Time period for which the data is given

Query:

```
select min(order_purchase_timestamp)
as oldest_order_date,
max(order_purchase_timestamp) as most_recent_order_date
from `sql_project.orders`
```

Output :

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

States from which customers have placed orders

Query:

```
select distinct customer_state
from `sql_project.customers`
```

Output :

Row	customer_state
1	RN
2	CE
3	RS
4	SC
5	SP
6	MG
7	BA
8	RJ
9	GO
10	MA
11	PE
12	PB
13	ES
14	PR
15	RO
16	MS
17	PA
18	TO
19	MT
20	PI
21	AL
22	AM
23	DF
24	SE
25	RR
26	AP
27	AC

Cities from which customers have placed orders

There are more than four thousand cities across all the states from which orders have been placed,upon running a query to find distinct cities from customers tables yielded a very big collection of city hence a snippet of the same containing the first ten cities is shown below:

```
select distinct customer_city,
#distinct customer_city
from `sql_project.customers`
order by customer_city
limit 10
```

Row	customer_city
1	abadia dos dourados
2	abadiania
3	abaete
4	abaetetuba
5	abaiara
6	abaira
7	abare
8	abatia
9	abdon batista
10	abelardo luz

Growing trend on e-commerce in Brazil

```
select year,count(order_id) as no_of_orders,
count(customer_id) as no_of_customers,sum(payment_value) as total_sales
from
(select t1.*,payment_value,
extract(year from order_purchase_timestamp) as year
from
`sql_project.orders` as t1 left join
`sql_project.payments` as t2
on t1.order_id=t2.order_id) t3
group by year
order by year
```

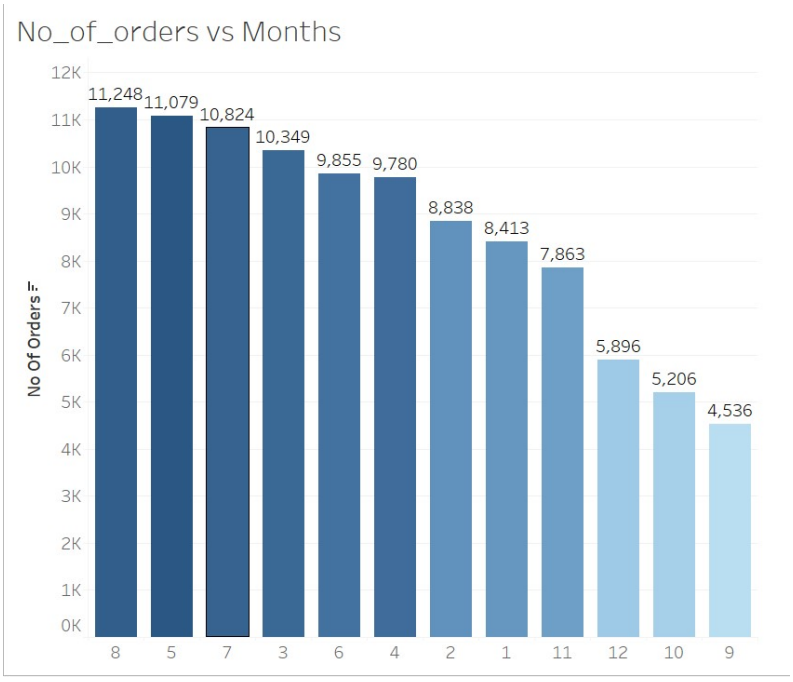
Row	year	no_of_orders	no_of_customer	total_sales
1	2016	347	347	59362.3400...
2	2017	47525	47525	7249746.72...
3	2018	56015	56015	8699763.04...

The above table shows that there is a meteoric increase in customers,orders and total_sales for the year 2017 when compared with the previous year and this trend holds true even for the year 2018 hence it can be clearly stated the e-commerce industry in Brazil has a very good scope of growth for the coming years.

Analysis of orders with respect to months

```
select month,count(order_id) as no_of_orders,sum(payment_value) as total_sales
from
(select t1.*,payment_value,
extract(year from order_purchase_timestamp) as year,extract(month from
order_purchase_timestamp) as month
from
`sql_project.orders` as t1 left join
`sql_project.payments` as t2
on t1.order_id=t2.order_id)
group by month
order by no_of_orders desc
```

Row	month	no_of_orders	total_sales
1	8	11248	1696821.64...
2	5	11079	1746900.96...
3	7	10824	1658923.67...
4	3	10349	1609515.71...
5	6	9855	1535156.88...
6	4	9780	1578573.50...
7	2	8838	1284371.35...
8	1	8413	1253492.22...
9	11	7863	1194882.80...
10	12	5896	878421.100...
11	10	5206	839358.029...
12	9	4536	732454.229...



The above table shows that the no of orders are relatively high for the months of august,may,july,march relative to the other months.The following insights can be drawn based on the above data:

1) The Carnival ,probably the biggest celebration event in Brazil falls in the month of march hence the higher orders.

When do Brazilians buy?

```
select time_of_the_day,count(order_id) as no_of_orders
from
(select *,
(case
when hour>=0 and hour<=6
then 'Dawn'
when hour>=7 and hour<=12
then 'Morning'
when hour>=13 and hour<=18
then 'Evening'
else 'Night'
end) as time_of_the_day
from
(select *,
extract(hour from order_purchase_timestamp) as hour,
#extract(year from order_purchase_timestamp) as year
from `sql_project.orders`) t3)t4
group by time_of_the_day
order by time_of_the_day desc
```

Row	time_of_the_day	no_of_orders
1	Evening	38135
2	Night	28331
3	Morning	27733
4	Dawn	5242

The above table shows that the people of Brazil would usually order more in the evening and not likely to order during dawn but considerable traction of orders is found along both night and morning.

The below query performs a left join on orders and customer table so as to obtain all the corresponding states for the given customer_id in the order table.The solution of the query is stored as a view in biq query going by the name '`scaler-dsml-sql-381611.sql_project.order_date_state`' that shall be used as a basis for deriving queries going forward.

```
(
  SELECT
    t1.*,
 date_diff(order_delivered_customer_date,order_purchase_timestamp,day) as
time_to_delivery,

date_diff(order_estimated_delivery_date,order_delivered_customer_date,day) as
diff_estimated_delivery,
t2.customer_zip_code_prefix,
t2.customer_city,
t2.customer_state
  FROM
    `scaler-dsml-sql-381611.sql_project.orders` t1
  LEFT JOIN
    `sql_project.customers` AS t2
  ON
    t1.customer_id=t2.customer_id)
```

Sample of the above query in table format showing the first 10 rows

Row	order_id	customer_id	order_status	order_purchase_timestamp	order_approved_at	order_delivered_carrier_date
1	7a4df5d8cff4090e541401a20a...	725e9c75605414b21fd8c8d5a...	created	2017-11-25 11:10:33 UTC	*null*	*null*
2	35de4050331c6c644cddc86f4...	4ee64f4bfc542546f422da0aeb...	created	2017-12-05 01:07:58 UTC	*null*	*null*
3	b5359909123fa03c50bdb0cfe...	438449d4af8980d107bf04571...	created	2017-12-05 01:07:52 UTC	*null*	*null*
4	dba5062fbda3af4fb6c33b1e04...	964a6df3d9bdf60fe3e7b8bb69...	created	2018-02-09 17:21:04 UTC	*null*	*null*
5	90ab3e7d52544ec7bc3363c82...	7d61b9f4f216052ba664f22e9c...	created	2017-11-06 13:12:34 UTC	*null*	*null*
6	fa65dad1b0e818e3ccc5cb0e3...	9af2372a1e49340278e7c1ef8...	shipped	2017-04-20 12:45:34 UTC	2017-04-22 09:10:13 UTC	2017-04-24 11:31:17 UTC
7	1df2775799eecdf9dd8502425...	1240c2e65c4601dd860e3a367...	shipped	2017-07-13 11:03:05 UTC	2017-07-13 11:10:22 UTC	2017-07-18 18:17:30 UTC
8	6190a94657e1012983a274b8...	5fc4c97dcb63903f996714524...	shipped	2017-07-11 13:36:30 UTC	2017-07-11 13:45:15 UTC	2017-07-13 17:55:46 UTC
9	58ce513a55c740a3a81e8c8b7...	530d41b47b9dda9bc6f31d856...	shipped	2017-07-29 18:05:07 UTC	2017-07-29 18:15:17 UTC	2017-07-31 16:41:59 UTC
10	088683f795a3d30bfd61152c4f...	58d89fd1f863819ff9b040734f...	shipped	2017-07-13 10:02:47 UTC	2017-07-14 02:25:54 UTC	2017-07-20 20:02:58 UTC

Row	order_delivered_carrier_date	order_delivered_customer_date	order_estimated_delivery_date	time_to_delivery	diff_estimated_c	customer_zip_cc	customer_city	customer_state
1	*null*	*null*	2017-12-12 00:00:00 UTC	*null*	*null*	22621	rio de janeiro	RJ
2	*null*	*null*	2018-01-08 00:00:00 UTC	*null*	*null*	93025	sao leopoldo	RS
3	*null*	*null*	2018-01-11 00:00:00 UTC	*null*	*null*	15300	general salgado	SP
4	*null*	*null*	2018-03-07 00:00:00 UTC	*null*	*null*	73401	brasilia	DF
5	*null*	*null*	2017-12-01 00:00:00 UTC	*null*	*null*	87720	paranavai	PR
6	2017-04-24 11:31:17 UTC	*null*	2017-05-18 00:00:00 UTC	*null*	*null*	78065	cuiaba	MT
7	2017-07-18 18:17:30 UTC	*null*	2017-08-14 00:00:00 UTC	*null*	*null*	65063	sao luis	MA
8	2017-07-13 17:55:46 UTC	*null*	2017-08-14 00:00:00 UTC	*null*	*null*	57010	maceio	AL
9	2017-07-31 16:41:59 UTC	*null*	2017-08-14 00:00:00 UTC	*null*	*null*	13185	hortolandia	SP
10	2017-07-20 20:02:58 UTC	*null*	2017-08-14 00:00:00 UTC	*null*	*null*	78138	varzea grande	MT

Distribution of customers across the states in Brazil

```sql
SELECT customer_state,
count(customer_id) as no_of_customers
 FROM `scaler-dsml-sql-381611.sql_project.order_date_state`
group by customer_state
order by no_of_customers desc
limit 10
```

Row	customer_state	no_of_customer
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

Month on month orders by states

```sql
select customer_state,month,count(order_id) as no_of_orders

from

(SELECT *,extract(month from order_purchase_timestamp)as month

 FROM `scaler-dsml-sql-381611.sql_project.order_date_state` ) t1
```

```
group by customer_state,month

order by customer_state,month

limit 10
```

Row	customer_state	month	no_of_orders
1	AC	1	8
2	AC	2	6
3	AC	3	4
4	AC	4	9
5	AC	5	10
6	AC	6	7
7	AC	7	9
8	AC	8	7
9	AC	9	5
10	AC	10	6

Mean & Sum of price and freight value by customer state

```
SELECT

  customer_state,

  round(sum(price) ,0)AS totaL_price_value,

  round(sum(freight_value),0) AS total_freight_value,

  round(AVG(price),0) AS avg_price,

  round(AVG(freight_value),0) AS avg_freight
```

```
FROM (
  SELECT
    t1.order_id,
    customer_id,
    customer_state,
    order_status,
    order_purchase_timestamp,
    order_delivered_customer_date,
    order_estimated_delivery_date,
    time_to_delivery,
    diff_estimated_delivery,
    price,
    freight_value
  FROM
    `scaler-dsml-sql-381611.sql_project.order_date_state` AS t1
LEFT JOIN
    `sql_project.order_items` AS t2
  ON
    t1.order_id=t2.order_id) t1
GROUP BY
  customer_state
```

```
order by total_price_value desc

limit 10
```

Row	customer_state	totaL_price_valu	total_freight_val	avg_price	avg_freight
1	SP	5202955.0	718723.0	110.0	15.0
2	RJ	1824093.0	305589.0	125.0	21.0
3	MG	1585308.0	270853.0	121.0	21.0
4	RS	750304.0	135523.0	120.0	22.0
5	PR	683084.0	117852.0	119.0	21.0
6	SC	520553.0	89660.0	125.0	21.0
7	BA	511350.0	100157.0	135.0	26.0
8	DF	302604.0	50625.0	126.0	21.0
9	GO	294592.0	53115.0	126.0	23.0
10	ES	275037.0	49765.0	122.0	22.0

YOY in cost of orders from 2017 to 2018

```
select year,tot,

(c/b )*100 as yoy_growth

from

(select

*,

lag(tot) over(order by tot) as b,

tot-lag(tot) over(order by tot) as c

from

(select year,sum(payment_value)as tot

from(select t1.order_id,
```

```
extract(year from t1.order_purchase_timestamp) as year,

extract(month from t1.order_purchase_timestamp) as month,payment_value

from

`sql_project.orders` t1

left join

`sql_project.payments` t2

on t1.order_id=t2.order_id) t1

where month<=8

group by year

having year in (2017,2018) ) t3

order by year) t5
```

Row	year	tot	yoy_growth
1	2017	3669022.11...	*null*
2	2018	8694733.83...	136.976871...

The below query performs a left join on orders and customers table so as to obtain all the corresponding state and other geographical features corresponding to customer_id in the order table.

```
(
  SELECT
    t1.*,
    DATE_DIFF(order_delivered_customer_date,order_purchase_timestamp,day) AS
time_to_delivery,
    DATE_DIFF(order_estimated_delivery_date,order_delivered_customer_date,day)
AS diff_estimated_delivery,
    t2.customer_zip_code_prefix,
    t2.customer_city,
    t2.customer_state
  FROM
    `scaler-dsml-sql-381611.sql_project.orders` t1
  LEFT JOIN
    `sql_project.customers` AS t2
  ON
    t1.customer_id=t2.customer_id)
```

Row	order_id	customer_id	order_status	order_purchase_timestamp	order_approved_at	order_delivered_carrier_date
1	7a4df5d8cff4090e541401a20a…	725e9c75605414b21fd8c8d5a…	created	2017-11-25 11:10:33 UTC	*null*	*null*
2	35de4050331c6c644cddc86f4…	4ee64f4bfc542546f422da0aeb…	created	2017-12-05 01:07:58 UTC	*null*	*null*
3	b5359909123fa03c50bdb0cfe…	438449d4af8980d107bf04571…	created	2017-12-05 01:07:52 UTC	*null*	*null*
4	dba5062fbda3af4fb6c33b1e04…	964a6df3d9bdf60fe3e7b8bb69…	created	2018-02-09 17:21:04 UTC	*null*	*null*
5	90ab3e7d52544ec7bc3363c82…	7d61b9f4f216052ba664f22e9c…	created	2017-11-06 13:12:34 UTC	*null*	*null*
6	fa65dad1b0e818e3ccc5cb0e3…	9af2372a1e49340278e7c1ef8…	shipped	2017-04-20 12:45:34 UTC	2017-04-22 09:10:13 UTC	2017-04-24 11:31:17 UTC
7	1df2775799eecdf9dd8502425…	1240c2e65c4601dd860e3a367…	shipped	2017-07-13 11:03:05 UTC	2017-07-13 11:10:22 UTC	2017-07-18 18:17:30 UTC
8	6190a94657e1012983a274b8…	5fc4c97dcb63903f996714524…	shipped	2017-07-11 13:36:30 UTC	2017-07-11 13:45:15 UTC	2017-07-13 17:55:46 UTC
9	58ce513a55c740a3a81e8c8b7…	530d41b47b9dda9bc6f31d856…	shipped	2017-07-29 18:05:07 UTC	2017-07-29 18:15:17 UTC	2017-07-31 16:41:59 UTC
10	088683f795a3d30bfd61152c4f…	58d89fd1f863819ff9b040734f…	shipped	2017-07-13 10:02:47 UTC	2017-07-14 02:25:54 UTC	2017-07-20 20:02:58 UTC

Row	order_delivered_carrier_date	order_delivered_customer_date	order_estimated_delivery_date	time_to_delivery	diff_estimated_c	customer_zip_cc	customer_city	customer_state
1	null	null	2017-12-12 00:00:00 UTC	null	null	22621	rio de janeiro	RJ
2	null	null	2018-01-08 00:00:00 UTC	null	null	93025	sao leopoldo	RS
3	null	null	2018-01-11 00:00:00 UTC	null	null	15300	general salgado	SP
4	null	null	2018-03-07 00:00:00 UTC	null	null	73401	brasilia	DF
5	null	null	2017-12-01 00:00:00 UTC	null	null	87720	paranavai	PR
6	2017-04-24 11:31:17 UTC	null	2017-05-18 00:00:00 UTC	null	null	78065	cuiaba	MT
7	2017-07-18 18:17:30 UTC	null	2017-08-14 00:00:00 UTC	null	null	65063	sao luis	MA
8	2017-07-13 17:55:46 UTC	null	2017-08-14 00:00:00 UTC	null	null	57010	maceio	AL
9	2017-07-31 16:41:59 UTC	null	2017-08-14 00:00:00 UTC	null	null	13185	hortolandia	SP
10	2017-07-20 20:02:58 UTC	null	2017-08-14 00:00:00 UTC	null	null	78138	varzea grande	MT

Grouping the data by state against average values of freight,time_to_delivery,price and estimate _delivery

The below query performs a left join on order_date_state and order_items table so as to obtain all the corresponding freight and price for the given order_id in the order table and then is being grouped as per state against the average average values of freight,time_to_delivery,price and estimate _delivery

```
SELECT
  customer_state,
  AVG(diff_estimated_delivery) AS avg_estimate_delivery,
  AVG(time_to_delivery) AS avg_time_to_delivery,
  AVG(price) AS avg_price,
  AVG(freight_value) AS avg_freight
FROM (
  SELECT
    T1.order_id,
    customer_id,
    customer_state,
    order_status,
    order_purchase_timestamp,
    order_delivered_customer_date,
    order_estimated_delivery_date,
    time_to_delivery,
    diff_estimated_delivery,
    price,
    freight_value
```

```
  FROM
    `scaler-dsml-sql-381611.sql_project.order_date_state` AS t1
  LEFT JOIN
    `sql_project.order_items` AS t2
  ON
    t1.order_id=t2.order_id) t1
GROUP BY
  customer_state
  limit 10
```

Row	customer_state	avg_estimate_de	avg_time_to_deli	avg_price	avg_freight
1	RJ	11.1444931...	14.6893821...	125.117818...	20.9609239...
2	RS	13.2030001...	14.7082993...	120.337453...	21.7358043...
3	SP	10.2655943...	8.25960855...	109.653629...	15.1472753...
4	DF	11.2747346...	12.5014861...	125.770548...	21.0413549...
5	PR	12.5338998...	11.4807930...	119.004139...	20.5316515...
6	MT	13.6393442...	17.5081967...	148.297184...	28.1662843...
7	MA	9.10999999...	21.2037500...	145.204150...	38.2570024...
8	AL	7.97658079...	23.9929742...	180.889211...	35.8436711...
9	MG	12.3971510...	11.5155221...	120.748574...	20.6301668...
10	PE	12.5521191...	17.7920962...	145.508322...	32.9178626...

Top 5 states with lowest average time to delivery

```
SELECT
  customer_state,
  #AVG(diff_estimated_delivery) AS avg_estimate_delivery,
  round(AVG(time_to_delivery),0) AS avg_time_to_delivery_in_days,
  #AVG(price) AS avg_price,
  #AVG(freight_value) AS avg_freight
FROM (
  SELECT
    t1.order_id,
    customer_id,
    customer_state,
```

```
    order_status,
    order_purchase_timestamp,
    order_delivered_customer_date,
    order_estimated_delivery_date,
    time_to_delivery,
    diff_estimated_delivery,
    price,
    freight_value
  FROM
    `scaler-dsml-sql-381611.sql_project.order_date_state` AS t1
  LEFT JOIN
    `sql_project.order_items` AS t2
  ON
    t1.order_id=t2.order_id) t1
GROUP BY
  customer_state
order by avg_time_to_delivery_in_days
  limit 5
```

Row	customer_state	avg_time_to_deli
1	SP	8.0
2	PR	11.0
3	MG	12.0
4	DF	13.0
5	RS	15.0

Top 5 states with highest average time to delivery

```
SELECT
  customer_state,
  #AVG(diff_estimated_delivery) AS avg_estimate_delivery,
  round(AVG(time_to_delivery),0) AS avg_time_to_delivery_in_days,
  #AVG(price) AS avg_price,
  #AVG(freight_value) AS avg_freight
FROM (
  SELECT
```

```
    t1.order_id,
    customer_id,
    customer_state,
    order_status,
    order_purchase_timestamp,
    order_delivered_customer_date,
    order_estimated_delivery_date,
    time_to_delivery,
    diff_estimated_delivery,
    price,
    freight_value
  FROM
    `scaler-dsml-sql-381611.sql_project.order_date_state` AS t1
  LEFT JOIN
    `sql_project.order_items` AS t2
  ON
    t1.order_id=t2.order_id) t1
GROUP BY
  customer_state
order by avg_time_to_delivery_in_days desc
  limit 5
```

Row	customer_state	avg_time_to_deli
1	AP	28.0
2	RR	28.0
3	AM	26.0
4	AL	24.0
5	PA	23.0

Top 5 states with lowest average freight value

```
SELECT
  customer_state,
  #AVG(diff_estimated_delivery) AS avg_estimate_delivery,
  #round(AVG(time_to_delivery),0) AS avg_time_to_delivery_in_days,
```

```
  #AVG(price) AS avg_price,
  round(AVG(freight_value),0) AS avg_freight
FROM (
  SELECT
    t1.order_id,
    customer_id,
    customer_state,
    order_status,
    order_purchase_timestamp,
    order_delivered_customer_date,
    order_estimated_delivery_date,
    time_to_delivery,
    diff_estimated_delivery,
    price,
    freight_value
  FROM
    `scaler-dsml-sql-381611.sql_project.order_date_state` AS t1
  LEFT JOIN
    `sql_project.order_items` AS t2
  ON
    t1.order_id=t2.order_id) t1
GROUP BY
  customer_state
order by avg_freight
  limit 5
```

Row	customer_state	avg_freight
1	SP	15.0
2	PR	21.0
3	RJ	21.0
4	DF	21.0
5	MG	21.0

Top 5 states with highest average freight value

```sql
SELECT
  customer_state,
  #AVG(diff_estimated_delivery) AS avg_estimate_delivery,
  #round(AVG(time_to_delivery),0) AS avg_time_to_delivery_in_days,
  #AVG(price) AS avg_price,
  round(AVG(freight_value),0) AS avg_freight
FROM (
  SELECT
    t1.order_id,
    customer_id,
    customer_state,
    order_status,
    order_purchase_timestamp,
    order_delivered_customer_date,
    order_estimated_delivery_date,
    time_to_delivery,
    diff_estimated_delivery,
    price,
    freight_value
  FROM
    `scaler-dsml-sql-381611.sql_project.order_date_state` AS t1
  LEFT JOIN
    `sql_project.order_items` AS t2
  ON
    t1.order_id=t2.order_id) t1
GROUP BY
  customer_state
order by avg_freight desc
  limit 5
```

Row	customer_state	avg_freight
1	RR	43.0
2	PB	43.0
3	RO	41.0
4	AC	40.0
5	PI	39.0

Top 5 states where delivery is really fast compared to estimated date

```
SELECT
  customer_state,
  ROUND(AVG(diff_estimated_delivery),0) AS avg_estimate_delivery
FROM (
  SELECT
    t1.order_id,
    customer_id,
    customer_state,
    order_status,
    order_purchase_timestamp,
    order_delivered_customer_date,
    order_estimated_delivery_date,
    time_to_delivery,
    diff_estimated_delivery,
    price,
    freight_value
  FROM
    `scaler-dsml-sql-381611.sql_project.order_date_state` AS t1
  LEFT JOIN
    `sql_project.order_items` AS t2
  ON
    t1.order_id=t2.order_id) t1
GROUP BY
  customer_state
order by avg_estimate_delivery desc
  limit 5
```

Row	customer_state	avg_estimate_de
1	AC	20.0
2	RO	19.0
3	AM	19.0
4	RR	17.0
5	AP	17.0

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

```sql
SELECT
  customer_state,
  ROUND(AVG(diff_estimated_delivery),0) AS avg_estimate_delivery
FROM (
  SELECT
    t1.order_id,
    customer_id,
    customer_state,
    order_status,
    order_purchase_timestamp,
    order_delivered_customer_date,
    order_estimated_delivery_date,
    time_to_delivery,
    diff_estimated_delivery,
    price,
    freight_value
  FROM
    `scaler-dsml-sql-381611.sql_project.order_date_state` AS t1
  LEFT JOIN
    `sql_project.order_items` AS t2
  ON
    t1.order_id=t2.order_id) t1
GROUP BY
  customer_state
order by avg_estimate_delivery desc
  limit 5
```

Row	customer_state	avg_estimate_de
1	AL	8.0
2	MA	9.0
3	SE	9.0
4	SP	10.0
5	BA	10.0

Count of orders based on the no. of payment installments

The above query performs a left join on orders and payments table so as to obtain all the corresponding payment related columns corresponding to order_id in orders table.

```sql
select payment_installments,count(order_id) as no_of_orders

from

(SELECT

  t1.order_id,

  extract(month from order_purchase_timestamp) as months ,

  payment_type,

  payment_installments

FROM

  `sql_project.orders` t1

LEFT JOIN

  `scaler-dsml-sql-381611.sql_project.payments` t2

ON

  t1.order_id=t2.order_id

) t5

group by payment_installments

order by payment_installments

limit 10
```

Row	payment_installr	no_of_orders
1	*null*	1
2	0	2
3	1	52546
4	2	12413
5	3	10461
6	4	7098
7	5	5239
8	6	3920
9	7	1626
10	8	4268

Month over Month count of orders for different payment types

```
select months,payment_type,count(order_id) as  no_of_orders
from
(SELECT
  t1.order_id,
  extract(month from order_purchase_timestamp) as months ,
  payment_type,
  payment_installments
FROM
  `sql_project.orders` t1
LEFT JOIN
```

```
  `scaler-dsml-sql-381611.sql_project.payments` t2
ON
  t1.order_id=t2.order_id) t6
  group by months,payment_type
  order by months,payment_type
```

Row	months	payment_type	no_of_orders
1	1	UPI	1715
2	1	credit_card	6103
3	1	debit_card	118
4	1	voucher	477
5	2	UPI	1723
6	2	credit_card	6609
7	2	debit_card	82
8	2	voucher	424
9	3	UPI	1942
10	3	credit_card	7707