In [15]:

```
import pandas as pd
from matplotlib import pyplot as plt
import seaborn as sns
import statsmodels.formula.api as smf
```

1. Importing Data

In [16]:

```
salary_data=pd.read_csv('Salary_Data.csv')
salary_data
```

Out[16]:

	YearsExperience	Salary
0	1.1	39343.0
1	1.3	46205.0
2	1.5	37731.0
3	2.0	43525.0
4	2.2	39891.0
5	2.9	56642.0
6	3.0	60150.0
7	3.2	54445.0
8	3.2	64445.0
9	3.7	57189.0
10	3.9	63218.0
11	4.0	55794.0
12	4.0	56957.0
13	4.1	57081.0
14	4.5	61111.0
15	4.9	67938.0
16	5.1	66029.0
17	5.3	83088.0
18	5.9	81363.0
19	6.0	93940.0
20	6.8	91738.0
21	7.1	98273.0
22	7.9	101302.0
23	8.2	113812.0
24	8.7	109431.0
25	9.0	105582.0
26	9.5	116969.0
27	9.6	112635.0
28	10.3	122391.0
29	10.5	121872.0

In [17]:

```
salary_data.shape
```

Out[17]:

```
(30, 2)
```

In [18]:

```
salary_data.head()
```

Out[18]:

	YearsExperience	Salary
0	1.1	39343.0
1	1.3	46205.0
2	1.5	37731.0
3	2.0	43525.0
4	2.2	39891.0

In [19]:

```
salary_data.isna().sum()
```

Out[19]:

```
YearsExperience    0
Salary             0
dtype: int64
```

In [20]:

```
salary_data.dtypes
```

Out[20]:

```
YearsExperience    float64
Salary             float64
dtype: object
```

In [21]:

```
salary_data.describe()
```

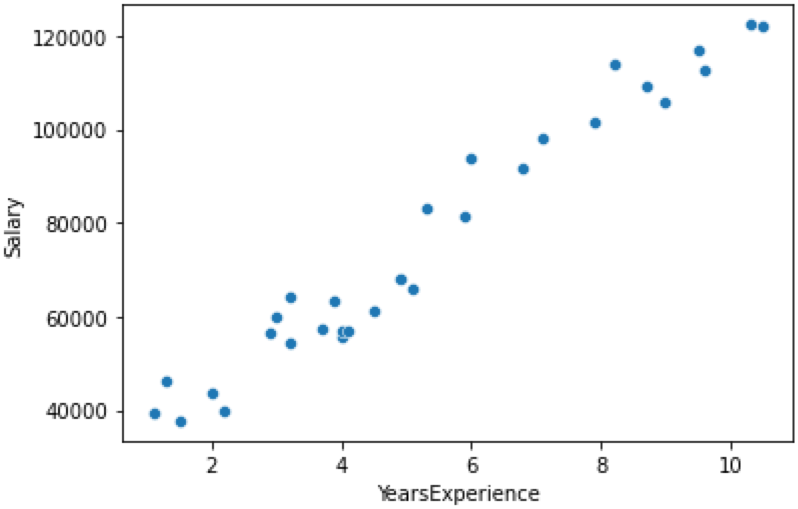
Out[21]:

	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

2. Check for assumotions

1. linearity check

In [22]:
```
sns.scatterplot(x='YearsExperience',y='Salary',data=salary_data)
plt.show()
```

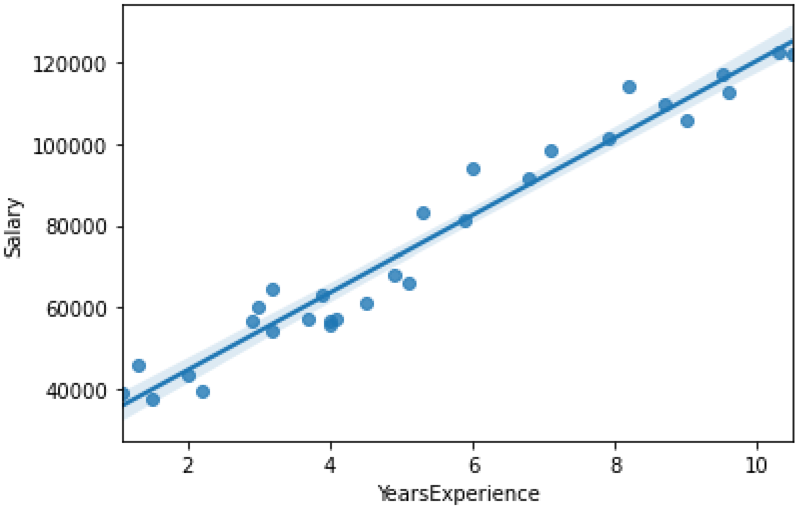


In [23]:
```
salary_data.corr()
```

Out[23]:

	YearsExperience	Salary
YearsExperience	1.000000	0.978242
Salary	0.978242	1.000000

In [24]:
```
sns.regplot(x='YearsExperience',y='Salary',data=salary_data)
plt.show()
```



3.Model Building || Model Training

In [25]:
```
#model training
lin_model=smf.ols(formula='Salary~YearsExperience',data=salary_data).fit()
lin_model
```

Out[25]:
```
<statsmodels.regression.linear_model.RegressionResultsWrapper at 0x17586620d30>
```

4.Model Testing

In [26]:
```
lin_model.params
```

Out[26]:
```
Intercept          25792.200199
YearsExperience     9449.962321
dtype: float64
```

In [27]:
```
lin_model.tvalues,lin_model.pvalues
```

Out[27]:
```
(Intercept           11.346940
 YearsExperience    24.950094
 dtype: float64,
 Intercept          5.511950e-12
 YearsExperience    1.143068e-20
 dtype: float64)
```

In [28]:
```
lin_model.rsquared
```

Out[28]:
```
0.9569566641435086
```

In [29]:
```
# Machine prediction
pred_data={'YearsExperience':[2,5,7]}
pred_data
```

Out[29]:
```
{'YearsExperience': [2, 5, 7]}
```

In [30]:
```
test_data=pd.DataFrame(data=pred_data)
test_data
```

Out[30]:

	YearsExperience
0	2
1	5
2	7

In [31]:
```
lin_model.predict(test_data)
```

Out[31]:
```
0    44692.124842
1    73042.011806
2    91941.936449
dtype: float64
```