KPMG Virtual Experience Program

TASK 1 - Data Quality Assessment

Assessment of data quality and completeness in preparation for analysis.

The 3 data sets provided to KPMG by the client:

1. Customer Demographic

2. Customer Address

3. Transaction data in the past 3 months

```
#importing library
import pandas as pd
```

Reading the data

```
#reading the excel file
data = pd.ExcelFile("KPMG.xlsx")
```

```
#reading individual sheets present in the excel file
Transactions = pd.read_excel(data, 'Transactions')
NewCustomerList = pd.read_excel(data, 'NewCustomerList')
CustomerDemographic = pd.read_excel(data, 'CustomerDemographic')
CustomerAddress = pd.read_excel(data, 'CustomerAddress')
```

```
<ipython-input-3-7e31c9dbc7f2>:2: FutureWarning: Inferring datetime64[ns] from data containing strings is deprecated and will be re
  NewCustomerList = pd.read_excel(data, 'NewCustomerList')
<ipython-input-3-7e31c9dbc7f2>:3: FutureWarning: Inferring datetime64[ns] from data containing strings is deprecated and will be re
  CustomerDemographic = pd.read_excel(data, 'CustomerDemographic')
```

Exploring **Transactions** dataset.

```
Transactions.head()
```

	transaction_id	product_id	customer_id	transaction_date	online_order	order_
0	1	2	2950	2017-02-25	0.0	A
1	2	3	3120	2017-05-21	1.0	A
2	3	37	402	2017-10-16	0.0	A
3	4	88	3135	2017-08-31	0.0	A
4	5	78	787	2017-10-01	1.0	A

```
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
```

```
Transactions.shape
```

```
(20000, 13)
```

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

```
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
```

There are missing values in 7 columns. They can be deleted or treated according to the nature of analysis.

```
Transactions.duplicated().sum()
```

```
0
```

There are no duplicate values. So we can tell the data is unique.

```
Transactions.nunique()
```

```
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
```

```
Transactions.columns
```

```
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')
```

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

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

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

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

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

```
Standard    14176
Road         3970
```

```
Touring      1234
Mountain      423
Name: product_line, dtype: int64
```

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

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

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

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

```
Transactions['product_first_sold_date']
```

```
0        41245.0
1        41701.0
2        36361.0
3        36145.0
4        42226.0
          ...
19995    37823.0
19996    35560.0
19997    40410.0
19998    38216.0
19999    36334.0
Name: product_first_sold_date, Length: 20000, dtype: float64
```

```
Transactions['product_first_sold_date'] = pd.to_datetime(Transactions['product_first_sold_date'], unit = 's')
```

```
Transactions['product_first_sold_date'].head(5)
```

```
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
Name: product_first_sold_date, dtype: datetime64[ns]
```

```
Transactions['product_first_sold_date'].head(25)
```

```
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
10   1970-01-01 11:19:44
11   1970-01-01 11:42:52
12   1970-01-01 09:35:27
13   1970-01-01 09:36:26
14   1970-01-01 10:36:33
15   1970-01-01 10:31:13
16   1970-01-01 10:36:46
17   1970-01-01 09:24:48
18   1970-01-01 11:05:15
19   1970-01-01 10:22:17
20   1970-01-01 10:05:34
21   1970-01-01 10:06:01
22   1970-01-01 11:42:25
23   1970-01-01 11:46:44
24   1970-01-01 09:27:59
Name: product_first_sold_date, dtype: datetime64[ns]
```

The values in the product_first_sold_date columns are incorrect as it shows everything happening on the same day but at different times.

Exploring **New Customer List** Dataset.

```
NewCustomerList.head()
```

s	DOB	job_title	job_industry_category	wealth_segment	deceased_indicator	own
6	1957-07-12	General Manager	Manufacturing	Mass Customer	N	
9	1970-03-22	Structural Engineer	Property	Mass Customer	N	
0	1974-08-28	Senior Cost Accountant	Financial Services	Affluent Customer	N	
4	1979-01-28	Account Representative III	Manufacturing	Affluent Customer	N	
4	1965-09-21	Financial Analyst	Financial Services	Affluent Customer	N	

```
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
```

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

```
NewCustomerList.head()
NewCustomerList.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1000 entries, 0 to 999
Data columns (total 18 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  Rank                                 1000 non-null   int64
 17  Value                                1000 non-null   float64
dtypes: datetime64[ns](1), float64(1), int64(5), object(11)
memory usage: 140.8+ KB
```

```
NewCustomerList.shape
```

```
(1000, 18)
```

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

```
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
```

There are missing values in 4 columns. They can be deleted or treated according to the nature of analysis

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

```
0
```

There are no duplicate values. So, we can tell the data is unique.

```
NewCustomerList.nunique()
```

```
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 the columns of **NewCustomerList**.

```
NewCustomerList.columns
```

```
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')
```

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

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

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

There are 17 columns with unknown/unspecified gender.

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

```
1998-02-05    2
1978-01-15    2
1977-11-08    2
1951-11-28    2
1979-07-28    2
             ..
1945-08-08    1
1943-08-27    1
1999-10-24    1
1976-01-24    1
1955-10-02    1
Name: DOB, Length: 958, dtype: int64
```

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

```
Associate Professor         15
Environmental Tech          14
Software Consultant         14
Chief Design Engineer       13
Assistant Manager           12
                            ..
Accountant II                1
Programmer IV                1
Administrative Officer       1
Accounting Assistant III     1
Web Developer I              1
Name: job_title, Length: 184, dtype: int64
```

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

```
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
```

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

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

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

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

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

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

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

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

Exploring **Customer Demographic** Data Set

```
CustomerDemographic.head()
```

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

```
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
```

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

```
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
```

There are missing values in 5 columns. They can be deleted or treated according to the nature of analysis

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

```
0
```

There are no duplicate values. So we can say the data is unique.

```
CustomerDemographic.nunique()
```

```
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
```

Exploring the columns of **Customer Demographic** Data Set

```
CustomerDemographic.columns
```

```
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')
```

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

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

Certain categories are not correctly titled. So, the names in these categories can be re-named.

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

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

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

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

```
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
```

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

```
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
```

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

```
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
```

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

```
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
```

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

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

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

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

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

```
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¸ëë°í ë¥´                                  27
,ãã»:*:ã»ãâ( â» Ï â» )ãã»:*:ã»ãâ           25
Name: default, Length: 90, dtype: int64
```

These values are inconsistent. Hence, we are dropping the column.

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

	customer_id	first_name	last_name	gender	past_3_years_bike_related_purchases	DOB	job_title	job_industry_categor
0	1	Laraine	Medendorp	Female	93	1953-10-12	Executive Secretary	Healt
1	2	Eli	Bockman	Male	81	1980-12-16	Administrative Officer	Financial Service
2	3	Arlin	Dearle	Male	61	1954-01-20	Recruiting Manager	Propert
3	4	Talbot	NaN	Male	33	1961-10-03	NaN	I
4	5	Sheila-kathryn	Calton	Female	56	1977-05-13	Senior Editor	Nal
...	...	...	...	...	...	...	...	.
3995	3996	Rosalia	Halgarth	Female	8	1975-08-09	VP Product Management	Healt

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

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

3998 3999 Patrizius NaN Male 11 NaN Manufacturin

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

```
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
```

Exploring **Customer Address** Data Set

```
CustomerAddress.head()
```

	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

```
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
```

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

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

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

```
0
```

There are no duplicate values.

```
CustomerAddress.nunique()
```

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

Exploring the columns of **Customer Address** Data Set

```
CustomerAddress.columns
```

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

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

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

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

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

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

```
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 columns are having consistent information.

✓ 0s completed at 02:00 ● ×