PRACTICAL 2

Aim : Extract the data from database using python (import and export data using Pandas library functions) and Demonstrate various data pre-processing techniques for a given dataset. Then build ML linear regression model.

List out directory contents

In [36]:
```
import os
print(os.listdir("."))
```

```
['.git', '.ipynb_checkpoints', 'endcode_check.py', 'Practical1.ipynb', 'Practical1.ipynb - Colaboratory.pdf', 'Practical1.pdf', 'Practical2.ipynb', 'random_dataset.csv', 'salary_data.csv']
```

Let's us build model follwing linear regression

Import necessary libraries

In [21]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import plotly.express as px
from sklearn.linear_model import LinearRegression
from sklearn.model_selection import train_test_split
from sklearn.metrics import r2_score,mean_squared_error
```

In [22]:
```
data = pd.read_csv(r"salary_Data.csv")
```

In [23]:
```
data.isna().sum()
```

Out[23]:
```
YearsExperience    0
Salary             0
dtype: int64
```

In [24]:
```
data.head()
```

Out[24]:

	YearsExperience	Salary
0	1.1	39343
1	1.3	46205
2	1.5	37731
3	2.0	43525
4	2.2	39891

In [25]:
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 30 entries, 0 to 29
Data columns (total 2 columns):
 #   Column           Non-Null Count  Dtype  
---  ------           --------------  -----  
 0   YearsExperience  30 non-null     float64
 1   Salary           30 non-null     int64  
dtypes: float64(1), int64(1)
memory usage: 608.0 bytes
```

In [26]:
```
data.describe()
```

Out[26]:

	YearsExperience	Salary
count	30.000000	30.000000
mean	5.313333	76003.000000
std	2.837888	27414.429785
min	1.100000	37731.000000
25%	3.200000	56720.750000
50%	4.700000	65237.000000
75%	7.700000	100544.750000
max	10.500000	122391.000000

In [29]:
```
fig = px.line(data,x="YearsExperience", y="Salary",markers=True,width=500, height=500)
fig.show()
```

Fitting the model

In [32]:
```
lr = LinearRegression()
lr.fit(x_train,y_train)
```

Out[32]:
```
▾ LinearRegression
LinearRegression()
```

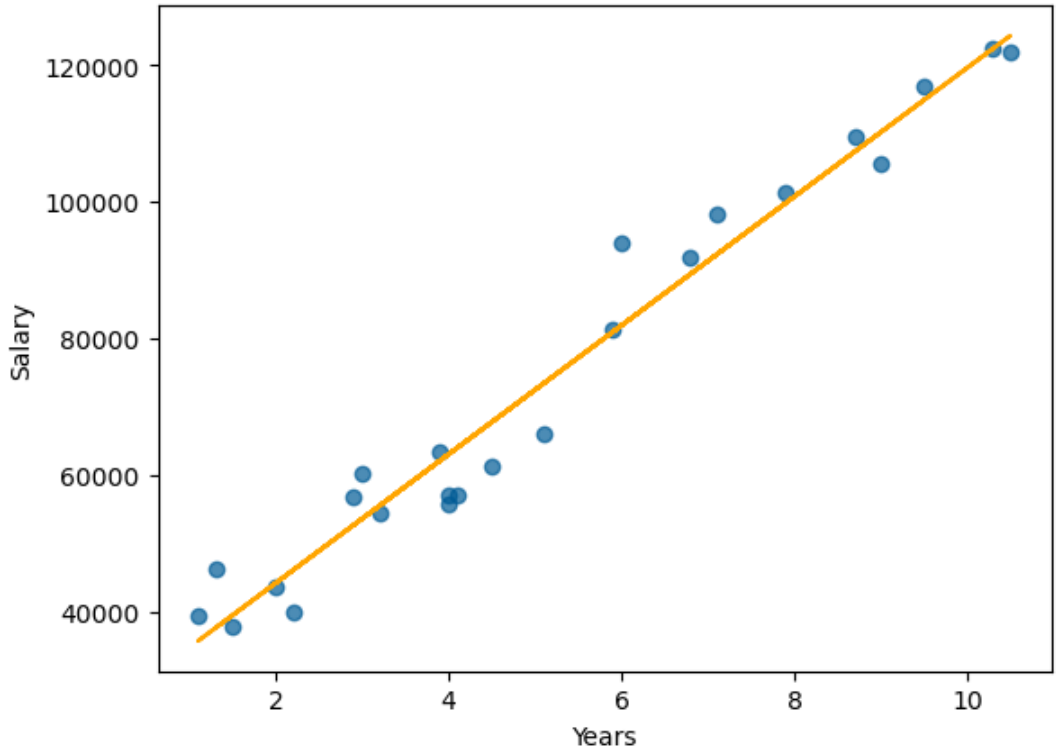
Split the data for train and test

In [31]:
```
x_train,x_test,y_train,y_test = train_test_split(x,y,test_size=0.2,random_state=42)
```

In [30]:
```
x = data['YearsExperience'].values.reshape(-1,1)
y = data['Salary'].values.reshape(-1,1)
```

again plotting prediction curve

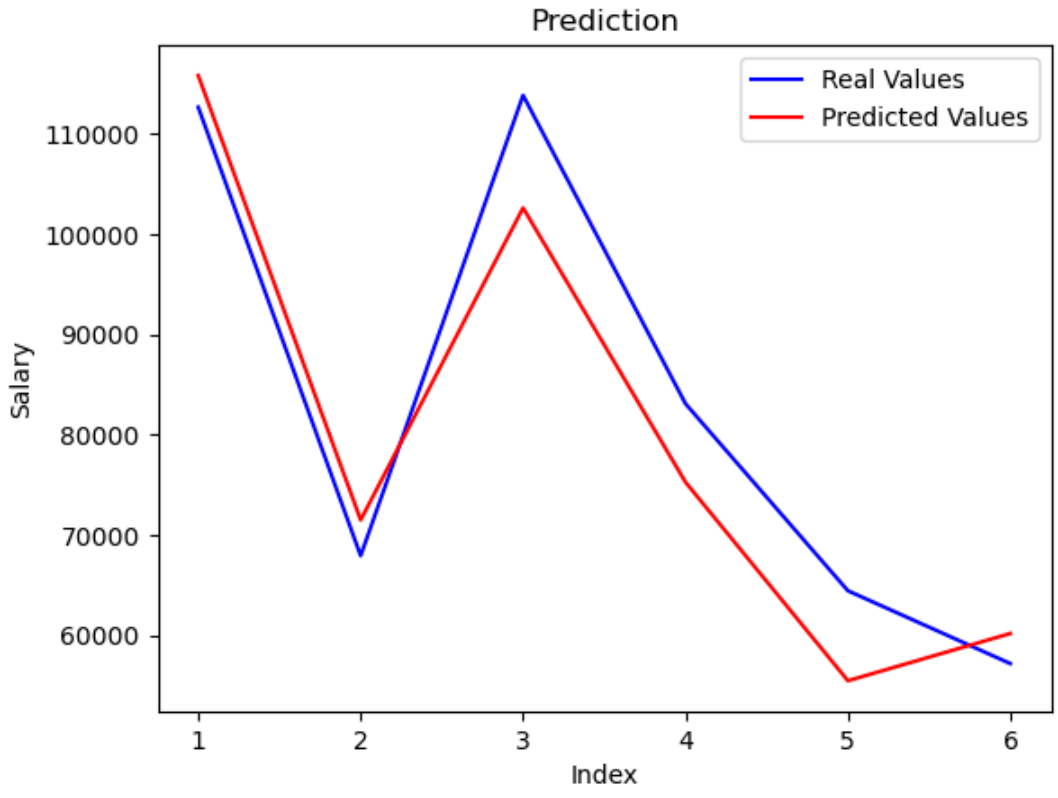
In [33]:
```
plt.scatter(x_train, y_train, color = '#005b96', alpha= 0.7)
plt.plot(x_train, lr.predict(x_train), color = "orange")
plt.xlabel('Years')
plt.ylabel('Salary')
plt.show()
```



In [34]:
```
y_head = lr.predict(x_test)
```

Plotting the actual and predicted values

In [35]:
```
c = [i for i in range (1,len(y_test)+1,1)]
plt.plot(c,y_test,color='b',linestyle='-',label="Real Values")
plt.plot(c,y_head,color='r',linestyle='-',label="Predicted Values")
plt.xlabel('Index')
plt.ylabel('Salary')
plt.title('Prediction')
plt.legend()
plt.show()
```



Doing same opertions on randomly generated dataset

1: Created a random dataset and exported it to a CSV file

In [4]:
```
np.random.seed(42)
data = {
    'X': np.random.rand(100) * 10,
    'Y': 3 * np.random.rand(100) * 10 + 2 + np.random.randn(100) * 2
}
df = pd.DataFrame(data)
df.to_csv('random_dataset.csv', index=False)
```

2: Simulate extracting data from a database using Pandas

In [5]:
```
df_from_csv = pd.read_csv('random_dataset.csv')
```

In [6]:
```
print("Extracted Dataset:")
print(df_from_csv.head())
```

```
Extracted Dataset:
          X          Y
0  3.745401   1.582826
1  9.507143  21.556820
2  7.319939  12.016824
3  5.986585  15.828418
4  1.560186  32.958543
```

3: Data Pre-processing Techniques

In this example, we won't have any missing values, so we'll focus on feature selection.

In [7]:
```
X = df_from_csv[['X']]
y = df_from_csv['Y']
```

4: Build a Linear Regression Model

i) Split the data into training and testing sets

In [10]:
```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

Initialize the Linear Regression model

In [13]:
```
model = LinearRegression()
```

Fit the model to the training data

In [15]:
```
model.fit(X_train, y_train)
```

Out[15]:
```
LinearRegression
LinearRegression()
```

Make predictions on the test set

In [16]:
```
y_pred = model.predict(X_test)
```

Evaluate the model

In [17]:
```
mse = mean_squared_error(y_test, y_pred)
print(f"\nMean Squared Error: {mse}")
```

```
Mean Squared Error: 93.8843765027589
```

Visualize the linear regression line

In [18]:
```
plt.scatter(X_test, y_test, color='blue', label='Actual data')
plt.plot(X_test, y_pred, color='red', linewidth=3, label='Linear Regression Line')
plt.title('Linear Regression Model')
plt.xlabel('X')
plt.ylabel('Y')
plt.legend()
plt.show()
```