# Raw data to clean data conversion using Python EDA

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

In [2]:
```
emp = pd.read_excel(r'C:\Users\Gopi Reddy\Downloads\rawdata.xlsx') # it will rea
```

In [3]:
```
emp # it will shows the total raw data
```

Out[3]:

| | Name | Domain | Age | Location | Salary | Exp |
|---|---|---|---|---|---|---|
| **0** | Mike | Datascience#$ | 34 years | Mumbai | 5^00#0 | 2+ |
| **1** | Teddy^ | Testing | 45' yr | Bangalore | 10%%000 | <3 |
| **2** | Uma#r | Dataanalyst^^# | NaN | NaN | 1$5%000 | 4> yrs |
| **3** | Jane | Ana^^lytics | NaN | Hyderbad | 2000^0 | NaN |
| **4** | Uttam* | Statistics | 67-yr | NaN | 30000- | 5+ year |
| **5** | Kim | NLP | 55yr | Delhi | 6000^$0 | 10+ |

In [4]:
```
emp.columns # it will shows column or attribute names
```

Out[4]:
```
Index(['Name', 'Domain', 'Age', 'Location', 'Salary', 'Exp'], dtype='object')
```

In [5]:
```
emp.shape # it will show the dimensions
```

Out[5]:
```
(6, 6)
```

In [6]:
```
emp.head() # it will show the first five rows
```

Out[6]:

| | Name | Domain | Age | Location | Salary | Exp |
|---|---|---|---|---|---|---|
| **0** | Mike | Datascience#$ | 34 years | Mumbai | 5^00#0 | 2+ |
| **1** | Teddy^ | Testing | 45' yr | Bangalore | 10%%000 | <3 |
| **2** | Uma#r | Dataanalyst^^# | NaN | NaN | 1$5%000 | 4> yrs |
| **3** | Jane | Ana^^lytics | NaN | Hyderbad | 2000^0 | NaN |
| **4** | Uttam* | Statistics | 67-yr | NaN | 30000- | 5+ year |

In [7]:
```
emp.tail() # it will show the last five rows
```

Out[7]:

| | Name | Domain | Age | Location | Salary | Exp |
|---|---|---|---|---|---|---|
| **1** | Teddy^ | Testing | 45' yr | Bangalore | 10%%000 | <3 |
| **2** | Uma#r | Dataanalyst^^# | NaN | NaN | 1$5%000 | 4> yrs |
| **3** | Jane | Ana^^lytics | NaN | Hyderbad | 2000^0 | NaN |
| **4** | Uttam* | Statistics | 67-yr | NaN | 30000- | 5+ year |
| **5** | Kim | NLP | 55yr | Delhi | 6000^$0 | 10+ |

In [8]:
```
emp.info() # it will shows the information about the data
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 6 entries, 0 to 5
Data columns (total 6 columns):
 #   Column    Non-Null Count  Dtype 
---  ------    --------------  ----- 
 0   Name      6 non-null      object
 1   Domain    6 non-null      object
 2   Age       4 non-null      object
 3   Location  4 non-null      object
 4   Salary    6 non-null      object
 5   Exp       5 non-null      object
dtypes: object(6)
memory usage: 420.0+ bytes
```

In [9]:
```
emp
```

Out[9]:

| | Name | Domain | Age | Location | Salary | Exp |
|---|---|---|---|---|---|---|
| **0** | Mike | Datascience#$ | 34 years | Mumbai | 5^00#0 | 2+ |
| **1** | Teddy^ | Testing | 45' yr | Bangalore | 10%%000 | <3 |
| **2** | Uma#r | Dataanalyst^^# | NaN | NaN | 1$5%000 | 4> yrs |
| **3** | Jane | Ana^^lytics | NaN | Hyderbad | 2000^0 | NaN |
| **4** | Uttam* | Statistics | 67-yr | NaN | 30000- | 5+ year |
| **5** | Kim | NLP | 55yr | Delhi | 6000^$0 | 10+ |

In [10]:
```
emp['Domain'] # it will shows detail about the Domain column
```

Out[10]:
```
0     Datascience#$
1           Testing
2    Dataanalyst^^#
3       Ana^^lytics
4        Statistics
5               NLP
Name: Domain, dtype: object
```

In [11]:
```
emp.isnull() # it will detects a missing value.
```

Out[11]:

| | Name | Domain | Age | Location | Salary | Exp |
|---|---|---|---|---|---|---|
| **0** | False | False | False | False | False | False |
| **1** | False | False | False | False | False | False |
| **2** | False | False | True | True | False | False |
| **3** | False | False | True | False | False | True |
| **4** | False | False | False | True | False | False |
| **5** | False | False | False | False | False | False |

In [12]:
```
emp.isnull().sum() # it shows missing value information
```

Out[12]:
```
Name        0
Domain      0
Age         2
Location    2
Salary      0
Exp         1
dtype: int64
```

# Data cleaning

In [13]:
```
emp['Name']  # it will shows info about the name column
```

Out[13]:
```
0      Mike
1    Teddy^
2     Uma#r
3      Jane
4    Uttam*
5       Kim
Name: Name, dtype: object
```

In [14]:
```
emp['Name'] = emp['Name'].str.replace(r'\W','',regex=True)
```

In [15]:
```
emp['Name']
```

Out[15]:
```
0     Mike
1    Teddy
2     Umar
3     Jane
4    Uttam
5      Kim
Name: Name, dtype: object
```

In [16]:
```
emp['Domain'] = emp['Domain'].str.replace(r'\W','',regex=True)
```

In [17]:
```
emp['Domain']
```

Out[17]:
```
0    Datascience
1        Testing
2    Dataanalyst
3      Analytics
4     Statistics
5            NLP
Name: Domain, dtype: object
```

In [18]:
```
emp
```

Out[18]:

| | Name | Domain | Age | Location | Salary | Exp |
|---|---|---|---|---|---|---|
| **0** | Mike | Datascience | 34 years | Mumbai | 5^00#0 | 2+ |
| **1** | Teddy | Testing | 45' yr | Bangalore | 10%%000 | <3 |
| **2** | Umar | Dataanalyst | NaN | NaN | 1$5%000 | 4> yrs |
| **3** | Jane | Analytics | NaN | Hyderbad | 2000^0 | NaN |
| **4** | Uttam | Statistics | 67-yr | NaN | 30000- | 5+ year |
| **5** | Kim | NLP | 55yr | Delhi | 6000^$0 | 10+ |

In [19]:
```
emp['Age'] = emp['Age'].str.replace(r'\W','',regex=True)
```

In [20]:
```
emp['Age']
```

Out[20]:
```
0    34years
1       45yr
2        NaN
3        NaN
4       67yr
5       55yr
Name: Age, dtype: object
```

In [21]:
```
emp['Age'] = emp['Age'].str.extract(r'(\d)')
```

In [22]:
```
emp['Age']
```

Out[22]:
```
0      3
1      4
2    NaN
3    NaN
4      6
5      5
Name: Age, dtype: object
```

In [23]:
```
emp
```

Out[23]:

| | Name | Domain | Age | Location | Salary | Exp |
|---|---|---|---|---|---|---|
| **0** | Mike | Datascience | 3 | Mumbai | 5^00#0 | 2+ |
| **1** | Teddy | Testing | 4 | Bangalore | 10%%000 | <3 |
| **2** | Umar | Dataanalyst | NaN | NaN | 1$5%000 | 4> yrs |
| **3** | Jane | Analytics | NaN | Hyderbad | 2000^0 | NaN |
| **4** | Uttam | Statistics | 6 | NaN | 30000- | 5+ year |
| **5** | Kim | NLP | 5 | Delhi | 6000^$0 | 10+ |

In [24]:
```
emp['Location'] = emp['Location'].str.replace(r'\W','',regex=True)
```

In [25]:
```
emp['Location']
```

Out[25]:
```
0       Mumbai
1    Bangalore
2          NaN
3     Hyderbad
4          NaN
5        Delhi
Name: Location, dtype: object
```

In [26]:
```
emp['Salary'] = emp['Salary'].str.replace(r'\W','',regex=True)
```

In [27]:
```
emp['Salary']
```

Out[27]:
```
0     5000
1    10000
2    15000
3    20000
4    30000
5    60000
Name: Salary, dtype: object
```

In [28]:
```
emp
```

Out[28]:

| | Name | Domain | Age | Location | Salary | Exp |
|---|---|---|---|---|---|---|
| **0** | Mike | Datascience | 3 | Mumbai | 5000 | 2+ |
| **1** | Teddy | Testing | 4 | Bangalore | 10000 | <3 |
| **2** | Umar | Dataanalyst | NaN | NaN | 15000 | 4> yrs |
| **3** | Jane | Analytics | NaN | Hyderbad | 20000 | NaN |
| **4** | Uttam | Statistics | 6 | NaN | 30000 | 5+ year |
| **5** | Kim | NLP | 5 | Delhi | 60000 | 10+ |

In [29]:
```
emp['Exp'] = emp['Exp'].str.replace(r'\W','',regex=True)
```

In [30]:
```
emp['Exp']
```

Out[30]:
```
0        2
1        3
2     4yrs
3      NaN
4    5year
5       10
Name: Exp, dtype: object
```

In [31]:
```
emp['Exp'] = emp['Exp'].str.extract(r'(\d+)')
```

In [32]:
```
emp['Exp']
```

Out[32]:
```
0      2
1      3
2      4
3    NaN
4      5
5     10
Name: Exp, dtype: object
```

In [33]:
```
emp
```

Out[33]:

| | Name | Domain | Age | Location | Salary | Exp |
|---|---|---|---|---|---|---|
| **0** | Mike | Datascience | 3 | Mumbai | 5000 | 2 |
| **1** | Teddy | Testing | 4 | Bangalore | 10000 | 3 |
| **2** | Umar | Dataanalyst | NaN | NaN | 15000 | 4 |
| **3** | Jane | Analytics | NaN | Hyderbad | 20000 | NaN |
| **4** | Uttam | Statistics | 6 | NaN | 30000 | 5 |
| **5** | Kim | NLP | 5 | Delhi | 60000 | 10 |

In [34]:
```
clean_data = emp.copy()
```

In [35]:
```
clean_data
```

Out[35]:

| | Name | Domain | Age | Location | Salary | Exp |
|---|---|---|---|---|---|---|
| **0** | Mike | Datascience | 3 | Mumbai | 5000 | 2 |
| **1** | Teddy | Testing | 4 | Bangalore | 10000 | 3 |
| **2** | Umar | Dataanalyst | NaN | NaN | 15000 | 4 |
| **3** | Jane | Analytics | NaN | Hyderbad | 20000 | NaN |
| **4** | Uttam | Statistics | 6 | NaN | 30000 | 5 |
| **5** | Kim | NLP | 5 | Delhi | 60000 | 10 |

## till now we have raw data we use regex to clean the data and removed all noise characted from the dataset

## you can also work in same things in sql query as well

# Missing values treatment for numerical data

In [36]:
```
clean_data
```

Out[36]:

| | Name | Domain | Age | Location | Salary | Exp |
|---|---|---|---|---|---|---|
| **0** | Mike | Datascience | 3 | Mumbai | 5000 | 2 |
| **1** | Teddy | Testing | 4 | Bangalore | 10000 | 3 |
| **2** | Umar | Dataanalyst | NaN | NaN | 15000 | 4 |
| **3** | Jane | Analytics | NaN | Hyderbad | 20000 | NaN |
| **4** | Uttam | Statistics | 6 | NaN | 30000 | 5 |
| **5** | Kim | NLP | 5 | Delhi | 60000 | 10 |

In [37]:
```
clean_data['Age']
```

Out[37]:
```
0      3
1      4
2    NaN
3    NaN
4      6
5      5
Name: Age, dtype: object
```

In [38]:
```
import numpy as np
```

In [39]:
```
clean_data['Age'] = clean_data['Age'].fillna(np.mean(pd.to_numeric(clean_data['A
```

In [40]:
```
clean_data['Age']
```

Out[40]:
```
0      3
1      4
2    4.5
3    4.5
4      6
5      5
Name: Age, dtype: object
```

In [41]:
```
clean_data['Exp']
```

Out[41]:
```
0      2
1      3
2      4
3    NaN
4      5
5     10
Name: Exp, dtype: object
```

In [42]:
```
clean_data['Exp'] = clean_data['Exp'].fillna(np.mean(pd.to_numeric(clean_data['E
```

In [43]:
```
clean_data['Exp']
```

Out[43]:
```
0      2
1      3
2      4
3    4.8
4      5
5     10
Name: Exp, dtype: object
```

In [44]:
```
clean_data
```

Out[44]:

| | Name | Domain | Age | Location | Salary | Exp |
|---|---|---|---|---|---|---|
| **0** | Mike | Datascience | 3 | Mumbai | 5000 | 2 |
| **1** | Teddy | Testing | 4 | Bangalore | 10000 | 3 |
| **2** | Umar | Dataanalyst | 4.5 | NaN | 15000 | 4 |
| **3** | Jane | Analytics | 4.5 | Hyderbad | 20000 | 4.8 |
| **4** | Uttam | Statistics | 6 | NaN | 30000 | 5 |
| **5** | Kim | NLP | 5 | Delhi | 60000 | 10 |

In [45]:
```
clean_data['Location'].isnull().sum()
```

Out[45]:
```
2
```

In [46]:
```
clean_data['Location']
```

Out[46]:
```
0       Mumbai
1    Bangalore
2          NaN
3     Hyderbad
4          NaN
5        Delhi
Name: Location, dtype: object
```

In [47]:
```
clean_data['Location'] = clean_data['Location'].fillna(clean_data['Location'].mc
```

In [48]:
```
clean_data['Location']
```

Out[48]:
```
0       Mumbai
1    Bangalore
2    Bangalore
3     Hyderbad
4    Bangalore
5        Delhi
Name: Location, dtype: object
```

In [49]:
```
clean_data
```

Out[49]:

| | Name | Domain | Age | Location | Salary | Exp |
|---|---|---|---|---|---|---|
| **0** | Mike | Datascience | 3 | Mumbai | 5000 | 2 |
| **1** | Teddy | Testing | 4 | Bangalore | 10000 | 3 |
| **2** | Umar | Dataanalyst | 4.5 | Bangalore | 15000 | 4 |
| **3** | Jane | Analytics | 4.5 | Hyderbad | 20000 | 4.8 |
| **4** | Uttam | Statistics | 6 | Bangalore | 30000 | 5 |
| **5** | Kim | NLP | 5 | Delhi | 60000 | 10 |

In [50]:
```
clean_data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 6 entries, 0 to 5
Data columns (total 6 columns):
 #   Column    Non-Null Count  Dtype 
---  ------    --------------  ----- 
 0   Name      6 non-null      object
 1   Domain    6 non-null      object
 2   Age       6 non-null      object
 3   Location  6 non-null      object
 4   Salary    6 non-null      object
 5   Exp       6 non-null      object
dtypes: object(6)
memory usage: 420.0+ bytes
```

In [51]:
```
clean_data['Age'] = clean_data['Age'].astype(int)
```

In [52]:
```
clean_data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 6 entries, 0 to 5
Data columns (total 6 columns):
 #   Column    Non-Null Count  Dtype 
---  ------    --------------  ----- 
 0   Name      6 non-null      object
 1   Domain    6 non-null      object
 2   Age       6 non-null      int32 
 3   Location  6 non-null      object
 4   Salary    6 non-null      object
 5   Exp       6 non-null      object
dtypes: int32(1), object(5)
memory usage: 396.0+ bytes
```

In [53]:
```
clean_data['Exp'] = clean_data['Exp'].astype(int)
```

In [54]:
```
clean_data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 6 entries, 0 to 5
Data columns (total 6 columns):
 #   Column    Non-Null Count  Dtype 
---  ------    --------------  ----- 
 0   Name      6 non-null      object
 1   Domain    6 non-null      object
 2   Age       6 non-null      int32 
 3   Location  6 non-null      object
 4   Salary    6 non-null      object
 5   Exp       6 non-null      int32 
dtypes: int32(2), object(4)
memory usage: 372.0+ bytes
```

In [55]:
```
clean_data['Salary'] = clean_data['Salary'].astype(int)
```

In [56]:
```
clean_data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 6 entries, 0 to 5
Data columns (total 6 columns):
 #   Column    Non-Null Count  Dtype 
---  ------    --------------  ----- 
 0   Name      6 non-null      object
 1   Domain    6 non-null      object
 2   Age       6 non-null      int32 
 3   Location  6 non-null      object
 4   Salary    6 non-null      int32 
 5   Exp       6 non-null      int32 
dtypes: int32(3), object(3)
memory usage: 348.0+ bytes
```

In [57]:
```
clean_data['Name'] = clean_data['Location'].astype('category')
clean_data['Domain'] = clean_data['Location'].astype('category')
clean_data['Location'] = clean_data['Location'].astype('category')
```

In [58]:
```
clean_data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 6 entries, 0 to 5
Data columns (total 6 columns):
 #   Column    Non-Null Count  Dtype   
---  ------    --------------  -----   
 0   Name      6 non-null      category
 1   Domain    6 non-null      category
 2   Age       6 non-null      int32   
 3   Location  6 non-null      category
 4   Salary    6 non-null      int32   
 5   Exp       6 non-null      int32   
dtypes: category(3), int32(3)
memory usage: 834.0 bytes
```

In [59]:
```
clean_data
```

Out[59]:

| | Name | Domain | Age | Location | Salary | Exp |
|---|---|---|---|---|---|---|
| **0** | Mumbai | Mumbai | 3 | Mumbai | 5000 | 2 |
| **1** | Bangalore | Bangalore | 4 | Bangalore | 10000 | 3 |
| **2** | Bangalore | Bangalore | 4 | Bangalore | 15000 | 4 |
| **3** | Hyderbad | Hyderbad | 4 | Hyderbad | 20000 | 4 |
| **4** | Bangalore | Bangalore | 6 | Bangalore | 30000 | 5 |
| **5** | Delhi | Delhi | 5 | Delhi | 60000 | 10 |

In [60]:
```
clean_data.to_csv('clean_data.csv')
```

In [61]:
```
import os
os.getcwd()
```

Out[61]:
```
'C:\\Users\\Gopi Reddy'
```

In [62]:
```
clean_data
```

Out[62]:

| | Name | Domain | Age | Location | Salary | Exp |
|---|---|---|---|---|---|---|
| **0** | Mumbai | Mumbai | 3 | Mumbai | 5000 | 2 |
| **1** | Bangalore | Bangalore | 4 | Bangalore | 10000 | 3 |
| **2** | Bangalore | Bangalore | 4 | Bangalore | 15000 | 4 |
| **3** | Hyderbad | Hyderbad | 4 | Hyderbad | 20000 | 4 |
| **4** | Bangalore | Bangalore | 6 | Bangalore | 30000 | 5 |
| **5** | Delhi | Delhi | 5 | Delhi | 60000 | 10 |

# EDA Technique Apply

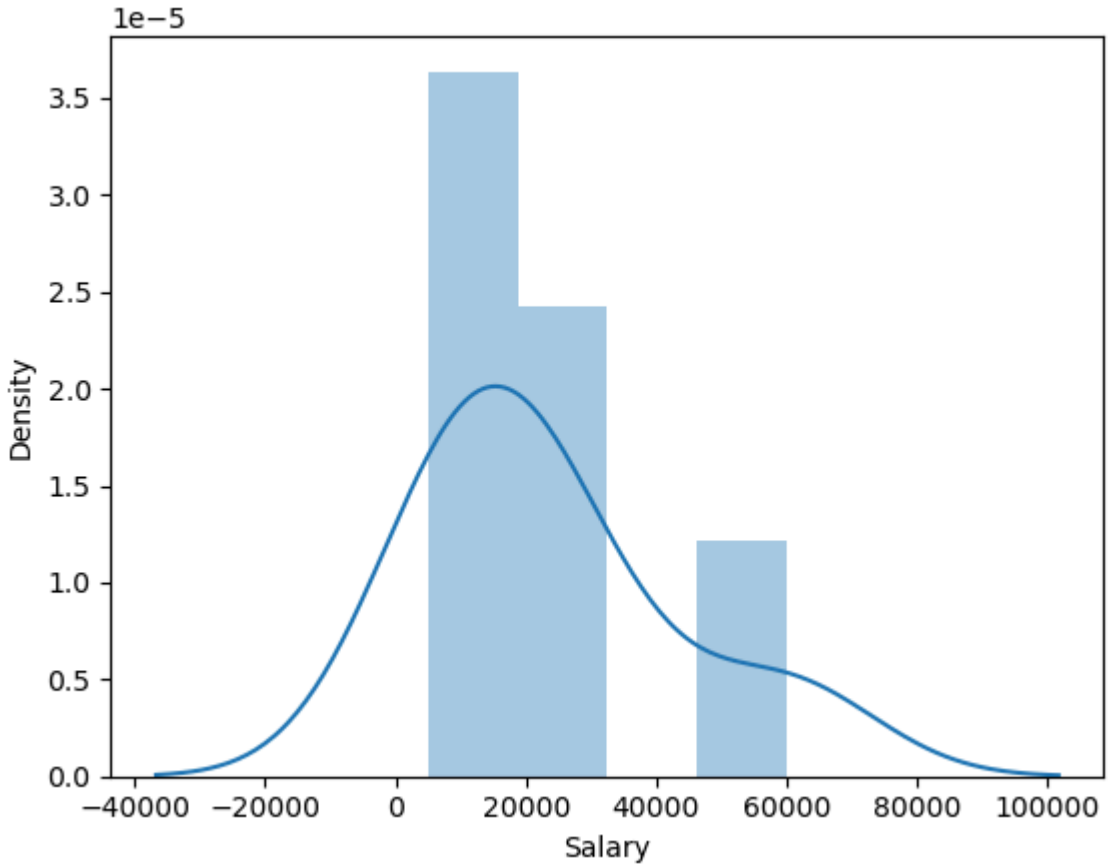
In [63]:
```
import matplotlib.pyplot as plt # data visualization
import seaborn as sns
```

In [64]:
```
import warnings
warnings.filterwarnings('ignore')
```

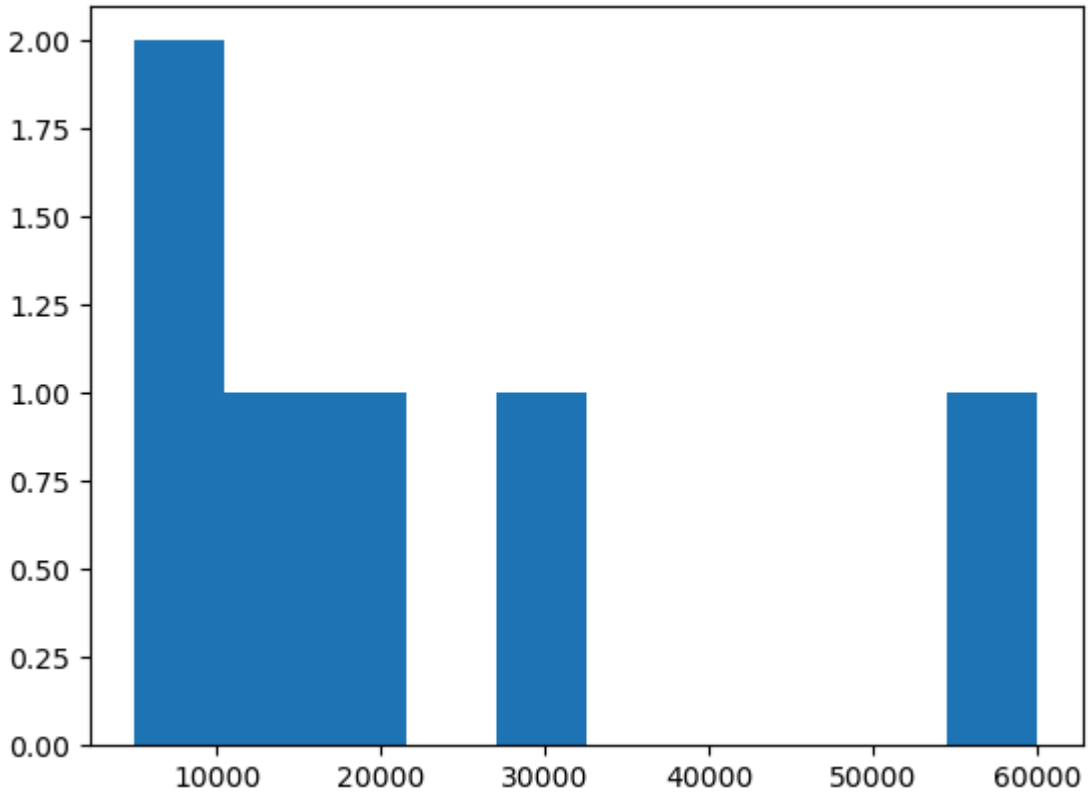
In [65]:
```
clean_data['Salary']
```

Out[65]:
```
0     5000
1    10000
2    15000
3    20000
4    30000
5    60000
Name: Salary, dtype: int32
```

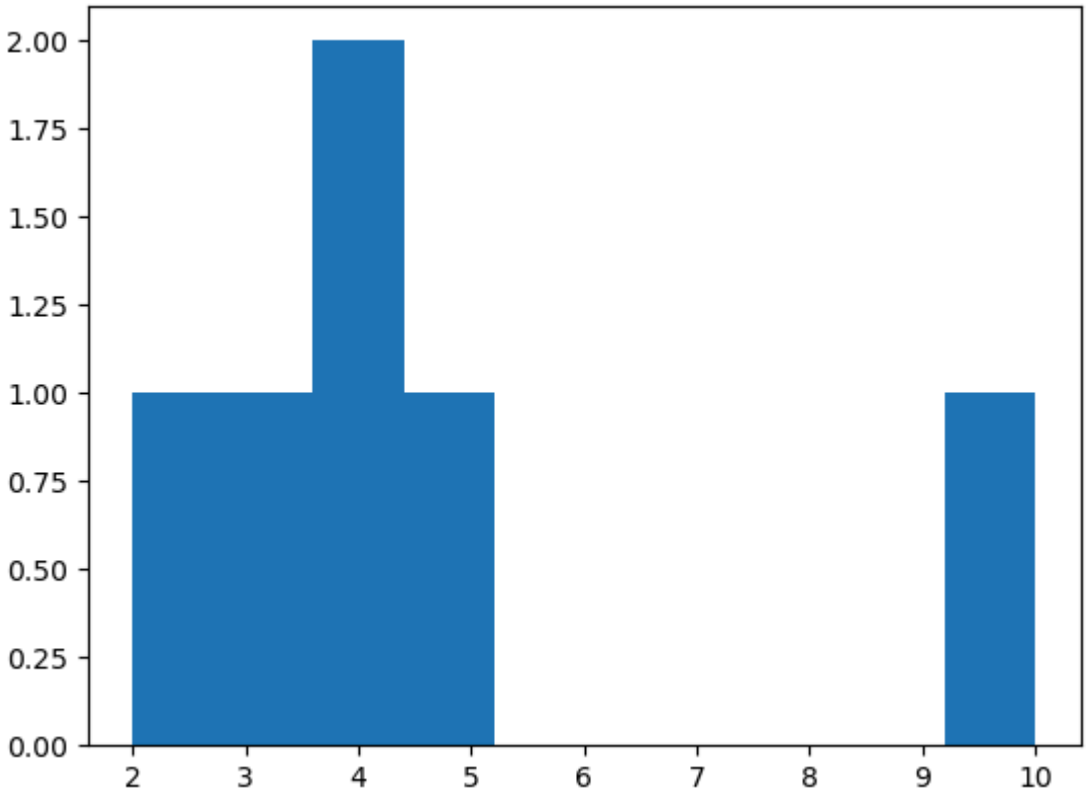
In [66]:
```
vis1 = sns.distplot(clean_data['Salary'])
```



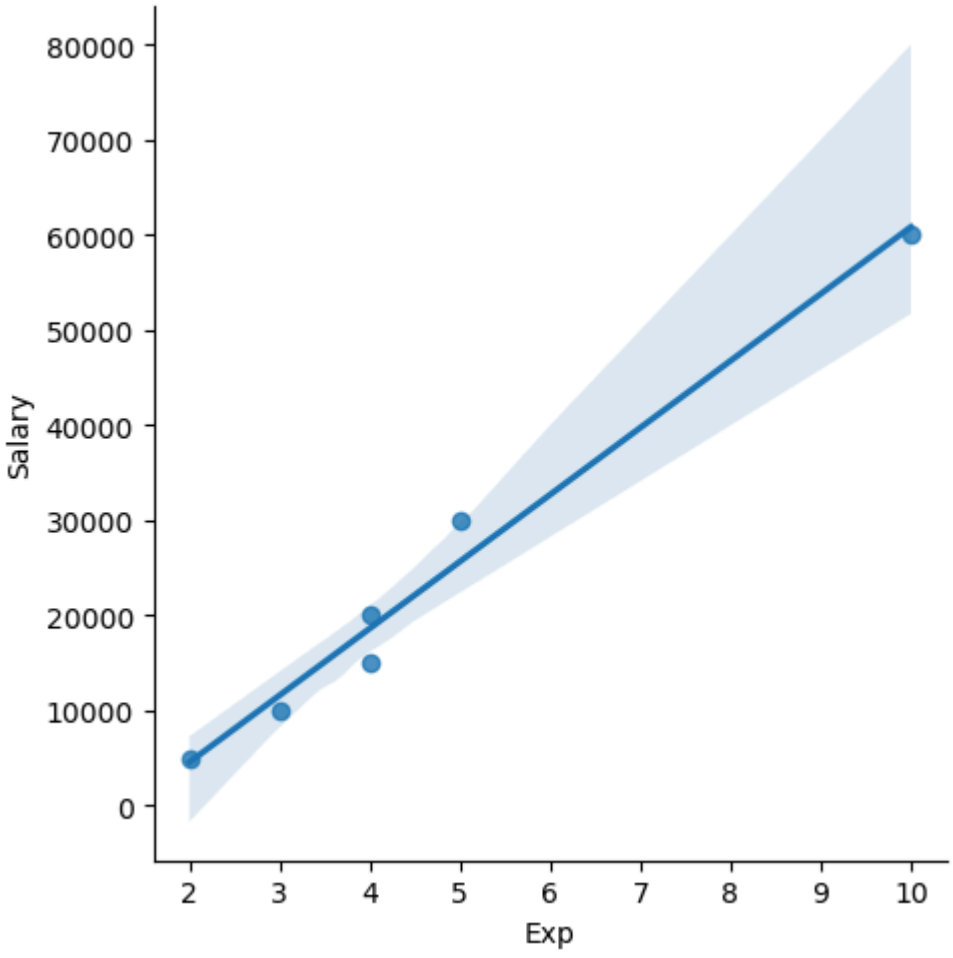

In [67]:
```
vis2 = plt.hist(clean_data['Salary'])
```



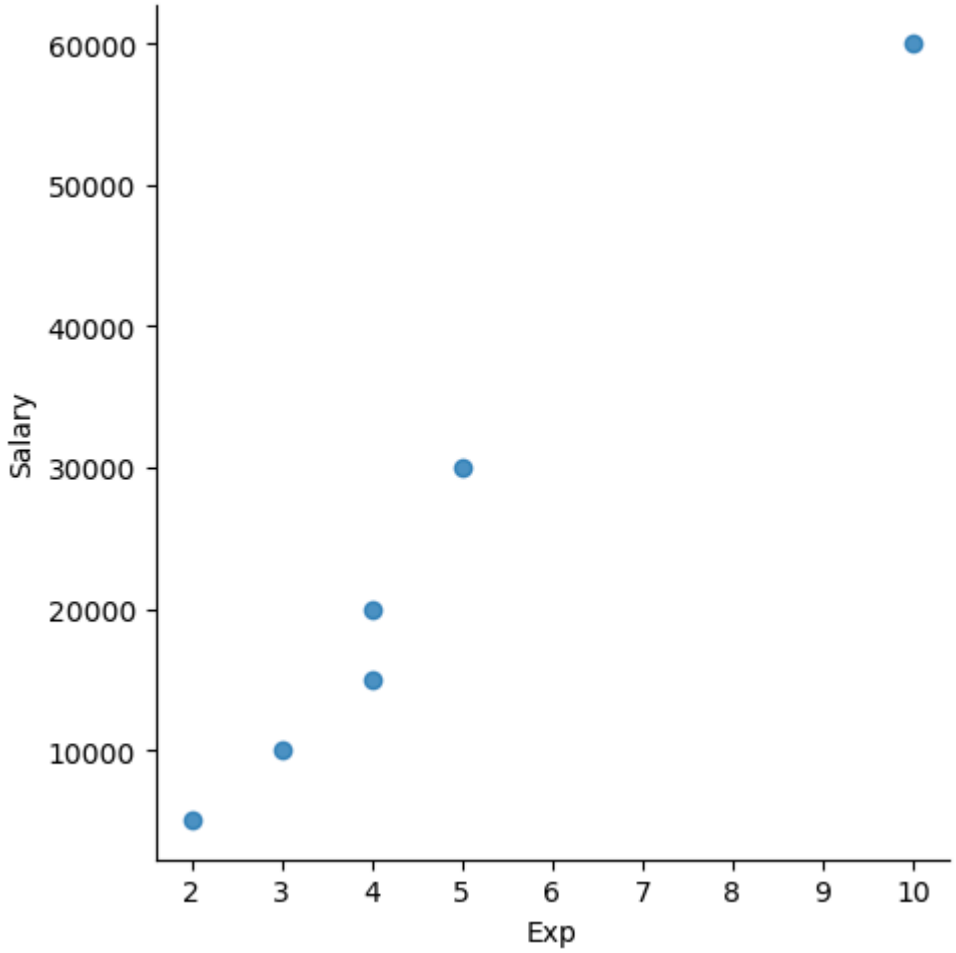

In [68]:
```
vis3 = plt.hist(clean_data['Exp'])
```



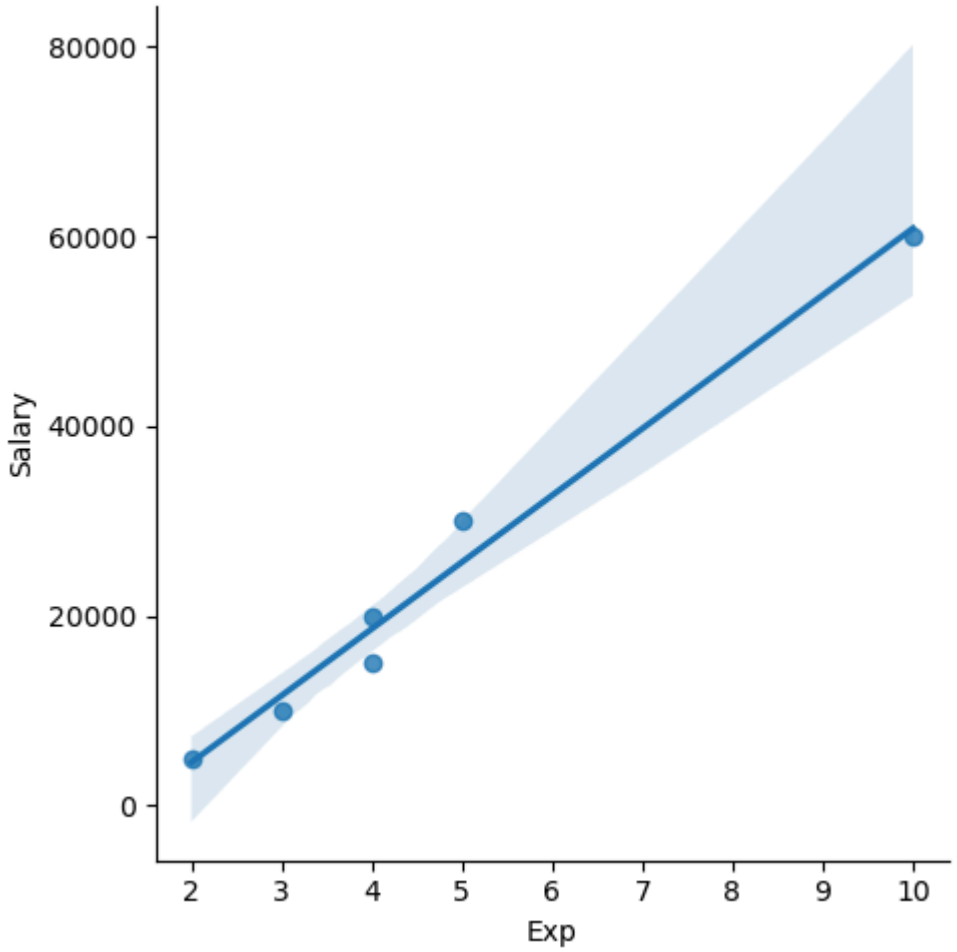

In [69]:
```
vis4 = sns.lmplot(data=clean_data,x= 'Exp',y ='Salary')
```




In [70]:
```
vis5 = sns.lmplot(data=clean_data,x = 'Exp', y= 'Salary',fit_reg = False)
```




In [71]:
```
vis6 = sns.lmplot(data=clean_data,x = 'Exp', y= 'Salary',fit_reg = True)
```




In [72]:
```
clean_data[:]
```

Out[72]:

| | Name | Domain | Age | Location | Salary | Exp |
|---|---|---|---|---|---|---|
| **0** | Mumbai | Mumbai | 3 | Mumbai | 5000 | 2 |
| **1** | Bangalore | Bangalore | 4 | Bangalore | 10000 | 3 |
| **2** | Bangalore | Bangalore | 4 | Bangalore | 15000 | 4 |
| **3** | Hyderbad | Hyderbad | 4 | Hyderbad | 20000 | 4 |
| **4** | Bangalore | Bangalore | 6 | Bangalore | 30000 | 5 |
| **5** | Delhi | Delhi | 5 | Delhi | 60000 | 10 |

In [73]:
```
clean_data[:2]
```

Out[73]:

| | Name | Domain | Age | Location | Salary | Exp |
|---|---|---|---|---|---|---|
| **0** | Mumbai | Mumbai | 3 | Mumbai | 5000 | 2 |
| **1** | Bangalore | Bangalore | 4 | Bangalore | 10000 | 3 |

In [74]:
```
clean_data[2:]
```

Out[74]:

| | Name | Domain | Age | Location | Salary | Exp |
|---|---|---|---|---|---|---|
| **2** | Bangalore | Bangalore | 4 | Bangalore | 15000 | 4 |
| **3** | Hyderbad | Hyderbad | 4 | Hyderbad | 20000 | 4 |
| **4** | Bangalore | Bangalore | 6 | Bangalore | 30000 | 5 |
| **5** | Delhi | Delhi | 5 | Delhi | 60000 | 10 |

In [75]:
```
clean_data[0:1]
```

Out[75]:

| | Name | Domain | Age | Location | Salary | Exp |
|---|---|---|---|---|---|---|
| **0** | Mumbai | Mumbai | 3 | Mumbai | 5000 | 2 |

In [76]:
```
clean_data[0:6:2]
```

Out[76]:

| | Name | Domain | Age | Location | Salary | Exp |
|---|---|---|---|---|---|---|
| **0** | Mumbai | Mumbai | 3 | Mumbai | 5000 | 2 |
| **2** | Bangalore | Bangalore | 4 | Bangalore | 15000 | 4 |
| **4** | Bangalore | Bangalore | 6 | Bangalore | 30000 | 5 |

In [77]:
```
clean_data[::-1]
```

Out[77]:

| | Name | Domain | Age | Location | Salary | Exp |
|---|---|---|---|---|---|---|
| **5** | Delhi | Delhi | 5 | Delhi | 60000 | 10 |
| **4** | Bangalore | Bangalore | 6 | Bangalore | 30000 | 5 |
| **3** | Hyderbad | Hyderbad | 4 | Hyderbad | 20000 | 4 |
| **2** | Bangalore | Bangalore | 4 | Bangalore | 15000 | 4 |
| **1** | Bangalore | Bangalore | 4 | Bangalore | 10000 | 3 |
| **0** | Mumbai | Mumbai | 3 | Mumbai | 5000 | 2 |

In [78]:
```
clean_data.columns
```

Out[78]:
```
Index(['Name', 'Domain', 'Age', 'Location', 'Salary', 'Exp'], dtype='object')
```

In [79]:
```
X_iv = clean_data[['Name','Domain','Age','Location','Exp']]
```

In [80]:
```
X_iv
```

Out[80]:

| | Name | Domain | Age | Location | Exp |
|---|---|---|---|---|---|
| **0** | Mumbai | Mumbai | 3 | Mumbai | 2 |
| **1** | Bangalore | Bangalore | 4 | Bangalore | 3 |
| **2** | Bangalore | Bangalore | 4 | Bangalore | 4 |
| **3** | Hyderbad | Hyderbad | 4 | Hyderbad | 4 |
| **4** | Bangalore | Bangalore | 6 | Bangalore | 5 |
| **5** | Delhi | Delhi | 5 | Delhi | 10 |

In [81]:
```
y_dv = clean_data[['Salary']]
```

In [82]:
```
y_dv
```

Out[82]:

| | Salary |
|---|---|
| **0** | 5000 |
| **1** | 10000 |
| **2** | 15000 |
| **3** | 20000 |
| **4** | 30000 |
| **5** | 60000 |

In [83]:
```
emp
```

Out[83]:

| | Name | Domain | Age | Location | Salary | Exp |
|---|---|---|---|---|---|---|
| **0** | Mike | Datascience | 3 | Mumbai | 5000 | 2 |
| **1** | Teddy | Testing | 4 | Bangalore | 10000 | 3 |
| **2** | Umar | Dataanalyst | NaN | NaN | 15000 | 4 |
| **3** | Jane | Analytics | NaN | Hyderbad | 20000 | NaN |
| **4** | Uttam | Statistics | 6 | NaN | 30000 | 5 |
| **5** | Kim | NLP | 5 | Delhi | 60000 | 10 |

In [84]:
```
clean_data
```

Out[84]:

| | Name | Domain | Age | Location | Salary | Exp |
|---|---|---|---|---|---|---|
| **0** | Mumbai | Mumbai | 3 | Mumbai | 5000 | 2 |
| **1** | Bangalore | Bangalore | 4 | Bangalore | 10000 | 3 |
| **2** | Bangalore | Bangalore | 4 | Bangalore | 15000 | 4 |
| **3** | Hyderbad | Hyderbad | 4 | Hyderbad | 20000 | 4 |
| **4** | Bangalore | Bangalore | 6 | Bangalore | 30000 | 5 |
| **5** | Delhi | Delhi | 5 | Delhi | 60000 | 10 |

In [85]:
```
X_iv
```

Out[85]:

| | Name | Domain | Age | Location | Exp |
|---|---|---|---|---|---|
| **0** | Mumbai | Mumbai | 3 | Mumbai | 2 |
| **1** | Bangalore | Bangalore | 4 | Bangalore | 3 |
| **2** | Bangalore | Bangalore | 4 | Bangalore | 4 |
| **3** | Hyderbad | Hyderbad | 4 | Hyderbad | 4 |
| **4** | Bangalore | Bangalore | 6 | Bangalore | 5 |
| **5** | Delhi | Delhi | 5 | Delhi | 10 |

In [86]:
```
y_dv
```

Out[86]:

| | Salary |
|---|---|
| **0** | 5000 |
| **1** | 10000 |
| **2** | 15000 |
| **3** | 20000 |
| **4** | 30000 |
| **5** | 60000 |

In [87]:
```
clean_data
```

Out[87]:

| | Name | Domain | Age | Location | Salary | Exp |
|---|---|---|---|---|---|---|
| **0** | Mumbai | Mumbai | 3 | Mumbai | 5000 | 2 |
| **1** | Bangalore | Bangalore | 4 | Bangalore | 10000 | 3 |
| **2** | Bangalore | Bangalore | 4 | Bangalore | 15000 | 4 |
| **3** | Hyderbad | Hyderbad | 4 | Hyderbad | 20000 | 4 |
| **4** | Bangalore | Bangalore | 6 | Bangalore | 30000 | 5 |
| **5** | Delhi | Delhi | 5 | Delhi | 60000 | 10 |

In [88]:
```
imputation = pd.get_dummies(clean_data)
```

In [89]:
```
imputation=imputation.astype(int)
```

In [90]:
```
imputation
```

Out[90]:

| | Age | Salary | Exp | Name_Bangalore | Name_Delhi | Name_Hyderbad | Name_Mumbai | I |
|---|---|---|---|---|---|---|---|---|
| **0** | 3 | 5000 | 2 | 0 | 0 | 0 | 1 | |
| **1** | 4 | 10000 | 3 | 1 | 0 | 0 | 0 | |
| **2** | 4 | 15000 | 4 | 1 | 0 | 0 | 0 | |
| **3** | 4 | 20000 | 4 | 0 | 0 | 1 | 0 | |
| **4** | 6 | 30000 | 5 | 1 | 0 | 0 | 0 | |
| **5** | 5 | 60000 | 10 | 0 | 1 | 0 | 0 | |

# Raw data with lot of regex, missing and uncleaned data is there

**Regex, clean**

**Fill missing numerical & cateigroical**

**Clean_dataset ( data cleaning)**

**Outlier treatement, Univatie Analysis, Bivariate Analysis and Corelation**

**Split the data into x_i.v & y_dv**

**Impute cateogrical data to numerical**

**Eda part complete**

In [ ]: