RETAIL CASE STUDY

RETAIL CASE STUDY:SOLUTION

Solution

1 Exploratory Analysis

1.1. Data type of columns in a table

The schema of imported dataset in Google Big Query is as below

Customer

Field name	Type
customer_id	STRING
customer_unique_id	STRING
customer_zip_code_prefix	INTEGER
customer_city	STRING
customer_state	STRING

ORDERS

Field name	Type
order_id	STRING
customer_id	STRING
order_status	STRING
order_purchase_timestamp	TIMESTAMP
order_approved_at	TIMESTAMP
order_delivered_carrier_date	TIMESTAMP
order_delivered_customer_date	TIMESTAMP
order_estimated_delivery_date	TIMESTAMP

ORDER_ITEMS

Field name	Type
order_id	STRING
order_item_id	INTEGER
product_id	STRING
seller_id	STRING
shipping_limit_date	TIMESTAMP
price	FLOAT
freight_value	FLOAT

ORDER_REVIEWS

Field name	Type
review_id	STRING
order_id	STRING
review_score	INTEGER
review_comment_title	STRING
review_creation_date	TIMESTAMP
review_answer_timestamp	TIMESTAMP

SELLERS

Field name	Type
seller_id	STRING
seller_zip_code_prefix	INTEGER
seller_city	STRING
seller_state	STRING

PRODUCTS

Field name	Type
product_id	STRING
product_category	STRING
product_name_length	INTEGER
product_description_length	INTEGER
product_photos_qty	INTEGER
product_weight_g	INTEGER
product_length_cm	INTEGER
product_height_cm	INTEGER

GEOLOCATION

Field name	Type
geolocation_zip_code_prefix	INTEGER
geolocation_lat	FLOAT
geolocation_lng	FLOAT
geolocation_city	STRING
geolocation_state	STRING

PAYMENTS

Field name	Type
order_id	STRING
payment_sequential	INTEGER
payment_type	STRING
payment_installments	INTEGER
payment_value	FLOAT

1.2. Time period for which the data is given

```
SELECT min(order_purchase_timestamp ) min,
max(order_purchase_timestamp ) max
FROM `sodium-diode-364810.STORE.ORDERS`
```

SQL output

min	max
2016-09-04 21:15:19 UTC	2018-10-17 17:30:18 UTC

The dataset contains orders between dates 2016-09-04 and 2018-10-17.
Since the data is time dependent ,time series analysis is best suited for this data.

1.3. Cities and States covered in the dataset

The location of customers having orders is given by sql

```
SELECT DISTINCT
customer_state,
customer_city,
count(distinct(customer_city)) over(partition by customer_state)
ct_cnt
FROM `sodium-diode-364810.STORE.CUST` C
join `sodium-diode-364810.STORE.ORDERS` O
ON C.customer_id = O.customer_id
ORDER BY st_cnt desc,customer_state,customer_city
```

The SQL output is

customer_state	customer_city	ct_cnt
MG	abadia dos dourados	745
MG	abaete	745
MG	abre campo	745
MG	acaiaca	745
MG	acucena	745
MG	agua comprida	745
MG	aguas formosas	745
MG	aguas vermelhas	745
MG	aimores	745

Using data from above, the top5 customer states for city count.

customer_state	ct_cnt
MG	745
SP	629
RS	379
PR	364
BA	353

The bottom 5 customer states for city count.

customer_state	ct_cnt
AC	8
AP	6
DF	6
AM	5
RR	2

Total customer cities is 4310

2 In-depth Exploration:

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?

The SQL used is

```
SELECT
EXTRACT(YEAR from order_purchase_timestamp) year,
EXTRACT(MONTH from order_purchase_timestamp) month,
COUNT(DISTINCT(order_id)) o_count
FROM `sodium-diode-364810.STORE.ORDERS`
GROUP BY month,year
ORDER BY year,month
```

Output is

year	month	o_count
2016	9	4
2016	10	324
2016	12	1
2017	1	800
2017	2	1780
2017	3	2682
2017	4	2404
2017	5	3700
2017	6	3245
2017	7	4026
2017	8	4331
2017	9	4285
2017	10	4631
2017	11	7544
2017	12	5673
2018	1	7269
2018	2	6728
2018	3	7211
2018	4	6939
2018	5	6873

2018	6	6167
2018	7	6292
2018	8	6512
2018	9	16
2018	10	4

The top 3 order count are in the months

year	month	o_count
2017	11	7544
2018	1	7269
2018	3	7211

- The Holiday season of Christmas-New Year time period has the highest continuous orders (Nov-2017 to Jan-2018)
- In Brazil the Summertime months from Dec to Mar is having high order count.

2.2 What time do Brazilian customers tend to buy?

(Dawn, Morning, Afternoon or Night)?

We use following hours for time classification

Day Time	Hours
Dawn	6 to 8
Morning	8 to 12
Afternoon	12 to 17
Evening	17 to 22
Night	22 to 6

The SQL used to get order count for different hours is

```
SELECT
COUNT(DISTINCT(order_id)) cnt,
CASE WHEN EXTRACT(HOUR from order_purchase_timestamp)
BETWEEN 6 and 7.99 THEN 'Dawn'
WHEN EXTRACT(HOUR from order_purchase_timestamp)
BETWEEN 8 and 11.99 THEN 'Morning'
WHEN EXTRACT(HOUR from order_purchase_timestamp)
BETWEEN 12 and 16.99 THEN 'Afternoon'
WHEN EXTRACT(HOUR from order_purchase_timestamp)
BETWEEN 17 and 21.99 THEN 'Evening'
ELSE 'Night'
END AS day_time
FROM `sodium-diode-364810.STORE.ORDERS`
```

```
GROUP BY day_time
order by 1 desc
```

	cnt	day_time
1	32211	Afternoon
2	30311	Evening
3	20507	Morning
4	14679	Night
5	1733	Dawn

From above output we see that highest orders are in Afternoon (12-17 hours) followed by Evening(17-22 hours) , Morning(8-12 hours) . So more stocks and staff are required in these busy hours.

3 Evolution of E-commerce orders in Brazil region

3.1 Get month on month orders by region, states

Solution Approach

- Created new table with state and region to which state belongs(refer map in section 3.3)
- Use inner SQL to join the require tables and select required columns
- Outer SQL calculates region total
- The top 3 regions with highest orders is Southeast, South, Northeast

```
SELECT x.*,
sum(ocnt) over(PARTITION BY year,month,region) reg_cnt,
sum(val) over(PARTITION BY year,month,region) reg_tot,
FROM (
SELECT
EXTRACT(YEAR from order_purchase_timestamp) year,
EXTRACT(MONTH from order_purchase_timestamp) month,
region,
state,
COUNT(DISTINCT(O.order_id)) ocnt,
SUM(payment_value) val
FROM `sodium-diode-364810.STORE.ORDERS` O
JOIN `sodium-diode-364810.STORE.PAYMENTS` P
ON O.order_id =P.order_id
JOIN `sodium-diode-364810.STORE.CUST`C
ON O.customer_id = C.customer_id
JOIN `sodium-diode-364810.STORE.REGION`R
ON C.customer_state = R.state
GROUP BY year,month,region,state
)x
ORDER BY year,month,region,state
```

The top 25 records of output are as given below.

year	month	region	state	ocnt	val	reg_cnt	reg_tot
2016	9	North	RR	1	136.23	1	136.23
2016	9	South	RS	1	75.06	1	75.06
2016	9	Southeast	SP	1	40.95	1	40.95
2016	10	CentreWest	DF	6	1200.11	18	3377.24
2016	10	CentreWest	GO	9	1223.06	18	3377.24
2016	10	CentreWest	MT	3	954.07	18	3377.24
2016	10	North	PA	4	1283.09	5	1352.11
2016	10	North	RR	1	69.02	5	1352.11
2016	10	Northeast	AL	2	129.9	34	7372.44
2016	10	Northeast	BA	4	995.34	34	7372.44
2016	10	Northeast	CE	8	2011.77	34	7372.44
2016	10	Northeast	MA	4	998.85	34	7372.44
2016	10	Northeast	PB	1	74.74	34	7372.44
2016	10	Northeast	PE	7	1688.49	34	7372.44
2016	10	Northeast	PI	1	246.09	34	7372.44
2016	10	Northeast	RN	4	881.34	34	7372.44
2016	10	Northeast	SE	3	345.92	34	7372.44
2016	10	South	PR	19	2580.35	54	10026.41
2016	10	South	RS	24	4715.64	54	10026.41
2016	10	South	SC	11	2730.42	54	10026.41
2016	10	Southeast	ES	4	1067.14	213	36962.28
2016	10	Southeast	MG	40	5642.97	213	36962.28
2016	10	Southeast	RJ	56	13407.58	213	36962.28
2016	10	Southeast	SP	113	16844.59	213	36962.28

3.2 How are customers distributed in Brazil

```
SELECT
customer_state,
count(customer_id) cnt
FROM `sodium-diode-364810.STORE.CUST`
group by customer_state
order by cnt desc,customer_state
```

The output is

customer_state	cnt
SP	41746
RJ	12852
MG	11635
RS	5466
PR	5045
SC	3637
BA	3380
DF	2140
ES	2033

GO	2020
PE	1652
CE	1336
PA	975
MT	907
MA	747
MS	715
PB	536
PI	495
RN	485
AL	413
SE	350
TO	280
RO	253
AM	148
AC	81
AP	68
RR	46

SQL below uses region info to get clearer picture

```
SELECT
region,
customer_state,
count(customer_id) cnt
FROM `sodium-diode-364810.STORE.CUST`
join `sodium-diode-364810.STORE.REGION`
ON customer_state= state
group by region,customer_state
order by cnt desc
```

The output is as below. Most customers are from Southeast,South regions

CentreWest	customer_state	cnt
Southeast	SP	41746
Southeast	RJ	12852
Southeast	MG	11635
South	RS	5466
South	PR	5045
South	SC	3637
Northeast	BA	3380
CentreWest	DF	2140
Southeast	ES	2033
CentreWest	GO	2020
Northeast	PE	1652
Northeast	CE	1336
North	PA	975
CentreWest	MT	907
Northeast	MA	747
CentreWest	MS	715
Northeast	PB	536
Northeast	PI	495

Northeast	RN	485
Northeast	AL	413
Northeast	SE	350
North	TO	280
North	RO	253
North	AM	148
North	AC	81
North	AP	68
North	RR	46

3.3 Map of Brazil Region and States

Source: https://en.m.wikipedia.org/wiki/Regions_of_Brazil

The Region and States of Brazil are as given in map below

4 Impact on Economy:

Analyze the money movemented 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)

Solution : Create CTE with one query for 2017 (Y17) and another for 2018(Y18)

```
with y17 as (
SELECT
EXTRACT(YEAR from order_purchase_timestamp) year,
EXTRACT(MONTH from order_purchase_timestamp) month,
SUM(payment_value) val
FROM `sodium-diode-364810.STORE.ORDERS` o
JOIN `sodium-diode-364810.STORE.PAYMENTS` P
ON O.order_id =P.order_id
where EXTRACT(YEAR from order_purchase_timestamp) = 2017
and EXTRACT(MONTH from order_purchase_timestamp) between 01
and 08
GROUP BY year,month
),
y18 as (
SELECT
EXTRACT(YEAR from order_purchase_timestamp) year,
EXTRACT(MONTH from order_purchase_timestamp) month,
SUM(payment_value) val
FROM `sodium-diode-364810.STORE.ORDERS` o
JOIN `sodium-diode-364810.STORE.PAYMENTS` P
ON O.order_id =P.order_id
where EXTRACT(YEAR from order_purchase_timestamp) = 2018
and EXTRACT(MONTH from order_purchase_timestamp) between 01
and 08
GROUP BY year,month
)
select y17.month,
y17.val v17,
y18.val v18,
round(( y18.val- y17.val)*100/ y17.val,2) yoy_growth
from y17 join y18
on y17.month = y18.month
order by y17.month
```

SQL output is

month	v17	v18	yoy_growth
1	138488.04	1115004.18	705.13
2	291908.01	992463.34	239.99
3	449863.6	1159652.119	157.78
4	417788.03	1160785.48	177.84

5	592918.82	1153982.15	94.63
6	511276.38	1023880.5	100.26
7	592382.92	1066540.75	80.04
8	674396.32	1022425.32	51.61

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

```
SELECT
customer_state,
avg(price) avg_price,
sum(price) sum_price,
avg(freight_value) avg_frei,
sum(freight_value) sum_frei
FROM `sodium-diode-364810.STORE.ORDER_ITEMS` i
join `sodium-diode-364810.STORE.ORDERS` o
on i.order_id = o.order_id
join `sodium-diode-364810.STORE.CUST` c
on c.customer_id = o.customer_id
group by customer_state
```

The SQL output is as below.

customer_state	avg_price	sum_price	avg_frei	sum_frei
SP	109.653629159729	5202955.05000274	15.1472753904191	718723.069999994
RJ	125.117818094519	1824092.66999965	20.9609239316825	305589.310000004
PR	119.004139372822	683083.760000037	20.5316515679443	117851.680000001
SC	124.653577586207	520553.340000022	21.4703687739463	89660.2600000001
DF	125.770548628429	302603.939999996	21.0413549459684	50625.4999999994
MG	120.748574148831	1585308.02999971	20.6301668063066	270853.460000007
PA	165.692416666667	178947.809999998	35.8326851851852	38699.3
BA	134.601208212687	511349.990000021	26.3639589365623	100156.679999999
GO	126.271731675954	294591.949999995	22.7668152593228	53114.9799999997
RS	120.33745308741	750304.020000042	21.735804330393	135522.740000002
TO	157.529333333333	49621.74	37.2466031746032	11732.68
AM	135.496	22356.84	33.2053939393939	5478.89
MA	145.204150485437	119648.22	38.2570024271845	31523.77
PE	145.508322259136	262788.029999994	32.9178626799557	59449.6599999999
ES	121.913701241135	275037.309999995	22.0587765957447	49764.5999999997
AL	180.889211711712	80314.8099999996	35.8436711711712	15914.59
MT	148.297184834123	156453.529999999	28.1662843601896	29715.4300000001
RN	156.965935727788	83034.9799999994	35.6523629489604	18860.1
CE	153.758261163735	227254.709999996	32.714201623816	48351.59
PI	160.358081180812	86914.0799999996	39.1479704797048	21218.2
MS	142.628376068376	116812.639999999	23.374884004884	19144.03
PB	191.475215946844	115268.079999999	42.723803986711	25719.73
RO	165.973525179856	46140.6400000002	41.0697122302158	11417.38

SE	153.041168831169	58920.8500000001	36.6531688311689	14111.47
AC	173.727717391304	15982.95	40.0733695652174	3686.75
RR	150.565961538462	7829.42999999999	42.9844230769231	2235.19
AP	164.320731707317	13474.3	34.0060975609756	2788.5

The top5 states with highest sum of freight is

customer_state	avg_frei	sum_frei
SP	15.15	7,18,723.07
RJ	20.96	3,05,589.31
MG	20.63	2,70,853.46
RS	21.74	1,35,522.74
PR	20.53	1,17,851.68

5 Analysis on sales, freight and delivery time:

5.2 Create columns:

time_to_delivery = order_purchase_timestamp - order_delivered_customer_date
diff_estimated_delivery = order_estimated_delivery_date - order_delivered_customer_date

Create new columns in ORDER table as below

```
ALTER TABLE `sodium-diode-364810.STORE.ORDERS`
ADD COLUMN time_to_delivery SET DATA TYPE NUMERIC,
ADD COLUMN diff_estimated_delivery SET DATA TYPE NUMERIC
```

5.1 Calculate days

Calculate days between purchasing, delivering and estimated delivery

Update data in new columns in ORDER table as below

```
UPDATE `sodium-diode-364810.STORE.ORDERS`
SET time_to_delivery =
TIMESTAMP_DIFF(order_delivered_customer_date,
order_purchase_timestamp,DAY) ,
diff_estimated_delivery =
TIMESTAMP_DIFF(order_estimated_delivery_date,
order_delivered_customer_date,DAY)
```

display data from new columns of table

```
SELECT
order_purchase_timestamp,
order_delivered_customer_date,
order_estimated_delivery_date,
time_to_delivery ,
diff_estimated_delivery
FROM `sodium-diode-364810.STORE.ORDER`
```

```
where order_delivered_customer_date is not null
limit 100
```

SQL output

order_purchase_timestamp	order_delivered_customer_date	order_estimated_delivery_date	time_to_deliv	diff_estimated_delivery
2017-05-15 11:50:53.000000 UTC	2017-05-16 10:21:52.000000 UTC	2017-05-24 00:00:00.000000 UTC	0	7
2018-06-18 12:59:42.000000 UTC	2018-06-19 12:43:27.000000 UTC	2018-06-28 00:00:00.000000 UTC	0	8
2018-05-14 12:20:06.000000 UTC	2018-05-15 12:17:46.000000 UTC	2018-05-25 00:00:00.000000 UTC	0	9
2018-05-18 15:03:19.000000 UTC	2018-05-19 12:28:30.000000 UTC	2018-05-29 00:00:00.000000 UTC	0	9
2017-06-19 08:19:45.000000 UTC	2017-06-19 21:07:52.000000 UTC	2017-06-30 00:00:00.000000 UTC	0	10
2017-05-31 12:00:35.000000 UTC	2017-06-01 10:28:24.000000 UTC	2017-06-13 00:00:00.000000 UTC	0	11
2017-07-04 11:37:47.000000 UTC	2017-07-05 08:09:26.000000 UTC	2017-07-17 00:00:00.000000 UTC	0	11
2017-11-16 13:54:08.000000 UTC	2017-11-17 13:49:40.000000 UTC	2017-11-29 00:00:00.000000 UTC	0	11
2018-06-28 14:34:48.000000 UTC	2018-06-29 14:12:18.000000 UTC	2018-07-12 00:00:00.000000 UTC	0	12
2018-02-02 15:26:38.000000 UTC	2018-02-03 15:05:56.000000 UTC	2018-02-20 00:00:00.000000 UTC	0	16
2017-05-29 13:21:46.000000 UTC	2017-05-30 08:06:56.000000 UTC	2017-06-19 00:00:00.000000 UTC	0	19
2017-05-31 11:11:55.000000 UTC	2017-06-01 08:34:36.000000 UTC	2017-06-27 00:00:00.000000 UTC	0	25
2018-06-26 20:48:33.000000 UTC	2018-06-27 17:31:53.000000 UTC	2018-07-25 00:00:00.000000 UTC	0	27
2018-08-14 19:48:58.000000 UTC	2018-08-16 14:39:44.000000 UTC	2018-08-17 00:00:00.000000 UTC	1	0
2018-07-31 04:03:02.000000 UTC	2018-08-02 01:26:45.000000 UTC	2018-08-03 00:00:00.000000 UTC	1	0
2018-08-13 16:15:03.000000 UTC	2018-08-15 11:18:46.000000 UTC	2018-08-16 00:00:00.000000 UTC	1	0
2018-08-13 20:45:54.000000 UTC	2018-08-15 12:33:38.000000 UTC	2018-08-16 00:00:00.000000 UTC	1	0
2018-07-31 18:30:33.000000 UTC	2018-08-02 18:24:56.000000 UTC	2018-08-03 00:00:00.000000 UTC	1	0

5.3 Group data

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

```
SELECT customer_state,
avg(freight_value) av_freight,
avg(time_to_delivery) av_time ,
avg(diff_estimated_delivery) av_diff
FROM `sodium-diode-364810.STORE.CUST` c
join `sodium-diode-364810.STORE.ORDER` o
ON c.customer_id = o.customer_id
JOIN `sodium-diode-364810.STORE.ORDER_ITEMS` i
on o.order_id = i.order_id
GROUP BY customer_state
```

customer_state	av_freight	av_time	av_diff
PR	20.5316515679442	11.4807930607187	12.5338998052753
SP	15.1472753904192	8.25960855241911	10.2655943845144
RJ	20.9609239316826	14.6893821575004	11.1444931429379
MG	20.6301668063067	11.5155221800727	12.3971510412635
CE	32.714201623816	20.5371669004208	10.2566619915849
BA	26.3639589365623	18.7746402389356	10.1194678251425
PE	32.9178626799557	17.7920962199313	12.5521191294387
RO	41.0697122302158	19.2820512820513	19.0805860805861
PA	35.8326851851851	23.3017077798862	13.3747628083492

RS	21.735804330393	14.7082993640959	13.2030001630523
AL	35.8436711711712	23.9929742388759	7.97658079625294
SC	21.4703687739463	14.5209858467545	10.6688628599317
DF	21.0413549459684	12.5014861995754	11.2747346072187
PI	39.1479704797048	18.9311663479924	10.6826003824092
GO	22.7668152593228	14.9481774264383	11.3728590250329
MT	28.1662843601896	17.5081967213115	13.6393442622951
AM	33.2053939393939	25.9631901840491	18.9754601226994
ES	22.0587765957447	15.1928089887641	9.7685393258427
TO	37.2466031746032	17.0032258064516	11.4612903225806
SE	36.6531688311688	20.9786666666667	9.16533333333334
MA	38.2570024271844	21.20375	9.11
PB	42.723803986711	20.1194539249147	12.1501706484642
MS	23.374884004884	15.1072749691739	10.3378545006165
RN	35.6523629489603	18.873320537428	13.0556621880998
AP	34.0060975609756	27.7530864197531	17.4444444444444
RR	42.9844230769231	27.8260869565217	17.4347826086957
AC	40.0733695652174	20.3296703296703	20.010989010989

5.4 Sort the data

Sort data to get the following:

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

Top 5 states with highest/lowest average freight value

```
SELECT customer_state,
avg(freight_value) av_freight,
avg(time_to_delivery) av_time ,
avg(diff_estimated_delivery) av_diff
FROM `sodium-diode-364810.STORE.CUST` c
join `sodium-diode-364810.STORE.ORDER` o
ON c.customer_id = o.customer_id
JOIN `sodium-diode-364810.STORE.ORDER_ITEMS` i
on o.order_id = i.order_id
GROUP BY customer_state
order by av_freight desc
Limit 5
```

SQL output : Top 5 states with highest/lowest average freight value

	customer_state	av_freight	av_time	av_diff
1	RR	42.984423076923079	27.826086956521731	17.434782608695649
2	PB	42.723803986710926	20.119453924914669	12.150170648464169
3	RO	41.069712230215835	19.282051282051277	19.080586080586091
4	AC	40.0733695652174	20.329670329670336	20.010989010989011
5	PI	39.1479704797048	18.931166347992349	10.6826003

5 states with lowest average freight value

```
SELECT customer_state,
avg(freight_value) av_freight,
avg(time_to_delivery) av_time ,
avg(diff_estimated_delivery) av_diff
FROM `sodium-diode-364810.STORE.CUST` c
join `sodium-diode-364810.STORE.ORDER` o
ON c.customer_id = o.customer_id
JOIN `sodium-diode-364810.STORE.ORDER_ITEMS` i
on o.order_id = i.order_id
GROUP BY customer_state
order by av_freight
Limit 5
```

SQL output : 5 states with lowest average freight value

	customer_state	av_freight	av_time	av_diff
1	SP	15.147275390419157	8.2596085524191079	10.265594384514349
2	PR	20.531651567944177	11.480793060718721	12.533899805275295
3	MG	20.630166806306732	11.515522180072711	12.397151041263486
4	RJ	20.960923931682593	14.689382157500356	11.144493142937932
5	DF	21.041354945968418	12.501486199575384	11.274734607

2. Top 5 states with highest/lowest average time to delivery

Top 5 states with highest average time to delivery

```
SELECT customer_state,
avg(freight_value) av_freight,
avg(time_to_delivery) av_time ,
avg(diff_estimated_delivery) av_diff
FROM `sodium-diode-364810.STORE.CUST` c
join `sodium-diode-364810.STORE.ORDER` o
ON c.customer_id = o.customer_id
JOIN `sodium-diode-364810.STORE.ORDER_ITEMS` i
on o.order_id = i.order_id
GROUP BY customer_state
order by av_time desc
Limit 5
```

	customer_state	av_freight	av_time	av_diff
1	RR	42.984423076923058	27.826086956521742	17.434782608695649
2	AP	34.006097560975604	27.753086419753085	17.444444444444446
3	AM	33.205393939393922	25.963190184049079	18.975460122699388
4	AL	35.843671171171188	23.992974238875881	7.9765807962529438

	customer_state	av_freight	av_time	av_diff
5	PA	35.832685185185113	23.301707779886186	13.37476280

5 states with lowest average time to delivery

```
SELECT customer_state,
avg(freight_value) av_freight,
avg(time_to_delivery) av_time ,
avg(diff_estimated_delivery) av_diff
FROM `sodium-diode-364810.STORE.CUST` c
join `sodium-diode-364810.STORE.ORDER` o
ON c.customer_id = o.customer_id
JOIN `sodium-diode-364810.STORE.ORDER_ITEMS` i
on o.order_id = i.order_id
GROUP BY customer_state
order by av_time
Limit 5
```

	customer_state	av_freight	av_time	av_diff
1	SP	15.147275390419157	8.2596085524191079	10.265594384514349
2	PR	20.531651567944177	11.480793060718721	12.533899805275295
3	MG	20.630166806306732	11.515522180072711	12.397151041263486
4	DF	21.041354945968418	12.501486199575384	11.274734607218665
5	SC	21.47036877394634	14.520985846754542	10.668862859

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

```
SELECT customer_state,
avg(freight_value) av_freight,
avg(time_to_delivery) av_time ,
avg(diff_estimated_delivery) av_diff
FROM `sodium-diode-364810.STORE.CUST` c
join `sodium-diode-364810.STORE.ORDER` o
ON c.customer_id = o.customer_id
JOIN `sodium-diode-364810.STORE.ORDER_ITEMS` i
on o.order_id = i.order_id
GROUP BY customer_state
order by av_diff desc
Limit 5
```

	customer_state	av_freight	av_time	av_diff
1	AC	40.073369565217391	20.329670329670332	20.010989010989
2	RO	41.069712230215814	19.282051282051277	19.080586080586084
3	AM	33.205393939393922	25.963190184049079	18.975460122699388

	customer_state	av_freight	av_time	av_diff
4	AP	34.006097560975604	27.753086419753085	17.444444444444446
5	RR	42.984423076923058	27.826086956521742	17.43478260

```
SELECT customer_state,
avg(freight_value) av_freight,
avg(time_to_delivery) av_time ,
avg(diff_estimated_delivery) av_diff
FROM `sodium-diode-364810.STORE.CUST` c
join `sodium-diode-364810.STORE.ORDER` o
ON c.customer_id = o.customer_id
JOIN `sodium-diode-364810.STORE.ORDER_ITEMS` i
on o.order_id = i.order_id
GROUP BY customer_state
order by av_diff
Limit 5
```

	customer_state	av_freight	av_time	av_diff
1	AL	35.843671171171188	23.992974238875881	7.9765807962529438
2	MA	38.257002427184446	21.203749999999978	9.1099999999999977
3	SE	36.653168831168827	20.978666666666658	9.1653333333333435
4	ES	22.058776595744703	15.19280898876406	9.7685393258427045
5	BA	26.363958936562288	18.774640238935639	10.11946782

6 Payment type analysis:

6.1 Month over Month count

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

```
SELECT
EXTRACT(YEAR from order_purchase_timestamp) year,
EXTRACT(MONTH from order_purchase_timestamp) month,
payment_type,
count(distinct(O.order_id)) dcnt
FROM `sodium-diode-364810.STORE.PAYMENTS` p
join `sodium-diode-364810.STORE.ORDER` o
on p.order_id = o.order_id
GROUP BY year,month,payment_type
ORDER BY year,month, dcnt desc
```

SQL output: top 20 records

year	month	payment_type	dcnt
2016	9	credit_card	3
2016	10	credit_card	253
2016	10	UPI	63
2016	10	voucher	11
2016	10	debit_card	2
2016	12	credit_card	1
2017	1	credit_card	582
2017	1	UPI	197
2017	1	voucher	33
2017	1	debit_card	9
2017	2	credit_card	1347
2017	2	UPI	398
2017	2	voucher	69
2017	2	debit_card	13
2017	3	credit_card	2008
2017	3	UPI	590
2017	3	voucher	123
2017	3	debit_card	31

The most popular payment type is credit_card,UPI

6.2 payment installments

2. Distribution of payment installments and count of orders

```
SELECT
payment_installments,
COUNT(DISTINCT(order_id)) cnt
FROM `sodium-diode-364810.STORE.PAYMENTS`
GROUP BY payment_installments
ORDER BY payment_installments
```

payment_installments	cnt
0	2
1	49060
2	12389
3	10443
4	7088
5	5234
6	3916
7	1623
8	4253
9	644
10	5315
11	23
12	133
13	16
14	15
15	74
16	5

17	8
18	27
20	17
21	3

The highest number of installments are 5 or less

7 Insights

1. Highest Customers are in South,SouthEast Region of Brazil
2. Highest revenue are is also from South,SouthEast Region of Brazil
3. Highest Sales are Summer Month and during Chirstmas,New Year
4. Highest orders are in Afternoon (12-17 hours) followed by Evening(17-22 hours) ,Morning(8-12 hours) .
5. Highest sum freight is from South,SouthEast Region of Brazil
6. North,NorthEast Region have highest average time to delivery
7. South,SouthEast Region have lowest average time to delivery
8. North Region has highest difference for estimated delivery
9. Popular Payment method is Credit Card,UPI
10. Many payments use 5 or less installments for payment

8 Recommendations

1. The average time to delivery, difference for estimated delivery for South,Southeast is the lowest. to increase profitability for regions these metrics can be reduced
2. Customers in South,Southeast are high, cross selling can be used here to futher promote sales
3. Other than South,Souheast the remaining region are less penetrated and potential for increasing customers
4. Stock should be increased in peak time of morning,afternoon,evening due to higher demand
5. There is scope to incease customers in the mornings(8-12hours)
6. The Summer Months and Christmas/New Year season has high sales. So Stock should be inceased in these months
7. During other months there can be promotions to increase sales
8. Make popular payment method of Credit Card,UPI easily available to be uset friendly
9. Most payments use 5 or less installments,schemes can devised for these for these customers to improve customer experience
10. The average price is less for South,Southeast region.So the sales can be improved if this can be done in other regions also