

KPMG VIRTUAL INTERNSHIP PROJECT

TASK: 1 - Data Quality Assessment

Assessment of data quality and completeness in preparation for analysis. The client provided KPMG with 3 datasets:

- 1.Customer Demographic
- 2.Customer Addresses
- 3.Transactions data in the past 3 months

In [1]:

```
import pandas as pd
```

Reading each file separately

In [2]:

```
Transactions = pd.read_excel(r"E:\Forage Internship\KPMG\KPMG_VI_New_raw_data_update_final.  
NewCustomerList = pd.read_excel(r"E:\Forage Internship\KPMG\KPMG_VI_New_raw_data_update_fin  
CustomerDemographic = pd.read_excel(r"E:\Forage Internship\KPMG\KPMG_VI_New_raw_data_update  
CustomerAddress = pd.read_excel(r"E:\Forage Internship\KPMG\KPMG_VI_New_raw_data_update_fin
```

Exploring Transactions Data Set

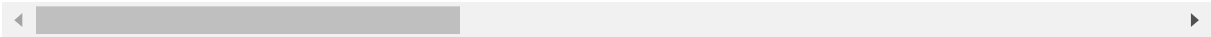
In [3]:

```
Transactions.head(10)
```

Out[3]:

	transaction_id	product_id	customer_id	transaction_date	online_order	order_status	brand
0	1	2	2950	2017-02-25	0.0	Approved	Sc
1	2	3	3120	2017-05-21	1.0	Approved	T Bicyc
2	3	37	402	2017-10-16	0.0	Approved	Ol Cyc
3	4	88	3135	2017-08-31	0.0	Approved	No Bicyc
4	5	78	787	2017-10-01	1.0	Approved	Gi Bicyc
5	6	25	2339	2017-03-08	1.0	Approved	Gi Bicyc
6	7	22	1542	2017-04-21	1.0	Approved	WeareA
7	8	15	2459	2017-07-15	0.0	Approved	WeareA
8	9	67	1305	2017-08-10	0.0	Approved	Sc
9	10	12	3262	2017-08-30	1.0	Approved	WeareA

10 rows × 26 columns



In [4]:

```
Transactions.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 20000 entries, 0 to 19999
Data columns (total 26 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
13  Unnamed: 13                           0 non-null      float64
14  Unnamed: 14                           0 non-null      float64
15  Unnamed: 15                           0 non-null      float64
16  Unnamed: 16                           0 non-null      float64
17  Unnamed: 17                           0 non-null      float64
18  Unnamed: 18                           0 non-null      float64
19  Unnamed: 19                           0 non-null      float64
20  Unnamed: 20                           0 non-null      float64
21  Unnamed: 21                           0 non-null      float64
22  Unnamed: 22                           0 non-null      float64
23  Unnamed: 23                           0 non-null      float64
24  Unnamed: 24                           0 non-null      float64
25  Unnamed: 25                           0 non-null      float64
dtypes: datetime64[ns](1), float64(17), int64(3), object(5)
memory usage: 4.0+ MB
```

In [5]:

```
#Using only the required columns
Transactions = Transactions.iloc[:, 0:13]
Transactions.head(10)
```

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	Sc
1	2	3	3120	2017-05-21	1.0	Approved	T Bicyc
2	3	37	402	2017-10-16	0.0	Approved	Ol Cyc
3	4	88	3135	2017-08-31	0.0	Approved	No Bicyc
4	5	78	787	2017-10-01	1.0	Approved	Gi Bicyc
5	6	25	2339	2017-03-08	1.0	Approved	Gi Bicyc
6	7	22	1542	2017-04-21	1.0	Approved	WeareA
7	8	15	2459	2017-07-15	0.0	Approved	WeareA
8	9	67	1305	2017-08-10	0.0	Approved	Sc
9	10	12	3262	2017-08-30	1.0	Approved	WeareA

In [6]:

```
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 [7]:

```
#Checking the shape of the data  
Transactions.shape
```

Out[7]:

```
(20000, 13)
```

In [8]:

```
#Checking for null values  
Transactions.isnull().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
```

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

In [9]:

```
#Checking for duplicate values  
Transactions.duplicated().sum()
```

Out[9]:

```
0
```

There are no duplicate values, so the data is unique

In [10]:

```
#check for uniqueness of each column  
Transactions.nunique()
```

Out[10]:

```
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 [11]:

```
Transactions.columns
```

Out[11]:

```
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 [12]:

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

Out[12]:

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

In [13]:

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

Out[13]:

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

In [14]:

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

Out[14]:

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

In [15]:

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

Out[15]:

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

In [16]:

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

Out[16]:

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

In [17]:

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

Out[17]:

```
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
```

In [18]:

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

Out[18]:

```
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]
```

In [19]:

```
Transactions['product_first_sold_date'].head(20)
```

Out[19]:

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

The values in the product_first_sold_date columns are not correct as it shows everything happening the same day at different times.

Exploring New Customer List Data Set

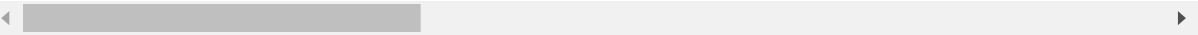
In [20]:

```
NewCustomerList.head(10)
```

Out[20]:

	first_name	last_name	gender	past_3_years_bike_related_purchases	DOB	job_title	jc
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	Druci	Brandli	Female		39 1951-04-29	Assistant Media Planner	
6	Rutledge	Hallt	Male		23 1976-10-06	Compensation Analyst	
7	Nancie	Vian	Female		74 1972-12-27	Human Resources Assistant II	
8	Duff	Karlowicz	Male		50 1972-04-28	Speech Pathologist	
9	Barthel	Docket	Male		72 1985-08-02	Accounting Assistant IV	

10 rows × 23 columns



In [21]:

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 [22]:

#Dropping the unnamed columns

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

In [23]:

#Checking the shape of the dataset

NewCustomerList.shape

Out[23]:

(1000, 18)

In [24]:

```
#Checking for null values  
NewCustomerList.isnull().sum()
```

Out[24]:

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 dropped or treated according to the nature of analysis

In [25]:

```
#Checking for duplicate values  
NewCustomerList.duplicated().sum()
```

Out[25]:

0

There are no duplicate values

In [26]:

```
#Checking for uniqueness of each column  
NewCustomerList.nunique()
```

Out[26]:

```
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

In [27]:

```
NewCustomerList.columns
```

Out[27]:

```
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 [28]:

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

Out[28]:

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

In [29]:

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

Out[29]:

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

There are 17 columns with unknown/unspecified gender.

In [30]:

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

Out[30]:

```
1993-11-02    2
1994-04-15    2
1963-08-25    2
1995-08-13    2
1987-01-15    2
..
1958-05-14    1
1977-12-08    1
1993-12-19    1
1954-10-06    1
1995-10-19    1
Name: DOB, Length: 958, dtype: int64
```

In [31]:

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

Out[31]:

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

In [32]:

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

Out[32]:

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

In [33]:

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

Out[33]:

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

In [34]:

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

Out[34]:

No 507
Yes 493
Name: owns_car, dtype: int64

In [35]:

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

Out[35]:

N 1000
Name: deceased_indicator, dtype: int64

Exploring Customer Demographic Data Set

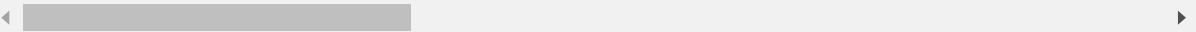
In [36]:

```
CustomerDemographic.head(10)
```

Out[36]:

	customer_id	first_name	last_name	gender	past_3_years_bike_related_purchases	DOB	
0	1	Laraine	Medendorp	F	93	1953-10-12	
1	2	Eli	Bockman	Male	81	1980-12-16	Ad
2	3	Arlin	Dearle	Male	61	1954-01-20	
3	4	Talbot	NaN	Male	33	1961-10-03	
4	5	Sheila-kathryn	Calton	Female	56	1977-05-13	S
5	6	Curr	Duckhouse	Male	35	1966-09-16	
6	7	Fina	Merali	Female	6	1976-02-23	
7	8	Rod	Inder	Male	31	1962-03-30	
8	9	Mala	Lind	Female	97	1973-03-10	D.
9	10	Fiorenze	Birdall	Female	49	1988-10-11	Se

10 rows × 26 columns



In [37]:

CustomerDemographic.info()

<class 'pandas.core.frame.DataFrame'>

RangeIndex: 4000 entries, 0 to 3999

Data columns (total 26 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
13	Unnamed: 13	0 non-null	float64
14	Unnamed: 14	0 non-null	float64
15	Unnamed: 15	0 non-null	float64
16	Unnamed: 16	0 non-null	float64
17	Unnamed: 17	0 non-null	float64
18	Unnamed: 18	0 non-null	float64
19	Unnamed: 19	0 non-null	float64
20	Unnamed: 20	0 non-null	float64
21	Unnamed: 21	0 non-null	float64
22	Unnamed: 22	0 non-null	float64
23	Unnamed: 23	0 non-null	float64
24	Unnamed: 24	0 non-null	float64
25	Unnamed: 25	0 non-null	float64

dtypes: datetime64[ns](1), float64(14), int64(2), object(9)

memory usage: 812.6+ KB

In [38]:

```
#Checking for null values  
CustomerDemographic.isnull().sum()
```

Out[38]:

```
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  
Unnamed: 13          4000  
Unnamed: 14          4000  
Unnamed: 15          4000  
Unnamed: 16          4000  
Unnamed: 17          4000  
Unnamed: 18          4000  
Unnamed: 19          4000  
Unnamed: 20          4000  
Unnamed: 21          4000  
Unnamed: 22          4000  
Unnamed: 23          4000  
Unnamed: 24          4000  
Unnamed: 25          4000  
dtype: int64
```

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

In [39]:

```
#Checking for duplicate data  
CustomerDemographic.duplicated().sum()
```

Out[39]:

0

There are no duplicate values

In [40]:

```
#Checking for uniqueness of each column  
CustomerDemographic.nunique()
```

Out[40]:

```
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  
Unnamed: 13           0  
Unnamed: 14           0  
Unnamed: 15           0  
Unnamed: 16           0  
Unnamed: 17           0  
Unnamed: 18           0  
Unnamed: 19           0  
Unnamed: 20           0  
Unnamed: 21           0  
Unnamed: 22           0  
Unnamed: 23           0  
Unnamed: 24           0  
Unnamed: 25           0  
dtype: int64
```

Exploring the columns

In [41]:

```
CustomerDemographic.columns
```

Out[41]:

```
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', 'Unnamed: 13', 'Unnamed: 14',  
      'Unnamed: 15', 'Unnamed: 16', 'Unnamed: 17', 'Unnamed: 18',  
      'Unnamed: 19', 'Unnamed: 20', 'Unnamed: 21', 'Unnamed: 22',  
      'Unnamed: 23', 'Unnamed: 24', 'Unnamed: 25'],  
      dtype='object')
```

In [42]:

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

Out[42]:

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

Certain categories are not correctly titled. The names in these categories are re-named

In [43]:

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

In [44]:

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

Out[44]:

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

In [45]:

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

Out[45]:

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

In [46]:

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

Out[46]:

```
1978-01-30    7
1978-08-19    4
1964-07-08    4
1976-09-25    4
1976-07-16    4
..
2001-01-22    1
1955-03-06    1
1966-08-05    1
1968-11-16    1
1958-08-02    1
Name: DOB, Length: 3448, dtype: int64
```

In [47]:

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

Out[47]:

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

In [48]:

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

Out[48]:

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

In [49]:

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

Out[49]:

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

In [50]:

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

Out[50]:

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

In [51]:

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

Out[51]:

```
100      113
1         112
-1        111
-100       99
ÛÿÛÿÛÿ    53
...
<img src=x onerror=alert('hi') />    31
/dev/null; touch /tmp/blns.fail ; echo 30
âââtestââ    29
ì,ëë°í ë¥´    27
,ãã»:»:ããã( â» Ì â» )ãã»:»:ããã    25
Name: default, Length: 90, dtype: int64
```

In [52]:

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

The values are inconsistent, hence dropping the column.

In [53]:

```
CustomerDemographic.head(10)
```

Out[53]:

	customer_id	first_name	last_name	gender	past_3_years_bike_related_purchases	DOB	
0	1	Laraine	Medendorp	Female	93	1953-10-12	
1	2	Eli	Bockman	Male	81	1980-12-16	Ad
2	3	Arlin	Dearle	Male	61	1954-01-20	
3	4	Talbot	NaN	Male	33	1961-10-03	
4	5	Sheila-kathryn	Calton	Female	56	1977-05-13	S
5	6	Curr	Duckhouse	Male	35	1966-09-16	
6	7	Fina	Merali	Female	6	1976-02-23	
7	8	Rod	Inder	Male	31	1962-03-30	
8	9	Mala	Lind	Female	97	1973-03-10	D
9	10	Fiorenze	Birdall	Female	49	1988-10-11	Se

10 rows × 25 columns



In [54]:

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

Out[54]:

Yes 2024
No 1976
Name: owns_car, dtype: int64

In [55]:

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

Out[55]:

7.0	235
5.0	228
11.0	221
10.0	218
16.0	215
8.0	211
18.0	208
12.0	202
14.0	200
9.0	200
6.0	192
4.0	191
13.0	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

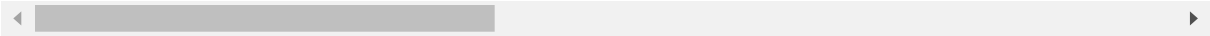
In [56]:

```
CustomerAddress.head(10)
```

Out[56]:

	customer_id	address	postcode	state	country	property_valuation	Unnamed: 6	Unnamed: 7
0	1	060 Morning Avenue	2016	New South Wales	Australia	10	NaN	NaN
1	2	6 Meadow Vale Court	2153	New South Wales	Australia	10	NaN	NaN
2	4	0 Holy Cross Court	4211	QLD	Australia	9	NaN	NaN
3	5	17979 Del Mar Point	2448	New South Wales	Australia	4	NaN	NaN
4	6	9 Oakridge Court	3216	VIC	Australia	9	NaN	NaN
5	7	4 Delaware Trail	2210	New South Wales	Australia	9	NaN	NaN
6	8	49 Londonderry Lane	2650	New South Wales	Australia	4	NaN	NaN
7	9	97736 7th Trail	2023	New South Wales	Australia	12	NaN	NaN
8	11	93405 Ludington Park	3044	VIC	Australia	8	NaN	NaN
9	12	44339 Golden Leaf Alley	4557	QLD	Australia	4	NaN	NaN

10 rows × 26 columns



In [57]:

```
CustomerAddress.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 3999 entries, 0 to 3998
Data columns (total 26 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
6   Unnamed: 6            0 non-null      float64
7   Unnamed: 7            0 non-null      float64
8   Unnamed: 8            0 non-null      float64
9   Unnamed: 9            0 non-null      float64
10  Unnamed: 10           0 non-null      float64
11  Unnamed: 11           0 non-null      float64
12  Unnamed: 12           0 non-null      float64
13  Unnamed: 13           0 non-null      float64
14  Unnamed: 14           0 non-null      float64
15  Unnamed: 15           0 non-null      float64
16  Unnamed: 16           0 non-null      float64
17  Unnamed: 17           0 non-null      float64
18  Unnamed: 18           0 non-null      float64
19  Unnamed: 19           0 non-null      float64
20  Unnamed: 20           0 non-null      float64
21  Unnamed: 21           0 non-null      float64
22  Unnamed: 22           0 non-null      float64
23  Unnamed: 23           0 non-null      float64
24  Unnamed: 24           0 non-null      float64
25  Unnamed: 25           0 non-null      float64
dtypes: float64(20), int64(3), object(3)
memory usage: 812.4+ KB
```

In [58]:

```
#Checking for null values.  
CustomerAddress.isnull().sum()
```

Out[58]:

```
customer_id      0  
address          0  
postcode         0  
state            0  
country          0  
property_valuation  0  
Unnamed: 6       3999  
Unnamed: 7       3999  
Unnamed: 8       3999  
Unnamed: 9       3999  
Unnamed: 10      3999  
Unnamed: 11      3999  
Unnamed: 12      3999  
Unnamed: 13      3999  
Unnamed: 14      3999  
Unnamed: 15      3999  
Unnamed: 16      3999  
Unnamed: 17      3999  
Unnamed: 18      3999  
Unnamed: 19      3999  
Unnamed: 20      3999  
Unnamed: 21      3999  
Unnamed: 22      3999  
Unnamed: 23      3999  
Unnamed: 24      3999  
Unnamed: 25      3999  
dtype: int64
```

There are no null values

In [59]:

```
#Checking for duplicate values  
CustomerAddress.duplicated().sum()
```

Out[59]:

```
0
```

There are no duplicate values.

In [60]:

```
#Checking for uniqueness of each column  
CustomerAddress.nunique()
```

Out[60]:

```
customer_id      3999  
address          3996  
postcode         873  
state            5  
country          1  
property_valuation 12  
Unnamed: 6       0  
Unnamed: 7       0  
Unnamed: 8       0  
Unnamed: 9       0  
Unnamed: 10      0  
Unnamed: 11      0  
Unnamed: 12      0  
Unnamed: 13      0  
Unnamed: 14      0  
Unnamed: 15      0  
Unnamed: 16      0  
Unnamed: 17      0  
Unnamed: 18      0  
Unnamed: 19      0  
Unnamed: 20      0  
Unnamed: 21      0  
Unnamed: 22      0  
Unnamed: 23      0  
Unnamed: 24      0  
Unnamed: 25      0  
dtype: int64
```

Exploring the columns

In [61]:

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

Out[61]:

```
2170    31  
2145    30  
2155    30  
2153    29  
3977    26  
..  
3331     1  
3036     1  
3321     1  
3305     1  
2143     1  
Name: postcode, Length: 873, dtype: int64
```

In [62]:

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

Out[62]:

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

In [63]:

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

Out[63]:

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

In [64]:

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

Out[64]:

```
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 the columns appear to have consistent and correct information

CONCLUSION

The given data is analysed and investigated in all aspects to inspect the Quality of data with respect to all qualities mentioned in the Data Quality Framework Table

- Accuracy
- Completeness
- Uniqueness
- Validity
- Consistency
- Relevancy
- Timeliness

