Table of Contents

Load Libraries

In [1]:

```
# Import PySpark related modules
import pyspark
from pyspark.rdd import RDD
from pyspark.sql import Row
from pyspark.sql import DataFrame
from pyspark.sql import SparkSession
from pyspark.sql import SQLContext
from pyspark.sql import functions
from pyspark.sql.functions import lit, desc, col, size, array_contains, isnan, u
from pyspark.sql.functions import *
from pyspark.sql.types import *
from pyspark import SparkConf, SparkContext
from pyspark.ml.evaluation import RegressionEvaluator
from pyspark.ml.regression import LinearRegression
from pyspark.ml.feature import StandardScaler
from pyspark.ml.feature import VectorAssembler

# Import other modules not related to PySpark
import os
import sys
import pandas as pd
from pandas import DataFrame
import numpy as np
import matplotlib.pyplot as plt
import matplotlib.ticker as mtick
import matplotlib
from mpl_toolkits.mplot3d import Axes3D
import math
```

```
from IPython.core.interactiveshell import InteractiveShell
from datetime import *
import seaborn as sns
import statistics as stats
# This helps auto print out the items without explixitly using 'print'
InteractiveShell.ast_node_interactivity = "all"
%matplotlib inline
from matplotlib.pyplot import figure
import numpy as np
from pandas.plotting import scatter_matrix
import warnings
warnings.filterwarnings("ignore")
```

Initialize pyspark framework

In [2]:

```
conf = pyspark.SparkConf().setAll([('spark.master', 'local[*]'),
                                   ('spark.app.name', 'Python Spark SQL Demo')])
spark = SparkSession.builder.config(conf=conf).getOrCreate()
```

Load data

In [3]:

```
!pwd
```

```
/home/work/ecommerce
```

In [4]:

```
!ls
```

```
'E-commerce EDA.ipynb'                 order_items_dataset.csv
'E-commerce Sales Forecast.ipynb'      order_payments_dataset.csv
 customer_reviews_dataset.csv          orders_dataset.csv
 customers_dataset.csv                 product_category_name_translation.csv
 geolocation_dataset.csv               products_dataset.csv
 launch.sh                             sellers_dataset.csv
```

In [5]:

```
!hadoop fs -mkdir /data
```

```
mkdir: `/data': File exists
```

In [6]:

```
!hadoop fs -copyFromLocal products_dataset.csv /data
```

```
copyFromLocal: `/data/products_dataset.csv': File exists
```

In [7]:

```
!hadoop fs -copyFromLocal product_category_name_translation.csv /data
```

```
copyFromLocal: `/data/product_category_name_translation.csv': File exists
```

In [8]:

```
!hadoop fs -copyFromLocal customers_dataset.csv /data
```

```
copyFromLocal: `/data/customers_dataset.csv': File exists
```

In [9]:

```
!hadoop fs -copyFromLocal sellers_dataset.csv /data
```

```
copyFromLocal: `/data/sellers_dataset.csv': File exists
```

In [10]:

```
!hadoop fs -copyFromLocal orders_dataset.csv /data
```

```
copyFromLocal: `/data/orders_dataset.csv': File exists
```

In [11]:

```
!hadoop fs -copyFromLocal order_payments_dataset.csv /data
```

```
copyFromLocal: `/data/order_payments_dataset.csv': File exists
```

In [12]:

```
!hadoop fs -copyFromLocal order_items_dataset.csv /data
```

```
copyFromLocal: `/data/order_items_dataset.csv': File exists
```

In [13]:

```
!hadoop fs -copyFromLocal geolocation_dataset.csv /data
```

```
copyFromLocal: `/data/geolocation_dataset.csv': File exists
```

In [14]:

```
!hadoop fs -copyFromLocal customer_reviews_dataset.csv /data
```

```
copyFromLocal: `/data/customer_reviews_dataset.csv': File exists
```

In [15]:

```
DATA_PATH="hdfs:///data/"
products_dataset = spark.read.csv(DATA_PATH+"products_dataset.csv", header=True,
product_category_name_translation = spark.read.csv(DATA_PATH+"product_category_n
customers_dataset = spark.read.csv(DATA_PATH+"customers_dataset.csv", header=Tru
sellers_dataset = spark.read.csv(DATA_PATH+"sellers_dataset.csv", header=True, i
orders_dataset = spark.read.csv(DATA_PATH+"orders_dataset.csv", header=True, inf
order_payments_dataset = spark.read.csv(DATA_PATH+"order_payments_dataset.csv",
order_items_dataset = spark.read.csv(DATA_PATH+"order_items_dataset.csv", header
geolocation_dataset = spark.read.csv(DATA_PATH+"geolocation_dataset.csv", header
customer_reviews_dataset = spark.read.csv(DATA_PATH+"customer_reviews_dataset.cs
```

Overview of Dataset

Data schema

In [16]:

```
print('Data overview')
products_dataset.printSchema()
```

```
Data overview
root
 |-- product_id: string (nullable = true)
 |-- product_category_name: string (nullable = true)
 |-- product_name_lenght: integer (nullable = true)
 |-- product_description_lenght: integer (nullable = true)
 |-- product_photos_qty: integer (nullable = true)
 |-- product_weight_g: integer (nullable = true)
 |-- product_length_cm: integer (nullable = true)
 |-- product_height_cm: integer (nullable = true)
 |-- product_width_cm: integer (nullable = true)
```

In [17]:
```
print('Data overview')
product_category_name_translation.printSchema()
```

```
Data overview
root
 |-- product_category_name: string (nullable = true)
 |-- product_category_name_english: string (nullable = true)
```

In [18]:
```
print('Data overview')
customers_dataset.printSchema()
```

```
Data overview
root
 |-- customer_id: string (nullable = true)
 |-- customer_unique_id: string (nullable = true)
 |-- customer_zip_code_prefix: integer (nullable = true)
 |-- customer_city: string (nullable = true)
 |-- customer_state: string (nullable = true)
```

In [19]:
```
print('Data overview')
sellers_dataset.printSchema()
```

```
Data overview
root
 |-- seller_id: string (nullable = true)
 |-- seller_zip_code_prefix: integer (nullable = true)
 |-- seller_city: string (nullable = true)
 |-- seller_state: string (nullable = true)
```

In [20]:
```
print('Data overview')
orders_dataset.printSchema()
```

```
Data overview
root
 |-- order_id: string (nullable = true)
 |-- customer_id: string (nullable = true)
 |-- order_status: string (nullable = true)
 |-- order_purchase_timestamp: string (nullable = true)
 |-- order_approved_at: string (nullable = true)
 |-- order_carrier_delivery_date: string (nullable = true)
 |-- order_customer_delivery_date: string (nullable = true)
 |-- order_estimated_delivery_date: string (nullable = true)
```

In [21]:
```
print('Data overview')
order_payments_dataset.printSchema()
```

```
Data overview
root
 |-- order_id: string (nullable = true)
 |-- payment_sequential: integer (nullable = true)
 |-- payment_type: string (nullable = true)
 |-- payment_installments: integer (nullable = true)
 |-- payment_value: double (nullable = true)
```

In [22]:
```
print('Data overview')
```

```
order_items_dataset.printSchema()
```

```
Data overview
root
 |-- order_id: string (nullable = true)
 |-- order_item_id: integer (nullable = true)
 |-- product_id: string (nullable = true)
 |-- seller_id: string (nullable = true)
 |-- shipping_limit_date: string (nullable = true)
 |-- price: double (nullable = true)
 |-- freight_value: double (nullable = true)
```

In [23]:

```
print('Data overview')
geolocation_dataset.printSchema()
```

```
Data overview
root
 |-- geo_zip_code_prefix: integer (nullable = true)
 |-- geo_lat: double (nullable = true)
 |-- geo_lng: double (nullable = true)
 |-- geo_city: string (nullable = true)
 |-- geo_state: string (nullable = true)
```

In [24]:

```
print('Data overview')
customer_reviews_dataset.printSchema()
```

```
Data overview
root
 |-- review_id: string (nullable = true)
 |-- order_id: string (nullable = true)
 |-- survey_score: string (nullable = true)
 |-- survey_review_title: string (nullable = true)
 |-- survey_review_content: string (nullable = true)
 |-- survey_send_date: string (nullable = true)
 |-- survey_completion_date: string (nullable = true)
```

Columns overview

In [25]:

```
print('Columns overview')
pd.DataFrame(products_dataset.dtypes, columns = ['Column Name','Data type'])
```

```
Columns overview
```

Out[25]:

	Column Name	Data type
0	product_id	string
1	product_category_name	string
2	product_name_lenght	int
3	product_description_lenght	int
4	product_photos_qty	int
5	product_weight_g	int
6	product_length_cm	int

	Column Name	Data type
7	product_height_cm	int
8	product_width_cm	int

In [26]:

```
print('Columns overview')
pd.DataFrame(product_category_name_translation.dtypes, columns = ['Column Name',
```

```
Columns overview
```

Out[26]:

	Column Name	Data type
0	product_category_name	string
1	product_category_name_english	string

In [27]:

```
print('Columns overview')
pd.DataFrame(customers_dataset.dtypes, columns = ['Column Name','Data type'])
```

```
Columns overview
```

Out[27]:

	Column Name	Data type
0	customer_id	string
1	customer_unique_id	string
2	customer_zip_code_prefix	int
3	customer_city	string
4	customer_state	string

In [28]:

```
print('Columns overview')
pd.DataFrame(sellers_dataset.dtypes, columns = ['Column Name','Data type'])
```

```
Columns overview
```

Out[28]:

	Column Name	Data type
0	seller_id	string
1	seller_zip_code_prefix	int
2	seller_city	string
3	seller_state	string

In [29]:

```
print('Columns overview')
pd.DataFrame(orders_dataset.dtypes, columns = ['Column Name','Data type'])
```

```
Columns overview
```

Out[29]:

	Column Name	Data type
0	order_id	string
1	customer_id	string
2	order_status	string

	Column Name	Data type
3	order_purchase_timestamp	string
4	order_approved_at	string
5	order_carrier_delivery_date	string
6	order_customer_delivery_date	string
7	order_estimated_delivery_date	string

In [30]:
```
print('Columns overview')
pd.DataFrame(order_payments_dataset.dtypes, columns = ['Column Name','Data type'
```
```
Columns overview
```
Out[30]:

	Column Name	Data type
0	order_id	string
1	payment_sequential	int
2	payment_type	string
3	payment_installments	int
4	payment_value	double

In [31]:
```
print('Columns overview')
pd.DataFrame(order_items_dataset.dtypes, columns = ['Column Name','Data type'])
```
```
Columns overview
```
Out[31]:

	Column Name	Data type
0	order_id	string
1	order_item_id	int
2	product_id	string
3	seller_id	string
4	shipping_limit_date	string
5	price	double
6	freight_value	double

In [32]:
```
print('Columns overview')
pd.DataFrame(geolocation_dataset.dtypes, columns = ['Column Name','Data type'])
```
```
Columns overview
```
Out[32]:

	Column Name	Data type
0	geo_zip_code_prefix	int
1	geo_lat	double
2	geo_lng	double

	Column Name	Data type
3	geo_city	string
4	geo_state	string

In [33]:
```
print('Columns overview')
pd.DataFrame(customer_reviews_dataset.dtypes, columns = ['Column Name','Data typ
```

```
Columns overview
```

Out[33]:

	Column Name	Data type
0	review_id	string
1	order_id	string
2	survey_score	string
3	survey_review_title	string
4	survey_review_content	string
5	survey_send_date	string
6	survey_completion_date	string

Summary statistics for numeric variables

In [34]:
```
print('Data frame describe (string and numeric columns only):')
products_dataset.describe().toPandas()
```

```
Data frame describe (string and numeric columns only):
```

Out[34]:

	summary	product_id	product_category_name	product_name_lenght
0	count	32951	32341	32341
1	mean	None	None	48.47694876472589
2	stddev	None	None	10.245740725237287
3	min	00066f42aeeb9f3007548bb9d3f33c38	agro_industria_e_comercio	5
4	max	fffe9eeff12fcbd74a2f2b007dde0c58	utilidades_domesticas	76

In [35]:
```
print('Data frame describe (string and numeric columns only):')
product_category_name_translation.describe().toPandas()
```

```
Data frame describe (string and numeric columns only):
```

Out[35]:

	summary	product_category_name	product_category_name_english
0	count	71	71
1	mean	None	None
2	stddev	None	None
3	min	agro_industria_e_comercio	agro_industry_and_commerce
4	max	utilidades_domesticas	watches_gifts

In [36]:

```
print('Data frame describe (string and numeric columns only):')
customers_dataset.describe().toPandas()
```

```
Data frame describe (string and numeric columns only):
```

Out[36]:

	summary	customer_id	customer_unique_id	customer_
0	count	99441	99441	
1	mean	None	None	351
2	stddev	None	None	297
3	min	00012a2ce6f8dcda20d059ce98491703	0000366f3b9a7992bf8c76cfdf3221e2	
4	max	ffffe8b65bbe3087b653a978c870db99	ffffd2657e2aad2907e67c3e9daecbeb	

In [37]:

```
print('Data frame describe (string and numeric columns only):')
sellers_dataset.describe().toPandas()
```

```
Data frame describe (string and numeric columns only):
```

Out[37]:

	summary	seller_id	seller_zip_code_prefix	seller_city	seller_state
0	count	3095	3095	3095	3095
1	mean	None	32291.059450726978	4482255.0	None
2	stddev	None	32713.45382950901	None	None
3	min	0015a82c2db000af6aaaf3ae2ecb0532	1001	04482255	AC
4	max	ffff564a4f9085cd26170f4732393726	99730	xaxim	SP

In [38]:

```
print('Data frame describe (string and numeric columns only):')
orders_dataset.describe().toPandas()
```

```
Data frame describe (string and numeric columns only):
```

Out[38]:

	summary	order_id	customer_id	order_statu
0	count	99441	99441	994
1	mean	None	None	No
2	stddev	None	None	No
3	min	00010242fe8c5a6d1ba2dd792cb16214	00012a2ce6f8dcda20d059ce98491703	approve
4	max	fffe41c64501cc87c801fd61db3f6244	ffffe8b65bbe3087b653a978c870db99	unavailab

In [39]:

```
print('Data frame describe (string and numeric columns only):')
order_payments_dataset.describe().toPandas()
```

```
Data frame describe (string and numeric columns only):
```

Out[39]:

	summary	order_id	payment_sequential	payment_type	payment_i

	summary	order_id	payment_sequential	payment_type	payment_i
0	count	103886	103886	103886	
1	mean	None	1.0926785129853878	None	2.853348
2	stddev	None	0.7065837791949958	None	2.687050
3	min	00010242fe8c5a6d1ba2dd792cb16214	1	boleto	
4	max	fffe41c64501cc87c801fd61db3f6244	29	voucher	

```
In [40]:
print('Data frame describe (string and numeric columns only):')
order_items_dataset.describe().toPandas()
```

```
Data frame describe (string and numeric columns only):
```

Out[40]:

	summary	order_id	order_item_id	pı
0	count	112650	112650	
1	mean	None	1.1978339991122948	
2	stddev	None	0.7051240313951721	
3	min	00010242fe8c5a6d1ba2dd792cb16214	1	00066f42aeeb9f3007548bb9
4	max	fffe41c64501cc87c801fd61db3f6244	21	fffe9eeff12fcbd74a2f2b00

```
In [41]:
print('Data frame describe (string and numeric columns only):')
geolocation_dataset.describe().toPandas()
```

```
Data frame describe (string and numeric columns only):
```

Out[41]:

	summary	geo_zip_code_prefix	geo_lat	geo_lng	geo_city	geo_state
0	count	1000163	1000163	1000163	1000163	1000163
1	mean	36574.16646586607	-21.17615291038385	-46.39054132093571	None	None
2	stddev	30549.335710320098	5.715866308822862	4.269748306619432	None	None
3	min	1001	-36.6053744107061	-101.46676644931476	* cidade	AC
4	max	99990	45.06593318269697	121.10539381057764	óleo	TC

```
In [42]:
print('Data frame describe (string and numeric columns only):')
customer_reviews_dataset.describe().toPandas()
```

```
Data frame describe (string and numeric columns only):
```

Out[42]:

	summary	review_id	order_id	survey_score	survey_review_title	survey_
0	count	105188	102859	102692	12176	
1	mean	4.5	0.0	4.071667849964501	3.1554493965252365E10	1.11111
2	stddev	0.7071067811865476	0.0	1.386648877434681	5.616554832455847E11	

	summary	review_id	order_id	survey_score	survey_review_title	survey_
3	min	"		"		
4	max	👍🏼👏🏼👏🏼"	visando sempre o melhor para os clientes!	seria mais coerente."	🔟	😡😡

Show data and data count

In [43]:

```
print(f'There are total {products_dataset.count()} row, Let print first 2 data r
products_dataset.limit(2).toPandas()
```

```
There are total 32951 row, Let print first 2 data rows:
```

Out[43]:

	product_id	product_category_name	product_name_lenght	product_de
0	1e9e8ef04dbcff4541ed26657ea517e5	perfumaria	40	
1	3aa071139cb16b67ca9e5dea641aaa2f	artes	44	

In [44]:

```
print(f'There are total {product_category_name_translation.count()} row, Let pri
product_category_name_translation.limit(2).toPandas()
```

```
There are total 71 row, Let print first 2 data rows:
```

Out[44]:

	product_category_name	product_category_name_english
0	beleza_saude	health_beauty
1	informatica_acessorios	computers_accessories

In [45]:

```
print(f'There are total {customers_dataset.count()} row, Let print first 2 data
customers_dataset.limit(2).toPandas()
```

```
There are total 99441 row, Let print first 2 data rows:
```

Out[45]:

	customer_id	customer_unique_id	customer_zip_code_
0	06b8999e2fba1a1fbc88172c00ba8bc7	861eff4711a542e4b93843c6dd7febb0	1
1	18955e83d337fd6b2def6b18a428ac77	290c77bc529b7ac935b93aa66c333dc3	

In [46]:

```
print(f'There are total {sellers_dataset.count()} row, Let print first 2 data ro
sellers_dataset.limit(2).toPandas()
```

```
There are total 3095 row, Let print first 2 data rows:
```

Out[46]:

	seller_id	seller_zip_code_prefix	seller_city	seller_state
0	3442f8959a84dea7ee197c632cb2df15	13023	campinas	SP
1	d1b65fc7debc3361ea86b5f14c68d2e2	13844	mogi guacu	SP

In [47]:

```
print(f'There are total {orders_dataset.count()} row, Let print first 2 data row
orders_dataset.limit(2).toPandas()
```

```
There are total 99441 row, Let print first 2 data rows:
```

Out[47]:

	order_id	customer_id	order_status	order_p
0	e481f51cbdc54678b7cc49136f2d6af7	9ef432eb6251297304e76186b10a928d	delivered	
1	53cdb2fc8bc7dce0b6741e2150273451	b0830fb4747a6c6d20dea0b8c802d7ef	delivered	

In [48]:

```
print(f'There are total {order_payments_dataset.count()} row, Let print first 2
order_payments_dataset.limit(2).toPandas()
```

```
There are total 103886 row, Let print first 2 data rows:
```

Out[48]:

	order_id	payment_sequential	payment_type	payment_installments
0	b81ef226f3fe1789b1e8b2acac839d17	1	credit_card	8
1	a9810da82917af2d9aefd1278f1dcfa0	1	credit_card	1

In [49]:

```
print(f'There are total {order_items_dataset.count()} row, Let print first 2 dat
order_items_dataset.limit(2).toPandas()
```

```
There are total 112650 row, Let print first 2 data rows:
```

Out[49]:

	order_id	order_item_id	product_id	
0	00010242fe8c5a6d1ba2dd792cb16214	1	4244733e06e7ecb4970a6e2683c13e61	4843
1	00018f77f2f0320c557190d7a144bdd3	1	e5f2d52b802189ee658865ca93d83a8f	dd7d

In [50]:

```
print(f'There are total {geolocation_dataset.count()} row, Let print first 2 dat
geolocation_dataset.limit(2).toPandas()
```

```
There are total 1000163 row, Let print first 2 data rows:
```

Out[50]:

	geo_zip_code_prefix	geo_lat	geo_lng	geo_city	geo_state
0	1037	-23.545621	-46.639292	sao paulo	SP
1	1046	-23.546081	-46.644820	sao paulo	SP

In [51]:

```
print(f'There are total {customer_reviews_dataset.count()} row, Let print first
customer_reviews_dataset.limit(2).toPandas()
```

```
There are total 105189 row, Let print first 2 data rows:
```

Out[51]:

	review_id	order_id	survey_score	survey
0	7bc2406110b926393aa56f80a40eba40	73fc7af87114b39712e6da79b0a377eb	4	
1	80e641a11e56f04c1ad469d5645fdfde	a548910a1c6147796b98fdf73dbeba33	5	

Correlations

Checking Correlations between independent variables

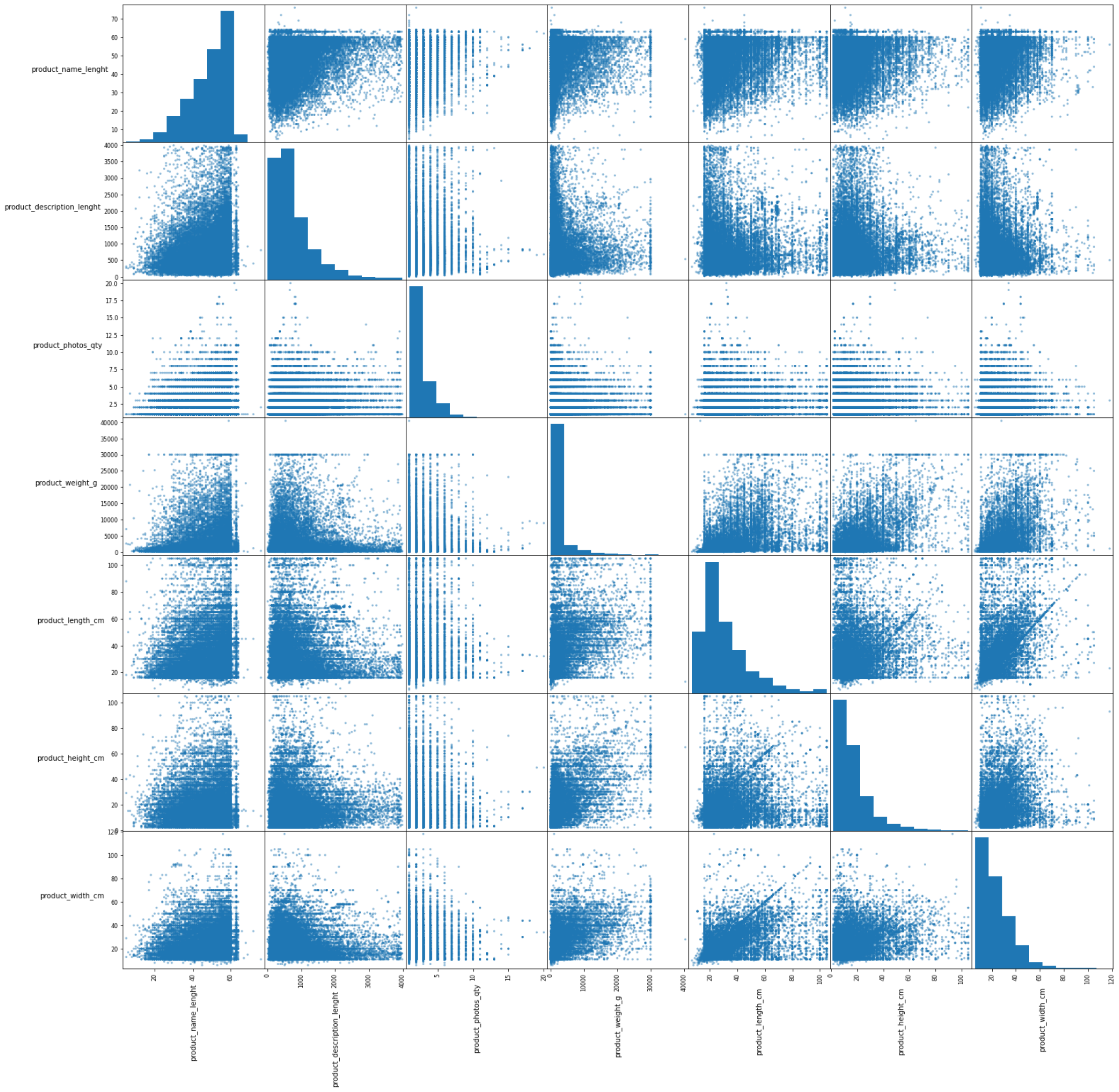
In [52]:

```
# Merge these two dataframes together: products_dataset, product_category_name_t
df_merge_product_and_category = products_dataset.join(product_category_name_tran
df_merge_product_and_category = df_merge_product_and_category.drop('product_cate
df_merge_product_and_category = df_merge_product_and_category.drop_duplicates(['
df_merge_product_and_category = df_merge_product_and_category.dropna()

df_merge_product_and_category.show(2)
```

```
+--------------------+-------------------+--------------------------+-----------
-------+-----------------+-----------------+------------------+-----------------+--
---------------------------+
|          product_id|product_name_lenght|product_description_lenght|product_pho
tos_qty|product_weight_g|product_length_cm|product_height_cm|product_width_cm|pr
oduct_category_name_english|
+--------------------+-------------------+--------------------------+-----------
-------+-----------------+-----------------+------------------+-----------------+--
---------------------------+
|00e4ded51458037ec...|                 25|                       978|
3|             1400|                 25|                 15|               25|
computers_accesso...|
|03d7ad0ce97624c93...|                 48|                       259|
1|              249|                 19|                 13|               15|
perfumery|
+--------------------+-------------------+--------------------------+-----------
-------+-----------------+-----------------+------------------+-----------------+--
---------------------------+
only showing top 2 rows
```

In [53]:

```
# Checking Correlations between independent variables
numeric_features = [t[0] for t in df_merge_product_and_category.dtypes if t[1] =
numeric_data = df_merge_product_and_category.select(numeric_features).toPandas()

axs = scatter_matrix(numeric_data, figsize=(25, 25));

# Rotate axis labels and remove axis ticks
n = len(numeric_data.columns)
for i in range(n):
    v = axs[i, 0]
    v.yaxis.label.set_rotation(0)
    v.yaxis.label.set_ha('right')
    h = axs[n-1, i]
    h.xaxis.label.set_rotation(90)
```

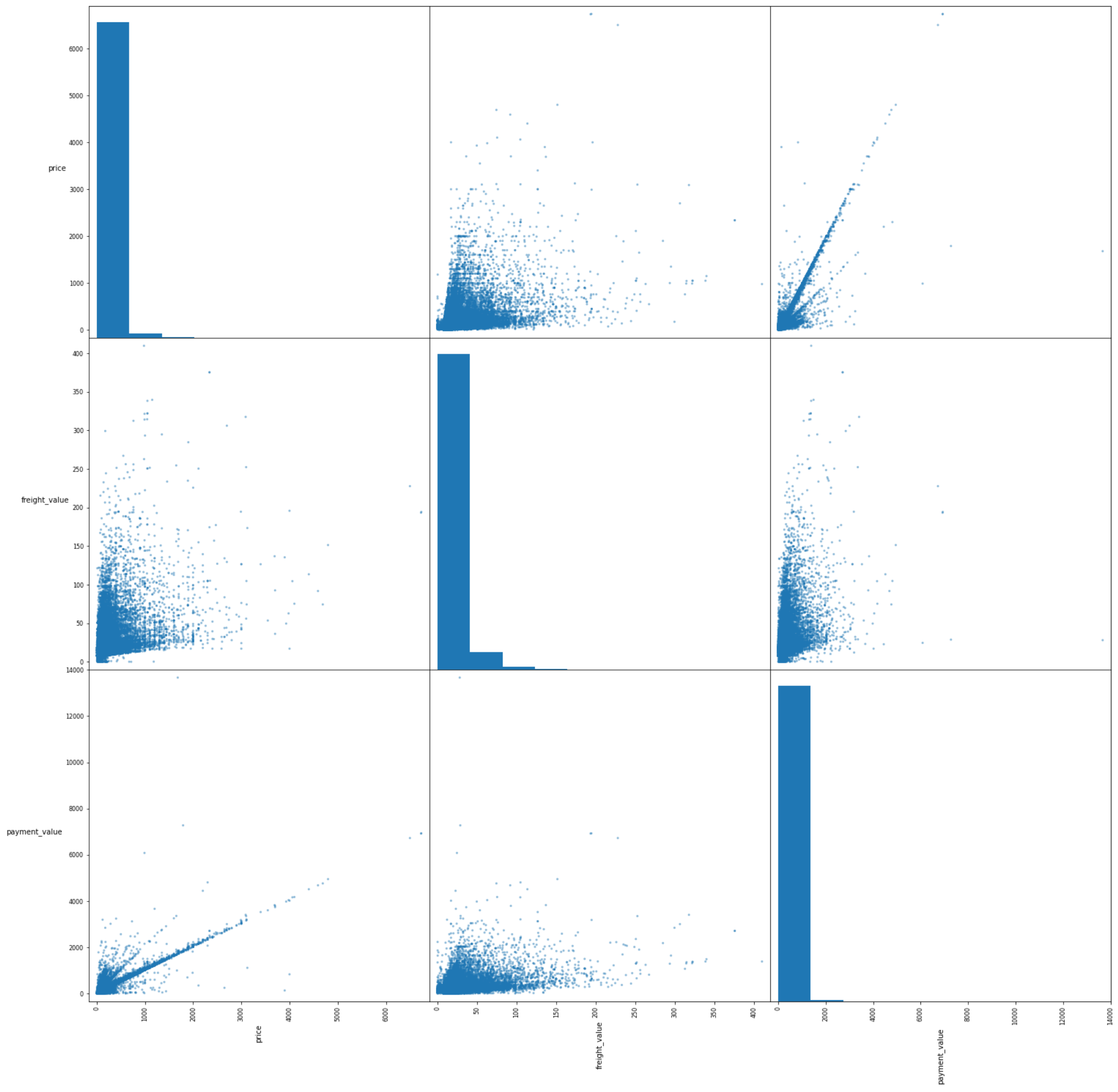


```
In [54]:
# Merge these three dataframes together: orders_dataset, order_payments_dataset,
df_merge_1 = orders_dataset.join(order_payments_dataset, on=['order_id'], how='i
df_merge_2 = df_merge_1.join(order_items_dataset, on=['order_id'], how='inner')

#df_merge = df_merge_2.drop('order_approved_at', 'order_carrier_delivery_date',
df_merge = df_merge_2.drop_duplicates(['order_id'])
df_merge_order = df_merge.dropna()
df_merge_order = df_merge.select('order_id', 'order_item_id', 'customer_id', 'se

df_merge_order.show(2)
```

```
+--------------------+-------------+--------------------+--------------------+--
------------------+------------+------------------------+-----+-------------+---
----------+
|            order_id|order_item_id|         customer_id|           seller_id|
product_id|order_status|order_purchase_timestamp|price|freight_value|payment_val
ue|
+--------------------+-------------+--------------------+--------------------+--
------------------+------------+------------------------+-----+-------------+---
----------+
|014405982914c2cde...|            1|2de342d6e5905a5a8...|325f3178fb58e2a97...|67
82d593f63105318...|   delivered|     2017-07-26 17:38:47| 27.9|         3.81|
78.43|
|019886de8f385a39b...|            1|8cf88d7ba142365ef...|1b4c3a6f53068f0b6...|e9
a69340883a438c3...|   delivered|     2018-02-10 12:52:51|159.9|         28.5|
188.4|
+--------------------+-------------+--------------------+--------------------+--
------------------+------------+------------------------+-----+-------------+---
----------+
only showing top 2 rows
```

In [55]:

```python
# Checking Correlations between independent variables
numeric_features = [t[0] for t in df_merge_order.dtypes if t[1] == 'double']
numeric_data = df_merge_order.select(numeric_features).toPandas()

axs = scatter_matrix(numeric_data, figsize=(25, 25));

# Rotate axis labels and remove axis ticks
n = len(numeric_data.columns)
for i in range(n):
    v = axs[i, 0]
    v.yaxis.label.set_rotation(0)
    v.yaxis.label.set_ha('right')
    h = axs[n-1, i]
    h.xaxis.label.set_rotation(90)
```



Explore relationships across the entire dataset

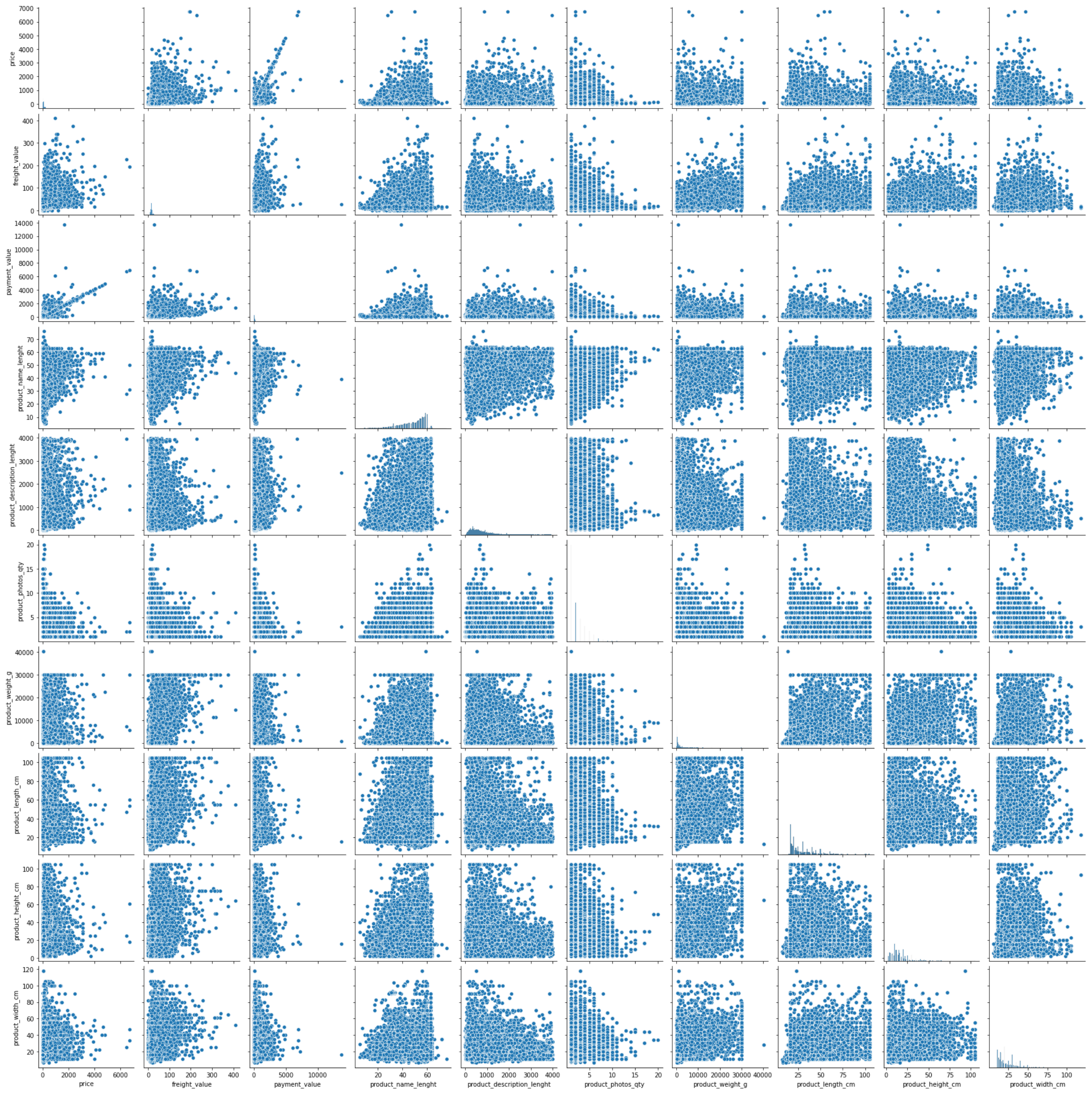
In [56]:
```
# merge all above dataframes together
merge_df = df_merge_product_and_category.join(df_merge_order, on=["product_id"],
merge_df = merge_df.select('product_id', 'price', 'freight_value', 'payment_valu
                           'product_length_cm', 'product_height_cm', 'product_wid

merge_df.show(2)
```

```
+--------------------+-----+-------------+-------------+-------------------+----
----------------------+------------------+-----------------+-----------------+---
--------------+----------------+----------------------------+-----------------
------+
|          product_id|price|freight_value|payment_value|product_name_lenght|prod
uct_description_lenght|product_photos_qty|product_weight_g|product_length_cm|pro
duct_height_cm|product_width_cm|product_category_name_english|order_purchase_tim
estamp|
+--------------------+-----+-------------+-------------+-------------------+----
----------------------+------------------+-----------------+-----------------+---
--------------+----------------+----------------------------+-----------------
```

```
------+
|00e4ded51458037ec...|130.0|        38.46|        168.46|                    25|
978|                    3|                1400|                    25|                    15|
25|         computers_accesso...|     2017-08-17 10:06:55|
|03d7ad0ce97624c93...| 79.9|        25.05|         54.95|                    48|
259|                    1|                 249|                    19|                    13|
15|                     perfumery|     2017-04-06 15:26:20|
+--------------------+-----+-------------+-------------+-------------------+----
----------------------+------------------+-----------------+-----------------+---
---------------+----------------+------------------------------+------------------
------+
only showing top 2 rows
```

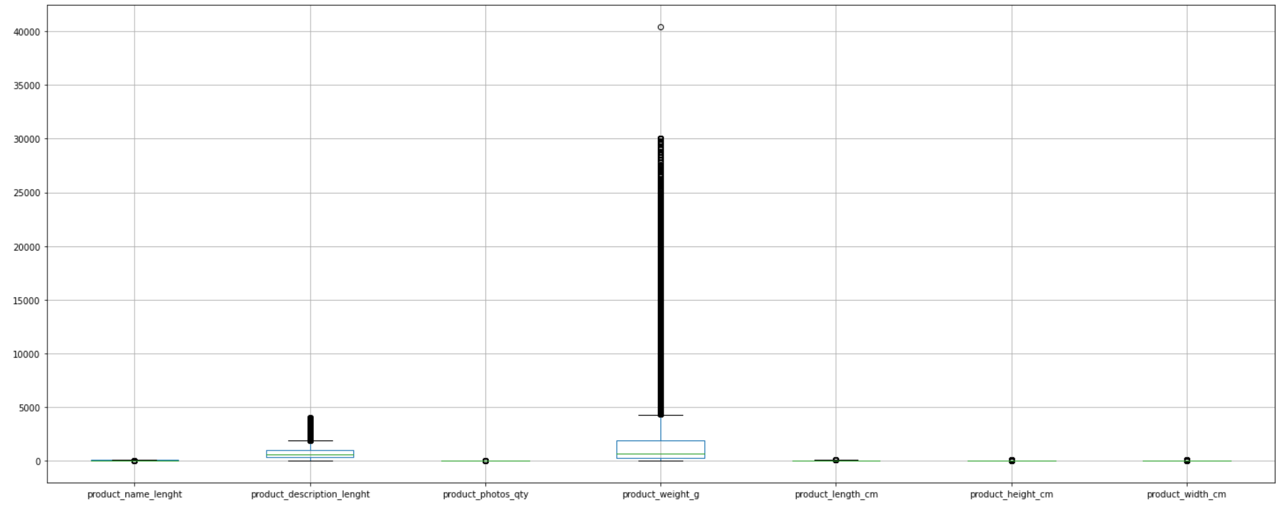
In [57]:
```
sns.pairplot(merge_df.toPandas())
plt.show()
```

Out[57]:
```
<seaborn.axisgrid.PairGrid at 0x7f6a3b9d1b20>
```



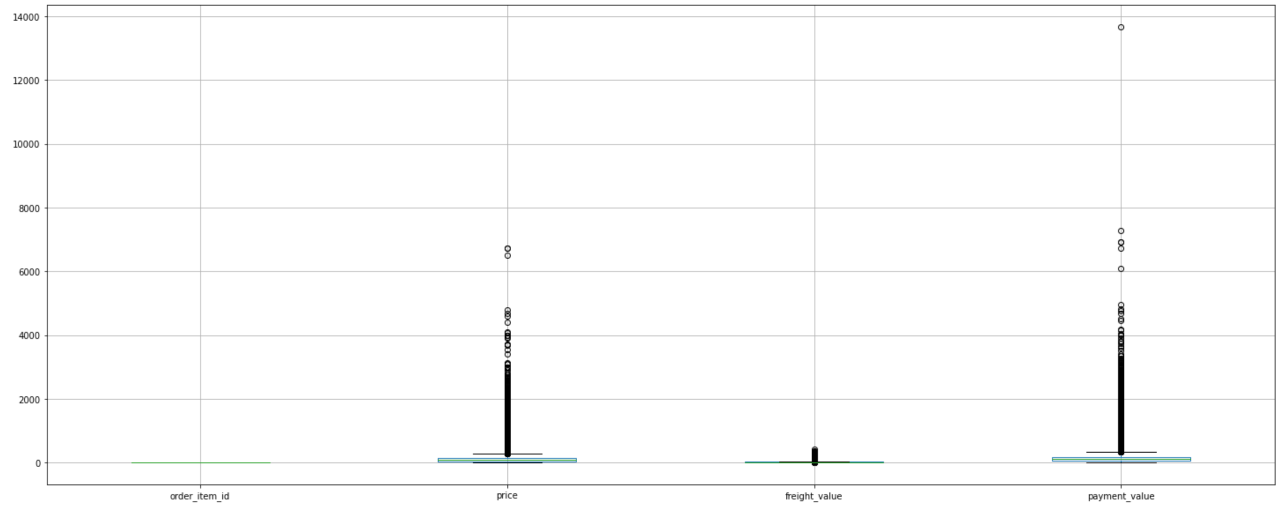
Distribution of Data

```
In [58]: plot = df_merge_product_and_category.toPandas().boxplot(figsize = (25,10))
```

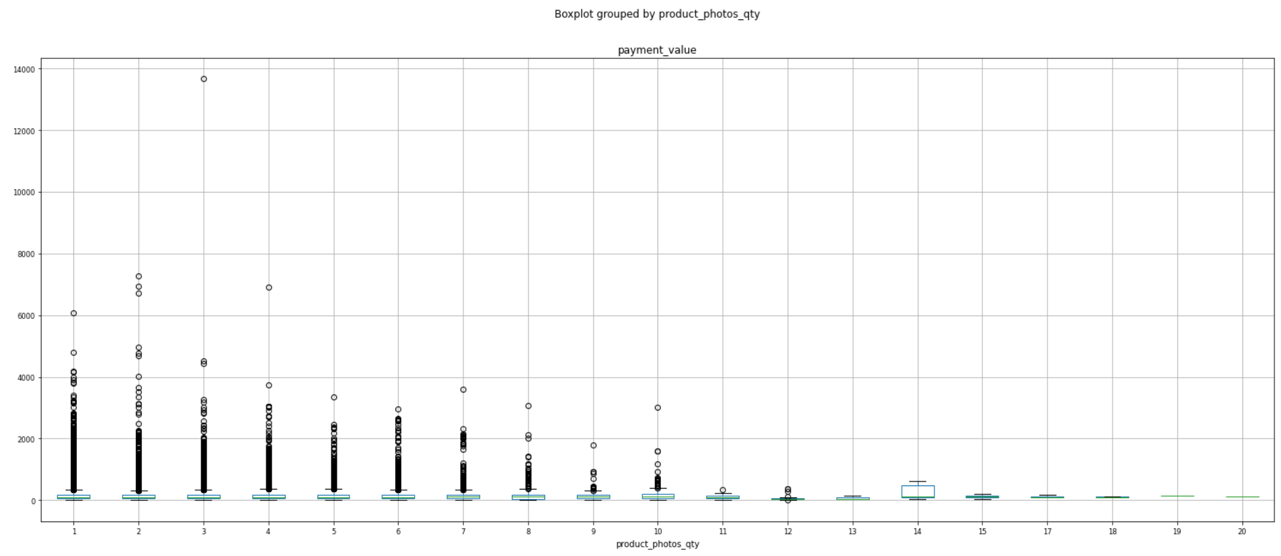


```
In [59]: df_merge_order.toPandas().boxplot(figsize = (25,10))
```

```
Out[59]: <AxesSubplot:>
```



```
In [60]: plot = merge_df.toPandas().boxplot(column='payment_value', by='product_photos_qt
```



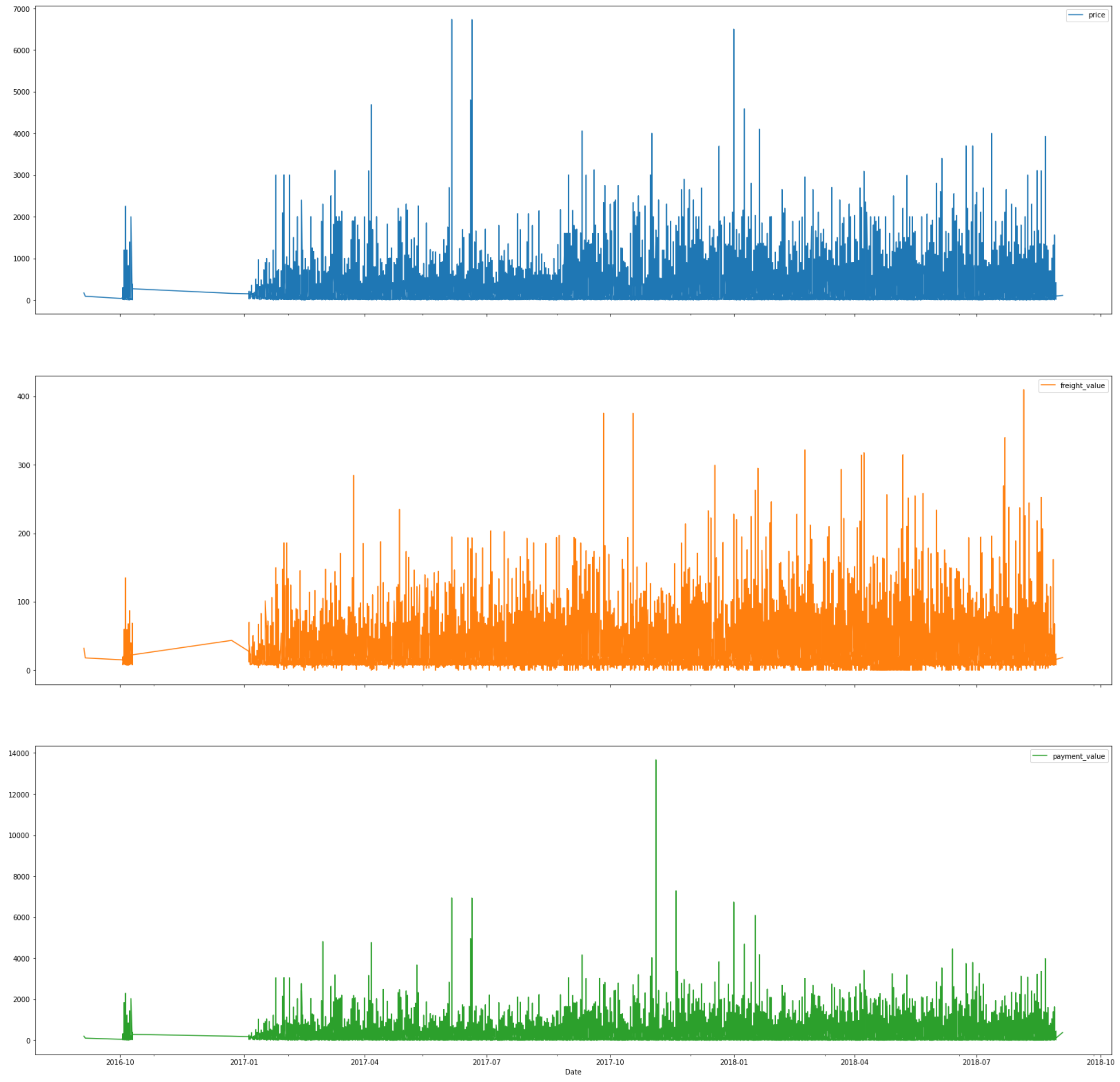
The boxplot showed that except the outliers, product photos quantity within 10 have higner payment values.

In [61]:
```
df_merge_new = merge_df.select('product_id', 'order_purchase_timestamp', 'price'
date_col = df_merge_new.select(date_format(col('order_purchase_timestamp'),"yyyy
date_col = date_col.withColumn("id", monotonically_increasing_id())
df_merge_new = df_merge_new.withColumn("id", monotonically_increasing_id())
df3 = df_merge_new.join(date_col, on=["id"], how="left").drop("id", "order_purch
df3 = df3.dropna()
df3.show(2)
```

```
+--------------------+-----+-------------+-------------+----------+
|          product_id|price|freight_value|payment_value|      Date|
+--------------------+-----+-------------+-------------+----------+
|08574b074924071f4...| 99.0|        41.08|       280.16|2017-10-08|
|08574b074924071f4...| 99.0|        41.08|       140.08|2017-12-12|
+--------------------+-----+-------------+-------------+----------+
only showing top 2 rows
```

In [62]:
```
df = df3.toPandas()
df[['Date','price', 'freight_value','payment_value']].plot(x='Date', subplots=Tr
plt.show()
```

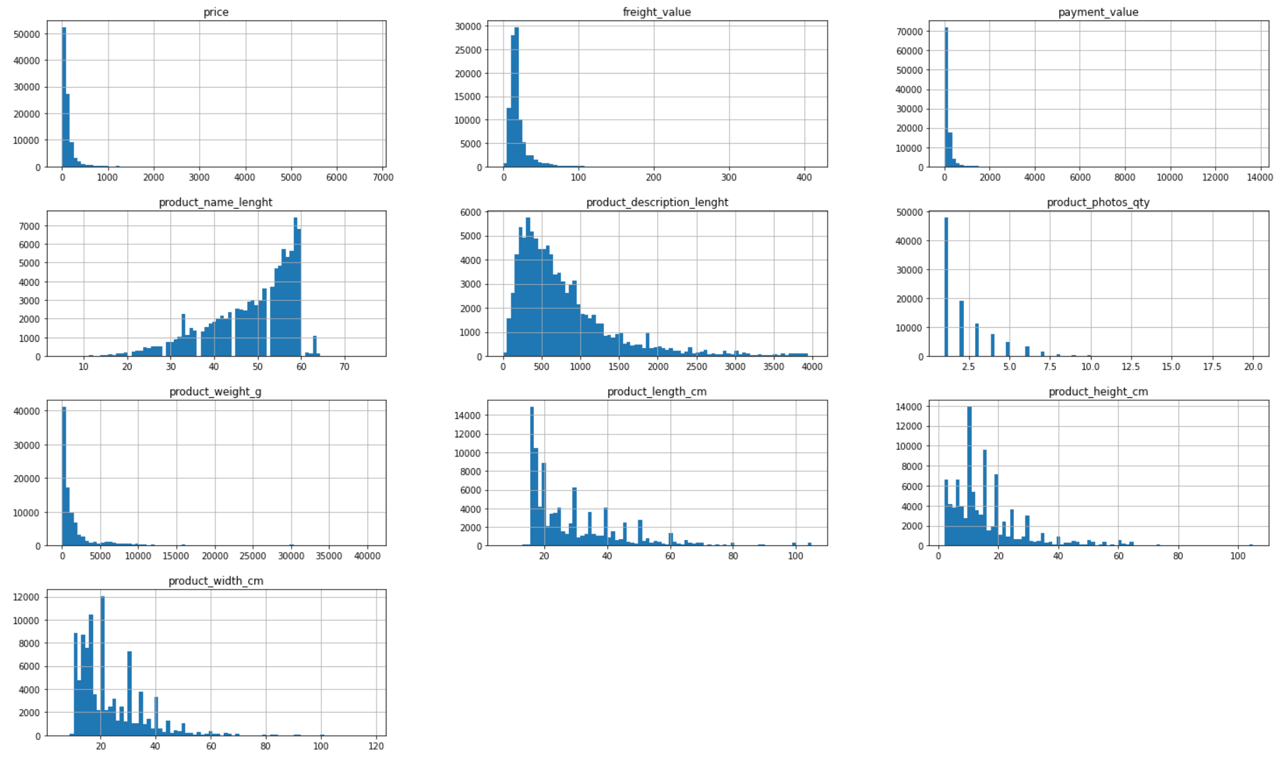
Out[62]:
```
array([<AxesSubplot:xlabel='Date'>, <AxesSubplot:xlabel='Date'>,
       <AxesSubplot:xlabel='Date'>], dtype=object)
```



Common trend

```
In [63]: merge_df.toPandas().hist(figsize = (25,15), bins = 80)
```

```
Out[63]: array([[<AxesSubplot:title={'center':'price'}>,
                 <AxesSubplot:title={'center':'freight_value'}>,
                 <AxesSubplot:title={'center':'payment_value'}>],
                [<AxesSubplot:title={'center':'product_name_lenght'}>,
                 <AxesSubplot:title={'center':'product_description_lenght'}>,
                 <AxesSubplot:title={'center':'product_photos_qty'}>],
                [<AxesSubplot:title={'center':'product_weight_g'}>,
                 <AxesSubplot:title={'center':'product_length_cm'}>,
                 <AxesSubplot:title={'center':'product_height_cm'}>],
                [<AxesSubplot:title={'center':'product_width_cm'}>,
                 <AxesSubplot:>, <AxesSubplot:>]], dtype=object)
```



Heatmap for comprehensive overview

In [64]:

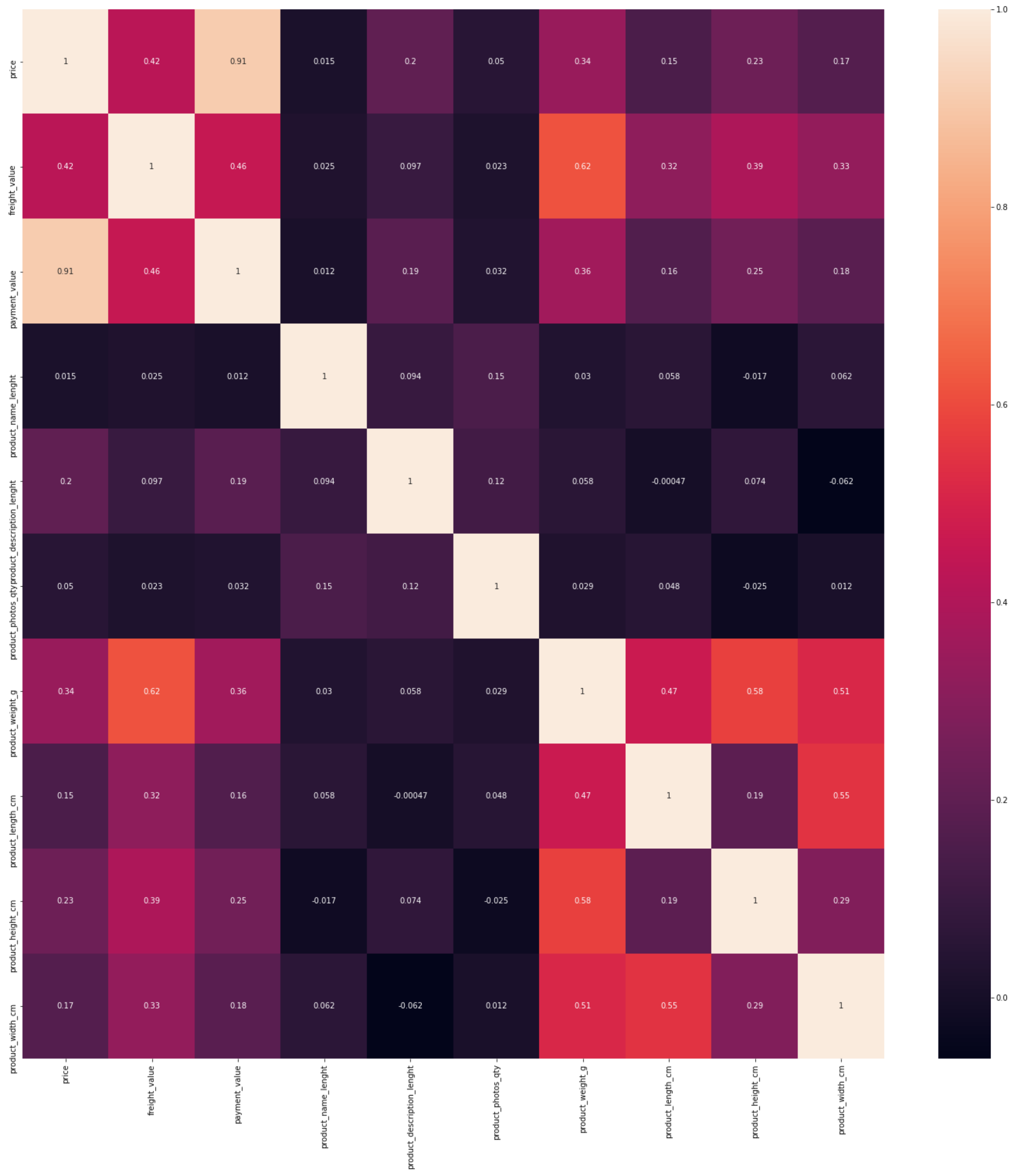
```
Var_Corr = merge_df.toPandas().corr()
# plot the heatmap and annotation on it
plt.figure(figsize = (25,25))
sns.heatmap(Var_Corr, xticklabels=Var_Corr.columns, yticklabels=Var_Corr.columns
```

Out[64]:

```
<Figure size 1800x1800 with 0 Axes>
```

Out[64]:

```
<AxesSubplot:>
```



Stop the spark session

In [65]:
```
spark.stop()
```

In [ ]:
```

```