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

In [2]:
```
emp=pd.read_excel(r"C:\Users\Admin\Downloads\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+ |

In [4]:
```
emp.shape
```

Out[4]:
```
(6, 6)
```

In [5]:
```
len(emp)
```

Out[5]:
```
6
```

In [6]:
```
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 [7]:
```
emp.columns
```

Out[7]:
```
Index(['Name', 'Domain', 'Age', 'Location', 'Salary', 'Exp'], dtype='object')
```

In [8]:
```
len(emp.columns)
```

Out[8]:
```
6
```

In [9]:
```
emp.info() #missing data is shown ##
```

```
<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['Name']
```

Out[10]:
```
0      Mike
1    Teddy^
2     Uma#r
3      Jane
4    Uttam*
5       Kim
Name: Name, dtype: object
```

In [11]:
```
emp['Domain']
```

Out[11]:
```
0     Datascience#$
1           Testing
2    Dataanalyst^^#
3       Ana^^lytics
4        Statistics
5               NLP
Name: Domain, dtype: object
```

In [12]:
```
emp[['Name', 'Domain']]
```

Out[12]:

| | Name | Domain |
|---|---|---|
| **0** | Mike | Datascience#$ |
| **1** | Teddy^ | Testing |
| **2** | Uma#r | Dataanalyst^^# |
| **3** | Jane | Ana^^lytics |
| **4** | Uttam* | Statistics |
| **5** | Kim | NLP |

In [13]:
```
emp['Name']
```

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) #this means all non wor
```

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('(\d+)') #this will extract only digits
```

```
<>:1: SyntaxWarning: invalid escape sequence '\d'
<>:1: SyntaxWarning: invalid escape sequence '\d'
C:\Users\Admin\AppData\Local\Temp\ipykernel_2580\1027326802.py:1: SyntaxWarning:
invalid escape sequence '\d'
  emp['Age']=emp['Age'].str.extract('(\d+)') #this will extract only digits
```

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

Out[22]:
```
0     34
1     45
2    NaN
3    NaN
4     67
5     55
Name: Age, dtype: object
```

In [23]:
```
emp['Location']=emp['Location'].str.replace(r'\W','',regex=True)
```

In [24]:
```
emp['Location']
```

Out[24]:
```
0       Mumbai
1    Bangalore
2          NaN
3     Hyderbad
4          NaN
5        Delhi
Name: Location, dtype: object
```

In [25]:
```
emp.columns
```

Out[25]:
```
Index(['Name', 'Domain', 'Age', 'Location', 'Salary', 'Exp'], dtype='object')
```

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

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

```
<>:1: SyntaxWarning: invalid escape sequence '\d'
<>:1: SyntaxWarning: invalid escape sequence '\d'
C:\Users\Admin\AppData\Local\Temp\ipykernel_2580\1466635560.py:1: SyntaxWarning:
invalid escape sequence '\d'
  emp['Exp']=emp['Exp'].str.extract('(\d+)')
```

In [29]:
```
emp['Exp']
```

Out[29]:
```
0      2
1      3
2      4
3    NaN
4      5
5     10
Name: Exp, dtype: object
```

In [30]:
```
clean_data=emp.copy()
```

In [31]:
```
clean_data
```

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.isnull().sum() #number of null values in each columns
```

Out[32]:
```
Name        0
Domain      0
Age         2
Location    2
Salary      0
Exp         1
dtype: int64
```

In [33]:
```
import numpy as np
```

In [34]:
```
clean_data['Age']=clean_data['Age'].fillna(np.mean(pd.to_numeric(clean_data['Age
```

In [35]:
```
clean_data['Age']
```

Out[35]:
```
0       34
1       45
2    50.25
3    50.25
4       67
5       55
Name: Age, dtype: object
```

In [36]:
```
clean_data['Exp']=clean_data['Exp'].fillna(np.mean(pd.to_numeric(clean_data['Exp
```

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

Out[37]:
```
0      2
1      3
2      4
3    4.8
4      5
5     10
Name: Exp, dtype: object
```

In [38]:
```
clean_data['Domain']=clean_data['Domain'].fillna(clean_data['Domain'].mode()[0])
```

In [39]:
```
clean_data['Domain']
```

Out[39]:
```
0    Datascience
1        Testing
2    Dataanalyst
3      Analytics
4     Statistics
5            NLP
Name: Domain, dtype: object
```

In [40]:
```
clean_data['Location']=clean_data['Location'].fillna(clean_data['Location'].mode
```

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

Out[41]:
```
0       Mumbai
1    Bangalore
2    Bangalore
3     Hyderbad
4    Bangalore
5        Delhi
Name: Location, dtype: object
```

In [42]:
```
clean_data
```

Out[42]:

| | 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 [43]:
```
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 [44]:
```
clean_data['Age']=clean_data['Age'].astype(int) #conversion of data types
clean_data['Salary']=emp['Salary'].astype(int)
clean_data['Exp']=clean_data['Exp'].astype(int)
clean_data['Domain']=clean_data['Domain'].astype('category')
clean_data['Location']=clean_data['Location'].astype('category')
clean_data['Name']=clean_data['Name'].astype('category')
```

In [45]:
```
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
```

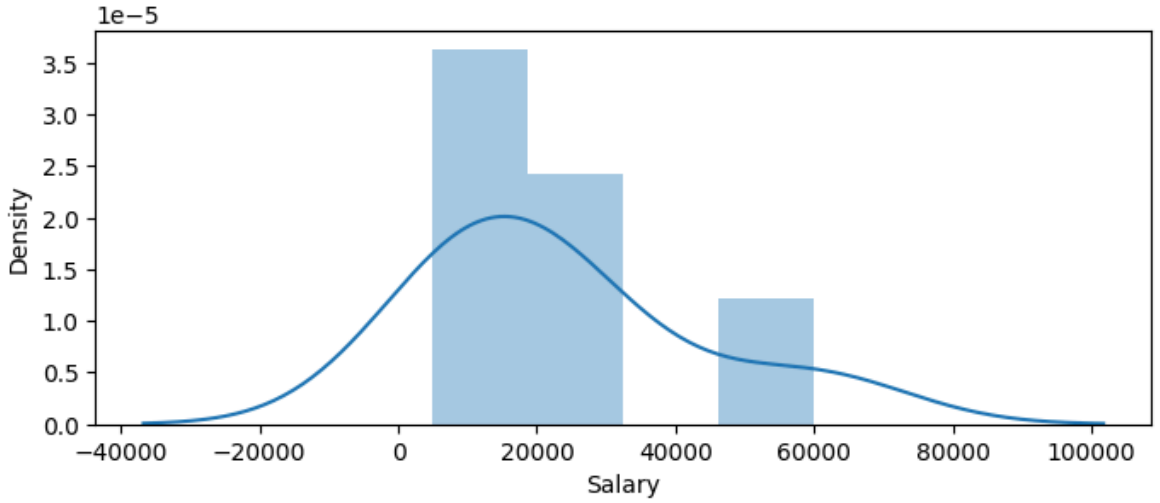
exploratary data analysis(eda)

In [46]:
```
import matplotlib.pyplot as plt
import seaborn as sns
```

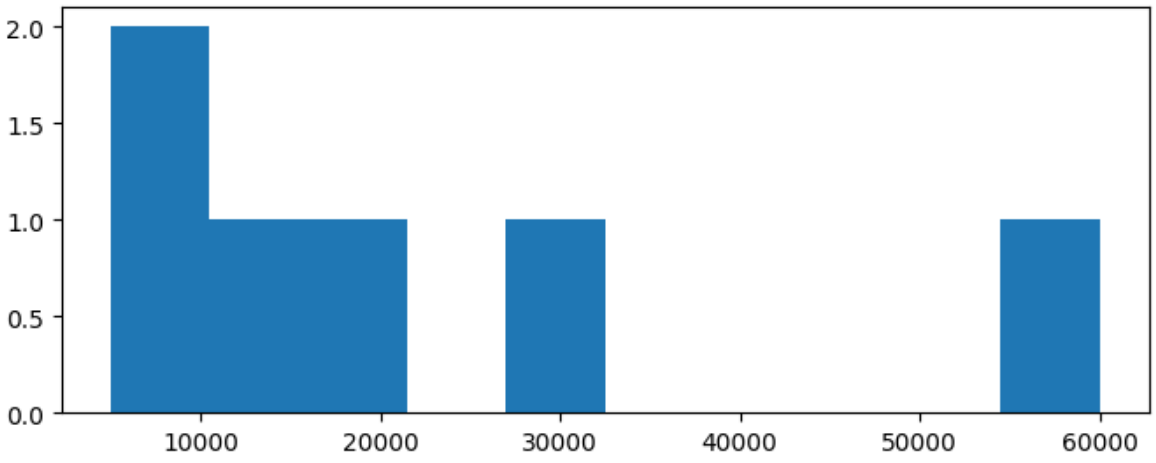
In [47]:
```
import warnings
warnings.filterwarnings('ignore')
```

In [48]:
```
plt.rcParams['figure.figsize']=8,3
```

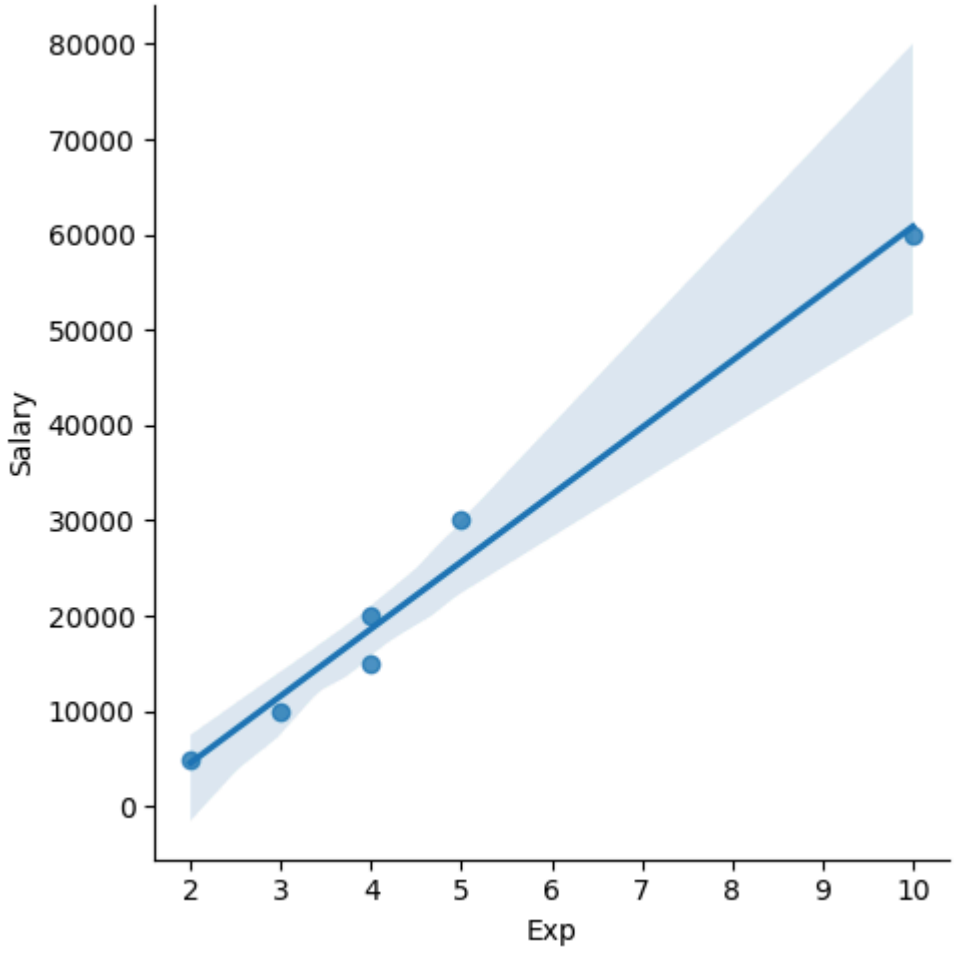
In [49]:
```
vis1=sns.distplot(clean_data['Salary']) #univariante analysis
```



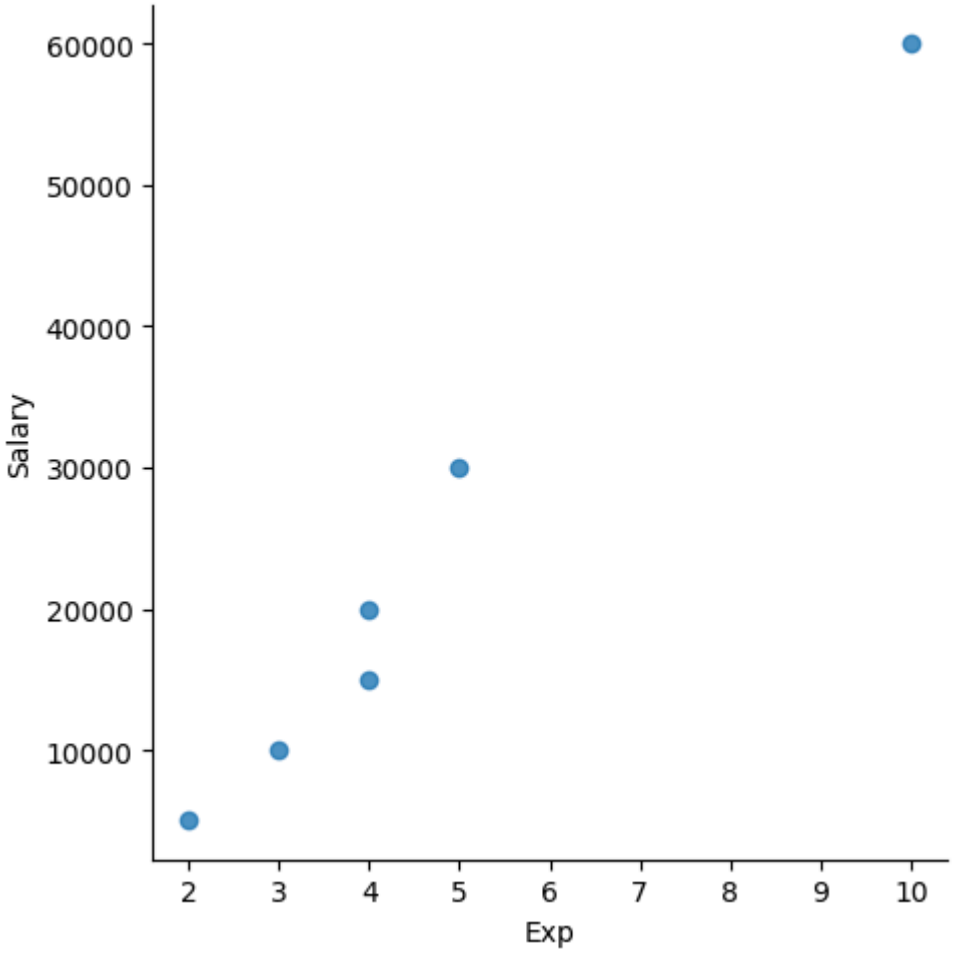

In [50]:
```
vis2=plt.hist(clean_data['Salary'])
```




In [51]:
```
vis3=sns.lmplot(data=clean_data,x='Exp',y='Salary') # bivariante analysis
```




In [52]:
```
vis4=sns.lmplot(data=clean_data,x='Exp',y='Salary',fit_reg=False) #here we can s
```




In [53]:
```
clean_data.columns
```

Out[53]:
```
Index(['Name', 'Domain', 'Age', 'Location', 'Salary', 'Exp'], dtype='object')
```

In [54]:
```
x_iv=clean_data[['Name', 'Domain', 'Age', 'Location','Exp']] #independent variab
```

In [55]:
```
x_iv
```

Out[55]:

| | 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 [56]:
```
y_dv=clean_data['Salary'] # dependent variale
```

In [57]:
```
y_dv
```

Out[57]:
```
0     5000
1    10000
2    15000
3    20000
4    30000
5    60000
Name: Salary, dtype: int32
```

In [58]:
```
clean_data
```

Out[58]:

| | 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 [59]:
```
x_iv
```

Out[59]:

| | 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 [60]:
```
y_dv
```

Out[60]:
```
0     5000
1    10000
2    15000
3    20000
4    30000
5    60000
Name: Salary, dtype: int32
```

In [61]:
```
imputation=pd.get_dummies(clean_data,dtype=int) #varibale transformation and cre
```

In [62]:
```
imputation
```

Out[62]:

| | 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 [ ]: