In [1]:
```
# Import the Packages.

import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline

import sqlite3 as db
from pandasql import sqldf

pysqldf = lambda q: sqldf(q, globals())
```

In [2]:
```
# Read the file from a Git Url
df = pd.read_csv('https://raw.githubusercontent.com/jackiekazil/data-wrangling/ma
df.head(2)
```

Out[2]:

	Indicator	PUBLISH STATES	Year	WHO region	World Bank income group	Country	Sex	Display Value	Numeric	Low	High	Co
0	Life expectancy at birth (years)	Published	1990	Europe	High-income	Andorra	Both sexes	77	77.0	NaN	NaN	
1	Life expectancy at birth (years)	Published	2000	Europe	High-income	Andorra	Both sexes	80	80.0	NaN	NaN	

In [3]:
```
# Read the file from a Git Url
df2 = pd.read_csv('https://raw.githubusercontent.com/kjam/data-wrangling-pycon/ma
df2.head(2)
```

Out[3]:

	STATION	STATION_NAME	DATE	PRCP	SNWD	SNOW	TMAX	TMIN	WDFG	F
0	GHCND:GME00111445	BERLIN TEMPELHOF GM	19310101	46	-9999	-9999	-9999	-11	-9999	
1	GHCND:GME00111445	BERLIN TEMPELHOF GM	19310102	107	-9999	-9999	50	11	-9999	

2 rows × 21 columns

In [4]:
```
# 1. Get the Metadata for first Data frame 1
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 4656 entries, 0 to 4655
Data columns (total 12 columns):
Indicator                  4656 non-null object
PUBLISH STATES             4656 non-null object
Year                       4656 non-null int64
WHO region                 4656 non-null object
World Bank income group    4656 non-null object
Country                    4656 non-null object
Sex                        4656 non-null object
Display Value              4656 non-null int64
Numeric                    4656 non-null float64
Low                        0 non-null float64
High                       0 non-null float64
Comments                   0 non-null float64
dtypes: float64(4), int64(2), object(6)
memory usage: 436.6+ KB
```

In [5]:
```
# 1. Get the Metadata for Second Data frame 1
df2.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 117208 entries, 0 to 117207
Data columns (total 21 columns):
STATION         117208 non-null object
STATION_NAME    117208 non-null object
DATE            117208 non-null int64
PRCP            117208 non-null int64
SNWD            117208 non-null int64
SNOW            117208 non-null int64
TMAX            117208 non-null int64
TMIN            117208 non-null int64
WDFG            117208 non-null int64
PGTM            117208 non-null int64
WSFG            117208 non-null int64
WT09            117208 non-null int64
WT07            117208 non-null int64
WT01            117208 non-null int64
WT06            117208 non-null int64
WT05            117208 non-null int64
```

In [6]:
```
# 2. Get the row names from the above files.
dfRowNames = np.where(df["Indicator"].isnull() != True)
dfRowNames
```

Out[6]:
```
(array([   0,    1,    2, ..., 4653, 4654, 4655], dtype=int64),)
```

In [7]:
```
# 2. Get the row names from the above files.
dfRowNames = np.where(df2["STATION"].isnull() != True)
dfRowNames
```

Out[7]:
```
(array([     0,      1,      2, ..., 117205, 117206, 117207], dtype=int64),)
```

In [8]:
```
# 3 & 4. Change the column name from any of the above file.
df.rename(columns={'Indicator':'Indicator_id'}, inplace = True)
df.head(1)
```

Out[8]:

	Indicator_id	PUBLISH STATES	Year	WHO region	World Bank income group	Country	Sex	Display Value	Numeric	Low	High	C
0	Life expectancy at birth (years)	Published	1990	Europe	High-income	Andorra	Both sexes	77	77.0	NaN	NaN	

In [9]:
```
# 5.Change the column name from any of the above file and store the changes made

df.rename(columns = {'PUBLISH STATES':'Publication Status' , 'WHO region' : 'WHO
df.head(1)
```

Out[9]:

	Indicator_id	Publication Status	Year	WHO Region	World Bank income group	Country	Sex	Display Value	Numeric	Low	High
0	Life expectancy at birth (years)	Published	1990	Europe	High-income	Andorra	Both sexes	77	77.0	NaN	NaN

In [10]:
```
#6. Arrange values of a particular column in ascending order
df.sort_values(['Year'], ascending=[True])
```

Out[10]:

	Indicator_id	Publication Status	Year	WHO Region	World Bank income group	Country	Sex	Display Value	Numer
0	Life expectancy at birth (years)	Published	1990	Europe	High-income	Andorra	Both sexes	77	77
1270	Life expectancy at birth (years)	Published	1990	Europe	High-income	Germany	Male	72	72
3193	Life expectancy at birth (years)	Published	1990	Europe	Lower-middle-income	Republic of Moldova	Male	65	65
3194	Life expectancy at birth	Published	1990	Europe	Lower-middle-	Republic of	Both	68	68

In [11]:
```
#7. Arrange multiple column values in ascending order
dfcopy = df[['Indicator_id', 'Country','Year','WHO Region', 'Publication Status']]
dfcopy
```

Out[11]:

	Indicator_id	Country	Year	WHO Region	Publication Status
554	Life expectancy at birth (years)	Afghanistan	1990	Eastern Mediterranean	Published
965	Life expectancy at birth (years)	Afghanistan	1990	Eastern Mediterranean	Published
1792	Life expectancy at birth (years)	Afghanistan	1990	Eastern Mediterranean	Published
146	Life expectancy at birth (years)	Afghanistan	2000	Eastern Mediterranean	Published
1393	Life expectancy at birth (years)	Afghanistan	2000	Eastern Mediterranean	Published
2957	Life expectancy at birth (years)	Afghanistan	2000	Eastern Mediterranean	Published
966	Life expectancy at birth (years)	Afghanistan	2012	Eastern Mediterranean	Published
1394	Life expectancy at birth (years)	Afghanistan	2012	Eastern Mediterranean	Published
2958	Life expectancy at birth (years)	Afghanistan	2012	Eastern Mediterranean	Published
299	Life expectancy at birth (years)	Albania	1990	Europe	Published
689	Life expectancy at birth (years)	Albania	1990	Europe	Published
3113	Life expectancy at birth (years)	Albania	1990	Europe	Published
1087	Life expectancy at birth (years)	Albania	2000	Europe	Published
1520	Life expectancy at birth (years)	Albania	2000	Europe	Published
1929	Life expectancy at birth (years)	Albania	2000	Europe	Published
300	Life expectancy at birth (years)	Albania	2012	Europe	Published
688	Life expectancy at birth (years)	Albania	2012	Europe	Published
3112	Life expectancy at birth (years)	Albania	2012	Europe	Published
2145	Life expectancy at birth (years)	Algeria	1990	Africa	Published
2510	Life expectancy at birth (years)	Algeria	1990	Africa	Published
4358	Life expectancy at birth (years)	Algeria	1990	Africa	Published
2146	Life expectancy at birth (years)	Algeria	2000	Africa	Published
3968	Life expectancy at birth (years)	Algeria	2000	Africa	Published
3969	Life expectancy at birth (years)	Algeria	2000	Africa	Published
3583	Life expectancy at birth (years)	Algeria	2012	Africa	Published
3584	Life expectancy at birth (years)	Algeria	2012	Africa	Published
4357	Life expectancy at birth (years)	Algeria	2012	Africa	Published

	Indicator_id	Country	Year	WHO Region	Publication Status
0	Life expectancy at birth (years)	Andorra	1990	Europe	Published
818	Life expectancy at birth (years)	Andorra	1990	Europe	Published
2799	Life expectancy at birth (years)	Andorra	1990	Europe	Published
...	...	...	...	...	...
2417	Healthy life expectancy (HALE) at birth (years)	Venezuela (Bolivarian Republic of)	2000	Americas	Published
2418	Healthy life expectancy (HALE) at birth (years)	Venezuela (Bolivarian Republic of)	2000	Americas	Published
2790	Healthy life expectancy (HALE) at birth (years)	Venezuela (Bolivarian Republic of)	2000	Americas	Published
2419	Healthy life expectancy (HALE) at birth (years)	Venezuela (Bolivarian Republic of)	2012	Americas	Published
4255	Healthy life expectancy (HALE) at birth (years)	Venezuela (Bolivarian Republic of)	2012	Americas	Published
4648	Healthy life expectancy (HALE) at birth (years)	Venezuela (Bolivarian Republic of)	2012	Americas	Published
2791	Healthy life expectancy (HALE) at birth (years)	Viet Nam	2000	Western Pacific	Published
3539	Healthy life expectancy (HALE) at birth (years)	Viet Nam	2000	Western Pacific	Published
4256	Healthy life expectancy (HALE) at birth (years)	Viet Nam	2000	Western Pacific	Published
2420	Healthy life expectancy (HALE) at birth (years)	Viet Nam	2012	Western Pacific	Published
2792	Healthy life expectancy (HALE) at birth (years)	Viet Nam	2012	Western Pacific	Published
3880	Healthy life expectancy (HALE) at birth (years)	Viet Nam	2012	Western Pacific	Published
2793	Healthy life expectancy (HALE) at birth (years)	Yemen	2000	Eastern Mediterranean	Published
2794	Healthy life expectancy (HALE) at birth (years)	Yemen	2000	Eastern Mediterranean	Published
3882	Healthy life expectancy (HALE) at birth (years)	Yemen	2000	Eastern Mediterranean	Published
2424	Healthy life expectancy (HALE) at birth (years)	Yemen	2012	Eastern Mediterranean	Published
3883	Healthy life expectancy (HALE) at birth (years)	Yemen	2012	Eastern Mediterranean	Published
4652	Healthy life expectancy (HALE) at birth (years)	Yemen	2012	Eastern Mediterranean	Published
3544	Healthy life expectancy (HALE) at birth (years)	Zambia	2000	Africa	Published
3885	Healthy life expectancy (HALE) at birth (years)	Zambia	2000	Africa	Published

	Indicator_id	Country	Year	WHO Region	Publication Status
4654	Healthy life expectancy (HALE) at birth (years)	Zambia	2000	Africa	Published
2796	Healthy life expectancy (HALE) at birth (years)	Zambia	2012	Africa	Published
3545	Healthy life expectancy (HALE) at birth (years)	Zambia	2012	Africa	Published
4260	Healthy life expectancy (HALE) at birth (years)	Zambia	2012	Africa	Published
2426	Healthy life expectancy (HALE) at birth (years)	Zimbabwe	2000	Africa	Published
2797	Healthy life expectancy (HALE) at birth (years)	Zimbabwe	2000	Africa	Published
3886	Healthy life expectancy (HALE) at birth (years)	Zimbabwe	2000	Africa	Published
3546	Healthy life expectancy (HALE) at birth (years)	Zimbabwe	2012	Africa	Published
4261	Healthy life expectancy (HALE) at birth (years)	Zimbabwe	2012	Africa	Published
4655	Healthy life expectancy (HALE) at birth (years)	Zimbabwe	2012	Africa	Published

4656 rows × 5 columns

In [12]:
```
# 8. Make country as the first column of the dataframe
df = df[['Country', 'Indicator_id', 'Publication Status', 'Year', 'WHO Region', '
df.head(2)
```

Out[12]:

	Country	Indicator_id	Publication Status	Year	WHO Region	World Bank income group	Sex	Display Value	Numeric	Low	High
0	Andorra	Life expectancy at birth (years)	Published	1990	Europe	High-income	Both sexes	77	77.0	NaN	NaN
1	Andorra	Life expectancy at birth (years)	Published	2000	Europe	High-income	Both sexes	80	80.0	NaN	NaN

In [13]:
```
#9. Get the column array using a variable
dfcolumns = df.loc[df['WHO Region'].isnull() != True,'WHO Region' ].values
dfcolumns
```

Out[13]:
```
array(['Europe', 'Europe', 'Europe', ..., 'Africa', 'Africa', 'Africa'],
      dtype=object)
```

In [14]:
```
#10.  Get the subset rows 11,24,37
df.loc[[11, 24,37]]
```

Out[14]:

	Country	Indicator_id	Publication Status	Year	WHO Region	World Bank income group	Sex	Display Value	Numeric	Low
11	Austria	Life expectancy at birth (years)	Published	2012	Europe	High-income	Female	83	83.0	NaN
24	Brunei Darussalam	Life expectancy at age 60 (years)	Published	2012	Western Pacific	High-income	Female	21	21.0	NaN
37	Cyprus	Life expectancy at age 60 (years)	Published	2012	Europe	High-income	Female	26	26.0	NaN

In [15]:
```
#11. Get the subset rows excluding 5, 12, 23, and 56
df.drop(df.index[[5,12,23,56]])
```

Out[15]:

	Country	Indicator_id	Publication Status	Year	WHO Region	World Bank income group	Sex	Display Value	Numer
0	Andorra	Life expectancy at birth (years)	Published	1990	Europe	High-income	Both sexes	77	77
1	Andorra	Life expectancy at birth (years)	Published	2000	Europe	High-income	Both sexes	80	80
2	Andorra	Life expectancy at age 60 (years)	Published	2012	Europe	High-income	Female	28	28
3	Andorra	Life expectancy at age 60	Published	2000	Europe	High-income	Both sexes	23	23

In [16]:
```
df
```

Out[16]:

	Country	Indicator_id	Publication Status	Year	WHO Region	World Bank income group	Sex	Display Value	Numer
0	Andorra	Life expectancy at birth (years)	Published	1990	Europe	High-income	Both sexes	77	77
1	Andorra	Life expectancy at birth (years)	Published	2000	Europe	High-income	Both sexes	80	80
2	Andorra	Life expectancy at age 60 (years)	Published	2012	Europe	High-income	Female	28	28
3	Andorra	Life expectancy	Published	2000	Europe	High-	Both	23	23

In [17]:
```
# Load datasets from CSV
users = pd.read_csv('https://raw.githubusercontent.com/ben519/DataWrangling/maste
```

In [18]:
```
sessions = pd.read_csv('https://raw.githubusercontent.com/ben519/DataWrangling/ma
```

In [19]:
```
products = pd.read_csv('https://raw.githubusercontent.com/ben519/DataWrangling/ma
```

In [20]:
```
transactions = pd.read_csv('https://raw.githubusercontent.com/ben519/DataWranglin
```

In [21]:
```
users.head(2)
```

Out[21]:

	UserID	User	Gender	Registered	Cancelled
0	1	Charles	male	2012-12-21	NaN
1	2	Pedro	male	2010-08-01	2010-08-08

In [22]:
```
sessions.head(2)
```

Out[22]:

	SessionID	SessionDate	UserID
0	1	2010-01-05	2
1	2	2010-08-01	2

In [23]:
```
products.head(2)
```

Out[23]:

	ProductID	Product	Price
0	1	A	14.16
1	2	B	33.04

In [24]:
```
transactions.head(2)
```

Out[24]:

	TransactionID	TransactionDate	UserID	ProductID	Quantity
0	1	2010-08-21	7.0	2	1
1	2	2011-05-26	3.0	4	1

In [25]:
```
#12. Join users to transactions, keeping all rows from transactions and only matc
q = """ SELECT t.*, u.user, u.gender, u.Registered, u.Cancelled FROM transactions
```

In [26]:
```
result = pysqldf(q)
result
```

Out[26]:

	TransactionID	TransactionDate	UserID	ProductID	Quantity	User	Gender	Registered	Canc
0	1	2010-08-21	7.0	2	1	None	None	None	
1	2	2011-05-26	3.0	4	1	Caroline	female	2012-10-23	201
2	3	2011-06-16	3.0	3	1	Caroline	female	2012-10-23	201
3	4	2012-08-26	1.0	2	3	Charles	male	2012-12-21	
4	5	2013-06-06	2.0	4	1	Pedro	male	2010-08-01	201
5	6	2013-12-23	2.0	5	6	Pedro	male	2010-08-01	201
6	7	2013-12-30	3.0	4	1	Caroline	female	2012-10-23	201
7	8	2014-04-24	NaN	2	3	None	None	None	
8	9	2015-04-24	7.0	4	3	None	None	None	
9	10	2016-05-08	3.0	4	4	Caroline	female	2012-10-23	201

In [27]:
```
#13. Which transactions have a UserID not in users?
q = """ SELECT t.* FROM transactions t LEFT JOIN users u on t.UserID = u.UserID WH
```

In [28]:
```
result = pysqldf(q)
result
```

Out[28]:

	TransactionID	TransactionDate	UserID	ProductID	Quantity
0	1	2010-08-21	7.0	2	1
1	8	2014-04-24	NaN	2	3
2	9	2015-04-24	7.0	4	3

In [29]:
```
#14 Join users to transactions, keeping only rows from transactions and users tha
pd.merge(transactions, users, on='UserID', how='inner')
```

Out[29]:

	TransactionID	TransactionDate	UserID	ProductID	Quantity	User	Gender	Registered	Canc
0	2	2011-05-26	3.0	4	1	Caroline	female	2012-10-23	201
1	3	2011-06-16	3.0	3	1	Caroline	female	2012-10-23	201
2	7	2013-12-30	3.0	4	1	Caroline	female	2012-10-23	201
3	10	2016-05-08	3.0	4	4	Caroline	female	2012-10-23	201
4	4	2012-08-26	1.0	2	3	Charles	male	2012-12-21	
5	5	2013-06-06	2.0	4	1	Pedro	male	2010-08-01	201
6	6	2013-12-23	2.0	5	6	Pedro	male	2010-08-01	201

In [30]:
```
#15. Join users to transactions, displaying all matching rows AND all non-matchin
pd.merge(transactions, users, on='UserID', how='outer')
```

Out[30]:

	TransactionID	TransactionDate	UserID	ProductID	Quantity	User	Gender	Registered	Ca
0	1.0	2010-08-21	7.0	2.0	1.0	NaN	NaN	NaN	
1	9.0	2015-04-24	7.0	4.0	3.0	NaN	NaN	NaN	
2	2.0	2011-05-26	3.0	4.0	1.0	Caroline	female	2012-10-23	2
3	3.0	2011-06-16	3.0	3.0	1.0	Caroline	female	2012-10-23	2
4	7.0	2013-12-30	3.0	4.0	1.0	Caroline	female	2012-10-23	2
5	10.0	2016-05-08	3.0	4.0	4.0	Caroline	female	2012-10-23	2
6	4.0	2012-08-26	1.0	2.0	3.0	Charles	male	2012-12-21	
7	5.0	2013-06-06	2.0	4.0	1.0	Pedro	male	2010-08-01	2
8	6.0	2013-12-23	2.0	5.0	6.0	Pedro	male	2010-08-01	2
9	8.0	2014-04-24	NaN	2.0	3.0	NaN	NaN	NaN	
10	NaN	NaN	4.0	NaN	NaN	Brielle	female	2013-07-17	
11	NaN	NaN	5.0	NaN	NaN	Benjamin	male	2010-11-25	

In [31]:
```
#16. Determine which sessions occurred on the same day each user registered
pd.merge(users, sessions, left_on=['Registered','UserID'], right_on = ['SessionDa
```

Out[31]:

	UserID	User	Gender	Registered	Cancelled	SessionID	SessionDate
0	2	Pedro	male	2010-08-01	2010-08-08	2	2010-08-01
1	4	Brielle	female	2013-07-17	NaN	9	2013-07-17

In [32]:
```
#17. Build a dataset with every possible (UserID, ProductID) pair (cross join)
df1 = pd.DataFrame({'key': np.repeat(1, users.shape[0]), 'UserID': users.UserID})
df2 = pd.DataFrame({'key': np.repeat(1, products.shape[0]), 'ProductID': products
pd.merge(df1, df2, on='key')[['UserID', 'ProductID']]
```

Out[32]:

	UserID	ProductID
0	1	1
1	1	2
2	1	3
3	1	4
4	1	5
5	2	1
6	2	2
7	2	3
8	2	4
9	2	5
10	3	1
11	3	2

In [33]:
```
#18. Determine how much quantity of each product was purchased by each usedf1 = p
df2 = pd.DataFrame({'key': np.repeat(1, products.shape[0]), 'ProductID': products

user_products = pd.merge(df1, df2,on='key')[['UserID', 'ProductID']]

pd.merge(user_products, transactions, how='left', on=['UserID', 'ProductID']).gro
                    lambda x: pd.Series(dict(Quantity = x.Quantity.sum() )) ).re
```

Out[33]:

	UserID	ProductID	Quantity
0	1	1	0.0
1	1	2	3.0
2	1	3	0.0
3	1	4	0.0
4	1	5	0.0
5	2	1	0.0
6	2	2	0.0
7	2	3	0.0
8	2	4	1.0
9	2	5	6.0
10	3	1	0.0
11	3	2	0.0
12	3	3	1.0
13	3	4	6.0
14	3	5	0.0
15	4	1	0.0
16	4	2	0.0
17	4	3	0.0
18	4	4	0.0
19	4	5	0.0
20	5	1	0.0
21	5	2	0.0
22	5	3	0.0
23	5	4	0.0
24	5	5	0.0

In [34]:
```
#19. For each user, get each possible pair of pair transactions (TransactionID1,
pd.merge(transactions, transactions, on='UserID')
```

Out[34]:

	TransactionID_x	TransactionDate_x	UserID	ProductID_x	Quantity_x	TransactionID_y	Transact
0	1	2010-08-21	7.0	2	1	1	2
1	1	2010-08-21	7.0	2	1	9	2
2	9	2015-04-24	7.0	4	3	1	2
3	9	2015-04-24	7.0	4	3	9	2
4	2	2011-05-26	3.0	4	1	2	2
5	2	2011-05-26	3.0	4	1	3	2
6	2	2011-05-26	3.0	4	1	7	2
7	2	2011-05-26	3.0	4	1	10	2
8	3	2011-06-16	3.0	3	1	2	2
9	3	2011-06-16	3.0	3	1	3	2
10	3	2011-06-16	3.0	3	1	7	2
11	3	2011-06-16	3.0	3	1	10	2
12	7	2013-12-30	3.0	4	1	2	2
13	7	2013-12-30	3.0	4	1	3	2
14	7	2013-12-30	3.0	4	1	7	2
15	7	2013-12-30	3.0	4	1	10	2
16	10	2016-05-08	3.0	4	4	2	2
17	10	2016-05-08	3.0	4	4	3	2
18	10	2016-05-08	3.0	4	4	7	2
19	10	2016-05-08	3.0	4	4	10	2
20	4	2012-08-26	1.0	2	3	4	2
21	5	2013-06-06	2.0	4	1	5	2
22	5	2013-06-06	2.0	4	1	6	2
23	6	2013-12-23	2.0	5	6	5	2
24	6	2013-12-23	2.0	5	6	6	2
25	8	2014-04-24	NaN	2	3	8	2

In [35]:
```
#20. Join each user to his/her first occuring transaction in the transactions tab
first_occuring_transactions = transactions.groupby('UserID').first().reset_index(

pd.merge(users,first_occuring_transactions, how='left', on='UserID')
```

Out[35]:

	UserID	User	Gender	Registered	Cancelled	TransactionID	TransactionDate	ProductID	Qu
0	1	Charles	male	2012-12-21	NaN	4.0	2012-08-26	2.0	
1	2	Pedro	male	2010-08-01	2010-08-08	5.0	2013-06-06	4.0	
2	3	Caroline	female	2012-10-23	2016-06-07	2.0	2011-05-26	4.0	
3	4	Brielle	female	2013-07-17	NaN	NaN	NaN	NaN	
4	5	Benjamin	male	2010-11-25	NaN	NaN	NaN	NaN	

In [36]:
```
#21. Test to see if we can drop columns

#Display the columns
data = pd.merge(users,first_occuring_transactions, how='left', on='UserID')
my_columns = list(data.columns)
my_columns
```

Out[36]:
```
['UserID',
 'User',
 'Gender',
 'Registered',
 'Cancelled',
 'TransactionID',
 'TransactionDate',
 'ProductID',
 'Quantity']
```

In [37]:
```
#Display the columns having na.
list(data.dropna(thresh=int(data.shape[0] * .9), axis=1).columns) #set threshold
```

Out[37]:
```
['UserID', 'User', 'Gender', 'Registered']
```

In [38]:
```
# Display the columns if it has nan data
missing_info = list(data.columns[data.isnull().any()])
missing_info
```

Out[38]:
```
['Cancelled', 'TransactionID', 'TransactionDate', 'ProductID', 'Quantity']
```

In [39]:
```
# Count of missing data
for col in missing_info:
  num_missing = data[data[col].isnull() == True].shape[0]
  print('number missing for column {}: {}'.format(col, num_missing))
```

```
number missing for column Cancelled: 3
number missing for column TransactionID: 2
number missing for column TransactionDate: 2
number missing for column ProductID: 2
number missing for column Quantity: 2
```

In [40]:
```
#Percentage missing.
for col in missing_info:
   percent_missing = data[data[col].isnull() == True].shape[0] / data.shape[0]
   print('percent missing for column {}: {} '.format(
col, percent_missing))
```

```
percent missing for column Cancelled: 0.6
percent missing for column TransactionID: 0.4
percent missing for column TransactionDate: 0.4
percent missing for column ProductID: 0.4
percent missing for column Quantity: 0.4
```

In [ ]:
```

```