KPMG VIRTUAL INTERNSHIP TASK -1

Task - Data Quality Assignment

Sprocket Central Pty Ltd , a medium size bikes & cycling accessories organisation, has approached Tony Smith (Partner) in KPMG's Lighthouse & Innovation Team. Sprocket Central Pty Ltd is keen to learn more about KPMG's expertise in its Analytics, Information & Modelling team.

Smith discusses KPMG's expertise in this space. In particular, he speaks about how the team can effectively analyse the datasets to help Sprocket Central Pty Ltd grow its business.

Primarily, Sprocket Central Pty Ltd needs help with its customer and transactions data. The organisation has a large dataset relating to its customers, but their team is unsure how to effectively analyse it to help optimise its marketing strategy.

However, in order to support the analysis, you speak to the Associate Director for some ideas and she advised that “the importance of optimising the quality of customer datasets cannot be underestimated. The better the quality of the dataset, the better chance you will be able to use it drive company growth.”

please find the 3 datasets attached from Sprocket Central Pty Ltd:

- Customer Demographic
- Customer Addresses
- Transaction data in the past three months

Can you please review the data quality to ensure that it is ready for our analysis in phase two.

Importing all required libraries

In [1]:

```
import pandas as pd
```

Read the data

In [9]:

```
Data = pd.ExcelFile("E:\Data Science\Forage virtual internship\KPMG_VI_New_raw_data_update_final2.xlsx")
```

In [10]:

```
#reading of each file seperatly
Transactions=pd.read_excel(Data, 'Transactions')
NewCustomerList=pd.read_excel(Data, 'NewCustomerList')
CustomerDemographic=pd.read_excel(Data, 'CustomerDemographic')
CustomerAddress=pd.read_excel(Data, 'CustomerAddress')
```

```
C:\Users\Admin\AppData\Local\Temp\ipykernel_13364\2500845031.py:3: FutureWarning: Inferring datetime64[ns] from data contai
ning strings is deprecated and will be removed in a future version. To retain the old behavior explicitly pass Series(data,
dtype=datetime64[ns])
  NewCustomerList=pd.read_excel(Data, 'NewCustomerList')
C:\Users\Admin\AppData\Local\Temp\ipykernel_13364\2500845031.py:4: FutureWarning: Inferring datetime64[ns] from data contai
ning strings is deprecated and will be removed in a future version. To retain the old behavior explicitly pass Series(data,
dtype=datetime64[ns])
  CustomerDemographic=pd.read_excel(Data, 'CustomerDemographic')
```

In [12]:

```
Transactions.head(5)
```

Out[12]:

	transaction_id	product_id	customer_id	transaction_date	online_order	order_status	brand	product_line	product_class	product_size	list_price	sta
0	1	2	2950	2017-02-25	0.0	Approved	Solex	Standard	medium	medium	71.49	
1	2	3	3120	2017-05-21	1.0	Approved	Trek Bicycles	Standard	medium	large	2091.47	
2	3	37	402	2017-10-16	0.0	Approved	OHM Cycles	Standard	low	medium	1793.43	
3	4	88	3135	2017-08-31	0.0	Approved	Norco Bicycles	Standard	medium	medium	1198.46	
4	5	78	787	2017-10-01	1.0	Approved	Giant Bicycles	Standard	medium	large	1765.30	

In [13]:

```
Transactions.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 20000 entries, 0 to 19999
Data columns (total 13 columns):
 #   Column                   Non-Null Count  Dtype
---  ------                   --------------  -----
 0   transaction_id           20000 non-null  int64
 1   product_id               20000 non-null  int64
 2   customer_id              20000 non-null  int64
 3   transaction_date         20000 non-null  datetime64[ns]
 4   online_order             19640 non-null  float64
 5   order_status             20000 non-null  object
 6   brand                    19803 non-null  object
 7   product_line             19803 non-null  object
 8   product_class            19803 non-null  object
 9   product_size             19803 non-null  object
 10  list_price               20000 non-null  float64
 11  standard_cost            19803 non-null  float64
 12  product_first_sold_date  19803 non-null  float64
dtypes: datetime64[ns](1), float64(4), int64(3), object(5)
memory usage: 2.0+ MB
```

In [15]:

```
#using reuireed column
Transactions= Transactions.iloc[:,0:13]
Transactions.head(5)
```

Out[15]:

	transaction_id	product_id	customer_id	transaction_date	online_order	order_status	brand	product_line	product_class	product_size	list_price	sta
0	1	2	2950	2017-02-25	0.0	Approved	Solex	Standard	medium	medium	71.49	
1	2	3	3120	2017-05-21	1.0	Approved	Trek Bicycles	Standard	medium	large	2091.47	
2	3	37	402	2017-10-16	0.0	Approved	OHM Cycles	Standard	low	medium	1793.43	
3	4	88	3135	2017-08-31	0.0	Approved	Norco Bicycles	Standard	medium	medium	1198.46	
4	5	78	787	2017-10-01	1.0	Approved	Giant Bicycles	Standard	medium	large	1765.30	

In [17]:

```
Transactions.shape
```

Out[17]:

```
(20000, 13)
```

In [18]:

```
Transactions.isnull().sum()
```

Out[18]:

```
transaction_id               0
product_id                   0
customer_id                  0
transaction_date             0
online_order               360
order_status                 0
brand                      197
product_line               197
product_class              197
product_size               197
list_price                   0
standard_cost              197
product_first_sold_date    197
dtype: int64
```

```
- In that table there are missing value in 7 column. next remove all missing value.
```

In [19]:

```
#duplicate value
Transactions.duplicated().sum()
```

Out[19]:

```
0
```

- No duplicate value

In [20]:

```
#unique value of each column
Transactions.nunique()
```

Out[20]:

```
transaction_id             20000
product_id                   101
customer_id                 3494
transaction_date             364
online_order                   2
order_status                   2
brand                          6
product_line                   4
product_class                  3
product_size                   3
list_price                   296
standard_cost                103
product_first_sold_date      100
dtype: int64
```

Exploring the columns

In [21]:

```
Transactions.columns
```

Out[21]:

```
Index(['transaction_id', 'product_id', 'customer_id', 'transaction_date',
       'online_order', 'order_status', 'brand', 'product_line',
       'product_class', 'product_size', 'list_price', 'standard_cost',
       'product_first_sold_date'],
      dtype='object')
```

In [22]:

```
Transactions['order_status'].value_counts()
```

Out[22]:

```
Approved     19821
Cancelled      179
Name: order_status, dtype: int64
```

In [23]:

```
Transactions['brand'].value_counts()
```

Out[23]:

```
Solex             4253
Giant Bicycles    3312
WeareA2B          3295
OHM Cycles        3043
Trek Bicycles     2990
Norco Bicycles    2910
Name: brand, dtype: int64
```

In [24]:

```
Transactions['product_line'].value_counts()
```

Out[24]:

```
Standard    14176
Road         3970
Touring      1234
Mountain      423
Name: product_line, dtype: int64
```

In [25]:

```
Transactions['product_class'].value_counts()
```

Out[25]:

```
medium    13826
high       3013
low        2964
Name: product_class, dtype: int64
```

In [26]:

```
Transactions['product_size'].value_counts()
```

Out[26]:

```
medium    12990
large      3976
small      2837
Name: product_size, dtype: int64
```

In [27]:

```
Transactions['product_first_sold_date'].value_counts()
```

Out[27]:

```
33879.0    234
41064.0    229
37823.0    227
39880.0    222
38216.0    220
          ... 
41848.0    169
42404.0    168
41922.0    166
37659.0    163
34586.0    162
Name: product_first_sold_date, Length: 100, dtype: int64
```

In [29]:

```
#convert integer to datetime
Transactions['product_first_sold_date']=pd.to_datetime(Transactions['product_first_sold_date'], unit='s')
```

In [31]:

```
Transactions['product_first_sold_date'].head(10)
```

Out[31]:

```
0   1970-01-01 11:27:25
1   1970-01-01 11:35:01
2   1970-01-01 10:06:01
3   1970-01-01 10:02:25
4   1970-01-01 11:43:46
5   1970-01-01 10:50:31
6   1970-01-01 09:29:25
7   1970-01-01 11:05:15
8   1970-01-01 09:17:35
9   1970-01-01 10:36:56
Name: product_first_sold_date, dtype: datetime64[ns]
```

New CustomerList Dataset

In [33]:

```
NewCustomerList.head()
```

Out[33]:

	first_name	last_name	gender	past_3_years_bike_related_purchases	DOB	job_title	job_industry_category	wealth_segment	deceased_indicator
0	Chickie	Brister	Male	86	1957-07-12	General Manager	Manufacturing	Mass Customer	N
1	Morly	Genery	Male	69	1970-03-22	Structural Engineer	Property	Mass Customer	N
2	Ardelis	Forrester	Female	10	1974-08-28	Senior Cost Accountant	Financial Services	Affluent Customer	N
3	Lucine	Stutt	Female	64	1979-01-28	Account Representative III	Manufacturing	Affluent Customer	N
4	Melinda	Hadlee	Female	34	1965-09-21	Financial Analyst	Financial Services	Affluent Customer	N

5 rows × 23 columns

In [34]:

```
NewCustomerList.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1000 entries, 0 to 999
Data columns (total 23 columns):
 #   Column                               Non-Null Count  Dtype
---  ------                               --------------  -----
 0   first_name                           1000 non-null   object
 1   last_name                            971 non-null    object
 2   gender                               1000 non-null   object
 3   past_3_years_bike_related_purchases  1000 non-null   int64
 4   DOB                                  983 non-null    datetime64[ns]
 5   job_title                            894 non-null    object
 6   job_industry_category                835 non-null    object
 7   wealth_segment                       1000 non-null   object
 8   deceased_indicator                   1000 non-null   object
 9   owns_car                             1000 non-null   object
 10  tenure                               1000 non-null   int64
 11  address                              1000 non-null   object
 12  postcode                             1000 non-null   int64
 13  state                                1000 non-null   object
 14  country                              1000 non-null   object
 15  property_valuation                   1000 non-null   int64
 16  Unnamed: 16                          1000 non-null   float64
 17  Unnamed: 17                          1000 non-null   float64
 18  Unnamed: 18                          1000 non-null   float64
 19  Unnamed: 19                          1000 non-null   float64
 20  Unnamed: 20                          1000 non-null   int64
 21  Rank                                 1000 non-null   int64
 22  Value                                1000 non-null   float64
dtypes: datetime64[ns](1), float64(5), int64(6), object(11)
memory usage: 179.8+ KB
```

In [35]:

```
#Drop the unnamed column
NewCustomerList.drop(['Unnamed: 16','Unnamed: 17','Unnamed: 18','Unnamed: 19','Unnamed: 20'], axis=1, inplace=True)
```

In [36]:

```
NewCustomerList.shape
```

Out[36]:

```
(1000, 18)
```

In [37]:

```
NewCustomerList.isnull().sum()
```

Out[37]:

```
first_name                               0
last_name                               29
gender                                   0
past_3_years_bike_related_purchases      0
DOB                                     17
job_title                              106
job_industry_category                  165
wealth_segment                           0
deceased_indicator                       0
owns_car                                 0
tenure                                   0
address                                  0
postcode                                 0
state                                    0
country                                  0
property_valuation                       0
Rank                                     0
Value                                    0
dtype: int64
```

```
- In that table there are missing value in 4 column. next remove all missing value.
```

In [38]:

```
NewCustomerList.duplicated().sum()
```

Out[38]:

```
0
```

- No duplicate value

In [39]:

```
NewCustomerList.nunique()
```

Out[39]:

```
first_name                             940
last_name                              961
gender                                   3
past_3_years_bike_related_purchases    100
DOB                                    958
job_title                              184
job_industry_category                    9
wealth_segment                           3
deceased_indicator                       1
owns_car                                 2
tenure                                  23
address                               1000
postcode                               522
state                                    3
country                                  1
property_valuation                      12
Rank                                   324
Value                                  324
dtype: int64
```

Exploring column

In [40]:

```
NewCustomerList.columns
```

Out[40]:

```
Index(['first_name', 'last_name', 'gender',
       'past_3_years_bike_related_purchases', 'DOB', 'job_title',
       'job_industry_category', 'wealth_segment', 'deceased_indicator',
       'owns_car', 'tenure', 'address', 'postcode', 'state', 'country',
       'property_valuation', 'Rank', 'Value'],
      dtype='object')
```

In [41]:

```
NewCustomerList['gender'].value_counts()
```

Out[41]:

```
Female    513
Male      470
U          17
Name: gender, dtype: int64
```

In [42]:

```
NewCustomerList[NewCustomerList.gender=='U']
```

Out[42]:

	first_name	last_name	gender	past_3_years_bike_related_purchases	DOB	job_title	job_industry_category	wealth_segment	deceased_indicator
59	Normy	Goodinge	U	5	NaT	Associate Professor	IT	Mass Customer	N
226	Hatti	Carletti	U	35	NaT	Legal Assistant	IT	Affluent Customer	N
324	Rozamond	Turtle	U	69	NaT	Legal Assistant	IT	Mass Customer	N
358	Tamas	Swatman	U	65	NaT	Assistant Media Planner	Entertainment	Affluent Customer	N
360	Tracy	Andrejevic	U	71	NaT	Programmer II	IT	Mass Customer	N
374	Agneta	McAmish	U	66	NaT	Structural Analysis Engineer	IT	Mass Customer	N
434	Gregg	Aimeric	U	52	NaT	Internal Auditor	IT	Mass Customer	N
439	Johna	Bunker	U	93	NaT	Tax Accountant	IT	Mass Customer	N
574	Harlene	Nono	U	69	NaT	Human Resources Manager	IT	Mass Customer	N
598	Gerianne	Kaysor	U	15	NaT	Project Manager	IT	Affluent Customer	N
664	Chicky	Sinclar	U	43	NaT	Operator	IT	High Net Worth	N
751	Adriana	Saundercock	U	20	NaT	Nurse	IT	High Net Worth	N
775	Dmitri	Viant	U	62	NaT	Paralegal	Financial Services	Affluent Customer	N
835	Porty	Hansed	U	88	NaT	General Manager	IT	Mass Customer	N
883	Shara	Bramhill	U	24	NaT	NaN	IT	Affluent Customer	N
904	Roth	Crum	U	0	NaT	Legal Assistant	IT	Mass Customer	N
984	Pauline	Dallosso	U	82	NaT	Desktop Support Technician	IT	Affluent Customer	N

- 17 unknown and unspecified gender

In [43]:

```
NewCustomerList['job_industry_category'].value_counts()
```

Out[43]:

```
Financial Services    203
Manufacturing         199
Health                152
Retail                 78
Property               64
IT                     51
Entertainment          37
Argiculture            26
Telecommunications     25
Name: job_industry_category, dtype: int64
```

In [44]:

```
NewCustomerList['wealth_segment'].value_counts()
```

Out[44]:

```
Mass Customer        508
High Net Worth       251
Affluent Customer    241
Name: wealth_segment, dtype: int64
```

In [45]:

```
NewCustomerList['state'].value_counts()
```

Out[45]:

```
NSW    506
VIC    266
QLD    228
Name: state, dtype: int64
```

In [46]:

```
NewCustomerList['owns_car'].value_counts()
```

Out[46]:

```
No     507
Yes    493
Name: owns_car, dtype: int64
```

In [47]:

```
NewCustomerList['deceased_indicator'].value_counts()
```

Out[47]:

```
N    1000
Name: deceased_indicator, dtype: int64
```

Customer Demographic Dataset

In [48]:

```
CustomerDemographic.head()
```

Out[48]:

	customer_id	first_name	last_name	gender	past_3_years_bike_related_purchases	DOB	job_title	job_industry_category	wealth_segment	deceas
0	1	Laraine	Medendorp	F	93	1953-10-12	Executive Secretary	Health	Mass Customer	
1	2	Eli	Bockman	Male	81	1980-12-16	Administrative Officer	Financial Services	Mass Customer	
2	3	Arlin	Dearle	Male	61	1954-01-20	Recruiting Manager	Property	Mass Customer	
3	4	Talbot	NaN	Male	33	1961-10-03	NaN	IT	Mass Customer	
4	5	Sheila-kathryn	Calton	Female	56	1977-05-13	Senior Editor	NaN	Affluent Customer	

In [49]:

```
CustomerDemographic.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 4000 entries, 0 to 3999
Data columns (total 13 columns):
 #   Column                               Non-Null Count  Dtype
---  ------                               --------------  -----
 0   customer_id                          4000 non-null   int64
 1   first_name                           4000 non-null   object
 2   last_name                            3875 non-null   object
 3   gender                               4000 non-null   object
 4   past_3_years_bike_related_purchases  4000 non-null   int64
 5   DOB                                  3913 non-null   datetime64[ns]
 6   job_title                            3494 non-null   object
 7   job_industry_category                3344 non-null   object
 8   wealth_segment                       4000 non-null   object
 9   deceased_indicator                   4000 non-null   object
 10  default                              3698 non-null   object
 11  owns_car                             4000 non-null   object
 12  tenure                               3913 non-null   float64
dtypes: datetime64[ns](1), float64(1), int64(2), object(9)
memory usage: 406.4+ KB
```

In [50]:

```
CustomerDemographic.isnull().sum()
```

Out[50]:

```
customer_id                              0
first_name                               0
last_name                              125
gender                                   0
past_3_years_bike_related_purchases      0
DOB                                     87
job_title                              506
job_industry_category                  656
wealth_segment                           0
deceased_indicator                       0
default                                302
owns_car                                 0
tenure                                  87
dtype: int64
```

- In that table there are missing value in 6 column. next remove all missing value.

In [52]:

```
CustomerDemographic.duplicated().sum()
```

Out[52]:

```
0
```

- No duplicated value

In [53]:

```
#unique value
CustomerDemographic.nunique()
```

Out[53]:

```
customer_id                            4000
first_name                             3139
last_name                              3725
gender                                    6
past_3_years_bike_related_purchases     100
DOB                                    3448
job_title                               195
job_industry_category                     9
wealth_segment                            3
deceased_indicator                        2
default                                  90
owns_car                                  2
tenure                                   22
dtype: int64
```

Exploeing cloumns

In [54]:

```
CustomerDemographic.columns
```

Out[54]:

```
Index(['customer_id', 'first_name', 'last_name', 'gender',
       'past_3_years_bike_related_purchases', 'DOB', 'job_title',
       'job_industry_category', 'wealth_segment', 'deceased_indicator',
       'default', 'owns_car', 'tenure'],
      dtype='object')
```

In [55]:

```
CustomerDemographic['gender'].value_counts()
```

Out[55]:

```
Female    2037
Male      1872
U           88
F            1
Femal        1
M            1
Name: gender, dtype: int64
```

- In that nit correct title. so then rename all

In [56]:

```
CustomerDemographic['gender']=CustomerDemographic['gender'].replace('F','Female').replace('Femal','Female').replace('M','Male')
```

In [57]:

```
CustomerDemographic['gender'].value_counts()
```

Out[57]:

```
Female    2039
Male      1873
U           88
Name: gender, dtype: int64
```

In [58]:

```
CustomerDemographic['past_3_years_bike_related_purchases'].value_counts()
```

Out[58]:

```
16    56
19    56
67    54
20    54
2     50
      ..
8     28
95    27
85    27
86    27
92    24
Name: past_3_years_bike_related_purchases, Length: 100, dtype: int64
```

In [59]:

```
CustomerDemographic['DOB'].value_counts()
```

Out[59]:

```
1978-01-30    7
1964-07-08    4
1962-12-17    4
1978-08-19    4
1977-05-13    4
             ..
1989-06-16    1
1998-09-30    1
1985-03-11    1
1989-10-23    1
1991-11-05    1
Name: DOB, Length: 3448, dtype: int64
```

In [60]:

```
CustomerDemographic['job_title'].value_counts()
```

Out[60]:

```
Business Systems Development Analyst    45
Tax Accountant                          44
Social Worker                           44
Internal Auditor                        42
Recruiting Manager                      41
                                        ..
Database Administrator I                 4
Health Coach I                           3
Health Coach III                         3
Research Assistant III                   3
Developer I                              1
Name: job_title, Length: 195, dtype: int64
```

In [61]:

```
CustomerDemographic['job_industry_category'].value_counts()
```

Out[61]:

```
Manufacturing         799
Financial Services    774
Health                602
Retail                358
Property              267
IT                    223
Entertainment         136
Argiculture           113
Telecommunications     72
Name: job_industry_category, dtype: int64
```

In [62]:

```
CustomerDemographic['wealth_segment'].value_counts()
```

Out[62]:

```
Mass Customer        2000
High Net Worth       1021
Affluent Customer     979
Name: wealth_segment, dtype: int64
```

In [63]:

```
CustomerDemographic['deceased_indicator'].value_counts()
```

Out[63]:

```
N    3998
Y       2
Name: deceased_indicator, dtype: int64
```

In [64]:

```
CustomerDemographic['default'].value_counts()
```

Out[64]:

```
100                                       113
1                                         112
-1                                        111
-100                                       99
Ù¡Ù¢Ù£                                     53
                                        ...
testâ testâ«                               31
/dev/null; touch /tmp/blns.fail ; echo     30
âªâªtestâª                                 29
ì¸ëëºí ë¥´                                 27
,ãã»:*:ã»ãâ( â» Ï â» )ãã»:*:ã»ãâ           25
Name: default, Length: 90, dtype: int64
```

In [65]:

```
CustomerDemographic=CustomerDemographic.drop('default', axis=1)
```

In [66]:

```
CustomerDemographic.head(5)
```

Out[66]:

	customer_id	first_name	last_name	gender	past_3_years_bike_related_purchases	DOB	job_title	job_industry_category	wealth_segment	deceas
0	1	Laraine	Medendorp	Female	93	1953-10-12	Executive Secretary	Health	Mass Customer	
1	2	Eli	Bockman	Male	81	1980-12-16	Administrative Officer	Financial Services	Mass Customer	
2	3	Arlin	Dearle	Male	61	1954-01-20	Recruiting Manager	Property	Mass Customer	
3	4	Talbot	NaN	Male	33	1961-10-03	NaN	IT	Mass Customer	
4	5	Sheila-kathryn	Calton	Female	56	1977-05-13	Senior Editor	NaN	Affluent Customer	

In [67]:

```
CustomerDemographic['owns_car'].value_counts()
```

Out[67]:

```
Yes    2024
No     1976
Name: owns_car, dtype: int64
```

In [68]:

```
CustomerDemographic['tenure'].value_counts()
```

Out[68]:

```
7.0     235
5.0     228
11.0    221
10.0    218
16.0    215
8.0     211
18.0    208
12.0    202
9.0     200
14.0    200
6.0     192
13.0    191
4.0     191
17.0    182
15.0    179
1.0     166
3.0     160
19.0    159
2.0     150
20.0     96
22.0     55
21.0     54
Name: tenure, dtype: int64
```

CustomerAddress Dataset

In [70]:

```
CustomerAddress.head(5)
```

Out[70]:

	customer_id	address	postcode	state	country	property_valuation
0	1	060 Morning Avenue	2016	New South Wales	Australia	10
1	2	6 Meadow Vale Court	2153	New South Wales	Australia	10
2	4	0 Holy Cross Court	4211	QLD	Australia	9
3	5	17979 Del Mar Point	2448	New South Wales	Australia	4
4	6	9 Oakridge Court	3216	VIC	Australia	9

In [71]:

```
CustomerAddress.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 3999 entries, 0 to 3998
Data columns (total 6 columns):
 #   Column              Non-Null Count  Dtype 
---  ------              --------------  ----- 
 0   customer_id         3999 non-null   int64 
 1   address             3999 non-null   object
 2   postcode            3999 non-null   int64 
 3   state               3999 non-null   object
 4   country             3999 non-null   object
 5   property_valuation  3999 non-null   int64 
dtypes: int64(3), object(3)
memory usage: 187.6+ KB
```

In [72]:

```
CustomerAddress.isnull().sum()
```

Out[72]:

```
customer_id           0
address               0
postcode              0
state                 0
country               0
property_valuation    0
dtype: int64
```

- In that table 0 null value

In [73]:

```
CustomerAddress.duplicated().sum()
```

Out[73]:

```
0
```

In [74]:

```
CustomerAddress.nunique()
```

Out[74]:

```
customer_id           3999
address               3996
postcode               873
state                    5
country                  1
property_valuation      12
dtype: int64
```

Exploring the column

In [75]:

```
CustomerAddress.columns
```

Out[75]:

```
Index(['customer_id', 'address', 'postcode', 'state', 'country',
       'property_valuation'],
      dtype='object')
```

In [76]:

```
CustomerAddress['postcode'].value_counts()
```

Out[76]:

```
2170    31
2155    30
2145    30
2153    29
3977    26
        ..
3808     1
3114     1
4721     1
4799     1
3089     1
Name: postcode, Length: 873, dtype: int64
```

In [77]:

```
CustomerAddress['state'].value_counts()
```

Out[77]:

```
NSW                2054
VIC                 939
QLD                 838
New South Wales      86
Victoria             82
Name: state, dtype: int64
```

In [78]:

```
CustomerAddress['country'].value_counts()
```

Out[78]:

```
Australia    3999
Name: country, dtype: int64
```

In [79]:

```
CustomerAddress['property_valuation'].value_counts()
```

Out[79]:

```
9     647
8     646
10    577
7     493
11    281
6     238
5     225
4     214
12    195
3     186
1     154
2     143
Name: property_valuation, dtype: int64
```

All dataset are correct informatiom

In [ ]:

```

```