Data Analysis on Online Retail

In [1]:
```
# import the necessary libraries
import numpy as np
import pandas as pd

# for visuals
import seaborn as sns
import matplotlib.pyplot as plt
```

In [3]:
```
df = pd.read_csv(r'C:\Users\user\Downloads\OnlineRetail.csv')
df
```

Out[3]:

	InvoiceNo	StockCode	Description	Quantity	InvoiceDate	UnitPrice	CustomerID	Country
0	536365	85123A	WHITE HANGING HEART T-LIGHT HOLDER	6	12/1/2010 08:26	2.55	17850.0	United Kingdom
1	536365	71053	WHITE METAL LANTERN	6	12/1/2010 08:26	3.39	17850.0	United Kingdom
2	536365	84406B	CREAM CUPID HEARTS COAT HANGER	8	12/1/2010 08:26	2.75	17850.0	United Kingdom
3	536365	84029G	KNITTED UNION FLAG HOT WATER BOTTLE	6	12/1/2010 08:26	3.39	17850.0	United Kingdom
4	536365	84029E	RED WOOLLY HOTTIE WHITE HEART.	6	12/1/2010 08:26	3.39	17850.0	United Kingdom
...	...	...	...	...	...	...	...	...
541904	581587	22613	PACK OF 20 SPACEBOY NAPKINS	12	12/9/2011 12:50	0.85	12680.0	France
541905	581587	22899	CHILDREN'S APRON DOLLY GIRL	6	12/9/2011 12:50	2.10	12680.0	France
541906	581587	23254	CHILDRENS CUTLERY DOLLY GIRL	4	12/9/2011 12:50	4.15	12680.0	France
541907	581587	23255	CHILDRENS CUTLERY CIRCUS PARADE	4	12/9/2011 12:50	4.15	12680.0	France
541908	581587	22138	BAKING SET 9 PIECE RETROSPOT	3	12/9/2011 12:50	4.95	12680.0	France

541909 rows × 8 columns

In [4]:
```
df.shape
```

Out[4]:
```
(541909, 8)
```

In [5]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 541909 entries, 0 to 541908
Data columns (total 8 columns):
 #   Column       Non-Null Count   Dtype  
---  ------       --------------   -----  
 0   InvoiceNo    541909 non-null  object 
 1   StockCode    541909 non-null  object 
 2   Description  540455 non-null  object 
 3   Quantity     541909 non-null  int64  
 4   InvoiceDate  541909 non-null  object 
 5   UnitPrice    541909 non-null  float64
 6   CustomerID   406829 non-null  float64
 7   Country      541909 non-null  object 
dtypes: float64(2), int64(1), object(5)
memory usage: 33.1+ MB
```

In [6]:
```
# view summary statistics
df.describe()
```

Out[6]:

	Quantity	UnitPrice	CustomerID
count	541909.000000	541909.000000	406829.000000
mean	9.552250	4.611114	15287.690570
std	218.081158	96.759853	1713.600303
min	-80995.000000	-11062.060000	12346.000000
25%	1.000000	1.250000	13953.000000
50%	3.000000	2.080000	15152.000000
75%	10.000000	4.130000	16791.000000
max	80995.000000	38970.000000	18287.000000

Cleaning and Manipulation

- you may choose to delete rows or column, modify content as filling in empty values, etc.

Why clean data?

- to prevent a misrepresentation of your dataset
- to prevent time wastage
- to avoid biases of your analysis

Overview of cleaning steps

- Handle missing values: you can delete them, fill them with a value that make sense.
- Check for data consistency: case maybe important for strings, formatting etc.
- Handle outliers
- Remove duplicates
- validate correction of entries; age columns shouldn't contain text for instance

In [7]:
```
# check missing values
df.isna().sum()
```

Out[7]:
```
InvoiceNo           0
StockCode           0
Description      1454
Quantity            0
InvoiceDate         0
UnitPrice           0
CustomerID     135080
Country             0
dtype: int64
```

In [8]:
```
# what do the records with empty customerID mean?
# does it mean that the sales wasn't recorded to a customer?
'''it is either the sales was not recorded to a customer or somebody else[a new custom
```

Out[8]:
```
'it is either the sales was not recorded to a customer or somebody else[a new custome
r] carried out the sale'
```

In [9]:
```
df[df['CustomerID'].isna()]
```

Out[9]:

	InvoiceNo	StockCode	Description	Quantity	InvoiceDate	UnitPrice	CustomerID	Country
622	536414	22139	NaN	56	12/1/2010 11:52	0.00	NaN	United Kingdom
1443	536544	21773	DECORATIVE ROSE BATHROOM BOTTLE	1	12/1/2010 14:32	2.51	NaN	United Kingdom
1444	536544	21774	DECORATIVE CATS BATHROOM BOTTLE	2	12/1/2010 14:32	2.51	NaN	United Kingdom
1445	536544	21786	POLKADOT RAIN HAT	4	12/1/2010 14:32	0.85	NaN	United Kingdom
1446	536544	21787	RAIN PONCHO RETROSPOT	2	12/1/2010 14:32	1.66	NaN	United Kingdom
...	...	...	...	...	...	...	...	...
541536	581498	85099B	JUMBO BAG RED RETROSPOT	5	12/9/2011 10:26	4.13	NaN	United Kingdom
541537	581498	85099C	JUMBO BAG BAROQUE BLACK WHITE	4	12/9/2011 10:26	4.13	NaN	United Kingdom
541538	581498	85150	LADIES & GENTLEMEN METAL SIGN	1	12/9/2011 10:26	4.96	NaN	United Kingdom
541539	581498	85174	S/4 CACTI CANDLES	1	12/9/2011 10:26	10.79	NaN	United Kingdom
541540	581498	DOT	DOTCOM POSTAGE	1	12/9/2011 10:26	1714.17	NaN	United Kingdom

135080 rows × 8 columns

In [10]:
```
# what about records with empty description
df[df['Description'].isna()]
```

Out[10]:

	InvoiceNo	StockCode	Description	Quantity	InvoiceDate	UnitPrice	CustomerID	Country
622	536414	22139	NaN	56	12/1/2010 11:52	0.0	NaN	United Kingdom
1970	536545	21134	NaN	1	12/1/2010 14:32	0.0	NaN	United Kingdom
1971	536546	22145	NaN	1	12/1/2010 14:33	0.0	NaN	United Kingdom
1972	536547	37509	NaN	1	12/1/2010 14:33	0.0	NaN	United Kingdom
1987	536549	85226A	NaN	1	12/1/2010 14:34	0.0	NaN	United Kingdom
...	...	...	...	...	...	...	...	...
535322	581199	84581	NaN	-2	12/7/2011 18:26	0.0	NaN	United Kingdom
535326	581203	23406	NaN	15	12/7/2011 18:31	0.0	NaN	United Kingdom
535332	581209	21620	NaN	6	12/7/2011 18:35	0.0	NaN	United Kingdom
536981	581234	72817	NaN	27	12/8/2011 10:33	0.0	NaN	United Kingdom
538554	581408	85175	NaN	20	12/8/2011 14:06	0.0	NaN	United Kingdom

1454 rows × 8 columns

In [11]:
```
# How many unique stockcode codes have no description
df[df['Description'].isna()].StockCode.nunique()
```

Out[11]:
```
960
```

In [12]:
```
# what countries have sales with no description
df[df['Description'].isna()].Country.value_counts()
```

Out[12]:
```
Country
United Kingdom    1454
Name: count, dtype: int64
```

In [13]:
```
# what countries have sales with no customerID and how many records are affected?
df[df['CustomerID'].isna()].Country.value_counts()
```

Out[13]:
```
Country
United Kingdom    133600
EIRE                 711
Hong Kong            288
Unspecified          202
Switzerland          125
France                66
Israel                47
Portugal              39
Bahrain                2
Name: count, dtype: int64
```

In [14]:
```
'''Assuming you do not keep records that have no description, you can choose to delete

# check the number of records that will be affected # notna()

print('Total number of records: ', df.shape[0])
print('Number of records with missing description: ', df[df['Description'].isna()].sha
print('Number of records without missing description: ', df[df['Description'].notna()]
```

```
Total number of records:  541909
Number of records with missing description:  1454
Number of records without missing description:  540455
```

In [17]:
```
# numbers of rows with description
num_missing = df[df['Description'].isna()].shape[0]

# numbers of rows in the dataset
num_all = df.shape[0]

# percentage of rows with description
round((num_missing / num_all) * 100, 2)
```

Out[17]:
```
0.27
```

In [20]:
```
# we check records that are not NaN - notna()
df = df[df['Description'].notna()].copy()
df
```

Out[20]:

	InvoiceNo	StockCode	Description	Quantity	InvoiceDate	UnitPrice	CustomerID	Country
0	536365	85123A	WHITE HANGING HEART T-LIGHT HOLDER	6	12/1/2010 08:26	2.55	17850.0	United Kingdom
1	536365	71053	WHITE METAL LANTERN	6	12/1/2010 08:26	3.39	17850.0	United Kingdom
2	536365	84406B	CREAM CUPID HEARTS COAT HANGER	8	12/1/2010 08:26	2.75	17850.0	United Kingdom
3	536365	84029G	KNITTED UNION FLAG HOT WATER BOTTLE	6	12/1/2010 08:26	3.39	17850.0	United Kingdom
4	536365	84029E	RED WOOLLY HOTTIE WHITE HEART.	6	12/1/2010 08:26	3.39	17850.0	United Kingdom
...	...	...	...	...	...	...	...	...
541904	581587	22613	PACK OF 20 SPACEBOY NAPKINS	12	12/9/2011 12:50	0.85	12680.0	France
541905	581587	22899	CHILDREN'S APRON DOLLY GIRL	6	12/9/2011 12:50	2.10	12680.0	France
541906	581587	23254	CHILDRENS CUTLERY DOLLY GIRL	4	12/9/2011 12:50	4.15	12680.0	France
541907	581587	23255	CHILDRENS CUTLERY CIRCUS PARADE	4	12/9/2011 12:50	4.15	12680.0	France
541908	581587	22138	BAKING SET 9 PIECE RETROSPOT	3	12/9/2011 12:50	4.95	12680.0	France

540455 rows × 8 columns

In [21]:
```
# what are the rows with NaN values?
df.isna().sum()
```

Out[21]:
```
InvoiceNo           0
StockCode           0
Description         0
Quantity            0
InvoiceDate         0
UnitPrice           0
CustomerID     133626
Country             0
dtype: int64
```

Other Cleaning activities we can do...

In [22]:
```
# We will replace values e.g EIRE or RSA with a more recognizable name
df['Country'].unique()
```

Out[22]:
```
array(['United Kingdom', 'France', 'Australia', 'Netherlands', 'Germany',
       'Norway', 'EIRE', 'Switzerland', 'Spain', 'Poland', 'Portugal',
       'Italy', 'Belgium', 'Lithuania', 'Japan', 'Iceland',
       'Channel Islands', 'Denmark', 'Cyprus', 'Sweden', 'Austria',
       'Israel', 'Finland', 'Bahrain', 'Greece', 'Hong Kong', 'Singapore',
       'Lebanon', 'United Arab Emirates', 'Saudi Arabia',
       'Czech Republic', 'Canada', 'Unspecified', 'Brazil', 'USA',
       'European Community', 'Malta', 'RSA'], dtype=object)
```

In [23]:
```
# find records where country is EIRE
df[df['Country'] == 'EIRE']
```

Out[23]:

	InvoiceNo	StockCode	Description	Quantity	InvoiceDate	UnitPrice	Customer
1404	536540	22968	ROSE COTTAGE KEEPSAKE BOX	4	12/1/2010 14:05	9.95	14911
1405	536540	85071A	BLUE CHARLIE+LOLA PERSONAL DOORSIGN	6	12/1/2010 14:05	2.95	14911
1406	536540	85071C	CHARLIE+LOLA"EXTREMELY BUSY" SIGN	6	12/1/2010 14:05	2.55	14911
1407	536540	22355	CHARLOTTE BAG SUKI DESIGN	50	12/1/2010 14:05	0.85	14911
1408	536540	21579	LOLITA DESIGN COTTON TOTE BAG	6	12/1/2010 14:05	2.25	14911
...	...	...	...	...	...	...	
539151	581433	22192	BLUE DINER WALL CLOCK	2	12/8/2011 15:54	8.50	14911
539152	581433	48187	DOORMAT NEW ENGLAND	2	12/8/2011 15:54	8.25	14911
539153	581433	48184	DOORMAT ENGLISH ROSE	2	12/8/2011 15:54	8.25	14911
539154	581433	20685	DOORMAT RED RETROSPOT	2	12/8/2011 15:54	8.25	14911
539155	581433	79302M	ART LIGHTS,FUNK MONKEY	6	12/8/2011 15:54	2.95	14911

8196 rows × 8 columns

In [24]:
```
df[df['Country'] == 'EIRE'].Country
```

Out[24]:
```
1404      EIRE
1405      EIRE
1406      EIRE
1407      EIRE
1408      EIRE
          ...
539151    EIRE
539152    EIRE
539153    EIRE
539154    EIRE
539155    EIRE
Name: Country, Length: 8196, dtype: object
```

In [25]:
```
# using the replace operation, we will replace 'EIRE' with 'Ireland' # .replace()
df[df['Country'] == 'EIRE'].Country.replace('EIRE', 'Ireland')
```

Out[25]:
```
1404      Ireland
1405      Ireland
1406      Ireland
1407      Ireland
1408      Ireland
           ...
539151    Ireland
539152    Ireland
539153    Ireland
539154    Ireland
539155    Ireland
Name: Country, Length: 8196, dtype: object
```

In [26]:
```
# apply the replace operation
df['Country'] = df['Country'].replace('EIRE', 'Ireland')
df['Country']
```

Out[26]:
```
0         United Kingdom
1         United Kingdom
2         United Kingdom
3         United Kingdom
4         United Kingdom
               ...
541904           France
541905           France
541906           France
541907           France
541908           France
Name: Country, Length: 540455, dtype: object
```

loc

- it is a label based indexing, which means that we can specify rows and columns based on their row and column label ### iloc
- it is an integer based indexing, which means we can specify rows and columns by their integer.

In [27]:
```
df.loc[:, 'Country'] = df['Country'].replace('EIRE', 'Ireland')
```

In [28]:
```
# we now have rows with 'Ireland', we can check what have done
df[df['Country'] == 'Ireland'].head()
```

Out[28]:

	InvoiceNo	StockCode	Description	Quantity	InvoiceDate	UnitPrice	CustomerID
1404	536540	22968	ROSE COTTAGE KEEPSAKE BOX	4	12/1/2010 14:05	9.95	14911.0
1405	536540	85071A	BLUE CHARLIE+LOLA PERSONAL DOORSIGN	6	12/1/2010 14:05	2.95	14911.0
1406	536540	85071C	CHARLIE+LOLA"EXTREMELY BUSY" SIGN	6	12/1/2010 14:05	2.55	14911.0
1407	536540	22355	CHARLOTTE BAG SUKI DESIGN	50	12/1/2010 14:05	0.85	14911.0
1408	536540	21579	LOLITA DESIGN COTTON TOTE BAG	6	12/1/2010 14:05	2.25	14911.0

In [29]:
```
df[df['Country'] == 'EIRE'].head()
```

Out[29]:

	InvoiceNo	StockCode	Description	Quantity	InvoiceDate	UnitPrice	CustomerID	Country

We do not have a country EIRE

Replacing missing CustomerIDs

In [30]:
```
df['CustomerID'].value_counts()
```

Out[30]:
```
CustomerID
17841.0    7983
14911.0    5903
14096.0    5128
12748.0    4642
14606.0    2782
           ...
15070.0       1
15753.0       1
17065.0       1
16881.0       1
16995.0       1
Name: count, Length: 4372, dtype: int64
```

In [31]:
```
df['CustomerID'].dtype
```

Out[31]:
```
dtype('float64')
```

In [32]:
```
# we first convert to int64 to drop the decimals before converting to string
df['CustomerID'].astype('Int64').astype(str)
```

Out[32]:
```
0         17850
1         17850
2         17850
3         17850
4         17850
          ...
541904    12680
541905    12680
541906    12680
541907    12680
541908    12680
Name: CustomerID, Length: 540455, dtype: object
```

In [33]:
```
df['CustomerID'] = df['CustomerID'].astype('Int64').astype(str)
```

In [34]:
```
df.dtypes
```

Out[34]:
```
InvoiceNo       object
StockCode       object
Description     object
Quantity         int64
InvoiceDate     object
UnitPrice      float64
CustomerID      object
Country         object
dtype: object
```

In [35]:
```
# checking missing values
df.isna().sum()
```

Out[35]:
```
InvoiceNo      0
StockCode      0
Description    0
Quantity       0
InvoiceDate    0
UnitPrice      0
CustomerID     0
Country        0
dtype: int64
```

In [36]:
```
# use vaue_counts() to group CustomerID
'''it will show us that the null values has been converted to an integer'''
df['CustomerID'].value_counts()
```

Out[36]:
```
CustomerID
<NA>     133626
17841      7983
14911      5903
14096      5128
12748      4642
           ...
13270         1
17763         1
17291         1
15668         1
15562         1
Name: count, Length: 4373, dtype: int64
```

In [37]:
```
# we found out that NaN was now represented by <NA> after it was converted
df[df['CustomerID'] == '<NA>']
```

Out[37]:

	InvoiceNo	StockCode	Description	Quantity	InvoiceDate	UnitPrice	CustomerID	Country
1443	536544	21773	DECORATIVE ROSE BATHROOM BOTTLE	1	12/1/2010 14:32	2.51	<NA>	United Kingdom
1444	536544	21774	DECORATIVE CATS BATHROOM BOTTLE	2	12/1/2010 14:32	2.51	<NA>	United Kingdom
1445	536544	21786	POLKADOT RAIN HAT	4	12/1/2010 14:32	0.85	<NA>	United Kingdom
1446	536544	21787	RAIN PONCHO RETROSPOT	2	12/1/2010 14:32	1.66	<NA>	United Kingdom
1447	536544	21790	VINTAGE SNAP CARDS	9	12/1/2010 14:32	1.66	<NA>	United Kingdom
...	...	...	...	...	...	...	...	...
541536	581498	85099B	JUMBO BAG RED RETROSPOT	5	12/9/2011 10:26	4.13	<NA>	United Kingdom
541537	581498	85099C	JUMBO BAG BAROQUE BLACK WHITE	4	12/9/2011 10:26	4.13	<NA>	United Kingdom
541538	581498	85150	LADIES & GENTLEMEN METAL SIGN	1	12/9/2011 10:26	4.96	<NA>	United Kingdom
541539	581498	85174	S/4 CACTI CANDLES	1	12/9/2011 10:26	10.79	<NA>	United Kingdom
541540	581498	DOT	DOTCOM POSTAGE	1	12/9/2011 10:26	1714.17	<NA>	United Kingdom

133626 rows × 8 columns

In [38]:
```
# we will replace <NA> with 'Unidentified'
# using the parameter 'inplace' applies the operation
df['CustomerID'].replace('<NA>', 'Unidentified', inplace = True)
```

In [39]:
```
# do we still have <NA>?
df[df['CustomerID'] == '<NA>']
```

Out[39]:

	InvoiceNo	StockCode	Description	Quantity	InvoiceDate	UnitPrice	CustomerID	Country

In [40]:
```
# do we still have 'Unidentified'
df[df['CustomerID'] == 'Unidentified']
```

Out[40]:

	InvoiceNo	StockCode	Description	Quantity	InvoiceDate	UnitPrice	CustomerID	Country
1443	536544	21773	DECORATIVE ROSE BATHROOM BOTTLE	1	12/1/2010 14:32	2.51	Unidentified	United Kingdom
1444	536544	21774	DECORATIVE CATS BATHROOM BOTTLE	2	12/1/2010 14:32	2.51	Unidentified	United Kingdom
1445	536544	21786	POLKADOT RAIN HAT	4	12/1/2010 14:32	0.85	Unidentified	United Kingdom
1446	536544	21787	RAIN PONCHO RETROSPOT	2	12/1/2010 14:32	1.66	Unidentified	United Kingdom
1447	536544	21790	VINTAGE SNAP CARDS	9	12/1/2010 14:32	1.66	Unidentified	United Kingdom
...	...	...	...	...	...	...	...	...
541536	581498	85099B	JUMBO BAG RED RETROSPOT	5	12/9/2011 10:26	4.13	Unidentified	United Kingdom
541537	581498	85099C	JUMBO BAG BAROQUE BLACK WHITE	4	12/9/2011 10:26	4.13	Unidentified	United Kingdom
541538	581498	85150	LADIES & GENTLEMEN METAL SIGN	1	12/9/2011 10:26	4.96	Unidentified	United Kingdom
541539	581498	85174	S/4 CACTI CANDLES	1	12/9/2011 10:26	10.79	Unidentified	United Kingdom
541540	581498	DOT	DOTCOM POSTAGE	1	12/9/2011 10:26	1714.17	Unidentified	United Kingdom

133626 rows × 8 columns

In [41]:
```
df['CustomerID'].value_counts()
```

Out[41]:
```
CustomerID
Unidentified    133626
17841             7983
14911             5903
14096             5128
12748             4642
                  ...
13270                1
17763                1
17291                1
15668                1
15562                1
Name: count, Length: 4373, dtype: int64
```

In [42]:
```
df.isna().sum()
```

Out[42]:
```
InvoiceNo      0
StockCode      0
Description    0
Quantity       0
InvoiceDate    0
UnitPrice      0
CustomerID     0
Country        0
dtype: int64
```

In [44]:
```
# percentage of the CustomerID that are Unidentified
round((df[df['CustomerID'] == 'Unidentified'].shape[0] / df.shape[0]) * 100, 1)
```

Out[44]:
```
24.7
```

In [45]:
```
df.head()
```

Out[45]:

	InvoiceNo	StockCode	Description	Quantity	InvoiceDate	UnitPrice	CustomerID	Country
0	536365	85123A	WHITE HANGING HEART T-LIGHT HOLDER	6	12/1/2010 08:26	2.55	17850	United Kingdom
1	536365	71053	WHITE METAL LANTERN	6	12/1/2010 08:26	3.39	17850	United Kingdom
2	536365	84406B	CREAM CUPID HEARTS COAT HANGER	8	12/1/2010 08:26	2.75	17850	United Kingdom
3	536365	84029G	KNITTED UNION FLAG HOT WATER BOTTLE	6	12/1/2010 08:26	3.39	17850	United Kingdom
4	536365	84029E	RED WOOLLY HOTTIE WHITE HEART.	6	12/1/2010 08:26	3.39	17850	United Kingdom