Name : Bodhisatya Ghosh
Class : CSE DS
UID : 2021700026
Subject : ML
Experiment number : 1

Theory:

Exploratory data analysis (EDA) is used by data scientists to analyze and investigate data sets and summarize their main characteristics, often employing data visualization methods. It helps determine how best to manipulate data sources to get the answers you need, making it easier for data scientists to discover patterns, spot anomalies, test a hypothesis, or check assumptions.

EDA is primarily used to see what data can reveal beyond the formal modeling or hypothesis testing task and provides a provides a better understanding of data set variables and the relationships between them. It can also help determine if the statistical techniques you are considering for data analysis are appropriate.

In [ ]:
```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
```

In [ ]:
```
df = pd.read_csv('./salaries_final.csv')
df.head(1)
```

Out[ ]:

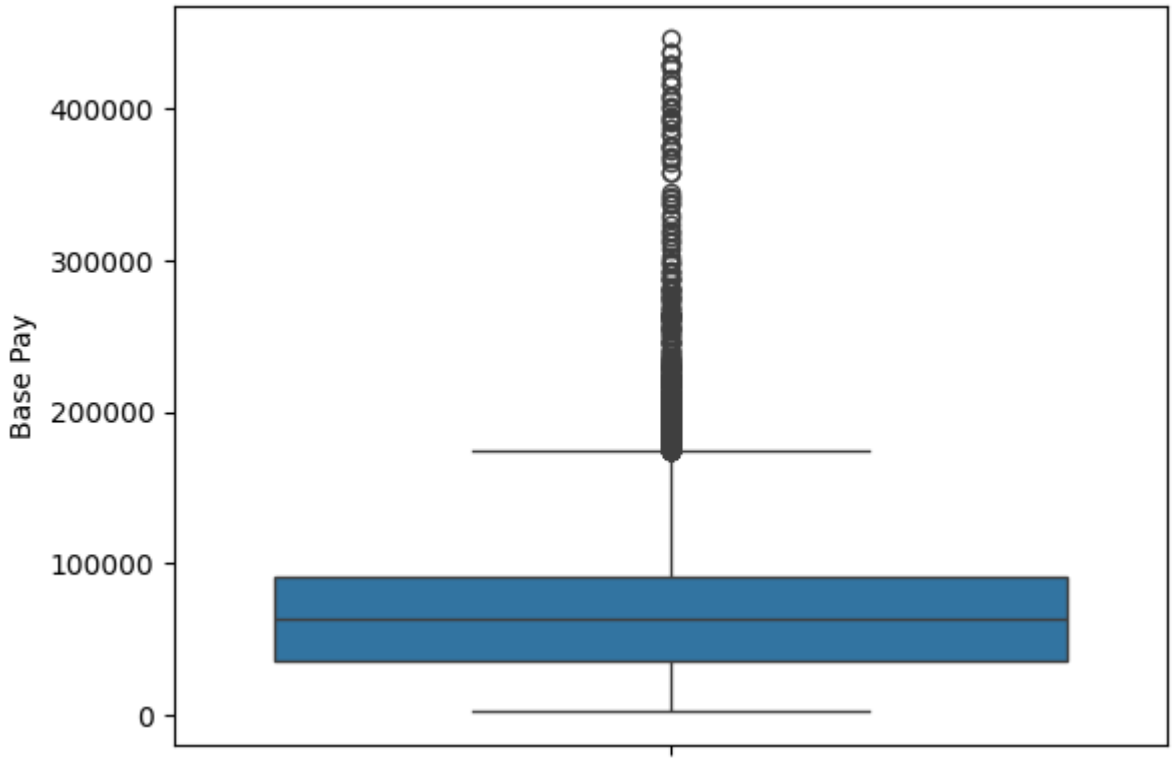
	Year	Name	Primary Job Title	Base Pay	Department	College
0	2010	Abaied, Jamie L.	Assistant Professor	64000.0	Department of Psychological Science	CAS

In [ ]:
```
df.drop(columns=['Name'], inplace=True)
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 14470 entries, 0 to 14469
Data columns (total 5 columns):
 #   Column             Non-Null Count  Dtype
---  ------             --------------  -----
 0   Year               14470 non-null  int64
 1   Primary Job Title  14470 non-null  object
 2   Base Pay           14470 non-null  float64
 3   Department         14470 non-null  object
 4   College            14470 non-null  object
dtypes: float64(1), int64(1), object(3)
memory usage: 565.4+ KB
```

In [ ]:
```
sns.boxplot(df['Base Pay'])
```

Out[ ]: <Axes: ylabel='Base Pay'>



In [ ]:
```
df['College'].value_counts()
```

Out[ ]:
```
College
COM                     6723
CAS                     3692
CEMS                     869
CALS                     793
CESS                     770
CNHS                     601
RSENR                    437
Business                 318
Library                  171
Department of Ext         74
LCOMEO                    12
Learning and Info Tech    10
Name: count, dtype: int64
```

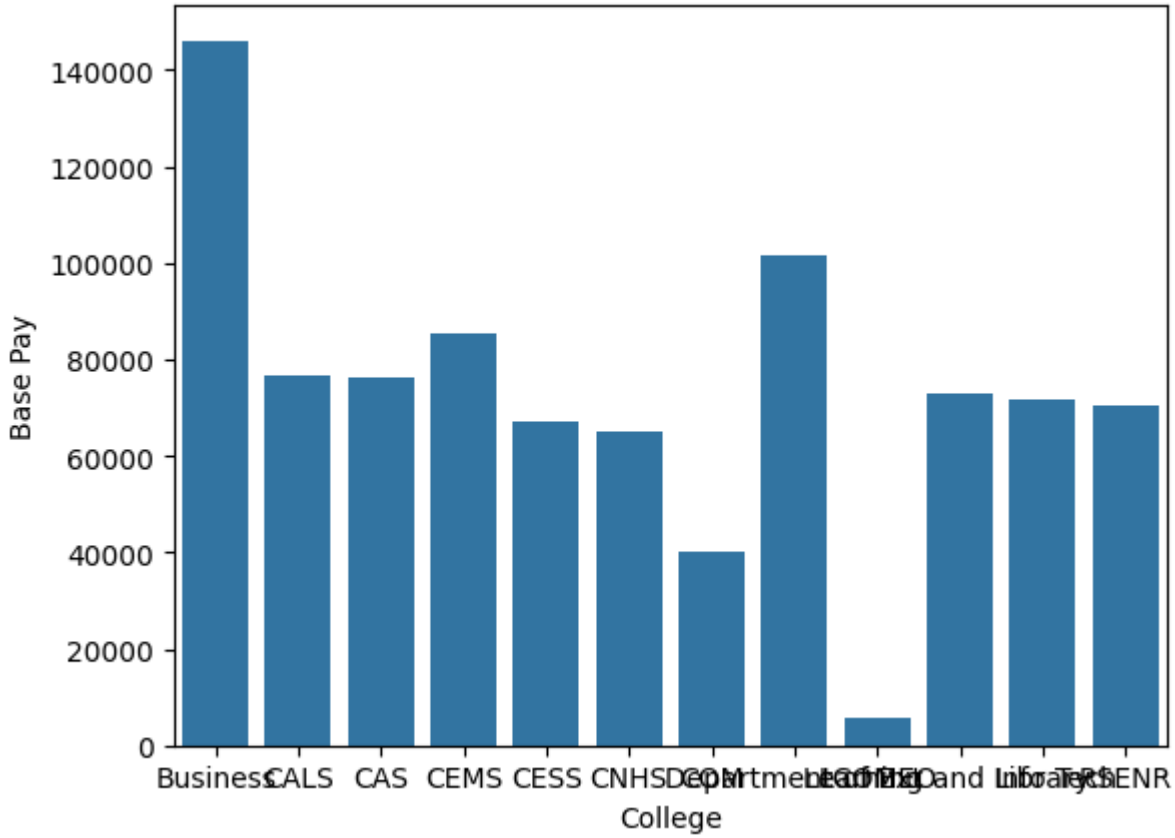
In [ ]:
```
collegeMedSal = pd.DataFrame(df.groupby(by='College')['Base Pay'].median()).reset_index()
collegeMedSal
```

Out[ ]:

	College	Base Pay
0	Business	146060.50
1	CALS	76448.22
2	CAS	76072.50
3	CEMS	85267.00
4	CESS	66939.00
5	CNHS	65000.00
6	COM	40000.00
7	Department of Ext	101627.50
8	LCOMEO	5727.00
9	Learning and Info Tech	72942.00
10	Library	71527.00
11	RSENR	70306.00

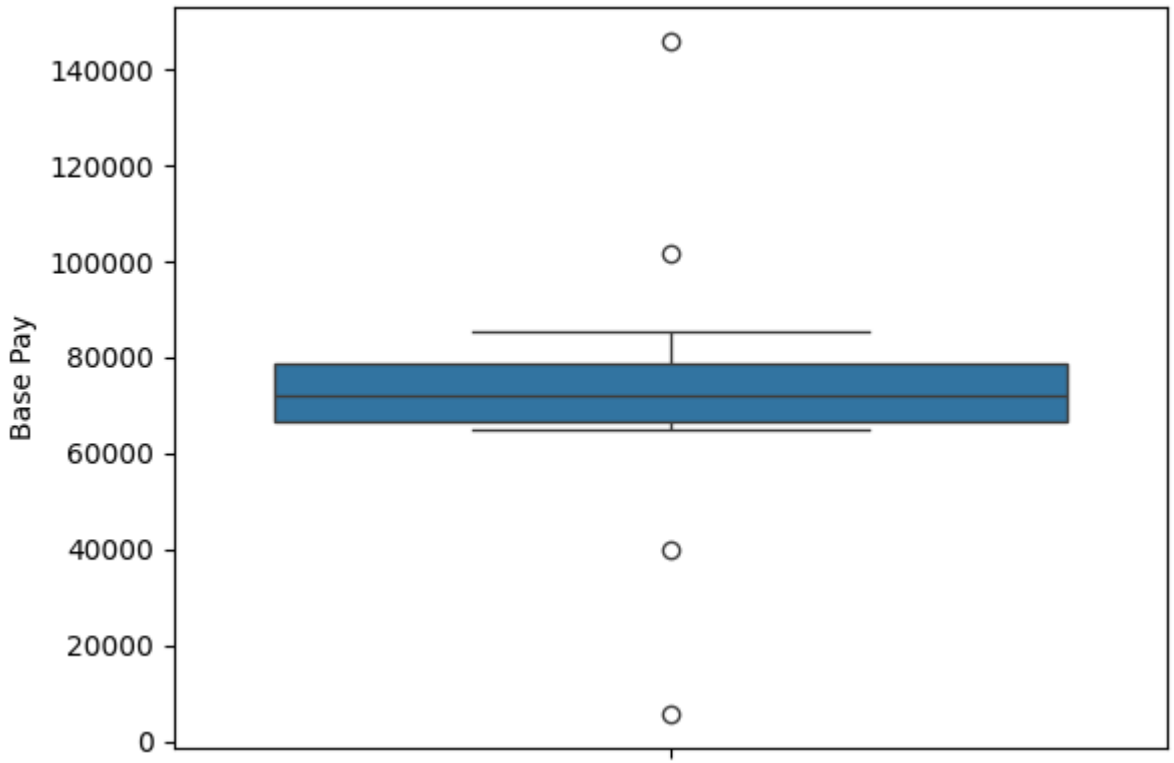
In [ ]:
```
sns.barplot(data=collegeMedSal,x = collegeMedSal['College'], y = collegeMedSal['Base Pay'])
```

Out[ ]:
```
<Axes: xlabel='College', ylabel='Base Pay'>
```



In [ ]:
```
sns.boxplot(collegeMedSal['Base Pay'])
```

Out[ ]:
```
<Axes: ylabel='Base Pay'>
```



In [ ]:
```
jobMedSal = pd.DataFrame(df.groupby(by='Primary Job Title')['Base Pay'].median()).reset_index()
jobMedSal
```

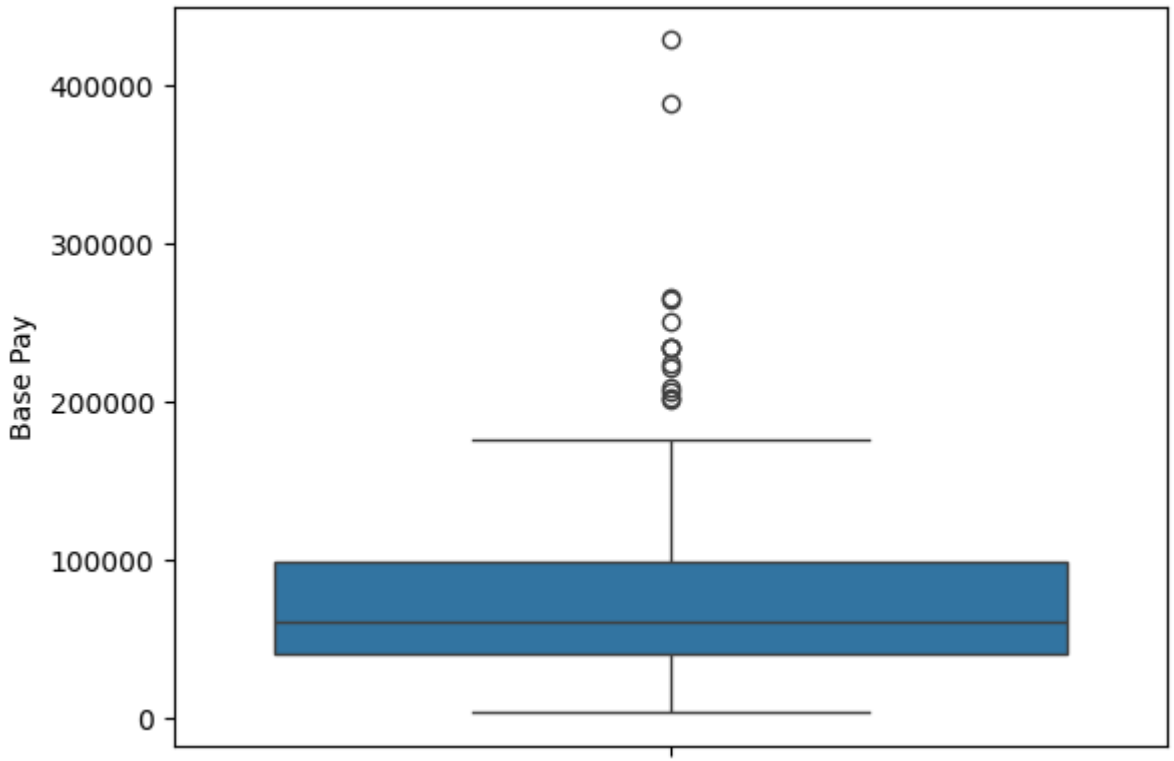
Out[ ]:

	Primary Job Title	Base Pay
0	Academic Srvcs Professional	35830.0
1	Academic Srvcs Professional Sr	53207.5
2	Academic Srvcs Professonal Sr	62276.0
3	Acting Director	87125.0
4	Acting Director Dana Medical Library	111450.0
...	...	...
142	VP Research	265753.5
143	Vice Pres for Enrollment Mgmnt	209589.5
144	Visiting Assistant Prof	55116.0
145	Visiting Instructor	62076.5
146	Visiting Lecturer	45100.0

147 rows × 2 columns

In [ ]:
```
sns.boxplot(jobMedSal['Base Pay'])
```

Out[ ]:
```
<Axes: ylabel='Base Pay'>
```



As there are too many distinct job titles, using this variable as a categorical independent variable will increase dimensionality too much and will not be of use in analysis

In [ ]:
```
# df.drop(columns=['Primary Job Title'], inplace=True)
# df.info()
```

In [ ]:
```
departmentMedSal = pd.DataFrame(df.groupby(by='Department')['Base Pay'].median()).reset_index()
departmentMedSal
```

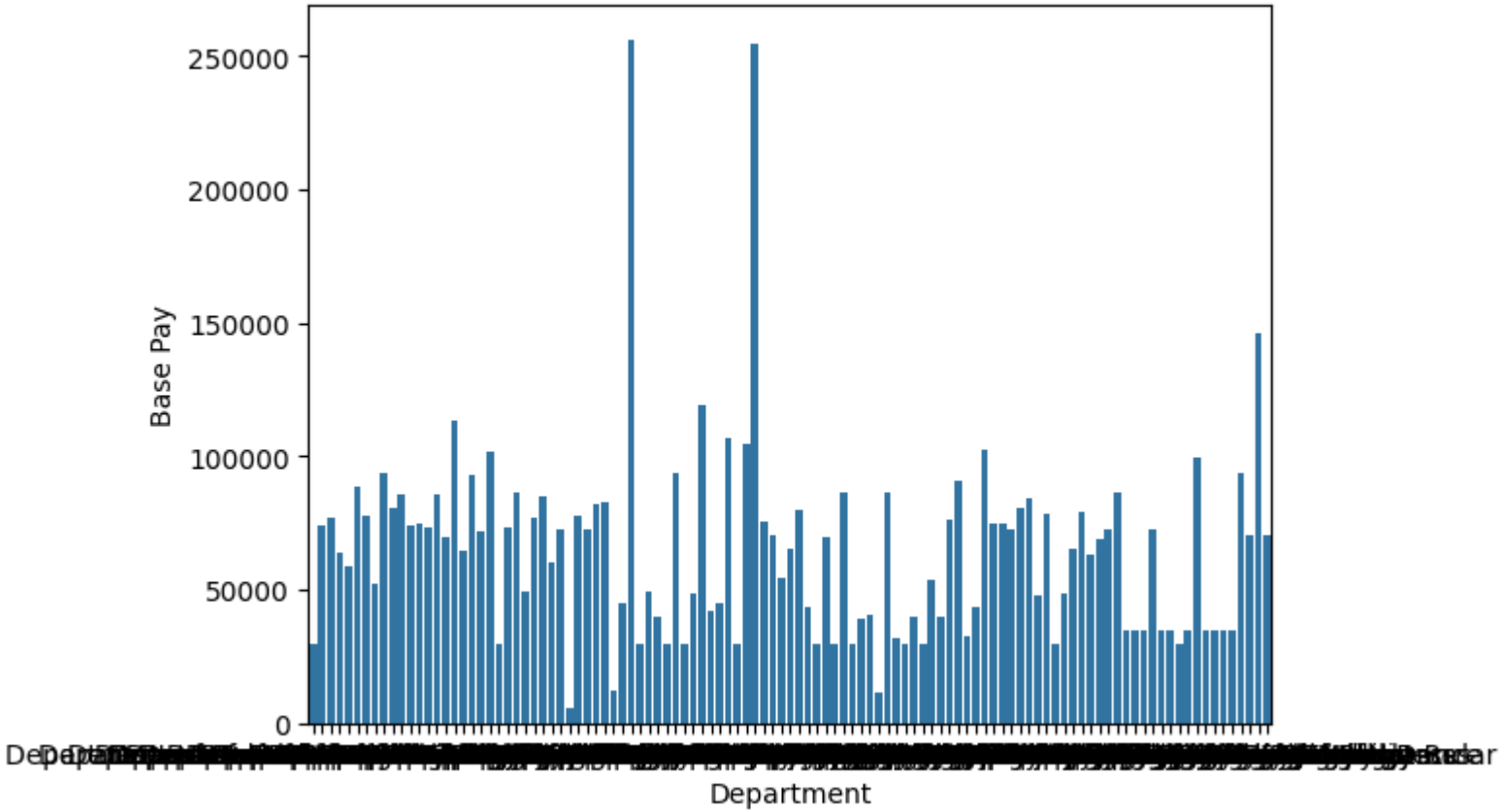
Out[ ]:

	Department	Base Pay
0	Department of Anesthesiology	30000.000
1	Department of Animal and Veterinary Sciences	74313.595
2	Department of Anthropology	76892.000
3	Department of Art & Art History	64193.000
4	Department of Asian Languages & Literatures	58917.965
...	...	...
104	Department of Surg-Vascular	35000.000
105	Department of Surgery	93967.500
106	Department of Theatre and Dance	70473.000
107	Grossman School of Business	146060.500
108	Rubenstein Sch Env & Nat Res	70306.000

109 rows × 2 columns

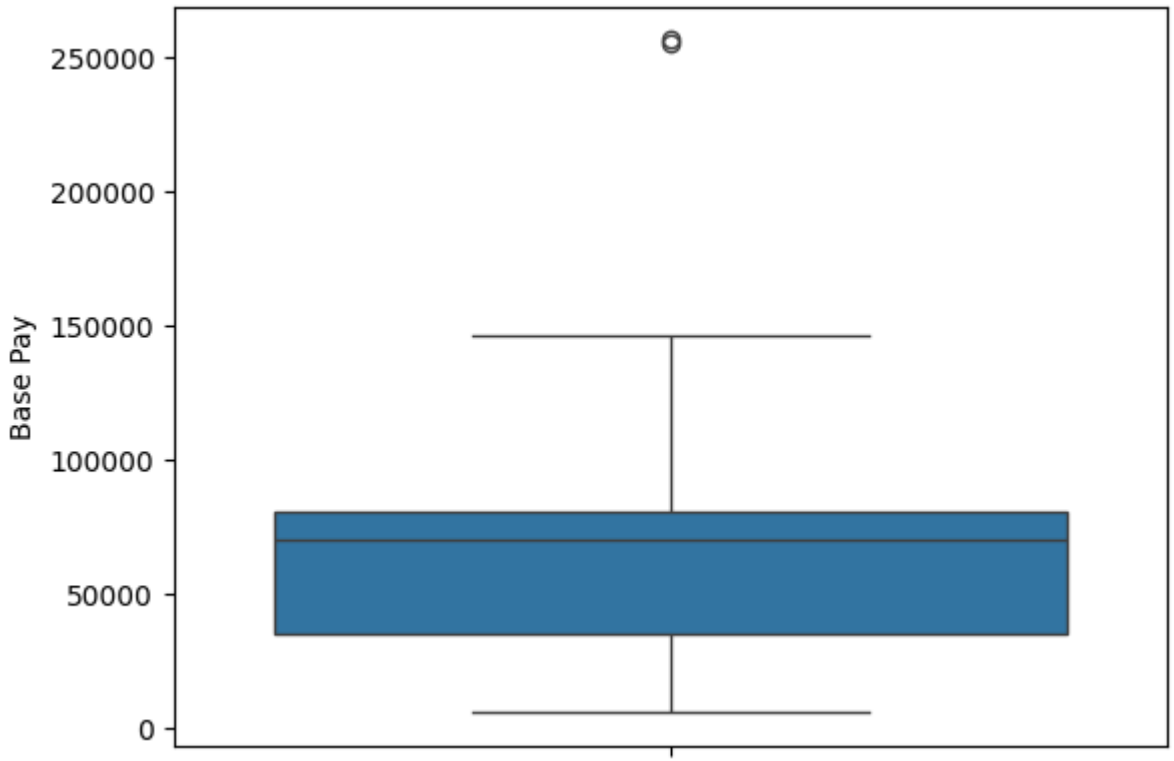
In [ ]:
```
sns.barplot(data=departmentMedSal,x = departmentMedSal['Department'], y = departmentMedSal['Base Pay'])
```

Out[ ]:
```
<Axes: xlabel='Department', ylabel='Base Pay'>
```



In [ ]:
```
sns.boxplot(departmentMedSal['Base Pay'])
```

Out[ ]:
```
<Axes: ylabel='Base Pay'>
```



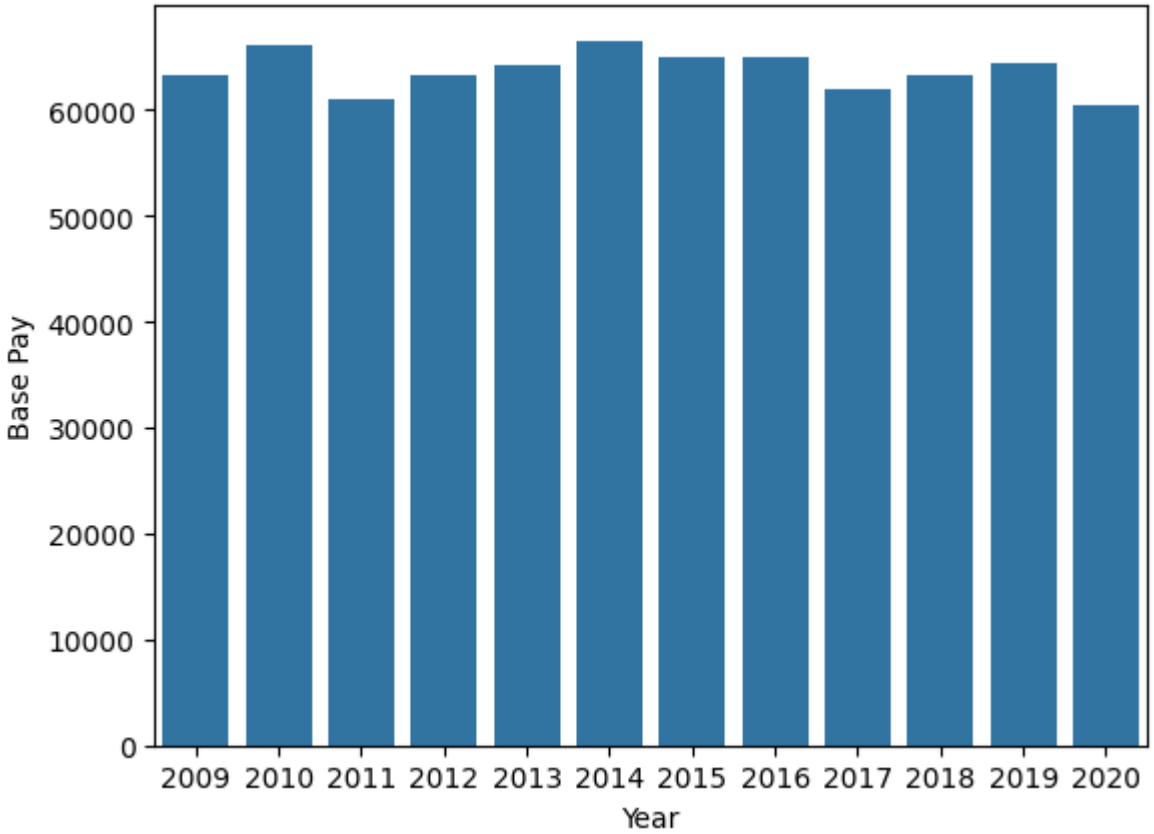
In [ ]:
```
YearMedSal = pd.DataFrame(df.groupby(by='Year')['Base Pay'].median()).reset_index()
YearMedSal
```

Out[ ]:

	Year	Base Pay
0	2009	63290.0
1	2010	66101.0
2	2011	61000.0
3	2012	63259.0
4	2013	64280.0
5	2014	66574.5
6	2015	64985.0
7	2016	64946.0
8	2017	61920.0
9	2018	63314.0
10	2019	64444.0
11	2020	60402.0

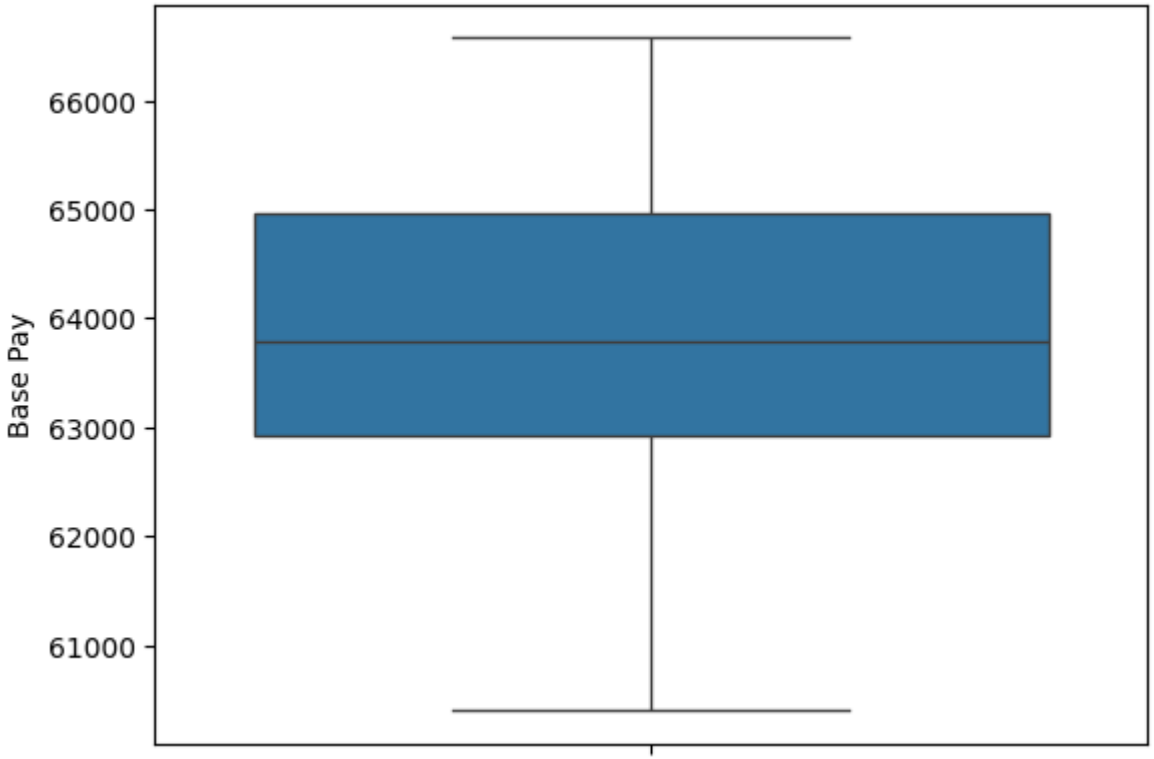
In [ ]:
```
sns.barplot(data=YearMedSal,x = YearMedSal['Year'], y = YearMedSal['Base Pay'])
```

Out[ ]:
```
<Axes: xlabel='Year', ylabel='Base Pay'>
```



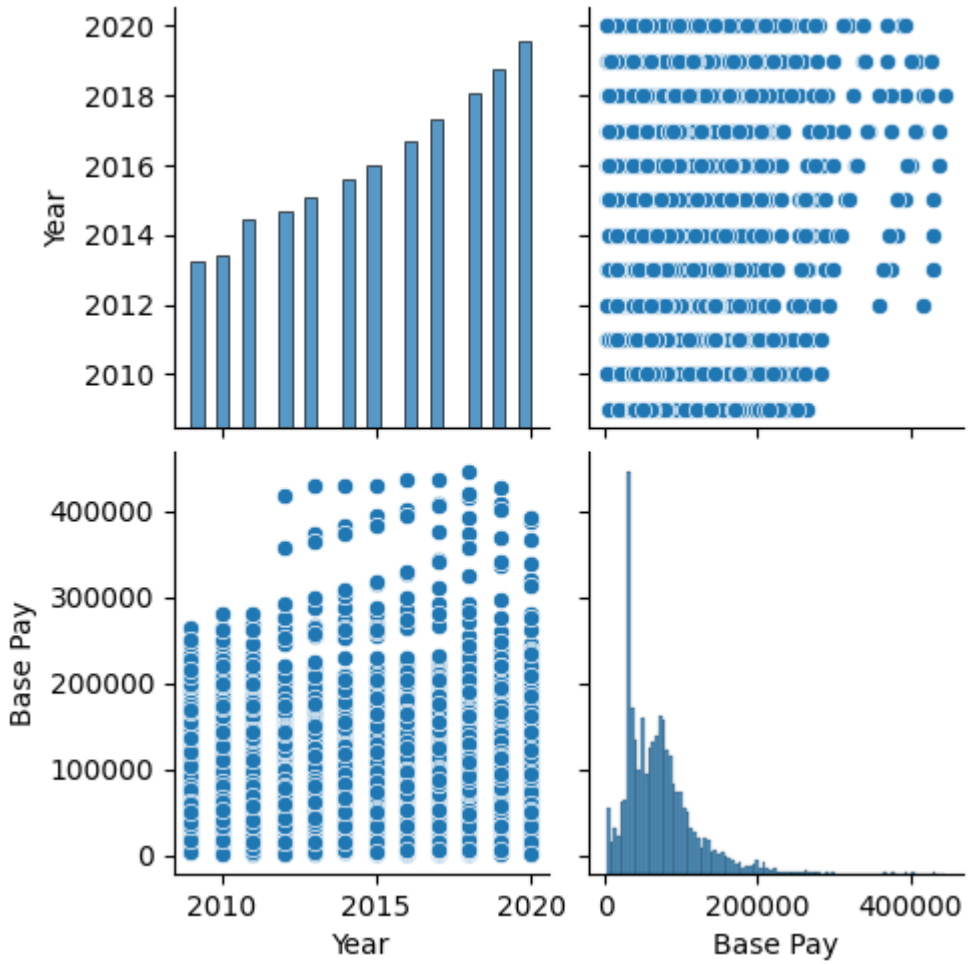
In [ ]:
```
sns.boxplot(YearMedSal['Base Pay'])
```

Out[ ]:
```
<Axes: ylabel='Base Pay'>
```



In [ ]:
```
sns.pairplot(df)
```

Out[ ]:
```
<seaborn.axisgrid.PairGrid at 0x246cb88e6b0>
```



In [ ]:
```
encodedDf = df.copy()
encodedDf.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 14470 entries, 0 to 14469
Data columns (total 5 columns):
 #   Column             Non-Null Count  Dtype
---  ------             --------------  -----
 0   Year               14470 non-null  int64
 1   Primary Job Title  14470 non-null  object
 2   Base Pay           14470 non-null  float64
 3   Department         14470 non-null  object
 4   College            14470 non-null  object
dtypes: float64(1), int64(1), object(3)
memory usage: 565.4+ KB
```

In [ ]:
```
from sklearn.preprocessing import LabelEncoder
le = LabelEncoder()
```

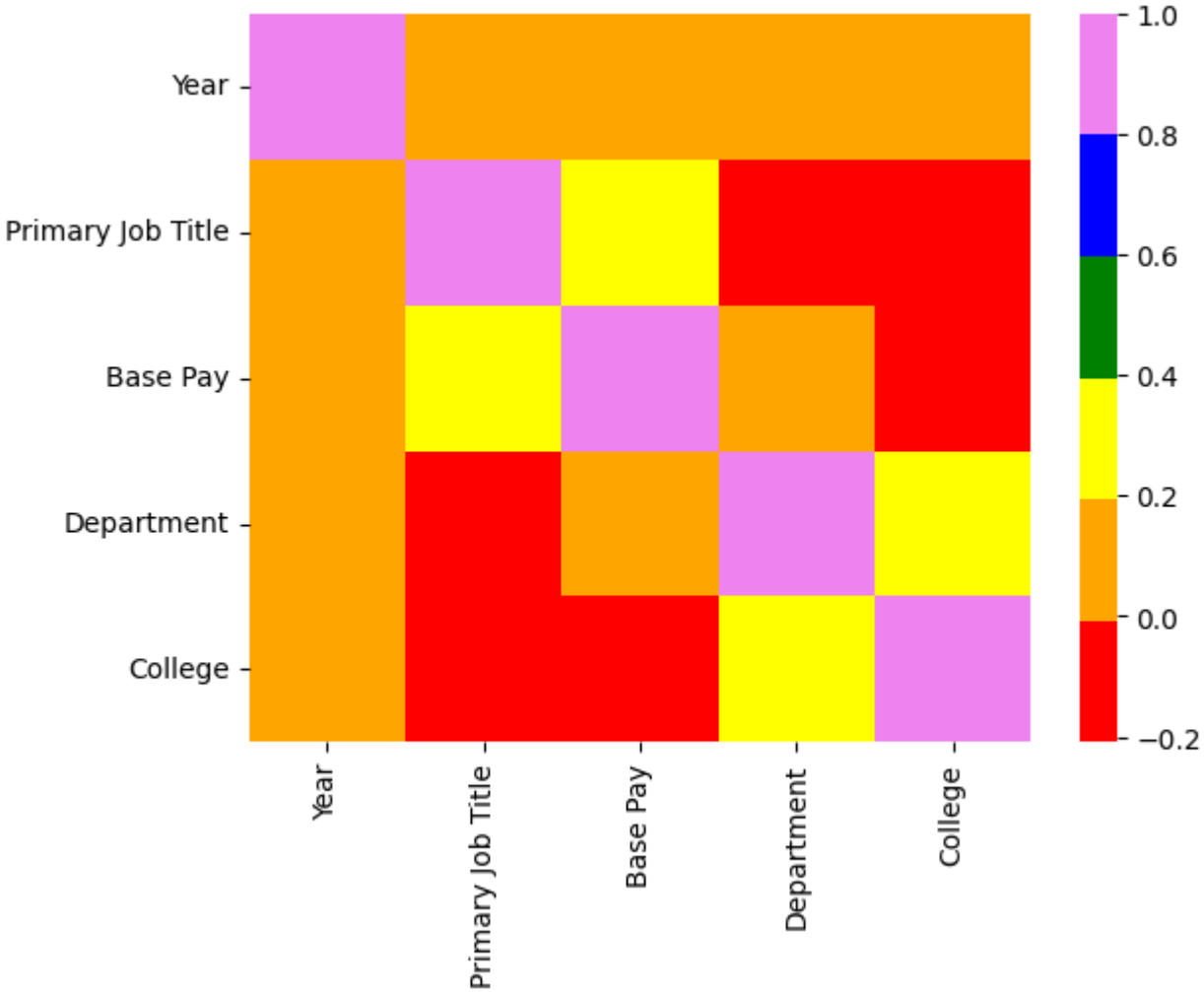
In [ ]:
```
encodedDf['College'] = le.fit_transform(encodedDf['College'])
encodedDf['Department'] = le.fit_transform(encodedDf['Department'])
encodedDf['Primary Job Title'] = le.fit_transform(encodedDf['Primary Job Title'])
encodedDf.corr()
```

Out[ ]:

	Year	Primary Job Title	Base Pay	Department	College
Year	1.000000	-0.002705	0.011595	0.003409	0.058346
Primary Job Title	-0.002705	1.000000	0.233902	-0.034217	-0.113989
Base Pay	0.011595	0.233902	1.000000	-0.005764	-0.209479
Department	0.003409	-0.034217	-0.005764	1.000000	0.222608
College	0.058346	-0.113989	-0.209479	0.222608	1.000000

In [ ]:
```
sns.heatmap(encodedDf.corr(), cmap=['red','orange','yellow','green','blue','violet'])
```

Out[ ]:
```
<Axes: >
```


Year
Primary Job Title
Base Pay
Department
College
Year
Primary Job Title
Base Pay
Department
College
1.0
0.8
0.6
0.4
0.2
0.0
−0.2