IMPORTING PANDAS

And Dataset

In [1]:

```
import pandas as pd
```

In [2]:

```
emp = pd.read_excel(r'C:\Users\91939\Desktop\AI&DS\16thAug\Rawdata.xlsx')
```

In [3]:

```
emp
```

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+

Performing basic operations

In [4]:

```
id(emp)
```

Out[4]:

```
1623088886704
```

In [5]:

```
emp.columns
```

Out[5]:

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

In [6]:

```
emp.shape
```

Out[6]:

```
(6, 6)
```

In [7]:

```
emp.head()
```

Out[7]:

	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 [8]:

```
emp.tail()
```

Out[8]:

	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 [9]:

```
emp.info() #returns info about dataframe(non-null count,dtype)
```

```
<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 [10]:

```
emp
```

Out[10]:

	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 [11]:

```
emp.isnull() #returns True if null else False
```

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() #returns no.of null values present
```

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

Data Cleaning or Data Cleansing

In [13]:
```
emp
```

Out[13]:

	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 [14]:
```
emp['Name']
```

Out[14]:
```
0      Mike
1    Teddy^
2     Uma#r
3      Jane
4    Uttam*
5       Kim
Name: Name, dtype: object
```

In [15]:
```
emp['Name'] = emp['Name'].str.replace(r'\W','',regex = True) #cleans data by rep
```

In [16]:
```
emp['Name']
```

Out[16]:
```
0     Mike
1    Teddy
2     Umar
3     Jane
4    Uttam
5      Kim
Name: Name, dtype: object
```

In [17]:
```
emp['Name'] = emp['Name'].str.replace(r'\W','',regex = False)
```

In [18]:
```
emp['Name']
```

Out[18]:

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

In [19]:

```
emp
```

Out[19]:

	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	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 [20]:

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

In [21]:

```
emp['Domain']
```

Out[21]:

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

In [22]:

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

In [23]:

```
emp['Age']
```

Out[23]:

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

In [24]:

```
emp['Age']=emp['Age'].str.extract((r'(\d+)')) #returns only digits by extracting
```

In [25]:

```
emp['Age']
```

Out[25]:

```
0     34
1     45
2    NaN
3    NaN
4     67
5     55
Name: Age, dtype: object
```

In [26]:
```
emp
```

Out[26]:

	Name	Domain	Age	Location	Salary	Exp
0	Mike	Datascience	34	Mumbai	5^00#0	2+
1	Teddy	Testing	45	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	NaN	30000-	5+ year
5	Kim	NLP	55	Delhi	6000^$0	10+

In [27]:
```
emp['Location']=emp['Location'].str.replace(r'\W','',regex = True)
```

In [28]:
```
emp['Location']
```

Out[28]:
```
0       Mumbai
1    Bangalore
2          NaN
3     Hyderbad
4          NaN
5        Delhi
Name: Location, dtype: object
```

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

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

Out[30]:
```
0     5000
1    10000
2    15000
3    20000
4    30000
5    60000
Name: Salary, 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	34	Mumbai	5000	2
1	Teddy	Testing	45	Bangalore	10000	3
2	Umar	Dataanalyst	NaN	NaN	15000	4
3	Jane	Analytics	NaN	Hyderbad	20000	NaN
4	Uttam	Statistics	67	NaN	30000	5
5	Kim	NLP	55	Delhi	60000	10

In [34]:

```
import warnings
warnings.filterwarnings('ignore')
```

In [35]:

```
clean_data = emp.copy()
```

In [36]:

```
clean_data
```

Out[36]:

	Name	Domain	Age	Location	Salary	Exp
0	Mike	Datascience	34	Mumbai	5000	2
1	Teddy	Testing	45	Bangalore	10000	3
2	Umar	Dataanalyst	NaN	NaN	15000	4
3	Jane	Analytics	NaN	Hyderbad	20000	NaN
4	Uttam	Statistics	67	NaN	30000	5
5	Kim	NLP	55	Delhi	60000	10

EDA Techniques

In [37]:

```
clean_data.isnull().sum()
```

Out[37]:

```
Name        0
Domain      0
Age         2
Location    2
Salary      0
Exp         1
dtype: int64
```

In [38]:

```
clean_data['Age']
```

Out[38]:

```
0     34
1     45
2    NaN
3    NaN
4     67
5     55
Name: Age, dtype: object
```

MISSING VALUE TREATMENT

Fill numerical data using mean

In [39]:
```
import numpy as np
```

In [40]:
```
clean_data['Age'] = clean_data['Age'].fillna(np.mean(pd.to_numeric(clean_data['A
```

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

Out[41]:
```
0       34
1       45
2    50.25
3    50.25
4       67
5       55
Name: Age, 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
```

Fill categorical data using mode

In [44]:
```
clean_data['Location'] = clean_data['Location'].fillna(clean_data['Location'].mc
```

In [45]:
```
clean_data['Location']
```

Out[45]:
```
0       Mumbai
1    Bangalore
2    Bangalore
3     Hyderbad
4    Bangalore
5        Delhi
Name: Location, dtype: object
```

In [46]:
```
clean_data
```

Out[46]:

	Name	Domain	Age	Location	Salary	Exp
0	Mike	Datascience	34	Mumbai	5000	2
1	Teddy	Testing	45	Bangalore	10000	3
2	Umar	Dataanalyst	50.25	Bangalore	15000	4
3	Jane	Analytics	50.25	Hyderbad	20000	4.8
4	Uttam	Statistics	67	Bangalore	30000	5
5	Kim	NLP	55	Delhi	60000	10

In [47]:
```
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
```

convert object dtype to int ,category using astype

In [48]:
```
clean_data['Age'] = clean_data['Age'].astype(int)
```

In [49]:
```
clean_data['Age']
```

Out[49]:
```
0    34
1    45
2    50
3    50
4    67
5    55
Name: Age, dtype: int32
```

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      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 [51]:

```
clean_data['Salary'] = clean_data['Salary'].astype(int)
clean_data['Exp'] = clean_data['Exp'].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      int32 
 5   Exp       6 non-null      int32 
dtypes: int32(3), object(3)
memory usage: 348.0+ bytes
```

In [53]:

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

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      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: 866.0 bytes
```

Export to os

In [55]:

```
clean_data.to_csv('clean_data.csv')
```

In [56]:

```
import os  # it displays path to os
os.getcwd()
```

Out[56]:

```
'C:\\Users\\91939'
```

In [57]:

```
clean_data
```

Out[57]:

	Name	Domain	Age	Location	Salary	Exp
0	Mike	Datascience	34	Mumbai	5000	2
1	Teddy	Testing	45	Bangalore	10000	3
2	Umar	Dataanalyst	50	Bangalore	15000	4
3	Jane	Analytics	50	Hyderbad	20000	4
4	Uttam	Statistics	67	Bangalore	30000	5
5	Kim	NLP	55	Delhi	60000	10

EDA Techniques through visualization

In [58]:

```
import matplotlib.pyplot as plt
import seaborn as sns
```

In [59]:

```
import warnings
warnings.filterwarnings('ignore')
```

In [60]:

```
clean_data['Salary']
```

Out[60]:

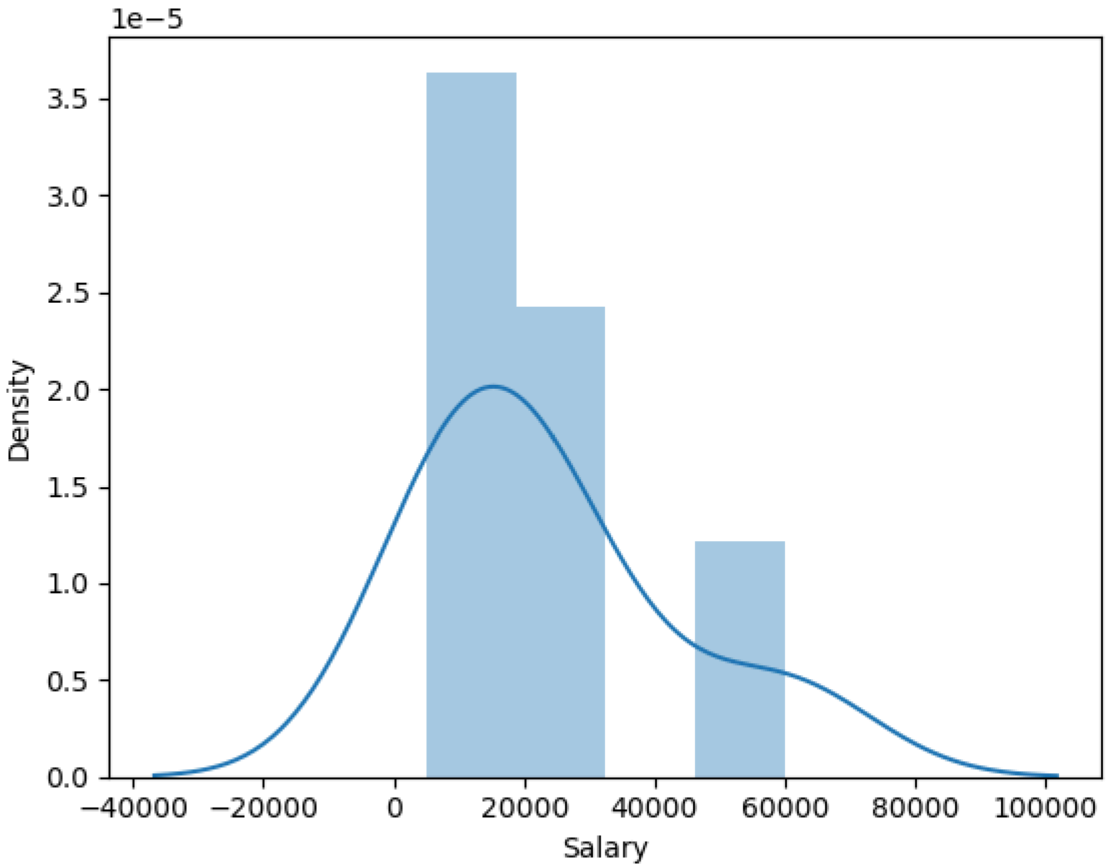
```
0     5000
1    10000
2    15000
3    20000
4    30000
5    60000
Name: Salary, dtype: int32
```

UNIVARIATE ANALYSIS

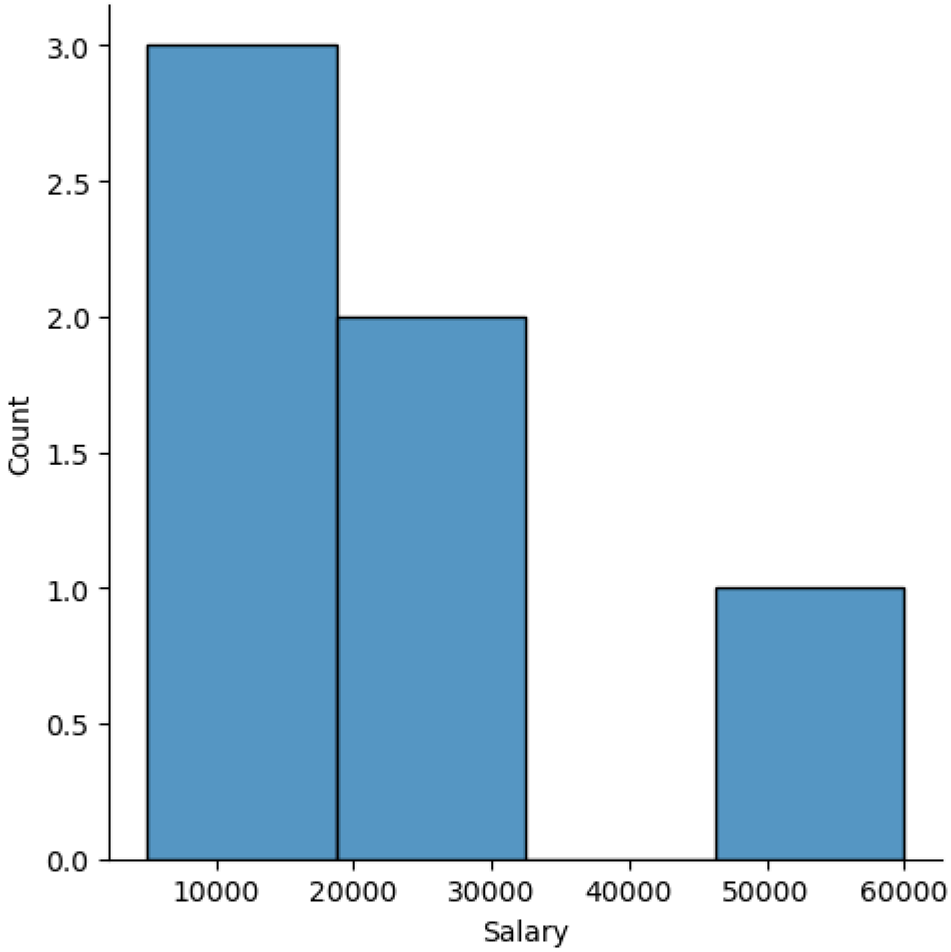
plotting with single variable

In [61]:

```
vis1 = sns.distplot(clean_data['Salary']) #distplot plots b/w density and salary
```

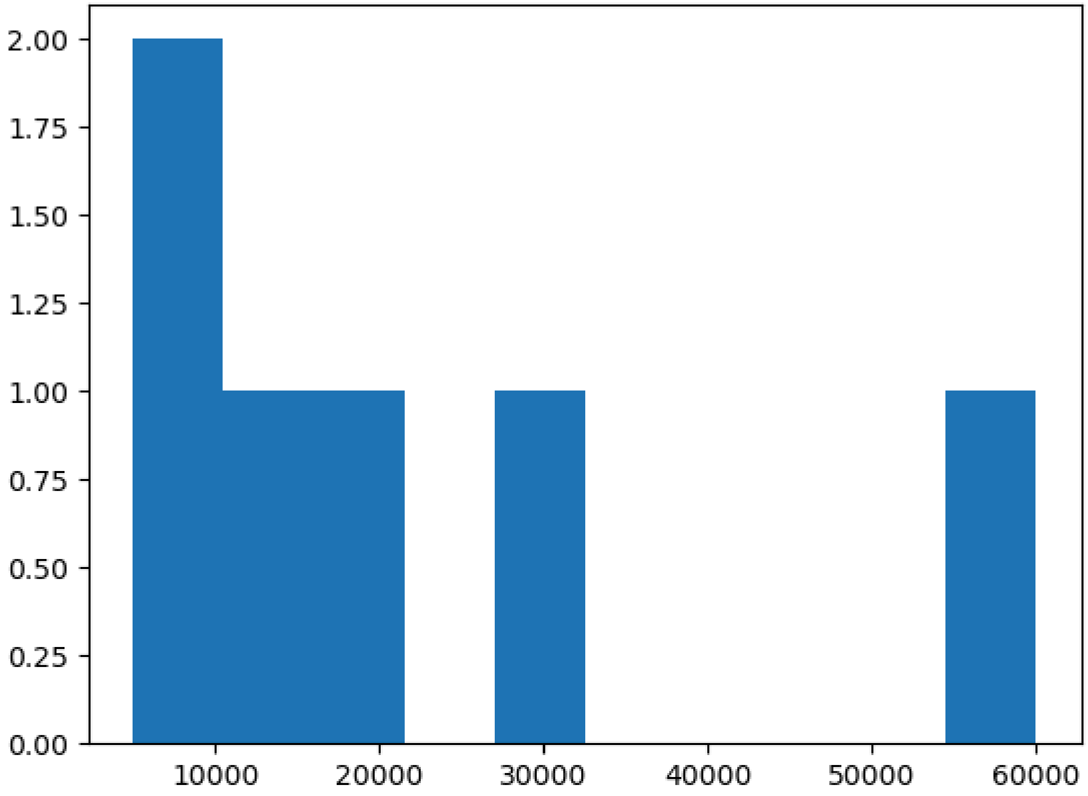


```
In [62]: vis2=sns.displot(clean_data['Salary'])
```



OUTLIER IDENTIFICATION

In [63]:
```
vis3=plt.hist(clean_data['Salary']) #60000 is far from remaining values
```



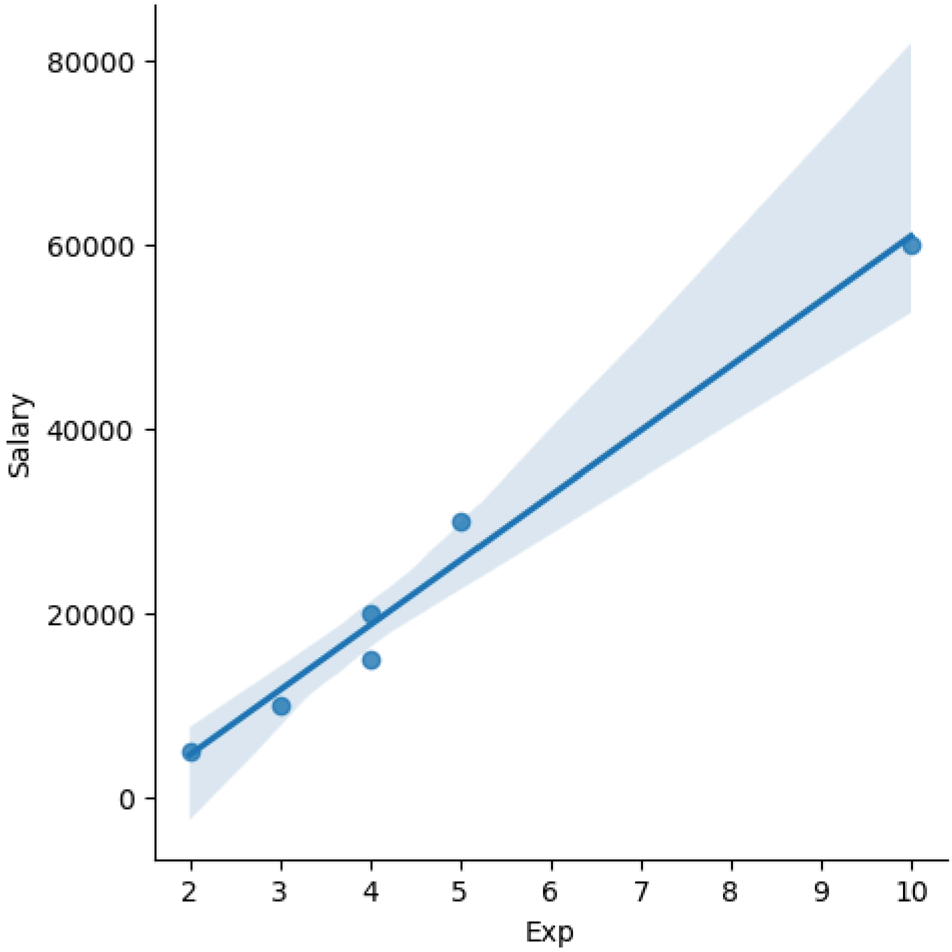
BIVARIATE ANALYSIS

plotting with two variables

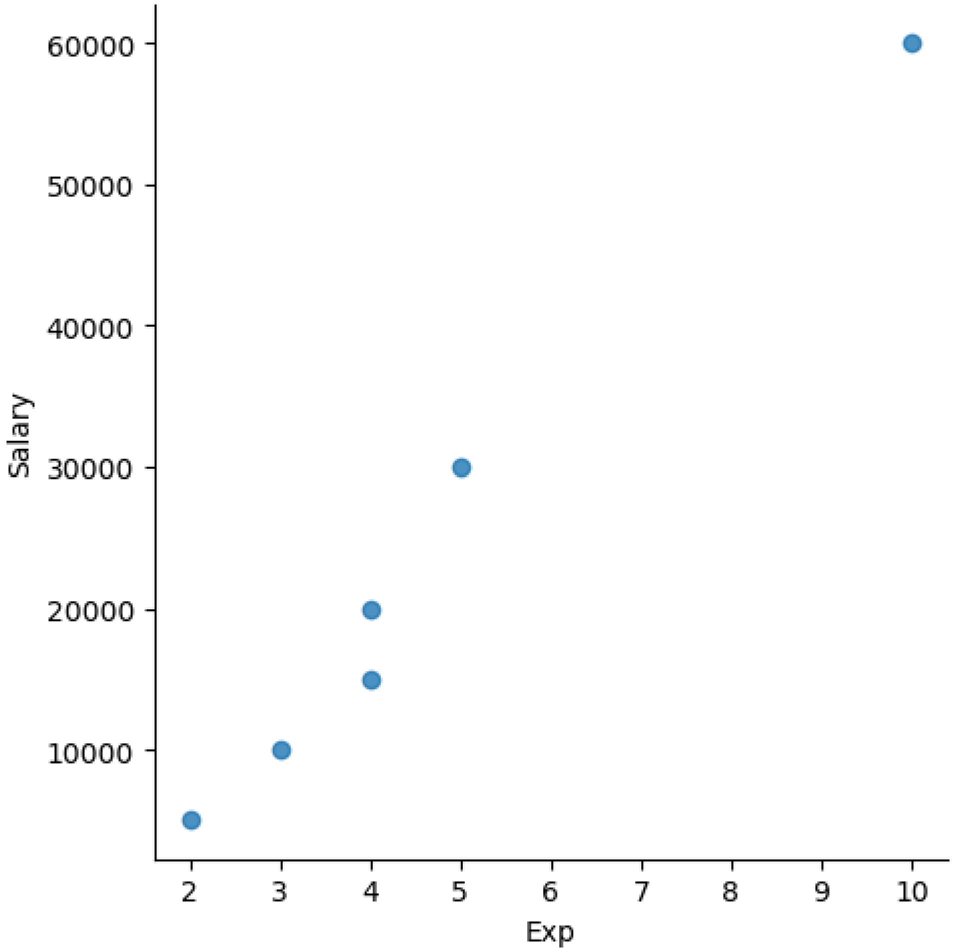
In [64]:
```
clean_data
```

Out[64]:
```
    Name       Domain  Age   Location  Salary  Exp
0   Mike  Datascience   34     Mumbai    5000    2
1  Teddy      Testing   45  Bangalore   10000    3
2   Umar  Dataanalyst   50  Bangalore   15000    4
3   Jane    Analytics   50    Hyderbad   20000    4
4  Uttam   Statistics   67  Bangalore   30000    5
5    Kim          NLP   55      Delhi   60000   10
```

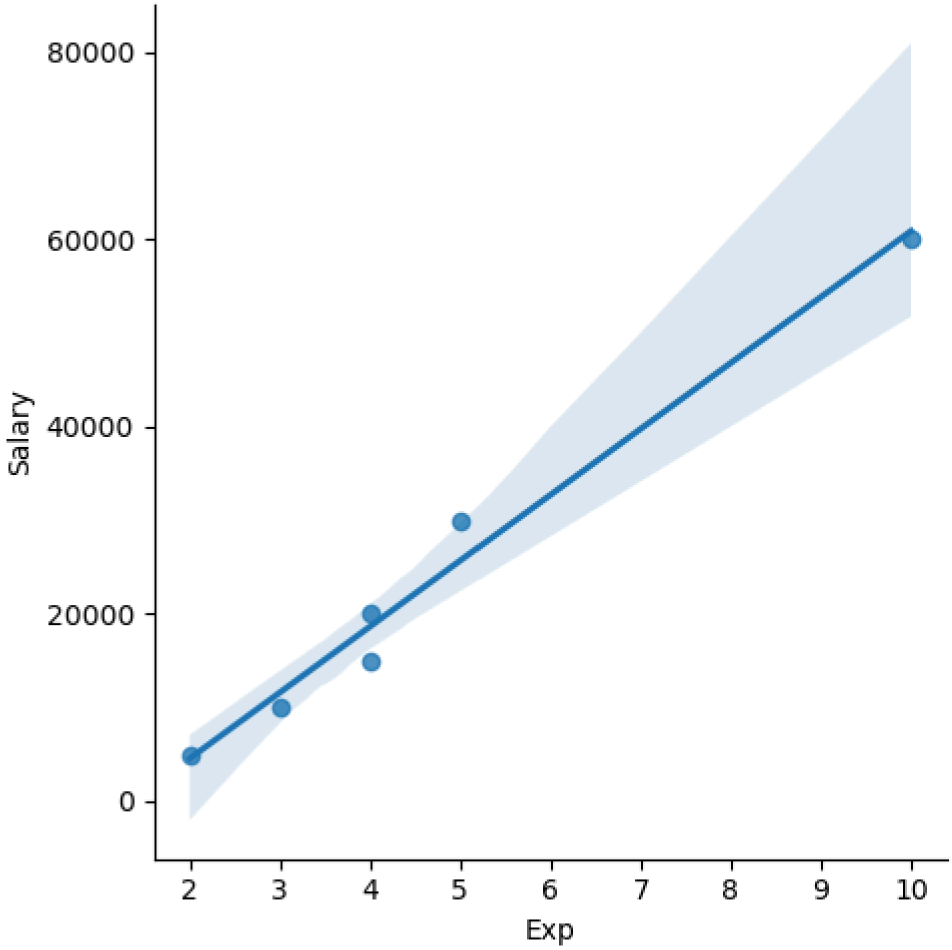
In [65]:
```
vis4 = sns.lmplot(data=clean_data,x='Exp',y='Salary')
```



```
In [66]: vis5 = sns.lmplot(data=clean_data,x='Exp',y='Salary',fit_reg=False)
```



```
In [67]: vis6 = sns.lmplot(data=clean_data,x='Exp',y='Salary',fit_reg=True)
```



In [68]:

```
clean_data[0:6:2] #Slicing
```

Out[68]:

	Name	Domain	Age	Location	Salary	Exp
0	Mike	Datascience	34	Mumbai	5000	2
2	Umar	Dataanalyst	50	Bangalore	15000	4
4	Uttam	Statistics	67	Bangalore	30000	5

In [69]:

```
clean_data
```

Out[69]:

	Name	Domain	Age	Location	Salary	Exp
0	Mike	Datascience	34	Mumbai	5000	2
1	Teddy	Testing	45	Bangalore	10000	3
2	Umar	Dataanalyst	50	Bangalore	15000	4
3	Jane	Analytics	50	Hyderbad	20000	4
4	Uttam	Statistics	67	Bangalore	30000	5
5	Kim	NLP	55	Delhi	60000	10

In [70]:

```
clean_data[::-1]
```

Out[70]:

	Name	Domain	Age	Location	Salary	Exp
5	Kim	NLP	55	Delhi	60000	10
4	Uttam	Statistics	67	Bangalore	30000	5
3	Jane	Analytics	50	Hyderbad	20000	4
2	Umar	Dataanalyst	50	Bangalore	15000	4
1	Teddy	Testing	45	Bangalore	10000	3
0	Mike	Datascience	34	Mumbai	5000	2

VARIABLE IDENTIFICATION

In [71]:

```
clean_data.columns
```

Out[71]:

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

In [72]:

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

In [73]:

```
X_iv
```

Out[73]:

	Name	Domain	Age	Location	Exp
0	Mike	Datascience	34	Mumbai	2
1	Teddy	Testing	45	Bangalore	3
2	Umar	Dataanalyst	50	Bangalore	4
3	Jane	Analytics	50	Hyderbad	4
4	Uttam	Statistics	67	Bangalore	5
5	Kim	NLP	55	Delhi	10

In [74]:

```
y_dv=clean_data['Salary']
```

In [75]:

```
y_dv
```

Out[75]:

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

In [76]:

```
clean_data
```

Out[76]:

	Name	Domain	Age	Location	Salary	Exp
0	Mike	Datascience	34	Mumbai	5000	2
1	Teddy	Testing	45	Bangalore	10000	3
2	Umar	Dataanalyst	50	Bangalore	15000	4
3	Jane	Analytics	50	Hyderbad	20000	4
4	Uttam	Statistics	67	Bangalore	30000	5
5	Kim	NLP	55	Delhi	60000	10

VARIABLE TRANSFORMATION

In [77]:

```
imputation = pd.get_dummies(clean_data,dtype=int)
```

In [78]:

```
imputation
```

Out[78]:

	Age	Salary	Exp	Name_Jane	Name_Kim	Name_Mike	Name_Teddy	Name_Umar
0	34	5000	2	0	0	1	0	0
1	45	10000	3	0	0	0	1	0
2	50	15000	4	0	0	0	0	1
3	50	20000	4	1	0	0	0	0
4	67	30000	5	0	0	0	0	0
5	55	60000	10	0	1	0	0	0

In [79]:

```
#imputation = pd.get_dummies(clean_data)
```

In [80]:

```
imputation
```

Out[80]:

	Age	Salary	Exp	Name_Jane	Name_Kim	Name_Mike	Name_Teddy	Name_Umar
0	34	5000	2	0	0	1	0	0
1	45	10000	3	0	0	0	1	0
2	50	15000	4	0	0	0	0	1
3	50	20000	4	1	0	0	0	0
4	67	30000	5	0	0	0	0	0
5	55	60000	10	0	1	0	0	0

In [81]:

```
clean_data.columns
```

Out[81]:

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

```
In [82]: imputation.columns
```

```
Out[82]: Index(['Age', 'Salary', 'Exp', 'Name_Jane', 'Name_Kim', 'Name_Mike',
                'Name_Teddy', 'Name_Umar', 'Name_Uttam', 'Domain_Analytics',
                'Domain_Dataanalyst', 'Domain_Datascience', 'Domain_NLP',
                'Domain_Statistics', 'Domain_Testing', 'Location_Bangalore',
                'Location_Delhi', 'Location_Hyderbad', 'Location_Mumbai'],
               dtype='object')
```

```
In [83]: len(imputation.columns)
```

```
Out[83]: 19
```

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

Out[84]:

	Name	Domain	Age	Location	Salary	Exp
0	Mike	Datascience	34	Mumbai	5000	2
1	Teddy	Testing	45	Bangalore	10000	3
2	Umar	Dataanalyst	50	Bangalore	15000	4
3	Jane	Analytics	50	Hyderbad	20000	4
4	Uttam	Statistics	67	Bangalore	30000	5
5	Kim	NLP	55	Delhi	60000	10