1

60004210155
Jigar Siddhpura
C22

Experiment 1

Aim: To perform data preprocessing in terms of handling missing data, removing outliers, eliminating duplicate rows & modifying data.

Theory:

Python is an easy to learn programming language, best for beginners. Most of the data is uncleaned, so it needs to be cleaned. It may include errors, duplicate records, contain incomplete or outdated data, contain improper formatting. Data cleaning is the process of cleaning dirty data & rectifying it.

Tasks to do in data cleaning:

1. Handling missing values — Dataset may contain missing values, making it incomplete.

To handle this, we do:
a) Data observations — Do this when percentage of data is less
b) Remove column — If significant data is missing, remove the column.
c) Impute missing values — Restore missing values with mean, median, etc accordingly.

2. Outliers — It is an unusual observation that lies away from the majority. This affects model significantly.

3. Remove duplicate rows — Data contain duplicate rows sometimes. So we drop it.
Drop duplicate rows based on primary key (eg: ID)

4. Fixing Data Type — Often in dataset values are not stored in correct data type. This creates issues in later stage.

Procedure:

1. Imported libraries like pandas, numpy, matplotlib.
2. Analyzed the data using functions like — info(), describe()
3. Calculated count of all missing values in all the columns with 'df.isna().sum()'.
4. Dropped 4 columns — country, country-code, has-expired, job-board, they being not important in creating model.
5. Now, the job-type column has lot of unique values. So I removed irrelevant characters like '\xa0' & used 'split()' func. Later ~~Converted~~ ~~Name~~ Lematized the column name into root form.
6. Also, 'location' column has lot of extra data. So, I cut the extra part.

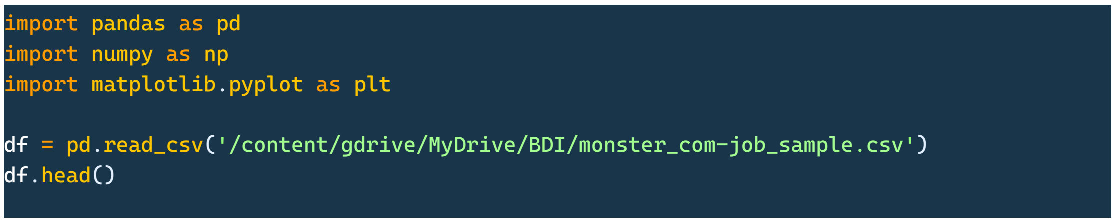
Observations :

Tasks performed for data cleaning : Handling missing values, null values, drop irrelevant data, duplicate records, etc.

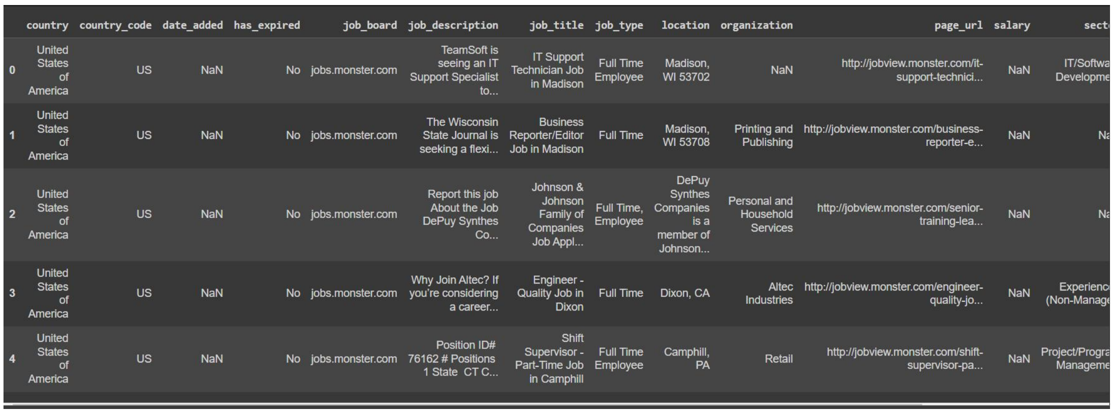
After performing data cleaning, we can now use the data for analysis .

Conclusion :

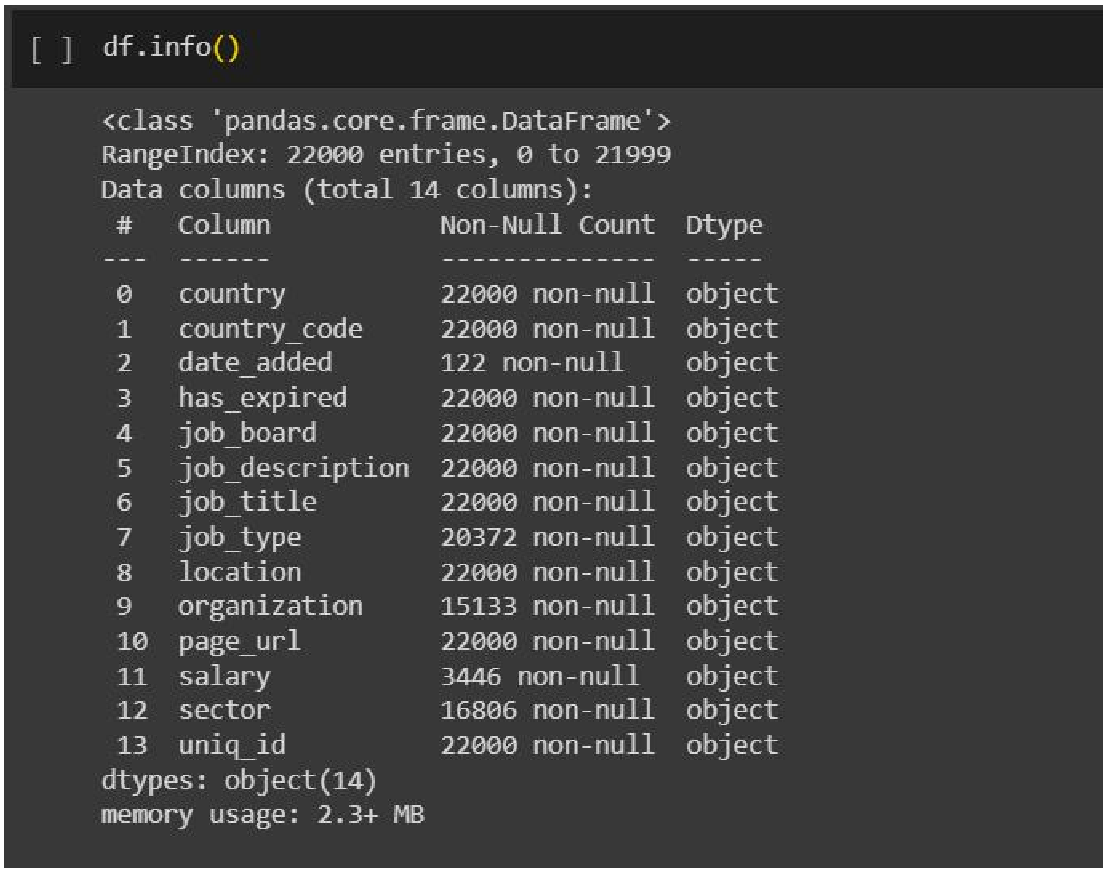
Hence, after performing this experiment, we can say that data cleaning is most important & base step to do ML as it can improve model performance.



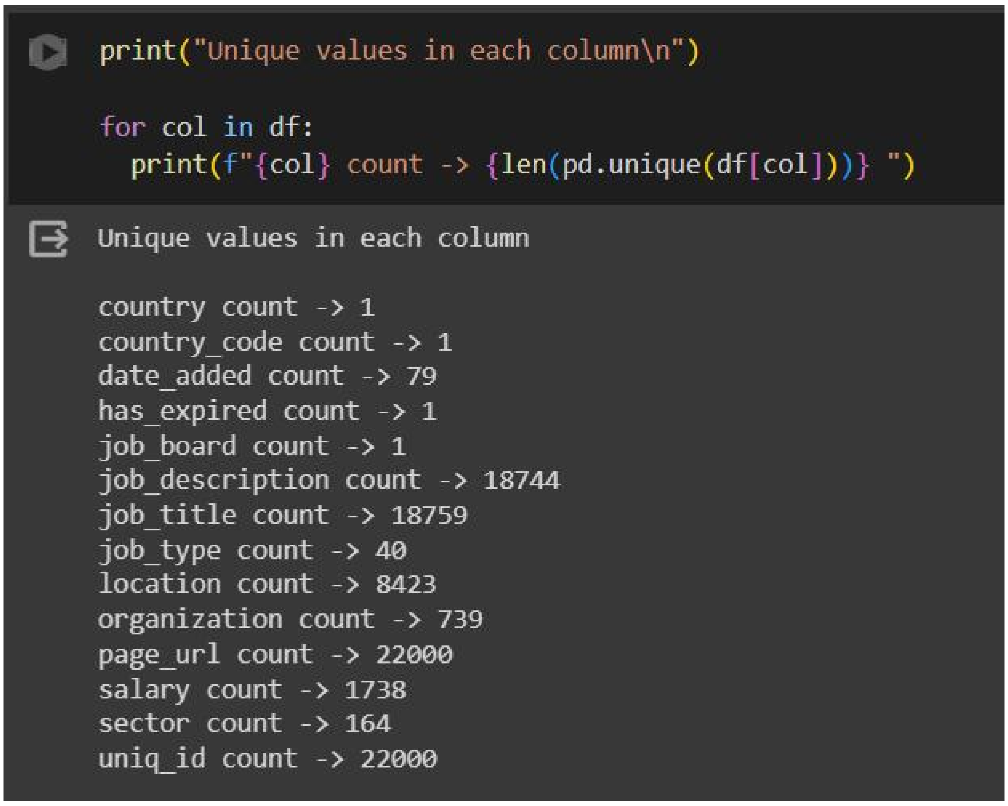
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt

df = pd.read_csv('/content/gdrive/MyDrive/BDI/monster_com-job_sample.csv')
df.head()
```



	country	country_code	date_added	has_expired	job_board	job_description	job_title	job_type	location	organization	page_url	salary	sect
0	United States of America	US	NaN	No	jobs.monster.com	TeamSoft is seeing an IT Support Specialist to...	IT Support Technician Job in Madison	Full Time Employee	Madison, WI 53702	NaN	http://jobview.monster.com/it-support-technici...	NaN	IT/Softwa Developme
1	United States of America	US	NaN	No	jobs.monster.com	The Wisconsin State Journal is seeking a flexi...	Business Reporter/Editor Job in Madison	Full Time	Madison, WI 53708	Printing and Publishing	http://jobview.monster.com/business-reporter-e...	NaN	Na
2	United States of America	US	NaN	No	jobs.monster.com	Report this job About the Job DePuy Synthes Co...	Johnson & Johnson Family of Companies Job Appl...	Full Time, Employee	DePuy Synthes Companies is a member of Johnson...	Personal and Household Services	http://jobview.monster.com/senior-training-lea...	NaN	Na
3	United States of America	US	NaN	No	jobs.monster.com	Why Join Altec? If you're considering a career...	Engineer - Quality Job in Dixon	Full Time	Dixon, CA	Altec Industries	http://jobview.monster.com/engineer-quality-jo...	NaN	Experienc (Non-Manag
4	United States of America	US	NaN	No	jobs.monster.com	Position ID# 76162 # Positions 1 State CT C...	Shift Supervisor - Part-Time Job in Camphill	Full Time Employee	Camphill, PA	Retail	http://jobview.monster.com/shift-supervisor-pa...	NaN	Project/Progra Manageme

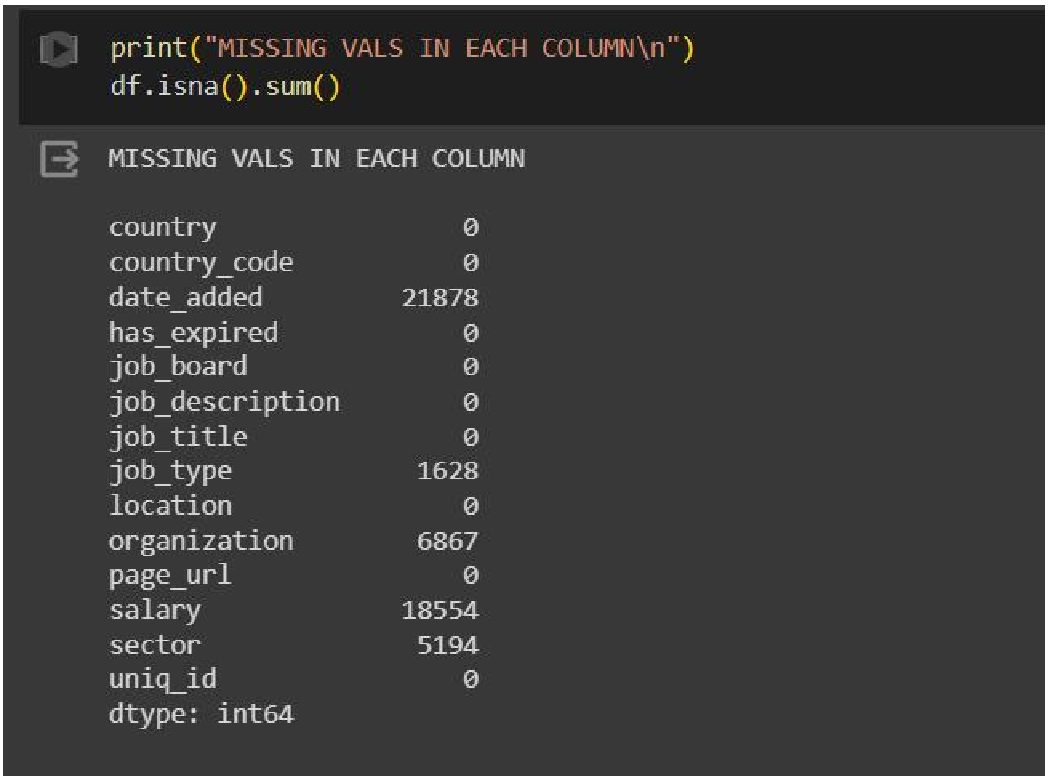


```
[ ]  df.info()

     <class 'pandas.core.frame.DataFrame'>
     RangeIndex: 22000 entries, 0 to 21999
     Data columns (total 14 columns):
      #   Column           Non-Null Count  Dtype
     ---  ------           --------------  -----
      0   country          22000 non-null  object
      1   country_code     22000 non-null  object
      2   date_added       122 non-null    object
      3   has_expired      22000 non-null  object
      4   job_board        22000 non-null  object
      5   job_description  22000 non-null  object
      6   job_title        22000 non-null  object
      7   job_type         20372 non-null  object
      8   location         22000 non-null  object
      9   organization     15133 non-null  object
      10  page_url         22000 non-null  object
      11  salary           3446 non-null   object
      12  sector           16806 non-null  object
      13  uniq_id          22000 non-null  object
     dtypes: object(14)
     memory usage: 2.3+ MB
```



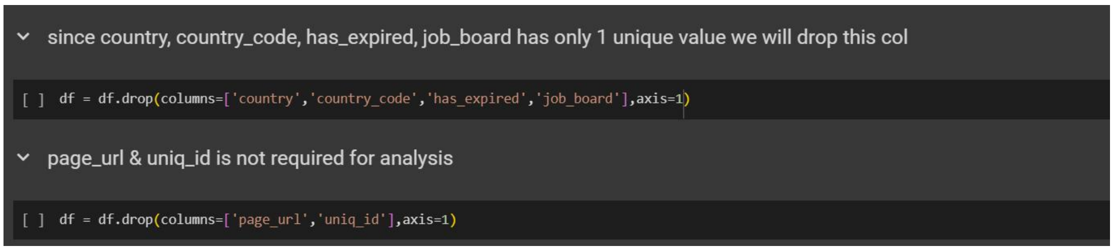
```
print("Unique values in each column\n")

for col in df:
  print(f"{col} count -> {len(pd.unique(df[col]))} ")
```

```
Unique values in each column

country count -> 1
country_code count -> 1
date_added count -> 79
has_expired count -> 1
job_board count -> 1
job_description count -> 18744
job_title count -> 18759
job_type count -> 40
location count -> 8423
organization count -> 739
page_url count -> 22000
salary count -> 1738
sector count -> 164
uniq_id count -> 22000
```



```
print("MISSING VALS IN EACH COLUMN\n")
df.isna().sum()
```

```
MISSING VALS IN EACH COLUMN

country                0
country_code           0
date_added         21878
has_expired            0
job_board              0
job_description        0
job_title              0
job_type            1628
location               0
organization        6867
page_url               0
salary             18554
sector              5194
uniq_id                0
dtype: int64
```

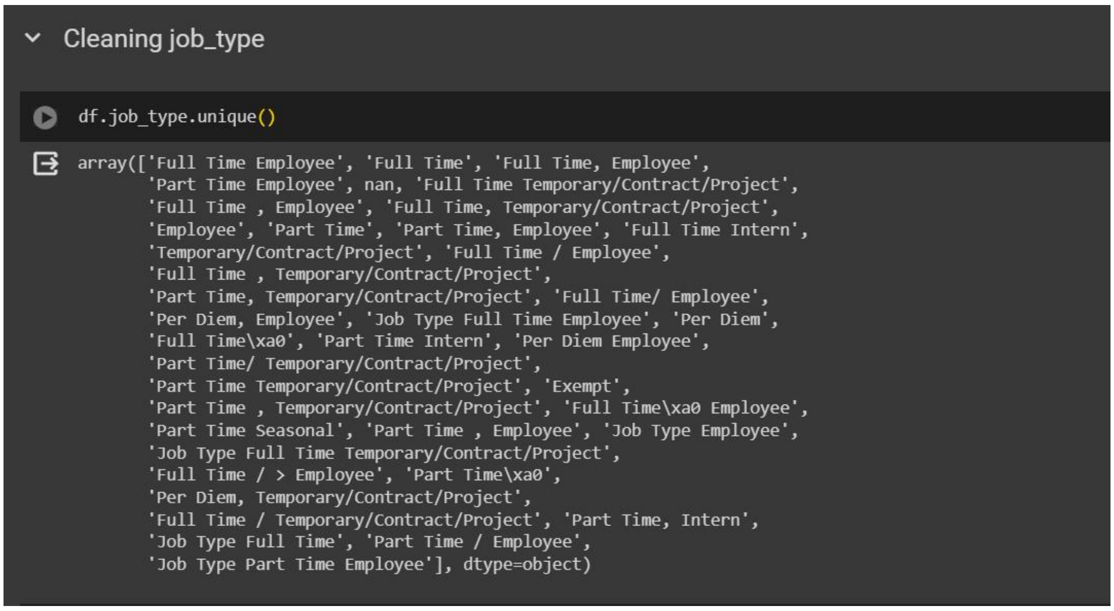


since country, country_code, has_expired, job_board has only 1 unique value we will drop this col

```
[ ]  df = df.drop(columns=['country','country_code','has_expired','job_board'],axis=1)
```

page_url & uniq_id is not required for analysis

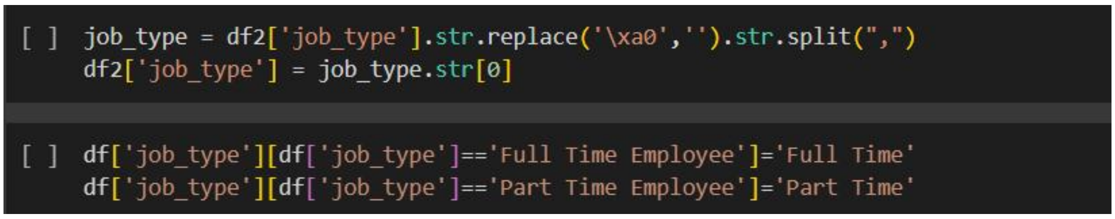
```
[ ]  df = df.drop(columns=['page_url','uniq_id'],axis=1)
```



Cleaning job_type

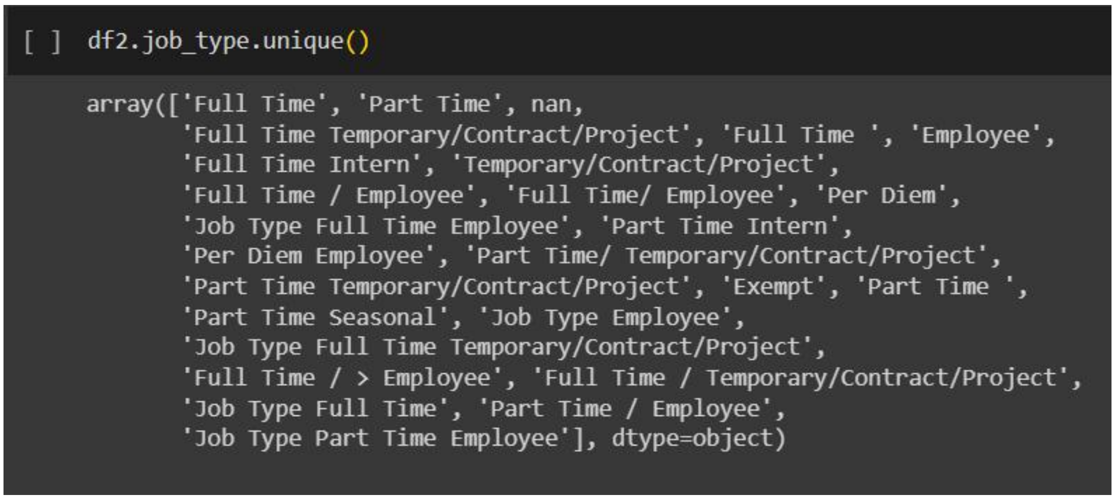
```
df.job_type.unique()
```

```
array(['Full Time Employee', 'Full Time', 'Full Time, Employee',
       'Part Time Employee', nan, 'Full Time Temporary/Contract/Project',
       'Full Time , Employee', 'Full Time, Temporary/Contract/Project',
       'Employee', 'Part Time', 'Part Time, Employee', 'Full Time Intern',
       'Temporary/Contract/Project', 'Full Time / Employee',
       'Full Time , Temporary/Contract/Project',
       'Part Time, Temporary/Contract/Project', 'Full Time/ Employee',
       'Per Diem, Employee', 'Job Type Full Time Employee', 'Per Diem',
       'Full Time\xa0', 'Part Time Intern', 'Per Diem Employee',
       'Part Time/ Temporary/Contract/Project',
       'Part Time Temporary/Contract/Project', 'Exempt',
       'Part Time , Temporary/Contract/Project', 'Full Time\xa0 Employee',
       'Part Time Seasonal', 'Part Time , Employee', 'Job Type Employee',
       'Job Type Full Time Temporary/Contract/Project',
       'Full Time / > Employee', 'Part Time\xa0',
       'Per Diem, Temporary/Contract/Project',
       'Full Time / Temporary/Contract/Project', 'Part Time, Intern',
       'Job Type Full Time', 'Part Time / Employee',
       'Job Type Part Time Employee'], dtype=object)
```



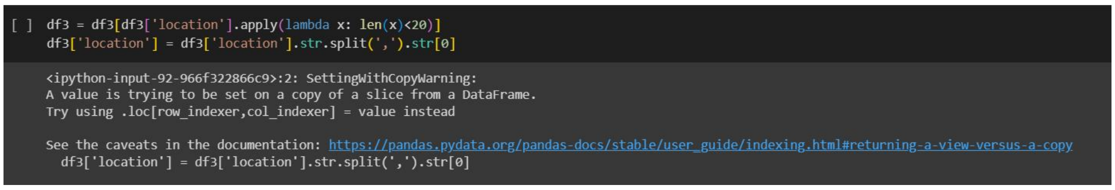
```
[ ]  job_type = df2['job_type'].str.replace('\xa0','').str.split(",")
     df2['job_type'] = job_type.str[0]
```

```
[ ]  df['job_type'][df['job_type']=='Full Time Employee']='Full Time'
     df['job_type'][df['job_type']=='Part Time Employee']='Part Time'
```

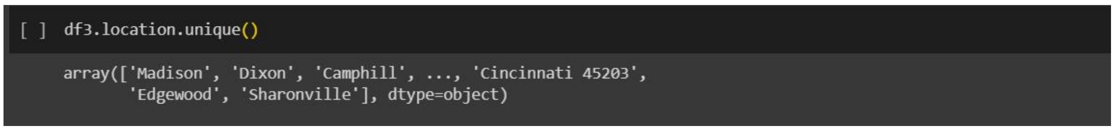


```
[ ]  df2.job_type.unique()
```

```
array(['Full Time', 'Part Time', nan,
       'Full Time Temporary/Contract/Project', 'Full Time ', 'Employee',
       'Full Time Intern', 'Temporary/Contract/Project',
       'Full Time / Employee', 'Full Time/ Employee', 'Per Diem',
       'Job Type Full Time Employee', 'Part Time Intern',
       'Per Diem Employee', 'Part Time/ Temporary/Contract/Project',
       'Part Time Temporary/Contract/Project', 'Exempt', 'Part Time ',
       'Part Time Seasonal', 'Job Type Employee',
       'Job Type Full Time Temporary/Contract/Project',
       'Full Time / > Employee', 'Full Time / Temporary/Contract/Project',
       'Job Type Full Time', 'Part Time / Employee',
       'Job Type Part Time Employee'], dtype=object)
```



```
[ ]  df3 = df3[df3['location'].apply(lambda x: len(x)<20)]
     df3['location'] = df3['location'].str.split(',').str[0]

     <ipython-input-92-966f322866c9>:2: SettingWithCopyWarning:
     A value is trying to be set on a copy of a slice from a DataFrame.
     Try using .loc[row_indexer,col_indexer] = value instead

     See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
       df3['location'] = df3['location'].str.split(',').str[0]
```



```
[ ]  df3.location.unique()

     array(['Madison', 'Dixon', 'Camphill', ..., 'Cincinnati 45203',
            'Edgewood', 'Sharonville'], dtype=object)
```