In [1]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import warnings
warnings.filterwarnings("ignore")
```

In [2]:
```
df=pd.read_csv("50_Startups.csv")
```

In [3]:
```
df
```

Out[3]:

	R&D Spend	Administration	Marketing Spend	State	Profit
0	165349.20	136897.80	471784.10	New York	192261.83
1	162597.70	151377.59	443898.53	California	191792.06
2	153441.51	101145.55	407934.54	Florida	191050.39
3	144372.41	118671.85	383199.62	New York	182901.99
4	142107.34	91391.77	366168.42	Florida	166187.94
5	131876.90	99814.71	362861.36	New York	156991.12
6	134615.46	147198.87	127716.82	California	156122.51
7	130298.13	145530.06	323876.68	Florida	155752.60
8	120542.52	148718.95	311613.29	New York	152211.77
9	123334.88	108679.17	304981.62	California	149759.96
10	101913.08	110594.11	229160.95	Florida	146121.95
11	100671.96	91790.61	249744.55	California	144259.40
12	93863.75	127320.38	249839.44	Florida	141585.52
13	91992.39	135495.07	252664.93	California	134307.35
14	119943.24	156547.42	256512.92	Florida	132602.65
15	114523.61	122616.84	261776.23	New York	129917.04
16	78013.11	121597.55	264346.06	California	126992.93
17	94657.16	145077.58	282574.31	New York	125370.37
18	91749.16	114175.79	294919.57	Florida	124266.90
19	86419.70	153514.11	0.00	New York	122776.86
20	76253.86	113867.30	298664.47	California	118474.03
21	78389.47	153773.43	299737.29	New York	111313.02
22	73994.56	122782.75	303319.26	Florida	110352.25
23	67532.53	105751.03	304768.73	Florida	108733.99
24	77044.01	99281.34	140574.81	New York	108552.04
25	64664.71	139553.16	137962.62	California	107404.34
26	75328.87	144135.98	134050.07	Florida	105733.54
27	72107.60	127864.55	353183.81	New York	105008.31
28	66051.52	182645.56	118148.20	Florida	103282.38
29	65605.48	153032.06	107138.38	New York	101004.64
30	61994.48	115641.28	91131.24	Florida	99937.59
31	61136.38	152701.92	88218.23	New York	97483.56
32	63408.86	129219.61	46085.25	California	97427.84
33	55493.95	103057.49	214634.81	Florida	96778.92
34	46426.07	157693.92	210797.67	California	96712.80
35	46014.02	85047.44	205517.64	New York	96479.51
36	28663.76	127056.21	201126.82	Florida	90708.19
37	44069.95	51283.14	197029.42	California	89949.14
38	20229.59	65947.93	185265.10	New York	81229.06
39	38558.51	82982.09	174999.30	California	81005.76
40	28754.33	118546.05	172795.67	California	78239.91
41	27892.92	84710.77	164470.71	Florida	77798.83
42	23640.93	96189.63	148001.11	California	71498.49
43	15505.73	127382.30	35534.17	New York	69758.98
44	22177.74	154806.14	28334.72	California	65200.33
45	1000.23	124153.04	1903.93	New York	64926.08
46	1315.46	115816.21	297114.46	Florida	49490.75
47	0.00	135426.92	0.00	California	42559.73
48	542.05	51743.15	0.00	New York	35673.41
49	0.00	116983.80	45173.06	California	14681.40

In [4]:
```
df.head(10)
```

Out[4]:

	R&D Spend	Administration	Marketing Spend	State	Profit
0	165349.20	136897.80	471784.10	New York	192261.83
1	162597.70	151377.59	443898.53	California	191792.06
2	153441.51	101145.55	407934.54	Florida	191050.39
3	144372.41	118671.85	383199.62	New York	182901.99
4	142107.34	91391.77	366168.42	Florida	166187.94
5	131876.90	99814.71	362861.36	New York	156991.12
6	134615.46	147198.87	127716.82	California	156122.51
7	130298.13	145530.06	323876.68	Florida	155752.60
8	120542.52	148718.95	311613.29	New York	152211.77
9	123334.88	108679.17	304981.62	California	149759.96

In [5]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 50 entries, 0 to 49
Data columns (total 5 columns):
 #   Column           Non-Null Count  Dtype  
---  ------           --------------  -----  
 0   R&D Spend        50 non-null     float64
 1   Administration   50 non-null     float64
 2   Marketing Spend  50 non-null     float64
 3   State            50 non-null     object 
 4   Profit           50 non-null     float64
dtypes: float64(4), object(1)
memory usage: 2.1+ KB
```

In [6]:
```
df.describe()
```

Out[6]:

	R&D Spend	Administration	Marketing Spend	Profit
count	50.000000	50.000000	50.000000	50.000000
mean	73721.615600	121344.639600	211025.097800	112012.639200
std	45902.256482	28017.802755	122290.310726	40306.180338
min	0.000000	51283.140000	0.000000	14681.400000
25%	39936.370000	103730.875000	129300.132500	90138.902500
50%	73051.080000	122699.795000	212716.240000	107978.190000
75%	101602.800000	144842.180000	299469.085000	139765.977500
max	165349.200000	182645.560000	471784.100000	192261.830000

In [7]:
```
df.corr()
```

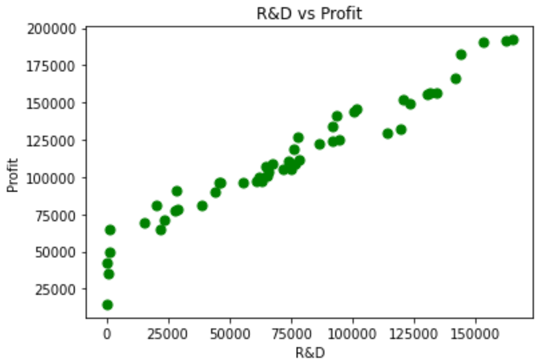
Out[7]:

	R&D Spend	Administration	Marketing Spend	Profit
R&D Spend	1.000000	0.241955	0.724248	0.972900
Administration	0.241955	1.000000	-0.032154	0.200717
Marketing Spend	0.724248	-0.032154	1.000000	0.747766
Profit	0.972900	0.200717	0.747766	1.000000

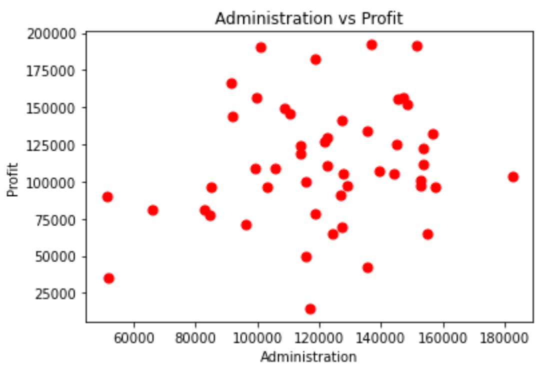
In [8]:
```
df.isnull().sum()
```

Out[8]:
```
R&D Spend          0
Administration     0
Marketing Spend    0
State              0
Profit             0
dtype: int64
```

In [9]:
```
#Plot R&D vs Profit............
x1 = df.iloc[:, 0].values
y1 = df.iloc[:, -1].values
plt.scatter(x1,y1,color='Green',s=50)
plt.xlabel('R&D')
plt.ylabel('Profit')
plt.title('R&D vs Profit')
plt.show()
```

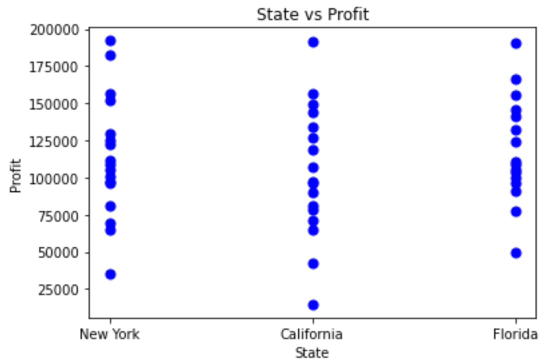


In [10]:
```
#Plot Administration vs Profit
x1 = df.iloc[:, 1].values
y1 = df.iloc[:, -1].values
plt.scatter(x1,y1,color='Red',s=50)
plt.xlabel('Administration')
plt.ylabel('Profit')
plt.title('Administration vs Profit')
plt.show()
```



In [11]:
```
#Plot Marketing Spend vs Profit
x1 = df.iloc[:, 2].values
y1 = df.iloc[:, -1].values
plt.scatter(x1,y1,color='Black',s=50)
plt.xlabel('Marketing Spend')
plt.ylabel('Profit')
plt.title('Marketing Spend vs Profit')
plt.show()
```

In [12]:
```
#High correlation between Marketing Spend and Profit.
#Plot State vs Profit
x1 = df.iloc[:, 3].values
y1 = df.iloc[:, -1].values
plt.scatter(x1,y1,color='Blue',s=50)
plt.xlabel('State')
plt.ylabel('Profit')
plt.title('State vs Profit')
plt.show()
```



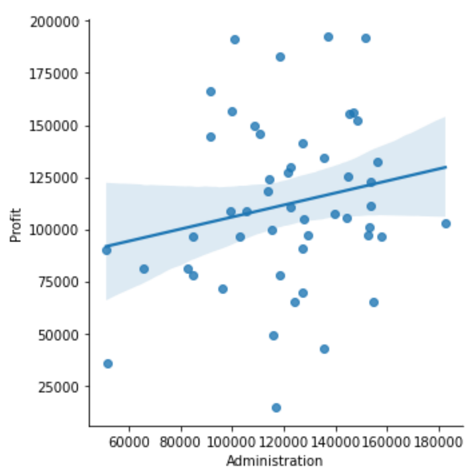
In [13]:
```
df.head()
```

Out[13]:

	R&D Spend	Administration	Marketing Spend	State	Profit
0	165349.20	136897.80	471784.10	New York	192261.83
1	162597.70	151377.59	443898.53	California	191792.06
2	153441.51	101145.55	407934.54	Florida	191050.39
3	144372.41	118671.85	383199.62	New York	182901.99
4	142107.34	91391.77	366168.42	Florida	166187.94

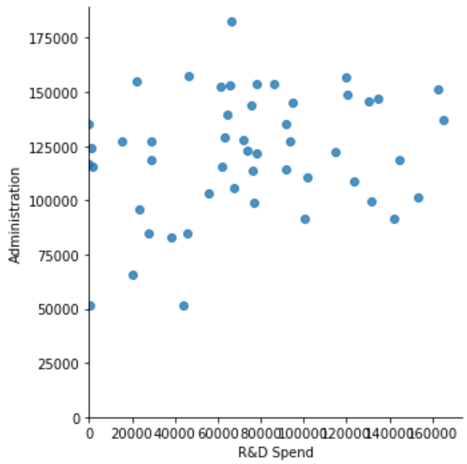
In [14]:
```
# Recommended way
sns.lmplot(x='Administration', y='Profit', data=df)

# Alternative way
# sns.lmplot(x=df.Administration, y=df.Profit)
```

Out[14]: <seaborn.axisgrid.FacetGrid at 0x27cafe5ecd0>



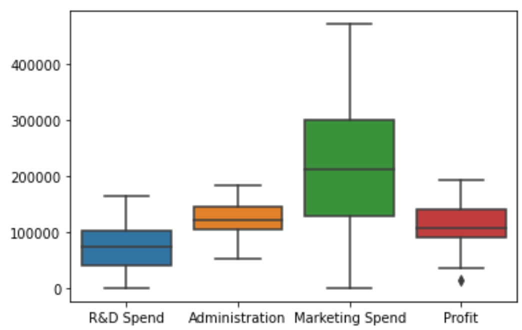
In [15]:
```
# Plot using Seaborn
sns.lmplot(x='R&D Spend', y='Administration', data=df, fit_reg=False)

# Tweak using Matplotlib
plt.ylim(0, None)
plt.xlim(0, None)
```

Out[15]: (0.0, 173616.66)



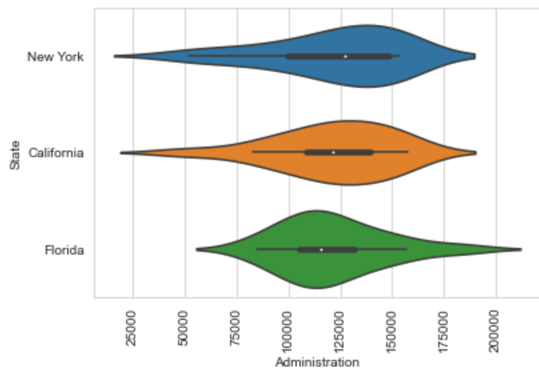
In [16]:
```
# Boxplot
sns.boxplot(data=df)
```

Out[16]: <AxesSubplot:>



In [17]:
```
# Set theme
sns.set_style('whitegrid')

# Violin plot
sns.violinplot(x='Administration', y='State', data=df)
plt.xticks(rotation=90)
```

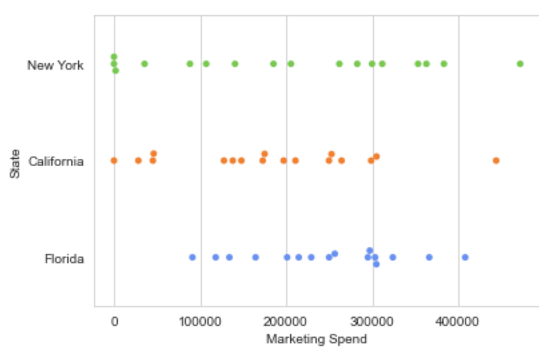
Out[17]:
```
(array([     0.,  25000.,  50000.,  75000., 100000., 125000., 150000.,
        175000., 200000., 225000.]),
 [Text(0, 0, ''),
  Text(0, 0, ''),
  Text(0, 0, ''),
  Text(0, 0, ''),
  Text(0, 0, ''),
  Text(0, 0, ''),
  Text(0, 0, ''),
  Text(0, 0, ''),
  Text(0, 0, ''),
  Text(0, 0, '')])
```



In [18]:
```
pkmn_type_colors = ['#78C850',  # Grass
                    '#F08030',  # Fire
                    '#6890F0',  # Water
                    '#A8B820',  # Bug
                    '#A8A878',  # Normal
                    '#A040A0',  # Poison
                    '#F8D030',  # Electric
                    '#E0C068',  # Ground
                    '#EE99AC',  # Fairy
                    '#C03028',  # Fighting
                    '#F85888',  # Psychic
                    '#B8A038',  # Rock
                    '#705898',  # Ghost
                    '#98D8D8',  # Ice
                    '#7038F8',  # Dragon
                   ]
```

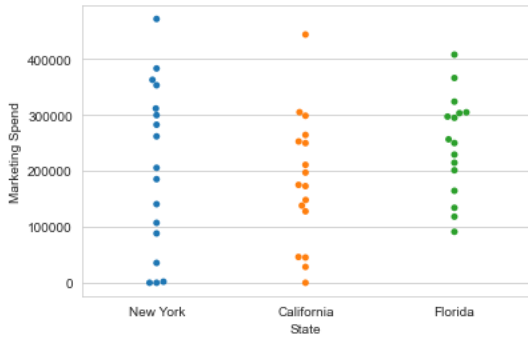
In [19]:
```
# Swarm plot with Pokemon color palette
sns.swarmplot(x='Marketing Spend', y='State', data=df,
              palette=pkmn_type_colors)
```

Out[19]: <AxesSubplot:xlabel='Marketing Spend', ylabel='State'>



In [20]:
```
# Swarmplot with melted_df
sns.swarmplot(x='State', y='Marketing Spend', data=df)
```

Out[20]: <AxesSubplot:xlabel='State', ylabel='Marketing Spend'>

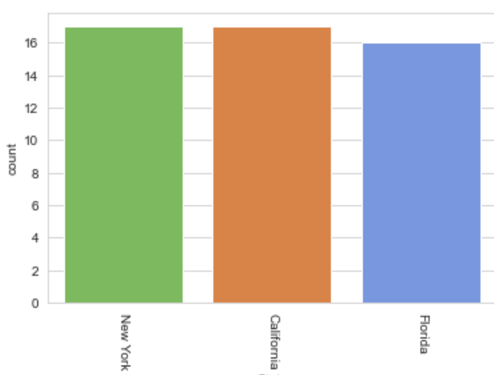


In [21]:
```
# Calculate correlations
corr = df.corr()

# Heatmap
sns.heatmap(corr)
```

Out[21]: <AxesSubplot:>

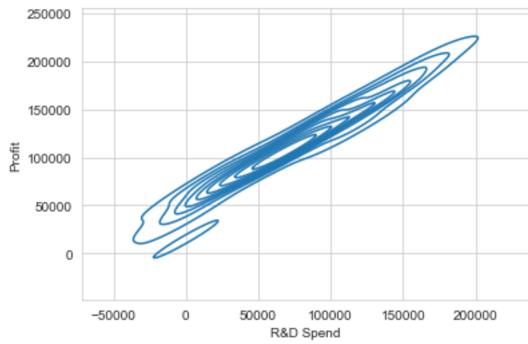
In [22]:
```
# Count Plot (a.k.a. Bar Plot)
sns.countplot(x='State', data=df, palette=pkmn_type_colors)

# Rotate x-labels
plt.xticks(rotation=-90)
```

Out[22]:
```
(array([0, 1, 2]),
 [Text(0, 0, 'New York'), Text(1, 0, 'California'), Text(2, 0, 'Florida')])
```



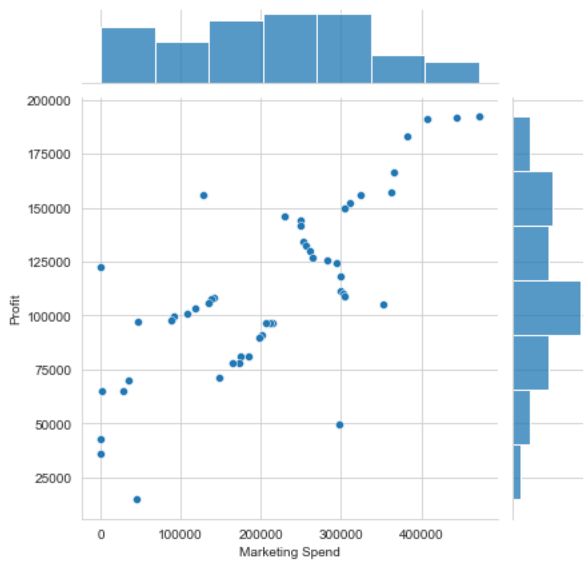
In [23]:
```
sns.kdeplot(df["R&D Spend"], df["Profit"])
```

Out[23]: <AxesSubplot:xlabel='R&D Spend', ylabel='Profit'>



In [24]:
```
# Joint Distribution Plot
sns.jointplot(x='Marketing Spend', y='Profit', data=df)
```

Out[24]: <seaborn.axisgrid.JointGrid at 0x27caf55f220>



In [25]:
```
df.head()
```

Out[25]:

	R&D Spend	Administration	Marketing Spend	State	Profit
0	165349.20	136897.80	471784.10	New York	192261.83
1	162597.70	151377.59	443898.53	California	191792.06
2	153441.51	101145.55	407934.54	Florida	191050.39
3	144372.41	118671.85	383199.62	New York	182901.99
4	142107.34	91391.77	366168.42	Florida	166187.94

Seperation of dependent and independent variables

In [26]:
```
x=df.iloc[:,:-2]
```

In [27]:
```
x
```

Out[27]:

	R&D Spend	Administration	Marketing Spend
0	165349.20	136897.80	471784.10
1	162597.70	151377.59	443898.53
2	153441.51	101145.55	407934.54
3	144372.41	118671.85	383199.62
4	142107.34	91391.77	366168.42
5	131876.90	99814.71	362861.36
6	134615.46	147198.87	127716.82
7	130298.13	145530.06	323876.68
8	120542.52	148718.95	311613.29
9	123334.88	108679.17	304981.62
10	101913.08	110594.11	229160.95
11	100671.96	91790.61	249744.55
12	93863.75	127320.38	249839.44
13	91992.39	135495.07	252664.93
14	119943.24	156547.42	256512.92
15	114523.61	122616.84	261776.23
16	78013.11	121597.55	264346.06
17	94657.16	145077.58	282574.31
18	91749.16	114175.79	294919.57
19	86419.70	153514.11	0.00
20	76253.86	113867.30	298664.47
21	78389.47	153773.43	299737.29
22	73994.56	122782.75	303319.26
23	67532.53	105751.03	304768.73
24	77044.01	99281.34	140574.81
25	64664.71	139553.16	137962.62
26	75328.87	144135.98	134050.07
27	72107.60	127864.55	353183.81
28	66051.52	182645.56	118148.20
29	65605.48	153032.06	107138.38
30	61994.48	115641.28	91131.24
31	61136.38	152701.92	88218.23
32	63408.86	129219.61	46085.25
33	55493.95	103057.49	214634.81
34	46426.07	157693.92	210797.67
35	46014.02	85047.44	205517.64
36	28663.76	127056.21	201126.82
37	44069.95	51283.14	197029.42
38	20229.59	65947.93	185265.10
39	38558.51	82982.09	174999.30
40	28754.33	118546.05	172795.67
41	27892.92	84710.77	164470.71
42	23640.93	96189.63	148001.11
43	15505.73	127382.30	35534.17
44	22177.74	154806.14	28334.72
45	1000.23	124153.04	1903.93
46	1315.46	115816.21	297114.46
47	0.00	135426.92	0.00
48	542.05	51743.15	0.00
49	0.00	116983.80	45173.06

In [28]:
```
y=df["Profit"]
```

In [29]:
```
y
```

Out[29]:
```
0     192261.83
1     191792.06
2     191050.39
3     182901.99
4     166187.94
5     156991.12
6     156122.51
7     155752.60
8     152211.77
9     149759.96
10    146121.95
11    144259.40
12    141585.52
13    134307.35
14    132602.65
15    129917.04
16    126992.93
17    125370.37
18    124266.90
19    122776.86
20    118474.03
21    111313.02
22    110352.25
23    108733.99
24    108552.04
25    107404.34
26    105733.54
27    105008.31
28    103282.38
29    101004.64
30     99937.59
31     97483.56
32     97427.84
33     96778.92
34     96712.80
35     96479.51
36     90708.19
37     89949.14
38     81229.06
39     81005.76
40     78239.91
41     77798.83
42     71498.49
43     69758.98
44     65200.33
45     64926.08
46     49490.75
47     42559.73
48     35673.41
49     14681.40
Name: Profit, dtype: float64
```

In [30]:
```
# Splitting the dataset into the Training set and Test set
from sklearn.model_selection import train_test_split
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size = 0.2, random_state = 0)
```

BUILDING OF MODEL

In [31]:
```
# Fitting Multiple Linear Regression to the Training set
from sklearn.linear_model import LinearRegression
regressor = LinearRegression()
regressor.fit(x_train, y_train)
```

Out[31]:
```
LinearRegression()
```

In [32]:
```
# Predicting the Test set results
y_pred = regressor.predict(x_test)
```

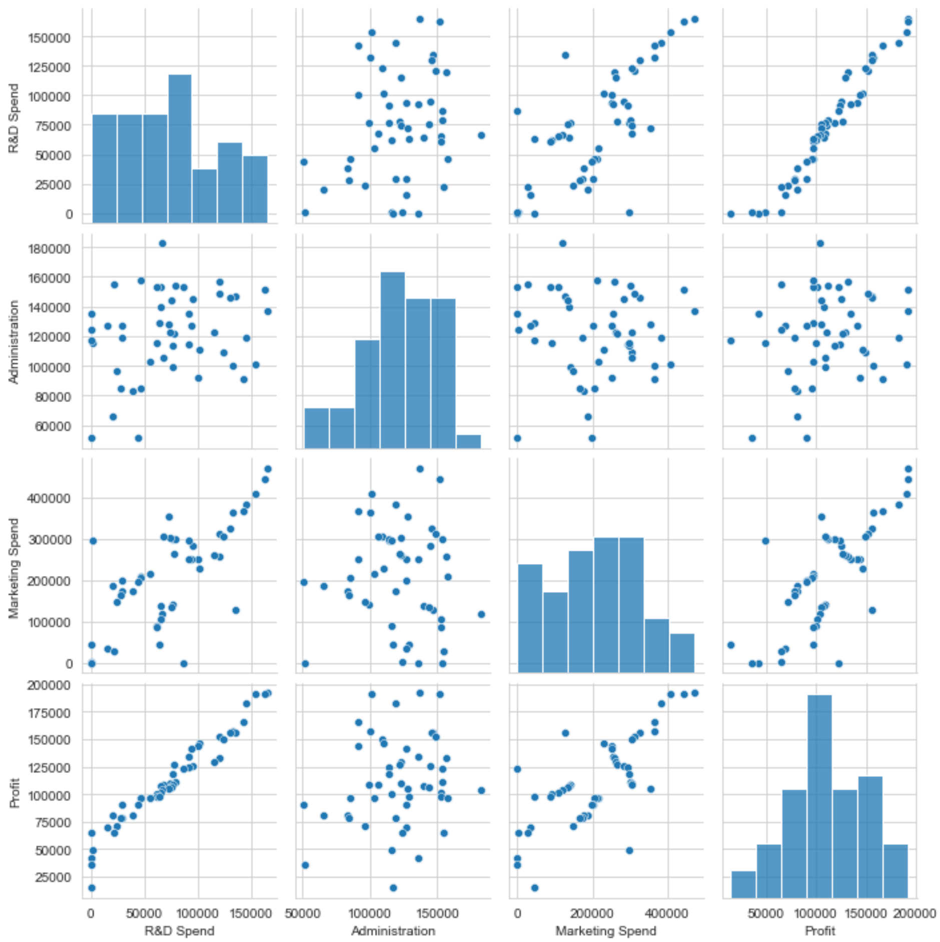
In [33]:
```
#evaluate the model
from sklearn.metrics import r2_score
```

In [34]:
```
r2_score(y_test,y_pred)
```

Out[34]:
```
0.9393955917820571
```

In [35]:
```
sns.pairplot(df)
```

Out[35]: <seaborn.axisgrid.PairGrid at 0x27cb16f2220>



In [36]:
```
# Print the dimensions of X and y
print(x.shape)
print(y.shape)
```

```
(50, 3)
(50,)
```

In [37]:
```
import pandas_profiling as pp
```

In [38]:
```
pp.ProfileReport(df)
```

Summarize dataset: 100% 30/30 [00:02<00:00, 12.34it/s, Completed]

Generate report structure: 100% 1/1 [00:01<00:00, 1.12s/it]

Render HTML: 100% 1/1 [00:00<00:00, 1.48it/s]

23640.93	1	2.0%
27892.92	1	2.0%
28663.76	1	2.0%

Value	Count	Frequency (%)
165349.2	1	2.0%
162597.7	1	2.0%
153441.51	1	2.0%
144372.41	1	2.0%
142107.34	1	2.0%
134615.46	1	2.0%
131876.9	1	2.0%
130298.13	1	2.0%
123334.88	1	2.0%
120542.52	1	2.0%

Interactions

Out[38]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]: