In [1]:

```
import pandas as pd
import numpy as np
import os
df = pd.read_csv('https://raw.githubusercontent.com/jackiekazil/data-wrangling/master/
```

Out[1]:

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

In [2]:

```
df1 = pd.read_csv('https://raw.githubusercontent.com/kjam/data-wrangling-pycon/master/
df1.head(2)
```

Out[2]:

	STATION	STATION_NAME	DATE	PRCP	SNWD	SNOW	TMAX	TMIN	WDFG	PGTM	...	WT
0	GHCND:GME00111445	BERLIN TEMPELHOF GM	19310101	46	-9999	-9999	-9999	-11	-9999	-9999	...	-99
1	GHCND:GME00111445	BERLIN TEMPELHOF GM	19310102	107	-9999	-9999	50	11	-9999	-9999	...	-99

2 rows × 21 columns

Task 1. Get the Metadata from the above files.

In [3]:

```
<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 [4]:
```

```
<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
WT04            117208 non-null int64
WT16            117208 non-null int64
WT08            117208 non-null int64
WT18            117208 non-null int64
WT03            117208 non-null int64
dtypes: int64(19), object(2)
memory usage: 18.8+ MB
```

Task 2. Get the row names from the above files

```
In [6]:

Out[6]: Index(['Indicator', 'PUBLISH STATES', 'Year', 'WHO region',
               'World Bank income group', 'Country', 'Sex', 'Display Value', 'Numeric',
               'Low', 'High', 'Comments'],
             dtype='object')
```

```
In [7]:

Out[7]: Index(['STATION', 'STATION_NAME', 'DATE', 'PRCP', 'SNWD', 'SNOW', 'TMAX',
               'TMIN', 'WDFG', 'PGTM', 'WSFG', 'WT09', 'WT07', 'WT01', 'WT06', 'WT05',
               'WT04', 'WT16', 'WT08', 'WT18', 'WT03'],
             dtype='object')
```

For testing:

```
In [8]:

Out[8]:
```

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

Task 3. Change the column name from any of the above file

In [9]:
```
df.rename(columns = {'Indicator':'Indicator_id'}, inplace=True)
```

Out[9]:

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

Task 4. Change the column name from any of the above file and store the changes made permanently.

In [10]:
```
df.rename(columns = {'Indicator':'Indicator_id'}, inplace=True)
```

Out[10]:

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

For Testing:

In [11]:

Out[11]:

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

Task 5. Change the names of multiple columns.

In [13]:

```
df.rename(columns = {'PUBLISH STATES':'Publication Status', 'WHO region':'WHO Region'}
```

Out[13]:

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

Task 6. Arrange values of a particular column in ascending order.

In [15]:

Out[15]:

	Indicator_id	Publication Status	Year	WHO Region	World Bank income group	Country	Sex	Display Value	Numeric	Low	High	Comments
0	Life expectancy at birth (years)	Published	1990	Europe	High-income	Andorra	Both sexes	77	77.0	NaN	NaN	NaN
1270	Life expectancy at birth (years)	Published	1990	Europe	High-income	Germany	Male	72	72.0	NaN	NaN	NaN
3193	Life expectancy at birth (years)	Published	1990	Europe	Lower-middle-income	Republic of Moldova	Male	65	65.0	NaN	NaN	NaN
3194	Life expectancy at birth (years)	Published	1990	Europe	Lower-middle-income	Republic of Moldova	Both sexes	68	68.0	NaN	NaN	NaN
3197	Life expectancy at age 60 (years)	Published	1990	Europe	Lower-middle-income	Republic of Moldova	Male	15	15.0	NaN	NaN	NaN

Task 7. Arrange multiple column values in ascending order

```
In [16]: df.sort_values(['Indicator_id', 'Country', 'Year', 'WHO Region', 'Publication Status']
```

Out[16]:

	Indicator_id	Publication Status	Year	WHO Region	World Bank income group	Country	Sex	Display Value	Numeric	Low	High
2798	Healthy life expectancy (HALE) at birth (years)	Published	2000	Eastern Mediterranean	Low-income	Afghanistan	Male	45	45.0	NaN	NaN
3363	Healthy life expectancy (HALE) at birth (years)	Published	2000	Eastern Mediterranean	Low-income	Afghanistan	Both sexes	45	45.0	NaN	NaN
4456	Healthy life expectancy (HALE) at birth (years)	Published	2000	Eastern Mediterranean	Low-income	Afghanistan	Female	45	45.0	NaN	NaN

Test Data for Task 8

```
In [17]:
```

```
Out[17]: Index(['Indicator_id', 'Publication Status', 'Year', 'WHO Region',
               'World Bank income group', 'Country', 'Sex', 'Display Value', 'Numeric',
               'Low', 'High', 'Comments'],
              dtype='object')
```

Task 8. Make country as the first column of the dataframe

```
In [18]: df = df[['Country', 'Indicator_id', 'Publication Status', 'Year', 'WHO Region',
                'World Bank income group', 'Sex', 'Display Value', 'Numeric',
                'Low', 'High', 'Comments']]
```

```
Out[18]: Index(['Country', 'Indicator_id', 'Publication Status', 'Year', 'WHO Region',
               'World Bank income group', 'Sex', 'Display Value', 'Numeric', 'Low',
               'High', 'Comments'],
              dtype='object')
```

```
In [19]:
```

Out[19]:

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

Task 9. Get the column array using a variable

In [23]:

Out[23]:

```
array([['Andorra', 'Life expectancy at birth (years)', 'Published', ...,
         nan, nan, nan],
        ['Andorra', 'Life expectancy at birth (years)', 'Published', ...,
         nan, nan, nan],
        ['Andorra', 'Life expectancy at age 60 (years)', 'Published', ...,
         nan, nan, nan],
        ...,
        ['South Africa',
         'Healthy life expectancy (HALE) at birth (years)', 'Published',
         ..., nan, nan, nan],
        ['Zambia', 'Healthy life expectancy (HALE) at birth (years)',
         'Published', ..., nan, nan, nan],
        ['Zimbabwe', 'Healthy life expectancy (HALE) at birth (years)',
         'Published', ..., nan, nan, nan]], dtype=object)
```

Task 10. Get the subset rows 11, 24, 37

In [24]:

Out[24]:

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

Task 11. Get the subset rows excluding 5, 12, 23, and 56

In [35]:

```
excludedRows = df.index.isin([5,12,23,34,56])
```

Out[35]:

	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
2	Andorra	Life expectancy at age 60 (years)	Published	2012	Europe	High-income	Female	28	28.0	NaN	NaN
3	Andorra	Life expectancy at age 60 (years)	Published	2000	Europe	High-income	Both sexes	23	23.0	NaN	NaN
4	United Arab Emirates	Life expectancy at birth (years)	Published	2012	Eastern Mediterranean	High-income	Female	78	78.0	NaN	NaN
6	Antigua and Barbuda	Life expectancy at age 60 (years)	Published	1990	Americas	High-income	Male	17	17.0	NaN	NaN
7	Antigua and Barbuda	Life expectancy at age 60 (years)	Published	2012	Americas	High-income	Both sexes	22	22.0	NaN	NaN
8	Australia	Life expectancy at birth (years)	Published	2012	Western Pacific	High-income	Male	81	81.0	NaN	NaN
9	Australia	Life expectancy at birth (years)	Published	2000	Western Pacific	High-income	Both sexes	80	80.0	NaN	NaN
10	Australia	Life expectancy at birth (years)	Published	2012	Western Pacific	High-income	Both sexes	83	83.0	NaN	NaN
11	Austria	Life expectancy at birth (years)	Published	2012	Europe	High-income	Female	83	83.0	NaN	NaN
13	Belgium	Life expectancy at birth (years)	Published	2012	Europe	High-income	Female	83	83.0	NaN	NaN
14	Bahrain	Life expectancy at birth (years)	Published	2000	Eastern Mediterranean	High-income	Male	73	73.0	NaN	NaN
15	Bahrain	Life expectancy at birth (years)	Published	1990	Eastern Mediterranean	High-income	Female	74	74.0	NaN	NaN
16	Bahrain	Life expectancy	Published	1990	Eastern	High-	Male	17	17.0	NaN	NaN

Load datasets from CSV

In [37]:

```
users=pd.read_csv('https://raw.githubusercontent.com/ben519/DataWrangling/master/Data/
sessions=pd.read_csv('https://raw.githubusercontent.com/ben519/DataWrangling/master/Da
products=pd.read_csv('https://raw.githubusercontent.com/ben519/DataWrangling/master/Da
```

In [38]:

Out[38]:

	UserID	User	Gender	Registered	Cancelled
0	1	Charles	male	2012-12-21	NaN
1	2	Pedro	male	2010-08-01	2010-08-08
2	3	Caroline	female	2012-10-23	2016-06-07
3	4	Brielle	female	2013-07-17	NaN
4	5	Benjamin	male	2010-11-25	NaN

In [39]:

Out[39]:

	SessionID	SessionDate	UserID
0	1	2010-01-05	2
1	2	2010-08-01	2
2	3	2010-11-25	2
3	4	2011-09-21	5
4	5	2011-10-19	4

In [40]:

Out[40]:

	ProductID	Product	Price
0	1	A	14.16
1	2	B	33.04
2	3	C	10.65
3	4	D	10.02
4	5	E	29.66

In [41]:

Out[41]:

	TransactionID	TransactionDate	UserID	ProductID	Quantity
0	1	2010-08-21	7.0	2	1
1	2	2011-05-26	3.0	4	1
2	3	2011-06-16	3.0	3	1
3	4	2012-08-26	1.0	2	3
4	5	2013-06-06	2.0	4	1

Task 12: Join users to transactions, keeping all rows from transactions and only matching rows from users (left join)

In [42]:

```
pd.merge
```

	TransactionID	TransactionDate	UserID	ProductID	Quantity	User	Gender	Registered	Cancelled
0	1	2010-08-21	7.0	2	1	NaN	NaN	NaN	NaN
1	2	2011-05-26	3.0	4	1	Caroline	female	2012-10-23	2016-06-07
2	3	2011-06-16	3.0	3	1	Caroline	female	2012-10-23	2016-06-07
3	4	2012-08-26	1.0	2	3	Charles	male	2012-12-21	NaN
4	5	2013-06-06	2.0	4	1	Pedro	male	2010-08-01	2010-08-08
5	6	2013-12-23	2.0	5	6	Pedro	male	2010-08-01	2010-08-08
6	7	2013-12-30	3.0	4	1	Caroline	female	2012-10-23	2016-06-07
7	8	2014-04-24	NaN	2	3	NaN	NaN	NaN	NaN
8	9	2015-04-24	7.0	4	3	NaN	NaN	NaN	NaN
9	10	2016-05-08	3.0	4	4	Caroline	female	2012-10-23	2016-06-07

Task 13: Which transactions have a UserID not in users?

In [43]:

	TransactionID	TransactionDate	UserID	ProductID	Quantity
0	1	2010-08-21	7.0	2	1
7	8	2014-04-24	NaN	2	3
8	9	2015-04-24	7.0	4	3

Task 14. Join users to transactions, keeping only rows from transactions and users that match via UserID (inner join)

In [46]:

	TransactionID	TransactionDate	UserID	ProductID	Quantity	User	Gender	Registered	Cancelled
0	2	2011-05-26	3.0	4	1	Caroline	female	2012-10-23	2016-06-07
1	3	2011-06-16	3.0	3	1	Caroline	female	2012-10-23	2016-06-07
2	7	2013-12-30	3.0	4	1	Caroline	female	2012-10-23	2016-06-07
3	10	2016-05-08	3.0	4	4	Caroline	female	2012-10-23	2016-06-07
4	4	2012-08-26	1.0	2	3	Charles	male	2012-12-21	NaN
5	5	2013-06-06	2.0	4	1	Pedro	male	2010-08-01	2010-08-08
6	6	2013-12-23	2.0	5	6	Pedro	male	2010-08-01	2010-08-08

Task 15. Join users to transactions, displaying all matching rows AND all non-matching rows (full outer join)

In [47]:

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

Task 16. Determine which sessions occurred on the same day each user registered

In [48]:

	SessionID	SessionDate	UserID	User	Gender	Registered	Cancelled
0	1	2010-01-05	2	Pedro	male	2010-08-01	2010-08-08
1	2	2010-08-01	2	Pedro	male	2010-08-01	2010-08-08
2	3	2010-11-25	2	Pedro	male	2010-08-01	2010-08-08
3	4	2011-09-21	5	Benjamin	male	2010-11-25	NaN
4	5	2011-10-19	4	Brielle	female	2013-07-17	NaN
5	6	2012-10-23	4	Brielle	female	2013-07-17	NaN
6	8	2013-05-22	4	Brielle	female	2013-07-17	NaN
7	9	2013-07-17	4	Brielle	female	2013-07-17	NaN
8	10	2016-01-11	4	Brielle	female	2013-07-17	NaN
9	7	2012-12-21	3	Caroline	female	2012-10-23	2016-06-07

In [49]:
```
sameDayUserReg=pd.merge(sessions,users, on='UserID', how='inner')
```

Out[49]:

	SessionID	SessionDate	UserID	User	Gender	Registered	Cancelled
0	1	2010-01-05	2	Pedro	male	2010-08-01	2010-08-08
1	2	2010-08-01	2	Pedro	male	2010-08-01	2010-08-08
2	3	2010-11-25	2	Pedro	male	2010-08-01	2010-08-08
3	4	2011-09-21	5	Benjamin	male	2010-11-25	NaN
4	5	2011-10-19	4	Brielle	female	2013-07-17	NaN
5	6	2012-10-23	4	Brielle	female	2013-07-17	NaN
6	8	2013-05-22	4	Brielle	female	2013-07-17	NaN
7	9	2013-07-17	4	Brielle	female	2013-07-17	NaN
8	10	2016-01-11	4	Brielle	female	2013-07-17	NaN
9	7	2012-12-21	3	Caroline	female	2012-10-23	2016-06-07

In [52]:

Out[52]:

	SessionID	SessionDate	UserID	User	Gender	Registered	Cancelled
1	2	2010-08-01	2	Pedro	male	2010-08-01	2010-08-08
7	9	2013-07-17	4	Brielle	female	2013-07-17	NaN

Task 17. Build a dataset with every possible (UserID, ProductID) pair (cross join)

In [53]:
```
possibleDataSet = users.assign(value=1).merge(products.assign(value=1)).drop('value',
```

	UserID	User	Gender	Registered	Cancelled	ProductID	Product	Price
0	1	Charles	male	2012-12-21	NaN	1	A	14.16
1	1	Charles	male	2012-12-21	NaN	2	B	33.04
2	1	Charles	male	2012-12-21	NaN	3	C	10.65
3	1	Charles	male	2012-12-21	NaN	4	D	10.02
4	1	Charles	male	2012-12-21	NaN	5	E	29.66
5	2	Pedro	male	2010-08-01	2010-08-08	1	A	14.16
6	2	Pedro	male	2010-08-01	2010-08-08	2	B	33.04
7	2	Pedro	male	2010-08-01	2010-08-08	3	C	10.65
8	2	Pedro	male	2010-08-01	2010-08-08	4	D	10.02
9	2	Pedro	male	2010-08-01	2010-08-08	5	E	29.66
10	3	Caroline	female	2012-10-23	2016-06-07	1	A	14.16
11	3	Caroline	female	2012-10-23	2016-06-07	2	B	33.04
12	3	Caroline	female	2012-10-23	2016-06-07	3	C	10.65
13	3	Caroline	female	2012-10-23	2016-06-07	4	D	10.02
14	3	Caroline	female	2012-10-23	2016-06-07	5	E	29.66
15	4	Brielle	female	2013-07-17	NaN	1	A	14.16
16	4	Brielle	female	2013-07-17	NaN	2	B	33.04
17	4	Brielle	female	2013-07-17	NaN	3	C	10.65
18	4	Brielle	female	2013-07-17	NaN	4	D	10.02
19	4	Brielle	female	2013-07-17	NaN	5	E	29.66
20	5	Benjamin	male	2010-11-25	NaN	1	A	14.16
21	5	Benjamin	male	2010-11-25	NaN	2	B	33.04
22	5	Benjamin	male	2010-11-25	NaN	3	C	10.65
23	5	Benjamin	male	2010-11-25	NaN	4	D	10.02
24	5	Benjamin	male	2010-11-25	NaN	5	E	29.66

Task 18. Determine how much quantity of each product was purchased by each user

```
In [54]:
```

	TransactionID	TransactionDate	UserID	ProductID	Quantity
0	1	2010-08-21	7.0	2	1
1	2	2011-05-26	3.0	4	1
2	3	2011-06-16	3.0	3	1
4	5	2013-06-06	2.0	4	1
6	7	2013-12-30	3.0	4	1
3	4	2012-08-26	1.0	2	3
7	8	2014-04-24	NaN	2	3
8	9	2015-04-24	7.0	4	3
9	10	2016-05-08	3.0	4	4
5	6	2013-12-23	2.0	5	6

Task 19. For each user, get each possible pair of pair transactions

In [55]:

Out[55]:

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

Task20. Join each user to his/her first occuring transaction in the transactions table

In [56]:

```
data=pd.merge(users, transactions.groupby('UserID').first().reset_index(), how='left',
```

Out[56]:

	UserID	User	Gender	Registered	Cancelled	TransactionID	TransactionDate	ProductID	Quantity
0	1	Charles	male	2012-12-21	NaN	4.0	2012-08-26	2.0	3.0
1	2	Pedro	male	2010-08-01	2010-08-08	5.0	2013-06-06	4.0	1.0
2	3	Caroline	female	2012-10-23	2016-06-07	2.0	2011-05-26	4.0	1.0
3	4	Brielle	female	2013-07-17	NaN	NaN	NaN	NaN	NaN
4	5	Benjamin	male	2010-11-25	NaN	NaN	NaN	NaN	NaN

Task 21. Test to see if we can drop columns

In [58]:

```
In [59]:

Out[59]: ['UserID', 'User', 'Gender', 'Registered']
```

```
In [60]: missingInfo = list(data.columns[data.isnull().any()])

Out[60]: ['Cancelled', 'TransactionID', 'TransactionDate', 'ProductID', 'Quantity']
```

```
In [61]: for col in missingInfo:
             missingNumber = data[data[col].isnull() == True].shape[0]
```

```
Missing Number for Col Cancelled: 3
Missing Number for Col TransactionID: 2
Missing Number for Col TransactionDate: 2
Missing Number for Col ProductID: 2
Missing Number for Col Quantity: 2
```

```
In [62]: for col in missingInfo:
             percentMissing = data[data[col].isnull() == True].shape[0] / data.shape[0]
```

```
Col Percent Missing Cancelled: 0.6
Col Percent Missing TransactionID: 0.4
Col Percent Missing TransactionDate: 0.4
Col Percent Missing ProductID: 0.4
Col Percent Missing Quantity: 0.4
```

End of Project