EDA PROJECT BY BISWARUP

In [316]:
```
# In this Project we have used top 1000 company's dataset from Kaggle and did Exploratory Data Analysis on it.
# The main moto was to clean the data and to answer these following questions

# 1. Top 10 companies with highest revenue.
# 2. Top 10 Companies with highest profit.
# 3. Top 10 Companies with highest assets.
# 4. Top 5 Companies with highest employees.
# 5. Top 10 Companies with lowest revenue.
# 6. Top 10 Companies with lowest profit.
# 7. Top 10 Companies with lowest assets.
# 8. Top 5 Companies with lowest employees.

#We plotted all these graphs as well.
```

In [3]:
```
#First we will import all the libraries that's necessary

import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

In [4]:
```
# Now lets import the csv that we will be working with

df = pd.read_csv(r"E:\Pandas\Fortune 1000 Companies by Revenue.csv")
```

In [5]:
```
# to view the csv file

df
```

Out[5]:

	rank	name	revenues	revenue_percent_change	profits	profits_percent_change	assets	market_value	change_in_rank	employees
0	1	Walmart	$572,754	2.40%	$13,673	1.20%	$244,860	$409,795	-	2,300,000
1	2	Amazon	$469,822	21.70%	$33,364	56.40%	$420,549	$1,658,807.30	-	1,608,000
2	3	Apple	$365,817	33.30%	$94,680	64.90%	$351,002	$2,849,537.60	-	154,000
3	4	CVS Health	$292,111	8.70%	$7,910	10.20%	$232,999	$132,839.20	-	258,000
4	5	UnitedHealth Group	$287,597	11.80%	$17,285	12.20%	$212,206	$479,830.30	-	350,000
...	...	...	...	...	...	...	...	...	...	...
995	996	Vizio Holding	$2,124	4%	($39.40)	-138.40%	$935.80	$1,705.10	-	800
996	997	1-800-Flowers.com	$2,122.20	42.50%	$118.70	101.10%	$1,076.70	$830	-	4,800
997	998	Cowen	$2,112.80	30.20%	$295.60	36.60%	$8,748.80	$744.10	-	1,534
998	999	Ashland Global Holdings	$2,111	-11.20%	$220	-	$6,612	$5,601.90	-130	4,100
999	1,000	DocuSign	$2,107.20	45%	($70)	-	$2,541.30	$21,302.80	-	7,461

1000 rows × 10 columns

In [13]:
```
# to see all the rows

pd.set_option('display.max.rows',999)
```

In [14]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1000 entries, 0 to 999
Data columns (total 10 columns):
 #   Column                  Non-Null Count  Dtype 
---  ------                  --------------  ----- 
 0   rank                    1000 non-null   object
 1   name                    1000 non-null   object
 2   revenues                1000 non-null   object
 3   revenue_percent_change  1000 non-null   object
 4   profits                 1000 non-null   object
 5   profits_percent_change  1000 non-null   object
 6   assets                  1000 non-null   object
 7   market_value            1000 non-null   object
 8   change_in_rank          1000 non-null   object
 9   employees               1000 non-null   object
dtypes: object(10)
memory usage: 78.3+ KB
```

In [18]:
```
df.dtypes
```

Out[18]:
```
rank                      object
name                      object
revenues                  object
revenue_percent_change    object
profits                   object
profits_percent_change    object
assets                    object
market_value              object
change_in_rank            object
employees                 object
dtype: object
```

In [47]:
```
df['assets'] = df['assets'].str.replace('$','')
df['assets'] = df['assets'].str.replace(',','')
df['assets'] = df['assets'].str.replace('.','')

df['assets'] = df['assets'].astype('int')
```

Out[47]:
```
1109781414
```

In [40]:
```
df.head(999)
```

Out[40]:

	rank	name	revenues	revenue_percent_change	profits	profits_percent_change	assets	market_value	change_in_rank	employees
0	1	Walmart	$572,754	2.40%	$13,673	1.20%	244860	$409,795	-	2,300,000
1	2	Amazon	$469,822	21.70%	$33,364	56.40%	420549	$1,658,807.30	-	1,608,000
2	3	Apple	$365,817	33.30%	$94,680	64.90%	351002	$2,849,537.60	-	154,000
3	4	CVS Health	$292,111	8.70%	$7,910	10.20%	232999	$132,839.20	-	258,000
4	5	UnitedHealth Group	$287,597	11.80%	$17,285	12.20%	212206	$479,830.30	-	350,000
5	6	Exxon Mobil	$285,640	57.40%	$23,040	-	338923	$349,652.40	4	63,000
6	7	Berkshire Hathaway	$276,094	12.50%	$89,795	111.20%	958784	$779,542.30	-1	372,000
7	8	Alphabet	$257,637	41.20%	$76,033	88.80%	359268	$1,842,326.10	1	156,500
8	9	McKesson	$238,228	3.10%	($4,539)	-604.30%	65015	$45,857.80	-2	67,500
9	10	AmerisourceBergen	$213,988.80	12.70%	$1,539.90	-	5733780	$32,355.70	-2	40,000
10	11	Costco Wholesale	$195,929	17.50%	$5,007	25.10%	59268	$255,230.70	1	288,000

In [61]:
```
#changed the column names for better understanding

df = df.rename(columns={'revenues ': 'Revenue_$Million', 'rank ': 'Rank', 'name ': 'Name', 'revenue_percent_change': 'Revenue_%Change', 'profits ': 'Profits_$Million', 'assets': 'Asset
```

In [62]:
```
df
```

Out[62]:

	Rank	Name	Revenue_$Million	Revenue_%Change	Profits_$Million	Profit_%Change	Assets_$Million	MarketValue_$Million	change_in_rank	Employees
0	1	Walmart	572,754	2.40%	$13,673	1.20%	244860	$409,795	-	2,300,000
1	2	Amazon	469,822	21.70%	$33,364	56.40%	420549	$1,658,807.30	-	1,608,000
2	3	Apple	365,817	33.30%	$94,680	64.90%	351002	$2,849,537.60	-	154,000
3	4	CVS Health	292,111	8.70%	$7,910	10.20%	232999	$132,839.20	-	258,000
4	5	UnitedHealth Group	287,597	11.80%	$17,285	12.20%	212206	$479,830.30	-	350,000
...	...	...	...	...	...	...	...	...	...	...
995	996	Vizio Holding	2,124	4%	($39.40)	-138.40%	93580	$1,705.10	-	800
996	997	1-800-Flowers.com	2,122.20	42.50%	$118.70	101.10%	107670	$830	-	4,800
997	998	Cowen	2,112.80	30.20%	$295.60	36.60%	874880	$744.10	-	1,534
998	999	Ashland Global Holdings	2,111	-11.20%	$220	-	6612	$5,601.90	-130	4,100
999	1,000	DocuSign	2,107.20	45%	($70)	-	254130	$21,302.80	-	7,461

1000 rows × 10 columns

In [65]:
```
#here we dropped the unnecessary column

df = df.drop('change_in_rank',axis = 1)
```

In [66]:
```
df
```

Out[66]:

	Rank	Name	Revenue_$Million	Revenue_%Change	Profits_$Million	Profit_%Change	Assets_$Million	MarketValue_$Million	Employees
0	1	Walmart	572,754	2.40%	$13,673	1.20%	244860	$409,795	2,300,000
1	2	Amazon	469,822	21.70%	$33,364	56.40%	420549	$1,658,807.30	1,608,000
2	3	Apple	365,817	33.30%	$94,680	64.90%	351002	$2,849,537.60	154,000
3	4	CVS Health	292,111	8.70%	$7,910	10.20%	232999	$132,839.20	258,000
4	5	UnitedHealth Group	287,597	11.80%	$17,285	12.20%	212206	$479,830.30	350,000
...	...	...	...	...	...	...	...	...	...
995	996	Vizio Holding	2,124	4%	($39.40)	-138.40%	93580	$1,705.10	800
996	997	1-800-Flowers.com	2,122.20	42.50%	$118.70	101.10%	107670	$830	4,800
997	998	Cowen	2,112.80	30.20%	$295.60	36.60%	874880	$744.10	1,534
998	999	Ashland Global Holdings	2,111	-11.20%	$220	-	6612	$5,601.90	4,100
999	1,000	DocuSign	2,107.20	45%	($70)	-	254130	$21,302.80	7,461

1000 rows × 9 columns

In [67]:
```
# Now lets see how the data looks

df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1000 entries, 0 to 999
Data columns (total 9 columns):
 #   Column               Non-Null Count  Dtype 
---  ------               --------------  ----- 
 0   Rank                 1000 non-null   object
 1   Name                 1000 non-null   object
 2   Revenue_$Million     1000 non-null   object
 3   Revenue_%Change      1000 non-null   object
 4   Profits_$Million     1000 non-null   object
 5   Profit_%Change       1000 non-null   object
 6   Assets_$Million      1000 non-null   int32 
 7   MarketValue_$Million 1000 non-null   object
 8   Employees            1000 non-null   object
dtypes: int32(1), object(8)
memory usage: 66.5+ KB
```

In [68]:
```
# Now we need to change the rank, revenue, profits, market value and employee to integer
```

In [69]:
```
#but before that we have to clean the strings
```

In [70]:
```
df['Revenue_$Million'] = df['Revenue_$Million'].str.replace('$','')
df['Revenue_$Million'] = df['Revenue_$Million'].str.replace(',','')
df['Revenue_$Million'] = df['Revenue_$Million'].str.replace('.','')
```

```
C:\Users\biswa\AppData\Local\Temp\ipykernel_7548\2364703093.py:1: FutureWarning: The default value of regex will change from True to False in a future version. In addition, single cha
racter regular expressions will *not* be treated as literal strings when regex=True.
  df['Revenue_$Million'] = df['Revenue_$Million'].str.replace('$','')
C:\Users\biswa\AppData\Local\Temp\ipykernel_7548\2364703093.py:3: FutureWarning: The default value of regex will change from True to False in a future version. In addition, single cha
racter regular expressions will *not* be treated as literal strings when regex=True.
  df['Revenue_$Million'] = df['Revenue_$Million'].str.replace('.','')
```

In [71]:
```
df
```

Out[71]:

	Rank	Name	Revenue_$Million	Revenue_%Change	Profits_$Million	Profit_%Change	Assets_$Million	MarketValue_$Million	Employees
0	1	Walmart	572754	2.40%	$13,673	1.20%	244860	$409,795	2,300,000
1	2	Amazon	469822	21.70%	$33,364	56.40%	420549	$1,658,807.30	1,608,000
2	3	Apple	365817	33.30%	$94,680	64.90%	351002	$2,849,537.60	154,000
3	4	CVS Health	292111	8.70%	$7,910	10.20%	232999	$132,839.20	258,000
4	5	UnitedHealth Group	287597	11.80%	$17,285	12.20%	212206	$479,830.30	350,000
...	...	...	...	...	...	...	...	...	...
995	996	Vizio Holding	2124	4%	($39.40)	-138.40%	93580	$1,705.10	800
996	997	1-800-Flowers.com	212220	42.50%	$118.70	101.10%	107670	$830	4,800
997	998	Cowen	211280	30.20%	$295.60	36.60%	874880	$744.10	1,534
998	999	Ashland Global Holdings	2111	-11.20%	$220	-	6612	$5,601.90	4,100
999	1,000	DocuSign	210720	45%	($70)	-	254130	$21,302.80	7,461

1000 rows × 9 columns

In [72]:
```
df['Profits_$Million'] = df['Profits_$Million'].str.replace('$','')
df['Profits_$Million'] = df['Profits_$Million'].str.replace(',','')
df['Profits_$Million'] = df['Profits_$Million'].str.replace('.','')
df['MarketValue_$Million'] = df['MarketValue_$Million'].str.replace('$','')
df['MarketValue_$Million'] = df['MarketValue_$Million'].str.replace(',','')
df['MarketValue_$Million'] = df['MarketValue_$Million'].str.replace('.','')
df['Employees'] = df['Employees'].str.replace(',','')
```

```
C:\Users\biswa\AppData\Local\Temp\ipykernel_7548\1267740177.py:1: FutureWarning: The default value of regex will change from True to False in a future version. In addition, single cha
racter regular expressions will *not* be treated as literal strings when regex=True.
  df['Profits_$Million'] = df['Profits_$Million'].str.replace('$','')
C:\Users\biswa\AppData\Local\Temp\ipykernel_7548\1267740177.py:3: FutureWarning: The default value of regex will change from True to False in a future version. In addition, single cha
racter regular expressions will *not* be treated as literal strings when regex=True.
  df['Profits_$Million'] = df['Profits_$Million'].str.replace('.','')
C:\Users\biswa\AppData\Local\Temp\ipykernel_7548\1267740177.py:4: FutureWarning: The default value of regex will change from True to False in a future version. In addition, single cha
racter regular expressions will *not* be treated as literal strings when regex=True.
  df['MarketValue_$Million'] = df['MarketValue_$Million'].str.replace('$','')
C:\Users\biswa\AppData\Local\Temp\ipykernel_7548\1267740177.py:6: FutureWarning: The default value of regex will change from True to False in a future version. In addition, single cha
racter regular expressions will *not* be treated as literal strings when regex=True.
  df['MarketValue_$Million'] = df['MarketValue_$Million'].str.replace('.','')
```

In [73]:
```
df
```

Out[73]:

	Rank	Name	Revenue_$Million	Revenue_%Change	Profits_$Million	Profit_%Change	Assets_$Million	MarketValue_$Million	Employees
0	1	Walmart	572754	2.40%	13673	1.20%	244860	409795	2300000
1	2	Amazon	469822	21.70%	33364	56.40%	420549	165880730	1608000
2	3	Apple	365817	33.30%	94680	64.90%	351002	284953760	154000
3	4	CVS Health	292111	8.70%	7910	10.20%	232999	13283920	258000
4	5	UnitedHealth Group	287597	11.80%	17285	12.20%	212206	47983030	350000
...	...	...	...	...	...	...	...	...	...
995	996	Vizio Holding	2124	4%	(3940)	-138.40%	93580	170510	800
996	997	1-800-Flowers.com	212220	42.50%	11870	101.10%	107670	830	4800
997	998	Cowen	211280	30.20%	29560	36.60%	874880	74410	1534
998	999	Ashland Global Holdings	2111	-11.20%	220	-	6612	560190	4100
999	1,000	DocuSign	210720	45%	(70)	-	254130	2130280	7461

1000 rows × 9 columns

In [74]:
```
df['Rank'] = df['Rank'].str.replace(',','')
```

In [86]:
```
df['Profits_$Million'] = df['Profits_$Million'].str.replace('(','')
df['Profits_$Million'] = df['Profits_$Million'].str.replace(')','')
df['Revenue_%Change'] = df['Revenue_%Change'].str.replace('%','')
df['Profit_%Change'] = df['Profit_%Change'].str.replace('%','')
```

```
C:\Users\biswa\AppData\Local\Temp\ipykernel_7548\738909111.py:1: FutureWarning: The default value of regex will change from True to False in a future version. In addition, single char
acter regular expressions will *not* be treated as literal strings when regex=True.
  df['Profits_$Million'] = df['Profits_$Million'].str.replace('(','')
C:\Users\biswa\AppData\Local\Temp\ipykernel_7548\738909111.py:2: FutureWarning: The default value of regex will change from True to False in a future version. In addition, single char
acter regular expressions will *not* be treated as literal strings when regex=True.
  df['Profits_$Million'] = df['Profits_$Million'].str.replace(')','')
```

In [89]:
```
df.head(999)
```

Out[89]:

	Rank	Name	Revenue_$Million	Revenue_%Change	Profits_$Million	Profit_%Change	Assets_$Million	MarketValue_$Million	Employees
0	1	Walmart	572754	2.40	13673	1.20	244860	409795	2300000
1	2	Amazon	469822	21.70	33364	56.40	420549	165880730	1608000
2	3	Apple	365817	33.30	94680	64.90	351002	284953760	154000
3	4	CVS Health	292111	8.70	7910	10.20	232999	13283920	258000
4	5	UnitedHealth Group	287597	11.80	17285	12.20	212206	47983030	350000
5	6	Exxon Mobil	285640	57.40	23040	-	338923	34965240	63000
6	7	Berkshire Hathaway	276094	12.50	89795	111.20	958784	77954230	372000
7	8	Alphabet	257637	41.20	76033	88.80	359268	184232610	156500
8	9	McKesson	238228	3.10	4539	-604.30	65015	4585780	67500
9	10	AmerisourceBergen	21398880	12.70	153990	-	5733780	3235570	40000
10	11	Costco Wholesale	195929	17.50	5007	25.10	59268	25523070	288000

In [124]:
```
# No we dropped all the null values

df = df.dropna(axis=1)
```

In [126]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 727 entries, 0 to 997
Data columns (total 9 columns):
 #   Column                Non-Null Count  Dtype 
---  ------                --------------  ----- 
 0   Rank                  727 non-null    object
 1   Name                  727 non-null    object
 2   Revenue_$Million      727 non-null    object
 3   Revenue_%Change       727 non-null    object
 4   Profits_$Million      727 non-null    object
 5   Profit_%Change        727 non-null    object
 6   Assets_$Million       727 non-null    int32 
 7   MarketValue_$Million  727 non-null    object
 8   Employees             727 non-null    object
dtypes: int32(1), object(8)
memory usage: 70.1+ KB
```

In [133]:
```
df
```

38	39	FedEx	83959	21.30	5231	306.80	82777	5997130	484000
39	40	Humana	83064	7.70	2933	-12.90	44358	5515490	95500
40	41	Wells Fargo	82407	2.60	21548	552.80	1948068	184225	247848
41	42	State Farm Insurance	8222470	4.20	128090	-65.70	32534930	-	53586
42	43	Pfizer	81288	94	21979	128.60	181476	29238530	79000
43	44	Citigroup	79865	-10.10	21952	98.70	2291413	10533010	221768
44	45	PepsiCo	79474	12.90	7618	7	92377	23152820	309000
45	46	Intel	79024	1.50	19868	-4.90	168406	20263580	121100
46	47	Procter & Gamble	76118	7.30	14306	9.80	119307	36627160	101000
47	48	General Electric	74196	-6.80	6520	-214.30	198874	10081020	168000
48	49	IBM	72344	-1.70	5743	2.70	132001	11692830	297800
49	50	MetLife	71080	4.80	6554	21.20	759708	5798650	43000
51	52	Albertsons	6969040	11.60	85020	82.30	26598	1606370	300000

In [139]:
```
#changed the data type
df['Profits_$Million'] = df['Profits_$Million'].astype('int')
df['Assets_$Million'] = df['Assets_$Million'].astype('int')
df['Rank'] = df['Rank'].astype('int')
df['Employees'] = df['Employees'].astype('int')
```

In [162]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 691 entries, 0 to 997
Data columns (total 9 columns):
 #   Column               Non-Null Count  Dtype 
---  ------               --------------  ----- 
 0   Rank                 691 non-null    int32 
 1   Name                 691 non-null    object
 2   Revenue_$Million     691 non-null    object
 3   Revenue_%Change      691 non-null    object
 4   Profits_$Million     691 non-null    int32 
 5   Profit_%Change       691 non-null    object
 6   Assets_$Million      691 non-null    int32 
 7   MarketValue_$Million 691 non-null    object
 8   Employees            691 non-null    int32 
dtypes: int32(4), object(5)
memory usage: 59.4+ KB
```

In [ ]:

In [148]:
```
df.dropna(axis=0)
```

Out[148]:

	Rank	Name	Revenue_$Million	Revenue_%Change	Profits_$Million	Profit_%Change	Assets_$Million	MarketValue_$Million	Employees
0	1	Walmart	572754	2.40	13673	1.20	244860	409795	2300000
1	2	Amazon	469822	21.70	33364	56.40	420549	165880730	1608000
2	3	Apple	365817	33.30	94680	64.90	351002	284953760	154000
3	4	CVS Health	292111	8.70	7910	10.20	232999	13283920	258000
4	5	UnitedHealth Group	287597	11.80	17285	12.20	212206	47983030	350000
6	7	Berkshire Hathaway	276094	12.50	89795	111.20	958784	77954230	372000
7	8	Alphabet	257637	41.20	76033	88.80	359268	184232610	156500
8	9	McKesson	238228	3.10	4539	-604.30	65015	4585780	67500
10	11	Costco Wholesale	195929	17.50	5007	25.10	59268	25523070	288000
11	12	Cigna	174078	8.50	5365	-36.60	154889	7628630	72963
13	14	Microsoft	168088	17.50	61271	38.40	333779	231135890	181000

In [149]:
```
#we used this loop to drop all the rows with null Market Values

for x in df.index:
    if df.loc[x,'MarketValue_$Million'] == '-':
        df.drop(x,inplace = True)
df
```

25	26	Centene	125982	13.40	1347	-25.50	78375	4907220	72500
26	27	Meta Platforms	117929	37.20	39370	35.10	165987	605251	71970
27	28	Comcast	116385	12.40	14159	34.40	275905	21224580	189000
30	31	Dell Technologies	106995	13.60	5563	71.20	92735	3816440	133000
31	32	Target	106005	13.30	6946	59	53811	9813440	450000
32	33	Fannie Mae	101543	-4.60	22176	87.90	4229166	90910	7400
33	34	UPS	97287	15	12890	859.80	69405	18681660	400945
34	35	Lowe's	96250	7.40	8442	44.70	44640	13376110	270000
35	36	Bank of America	93851	0.10	31978	78.70	3169495	33243330	208248
36	37	Johnson & Johnson	93775	13.60	20878	41.90	182018	46604670	141700
37	38	Archer Daniels Midland	85249	32.50	2709	52.90	56136	5076920	39979
38	39	FedEx	83959	21.30	5231	306.80	82777	5997130	484000

In [171]:
```
#dropped % revenue change and % profit change as we don't need them

#df = df.drop(['Revenue_%Change', 'Profit_%Change'],axis =1)
```

In [169]:
```
df
```

Out[169]:

	Rank	Name	Revenue_$Million	Profits_$Million	Assets_$Million	MarketValue_$Million	Employees
0	1	Walmart	572754	13673	244860	409795	2300000
1	2	Amazon	469822	33364	420549	165880730	1608000
2	3	Apple	365817	94680	351002	284953760	154000
3	4	CVS Health	292111	7910	232999	13283920	258000
4	5	UnitedHealth Group	287597	17285	212206	47983030	350000
6	7	Berkshire Hathaway	276094	89795	958784	77954230	372000
7	8	Alphabet	257637	76033	359268	184232610	156500
8	9	McKesson	238228	4539	65015	4585780	67500
10	11	Costco Wholesale	195929	5007	59268	25523070	288000
11	12	Cigna	174078	5365	154889	7628630	72963
13	14	Microsoft	168088	61271	333779	231135890	181000

In [172]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 691 entries, 0 to 997
Data columns (total 7 columns):
 #   Column                Non-Null Count  Dtype 
---  ------                --------------  ----- 
 0   Rank                  691 non-null    int32 
 1   Name                  691 non-null    object
 2   Revenue_$Million      691 non-null    object
 3   Profits_$Million      691 non-null    int32 
 4   Assets_$Million       691 non-null    int32 
 5   MarketValue_$Million  691 non-null    object
 6   Employees             691 non-null    int32 
dtypes: int32(4), object(3)
memory usage: 48.6+ KB
```

In [175]:
```
#we changed the data type

df['Revenue_$Million'] = df['Revenue_$Million'].astype('int')
df['MarketValue_$Million'] = df['MarketValue_$Million'].astype('int')
```

In [176]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 691 entries, 0 to 997
Data columns (total 7 columns):
 #   Column              Non-Null Count  Dtype 
---  ------              --------------  ----- 
 0   Rank                691 non-null    int32 
 1   Name                691 non-null    object
 2   Revenue_$Million    691 non-null    int32 
 3   Profits_$Million    691 non-null    int32 
 4   Assets_$Million     691 non-null    int32 
 5   MarketValue_$Million 691 non-null    int32 
 6   Employees           691 non-null    int32 
dtypes: int32(6), object(1)
memory usage: 43.2+ KB
```

In [178]:
```
#The mean, count, std, min, max and % values of each coulmns

df.describe().T
```

Out[178]:

	count	mean	std	min	25%	50%	75%	max
Rank	691.0	4.832735e+02	2.854399e+02	1.0	236.5	482.0	725.0	998.0
Revenue_$Million	691.0	4.302773e+05	6.601375e+05	2124.0	17088.5	276094.0	516570.0	6969040.0
Profits_$Million	691.0	5.190720e+04	9.188531e+04	49.0	2158.0	17890.0	60965.0	807530.0
Assets_$Million	691.0	9.065362e+05	2.259551e+06	1066.0	47817.0	295240.0	835910.0	30465720.0
MarketValue_$Million	691.0	5.074647e+06	1.850857e+07	830.0	383020.0	1280450.0	3688365.0	284953760.0
Employees	691.0	4.081510e+04	1.228366e+05	347.0	7206.5	14100.0	31000.0	2300000.0

In [181]:
```
#we did this to reset the index

df = df.reset_index(drop = True)
```

In [182]:
```
df
```

10	14	Microsoft	168088	61271	333779	231135890	181000
11	17	Home Depot	151157	16433	71876	30931260	490600
12	18	Walgreens Boots Alliance	148579	2542	81285	3867110	258500
13	20	Elevance Health	138639	6104	97460	11853350	98200
14	21	Kroger	137888	1655	49086	4149620	420000
15	23	Verizon Communications	133613	22065	366596	21383710	118400
16	24	JPMorgan Chase	127202	48334	3743567	40252690	271025
17	25	General Motors	127004	10019	244718	6355520	157000
18	26	Centene	125982	1347	78375	4907220	72500
19	27	Meta Platforms	117929	39370	165987	605251	71970
20	28	Comcast	116385	14159	275905	21224580	189000
21	31	Dell Technologies	106995	5563	92735	3816440	133000
22	32	Target	106005	6946	53811	9813440	450000

In [190]:
```
#Now Let's see the top 10 companies by revenue

hrevenue = df.nlargest(10,'Revenue_$Million')
hrevenue
```

Out[190]:

	Rank	Name	Revenue_$Million	Profits_$Million	Assets_$Million	MarketValue_$Million	Employees
40	52	Albertsons	6969040	85020	26598	1606370	300000
53	70	Sysco	5129780	52420	2141350	4143310	57710
64	87	StoneX Group	4253420	11630	1883960	149440	3242
65	89	Enterprise Products Partners	4080690	463770	6752560	5617240	6911
70	98	Nucor	3648390	682750	2582310	39954	28800
77	106	Dollar General	3422040	239920	2632740	50953	163000
80	109	TD Synnex	3161420	39510	2766640	985360	27000
83	115	Netflix	2969780	511620	4458470	16630410	11300
87	120	Starbucks	2906060	419930	3139260	10464280	383000
89	122	Eli Lilly	2831840	558170	48806	27272360	35238

In [208]:
```
#Now let's see the bottom 10 companies by revenue

lrevenue = df.nsmallest(10, 'Revenue_$Million')
lrevenue
```

Out[208]:

	Rank	Name	Revenue_$Million	Profits_$Million	Assets_$Million	MarketValue_$Million	Employees
688	996	Vizio Holding	2124	3940	93580	170510	800
680	981	Aerojet Rocketdyne Holdings	2188	14370	243360	316940	5000
660	949	Stepan	2346	13780	206560	221560	2439
656	942	Portland General Electric	2396	244	9494	4940	2839
654	940	Allison Transmission Holdings	2402	442	4457	381060	3400
650	935	Gray Television	2413	90	11108	209320	8801
645	928	Equity Residential	2464	133290	2116920	3380250	2400
640	917	NortonLifeLock	2552	554	6361	1544190	2800
639	915	ManTech International	2554	137	263960	351760	9800
638	912	Playtika Holding	2583	30850	280330	796580	4000

In [ ]:
```
# Top 10 compamnies by Profit
```

In [188]:
```
hprofit= df.nlargest(10, 'Profits_$Million')
hprofit
```

Out[188]:

	Rank	Name	Revenue_$Million	Profits_$Million	Assets_$Million	MarketValue_$Million	Employees
170	231	Regeneron Pharmaceuticals	1607170	807530	2543480	75806	10368
112	152	McDonald's	2322290	754520	5385430	18387360	200000
70	98	Nucor	3648390	682750	2582310	39954	28800
119	159	Blackstone	2257710	585740	4119640	8890720	3795
89	122	Eli Lilly	2831840	558170	48806	27272360	35238
83	115	Netflix	2969780	511620	4458470	16630410	11300
101	138	KKR	2614130	466650	26428540	5154290	3238
65	89	Enterprise Products Partners	4080690	463770	6752560	5617240	6911
96	131	Lennar	2713070	443010	3320780	23413	10753
603	850	Bio-Rad Laboratories	292250	424590	1777580	1698220	7900

In [210]:
```
lprofit= df.nsmallest(10, 'Profits_$Million')
lprofit
```

Out[210]:

	Rank	Name	Revenue_$Million	Profits_$Million	Assets_$Million	MarketValue_$Million	Employees
507	711	Ventas	3828	49	2471780	2467620	434
644	922	MYR Group	249830	85	112110	159450	7600
650	935	Gray Television	2413	90	11108	209320	8801
528	739	PriceSmart	361990	98	170580	243410	10400
217	295	Andersons	1272880	104	456920	170820	2334
342	476	Ingredion	6894	117	6999	581420	12000
685	992	Beazer Homes USA	214030	122	207880	47880	1052
199	275	Wayfair	13708	131	4570	1164160	16681
639	915	ManTech International	2554	137	263960	351760	9800
670	963	TTEC Holdings	227310	141	199680	3878	65000

In [212]:
```
# just to check the total number of null values

df.isnull().sum()
```

Out[212]:
```
Rank                    0
Name                    0
Revenue_$Million        0
Profits_$Million        0
Assets_$Million         0
MarketValue_$Million    0
Employees               0
dtype: int64
```

In [213]:
```
df
```

21	31	Dell Technologies	106995	5563	92735	3816440	133000
22	32	Target	106005	6946	53811	9813440	450000
23	33	Fannie Mae	101543	22176	4229166	90910	7400
24	34	UPS	97287	12890	69405	18681660	400945
25	35	Lowe's	96250	8442	44640	13376110	270000
26	36	Bank of America	93851	31978	3169495	33243330	208248
27	37	Johnson & Johnson	93775	20878	182018	46604670	141700
28	38	Archer Daniels Midland	85249	2709	56136	5076920	39979
29	39	FedEx	83959	5231	82777	5997130	484000
30	40	Humana	83064	2933	44358	5515490	95500
31	41	Wells Fargo	82407	21548	1948068	184225	247848
32	43	Pfizer	81288	21979	181476	29238530	79000
33	44	Citigroup	79865	21952	2291413	10533010	221768

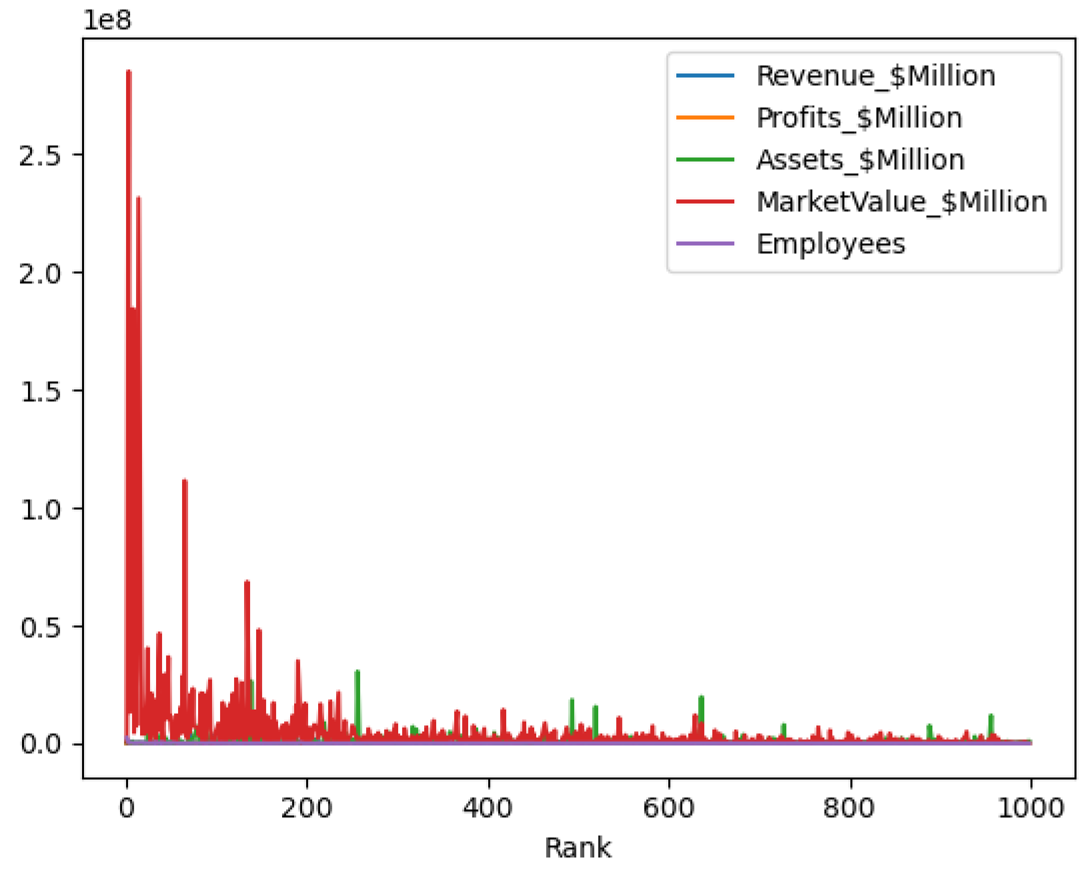
In [224]:
```
#df = df.set_index('Rank')
```

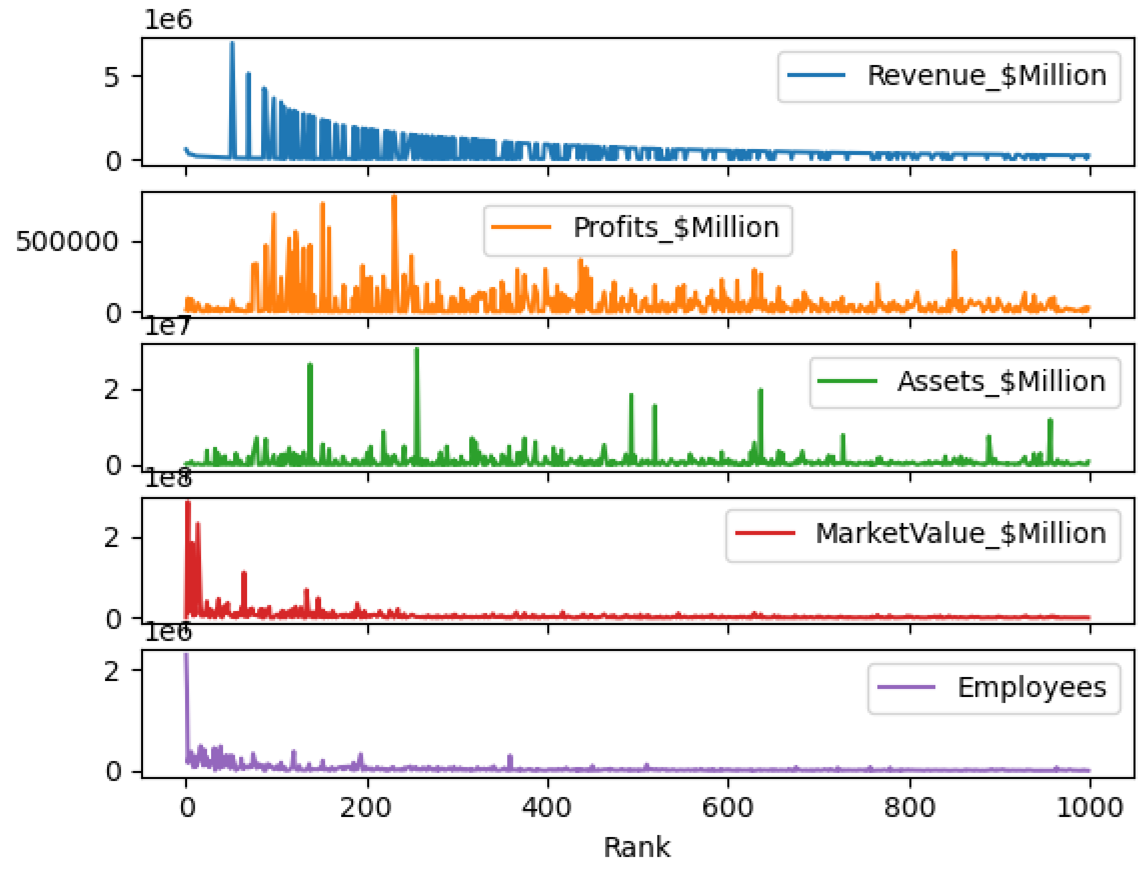
In [223]:
```
df
```

```
 7          Berkshire Hathaway   276094   89795    958784    77954230   372000
 8                    Alphabet   257637   76033    359268   184232610   156500
 9                    McKesson   238228    4539     65015     4585780    67500
11           Costco Wholesale   195929    5007     59268    25523070   288000
12                       Cigna   174078    5365    154889     7628630    72963
14                   Microsoft   168088   61271    333779   231135890   181000
17                  Home Depot   151157   16433     71876    30931260   490600
18    Walgreens Boots Alliance   148579    2542     81285     3867110   258500
20             Elevance Health   138639    6104     97460    11853350    98200
21                      Kroger   137888    1655     49086     4149620   420000
23     Verizon Communications   133613   22065    366596    21383710   118400
24              JPMorgan Chase   127202   48334   3743567    40252690   271025
25             General Motors   127004   10019    244718     6355520   157000
```

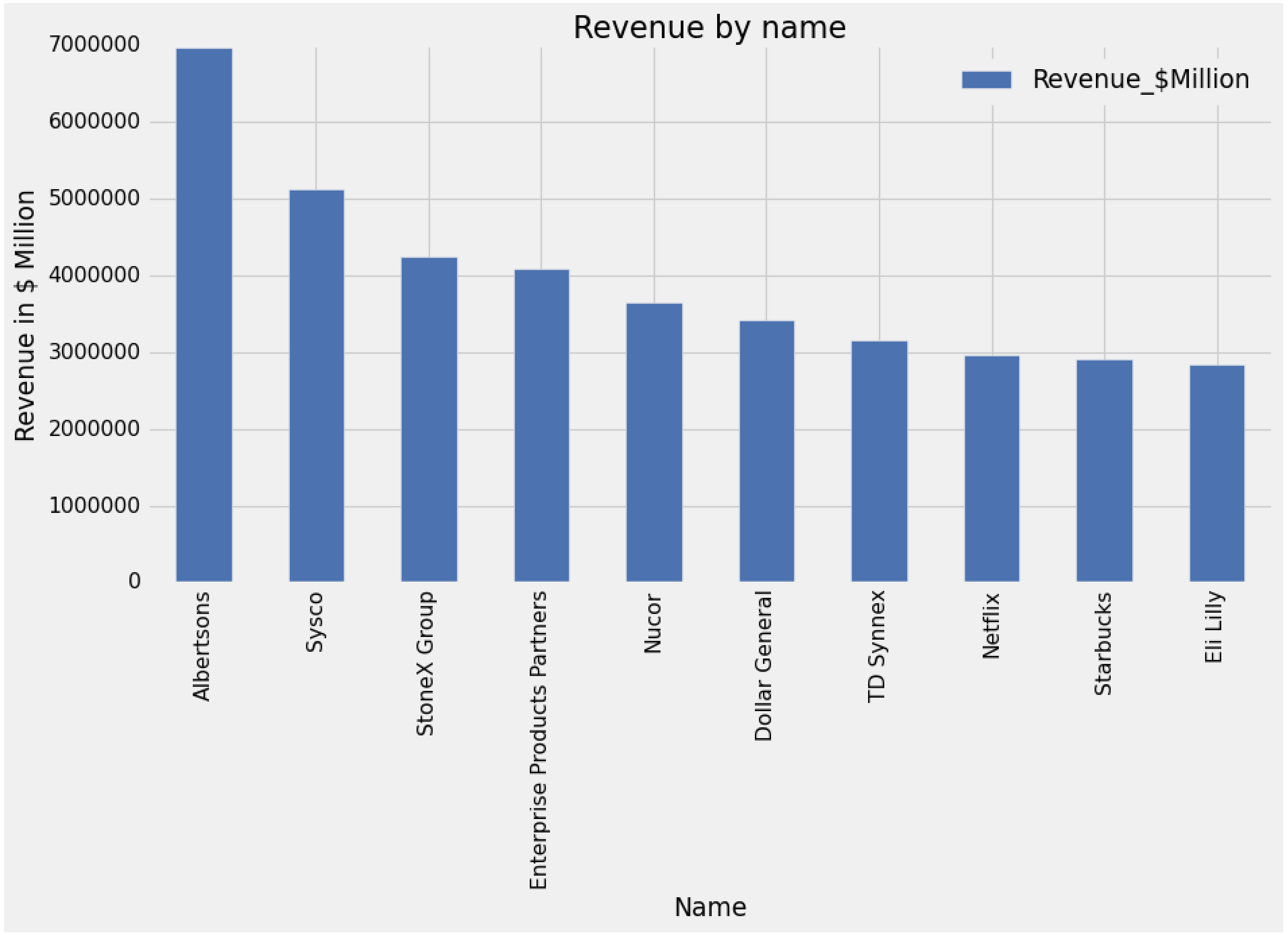
```
In [226]: #just a normal subplot

          df.plot()
          df.plot(kind = 'line', subplots = True)
```

```
Out[226]: array([<Axes: xlabel='Rank'>, <Axes: xlabel='Rank'>,
                 <Axes: xlabel='Rank'>, <Axes: xlabel='Rank'>,
                 <Axes: xlabel='Rank'>], dtype=object)
```




1e6
5
0
Revenue_$Million
500000
0
Profits_$Million
1e7
2
0
Assets_$Million
1e8
2
0
MarketValue_$Million
1e6
2
0
Employees
0
200
400
600
800
1000
Rank

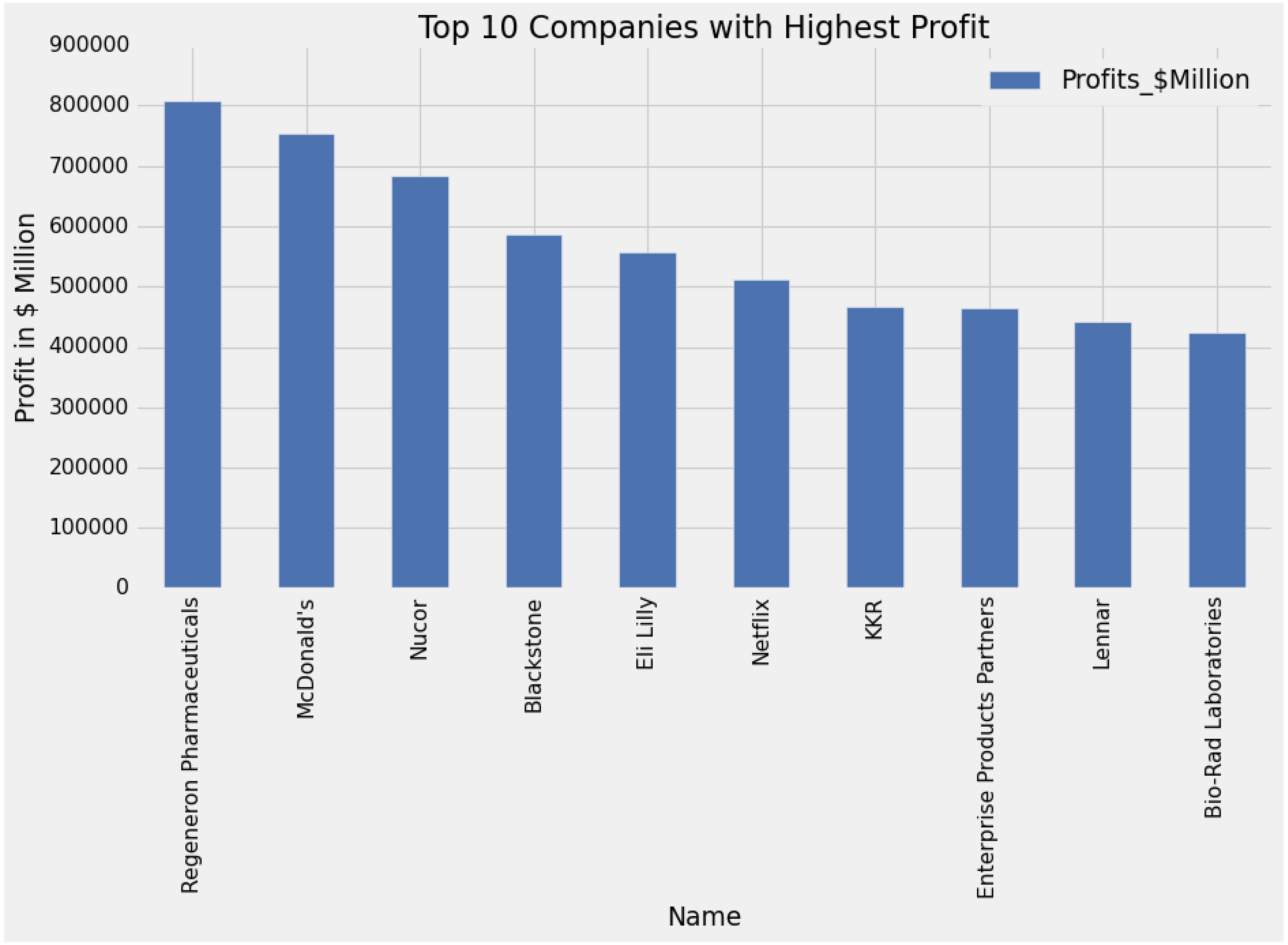
```
In [279]: #PLOT FOR TOP 10 REVENUE BY NAME

          hrevenue.plot.bar(y = 'Revenue_$Million', x= 'Name',figsize =(12,6),ylabel = 'Revenue in $ Million',
                            title = 'Revenue by name')

Out[279]: <Axes: title={'center': 'Revenue by name'}, xlabel='Name', ylabel='Revenue in $ Million'>
```



```
In [246]: #to see the available themes

          print(plt.style.available)

['Solarize_Light2', '_classic_test_patch', '_mpl-gallery', '_mpl-gallery-nogrid', 'bmh', 'classic', 'dark_background', 'fast', 'fivethirtyeight', 'ggplot', 'grayscale', 'seaborn-v0_
8', 'seaborn-v0_8-bright', 'seaborn-v0_8-colorblind', 'seaborn-v0_8-dark', 'seaborn-v0_8-dark-palette', 'seaborn-v0_8-darkgrid', 'seaborn-v0_8-deep', 'seaborn-v0_8-muted', 'seaborn-v0
_8-notebook', 'seaborn-v0_8-paper', 'seaborn-v0_8-pastel', 'seaborn-v0_8-poster', 'seaborn-v0_8-talk', 'seaborn-v0_8-ticks', 'seaborn-v0_8-white', 'seaborn-v0_8-whitegrid', 'tableau-c
olorblind10']
```

```
In [275]: #Theme that I used

          plt.style.use('seaborn-v0_8-deep')
```

```
In [276]: #PLOT FOR TOP 10 PROFIT BY NAME

          hprofit.sort_values(by = 'Profits_$Million', ascending = False).plot.bar(y = 'Profits_$Million', x= 'Name', figsize =(12,6), ylabel = 'Profit in $ Million',
                         title = 'Top 10 Companies with Highest Profit')

Out[276]: <Axes: title={'center': 'Top 10 Companies with Highest Profit'}, xlabel='Name', ylabel='Profit in $ Million'>
```



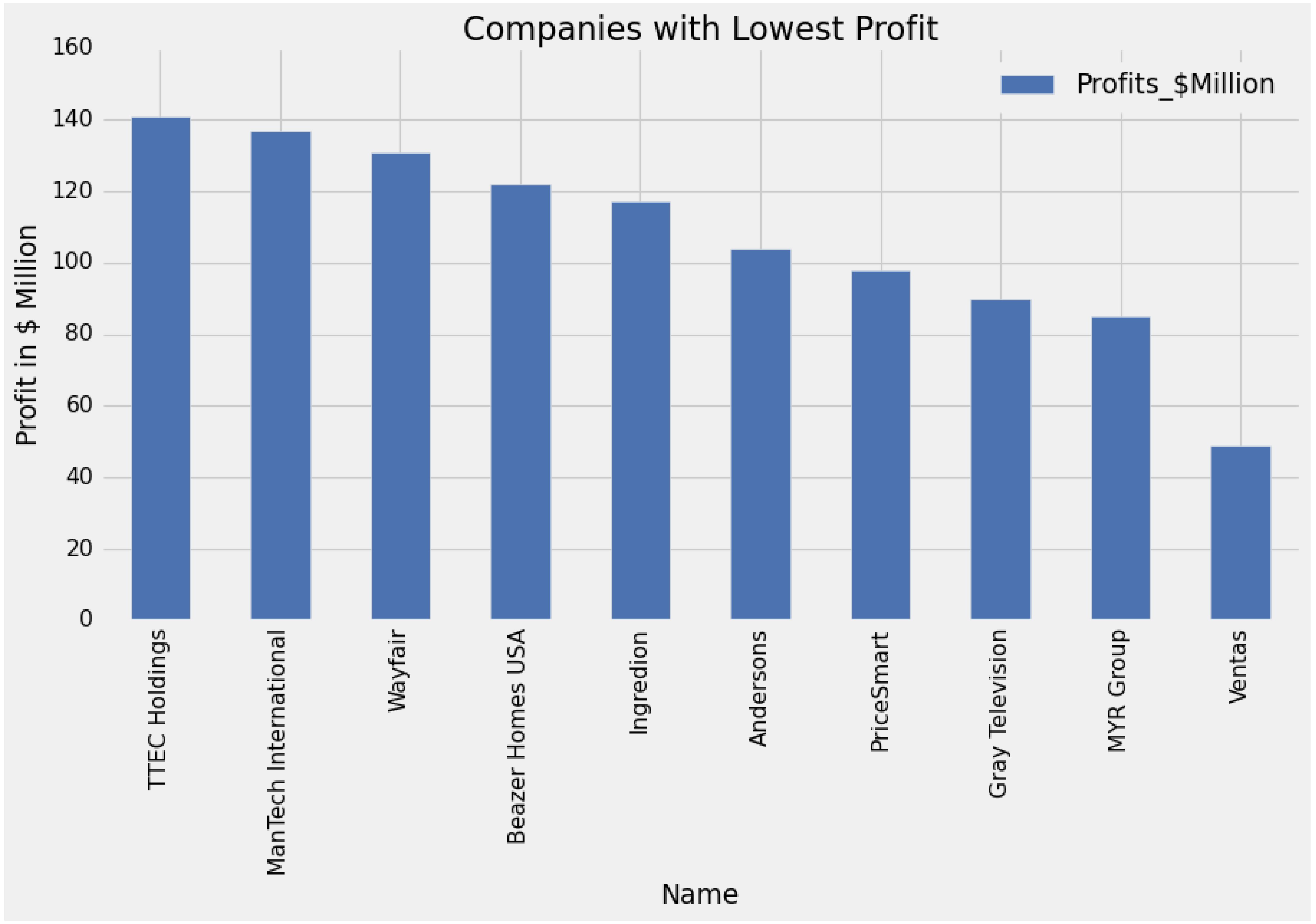
```
In [277]: df
```

Out[277]:

	Name	Revenue_$Million	Profits_$Million	Assets_$Million	MarketValue_$Million	Employees
Rank						
1	Walmart	572754	13673	244860	409795	2300000
2	Amazon	469822	33364	420549	165880730	1608000
3	Apple	365817	94680	351002	284953760	154000
4	CVS Health	292111	7910	232999	13283920	258000
5	UnitedHealth Group	287597	17285	212206	47983030	350000
7	Berkshire Hathaway	276094	89795	958784	77954230	372000
8	Alphabet	257637	76033	359268	184232610	156500
9	McKesson	238228	4539	65015	4585780	67500
11	Costco Wholesale	195929	5007	59268	25523070	288000
12	Cigna	174078	5365	154889	7628630	72963

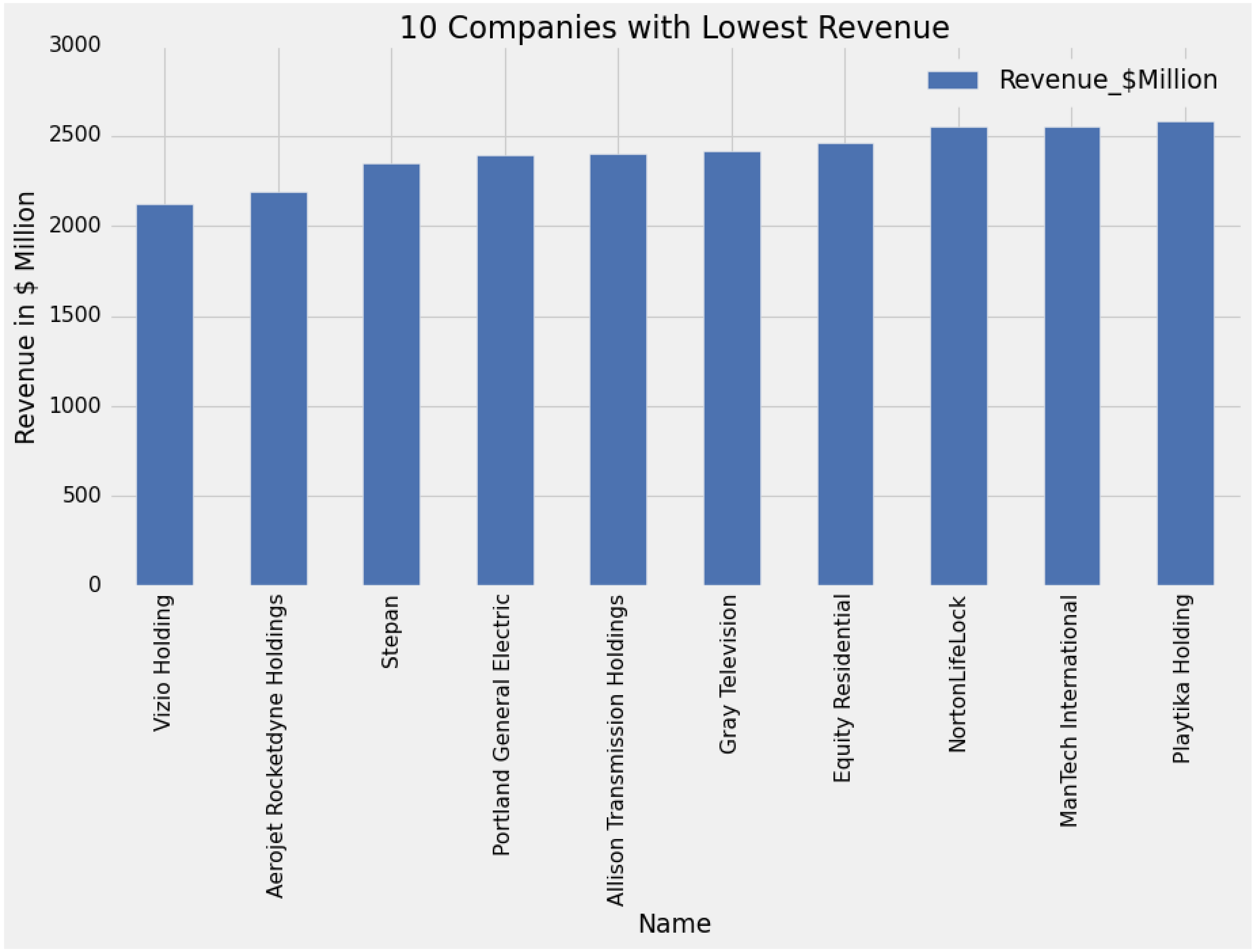
```
In [278]: #PLOT FOR LOWEST 10 PROFIT BY NAME

          lprofit.sort_values(by = 'Profits_$Million', ascending = False).plot.bar(y = 'Profits_$Million', x= 'Name', figsize =(12,6), ylabel = 'Profit in $ Million',
                              title = 'Companies with Lowest Profit')
```

```
Out[278]: <Axes: title={'center': 'Companies with Lowest Profit'}, xlabel='Name', ylabel='Profit in $ Million'>
```



In [281]:
```
#PLOT FOR BOTTOM 10 REVENUES OF COMPANIES

lrevenue.plot.bar(y = 'Revenue_$Million', x= 'Name',figsize =(12,6),ylabel = 'Revenue in $ Million',
                   title = '10 Companies with Lowest Revenue')
```

Out[281]:
```
<Axes: title={'center': '10 Companies with Lowest Revenue'}, xlabel='Name', ylabel='Revenue in $ Million'>
```



In [282]:
```python
df
```

38	Archer Daniels Midland	85249	2709	56136	5076920	39979
39	FedEx	83959	5231	82777	5997130	484000
40	Humana	83064	2933	44358	5515490	95500
41	Wells Fargo	82407	21548	1948068	184225	247848
43	Pfizer	81288	21979	181476	29238530	79000
44	Citigroup	79865	21952	2291413	10533010	221768
45	PepsiCo	79474	7618	92377	23152820	309000
46	Intel	79024	19868	168406	20263580	121100
47	Procter & Gamble	76118	14306	119307	36627160	101000
48	General Electric	74196	6520	198874	10081020	168000
49	IBM	72344	5743	132001	11692830	297800
50	MetLife	71080	6554	759708	5798650	43000
52	Albertsons	6969040	85020	26598	1606370	300000

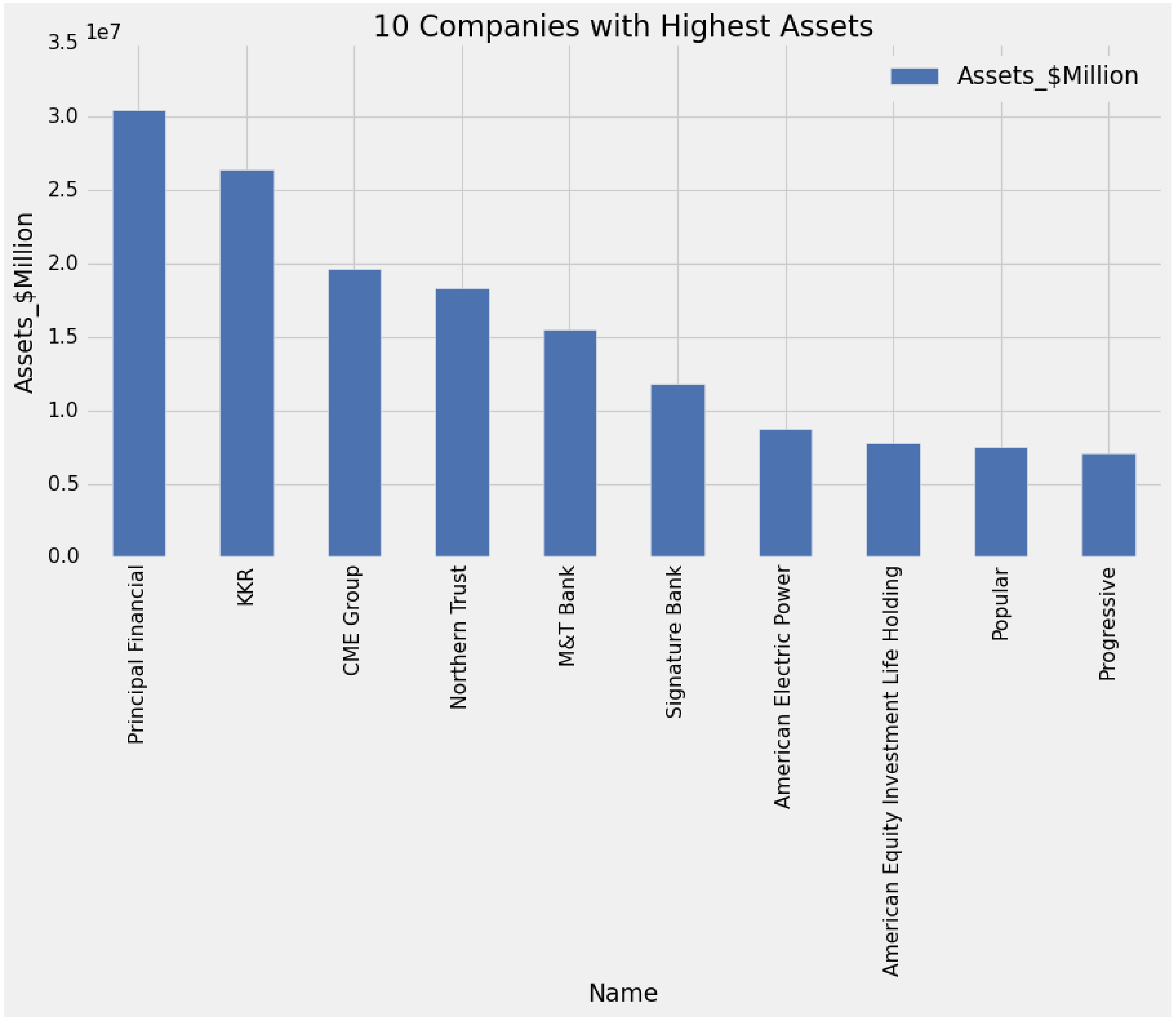
In [288]:
```python
# Top 10 Companies with highest Assets

hassets = df.nlargest(10, 'Assets_$Million')
hassets
```

Out[288]:

	Name	Revenue_$Million	Profits_$Million	Assets_$Million	MarketValue_$Million	Employees
Rank						
256	Principal Financial	1426270	171060	30465720	1917670	18600
138	KKR	2614130	466650	26428540	5154290	3238
636	CME Group	468970	263640	19678030	8549230	3480
493	Northern Trust	648830	154530	18388980	2421510	21100
519	M&T Bank	610580	185870	15510720	21875	17342
956	Signature Bank	231130	91840	11844540	1841690	1854
219	American Electric Power	16792	248810	8766870	5033860	16688
727	American Equity Investment Life Holding	368950	474	7834910	386920	850
888	Popular	276480	93490	7509790	625230	8351
79	Progressive	47702	335090	7113230	6667030	49077

```
In [289]: #TOP 10 COMPANIES WITH HIGEST ASSETS

          hassets.plot.bar(y = 'Assets_$Million', x= 'Name',figsize =(12,6),ylabel = 'Assets_$Million',
                        title = '10 Companies with Highest Assets')
```

```
Out[289]: <Axes: title={'center': '10 Companies with Highest Assets'}, xlabel='Name', ylabel='Assets_$Million'>
```



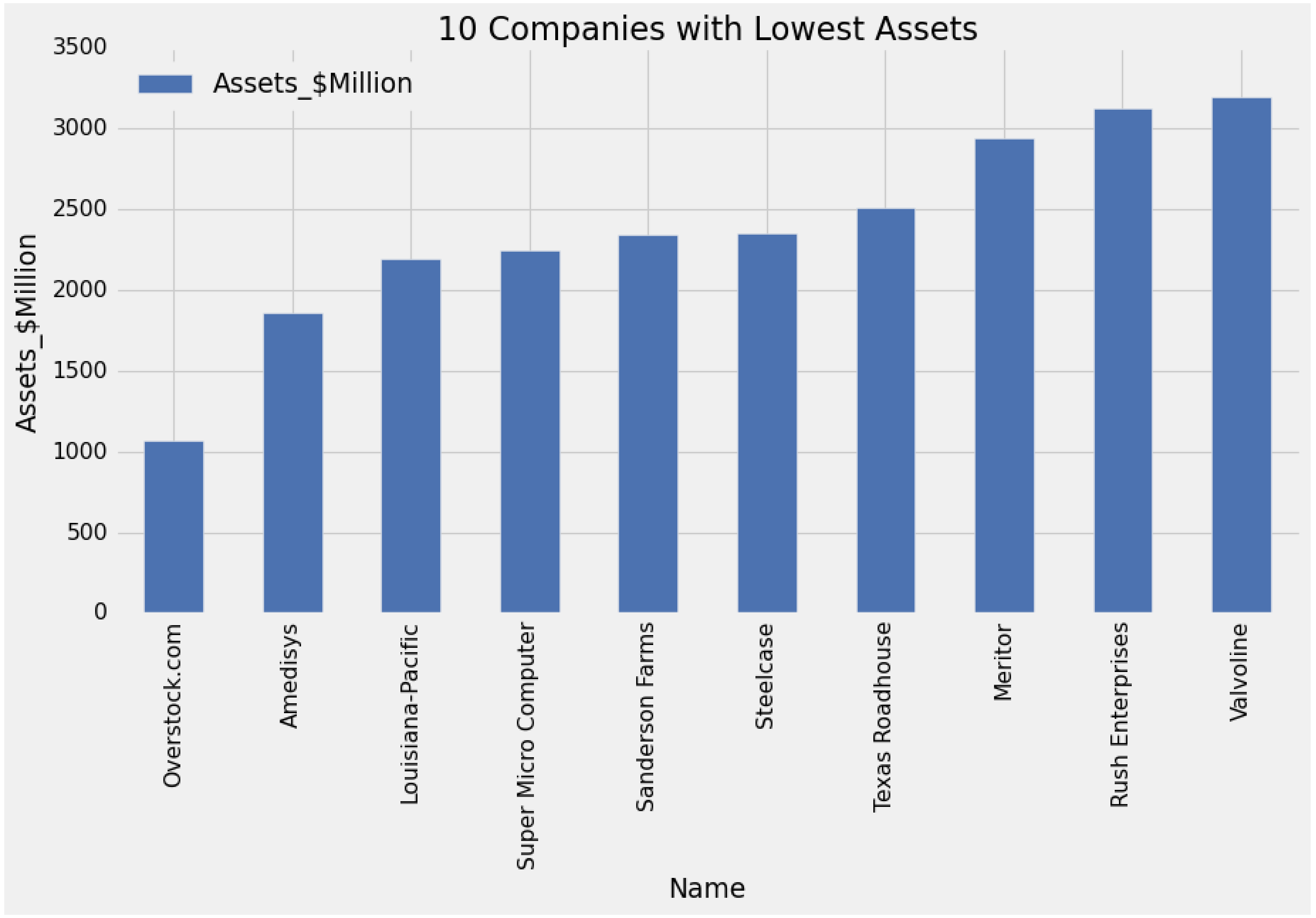
In [286]:
```
# Top 10 Companies with Lowest Assets

lassets = df.nsmallest(10, 'Assets_$Million')
lassets
```

Out[286]:

Rank	Name	Revenue_$Million	Profits_$Million	Assets_$Million	MarketValue_$Million	Employees
883	Overstock.com	277380	38940	1066	189780	1350
978	Amedisys	221410	20910	1857	560390	21000
645	Louisiana-Pacific	4553	1377	2194	534410	4800
743	Super Micro Computer	355740	11190	2242	1963	4155
623	Sanderson Farms	479970	45510	2345	418540	17662
909	Steelcase	259620	2610	2354	133950	11400
757	Texas Roadhouse	346390	24530	2512	578780	73300
710	Meritor	3833	199	2938	251940	9600
590	Rush Enterprises	512610	24140	3120	278850	7141
837	Valvoline	2981	420	3191	566050	9800

```
In [290]: #Plot for 10 COMPANIES WITH Lowest ASSETS

          lassets.plot.bar(y = 'Assets_$Million', x= 'Name',figsize =(12,6),ylabel = 'Assets_$Million',
                               title = '10 Companies with Lowest Assets')
```

```
Out[290]: <Axes: title={'center': '10 Companies with Lowest Assets'}, xlabel='Name', ylabel='Assets_$Million'>
```



In [297]:
```
df.describe().T
```

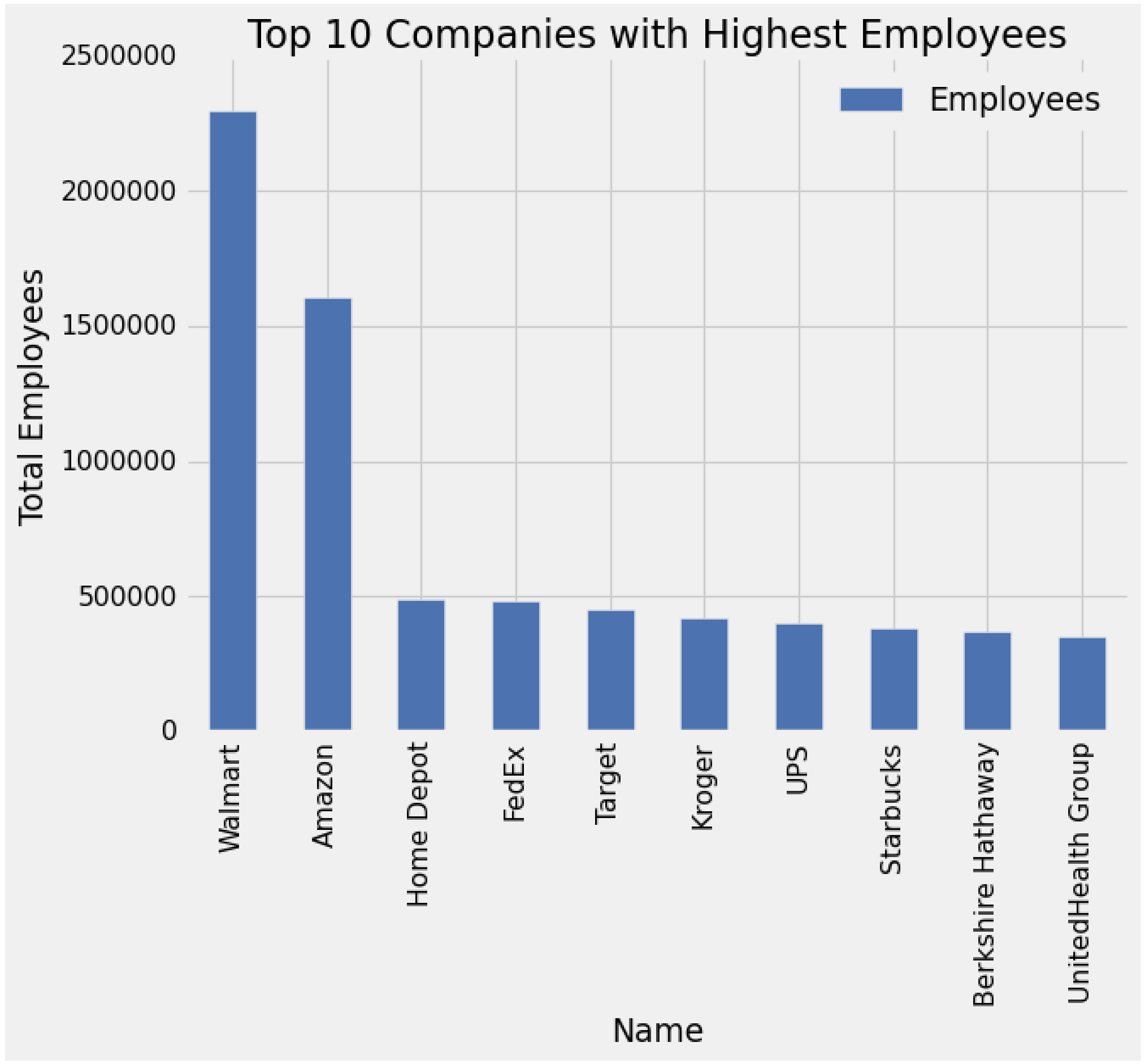
Out[297]:
```
                   count          mean           std     min       25%        50%        75%          max
Revenue_$Million   691.0  4.302773e+05  6.601375e+05  2124.0   17088.5   276094.0   516570.0    6969040.0
Profits_$Million   691.0  5.190720e+04  9.188531e+04    49.0    2158.0    17890.0    60965.0     807530.0
Assets_$Million    691.0  9.065362e+05  2.259551e+06  1066.0   47817.0   295240.0   835910.0   30465720.0
MarketValue_$Million 691.0 5.074647e+06 1.850857e+07  830.0  383020.0  1280450.0  3688365.0  284953760.0
Employees          691.0  4.081510e+04  1.228366e+05   347.0    7206.5    14100.0    31000.0    2300000.0
```

In [301]:
```
# TOP 5 Companies with highest number of employees

lemployees = df.nlargest(10, 'Employees')
lemployees
```

Out[301]:
```
                   Name  Revenue_$Million  Profits_$Million  Assets_$Million  MarketValue_$Million  Employees
Rank
   1            Walmart            572754             13673           244860                409795    2300000
   2             Amazon            469822             33364           420549             165880730    1608000
  17         Home Depot            151157             16433            71876              30931260     490600
  39              FedEx             83959              5231            82777               5997130     484000
  32             Target            106005              6946            53811               9813440     450000
  21             Kroger            137888              1655            49086               4149620     420000
  34                UPS             97287             12890            69405              18681660     400945
 120          Starbucks           2906060            419930          3139260              10464280     383000
   7 Berkshire Hathaway            276094             89795           958784              77954230     372000
   5  UnitedHealth Group            287597             17285           212206              47983030     350000
```

```
In [310]: # Plot

          lemployees.plot.bar(y = 'Employees', x= 'Name', ylabel = 'Total Employees', title = 'Top 10 Companies with Highest Employees')
```

```
Out[310]: <Axes: title={'center': 'Top 10 Companies with Highest Employees'}, xlabel='Name', ylabel='Total Employees'>
```



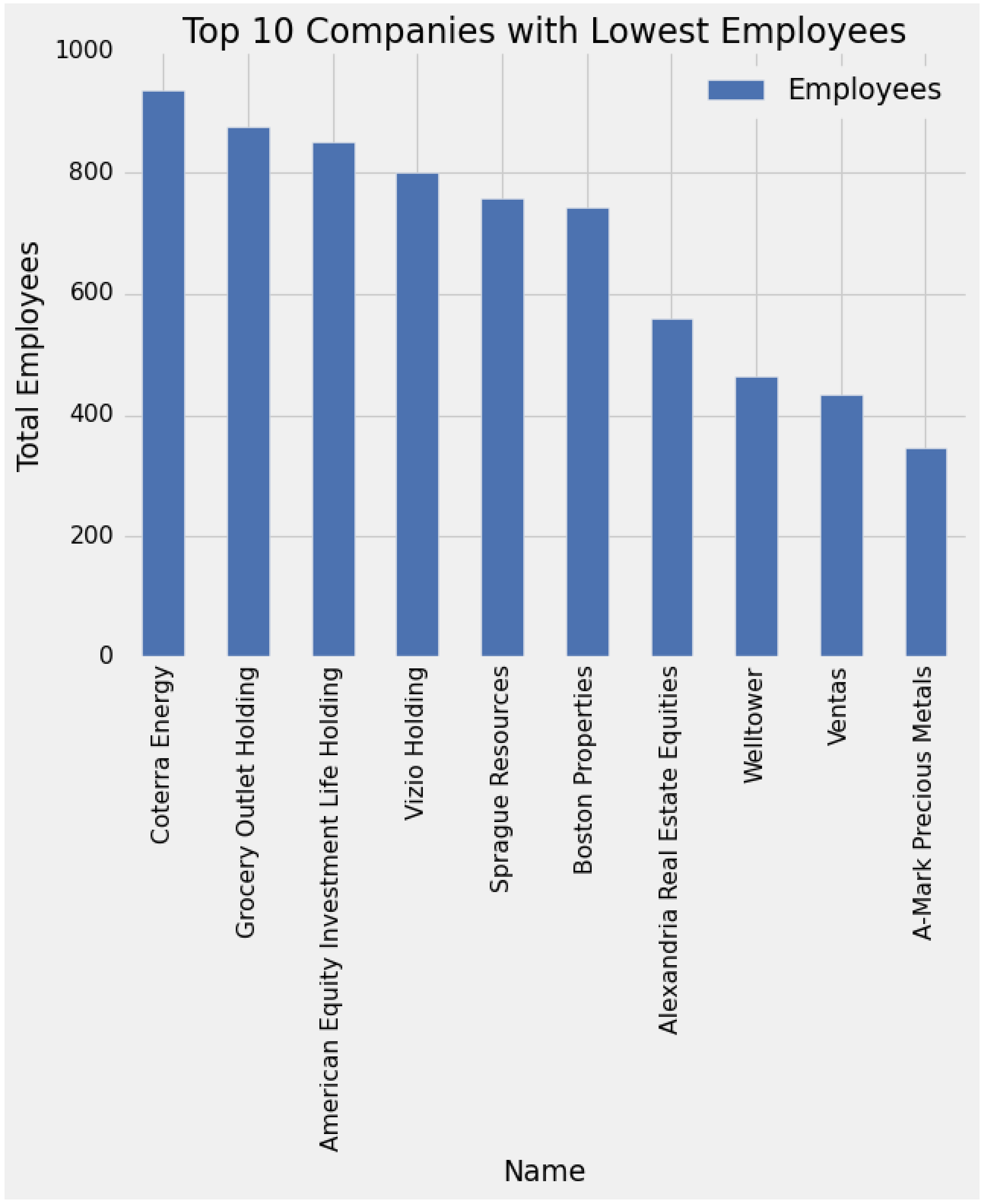
In [302]:
```
# 5 Companies with lowest employees

semployees = df.nsmallest(10, 'Employees')
semployees
```

Out[302]:

Rank	Name	Revenue_$Million	Profits_$Million	Assets_$Million	MarketValue_$Million	Employees
446	A-Mark Precious Metals	7613	15960	119160	89050	347
711	Ventas	3828	49	2471780	2467620	434
630	Welltower	474210	33610	3491030	4300150	464
945	Alexandria Real Estate Equities	237360	57120	3021940	3218860	559
857	Boston Properties	288860	50520	2236530	2017990	743
751	Sprague Resources	349820	6890	141830	43370	757
996	Vizio Holding	2124	3940	93580	170510	800
727	American Equity Investment Life Holding	368950	474	7834910	386920	850
816	Grocery Outlet Holding	307960	6230	266980	3148	875
762	Coterra Energy	3449	1158	19900	2187210	936

In [314]:
```
#Plot

semployees.sort_values(by = 'Employees', ascending = False).plot.bar(y = 'Employees', x= 'Name', ylabel = 'Total Employees', title = 'Top 10 Companies with Lowest Employees')
```

Out[314]:
```
<Axes: title={'center': 'Top 10 Companies with Lowest Employees'}, xlabel='Name', ylabel='Total Employees'>
```



```
In [315]: # Final Data Frame

          df
```

```
Out[315]:
                            Name  Revenue_$Million  Profits_$Million  Assets_$Million  MarketValue_$Million  Employees
          Rank
             1               Walmart          572754             13673           244860                409795    2300000
             2                Amazon          469822             33364           420549             165880730    1608000
             3                 Apple          365817             94680           351002             284953760     154000
             4            CVS Health          292111              7910           232999              13283920     258000
             5      UnitedHealth Group        287597             17285           212206              47983030     350000
             7      Berkshire Hathaway        276094             89795           958784              77954230     372000
             8              Alphabet          257637             76033           359268             184232610     156500
             9              McKesson          238228              4539            65015               4585780      67500
            11      Costco Wholesale          195929              5007            59268              25523070     288000
            12                 Cigna          174078              5365           154889               7628630      72963
```

```
In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:
```