Brazilian E-commerce datasets analysis

Hello One Mount judges, I have something to say before you go through my exercise. Due to my intense workload at current company and my own symptoms of Covid-19, I could not finalize this excercise properly and perfectly in my opinion. This exercise is my own work and during the process I have learnt a lot from other reference exercises from friends around the world. My Python is only in basic level so if I have time and appropriate condition, I will surely learn more to gain knowledge and do better in the future.

Thank you!

In [1]:
```
import numpy as np
import pandas as pd
from scipy import stats
import os
import matplotlib.pyplot as plt
import seaborn as sns

from plotly.offline import init_notebook_mode, iplot
import plotly.graph_objs as go
from plotly import tools
```

In [2]:
```
# IMPORTING DATA

df_customers = pd.read_csv("olist_customers_dataset.csv")
df_geolocation = pd.read_csv("olist_geolocation_dataset.csv")
df_item = pd.read_csv("olist_order_items_dataset.csv")
df_order_payment = pd.read_csv("olist_order_payments_dataset.csv")
df_reviews = pd.read_csv("olist_order_reviews_dataset.csv")
df_orders = pd.read_csv("olist_orders_dataset.csv")
df_products = pd.read_csv("olist_products_dataset.csv")
df_sellers = pd.read_csv("olist_sellers_dataset.csv")
df_category = pd.read_csv("product_category_name_translation.csv")
```

In [3]:
```
# JOINING TABLES

df_consol = df_orders.merge(df_item, on='order_id', how='left')
df_consol = df_consol.merge(df_order_payment, on='order_id', how='outer', validate='m:m')
df_consol = df_consol.merge(df_reviews, on='order_id', how='outer')
df_consol = df_consol.merge(df_products, on='product_id', how='outer')
df_consol = df_consol.merge(df_customers, on='customer_id', how='outer')
df_consol = df_consol.merge(df_sellers, on='seller_id', how='outer')
```

In [4]:
```
df_consol.head()
```

Out[4]:

	order_id	customer_id	order_status	order_purchase_timestamp	order_approved_at	order_d
0	e481f51cbdc54678b7cc49136f2d6af7	9ef432eb6251297304e76186b10a928d	delivered	2017-10-02 10:56:33	2017-10-02 11:07:15	
1	e481f51cbdc54678b7cc49136f2d6af7	9ef432eb6251297304e76186b10a928d	delivered	2017-10-02 10:56:33	2017-10-02 11:07:15	
2	e481f51cbdc54678b7cc49136f2d6af7	9ef432eb6251297304e76186b10a928d	delivered	2017-10-02 10:56:33	2017-10-02 11:07:15	
3	128e10d95713541c87cd1a2e48201934	a20e8105f23924cd00833fd87daa0831	delivered	2017-08-15 18:29:31	2017-08-15 20:05:16	
4	0e7e841ddf8f8f2de2bad69267ecfbcf	26c7ac168e1433912a51b924fbd34d34	delivered	2017-08-02 18:24:47	2017-08-02 18:43:15	

5 rows × 39 columns

In [5]:
```
# Descriptive Statistics

df_consol.describe()
```

Out[5]:

	order_item_id	price	freight_value	payment_sequential	payment_installments	payment_value	review_score	product_name_l
count	118310.000000	118310.000000	118310.000000	119140.000000	119140.000000	119140.000000	118146.000000	116601.00
mean	1.196543	120.646603	20.032387	1.094737	2.941246	172.735135	4.015582	48.76
std	0.699489	184.109691	15.836850	0.730141	2.777848	267.776077	1.400436	10.03
min	1.000000	0.850000	0.000000	1.000000	0.000000	0.000000	1.000000	5.00
25%	1.000000	39.900000	13.080000	1.000000	1.000000	60.850000	4.000000	42.00
50%	1.000000	74.900000	16.280000	1.000000	2.000000	108.160000	5.000000	52.00

75%	1.000000	134.900000	21.180000	1.000000	4.000000	189.240000	5.000000	57.0
max	21.000000	6735.000000	409.680000	29.000000	24.000000	13664.080000	5.000000	76.0

1. Price

In [6]:

```
# Price distribution

df_consol['price'].fillna(-1, inplace=True)

plt.figure(figsize=(16,12))
plt.suptitle('Price Distributions', fontsize=22)

plt.subplot(221)
g = sns.histplot(df_consol['price'])
g.set_title("Price Distributions", fontsize=18)
g.set_xlabel("Price Values")
g.set_ylabel("Probability", fontsize=15)

plt.subplot(222)
g1 = sns.histplot(np.log(df_consol['price']+1.5))
g1.set_title("Price log Distributions", fontsize=18)
g1.set_xlabel("Price Values")
g1.set_ylabel("Probability", fontsize=15)
```

Out[6]: `Text(0, 0.5, 'Probability')`

2. Payments

In [7]:

```
df_order_payment['value_log'] = df_order_payment['payment_value'].apply(lambda x: np.log(x) if x > 0 else 0)
unique_ = df_order_payment['order_id'].nunique()
print("DataFrame shape: {}; unique order ids: {}".format(df_order_payment.shape, unique_))
df_order_payment.head()
```

```
DataFrame shape: (103886, 6); unique order ids: 99440
```

Out[7]:

	order_id	payment_sequential	payment_type	payment_installments	payment_value	value_log
0	b81ef226f3fe1789b1e8b2acac839d17	1	credit_card	8	99.33	4.598448
1	a9810da82917af2d9aefd1278f1dcfa0	1	credit_card	1	24.39	3.194173
2	25e8ea4e93396b6fa0d3dd708e76c1bd	1	credit_card	1	65.71	4.185251
3	ba78997921bbcdc1373bb41e913ab953	1	credit_card	8	107.78	4.680092
4	42fdf880ba16b47b59251dd489d4441a	1	credit_card	2	128.45	4.855540

In [8]:

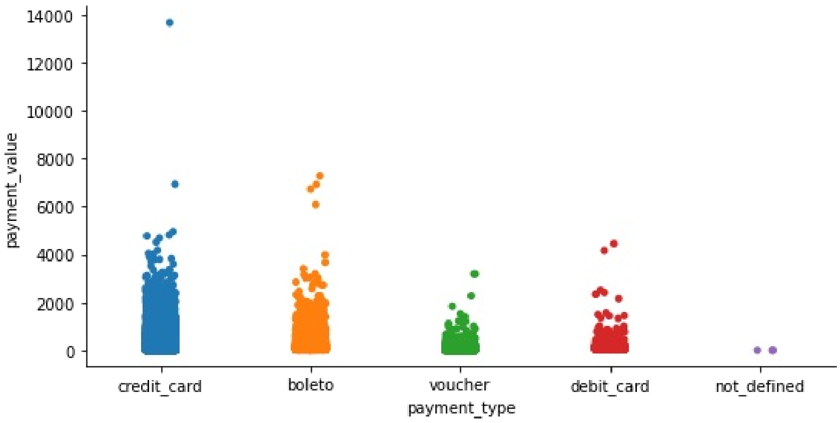
```
#Statistics

df_order_payment.describe()
```

Out[8]:

	payment_sequential	payment_installments	payment_value	value_log
count	103886.000000	103886.000000	103886.000000	103886.000000
mean	1.092679	2.853349	154.100380	4.597031
std	0.706584	2.687051	217.494064	0.937496
min	1.000000	0.000000	0.000000	-4.605170
25%	1.000000	1.000000	56.790000	4.039360
50%	1.000000	1.000000	100.000000	4.605170
75%	1.000000	4.000000	171.837500	5.146549
max	29.000000	24.000000	13664.080000	9.522526

In [9]:

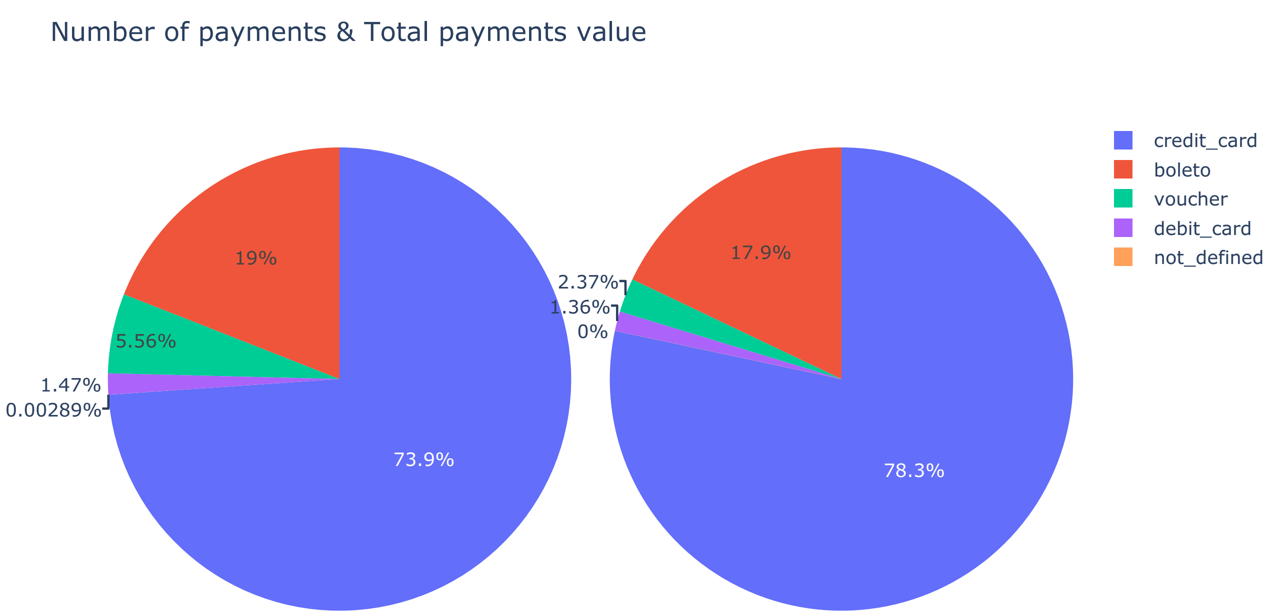
```
ax = sns.catplot(x="payment_type", y="payment_value",data=df_order_payment, aspect=2, height=3.8)
```



Value of orders mostly falls below 2000 currency unit

In [10]:

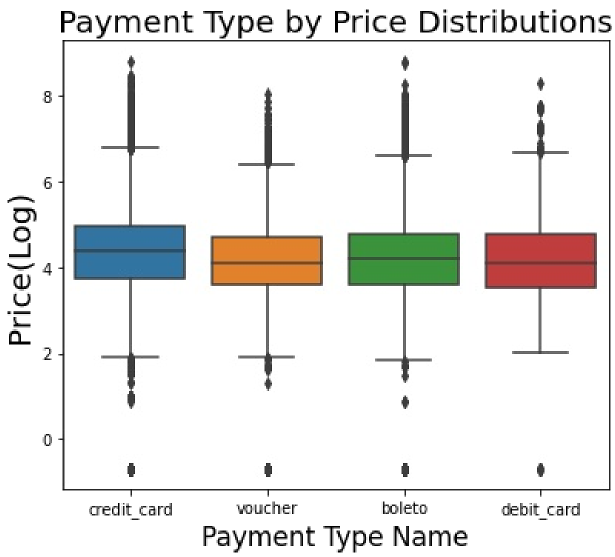
```
#Payment type distribution

method_count = df_order_payment['payment_type'].value_counts().to_frame().reset_index()
method_value = df_order_payment.groupby('payment_type')['payment_value'].sum().to_frame().reset_index()
# Plotly piechart
colors = None
trace1 = go.Pie(labels=method_count['index'], values=method_count['payment_type'],
                domain= {'x': [0, .48]}, marker=dict(colors= colors))
trace2 = go.Pie(labels=method_value['payment_type'], values=method_value['payment_value'],
                domain= {'x': [0.52, 1]}, marker=dict(colors=colors))
layout = dict(title= "Number of payments & Total payments value",
              height=520, width=850,)
fig = dict(data=[trace1, trace2], layout=layout)
iplot(fig)
```



There are 4 main payment types and credit card is the most used method for payment. 73.9% of payment comes from credit card and only 1.47% comes from debit card. From my findings, boleto is a payment method supplied by Brazilian Federation of Banks and it is very common in Brasil. Voucher is not usually used by customers and therefore it has no impact to the sales of products. The company should focus on improving the effectiveness of these vouchers for customer retention

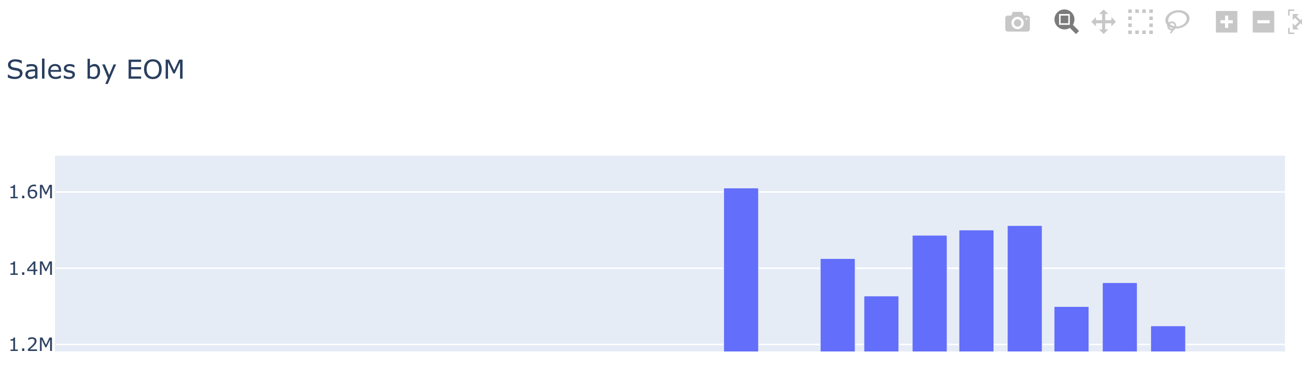
In [11]:

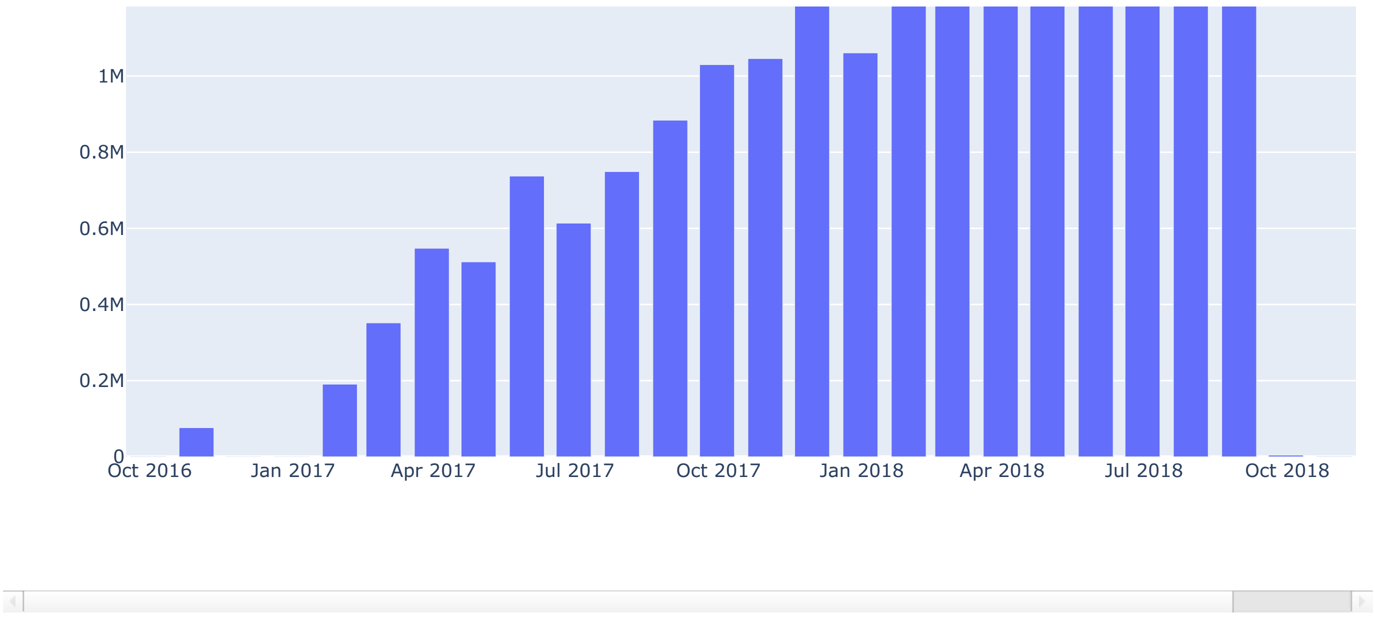
```
#Payment type by price distributions

total = len(df_consol)

plt.figure(figsize=(14,6))

#create price log
df_consol['price_log'] = np.log(df_consol['price'] + 1.5)

plt.subplot(122)
g = sns.boxplot(x='payment_type', y='price_log', data=df_consol[df_consol['payment_type'] != 'not_defined'])
g.set_title("Payment Type by Price Distributions", fontsize=20)
g.set_xlabel("Payment Type Name", fontsize=17)
g.set_ylabel("Price(Log)", fontsize=20)
plt.subplots_adjust(hspace = 0.7, top = 0.8)

plt.show()
```



In [12]:

```
%matplotlib inline
#Sales by end of month


df_consol['datetime'] =  pd.to_datetime(df_consol['order_purchase_timestamp'])
value_date = df_consol.groupby([df_consol['datetime'].dt.date])['payment_value'].sum()
freight_date = df_consol.groupby([df_consol['datetime'].dt.date])['freight_value'].sum()

value_month = df_consol[['datetime', 'payment_value']].copy()
value_month.set_index('datetime', inplace=True)
value_month = value_month.groupby(pd.Grouper(freq="M"))['payment_value'].sum()
trace = go.Bar(x= value_month.index, y= value_month.values)
layout = go.Layout(title='Sales by EOM', height=600, width=960)
fig = go.Figure(data=[trace], layout=layout)
iplot(fig)
```





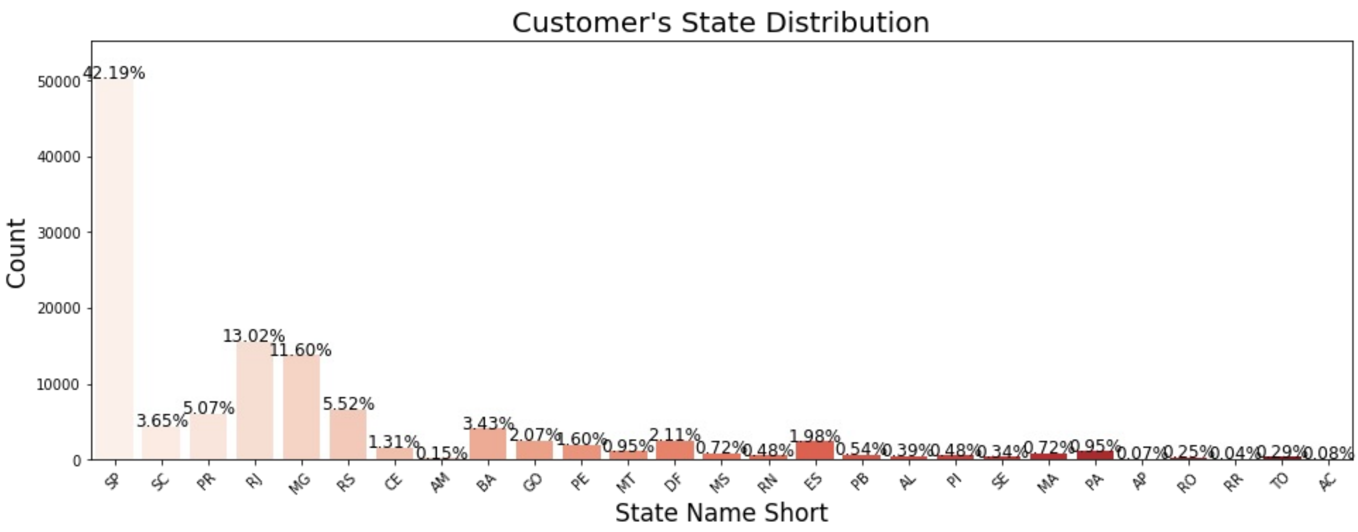
Sales from October 2016 to October 2017 has significantly increased, while from October 2017 to October 2018, the sales becomes stable.

3. Customers

In [13]:

```
#Customer's Location Distribution

plt.figure(figsize=(16,12))

plt.subplot(212)
g = sns.countplot(x='customer_state', data=df_consol, orient='h', palette = 'Reds')
g.set_title("Customer's State Distribution", fontsize=20)
g.set_xlabel("State Name Short", fontsize=17)
g.set_ylabel("Count", fontsize=17)
g.set_xticklabels(g.get_xticklabels(),rotation=45)
sizes = []
for p in g.patches:
    height = p.get_height()
    sizes.append(height)
    g.text(p.get_x()+p.get_width()/2.,
           height + 3,
           '{:1.2f}%'.format(height/total*100),
           ha="center", fontsize=12)
g.set_ylim(0, max(sizes) * 1.1)
```

Out[13]: (0.0, 55291.50000000001)

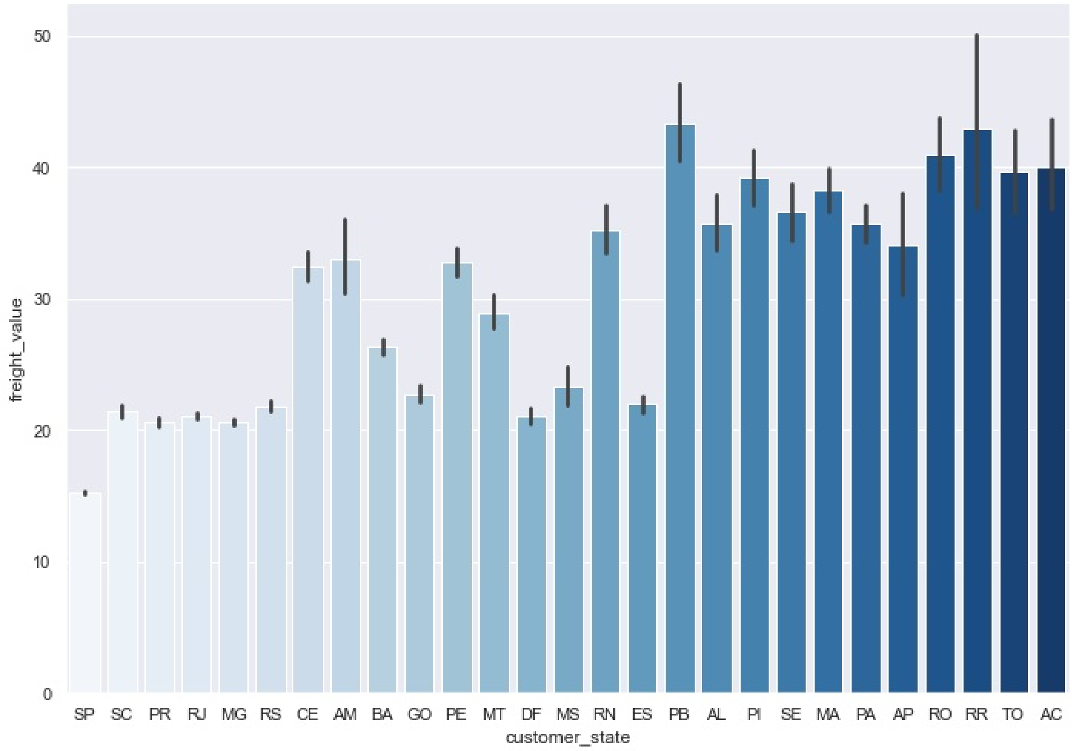


SP is the state that is where most customers located. There are many reason that can affect this fact: wealthiness, taste of customers, distance, etc. I will assume that the most common reason is because the shipping value is very high (maybe due to the distance from the supplier). To find out, I will plot the barchart.

In [14]:

```
#Relationship between States and freight value
```

```
sns.set(rc={'figure.figsize':(11.7,8.27)})

ax = sns.barplot(x="customer_state", y="freight_value",data=df_consol,palette = "Blues")
```



As we can see, SP has the cheapest freight value among all of the states. This is part of the reason why people in SP has the intention to buy the products. Therefore, to reach to other state markets, the company should have some policies to reduce the freight value to these states, or conduct campaigns to compensate the high value of delivery (distribute more vouchers to customers in these states since vouchers - as mentioned above - is not used effectively)

In [15]:

```
plt.figure(figsize=(16,12))

plt.suptitle('SELLER State Distributions', fontsize=22)

plt.subplot(221)
g2 = sns.boxplot(x='seller_state', y='price_log',
                 data=df_consol[df_consol['price'] != -1])
g2.set_title("Seller's State by Price", fontsize=20)
g2.set_xlabel("State Name Short", fontsize=17)
g2.set_ylabel("Price(Log)", fontsize=17)
g2.set_xticklabels(g2.get_xticklabels(),rotation=45)

plt.subplot(222)
g3 = sns.boxplot(x='seller_state', y='freight_value',
                 data=df_consol[df_consol['price'] != -1])
g3.set_title("Seller's State by Freight Value", fontsize=20)
g3.set_xlabel("State Name Short", fontsize=17)
g3.set_ylabel("Freight Value", fontsize=17)
g3.set_xticklabels(g3.get_xticklabels(),rotation=45)

plt.subplots_adjust(hspace = 0.5, top = 0.9)
```

5. Order Items

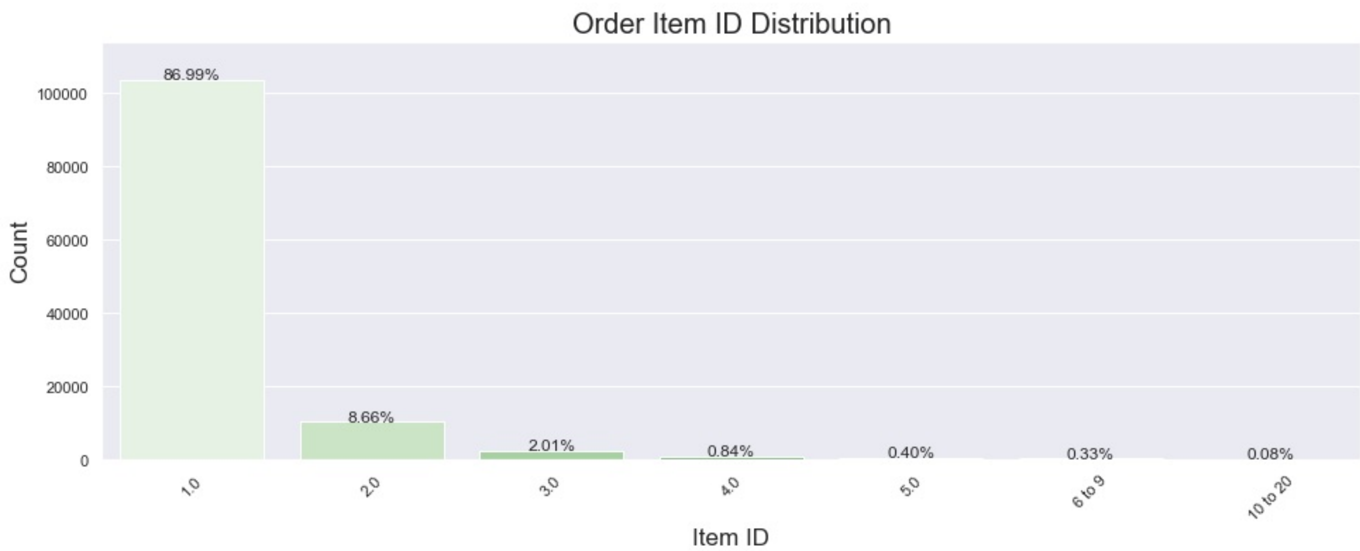
In [16]:

```
df_consol['order_short'] = df_consol['order_item_id'].copy()

df_consol.loc[df_consol['order_item_id'].isin([6,7,8,9]), 'order_short'] = '6 to 9'
df_consol.loc[(df_consol['order_item_id'] > 9), 'order_short'] = '10 to 20'
```

In [17]:

```
plt.figure(figsize=(16,12))

plt.subplot(212)
g = sns.countplot(x='order_short', data=df_consol, orient='h', palette = 'Greens')
g.set_title("Order Item ID Distribution", fontsize=20)
g.set_xlabel("Item ID", fontsize=17)
g.set_ylabel("Count", fontsize=17)
g.set_xticklabels(g.get_xticklabels(),rotation=45)
sizes = []
for p in g.patches:
    height = p.get_height()
    sizes.append(height)
    g.text(p.get_x()+p.get_width()/2.,
            height + 3,
            '{:1.2f}%'.format(height/total*100),
            ha="center", fontsize=12)
g.set_ylim(0, max(sizes) * 1.1)
```

Out[17]: (0.0, 114009.50000000001)



Only Item 1.0 which accounts for 87% of all the order items from customers. This means that the company only famous for its Item 1.0. Although it brings great profit for the company but beside this item, there are another 20 items and none of them can reach half of Item 1.0's orders amount. Therefore, to reduce the cost and maximize resources, the company should either focus on other Items to improve its quality, price and popularity, or cut down some unnecessary items to focus on core items. To find out more about the quality of these items, let's look at reviews

7. Reviews

In [18]:

```
round(pd.crosstab(df_consol['order_item_id'], df_consol['review_score'], normalize='index') *100,2)[:12].T
```

Out[18]:

order_item_id	1.0	2.0	3.0	4.0	5.0	6.0	7.0	8.0	9.0	10.0	11.0	12.0
review_score												
1.0	11.09	21.97	25.60	28.78	29.89	32.95	41.67	41.67	50.00	52.0	47.06	50.00
2.0	3.16	5.73	6.19	6.33	6.02	3.88	3.33	0.00	0.00	0.0	0.00	0.00
3.0	8.25	9.31	9.28	8.98	9.03	10.08	13.33	13.89	14.29	16.0	17.65	8.33
4.0	19.42	16.08	15.85	15.51	16.13	15.89	15.00	19.44	17.86	12.0	17.65	16.67
5.0	58.08	46.91	43.07	40.41	38.92	37.21	26.67	25.00	17.86	20.0	17.65	25.00

As we can see, Item 1.0 has the highest percentage of 5 stars review. The good review is reducing from Item 1 to Item 12, also aligning with the number of sales the company has, meaning customers do not often buy bad items.

In conclusion, the company has a significant increase in its GMV since Oct 2016 and keep the pace till today (data date). It has a wide range product pool and customers. However, Item 1.0 is the only well known item among 20 items and this should be an issue for the company. Moreover, promotion campaign is also not effective since not a lot of customers use vouchers. Besides, SP state has an enormous amount of customers due to the low price of delivery, while other states does not. To expand and scale up, the company should diversify its product portfolio to improve the sales of other products as well and target more customers in other states, along with making some proper marketing campaigns

The end.

```
In [ ]:
```