In [27]:

```
import pandas as pd
```

In [30]:

```
data =pd.read_csv("HousingData.csv")
```

In [31]:

```
data
```

Out[31]:

	CRIM	ZN	INDUS	CHAS	NOX	RM	AGE	DIS	RAD	TAX	PTRATIO	B	LSTAT	M
0	0.00632	18.0	2.31	0.0	0.538	6.575	65.2	4.0900	1	296	15.3	396.90	4.98	
1	0.02731	0.0	7.07	0.0	0.469	6.421	78.9	4.9671	2	242	17.8	396.90	9.14	
2	0.02729	0.0	7.07	0.0	0.469	7.185	61.1	4.9671	2	242	17.8	392.83	4.03	
3	0.03237	0.0	2.18	0.0	0.458	6.998	45.8	6.0622	3	222	18.7	394.63	2.94	
4	0.06905	0.0	2.18	0.0	0.458	7.147	54.2	6.0622	3	222	18.7	396.90	NaN	
...	...	...	...	...	...	...	...	...	...	...	...	...	...	
501	0.06263	0.0	11.93	0.0	0.573	6.593	69.1	2.4786	1	273	21.0	391.99	NaN	
502	0.04527	0.0	11.93	0.0	0.573	6.120	76.7	2.2875	1	273	21.0	396.90	9.08	
503	0.06076	0.0	11.93	0.0	0.573	6.976	91.0	2.1675	1	273	21.0	396.90	5.64	
504	0.10959	0.0	11.93	0.0	0.573	6.794	89.3	2.3889	1	273	21.0	393.45	6.48	
505	0.04741	0.0	11.93	0.0	0.573	6.030	NaN	2.5050	1	273	21.0	396.90	7.88	

506 rows × 14 columns

In [32]:

```
data.head(2)
```

Out[32]:

	CRIM	ZN	INDUS	CHAS	NOX	RM	AGE	DIS	RAD	TAX	PTRATIO	B	LSTAT	MEDV
0	0.00632	18.0	2.31	0.0	0.538	6.575	65.2	4.0900	1	296	15.3	396.9	4.98	24.0
1	0.02731	0.0	7.07	0.0	0.469	6.421	78.9	4.9671	2	242	17.8	396.9	9.14	21.6

In [33]:

```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 506 entries, 0 to 505
Data columns (total 14 columns):
 #   Column   Non-Null Count  Dtype  
---  ------   --------------  -----  
 0   CRIM     486 non-null    float64
 1   ZN       486 non-null    float64
 2   INDUS    486 non-null    float64
 3   CHAS     486 non-null    float64
 4   NOX      506 non-null    float64
 5   RM       506 non-null    float64
```

```
 6   AGE      486 non-null    float64
 7   DIS      506 non-null    float64
 8   RAD      506 non-null    int64
 9   TAX      506 non-null    int64
 10  PTRATIO  506 non-null    float64
 11  B        506 non-null    float64
 12  LSTAT    486 non-null    float64
 13  MEDV     506 non-null    float64
dtypes: float64(12), int64(2)
memory usage: 55.5 KB
```

In [34]:

```
data.describe()
```

Out[34]:

	CRIM	ZN	INDUS	CHAS	NOX	RM	AGE	DIS
count	486.000000	486.000000	486.000000	486.000000	506.000000	506.000000	486.000000	506.000000
mean	3.611874	11.211934	11.083992	0.069959	0.554695	6.284634	68.518519	3.795043
std	8.720192	23.388876	6.835896	0.255340	0.115878	0.702617	27.999513	2.105710
min	0.006320	0.000000	0.460000	0.000000	0.385000	3.561000	2.900000	1.129600
25%	0.081900	0.000000	5.190000	0.000000	0.449000	5.885500	45.175000	2.100175
50%	0.253715	0.000000	9.690000	0.000000	0.538000	6.208500	76.800000	3.207450
75%	3.560263	12.500000	18.100000	0.000000	0.624000	6.623500	93.975000	5.188425
max	88.976200	100.000000	27.740000	1.000000	0.871000	8.780000	100.000000	12.126500

In [40]:

```
data.corr()
```

Out[40]:

	CRIM	ZN	INDUS	CHAS	NOX	RM	AGE	DIS	RA
CRIM	1.000000	-0.191178	0.401863	-0.054355	0.417130	-0.219150	0.354342	-0.374166	0.6247
ZN	-0.191178	1.000000	-0.531871	-0.037229	-0.513704	0.320800	-0.563801	0.656739	-0.3109
INDUS	0.401863	-0.531871	1.000000	0.059859	0.764866	-0.390234	0.638431	-0.711709	0.6045
CHAS	-0.054355	-0.037229	0.059859	1.000000	0.075097	0.104885	0.078831	-0.093971	0.0014
NOX	0.417130	-0.513704	0.764866	0.075097	1.000000	-0.302188	0.731548	-0.769230	0.6114
RM	-0.219150	0.320800	-0.390234	0.104885	-0.302188	1.000000	-0.247337	0.205246	-0.2098
AGE	0.354342	-0.563801	0.638431	0.078831	0.731548	-0.247337	1.000000	-0.744844	0.4583
DIS	-0.374166	0.656739	-0.711709	-0.093971	-0.769230	0.205246	-0.744844	1.000000	-0.4945
RAD	0.624765	-0.310919	0.604533	0.001468	0.611441	-0.209847	0.458349	-0.494588	1.0000
TAX	0.580595	-0.312371	0.731055	-0.032304	0.668023	-0.292048	0.509114	-0.534432	0.9102
PTRATIO	0.281110	-0.414046	0.390954	-0.111304	0.188933	-0.355501	0.269226	-0.232471	0.4647
B	-0.381411	0.171303	-0.360532	0.051264	-0.380051	0.128069	-0.275303	0.291512	-0.4444
LSTAT	0.444943	-0.414193	0.590690	-0.047424	0.582641	-0.614339	0.602891	-0.493328	0.4795
MEDV	-0.391363	0.373136	-0.481772	0.181391	-0.427321	0.695360	-0.394656	0.249929	-0.3816

In [56]:

```
from seaborn import scatterplot
```

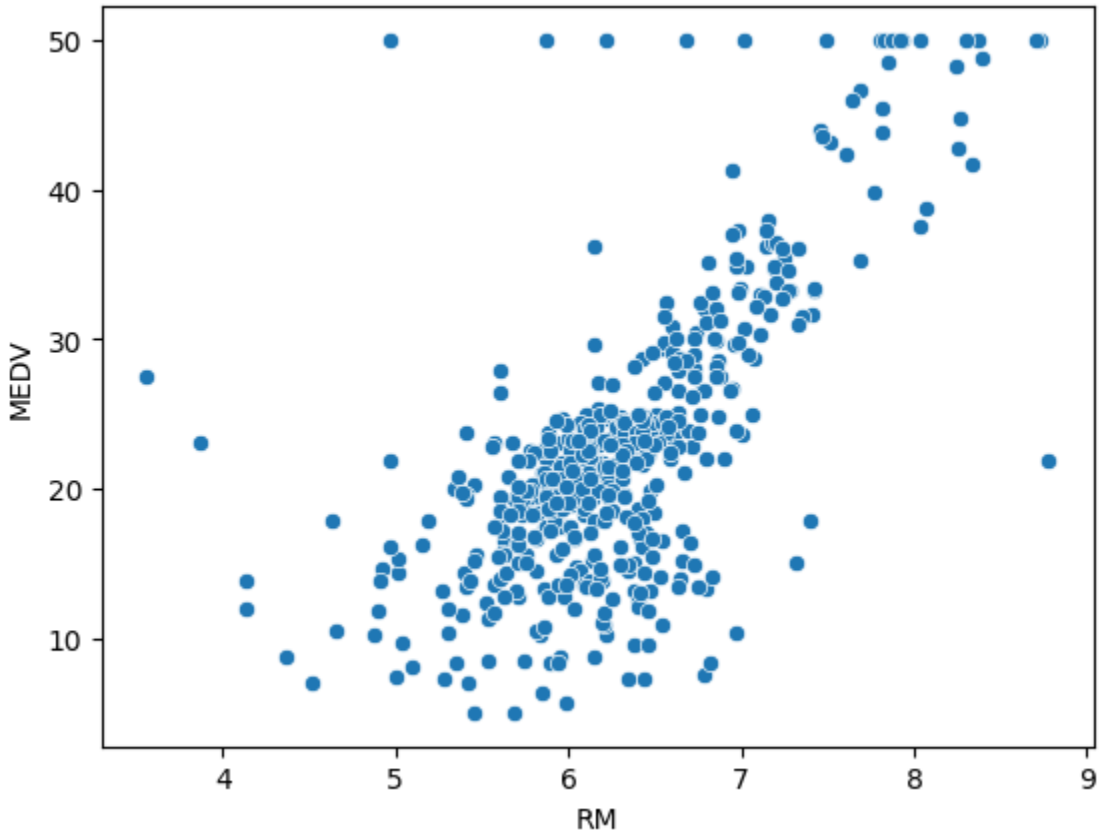
In [57]:

```
X = data["RM"]  # Extracting as a Series
Y = data["MEDV"]  # Extracting as a Series

scatterplot(x=X, y=Y)
```

Out[57]:

```
<Axes: xlabel='RM', ylabel='MEDV'>
```



In [44]:

```
from sklearn.model_selection import train_test_split
xtrain,xtest,ytrain,ytest=train_test_split(X,Y,test_size=0.2)
```

In [45]:

```
from sklearn.linear_model import LinearRegression
lm = LinearRegression()
model = lm.fit(xtrain,ytrain)
```

In [46]:

```
b1 = model.coef_
```

In [47]:

```
b0 = model.intercept_
```

In [48]:

```
b0
```

Out[48]:
```
array([-32.94106213])
```

In [49]:
```
b1
```

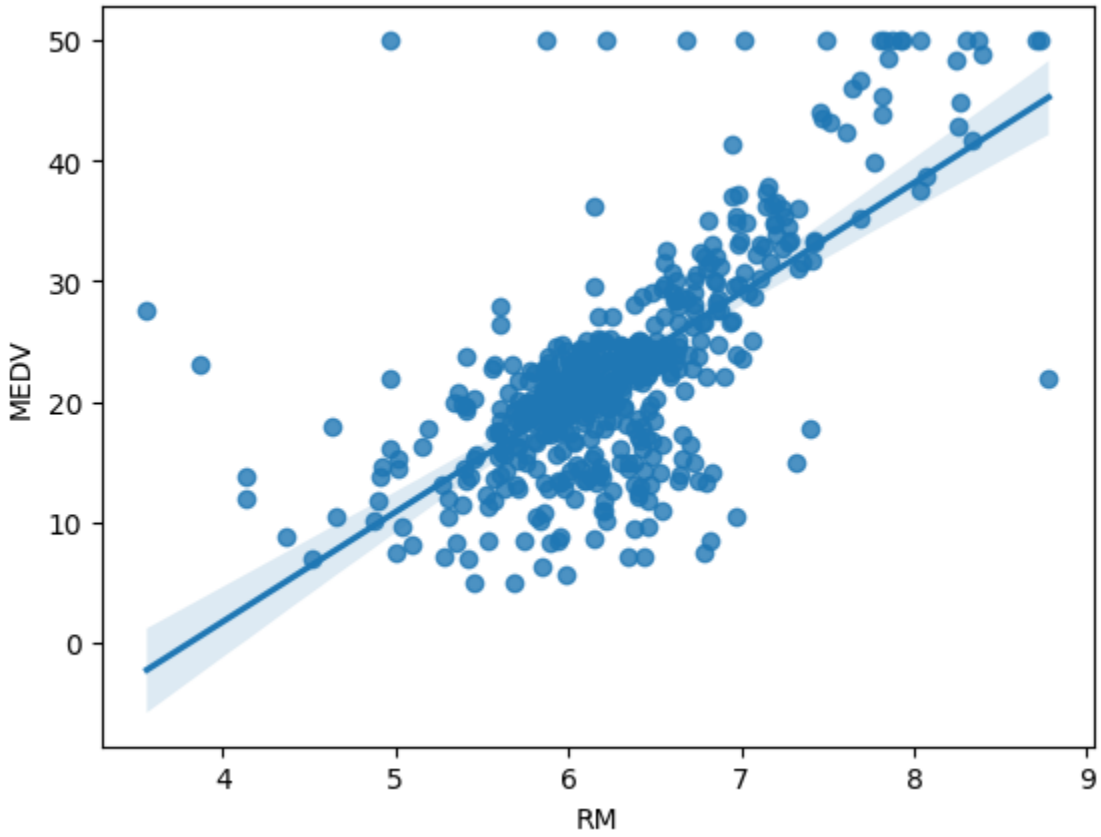
Out[49]:
```
array([[8.82738921]])
```

In [58]:
```
from seaborn import regplot

X = data["RM"]  # Extracting as a Series
Y = data["MEDV"]  # Extracting as a Series

regplot(x=X, y=Y)  # Correct parameter usage
```

Out[58]:
```
<Axes: xlabel='RM', ylabel='MEDV'>
```



In [51]:
```
pred = model.predict(xtest)
```

In [52]:
```
from sklearn.metrics import mean_absolute_error
mean_absolute_error(ytest,pred)
```

Out[52]:
```
4.600887017035727
```

In [53]:
```
from sklearn.metrics import mean_squared_error
mean_squared_error(ytest,pred)
```

Out[53]:

```
45.19963107244956
```

In [54]:

```
print(ytest)
```

```
     MEDV
374  13.8
190  37.0
383  12.3
284  32.2
129  14.3
..    ...
385   7.2
231  31.7
202  42.3
264  36.5
223  30.1

[102 rows x 1 columns]
```

In [55]:

```
print(pred)
```

```
[[ 3.58667443]
 [28.41812029]
 [15.78612633]
 [29.62747261]
 [16.81893086]
 [28.91245409]
 [26.44078511]
 [25.25791495]
 [21.92998922]
 [23.88966962]
 [23.84553268]
 [21.10021463]
 [16.64238308]
 [10.54265713]
 [22.64500774]
 [17.41919333]
 [21.27676242]
 [23.8278779 ]
 [23.30706194]
 [24.86950983]
 [19.67900497]
 [23.34237149]
 [19.5730763 ]
 [20.19982093]
 [21.08255985]
 [26.82036284]
 [19.37004635]
 [22.98044853]
 [20.0585827 ]
 [24.04856263]
 [24.30455692]
 [16.50997224]
 [23.65133011]
 [17.11023471]
 [19.09639728]
 [23.03341287]
 [13.87941026]
```

```
[23.39533583]
[39.96434538]
[23.92497918]
[23.59836578]
[14.96517913]
[17.21616338]
[20.26161266]
[33.16725569]
[26.74974373]
[32.52285627]
[18.94633166]
[10.33962718]
[21.14435158]
[28.63880502]
[21.74461404]
[23.51009189]
[36.1862228 ]
[21.3738637 ]
[25.80521308]
[25.27556973]
[36.38042536]
[16.91603214]
[10.36610935]
[28.97424581]
[19.89086231]
[14.85042307]
[20.10271965]
[22.23012045]
[18.13421186]
[21.01194074]
[18.37255136]
[31.61363518]
[34.95038831]
[22.80390075]
[26.54671378]
[27.54420876]
[28.67411458]
[26.91746412]
[29.76871084]
[21.88585227]
[40.65288174]
[27.08518452]
[31.73721863]
[22.6803173 ]
[19.81141581]
[20.86187512]
[21.30324458]
[24.09269958]
[24.18097347]
[25.01074805]
[19.38770112]
[26.92629151]
[20.07623748]
[36.57462792]
[43.89253358]
[25.23143278]
[28.56818591]
[21.25910764]
[29.0625197 ]
```

```
[18.58440871]
[13.64107075]
[32.48754672]
[34.23536978]
[30.66910454]
[25.47859968]]
```

In [ ]: