Environment: Python 3.8.5 and Jupyter notebook

Libraries used: please include the main libraries you used in your assignment here, e.g.,:

- pandas
- re
- numpy
- matplotlib
- missingno
- datetime

Import libraries

Use the below code if libraries are not installed.

```
# !pip install missingno
# !pip install pandas
# !pip install numpy
# !pip install matplotlib
# !pip install re
# !pip install datetime
```

```
# Code to import libraries as you need in this assessment, e.g.,
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import re
import missingno as msno
import datetime as dt
from datetime import datetime
```

```
# make sure the plots are shown in the notebook
%matplotlib inline
```

Auditing and cleansing the loaded data

Auditing data, identify data problems and fixing them, and recording founded errors as required

```
# Code to audit data
```

```
# Read dataset
df = pd.read_csv('dataset1_with_error.csv')
```

```
# Shape of data set
print(df.shape)
```

```
(55169, 11)
```

Preliminary data checks

```
print(df.describe)
```

```
<bound method NDFrame.describe of             Id                                              Title  \
0      12612628                        Engineering Systems Analyst
1      12612830                            Stress Engineer Glasgow
2      12612844                   Modelling and simulation analyst
3      12613049  Engineering Systems Analyst / Mathematical Mod...
4      12613647         Pioneer, Miser Engineering Systems Analyst
...         ...                                               ...
55164  72705203               TEACHER OF BUSINESS STUDIES AND LAW
55165  72705205       Pensions Administrators (Temporary/Contract)
55166  72705221                           Senior Financial Advisor
55167  72705240    Barclays Future Leaders Development Programmes
55168  72705244  Quality Assurance Environmental Manager Nottin...

          Location                     Company ContractType ContractTime  \
```

```
0         Dorking  Gregory Martin International          NaN    permanent
1         Glasgow  Gregory Martin International          NaN    permanent
2       Hampshire  Gregory Martin International          NaN    permanent
3          Surrey  Gregory Martin International          NaN    permanent
4          Surrey  Gregory Martin International          NaN    permanent
...           ...                           ...          ...          ...
55164   Salisbury                                              contract
55165          UK                   Abenefit2u          NaN     contract
55166      London        Fram Executive Search.            -    permanent
55167     Hackney                     Barclays            -            -
55168  Nottingham      Stephen James Consulting          NaN    permanent

                        Category   Salary        OpenDate        CloseDate  \
0               Engineering Jobs    25000  20130708T120000  20130906T120000
1               Engineering Jobs    30000  20120130T000000  20120330T000000
2               Engineering Jobs    30000  20121221T150000  20130120T150000
3               Engineering Jobs    27500  20131208T150000  20140206T150000
4               Engineering Jobs    25000  20130302T120000  20130501T120000
...                          ...      ...              ...              ...
55164              Teaching Jobs    22800  20120123T120000  20120206T120000
55165  Accounting & Finance Jobs    24000  20130801T150000  20130831T150000
55166  Accounting & Finance Jobs    40000  20130126T000000  20130225T000000
55167                   IT Jobs    36000  20121223T150000  20130221T150000
55168  Healthcare & Nursing Jobs  35000.0  20120110T150000  20120409T150000

                   Source
0         cv-library.co.uk
1         cv-library.co.uk
2         cv-library.co.uk
3         cv-library.co.uk
4         cv-library.co.uk
...                    ...
55164          hays.co.uk
55165     cv-library.co.uk
55166  ifaonlinejobs.co.uk
55167          grb.uk.com
55168        tntjobs.co.uk

[55169 rows x 11 columns]>
```

```
df.head()
```

	Id	Title	Location	Company	ContractType	ContractTime	Category	Salary	
0	12612628	Engineering Systems Analyst	Dorking	Gregory Martin International	NaN	permanent	Engineering Jobs	25000	2013070
1	12612830	Stress Engineer Glasgow	Glasgow	Gregory Martin International	NaN	permanent	Engineering Jobs	30000	2012013
2	12612844	Modelling and simulation analyst	Hampshire	Gregory Martin International	NaN	permanent	Engineering Jobs	30000	2012122
		Engineering Systems		Gregory					

```
df.tail()
```

	Id	Title	Location	Company	ContractType	ContractTime	Category	Salar
55164	72705203	TEACHER OF BUSINESS STUDIES AND LAW	Salisbury			contract	Teaching Jobs	2280
55165	72705205	Pensions Administrators (Temporary/Contract)	UK	Abenefit2u	NaN	contract	Accounting & Finance Jobs	2400
55166	72705221	Senior Financial Advisor	London	Fram Executive Search.	-	permanent	Accounting & Finance Jobs	4000
55167	72705240	Barclays Future Leaders Development	Hackney	Barclays	-	-	IT Jobs	3600

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 55169 entries, 0 to 55168
Data columns (total 11 columns):
 #   Column        Non-Null Count  Dtype
---  ------        --------------  -----
 0   Id            55169 non-null  int64
 1   Title         55169 non-null  object
```

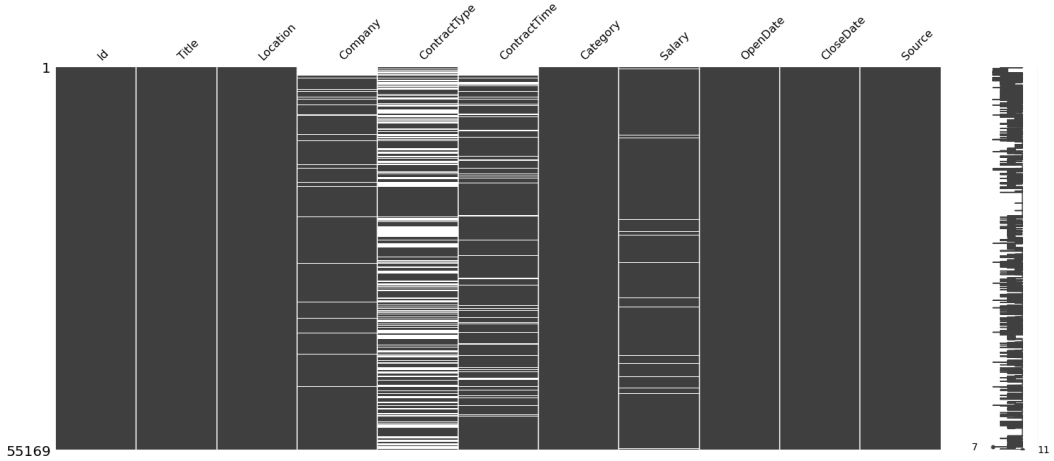
```
 2   Location      55169 non-null  object
 3   Company       51320 non-null  object
 4   ContractType  33493 non-null  object
 5   ContractTime  47047 non-null  object
 6   Category      55169 non-null  object
 7   Salary        53584 non-null  object
 8   OpenDate      55169 non-null  object
 9   CloseDate     55169 non-null  object
 10  Source        55169 non-null  object
dtypes: int64(1), object(10)
memory usage: 4.6+ MB
```

```
df.describe(include = ['O'])
```

	Title	Location	Company	ContractType	ContractTime	Category	Salary	OpenDate	
count	55169	55169	51320	33493	47047	55169	53584	55169	
unique	55166	489	9064	4	4	8	3757	2194	
top	Senior Financial Advisor	UK			permanent	IT Jobs	35000	20120415T150000	2013

```
#checking missing data using missingo library
msno.matrix(df)
```

```
<AxesSubplot:>
```




```
print(f'Total Number of missing Values in Salary:', df['Salary'].isnull().sum())
print(f'Total Number of missing Values in Company:', df['Company'].isnull().sum())
print(f'Total Number of missing Values in ContractTime:', df['ContractTime'].isnull().sum())
print(f'Total Number of missing Values in ContractType:', df['ContractType'].isnull().sum())
```

```
Total Number of missing Values in Salary: 1585
Total Number of missing Values in Company: 3849
Total Number of missing Values in ContractTime: 8122
Total Number of missing Values in ContractType: 21676
```

Checking Company

```
# Checking for any special characters.
df['Company'].str.contains(r'([^A-Za-z]|\s+|\W+|\.|\-|\_)').any()
```

```
/Users/Akshata/opt/anaconda3/lib/python3.8/site-packages/pandas/core/strings.py:2001: UserWarning: This pattern has match groups. T
  return func(self, *args, **kwargs)
True
```

```
# Checking if only if Limited and LTD both present
df['Company'].str.contains(r'(LIMITED|LTD)').any()
```

```
True
```

```
df['Company'].str.contains(r'(AND|&)').any()
```

```
True
```

```
# the following function is the clean and normalize the company field.
def removeEndSpecialChar(company):
    if pd.isnull(company):
        return company
    else:
        # normalize to upper case letters
        company = company.upper()
        # remove special character at the end
        company = re.sub(r"\W+$", "", company)
        # replace LIMITED with LTD
        company = company.replace("LIMITED", "LTD")
        # replace " AND " with " & " for processing
        company = company.replace(" & ", " AND ")
        # remove all special characters except space and dot
        company = re.sub(r'([^\w\s\.]|_)', '', company)
        # change " AND " back to " & "
        company = company.replace(" AND ", " & ")
        # replace multiple spaces and trim spaces
        company = re.sub( '\s+', ' ', company).strip()
        return company

df['Company'] = df.Company.apply(lambda x: removeEndSpecialChar(x))

# Reference:
# https://lms.monash.edu/course/view.php?id=142356&section=9
```

```
print(df['Company'].head(5))
```

```
0    GREGORY MARTIN INTERNATIONAL
1    GREGORY MARTIN INTERNATIONAL
2    GREGORY MARTIN INTERNATIONAL
3    GREGORY MARTIN INTERNATIONAL
4    GREGORY MARTIN INTERNATIONAL
Name: Company, dtype: object
```

```
df['Company'].tail(5)
```

```
55164
55165                  ABENEFIT2U
55166       FRAM EXECUTIVE SEARCH
55167                    BARCLAYS
55168    STEPHEN JAMES CONSULTING
Name: Company, dtype: object
```

```
# Replace all the empty strings to nan so it will be easier to replace nan values.
df['Company'].replace({'':np.nan}, inplace = True)
```

Company has 3849 rows of missing values, Deleting them will affect the model performace, so it should be imputed. This is a Categorical variable, so we will replace all the values using mode.

```
# Imputing Missing Values using mode
df['Company'].fillna(df['Company'].mode()[0], inplace = True)
```

```
# Check if still has any null values
print(f'Total Number of missing Values in Company:', df['Company'].isnull().sum())
```

```
Total Number of missing Values in Company: 0
```

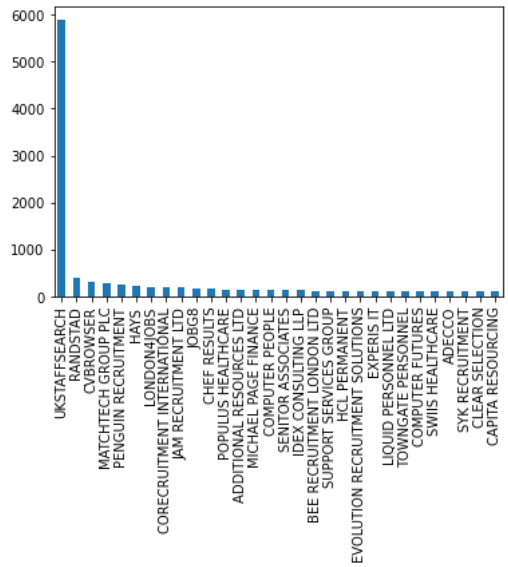
```
# Checking the Value counts
df['Company'].value_counts()
```

```
UKSTAFFSEARCH                5887
RANDSTAD                      403
CVBROWSER                     328
```

```
MATCHTECH GROUP PLC            286
PENGUIN RECRUITMENT            252
                              ...
AMBERSTONE RECRUITMENT LTD       1
IPEOPLE ASSOCIATES LTD           1
WALDERSEY HOUSE                  1
VOLVO FINANCIAL SERVICES         1
ALLSPEEDS LTD                    1
Name: Company, Length: 8542, dtype: int64
```

```
# Plotting top 30 companies
df['Company'].value_counts()[:30].plot.bar()
```

```
<AxesSubplot:>
```



Checking ContractType

```
df['ContractType'].unique()
```

```
array([nan, '-', ' ', 'full_time', 'part_time'], dtype=object)
```

As we can see above we have special characters and empty strings in ContractTypes. Also it has lots of missing Values. So we have to consider to impute them, We will impute missing Values for ContractType using mode as well.

```
# Let us do some further checks to see if it has any other special characters.
df['ContractType'].str.contains(r'([^A-Za-z]|\W+|\.|\_)').any()
```

```
True
```

```
def removeEndSpecialChar(ContractType):
    if pd.isnull(ContractType):
        return ContractType
    else:
        # normalize to upper case letters
        ContractType = ContractType.upper()
        # remove special character at the end
        ContractType= re.sub(r"\W+$", "", ContractType)
        # remove all special characters except space and dot
        ContractType= re.sub(r'(\_|\-)', '', ContractType)
        # replace multiple spaces with a single space, also trim spaces on both side
        ContractType = re.sub( '\s+', ' ', ContractType).strip()
        return ContractType
df['ContractType'] = df.ContractType.apply(lambda x: removeEndSpecialChar(x))
```

```
df['ContractType'].unique()
```

```
array([nan, '', 'FULLTIME', 'PARTTIME'], dtype=object)
```

It has removed the special characters, but we still have empty strings, which can be an issue. So lets convert this to the NA, so we can impute it all together.

```
df['ContractType'].replace({'':np.nan}, inplace = True)
```

```
df['ContractType'].unique()
```

```
array([nan, 'FULLTIME', 'PARTTIME'], dtype=object)
```

```
# Imputing Missing Values using mode
df['ContractType'].fillna(df['ContractType'].mode()[0], inplace = True)
print(f'Total Number of missing Values in Company:', df['ContractType'].isnull().sum())
```

```
Total Number of missing Values in Company: 0
```

```
df['ContractType'].unique()
```

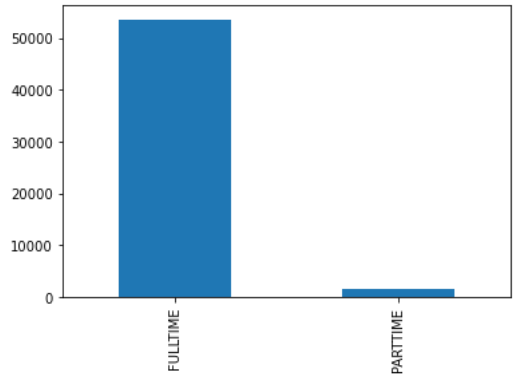
```
array(['FULLTIME', 'PARTTIME'], dtype=object)
```

```
# checking the Value counts
df.ContractType.value_counts()
```

```
FULLTIME    53601
PARTTIME     1568
Name: ContractType, dtype: int64
```

```
# Plottting bar plot
df.ContractType.value_counts().plot.bar()
```

```
<AxesSubplot:>
```



Check ContractTime

```
# Checking unique Values in ContractTime
df['ContractTime'].unique()
```

```
array(['permanent', '-', 'contract', nan, ' '], dtype=object)
```

Even ContractTime has the special characters and empty string. It has lots of missing Values as well which will be imputed using mode, after doing further checks, and replaces.

```
# Lets check, for any other special characters
df['ContractTime'].str.contains(r'([^A-Za-z]|\W+|\.|\_)').any()
```

```
True
```

```
def removeEndSpecialChar(ContractTime):
    if pd.isnull(ContractTime):
        return ContractTime
    else:
        # normalize to upper case letters
        ContractTime = ContractTime.upper()
        # remove special character at the end
        ContractTime= re.sub(r"\W+$", "", ContractTime)
        # remove all special characters except space and dot
        ContractTime= re.sub(r'([^\w\s\.]|_|-)', '', ContractTime)
        # replace multiple spaces with a single space, also trim spaces on both side
        ContractTime = re.sub( '\s+', ' ', ContractTime).strip()
        return ContractTime
df['ContractTime'] = df.ContractTime.apply(lambda x: removeEndSpecialChar(x))
```

```
# Checking the above code has worked
df['ContractTime'].unique()
```

```
array(['PERMANENT', '', 'CONTRACT', nan], dtype=object)
```

```
# We still have empty strings, let us replace all the empty string with NA
df['ContractTime'].replace({'':np.nan}, inplace = True)
```

```
df['ContractTime'].unique()
```

```
array(['PERMANENT', nan, 'CONTRACT'], dtype=object)
```

```
# Imputing Missing Values using mode
df['ContractTime'].fillna(df['ContractTime'].mode()[0], inplace = True)
print(f'Total Number of missing Values in ContractTime:', df['ContractTime'].isnull().sum())
```

```
Total Number of missing Values in ContractTime: 0
```

```
df['ContractTime'].unique()
```

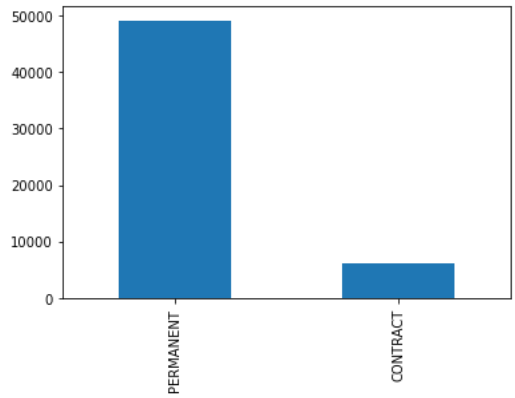
```
array(['PERMANENT', 'CONTRACT'], dtype=object)
```

```
#Checking Value counts for ContractTime
df.ContractTime.value_counts()
```

```
PERMANENT    49080
CONTRACT      6089
Name: ContractTime, dtype: int64
```

```
# Plotting bar plot
df.ContractTime.value_counts().plot.bar()
```

```
<AxesSubplot:>
```



Check OpenDate

```
# Check for any special characters in open date
df['OpenDate'].str.contains(r'([^\w+])').any()
```

```
False
```

```
# Let us do some further checks, and format the open date in format yyyy-mm-dd HH:MM:SS
df['OpenDate'].head(5)
```

```
0    20130708T120000
1    20120130T000000
2    20121221T150000
3    20131208T150000
4    20130302T120000
Name: OpenDate, dtype: object
```

```
# Let us remove T from the Open Date and save it in separate Column to do further formatting.
df['OpenDateNew'] = df['OpenDate'].replace('T','', regex=True)
```

```
df['OpenDateNew'].head(5)
```

```
0    20130708120000
1    20120130000000
2    20121221150000
3    20131208150000
4    20130302120000
Name: OpenDateNew, dtype: object
```

```
# Formatting the OpenDateNew  Column
df['OpenDateNew'] = pd.to_datetime(df['OpenDateNew'], format = '%Y%m%d%H%M%S%f' , errors='coerce')
print(df['OpenDateNew'])
```

```
0       2013-07-08 12:00:00
1       2012-01-30 00:00:00
2       2012-12-21 15:00:00
3       2013-12-08 15:00:00
4       2013-03-02 12:00:00
                   ...
55164   2012-01-23 12:00:00
55165   2013-08-01 15:00:00
55166   2013-01-26 00:00:00
55167   2012-12-23 15:00:00
55168   2012-01-10 15:00:00
Name: OpenDateNew, Length: 55169, dtype: datetime64[ns]
```

```
# Checking if the above New Date column has any missing Values.
print(df['OpenDateNew'].isnull().sum())
```

```
1
```

Lets investigate further, why we have missing Value in new Column as we did not have any missing Values in Original Open Date Column.

```
# Checking the row which has null Value for OpenDateNew.
df[df['OpenDateNew'].isnull()]
```

	Id	Title	Location	Company	ContractType	ContractTime	Category	Salary	OpenDate	CloseDate
		Compliance Technical &	South	MICHAEL PAGE			Accounting			

As we can see above that the month in open date does not seem accurate as it is 23, it could be typo error, So let us see all the opendate for Category Accounting & FinanceJobs, and try to replace the month Value.

```
# Storing all the data with Category Accounting and Finance Jobs in separate data set to investigate
df2 = df.where(df.Category == 'Accounting & Finance Jobs')
```

```
df2.head(5)
```

	Id	Title	Location	Company	ContractType	ContractTime	Category	Salary	OpenDate	CloseDate	Source	OpenDateNew
0	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaT
1	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaT
2	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaT
3	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaT
4	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaT

As we can see above it has lots of nan Values, Let us drop them all to get only Values matching to the filtered Category.

```
df2=df2.dropna(how='all')
```

```
df2.head(40)
```

	Id	Title	Location	Company	ContractType	ContractTime	Category	Salary	Ope
5	19047429.0	Trainee Mortgage Advisor East Midlands	East Midlands	BRITE RECRUITMENT	FULLTIME	PERMANENT	Accounting & Finance Jobs	21000	20130103T0(
51	34479775.0	Credit Controller	The City	UKSTAFFSEARCH	FULLTIME	PERMANENT	Accounting & Finance Jobs	24000	20130227T1:
66	38209887.0	Transaction Services Executive	Bristol	UKSTAFFSEARCH	FULLTIME	PERMANENT	Accounting & Finance Jobs	37500	20120120T0(
67	38209888.0	Audit Executive	Bristol	UKSTAFFSEARCH	FULLTIME	PERMANENT	Accounting & Finance Jobs	34000	20120517T0(
68	38209893.0	Audit Assistant Manager	Bristol	UKSTAFFSEARCH	FULLTIME	PERMANENT	Accounting & Finance Jobs	35000	20120328T1:
78	40880208.0	Compliance Manager	Salisbury	UKSTAFFSEARCH	FULLTIME	PERMANENT	Accounting & Finance Jobs	37500	20130417T1:
92	44325443.0	New Business Sales Various Financial Markets...	London	EXCELSIOR PROFESSIONAL SEARCH LTD	FULLTIME	PERMANENT	Accounting & Finance Jobs	70000	20121210T0(
100	46126154.0	Newly Qualified Rotation Scheme	London	CARNEGIE CONSULTING	FULLTIME	PERMANENT	Accounting & Finance Jobs	-	20120724T1:
105	46352406.0	Personal Tax Advisor	West Sussex	INDIGO 21 LTD	FULLTIME	PERMANENT	Accounting & Finance Jobs	32500	20130916T1:
843	46849661.0	Associate / Senior Associate Real Estate London	London	TRIDENT	FULLTIME	PERMANENT	Accounting & Finance Jobs	70000	20120116T0(
901	48219448.0	Financial Controller	Basingstoke	UKSTAFFSEARCH	FULLTIME	PERMANENT	Accounting & Finance Jobs	42000	20121205T0(
951	49042928.0	Accounts Assistant	Leeds	THE WORKS UK LTD	FULLTIME	PERMANENT	Accounting & Finance Jobs	17500	20121217T1:
1033	50679366.0	Fund Accountant SORP OEIC, Investment Bank	London	WORK PLACE GROUP	FULLTIME	CONTRACT	Accounting & Finance Jobs	57600	20130430T1:
1035	50738105.0	Finance Assistant Italian	Nottinghamshire	LINGUISTS DIRECT	FULLTIME	PERMANENT	Accounting & Finance Jobs	20000	20120327T0(
1082	51128691.0	Mixed Tax Senior	Bristol	UKSTAFFSEARCH	FULLTIME	PERMANENT	Accounting & Finance Jobs	30000	20130411T1:
1083	51128697.0	Tax Development Manager	Gloucestershire	UKSTAFFSEARCH	FULLTIME	PERMANENT	Accounting & Finance Jobs	38500	20120225T0(
1121	51860115.0	Financial Accountant	London	CMC CONSULTING LTD	FULLTIME	PERMANENT	Accounting & Finance Jobs	45000	20130626T1:
1128	52162571.0	Personal Tax Senior Manager	Bath	UKSTAFFSEARCH	FULLTIME	PERMANENT	Accounting & Finance Jobs	55000	20130124T1:
1142	52314212.0	Commercial Finance Analyst	Bedfordshire	CMC CONSULTING LTD	FULLTIME	PERMANENT	Accounting & Finance Jobs	30000	20120408T1:
1144	52410683.0	Tour Accountant	London	UKSTAFFSEARCH	FULLTIME	PERMANENT	Accounting & Finance Jobs	29000	20120110T0(
1156	52634102.0	Paraplanner (parttime)	Surrey	UKSTAFFSEARCH	FULLTIME	PERMANENT	Accounting & Finance Jobs	40000	20130915T1:
1172	52942632.0	Personal Tax Senior	Bristol	UKSTAFFSEARCH	FULLTIME	PERMANENT	Accounting & Finance Jobs	25000	20121201T1:
1208	53636956.0	Commercial Analyst	Berkshire	CMC CONSULTING LTD	FULLTIME	PERMANENT	Accounting & Finance Jobs	52500	20130630T1:

1257	54609038.0	Sales Opportunities	South West London	TFPL	FULLTIME	PERMANENT	Accounting & Finance Jobs	35000	20130625T0(
1258	54609105.0	Sector Research Specialist Financial Services	London	TFPL	FULLTIME	PERMANENT	Accounting & Finance Jobs	47500	20130904T1:
1284	55048084.0	Senior Risk Analyst / Quant (Reinsurance / Ris...	London	HANOVER SEARCH GLOBAL INSURANCE PARTNERS	FULLTIME	PERMANENT	Accounting & Finance Jobs	57500	20120106T1!
1291	55355091.0	Senior Finance Business Partner	Milton Keynes	CMC CONSULTING LTD	FULLTIME	CONTRACT	Accounting & Finance Jobs	90000	20120511T1:
1471	55410118.0	Senior Java Web Developer	Berkshire	SUPPORT SERVICES GROUP	FULLTIME	PERMANENT	Accounting & Finance Jobs	50000	20120303T0(
1505	55666906.0	Senior Internal Auditor	London	CMC CONSULTING LTD	FULLTIME	PERMANENT	Accounting & Finance Jobs	57500	20131213T1:
1536	56086197.0	Treasury Product Manager	London	CMC CONSULTING LTD	FULLTIME	PERMANENT	Accounting & Finance Jobs	65000	20120829T1!
1537	56088808.0	Administrator, Financial Services Investment B...	UK	WORK PLACE GROUP	FULLTIME	CONTRACT	Accounting & Finance Jobs	14640	20131204T1!
1540	56203932.0	Compliance Consultant	London	NOT DISCLOSED	FULLTIME	PERMANENT	Accounting & Finance Jobs	52500	20120506T1!
1561	56281216.0	Secured Loan Underwriter Manchester	Lancashire	JOBG8	FULLTIME	PERMANENT	Accounting & Finance Jobs	21400	20130203T1!
1562	56281226.0	Project Manager Insurance	Surrey	JOBG8	FULLTIME	PERMANENT	Accounting & Finance Jobs	50000	20121019T1:
1563	56281301.0	Project Accountant	West Midlands	JOBG8	FULLTIME	PERMANENT	Accounting & Finance Jobs	40000	20120104T1!
1564	56281394.0	Chinese Speaking PA Finance	London	LANGUAGE RECRUITMENT SERVICES	FULLTIME	PERMANENT	Accounting & Finance Jobs	37500	20130613T0(
1565	56281506.0	Risk Manager	Cheshire	JOBG8	FULLTIME	PERMANENT	Accounting & Finance Jobs	NaN	20130202T0(

As we can see for above Category, most of the jobs are open only for a month or two. So let us replace 23, month with 02, as the job close date is end of March Same year

```
df['OpenDate'] = df['OpenDate'].replace(df['OpenDate'].iloc[51062],'20120202T120000', regex=True)

## Let us Check if the Value is replaced Correctly.
df['OpenDate'].iloc[51062]
```

```
'20120202T120000'
```

Now we are confident that the open date doesnot have any more discrepancies. So Let us Format Original OpenDate Column

```
# removing T from OpenDate Column
df['OpenDate'] = df['OpenDate'].replace('T','', regex=True)


# Formatting the OpenDate Column
df['OpenDate'] = pd.to_datetime(df['OpenDate'], format = '%Y%m%d%H%M%S%f', errors='coerce')
print(df['OpenDate'])
```

```
0       2013-07-08 12:00:00
1       2012-01-30 00:00:00
2       2012-12-21 15:00:00
3       2013-12-08 15:00:00
4       2013-03-02 12:00:00
                ...
55164   2012-01-23 12:00:00
55165   2013-08-01 15:00:00
55166   2013-01-26 00:00:00
55167   2012-12-23 15:00:00
```

```
55168   2012-01-10 15:00:00
Name: OpenDate, Length: 55169, dtype: datetime64[ns]
```

```
## Let us Change the data type  of OpenDate to string
df['OpenDate'] = df['OpenDate'].dt.strftime('%Y-%m-%d %H:%M:%S')
```

```
print(df['OpenDate'].isnull().sum())
```

```
0
```

Checking CloseDate

```
# Checking for any special Characters in CloseDate
df['CloseDate'].str.contains(r'([^\w+])').any()
```

```
/Users/Akshata/opt/anaconda3/lib/python3.8/site-packages/pandas/core/strings.py:2001: UserWarning: This pattern has match groups. T
  return func(self, *args, **kwargs)
False
```

We will do some further investigation and then first create a new Column for CloseDate, to check for any discrepancy, and then format original CloseDate and change into string type.

```
df['CloseDate'].head(5)
```

```
0    20130906T120000
1    20120330T000000
2    20130120T150000
3    20140206T150000
4    20130501T120000
Name: CloseDate, dtype: object
```

```
# Removing T and creating a new Close Date Column
df['CloseDateNew'] = df['CloseDate'].replace('T','', regex=True)
```

```
df['CloseDateNew'].head(5)
```

```
0    20130906120000
1    20120330000000
2    20130120150000
3    20140206150000
4    20130501120000
Name: CloseDateNew, dtype: object
```

```
# Change the date format of new close date column
df['CloseDateNew'] = pd.to_datetime(df['CloseDateNew'], format = '%Y%m%d%H%M%S%f' , errors='coerce')
print(df['CloseDateNew'])
```

```
0       2013-09-06 12:00:00
1       2012-03-30 00:00:00
2       2013-01-20 15:00:00
3       2014-02-06 15:00:00
4       2013-05-01 12:00:00
                ...
55164   2012-02-06 12:00:00
55165   2013-08-31 15:00:00
55166   2013-02-25 00:00:00
55167   2013-02-21 15:00:00
55168   2012-04-09 15:00:00
Name: CloseDateNew, Length: 55169, dtype: datetime64[ns]
```

```
df['CloseDateNew'].head(5)
```

```
0   2013-09-06 12:00:00
1   2012-03-30 00:00:00
2   2013-01-20 15:00:00
3   2014-02-06 15:00:00
4   2013-05-01 12:00:00
Name: CloseDateNew, dtype: datetime64[ns]
```

```
print(df['CloseDateNew'].isnull().sum())
```

```
0
```

No Null Values found, So we can format original close Date

```
# Removing T from CloseDate
df['CloseDate'] = df['CloseDate'].replace('T','', regex=True)
```

```
df['CloseDate'].head(5)
```

```
0    20130906120000
1    20120330000000
2    20130120150000
3    20140206150000
4    20130501120000
Name: CloseDate, dtype: object
```

```
# Formatting CloseDate
df['CloseDate'] = pd.to_datetime(df['CloseDate'], format = '%Y%m%d%H%M%S%f', errors='coerce')
print(df['CloseDate'])
```

```
0       2013-09-06 12:00:00
1       2012-03-30 00:00:00
2       2013-01-20 15:00:00
3       2014-02-06 15:00:00
4       2013-05-01 12:00:00
                ...
55164   2012-02-06 12:00:00
55165   2013-08-31 15:00:00
55166   2013-02-25 00:00:00
55167   2013-02-21 15:00:00
55168   2012-04-09 15:00:00
Name: CloseDate, Length: 55169, dtype: datetime64[ns]
```

```
df['CloseDate'].head(5)
```

```
0   2013-09-06 12:00:00
1   2012-03-30 00:00:00
2   2013-01-20 15:00:00
3   2014-02-06 15:00:00
4   2013-05-01 12:00:00
Name: CloseDate, dtype: datetime64[ns]
```

```
print(df['CloseDate'].isnull().sum())
```

```
0
```

```
# Changing the Data type to string
df['CloseDate'] = df['CloseDate'].dt.strftime('%Y-%m-%d %H:%M:%S')
```

```
df.head(5)
```

	Id	Title	Location	Company	ContractType	ContractTime	Category	Salary	OpenDate	CloseDate	Source
0	12612628	Engineering Systems Analyst	Dorking	GREGORY MARTIN INTERNATIONAL	FULLTIME	PERMANENT	Engineering Jobs	25000	2013-07-08 12:00:00	2013-09-06 12:00:00	cv-library.co.uk
1	12612830	Stress Engineer Glasgow	Glasgow	GREGORY MARTIN INTERNATIONAL	FULLTIME	PERMANENT	Engineering Jobs	30000	2012-01-30 00:00:00	2012-03-30 00:00:00	cv-library.co.uk
2	12612844	Modelling and simulation analyst	Hampshire	GREGORY MARTIN INTERNATIONAL	FULLTIME	PERMANENT	Engineering Jobs	30000	2012-12-21 15:00:00	2013-01-20 15:00:00	cv-library.co.uk
3	12613049	Engineering Systems Analyst /	Surrey	GREGORY MARTIN	FULLTIME	PERMANENT	Engineering	27500	2013-12-08	2014-02-06	cv-

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 55169 entries, 0 to 55168
Data columns (total 13 columns):
 #   Column        Non-Null Count  Dtype
---  ------        --------------  -----
 0   Id            55169 non-null  int64
 1   Title         55169 non-null  object
 2   Location      55169 non-null  object
 3   Company       55169 non-null  object
 4   ContractType  55169 non-null  object
```

```
 5   ContractTime  55169 non-null  object
 6   Category      55169 non-null  object
 7   Salary        53584 non-null  object
 8   OpenDate      55169 non-null  object
 9   CloseDate     55169 non-null  object
 10  Source        55169 non-null  object
 11  OpenDateNew   55168 non-null  datetime64[ns]
 12  CloseDateNew  55169 non-null  datetime64[ns]
dtypes: datetime64[ns](2), int64(1), object(10)
memory usage: 5.5+ MB
```

▾ We added the new Date columns just to ensure there are no any discrepancy, So now we can delete them.

```
# Dropping the Column OpenDateNew, and CloseDateNew
df.drop(df.loc[:, 'OpenDateNew':'CloseDateNew'].columns, axis = 1, inplace=True)
```

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 55169 entries, 0 to 55168
Data columns (total 11 columns):
 #   Column        Non-Null Count  Dtype 
---  ------        --------------  ----- 
 0   Id            55169 non-null  int64 
 1   Title         55169 non-null  object
 2   Location      55169 non-null  object
 3   Company       55169 non-null  object
 4   ContractType  55169 non-null  object
 5   ContractTime  55169 non-null  object
 6   Category      55169 non-null  object
 7   Salary        53584 non-null  object
 8   OpenDate      55169 non-null  object
 9   CloseDate     55169 non-null  object
 10  Source        55169 non-null  object
dtypes: int64(1), object(10)
memory usage: 4.6+ MB
```

▾ Check Location

```
df['Location'].tail(5)
```

```
55164     Salisbury
55165            UK
55166        London
55167       Hackney
55168    Nottingham
Name: Location, dtype: object
```

```
# Check for any Special Characters
df['Location'].str.contains(r'([^a-zA_Z])').any()
```

```
/Users/Akshata/opt/anaconda3/lib/python3.8/site-packages/pandas/core/strings.py:2001: UserWarning: This pattern has match groups. T
  return func(self, *args, **kwargs)
True
```

```
def removeEndSpecialChar(Location):

   # normalize to upper case letters
    Location = Location.upper()
    # remove all characters except alphabets
    Location = re.sub(r'([^A-Z])', '', Location)
    # replace multiple spaces with a single space, also trim spaces on both side
    Location = re.sub( '\s+', ' ', Location).strip()
    return Location
df['Location'] = df.Location.apply(lambda x: removeEndSpecialChar(x))
```

```
df['Location'].str.contains(r'([^A-Z])').any()
```

```
False
```

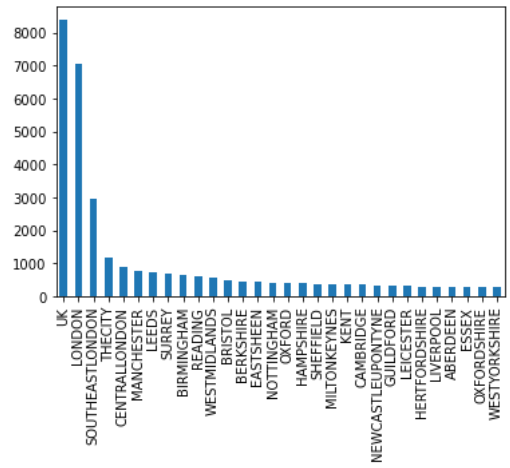
```
# Check Location Value counts
df['Location'].value_counts()
```

```
UK                 8397
LONDON             7048
SOUTHEASTLONDON    2961
THECITY            1184
```

```
CENTRALLONDON       889
                   ...
OXFORDS               2
MANCHASTER            1
LIVEPOOL              1
CEMBRIDGE             1
LEADS                 1
Name: Location, Length: 484, dtype: int64
```

```
#Plotting bar plot
df['Location'].value_counts()[:30].plot.bar()
```

```
<AxesSubplot:>
```



Check Salary

```
# Check Salary for any special characters or alphabets
df['Salary'].str.contains(r'([^A-Za-z]|\s+|\W+|\.|\-|\_)').any()
```

```
True
```

```
# Check What and where are the special characters
SalaryNames = df.Salary[df.Salary.str.contains(pat = r'\-|[a-zA-Z]')== True]
print(SalaryNames)
```

```
17                      -
21                      -
53                      -
71                      -
100                     -
                ...
54869           40500/year
54947    32500 per Annum
54950           30000/year
54967        29500 - 30500
55067           34000/year
Name: Salary, Length: 828, dtype: object
```

As we can see We have lots of special characters and alphabets in Salary column. Lets strip out everything from the special characters, so we are left with only numerical Value that is present, before / or -

```
df['Salary'] = df['Salary'].str.split(r'(-|\D+)').str[0]
df['Salary'].head(20)
```

```
0     25000
1     30000
2     30000
3     27500
4     25000
5     21000
6     37500
7     16000
8     22000
9     18000
10    39500
11    33500
12    39500
13    17280
14    41500
15    31500
16    32500
```

```
17
18    16000
19    26000
Name: Salary, dtype: object
```

```
SalaryCheck= df.Salary[df.Salary.str.contains(pat = r'-|[a-zA-Z]')== True]
print(SalaryCheck)
```

```
Series([], Name: Salary, dtype: object)
```

```
df['Salary'].head(20)
```

```
0     25000
1     30000
2     30000
3     27500
4     25000
5     21000
6     37500
7     16000
8     22000
9     18000
10    39500
11    33500
12    39500
13    17280
14    41500
15    31500
16    32500
17
18    16000
19    26000
Name: Salary, dtype: object
```

```
# We can see there are still empty string in Salary, Lets replace that to NA
df['Salary'].replace({'':np.nan }, inplace = True)
```

```
# Lets Change the salary type to float
df['Salary'] = df['Salary'].astype(float)
print(df.dtypes)
```

```
Id               int64
Title           object
Location        object
Company         object
ContractType    object
ContractTime    object
Category        object
Salary         float64
OpenDate        object
CloseDate       object
Source          object
dtype: object
```

```
df['Salary'].describe()
```

```
count    5.321100e+04
mean     3.425076e+04
std      6.323872e+04
min      0.000000e+00
25%      2.250000e+04
50%      3.086900e+04
75%      4.250000e+04
max      1.000000e+07
Name: Salary, dtype: float64
```

As we have lot of Salary data missing, We will use median to impute the Values, But instead of getting median of Salary column only, We will groupby, Category, ComtractType, Company and ContractTime, to get the median VAlue. Then We will impute it into the missing Values.

```
sumofsalaries = df.groupby(by=['Category', 'ContractType', 'Company', 'ContractTime'])['Salary'].sum()
print(sumofsalaries)
```

```
Category                  ContractType  Company                          ContractTime
Accounting & Finance Jobs  FULLTIME      .MICHAEL PAGE FINANCIAL SERVICES  PERMANENT       122500.0
                                         1ST 4 FX                         PERMANENT        22500.0
                                         1ST CHOICE RECRUITMENT           PERMANENT        18000.0
                                         1ST EXECUTIVE LTD                PERMANENT       171850.0
                                         2020 TECHNOLOGY                  PERMANENT        57500.0
                                                                                            ...
Teaching Jobs              PARTTIME      WOLVERHAMPTON YMCA               PERMANENT        14400.0
                                         WOODHILL PRIMARY SCHOOL          PERMANENT        16378.0
                                         WORKING MENS COLLEGE             PERMANENT        23259.0
```

```
                                        WWW.ESSEX EYJOBS.CO.UK            PERMANENT        26496.0
                                        WYKEHAM PRIMARY SCHOOL            CONTRACT          7672.0
Name: Salary, Length: 13428, dtype: float64
```

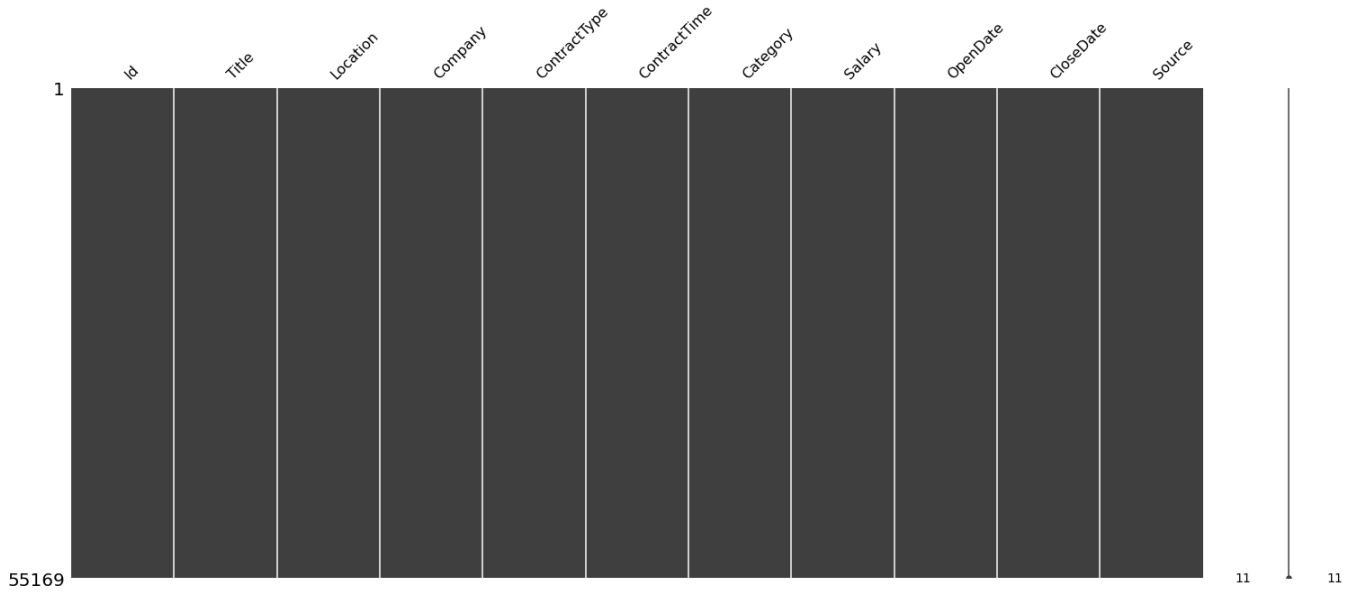
```
# Calculate the median of the above groupby Salary
s = sumofsalaries.median()
print(s)
```

```
42500.0
```

```
# Imputing missing Values
df['Salary'] = df['Salary'].fillna(s)
```

```
# Lets Check any more null Values
msno.matrix(df)
```

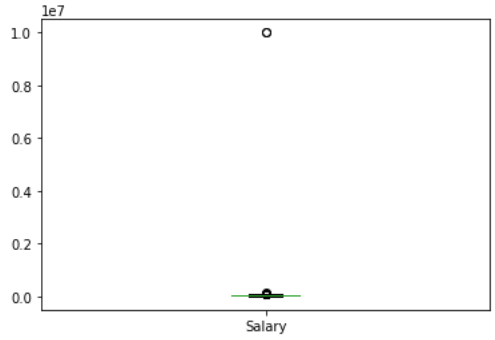
```
<AxesSubplot:>
```



- As now we can see all the data is clean and free of null Values, Lets plot a boxplot to check any outliers in Salary.

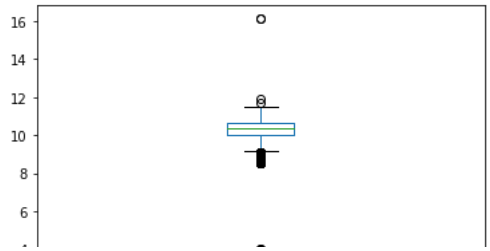
```
df['Salary'].plot.box()
```

```
<AxesSubplot:>
```




```
# To get a better view lets take log as the Values for Salaries are very high and then plot it.
np.log(df['Salary']).plot.box()
```

```
/Users/Akshata/opt/anaconda3/lib/python3.8/site-packages/pandas/core/series.py:726: RuntimeWarning: divide by zero encountered in l
  result = getattr(ufunc, method)(*inputs, **kwargs)
<AxesSubplot:>
```



In the above box plot we can see some outliers, that are very high and very low. Lets check what exactly they are. Our Maximum Value was 10000000.0, and min was 0.0, It is very weird to have Salary Value as 0.0, Lets investigate this further.

```
(df['Salary'] == 10000000.0).sum()
```

```
2
```

```
# We can see we have max Value for two rows,, Lets check What those rows are.
df[(df['Salary']== 10000000.0)]
```

	Id	Title	Location	Company	ContractType	ContractTime	Category	Salary	OpenDate	CloseDate
26971	69181134	Office Technical Administrator	CHESHIRE	GROUP CYTEK	FULLTIME	PERMANENT	Engineering Jobs	10000000.0	2013-03-25 12:00:00	2013-04-24 12:00:00
		New Business							2013-11-	2014-02-

This does not seem quite correct, so we will assume that it is 100000 and not 10million. They look like high paid jobs but 10 million is quite a big number. So we will replace those Values with 100000

```
# Replacing Values
df['Salary'] = np.where(df['Salary'] == 10000000.0, 100000.0, df['Salary'])
```

```
# Now lets check the lower outliers.
(df['Salary'] < 1000.0).sum()
```

```
828
```

```
# Lest check if we have anymore less than 5000
(df['Salary'] < 5000.0).sum()
```

```
828
```

```
# So we have 828 Salary below 1000, lets check few rows
df[(df['Salary'] < 1000.0)]
```

	Id	Title	Location	Company	ContractType	ContractTime	Category	Salary	OpenDate	Close
1354	55408278	Software Engineer, C++, MFC, STL ****k	WESTSUSSEX	SPECTRUM IT RECRUITMENT	FULLTIME	PERMANENT	IT Jobs	0.0	2013-08-24 15:00:00	201 15:
		Group IS							15:00:00	15:
		Cisco								

As we can see above, it doesnot make sense, that the jobs above, most of which are permanent IT jobs and Engineering Jobs, salary is advertised as 0.0 or less than 1000. As the number of rows are quite a few, it will affect the data, So we will replace all the Values, that are less than 1000 with the median above

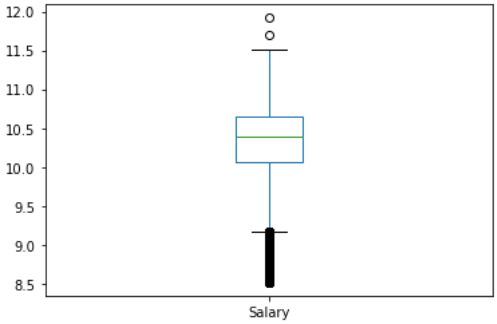
```
df['Salary'] = np.where(df['Salary'] < 1000, s, df['Salary'])
```

```
## Checking for min and max
df['Salary'].describe()
```

```
count     55169.000000
mean      34822.459171
std       15568.009090
min        5000.000000
25%       23500.000000
50%       32500.000000
75%       42500.000000
max      150000.000000
Name: Salary, dtype: float64
```

```
# Lets plot the boxplot
np.log(df['Salary']).plot.box()
```

```
<AxesSubplot:>
```



This looks much better.

+ Code + Text

```
# Check that data info before saving the output
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 55169 entries, 0 to 55168
Data columns (total 11 columns):
 #   Column        Non-Null Count  Dtype
---  ------        --------------  -----
 0   Id            55169 non-null  int64
 1   Title         55169 non-null  object
 2   Location      55169 non-null  object
 3   Company       55169 non-null  object
 4   ContractType  55169 non-null  object
 5   ContractTime  55169 non-null  object
 6   Category      55169 non-null  object
 7   Salary        55169 non-null  float64
 8   OpenDate      55169 non-null  object
 9   CloseDate     55169 non-null  object
 10  Source        55169 non-null  object
dtypes: float64(1), int64(1), object(9)
memory usage: 4.6+ MB
```

Saving data

Save the cleaned data

```
# code to save output data
df.to_csv('dataset1_solution.csv',index=False)
```

Summary

Give a short summary and anything you would like to talk about assessment 2 part 1 here.

The data had lots of issues, that need to dealt with.

We need to perform removing of special characters. Then imputing the data missing Values using mode and median. There were lots of inconsistency in using name for company which was cleaned.

All the Categorical data which had lots of missing Values, such as company, ContractType, ContractTime were imputed using mode, while Salary was imputed by grouping Company, ContractType, ContractTime and then finding median to replace the Values.

Median was chosen over mean , as we had lots of missing Values, and mean is very sensitive to outliers as well.

Two rows were with 10 million Value for Salary which did not seem alright. This Values need to be replaced with appropriate Values, So assuming that it should be 100000 after discussing with stakeholders, I replaced the Values to 100000.

There were lots of data that was below 1000 for salary, which was for IT and Engineering jobs, which also didnot seem correct. So Values below 1000 are replaced by median Value.

The Salary data type was changed to float, Id is integer, and rest all attributes are string.

Double-click (or enter) to edit

```
# Lets check, for any other special characters
df['Category'].str.contains(r'([^A-Za-z]|\W+|\.|\_)').any()
```

```
/Users/Akshata/opt/anaconda3/lib/python3.8/site-packages/pandas/core/strings.py:2001: UserWarning: This pattern has match groups. T
  return func(self, *args, **kwargs)
True
```

```
def removeEndSpecialChar(Category):

       # normalize to upper case letters
        Category = Category.upper()
        # remove all characters except alphabets
        Category = re.sub(r'([^A-Z])', '', Category)
        # replace multiple spaces with a single space, also trim spaces on both side
        Category = re.sub( '\s+', ' ', Category).strip()
        return Category
df['Category'] = df.Category.apply(lambda x: removeEndSpecialChar(x))
```

```
df['Category'].str.contains(r'([^A-Za-z]|\W+|\.|\_)').any()
```

```
False
```

```
df['Category']
```

```
0              ENGINEERINGJOBS
1              ENGINEERINGJOBS
2              ENGINEERINGJOBS
3              ENGINEERINGJOBS
4              ENGINEERINGJOBS
                  ...
55164            TEACHINGJOBS
55165    ACCOUNTINGFINANCEJOBS
55166    ACCOUNTINGFINANCEJOBS
55167                   ITJOBS
55168    HEALTHCARENURSINGJOBS
Name: Category, Length: 55169, dtype: object
```