

1.1 Data Types of columns in a table

Use `information_schema`;

```
SELECT TABLE_SCHEMA, TABLE_NAME, COLUMN_NAME, DATA_TYPE  
FROM COLUMNS  
WHERE TABLE_SCHEMA = 'target'
```

NOTE: target is name of the database where all the files are stored

The screenshot shows a database management tool interface. On the left, a 'SCHEMAS' tree view is expanded to show the 'target' database. The main window displays a SQL query in a script editor:

```
1 • use information_schema;  
2  
3 • SELECT TABLE_SCHEMA, TABLE_NAME, COLUMN_NAME, DATA_TYPE FROM COLUMNS  
4 • where table_schema = 'target';
```

Below the script editor, the 'Result Grid' shows the output of the query. The columns are TABLE_SCHEMA, TABLE_NAME, COLUMN_NAME, and DATA_TYPE. The results list columns for various tables in the 'target' database, including customers, geolocation, order_items, and order_reviews.

TABLE_SCHEMA	TABLE_NAME	COLUMN_NAME	DATA_TYPE
target	customers	customer_city	text
target	customers	customer_id	text
target	customers	customer_state	text
target	customers	customer_unique_id	text
target	customers	customer_zip_code_prefix	text
target	geolocation	geolocation_city	text
target	geolocation	geolocation_lat	double
target	geolocation	geolocation_lng	double
target	geolocation	geolocation_state	text
target	geolocation	geolocation_zip_code_prefix	text
target	order_items	freight_value	double
target	order_items	order_id	text
target	order_items	order_item_id	int
target	order_items	price	double
target	order_items	product_id	text
target	order_items	seller_id	text
target	order_items	shipping_limit_date	text
target	order_reviews	order_id	text
target	order_reviews	review_answer_timestamp	text
target	order_reviews	review_comment_title	text
target	order_reviews	review_creation_date	text
target	order_reviews	review_id	text

At the bottom, the 'Action Output' pane shows the execution of the 'use information_schema' command, indicating that 0 row(s) were affected.

1.2 Time period for which the data is given

WITH cte AS

```
(SELECT CAST(order_purchase_timestamp AS DATE) order_purchase_date
FROM orders)
```

```
SELECT DATEDIFF(max(order_purchase_date),min(order_purchase_date))
Num_of_days
FROM cte
```

The screenshot shows a SQL IDE interface with a query editor and a results pane. The query editor contains the following SQL code:

```
1 WITH cte AS
2 (SELECT CAST(order_purchase_timestamp AS DATE) order_purchase_date
3 FROM orders)
4
5 SELECT DATEDIFF(max(order_purchase_date),min(order_purchase_date)) Num_of_days
6 FROM cte
```

The results pane shows a single row with the value 773 for the column Num_of_days.

Num_of_days
773

The bottom pane shows the execution log with the following entries:

#	Time	Action	Message	Duration / Fetch
18	13:06:54	WITH orders AS (SELECT CAST(order_purchase_timestamp AS DATE) order_purchase_date FROM o...	1 row(s) returned	0.203 sec / 0.000 s
19	13:11:45	WITH cte AS (SELECT CAST(order_purchase_timestamp AS DATE) order_purchase_date FROM orde...	1 row(s) returned	0.219 sec / 0.000 s

Finding:- From above results we can see that the data is given of nearly more than 2 years. Or more precisely of 773 days.

1.3 Cities and States of customers ordered during the given period

```
SELECT customer_id, customer_city, customer_state
FROM target.customers;
```

SQL File 11* SQL File 12* orders customers

Limit to 200 rows

```

1 • SELECT customer_id, customer_city, customer_state
2 FROM target.customers;

```

customer_id	customer_city	customer_state
06b8999e2fba1a1fbc88172c00ba8bc7	franca	SP
18955e83d337fd6b2def6b18a428ac77	sao bernardo do campo	SP
4e7b3e00288586ebd08712fdd0374a03	sao paulo	SP
b2b6027bc5c5109e529d4dc6358b12c3	mogi das cruzeiras	SP
4f2d8ab171c80ec8364f7c12e35b23ad	campinas	SP
879864dab9bc3047522c92c82e1212b8	jaragua do sul	SC
fd826e7cf63160e536e0908c76c3f441	sao paulo	SP
5e274e7a0c3809e14aba7ad5aae0d407	timoteo	MG
5adf08e34b2e993982a47070956c5c65	curitiba	PR
4b7139f34592b3a31687243a302fa75b	belo horizonte	MG
9fb35e4ed6f0a14a4977cd9aea4042bb	montes claros	MG
5aa9e4fdd4fd20959cad2d772509598	rio de janeiro	RJ
b2d1536598b73a9abd18e0d75d92f0a3	lencois paulista	SP
eabebad39a88bb6f5b52376faec28612	sao paulo	SP
1f1c7bf1c9b041b292a6c1c4470b753	caxias do sul	RS
206f3129c0e4d7d0b9550426023f0a08	piracicaba	SP

customers 3 x

Output

Action Output

#	Time	Action	Message
30	14:13:55	SELECT customer_id, customer_city, customer_state FROM target.customers GROUP BY 1 LIMIT 0, 200	Error Code: 1055. Expression #2 of SELECT list is not in GROUP BY clause and co
31	14:14:39	SELECT customer_id, customer_city, customer_state FROM target.customers LIMIT 0, 200	200 row(s) returned

2.1 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?

WITH cte AS

(SELECT year,total_orders, ABS((COALESCE(lead(total_orders) over(),0) - total_orders))
YoY_Change

FROM

(SELECT EXTRACT(year FROM order_purchase_timestamp) as year, COUNT(*) total_orders

FROM orders

GROUP BY 1

order BY 2) order_table)

```
SELECT year, total_orders, ROUND(lag(( YoY_Change/total_orders)*100) over(),2) AS
percent_increase
```

```
FROM cte
```

Result Grid			
Filter Rows:		Export:	Wrap Cell Content: IA
	year	total_orders	percent_increase
▶	2016	329	NULL
	2017	45101	13608.51
	2018	54011	19.76

Result 24 x			
Output			
Action Output			
#	Time	Action	Message
✓ 118	16:14:24	WITH cte AS (SELECT year,total_orders, ABS((COALESCE(lead(total_orders) over(),0) - total_o...	3 row(s) returned
✓ 119	16:28:05	WITH cte AS (SELECT year,total_orders, ABS((COALESCE(lead(total_orders) over(),0) - total_o...	3 row(s) returned

Findings :-

- From the above picture we can infer that there is nearly 13608 percent of increment from 2016 to 2017. This huge increase is due to insufficient data from 2016 as data of only 3 months is available.
- There is 19.76 percent increment from 2017 to 2018 which show numbers of orders are increasing at quite high rate indicating the positive sign in e-commerce activities.

Part 2

```
WITH cte AS
```

```
(SELECT EXTRACT(month FROM order_purchase_timestamp) month,
DATE_FORMAT(order_purchase_timestamp, '%M') as order_purchase_month,
order_id
```

```
FROM orders
```

)

SELECT CASE

WHEN month IN (1,2,3) THEN 'Summer'

WHEN month IN (4,5,6) THEN 'Autumn'

WHEN month IN (7,8,9) THEN 'Winter'

WHEN month IN (10,11,12) THEN 'Spring'

END Season, order_purchase_month, COUNT(order_id) total_orders

FROM cte

GROUP BY 1,2

Result Grid				Filter Rows:	Export:	Wrap Cell Content:
	Season	order_purchase_month	total_orders			
▶	Winter	August	10843			
	Autumn	May	10573			
	Winter	July	10318			
	Summer	March	9893			
	Autumn	June	9412			
	Autumn	April	9343			
	Summer	February	8508			
	Summer	January	8069			
	Spring	November	7544			
	Spring	December	5674			
	Spring	October	4959			
	Winter	September	4305			

Result 24 x			
Output			
Action Output			
#	Time	Action	Message
✖ 137	21:21:12	WITH cte AS (SELECT EXTRACT(month FROM order_purchase_timestamp) month, DATE_FORMAT(...	Error Code: 1054. Unknown
✔ 138	21:21:44	WITH cte AS (SELECT EXTRACT(month FROM order_purchase_timestamp) month, DATE_FORMAT(...	12 row(s) returned
✔ 139	21:22:09	WITH cte AS (SELECT EXTRACT(month FROM order_purchase_timestamp) month, DATE_FORMAT(...	12 row(s) returned
✔ 140	21:22:31	WITH cte AS (SELECT EXTRACT(month FROM order_purchase_timestamp) month, DATE_FORMAT(...	12 row(s) returned

Findings :-

- The number of orders placed in Winter and Autumn is greater than any other seasons in Brazil. August month shows the peak of the orders.

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

WITH time as

(SELECT DATE_FORMAT(order_purchase_timestamp,'%H') times, order_id as Num_of_orders

FROM orders

)

SELECT CASE

WHEN times between 5 and 6 THEN 'Dawn'

WHEN times between 7 and 11 THEN 'Morning'

WHEN times between 12 and 20 THEN 'Afternoon'

ELSE 'Night'

END as Period,

COUNT(Num_of_orders) total_orders

FROM time

GROUP BY 1

ORDER BY 2 DESC

14

COUNT num_of_orders total_orders

13

FROM time

Result Grid

Filter Rows:

Export:

Wrap Cell Content:

	Period	total_orders
▶	Afternoon	56305
	Morning	21738
	Night	20708
	Dawn	690

Result 46

×

Output

Action Output

#	Time	Action	Message
✓ 179	17:35:18	WITH time as (SELECT DATE_FORMAT(order_purchase_timestamp,'%H') times, order_id as ...	4 row(s) returned
✓ 180	17:39:05	WITH time as (SELECT DATE_FORMAT(order_purchase_timestamp,'%H') times, order_id as ...	4 row(s) returned

Note :- In the above case to find the time period, we categorized the hours of a day in certain periods as follows

- 1) Dawn :- Between 5 and 6
- 2) Morning :- Between 7 and 11
- 3) Afternoon:- Between 12 and 20
- 4) Night:- Remaining left hours are considered as night.

Findings:-

- From above result we can see that the tendency of Brazilians to buy during Afternoon is significantly more than other time

3.1 Get month on month orders by states

WITH cte as

(SELECT customer_state, DATE_FORMAT(order_purchase_timestamp,'%M') as months,
DATE_FORMAT(order_purchase_timestamp,'%m') m_num, count(*) total_orders

FROM orders o

JOIN customers c

ON o.customer_id = c.customer_id

GROUP BY 1,2,3

ORDER BY 1,3)

SELECT customer_state, months, total_orders

FROM cte

Result Grid				Filter Rows:	Export:	Wrap Cell Content:
	customer_state	months	total_orders			
▶	AC	January	8			
	AC	February	6			
	AC	March	4			
	AC	April	9			
	AC	May	10			
	AC	June	7			
	AC	July	9			
	AC	August	7			
	AC	September	5			
	AC	October	6			
	AC	November	5			
	AC	December	5			

Result 20 x

Output

Action Output

#	Time	Action	Message
✓ 240	23:18:54	SELECT customer_state, DATE_FORMAT(order_purchase_timestamp,'%M') as months, DATE...	200 row(s) returned
✓ 241	23:19:17	WITH cte as (SELECT customer_state, DATE_FORMAT(order_purchase_timestamp,'%M') as ...	322 row(s) returned
✓ 242	23:20:50	WITH cte as (SELECT customer_state, DATE_FORMAT(order_purchase_timestamp,'%M') as ...	322 row(s) returned

3.2 Distribution of customers across the states in Brazil

SELECT customer_state,count(*) Customer_distribution

FROM orders o

JOIN customers c

ON o.customer_id = c.customer_id

GROUP BY 1

ORDER BY 2 DESC;

Result Grid		Filter Rows:	Export:	Wrap Cell Content:
customer_state	Customer_distribution			
SP	41746			
RJ	12852			
MG	11635			
RS	5466			
PR	5045			
SC	3637			
BA	3380			
DF	2140			
ES	83			
GO	2020			
PE	1652			
CF	1336			

Result 29 x

Output

Action Output

#	Time	Action	Message
✓ 146	22:17:15	SELECT customer_id, customer_city, customer_state FROM target.customers	99441 row(s) returned
✗ 147	22:56:42	WITH cte as (SELECT customer_state, DATE_FORMAT(order_purchase_timestamp,'%M') as months, ...	Error Code: 1109. Unknown
✓ 148	22:56:59	WITH cte as (SELECT customer_state, DATE_FORMAT(order_purchase_timestamp,'%M') as months, ...	322 row(s) returned
✓ 149	22:59:27	SELECT customer_state,count(*) Customer_distribution FROM orders o JOIN customers c ON o.custo...	27 row(s) returned

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

with cte1 as

(SELECT DATE_FORMAT(order_purchase_timestamp,'%Y') as year,
DATE_FORMAT(order_purchase_timestamp,'%m') as date1, SUM(payment_value) value

FROM orders o

JOIN payments p using(order_id)

GROUP BY 1,2),

cte2 as

```
(SELECT year, sum(value) cost, lead(sum(value)) over(order by year)-sum(value)
c_increment FROM cte1
```

```
WHERE date1 between 1 and 8 and year <> 2016
```

```
group by 1)
```

```
SELECT year, ROUND(cost,2) costs, COALESCE(lag(ROUND((c_increment/cost)*100,2))
over(),') as percent_increase
```

```
FROM cte2
```

The screenshot shows a SQL query result grid and an action output log. The result grid displays data for the years 2017 and 2018, showing a significant increase in costs. The action output log shows four successful actions, each returning 2 rows.

year	costs	percent_increase
2017	3669022.12	
2018	8694733.84	136.98

#	Time	Action	Message
154	23:03:10	with cte1 as (SELECT DATE_FORMAT(order_purchase_timestamp,"%Y") as year, DATE_FORMAT(orde...	2 row(s) returned
155	23:03:32	with cte1 as (SELECT DATE_FORMAT(order_purchase_timestamp,"%Y") as year, DATE_FORMAT(orde...	2 row(s) returned
156	23:04:14	with cte1 as (SELECT DATE_FORMAT(order_purchase_timestamp,"%Y") as year, DATE_FORMAT(orde...	2 row(s) returned
157	23:04:33	with cte1 as (SELECT DATE_FORMAT(order_purchase_timestamp,"%Y") as year, DATE_FORMAT(orde...	2 row(s) returned

Findings:-

There is more than 135 % increase in cost of orders from 2017 to 2018, indicating tremendous growth in terms of cost of orders

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

```

SELECT customer_state, ROUND(avg(price)) Mean_of_Price, ROUND(sum(price))
Sum_of_Price,

ROUND(avg(freight_value)) Mean_of_freight, ROUND(sum(freight_value)) Sum_of_freight

FROM order_items oi

JOIN orders o

using(order_id)

JOIN customers c

using(customer_id)

GROUP BY 1

```

Result Grid					
Filter Rows:					
Export:					
Wrap Cell Content:					
	customer_state	Mean_of_Price	Sum_of_Price	Mean_of_freight	Sum_of_freight
▶	GO	126	294592	23	53115
	DF	126	302604	21	50625
	MG	121	1585308	21	270853
	SP	110	5202955	15	718723
	MS	143	116813	23	19144
	RJ	125	1824093	21	305589
	MA	145	119648	38	31524
	AL	181	80315	36	15915
	BA	135	511350	26	100157
	RS	120	750304	22	135523
	PR	119	683084	21	117852
	SC	125	520553	21	89660
	PA	166	178948	36	38699
	ES	122	275037	22	49765
	RN	157	83035	36	18860
	PB	191	115268	43	25720
	PE	146	262788	33	59450
	CE	154	227255	33	48352

Result 8 x

Output

Action Output

#	Time	Action	Message
92	19:57:15	SELECT customer_state,avg(price),sum(price),avg(freight_value),sum(freight_value) FROM order_items ...	27 row(s) returned
93	19:59:52	SELECT customer_state, ROUND(avg(price)) Mean_of_Price, ROUND(sum(price)) SuM_of_Price, RO...	27 row(s) returned
94	20:00:05	SELECT customer_state, ROUND(avg(price)) Mean_of_Price, ROUND(sum(price)) Sum_of_Price, ROU...	27 row(s) returned

5.1 Calculate days between purchasing, delivering and estimated delivery

```

SELECT
DATEDIFF(order_delivered_customer_date,order_purchase_timestamp)
days_bt看_purchaseAnd_delivery,

DATEDIFF(order_estimated_delivery_date,order_purchase_timestamp)
days_bt看_purchaseAnd_Estimated_delivery

FROM orders

```

Result Grid		Filter Rows:	Export:	Wrap Cell Content:	Fetch rows:
	days_bt看_purchaseAnd_delivery	days_bt看_purchaseAnd_Estimated_delivery			
8		16			
14		20			
9		27			
14		27			
3		13			
17		23			
NULL		28			
10		22			
10		42			
18		25			
13		22			
6		26			

Result 4 x

Output

#	Time	Action	Message
113	22:26:50	SELECT DATEDIFF(order_delivered_customer_date,order_purchase_timestamp) gap1, DATEDIFF(ord...	99441 row(s) returned
114	22:29:27	SELECT DATEDIFF(order_delivered_customer_date,order_purchase_timestamp) as time_to_delivery, D...	99441 row(s) returned
115	22:32:17	SELECT DATEDIFF(order_delivered_customer_date,order_purchase_timestamp) days_bt看_purchaseA...	99441 row(s) returned

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

```

SELECT DATEDIFF(order_delivered_customer_date,order_purchase_timestamp) as
time_to_delivery,

```

```

DATEDIFF(order_estimated_delivery_date,order_delivered_customer_date) as
diff_estimated_delivery

```

FROM orders

Result Grid			
Filter Rows:		Export:	Wrap Cell Content: Fetch rows:
	time_to_delivery	diff_estimated_delivery	
8	8		
14	6		
9	18		
14	13		
3	10		
17	6		
NULL	NULL		
10	12		
10	32		
18	7		
13	9		
6	20		

Result 7 x

Output

#	Time	Action	Message
113	22:26:50	SELECT DATEDIFF(order_delivered_customer_date,order_purchase_timestamp) gap1, DATEDIFF(ord...	99441 row(s) returned
114	22:29:27	SELECT DATEDIFF(order_delivered_customer_date,order_purchase_timestamp) as time_to_delivery, D...	99441 row(s) returned
115	22:32:17	SELECT DATEDIFF(order_delivered_customer_date,order_purchase_timestamp) days_btw_purchaseA...	99441 row(s) returned

5.3 Group data by state, take mean of freight_value, time_to_delivery, diff_estimated_delivery

WITH cte AS

(SELECT DATEDIFF(order_delivered_customer_date,order_purchase_timestamp) as
time_to_delivery,

DATEDIFF(order_estimated_delivery_date,order_delivered_customer_date) as
diff_estimated_delivery, customer_id, order_id

FROM orders)

SELECT customer_state, ROUND(avg(freight_value),2) as freight_value_mean,
ROUND(avg(time_to_delivery),2) time_to_delivery_mean,

ROUND(avg(diff_estimated_delivery),2) diff_estimated_delivery_mean

FROM customers c



JOIN cte ct

using (customer_id)

JOIN order_items o

using (order_id)

GROUP BY customer_state

Result Grid				
Filter Rows:		Export: 		
Wrap Cell Content: 				
	customer_state	freight_value_mean	time_to_delivery_mean	diff_estimated_delivery_mean
▶	GO	22.77	15.34	12.29
	MG	20.63	11.92	13.34
	SP	15.15	8.66	11.21
	MS	23.37	15.46	11.23
	RJ	20.96	15.07	12.01
	MA	38.26	21.59	9.91
	BA	26.36	19.19	10.98
	PE	32.92	18.22	13.45
	RS	21.74	15.13	14.13
	PR	20.53	11.89	13.49
	SC	21.47	14.95	11.57
	ES	22.06	15.59	10.65

Result 8 x

Output

Action Output

#	Time	Action	Message
✓ 126	00:15:46	WITH cte AS (SELECT DATEDIFF(order_delivered_customer_date,order_purchase_timestamp) as time...	27 row(s) returned
✓ 127	00:16:32	WITH cte AS (SELECT DATEDIFF(order_delivered_customer_date,order_purchase_timestamp) as time...	27 row(s) returned
✓ 128	00:18:08	WITH cte AS (SELECT DATEDIFF(order_delivered_customer_date,order_purchase_timestamp) as time...	27 row(s) returned

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

Part 1

WITH cte AS

(SELECT DATEDIFF(order_delivered_customer_date,order_purchase_timestamp) as
time_to_delivery,

```

DATEDIFF(order_estimated_delivery_date,order_delivered_customer_date) as
diff_estimated_delivery, customer_id, order_id

FROM orders)

SELECT customer_state, ROUND(avg(freight_value),2) as freight_value_mean

FROM customers c

JOIN cte ct

using (customer_id)

JOIN order_items o

using (order_id)

GROUP BY customer_state

ORDER BY 2 DESC

LIMIT 5

```

Result Grid		Filter Rows:	Export:	Wrap Cell Content:
customer_state	freight_value_mean			
RR	42.98			
PB	42.72			
RO	41.07			
AC	40.07			
PI	39.15			

Result 48 x			
Output			
Action Output			
#	Time	Action	Message
✓ 177	23:50:49	WITH cte AS (SELECT DATEDIFF(order_delivered_customer_date,order_purchase_timestamp) as time...	5 row(s) returned
✓ 178	23:53:27	WITH cte AS (SELECT DATEDIFF(order_delivered_customer_date,order_purchase_timestamp) as time...	5 row(s) returned

Findings:- The above query gives the top 5 states having most freight_value_mean

Part 2

WITH cte AS

(SELECT DATEDIFF(order_delivered_customer_date,order_purchase_timestamp) as time_to_delivery,

```

DATEDIFF(order_estimated_delivery_date,order_delivered_customer_date) as
diff_estimated_delivery, customer_id, order_id

FROM orders)

SELECT customer_state, ROUND(avg(freight_value),2) as freight_value_mean,

FROM customers c

JOIN cte ct

using (customer_id)

JOIN order_items o

using (order_id)

GROUP BY customer_state

ORDER BY 2

LIMIT 5

```

Result Grid	Filter Rows:	Export:	Wrap Cell Content:
	customer_state	freight_value_mean	
▶	SP	15.15	
	PR	20.53	
	MG	20.63	
	RJ	20.96	
	DF	21.04	

Result 47	×		
Output			
Action Output			
#	Time	Action	Message
✓ 176	23:50:16	WITH cte AS (SELECT DATEDIFF(order_delivered_customer_date,order_purchase_timestamp) as time...	5 row(s) returned
✓ 177	23:50:49	WITH cte AS (SELECT DATEDIFF(order_delivered_customer_date,order_purchase_timestamp) as time...	5 row(s) returned

Findings:- The above query gives the 5 state with lowest freight_value_mean

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

Part 1

WITH cte AS

(SELECT DATEDIFF(order_delivered_customer_date,order_purchase_timestamp) as
time_to_delivery,

DATEDIFF(order_estimated_delivery_date,order_delivered_customer_date) as
diff_estimated_delivery, customer_id, order_id

FROM orders)

SELECT customer_state, ROUND(avg(time_to_delivery),2) time_to_delivery_mean

FROM customers c

JOIN cte ct




using (customer_id)

JOIN order_items o

using (order_id)

GROUP BY customer_state

ORDER BY 2 DESC LIMIT 5

Result Grid		Filter Rows: <input type="text"/>	Export: 	Wrap Cell Content: 
	customer_state	time_to_delivery_mean		
▶	AP	28.22		
	RR	28.17		
	AM	26.34		
	AL	24.45		
	PA	23.70		

Result 51 ×

Output

Action Output

	#	Time	Action	Message
✓	180	23:59:59	WITH cte AS (SELECT DATEDIFF(order_delivered_customer_date,order_purchase_timestamp) as time...	5 row(s) returned
✓	181	00:03:36	WITH cte AS (SELECT DATEDIFF(order_delivered_customer_date,order_purchase_timestamp) as time...	5 row(s) returned

Findings:- From above result, we can say that these are the 5 states where delivery is taking very long times as compared to other states.

Part 2

WITH cte AS

(SELECT DATEDIFF(order_delivered_customer_date,order_purchase_timestamp) as time_to_delivery,
DATEDIFF(order_estimated_delivery_date,order_delivered_customer_date) as diff_estimated_delivery,
customer_id, order_id

FROM orders)

SELECT customer_state, ROUND(avg(time_to_delivery),2) time_to_delivery_mean

FROM customers c

JOIN cte ct

using (customer_id)

JOIN order_items o

using (order_id)

GROUP BY customer_state

ORDER BY 2 LIMIT 5

Result Grid			Filter Rows:	Export:	Wrap Cell Content:
	customer_state	time_to_delivery_mean			
▶	SP	8.66			
	PR	11.89			
	MG	11.92			
	DF	12.89			
	SC	14.95			

Result 52 ×			
Output			
Action Output			
#	Time	Action	Message
✓ 181	00:03:36	WITH cte AS (SELECT DATEDIFF(order_delivered_customer_date,order_purchase_timestamp) as time...	5 row(s) returned
✓ 182	00:10:21	WITH cte AS (SELECT DATEDIFF(order_delivered_customer_date,order_purchase_timestamp) as time...	5 row(s) returned

Finding: - From above we can infer that the delivery in above 5 states is very fast as compared to other states.

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

Part 1

WITH CTE AS

(SELECT customer_id,
DATEDIFF(order_delivered_customer_date,order_purchase_timestamp)
delivery_days,

DATEDIFF(order_estimated_delivery_date,order_purchase_timestamp)
Estimated_delivery_days

FROM orders)

SELECT DISTINCT customer_state, delivery_days

FROM customers

JOIN CTE using(customer_id)

WHERE delivery_days is not null

ORDER BY 2

LIMIT 5

Result Grid			Filter Rows:	Export:	Wrap Cell Content:
	customer_state	delivery_days			
▶	RJ	0			
	RS	1			
	SP	1			
	DF	1			
	GO	1			

Result 29 ×			
Output			
Action Output			
#	Time	Action	Message
✓ 207	01:05:47	WITH CTE AS (SELECT customer_id, DATEDIFF(order_delivered_customer_date,order_purchase_tim...	5 row(s) returned
✓ 208	01:06:16	WITH CTE AS (SELECT customer_id, DATEDIFF(order_delivered_customer_date,order_purchase_tim...	5 row(s) returned

Part 2

WITH CTE AS

```
(SELECT customer_id,  
DATEDIFF(order_delivered_customer_date,order_purchase_timestamp) delivery_days,  
  
DATEDIFF(order_estimated_delivery_date,order_purchase_timestamp)  
Estimated_delivery_days  
  
FROM orders )
```

```
SELECT DISTINCT customer_state, delivery_days, Estimated_delivery_days
```

```
FROM customers
```

```
JOIN CTE using(customer_id)
```

```
WHERE delivery_days is not null
```

ORDER BY 2 DESC

LIMIT 5

Result Grid			
Filter Rows:		Export:	Wrap Cell Content:
	customer_state	delivery_days	Estimated_delivery_days
▶	ES	210	29
	RJ	208	20
	PA	196	31
	PI	195	40
	SE	195	29

Result 33 x			
Output			
Action Output			
#	Time	Action	Message
✓ 211	01:09:17	WITH CTE AS (SELECT customer_id, DATEDIFF(order_delivered_customer_date,order_purchase_tim...	15397 row(s) returned
✓ 212	01:09:49	WITH CTE AS (SELECT customer_id, DATEDIFF(order_delivered_customer_date,order_purchase_tim...	5 row(s) returned

6.1 Month over Month count of orders for different payment types

WITH CTE as

```
(SELECT payment_type, DATE_FORMAT(order_purchase_timestamp,'%m') month_num,  
DATE_FORMAT(order_purchase_timestamp,'%M') month, COUNT(*) count_Of_Order  
FROM orders o  
JOIN payments p  
ON o.order_id = p.order_id  
GROUP BY 1,2,3  
)
```

```
SELECT payment_type, month, count_of_order  
FROM cte  
order by payment_type, month_num
```

Result Grid				Filter Rows:	Export:	Wrap Cell Content:
	payment_type	month	count_of_order			
▶	credit_card	January	6103			
	credit_card	February	6609			
	credit_card	March	7707			
	credit_card	April	7301			
	credit_card	May	8350			
	credit_card	June	7276			
	credit_card	July	7841			
	credit_card	August	8269			
	credit_card	September	3286			
	credit_card	October	3778			
	credit_card	November	5897			
	credit_card	December	4378			
	debit_card	January	118			
	debit_card	February	82			
	debit_card	March	109			
	debit_card	April	124			
	debit_card	May	81			

Result 21 x

Output

Action Output

#	Time	Action	Message
✓ 213	01:13:35	WITH CTE as (SELECT payment_type, DATE_FORMAT(order_purchase_timestamp,"%m") month_num, ...	50 row(s) returned
✓ 214	01:13:43	WITH CTE as (SELECT payment_type, DATE_FORMAT(order_purchase_timestamp,"%m") month_num, ...	50 row(s) returned

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

```

SELECT payment_installments,count(o.order_id) count_order
FROM payments p
JOIN orders o
ON o.order_id = p.order_id
GROUP BY 1

```

Result Grid

Filter Rows:

Export:

Wrap Cell Content:

	payment_installments	count_order
1		52546
3		10461
8		4268
4		7098
2		12413
6		3920
5		5239
10		5328
7		1626
9		644
12		133
21		3
18		27
15		74
13		16
14		15
0		2

Result 6

Output

Action Output

#	Time	Action	Message
214	01:13:43	WITH CTE as (SELECT payment_type, DATE_FORMAT(order_purchase_timestamp,"%m") month_num, ...	50 row(s) returned
215	01:15:58	SELECT payment_installments,count(o.order_id) count_order FROM payments p JOIN orders o ON o.o...	24 row(s) returned

Actionable Insights

- 1) The number of orders increased more than 19.75% from 2017 to 2018
- 2) Highest number of orders are placed in the winter season having August month as its peak.
- 3) Customers generally buy more during afternoon period as compared to others.
- 4) Highest number of customers are present in SP state of Brazil.
- 5) From 2017 to 2018, there is more than 135% YoY increase in cost of orders.
- 6) States like RR, PB, RO, AC and PI have average freight value more than 40.
- 7) States like AP, RR, AM, AL, and PA have average delivery time nearly of 25 days.

- 8) In States like ES, RJ, PA, PI and SE some deliveries are taking nearly 200 days to complete.
- 9) Highest number of orders are made using credit card.
- 10) Count of order is significantly higher when there are just 1 payment instalments.

Recommendations

- 1) Advertise in the state having low number of customers to increase the reach.
- 2) Reduce the average number of freight days in state where it is taking too much time.
- 3) Find the reason and solve the problem of states where some products are taking drastically large number of days to deliver as compared to expected date of delivery.
- 4) Try to give some offers on payment methods which is not used so frequently.