In [1]:

```
#loading libraries
import pandas as pd

import warnings
warnings.filterwarnings('ignore')
```

In [2]:

```
# loading the file to get the sheet names
xls= pd.read_excel('KPMG_VI_New_raw_data_update_final.xlsx', sheet_name= None
xls.keys()
```

Out[2]:

```
dict_keys(['Title Sheet', 'Transactions', 'NewCustomerList', 'CustomerDemogra
phic', 'CustomerAddress'])
```

In [3]:

```
# loading each sheet to a dataframe
# from view in excel title sheet is not needed
# also the CustomerDemographic sheet was a test run to explain the task
Transactions_df= pd.read_excel('KPMG_VI_New_raw_data_update_final.xlsx', sheet
NewCustomerList_df= pd.read_excel('KPMG_VI_New_raw_data_update_final.xlsx', sh
CustomerAddress_df= pd.read_excel('KPMG_VI_New_raw_data_update_final.xlsx', sh

# the train dataset for basic checks
CustomerDemographic_df= pd.read_excel('KPMG_VI_New_raw_data_update_final.xlsx'
```

Data Quality Assessment

Assessing the Transaction file

In [4]:

```
# viewing the column and shape of the dataframe
Transactions_df.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
```

- It has 20000 entries and 13 columns
- Only 5 columns do not have null values, so i'll be analyzing these null values
- the id columns are int...as is common with excel files
- the product_first_sold_date should be datetime and not float. From further view...this column is not clear.

In [5]:

```
Transactions_df.head()
```

Out[5]:

	transaction_id	product_id	customer_id	transaction_date	online_order	order_status	brand
0	1	2	2950	2017-02-25	0.0	Approved	Solex
1	2	3	3120	2017-05-21	1.0	Approved	Trek Bicycles
2	3	37	402	2017-10-16	0.0	Approved	OHM Cycles
3	4	88	3135	2017-08-31	0.0	Approved	Norco Bicycles
4	5	78	787	2017-10-01	1.0	Approved	Giant Bicycles

In [6]:

```
Transactions_df.sample(5)
```

Out[6]:

	transaction_id	product_id	customer_id	transaction_date	online_order	order_status	b
7741	7742	43	3315	2017-12-29	0.0	Approved	S
6847	6848	98	920	2017-10-01	1.0	Approved	Bicy
259	260	31	3393	2017-12-28	0.0	Approved	(Bicy
10467	10468	77	173	2017-07-04	1.0	Approved	N Bicy
6292	6293	10	124	2017-03-18	1.0	Approved	S

- The dataset contains transations for the year 2017 only.

In [7]:

```
# checking for duplicates
Transactions_df.duplicated().sum()
```

Out[7]:

```
0
```

-There are no duplicates in the data.

In [8]:

```
# checking for null values
Transactions_df.isna().sum()
```

Out[8]:

```
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 [9]:

```
#selecting columns with null values in product_line column
Null_transactions= Transactions_df[Transactions_df['product_line'].isna()]
Null_transactions.head()
```

Out[9]:

	transaction_id	product_id	customer_id	transaction_date	online_order	order_status	brand
136	137	0	431	2017-09-23	0.0	Approved	NaN
159	160	0	3300	2017-08-27	0.0	Approved	NaN
366	367	0	1614	2017-03-10	0.0	Approved	NaN
406	407	0	2559	2017-06-14	1.0	Approved	NaN
676	677	0	2609	2017-07-02	0.0	Approved	NaN

In [10]:

```
Null_transactions.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 197 entries, 136 to 19871
Data columns (total 13 columns):
 #   Column                   Non-Null Count  Dtype         
---  ------                   --------------  -----         
 0   transaction_id           197 non-null    int64         
 1   product_id               197 non-null    int64         
 2   customer_id              197 non-null    int64         
 3   transaction_date         197 non-null    datetime64[ns]
 4   online_order             195 non-null    float64       
 5   order_status             197 non-null    object        
 6   brand                    0 non-null      object        
 7   product_line             0 non-null      object        
 8   product_class            0 non-null      object        
 9   product_size             0 non-null      object        
 10  list_price               197 non-null    float64       
 11  standard_cost            0 non-null      float64       
 12  product_first_sold_date  0 non-null      float64       
dtypes: datetime64[ns](1), float64(4), int64(3), object(5)
memory usage: 21.5+ KB
```

- All the values in the null_transactions_df have no product_id
- these columns should be dropped: this will take care of null values in 5 other columns as these are cancelled or uncompleted transactions

In [11]:

```
Transactions_df.dropna(subset = ['product_line'], inplace=True)
```

In [12]:

```
Transactions_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 19803 entries, 0 to 19999
Data columns (total 13 columns):
 #   Column                   Non-Null Count  Dtype
---  ------                   --------------  -----
 0   transaction_id           19803 non-null  int64
 1   product_id               19803 non-null  int64
 2   customer_id              19803 non-null  int64
 3   transaction_date         19803 non-null  datetime64[ns]
 4   online_order             19445 non-null  float64
 5   order_status             19803 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               19803 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: 2.1+ MB
```

- now i'm left with only one column with na values, the online_order column

In [13]:

```
Transactions_df.isna().sum()
```

Out[13]:

```
transaction_id               0
product_id                   0
customer_id                  0
transaction_date             0
online_order               358
order_status                 0
brand                        0
product_line                 0
product_class                0
product_size                 0
list_price                   0
standard_cost                0
product_first_sold_date      0
dtype: int64
```

-358 is still a lot of null values and this should be fixed wile gathering the data instead.

- this can be fixed y dropping the values or filling it with one of the two options...after clarifying which it most likely is

In [14]:

```
#dropping null values in the last column
#This can be fixed by replacing it with either online or offline
Transactions_df.dropna(subset=['online_order'], inplace= True)
Transactions_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 19445 entries, 0 to 19999
Data columns (total 13 columns):
 #   Column                   Non-Null Count  Dtype         
---  ------                   --------------  -----         
 0   transaction_id           19445 non-null  int64         
 1   product_id               19445 non-null  int64         
 2   customer_id              19445 non-null  int64         
 3   transaction_date         19445 non-null  datetime64[ns]
 4   online_order             19445 non-null  float64       
 5   order_status             19445 non-null  object        
 6   brand                    19445 non-null  object        
 7   product_line             19445 non-null  object        
 8   product_class            19445 non-null  object        
 9   product_size             19445 non-null  object        
 10  list_price               19445 non-null  float64       
 11  standard_cost            19445 non-null  float64       
 12  product_first_sold_date  19445 non-null  float64       
dtypes: datetime64[ns](1), float64(4), int64(3), object(5)
memory usage: 2.1+ MB
```

In [ ]:

```
```

-Finally the $ in the standard_cost column but not in the list_price column... for uniformity stick to one. make both currency datatype

Assessing the NewCustomerList file

In [15]:

```
NewCustomerList_df.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 [16]:

```
# checking for duplicates
NewCustomerList_df.duplicated().sum()
```

Out[16]:

```
0
```

- No duplicate values

In [17]:
```
NewCustomerList_df.isna().sum()
```

Out[17]:
```
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
Unnamed: 16                             0
Unnamed: 17                             0
Unnamed: 18                             0
Unnamed: 19                             0
Unnamed: 20                             0
Rank                                    0
Value                                   0
dtype: int64
```

- The last name na values should be left untouched
- those with no stated dob should be dropped as they have gender value of U also
-

In [18]:
```
#dropping na values in all but the last name column
NewCustomerList_df.dropna(subset=['DOB', 'job_title', 'job_industry_category']
NewCustomerList_df.isna().sum()
```

Out[18]:
```
first_name                             0
last_name                             20
gender                                 0
past_3_years_bike_related_purchases    0
DOB                                    0
job_title                              0
job_industry_category                  0
wealth_segment                         0
deceased_indicator                     0
owns_car                               0
tenure                                 0
address                                0
postcode                               0
state                                  0
country                                0
property_valuation                     0
Unnamed: 16                            0
Unnamed: 17                            0
Unnamed: 18                            0
Unnamed: 19                            0
Unnamed: 20                            0
Rank                                   0
Value                                  0
dtype: int64
```

In [19]:
```
#checking gender column values for U value
NewCustomerList_df.gender.value_counts()
```

Out[19]:
```
Female    380
Male      355
Name: gender, dtype: int64
```

- deceased indicator is all null so should be removed as it is redundant
- gender U value is undefined and is taken care of by removing DOB null values
- The five unnamed columns are also unclear and should be clarified
- The country column should also be dropped as they are all in Australia
- The state abbreviations should be written in full for people like me who knoww not what they stand for
- property_valuation should take only integers

In [20]:
```
# dropping redundant and unclear columns
NewCustomerList_df.drop(['deceased_indicator', 'country', 'Unnamed: 20', 'Unna
NewCustomerList_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 735 entries, 0 to 999
Data columns (total 16 columns):
 #   Column                               Non-Null Count  Dtype
---  ------                               --------------  -----
 0   first_name                           735 non-null    object
 1   last_name                            715 non-null    object
 2   gender                               735 non-null    object
 3   past_3_years_bike_related_purchases  735 non-null    int64
 4   DOB                                  735 non-null    datetime64[ns]
 5   job_title                            735 non-null    object
 6   job_industry_category                735 non-null    object
 7   wealth_segment                       735 non-null    object
 8   owns_car                             735 non-null    object
 9   tenure                               735 non-null    int64
 10  address                              735 non-null    object
 11  postcode                             735 non-null    int64
 12  state                                735 non-null    object
 13  property_valuation                   735 non-null    int64
```

- post code and address are related too...so why still have address?

In [ ]:

In [ ]:

Assessing the CustomerAddress file

In [21]:
```
CustomerAddress_df.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 [22]:
```
# checking for duplicates
CustomerAddress_df.duplicated().sum()
```

Out[22]:
```
0
```

In [23]:
```
# checking for null values
CustomerAddress_df.isna().sum()
```

Out[23]:
```
customer_id           0
address               0
postcode              0
state                 0
country               0
property_valuation    0
dtype: int64
```

- There are no duplicates and no null values

In [24]:
```
CustomerAddress_df.state.value_counts()
```

Out[24]:
```
NSW                2054
VIC                 939
QLD                 838
New South Wales      86
Victoria             82
Name: state, dtype: int64
```

- the names of the states should all be changed to abbreviatons

In [ ]:
```
```