered_customer_date, DAY)) asc
10 limit 5;
11
```

Press Alt+F1 for Accessibility Options.

Press Alt+F1 for Accessibility Options.

Query results					
JOB INFORMATION RESULTS JSON EXECUTION DETAILS EXECUTION GRAPH PREVIEW					
Row	customer_state	total_time_of_delivery	estimated_delivery_time	diff_estmtd_and_delivey	
1	AL	24.040302267002513	32.22518159806296	7.9471032745591943	
2	MA	21.117154811715473	30.1124497991968	8.76847977684797	
3	SE	21.029850746268657	30.351428571428578	9.1731343283582127	
4	ES	15.331829573934822	25.273487456960144	9.6185463659147885	
5	BA	18.866400491400498	29.036686390532516	9.93488943488941	

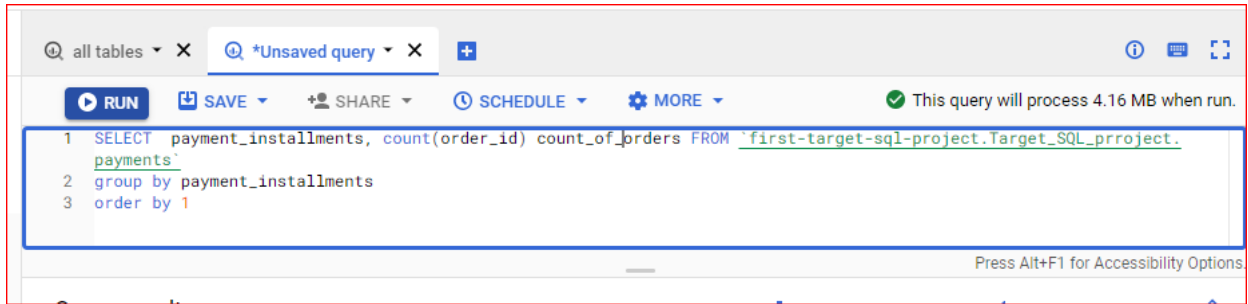
### b. DESC

Press Alt+F1 for Accessibility Options.

Query results					
JOB INFORMATION RESULTS JSON EXECUTION DETAILS EXECUTION GRAPH PREVIEW					
Row	customer_state	total_time_of_delivery	estimated_delivery_time	diff_estmtd_and_delivey	
1	AC	20.637500000000003	40.765432098765423	19.762500000000006	
2	RO	18.913580246913586	38.4071146245059	19.13168724279836	
3	AP	26.731343283582085	45.705882352941174	18.731343283582088	
4	AM	25.986206896551728	44.756756756756765	18.60689655172413	
5	RR	28.975609756097562	46.173913043478251	16.414634146341463	



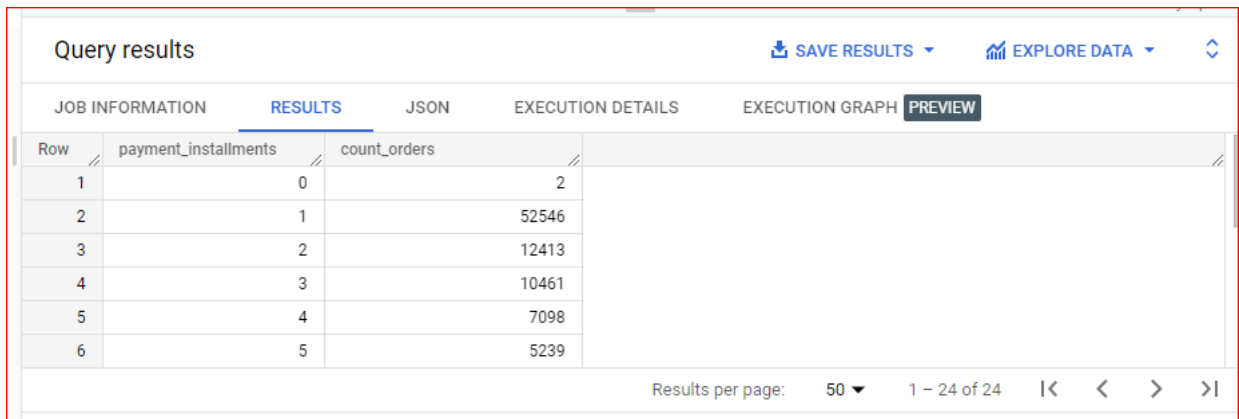
## 6.2 Count of orders based on the no. of payment installments



The screenshot shows a SQL query editor with a toolbar at the top containing buttons for RUN, SAVE, SHARE, SCHEDULE, and MORE. A status message on the right indicates the query will process 4.16 MB. The query text is as follows:

```
1 SELECT payment_installments, count(order_id) count_of_orders FROM `first-target-sql-project.Target_SQL_project.payments`  
2 group by payment_installments  
3 order by 1
```

Below the query editor, a small text prompt reads "Press Alt+F1 for Accessibility Options".



The screenshot displays the "Query results" section with tabs for JOB INFORMATION, RESULTS, JSON, EXECUTION DETAILS, and EXECUTION GRAPH. The RESULTS tab is active, showing a table with two columns: payment\_installments and count\_orders. The table contains six rows of data. At the bottom, there is a pagination control showing "Results per page: 50" and "1 - 24 of 24".

Row	payment_installments	count_orders
1	0	2
2	1	52546
3	2	12413
4	3	10461
5	4	7098
6	5	5239

**7. Actionable Insights for business case TARGET on BRAZIL sales data :**

- I. Customers tend to buy less at the dawn of the day compare to during the day - need to action on this to increase the sales,
- II. Customer base is less in states RN, CE - need to increase the promotions and advertisements in these states,
- III. The total cost of sales increased to 136% in 2018 compared with 2017(Jan - Aug) to 2018(Jan-Aug) data.
- IV. There is a huge difference between estimated delivery time and actual delivery time, it should match or should give an approximate near date to get customer satisfaction,
- V. Total freight values are less where average freight value is more in the states,
- VI. Also, the freight total amount is more where the freight value is less, need to adjust these rates to Increase the customer base.
- VII. Customers using DEBIT CARD as the least payment mode to buy,
- VIII.** There are around 2000 reviews less than rating 3 also 980 reviews less than or equal rating



## **8. Recommendations for the business case: TARGET.**

- I. work on the difference between estimated delivery time and actual delivery time, it should be precise or at least less gap between estimated and actual,
- II. it gains customer assurance and satisfaction,
- III. There should be a reduction in freight value in some states to increase the sales orders, customers expected to be delivered their orders without cost or with minimal cost, which impacts sales,
- IV. Order delivery time needs to be reduced in some of the states,
- V. Debit card is the least preferred option as per data, motivate the customers by offering Royalty/rewards to push customer who doesn't have other payment options,
- VI. There are around 2000 reviews less than the rating of 3 also 980 reviews less than or equal rating as 1, - need to work on reducing the bad ratings reason caused by company end.