Raw Data to Clean Data conversion using Python EDA

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

In [2]:
```
pd.__version__
```

Out[2]:
```
'2.2.2'
```

In [3]:
```
emp=pd.read_excel(r'C:\Users\world\Desktop\FullStackDSandAI\Day31-21July2025\Rawdat
```

In [4]:
```
emp
```

Out[4]:

	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 [5]:
```
id(emp)
```

Out[5]:
```
1491141010704
```

In [6]:
```
emp.columns
```

Out[6]:
```
Index(['Name', 'Domain', 'Age', 'Location', 'Salary', 'Exp'], dtype='object')
```

In [7]:
```
emp.shape
```

Out[7]:
```
(6, 6)
```

In [8]:
```
emp.head()
```

Out[8]:

	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 [9]:
```
emp.tail()
```

Out[9]:

	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 [10]:
```
emp.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       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 [11]:
```
emp.isna()
```

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()
```

Out[12]:

	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 [13]:
```
emp.isnull().sum()
```

Out[13]:
```
Name        0
Domain      0
Age         2
Location    2
Salary      0
Exp         1
dtype: int64
```

DATA CLEANING OR DATA CLEANSING

In [14]:
```
emp
```

Out[14]:

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

Out[15]:
```
0      Mike
1    Teddy^
2     Uma#r
3      Jane
4    Uttam*
5       Kim
Name: Name, dtype: object
```

\W refers to non-word chars. Only Special chars, underscore(_) not allowed

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

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

Out[17]:
```
0     Mike
1    Teddy
2     Umar
3     Jane
4    Uttam
5      Kim
Name: Name, dtype: object
```

In [18]:
```
emp['Domain']=emp['Domain'].str.replace(r'\W','',regex=True)
```

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

In [21]:
```
emp
```

Out[21]:

	Name	Domain	Age	Location	Salary	Exp
0	Mike	Datascience	34years	Mumbai	5^00#0	2+
1	Teddy	Testing	45yr	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	67yr	NaN	30000-	5+ year
5	Kim	NLP	55yr	Delhi	6000^$0	10+

extract number from Age

In [22]:
```
emp['Age']=emp['Age'].str.extract('(\\d+)')
```

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

Out[23]:

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

In [25]:
```
emp
```

Out[25]:

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

In [27]:
```
emp
```

Out[27]:

	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> yrs
3	Jane	Analytics	NaN	Hyderbad	20000	NaN
4	Uttam	Statistics	67	NaN	30000	5+ year
5	Kim	NLP	55	Delhi	60000	10+

In [28]:
```
emp['Exp']=emp['Exp'].str.replace(r'\W','',regex=True)
```

In [29]:
```
emp
```

Out[29]:

	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	4yrs
3	Jane	Analytics	NaN	Hyderbad	20000	NaN
4	Uttam	Statistics	67	NaN	30000	5year
5	Kim	NLP	55	Delhi	60000	10

In [30]:
```
emp['Exp']=emp['Exp'].str.extract('(\\d+)')
```

In [31]:
```
emp
```

Out[31]:

	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 [32]:
```
clean_data=emp.copy()
```

In [33]:
```
clean_data
```

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

Till now we use regex to clean the data and removed all the special characters from dataset

EDA Technique: Lets Apply

Missing value Treatment for numerical data

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

Out[35]:

	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 [36]:
```
clean_data.isna().sum()
```

Out[36]:
```
Name        0
Domain      0
Age         2
Location    2
Salary      0
Exp         1
dtype: int64
```

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

In [41]:
```
clean_data['Age'] = clean_data['Age'].fillna(np.mean(pd.to_numeric(clean_data['Age'
```

In [42]:
```
clean_data['Age']
```

Out[42]:
```
0       34
1       45
2    50.25
3    50.25
4       67
5       55
Name: Age, dtype: object
```

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

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

In [44]:
```
clean_data['Exp'] = clean_data['Exp'].fillna(np.mean(pd.to_numeric(clean_data['Exp'
```

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

Out[45]:
```
0      2
1      3
2      4
3    4.8
4      5
5     10
Name: Exp, 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	NaN	15000	4
3	Jane	Analytics	50.25	Hyderbad	20000	4.8
4	Uttam	Statistics	67	NaN	30000	5
5	Kim	NLP	55	Delhi	60000	10

Missing value treatment for categorical data

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

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	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 [50]:
```
emp.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       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 [51]:
```
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
```

Changing Datatype Object to integer

.astype() ---> converts system buidin data type to user wanted datatype

In [52]:
```
clean_data['Age']= clean_data['Age'].astype(int)
```

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

In [54]:
```
clean_data['Exp']= clean_data['Exp'].astype(int)
```

In [55]:
```
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 [56]:
```
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 [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: 866.0 bytes
```

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

Out[59]:

	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

save the clean_data to our system

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

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

Out[61]:
```
'C:\\Users\\world\\Desktop\\FullStackDSandAI\\Day31-21July2025'
```

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

Out[62]:

	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

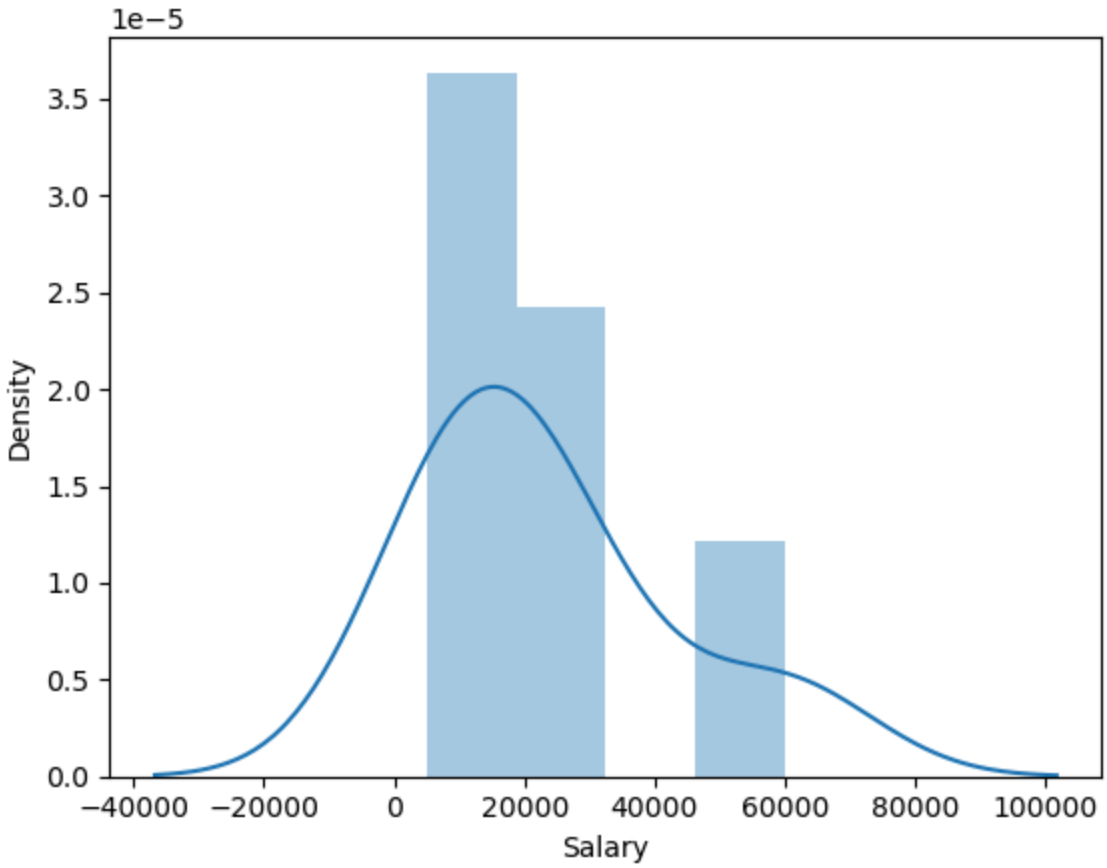
In [63]:
```
import matplotlib.pyplot as plt
import seaborn as sns
```

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

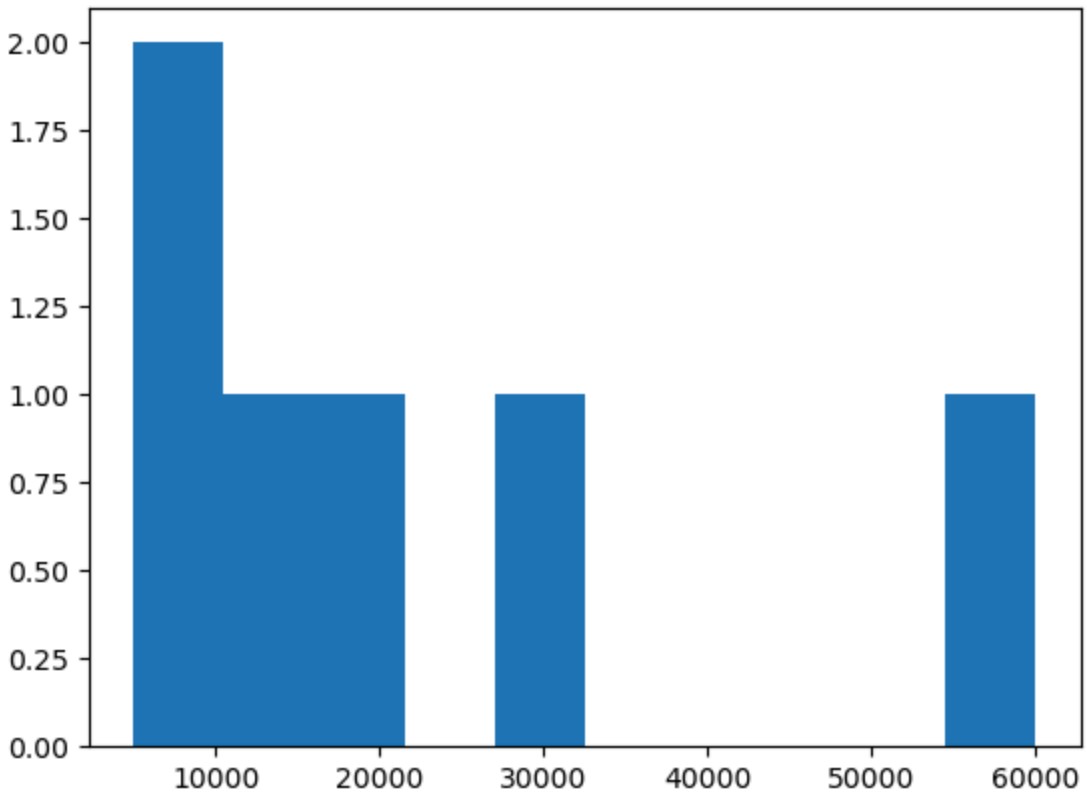
In [65]:
```
clean_data['Salary']
```

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

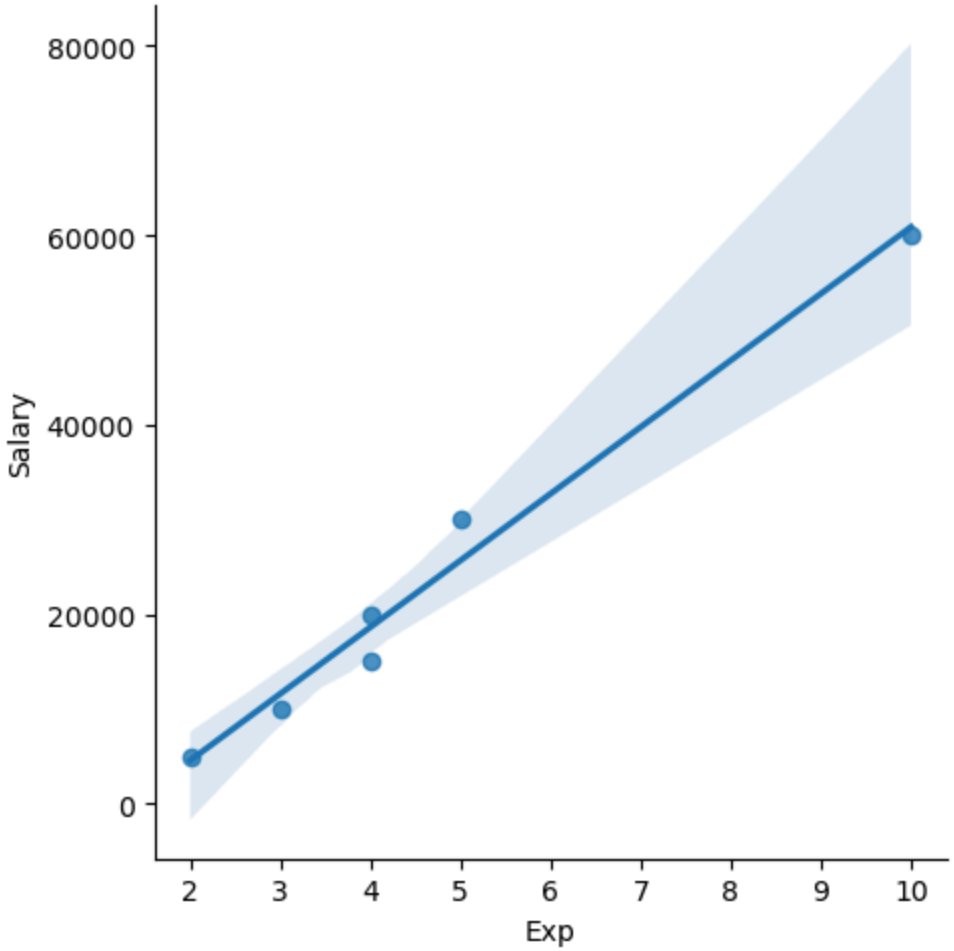
In [66]:
```
vis1=sns.distplot(clean_data['Salary'])
```



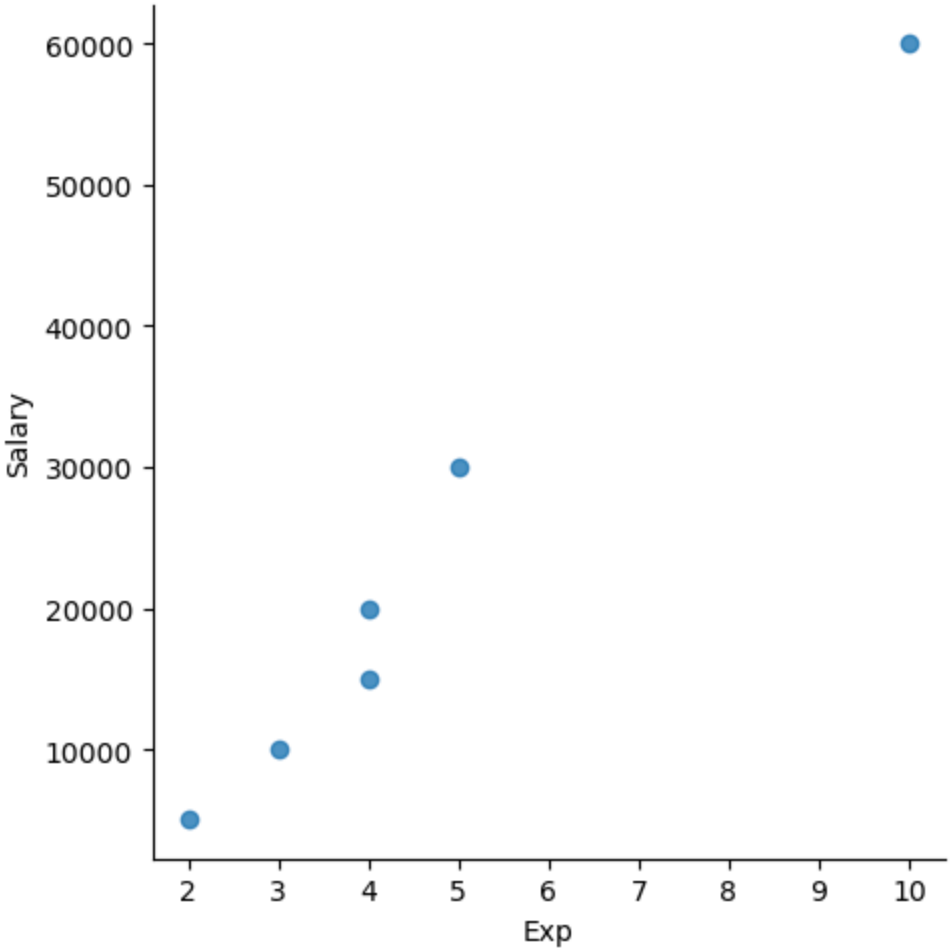
In [67]:
```
vis1=plt.hist(clean_data['Salary'])
```



In [68]:
```
vis3=sns.lmplot(data=clean_data, x='Exp', y='Salary')
```



In [69]:
```
vis3=sns.lmplot(data=clean_data, x='Exp', y='Salary', fit_reg=False)
```



In [70]:
```
clean_data
```

Out[70]:

	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 Identification

In [71]:
```
x_iv= clean_data[['Name','Domain','Age','Location','Exp']]
```

In [72]:
```
x_iv
```

Out[72]:

	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 [73]:
```
y_dv=clean_data[['Salary']]
```

In [74]:
```
y_dv
```

Out[74]:

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

Apply Imputation Techniques

In [75]:
```
imputation=pd.get_dummies(clean_data, dtype=int)
```

In [76]:
```
imputation
```

Out[76]:

	Age	Salary	Exp	Name_Jane	Name_Kim	Name_Mike	Name_Teddy	Name_Umar	Nan
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 [77]:
```
clean_data
```

Out[77]:

	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 [82]:
```
len(clean_data.columns)
```

Out[82]:
```
6
```

In [79]:
```
imputation.columns
```

Out[79]:
```
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 [81]:
```
len(imputation.columns)
```

Out[81]:
```
19
```