# Data Wrangling Lab

Estimated time needed: **45 to 60** minutes

In this assignment you will be performing data wrangling.

## Objectives

In this lab you will perform the following:

- Identify duplicate values in the dataset.
- Remove duplicate values from the dataset.
- Identify missing values in the dataset.
- Impute the missing values in the dataset.
- Normalize data in the dataset.

---

## Hands on Lab

Import pandas module.

In [1]:
```
import pandas as pd
```

```
C:\Users\Administrator\anaconda3\Lib\site-packages\pandas\core\arrays\masked.py:60: U
serWarning: Pandas requires version '1.3.6' or newer of 'bottleneck' (version '1.3.5'
currently installed).
  from pandas.core import (
```

Load the dataset into a dataframe.

In [ ]:
```
df = pd.read_csv("https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/I
```

In [2]:
```
df = pd.read_csv('m1_survey_data.csv')
```

# Finding duplicates

In this section you will identify duplicate values in the dataset.

Find how many duplicate rows exist in the dataframe.

In [3]:
```
# your code goes here
df.duplicated().sum()
```

Out[3]:
```
154
```

# Removing duplicates

Remove the duplicate rows from the dataframe.

In [4]:
```
# your code goes here
df.drop_duplicates(inplace = True)
```

Verify if duplicates were actually dropped.

In [5]:
```
# your code goes here
df.duplicated().sum()
```

Out[5]:
```
0
```

In [6]:
```
df
```

Out[6]:

| | Respondent | MainBranch | Hobbyist | OpenSourcer | OpenSource | Employment | Country | Stud |
|---|---|---|---|---|---|---|---|---|
| **0** | 4 | I am a developer by profession | No | Never | The quality of OSS and closed source software ... | Employed full-time | United States | |
| **1** | 9 | I am a developer by profession | Yes | Once a month or more often | The quality of OSS and closed source software ... | Employed full-time | New Zealand | |
| **2** | 13 | I am a developer by profession | Yes | Less than once a month but more than once per ... | OSS is, on average, of HIGHER quality than pro... | Employed full-time | United States | |
| **3** | 16 | I am a developer by profession | Yes | Never | The quality of OSS and closed source software ... | Employed full-time | United Kingdom | |
| **4** | 17 | I am a developer by profession | Yes | Less than once a month but more than once per ... | The quality of OSS and closed source software ... | Employed full-time | Australia | |
| **...** | ... | ... | ... | ... | ... | ... | ... | |
| **11547** | 25136 | I am a developer by profession | Yes | Never | OSS is, on average, of HIGHER quality than pro... | Employed full-time | United States | |
| **11548** | 25137 | I am a developer by profession | Yes | Never | The quality of OSS and closed source software ... | Employed full-time | Poland | |
| **11549** | 25138 | I am a developer by profession | Yes | Less than once per year | The quality of OSS and closed source software ... | Employed full-time | United States | |
| **11550** | 25141 | I am a developer by profession | Yes | Less than once a month but more than once per ... | OSS is, on average, of LOWER quality than prop... | Employed full-time | Switzerland | |
| **11551** | 25142 | I am a developer by profession | Yes | Less than once a month but more than once per ... | OSS is, on average, of HIGHER quality than pro... | Employed full-time | United Kingdom | |

11398 rows × 85 columns

# Finding Missing values

Find the missing values for all columns.

In [38]:
```
df['CompFreq'].value_counts()
```

Out[38]:
```
CompFreq
Yearly     6073
Monthly    4788
Weekly      331
Name: count, dtype: int64
```

In [8]:
```
# your code goes here
df.isna().sum()
```

Out[8]:
```
Respondent        0
MainBranch        0
Hobbyist          0
OpenSourcer       0
OpenSource       81
                ...
Sexuality       542
Ethnicity       675
Dependents      140
SurveyLength     19
SurveyEase       14
Length: 85, dtype: int64
```

Find out how many rows are missing in the column 'WorkLoc'

In [9]:
```
# your code goes here
df['WorkLoc'].isna().sum()
```

Out[9]:
```
32
```

# Imputing missing values

Find the value counts for the column WorkLoc.

In [10]:
```
# your code goes here
df['WorkLoc'].value_counts()
```

Out[10]:
```
WorkLoc
Office                                          6806
Home                                            3589
Other place, such as a coworking space or cafe   971
Name: count, dtype: int64
```

Identify the value that is most frequent (majority) in the WorkLoc column.

In [12]:
```
#make a note of the majority value here, for future reference
mode = df['WorkLoc'].mode()
```

```
mode
```

Out[12]:
```
0    Office
Name: WorkLoc, dtype: object
```

Impute (replace) all the empty rows in the column WorkLoc with the value that you have identified as majority.

In [13]:
```
# your code goes here
#df['WorkLoc'].fillna(mode, inplace = True)

df.fillna({'WorkLoc' : 'Office'}, inplace = True)
```

After imputation there should ideally not be any empty rows in the WorkLoc column.

Verify if imputing was successful.

In [14]:
```
# your code goes here
df['WorkLoc'].isna().sum()
```

Out[14]:
```
0
```

# Normalizing data

There are two columns in the dataset that talk about compensation.

One is "CompFreq". This column shows how often a developer is paid (Yearly, Monthly, Weekly).

The other is "CompTotal". This column talks about how much the developer is paid per Year, Month, or Week depending upon his/her "CompFreq".

This makes it difficult to compare the total compensation of the developers.

In this section you will create a new column called 'NormalizedAnnualCompensation' which contains the 'Annual Compensation' irrespective of the 'CompFreq'.

Once this column is ready, it makes comparison of salaries easy.

---

List out the various categories in the column 'CompFreq'

In [16]:
```
df.CompTotal.value_counts()
```

Out[16]:
```
CompTotal
70000.0      241
60000.0      239
50000.0      223
100000.0     218
80000.0      194
             ...
163500.0       1
2530000.0      1
1.0            1
162929.0       1
74400.0        1
Name: count, Length: 1509, dtype: int64
```

In [15]:
```
# your code goes here
df.CompFreq.value_counts()
```

Out[15]:
```
CompFreq
Yearly     6073
Monthly    4788
Weekly      331
Name: count, dtype: int64
```

Create a new column named 'NormalizedAnnualCompensation'. Use the hint given below if needed.

Double click to see the **Hint**.

In [27]:
```
# your code goes here
NormalizedAnnualCompensation = []
n = 0
for items in df.CompFreq:
    if df[df['CompFreq'] == 'Yearly']:
        n = df.CompTotal
    elif df[df.CompFreq == 'Monthly']:
        n == df.CompTotal*12
    else:
        n = df.CompTotal*52
    NormalizedAnnualCompensation.append(n)
print(NormalizedAnnualCompensation)
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
~\AppData\Local\Temp\ipykernel_10392\3809668315.py in ?()
      1 # your code goes here
      2 NormalizedAnnualCompensation = []
      3 n = 0
      4 for items in df.CompFreq:
----> 5     if df[df['CompFreq'] == 'Yearly']:
      6         n = df.CompTotal
      7     elif df[df.CompFreq == 'Monthly']:
      8         n == df.CompTotal*12

~\anaconda3\Lib\site-packages\pandas\core\generic.py in ?(self)
   1574     @final
   1575     def __nonzero__(self) -> NoReturn:
-> 1576         raise ValueError(
   1577             f"The truth value of a {type(self).__name__} is ambiguous. "
   1578             "Use a.empty, a.bool(), a.item(), a.any() or a.all()."
   1579         )

ValueError: The truth value of a DataFrame is ambiguous. Use a.empty, a.bool(), a.ite
m(), a.any() or a.all().
```

# Authors

Ramesh Sannareddy

## Other Contributors

Rav Ahuja

# Change Log

| Date (YYYY-MM-DD) | Version | Changed By | Change Description |
|---|---|---|---|
| 2020-10-17 | 0.1 | Ramesh Sannareddy | Created initial version of the lab |