KPMG Task1

(https://databricks.com)

```
import pandas as pd
from pandas import ExcelWriter
from pandas import ExcelFile
from openpyxl import load_workbook
```

Reading The Data

```
data = pd.ExcelFile(r"C:\Users\Muhammed\Documents\KPMG data.xlsx")
```

```
Transactions = pd.read_excel(data, 'Transactions',header = 1, index_col= None)
NewCustomerList = pd.read_excel(data, 'NewCustomerList',header = 1, index_col=
None)
CustomerDemographic = pd.read_excel(data, 'CustomerDemographic',header = 1,
index_col= None)
CustomerAddress = pd.read_excel(data, 'CustomerAddress',header = 1, index_col=
None)
```

```
C:\Users\Muhammed\AppData\Local\Temp\ipykernel_29008\2683626855.py:2: FutureWar
ning: Inferring datetime64[ns] from data containing strings is deprecated and w
ill be removed in a future version. To retain the old behavior explicitly pass
Series(data, dtype=datetime64[ns])
  NewCustomerList = pd.read_excel(data, 'NewCustomerList',header = 1, index_col
= None)
C:\Users\Muhammed\AppData\Local\Temp\ipykernel_29008\2683626855.py:3: FutureWar
ning: Inferring datetime64[ns] from data containing strings is deprecated and w
ill be removed in a future version. To retain the old behavior explicitly pass
Series(data, dtype=datetime64[ns])
  CustomerDemographic = pd.read_excel(data, 'CustomerDemographic',header = 1, i
ndex_col= None)
```

```
#defining functions for analysis
def initial_analysis(data):


    # Display the summary information of the DataFrame
    print("Info:")
    print(data.info())
    print()

    # Display the statistical summary of the DataFrame
    print("Describe:")
    print(data.describe())
    print()




    # Check for null values in the DataFrame
    print("Null values:")
    print(data.isnull().sum())
    print()

    # Check for duplicate rows in the DataFrame
    print("Duplicates:")
    print(data.duplicated().sum())


def show_value_counts(data):
    for column in data.columns:
        print(f"Value counts for column '{column}':")
        print(data[column].value_counts())
        print()
```

Exploring Transactions Dataset

```
Transactions.head(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

Exploring The Columns

Checking the columns to find if they have consistent and correct information

```
initial_analysis(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
```

Seven columns contain missing values

There are zero Duplicated Values

```
show_value_counts(Transactions)
```

```
Value counts for column 'transaction_id':
1        1
13331    1
13338    1
13337    1
13336    1
        ..
6667     1
6666     1
6665     1
6664     1
20000    1
Name: transaction_id, Length: 20000, dtype: int64

Value counts for column 'product_id':
0      1378
3       354
1       311
35      268
38      267
       ...
```

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

```
Out[58]: 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]
```

Converted product_first_sold_date column to datetime. Datetime not represented in appropriate format.

Exploring NewCustomerList Dataset

```
NewCustomerList.head()
```

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

5 rows × 23 columns

```
initial_analysis(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
```

There are four columns with Null Values

There are no duplicated values

Exploring The Columns

Checking the columns to find if they have consistent and correct information

```
show_value_counts(NewCustomerList)
```

```
Value counts for column 'first_name':
Rozamond     3
Dorian       3
Mandie       3
Inglebert    2
Ricki        2
            ..
Diego        1
Lucilia      1
Eddy         1
Caron        1
Sylas        1
Name: first_name, Length: 940, dtype: int64

Value counts for column 'last_name':
Sissel       2
Minshall     2
Borsi        2
Shoesmith    2
Sturch       2
            ..
```

```
# Replace 'U' with 'Unspecified'
NewCustomerList['gender'] =
NewCustomerList['gender'].str.replace('U','Unspecified')
```

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

```
Out[63]: Female         513
Male           470
Unspecified     17
Name: gender, dtype: int64
```

Changed U to Unspecified for better readability

Exploring Customer Address Dataset

```
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

```
initial_analysis(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
None

Describe:
       customer_id     postcode  property_valuation
count  3999.000000  3999.000000         3999.000000
mean   2003.987997  2985.755939            7.514379
std    1154.576912   844.878364            2.824663
```

0 Null cells and Duplicates

Exploring The Columns

Checking the columns to find if they have consistent and correct information

```
show_value_counts(CustomerAddress)
```

```
Value counts for column 'customer_id':
1       1
2676    1
2663    1
2664    1
2665    1
       ..
1343    1
1344    1
1345    1
1346    1
4003    1
Name: customer_id, Length: 3999, dtype: int64

Value counts for column 'address':
3 Mariners Cove Terrace      2
3 Talisman Place             2
64 Macpherson Junction       2
359 Briar Crest Road         1
4543 Service Terrace         1
                            ..
```

Exploring CustomerDemographic Dataset

```
CustomerDemographic.head()
```

	customer_id	first_name	last_name	gender	past_3_years_bike_related_purchases	DOB	j
0	1	Laraine	Medendorp	F	93	1953-10-12	Ex S
1	2	Eli	Bockman	Male	81	1980-12-16	Admin
2	3	Arlin	Dearle	Male	61	1954-01-20	Re M
3	4	Talbot	NaN	Male	33	1961-10-03	
4	5	Sheila-kathryn	Calton	Female	56	1977-05-13	Senic

```
initial_analysis(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
```

Contains Null Values in Six columns of the dataset.

Contains Zero dupilcated data

Exploring The Columns

Checking the columns to find if they have consistent and correct information

```
show_value_counts(CustomerDemographic)
```

```
Value counts for column 'customer_id':
1       1
2672    1
2659    1
2660    1
2661    1
        ..
1339    1
1340    1
1341    1
```

```
1342    1
4000    1
Name: customer_id, Length: 4000, dtype: int64

Value counts for column 'first_name':
Max         5
Tobe        5
Timmie      5
Kippy       4
Pail        4
```

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

Dropped default column because of inconsistent data

```
CustomerDemographic_DOB = CustomerDemographic.sort_values('DOB',
ascending=True)
print(CustomerDemographic_DOB['DOB'])
```

```
33     1843-12-21
719    1931-10-23
1091   1935-08-22
3409   1940-09-22
2412   1943-08-11
          ...
3778          NaT
3882          NaT
3930          NaT
3934          NaT
3997          NaT
Name: DOB, Length: 4000, dtype: datetime64[ns]
```

Data shows DOB of a person to be in 1843 this would imply the person is 180 years old this appears to be an error

```
#Re-naming the categories
CustomerDemographic['gender'] =
CustomerDemographic['gender'].replace('F','Female').replace('M','Male').replace
('Femal','Female').replace('U','Unspecified')
```

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

```
Out[73]: Female         2039
Male           1873
Unspecified      88
Name: gender, dtype: int64
```