

In [1]:

```
import pandas as pd
```

In [2]:

```
df = pd.read_csv('FT1000.csv')
```

In [3]:

```
df.head()
```

Out[3]:

	Rank	Name	Ranked2021	Ranked2020	Country	Sector	CAGR	Revenue2020	Revenue
0	1	Swappie	No	No	Finland	Technology	477.43	97611814	5
1	2	Kilo Health	No	No	Lithuania	Health	450.05	57318766	3
2	3	OCI	No	No	UK	Financial Services	409.59	568322073	43
3	4	OnlyFans	No	No	UK	Technology	393.63	316732986	26
4	5	Enpal	No	No	Germany	Energy	386.88	56109613	4

In [4]:

```
df.tail()
```

Out[4]:

	Rank	Name	Ranked2021	Ranked2020	Country	Sector	CAGR	Revenue2020
995	996	peopleForecast	No	No	Germany	Technology	36.59	2086411
996	997	Digitalpa	No	No	Italy	Management Consulting	36.59	1731340
997	998	Faktenkontor	No	No	Germany	Advertising	36.59	30967000
998	999	CLAREO	Yes	Yes	France	Retail	36.58	18854708
999	1000	Laca Trade	Yes	Yes	Italy	Property	36.55	24741510

In [5]:

```
df.shape
```

Out[5]:

(1000, 12)

In [6]:

```
df.columns
```

Out[6]:

```
Index(['Rank', 'Name', 'Ranked2021', 'Ranked2020', 'Country', 'Sector', 'CAGR',  
      'Revenue2020', 'Revenue2017', 'Employees2020', 'Employees2017',  
      'FoundingYear'],  
      dtype='object')
```

In [7]:

```
df.duplicated().sum()
```

Out[7]:

```
0
```

In [8]:

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

Out[8]:

```
Rank          0  
Name          0  
Ranked2021    0  
Ranked2020    0  
Country       0  
Sector        0  
CAGR          0  
Revenue2020   0  
Revenue2017   0  
Employees2020 0  
Employees2017 0  
FoundingYear   0  
dtype: int64
```

```
In [9]:
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1000 entries, 0 to 999
Data columns (total 12 columns):
#   Column                Non-Null Count  Dtype
---  -
0   Rank                   1000 non-null   int64
1   Name                   1000 non-null   object
2   Ranked2021             1000 non-null   object
3   Ranked2020             1000 non-null   object
4   Country                1000 non-null   object
5   Sector                1000 non-null   object
6   CAGR                   1000 non-null   float64
7   Revenue2020            1000 non-null   int64
8   Revenue2017            1000 non-null   int64
9   Employees2020          1000 non-null   int64
10  Employees2017          1000 non-null   int64
11  FoundingYear            1000 non-null   int64
dtypes: float64(1), int64(6), object(5)
memory usage: 93.9+ KB
```

```
In [10]:
df.describe()
```

Out[10]:

	Rank	CAGR	Revenue2020	Revenue2017	Employees2020	Employees2017	FoundingYear
count	1000.000000	1000.000000	1.000000e+03	1.000000e+03	1000.000000	1000.000000	1000.000000
mean	500.500000	78.881950	2.034009e+07	4.005901e+06	79.948000	26.480000	1999.500000
std	288.819436	52.471399	7.771805e+07	9.207763e+06	161.448155	63.526161	10.000000
min	1.000000	36.550000	1.507867e+06	1.007110e+05	1.000000	0.000000	1900.000000
25%	250.750000	46.582500	3.186724e+06	6.283870e+05	14.000000	4.000000	1950.000000
50%	500.500000	61.105000	6.361154e+06	1.271956e+06	30.000000	10.000000	1975.000000
75%	750.250000	93.522500	1.662410e+07	3.549436e+06	70.000000	23.000000	1990.000000
max	1000.000000	477.430000	2.120072e+09	1.453982e+08	1798.000000	767.000000	2000.000000

In [11]:

```
df.nunique()
```

Out[11]:

```
Rank      1000
Name      1000
Ranked2021      2
Ranked2020      2
Country     30
Sector      39
CAGR       931
Revenue2020   999
Revenue2017   999
Employees2020  225
Employees2017  122
FoundingYear   42
dtype: int64
```

In [12]:

```
df['Ranked2021'].unique()
```

Out[12]:

```
array(['No', 'Yes'], dtype=object)
```

In [13]:

```
df['Ranked2021'].value_counts()
```

Out[13]:

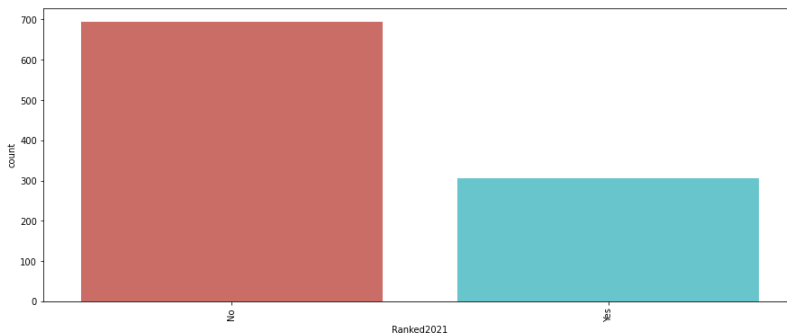
```
No      694
Yes     306
Name: Ranked2021, dtype: int64
```

In [14]:

```
import matplotlib.pyplot as plt
import seaborn as sns
import warnings
warnings.filterwarnings("ignore")
```

In [15]:

```
plt.figure(figsize=(15,6))
sns.countplot('Ranked2021', data = df, palette = 'hls')
plt.xticks(rotation = 90)
plt.show()
```



In [16]:

```
label_data = df['Ranked2021'].value_counts()

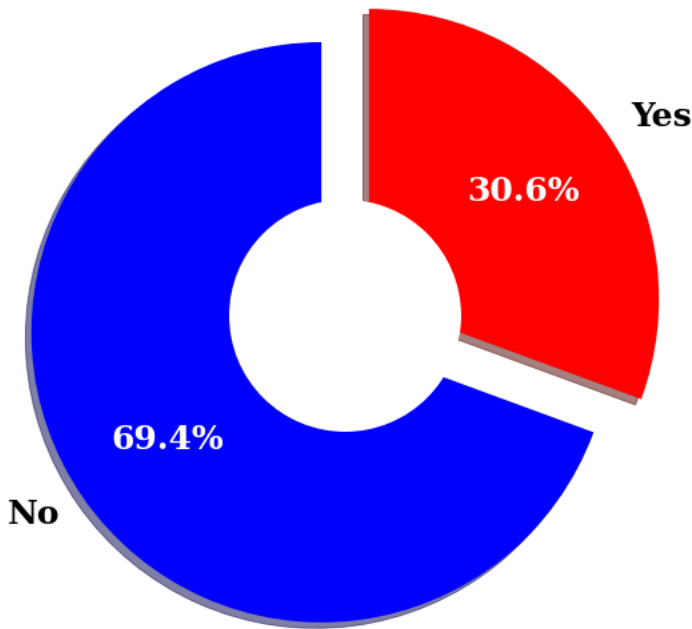
explode = (0.1, 0.1)
plt.figure(figsize=(14, 10))
patches, texts, pcts = plt.pie(label_data,
                                labels = label_data.index,
                                colors = ['blue', 'red'],
                                pctdistance = 0.65,
                                shadow = True,
                                startangle = 90,
                                explode = explode,
                                autopct = '%1.1f%%',
                                textprops={ 'fontsize': 25,
                                              'color': 'black',
                                              'weight': 'bold',
                                              'family': 'serif' })

plt.setp(pcts, color='white')

hfont = {'fontname':'serif', 'weight': 'bold'}
plt.title('Ranked2021', size=20, **hfont)

centre_circle = plt.Circle((0,0),0.40,fc='white')
fig = plt.gcf()
fig.gca().add_artist(centre_circle)
plt.show()
```

**Ranked2021**



In [17]:

```
df['Ranked2020'].unique()
```

Out[17]:

array(['No', 'Yes'], dtype=object)

In [18]:

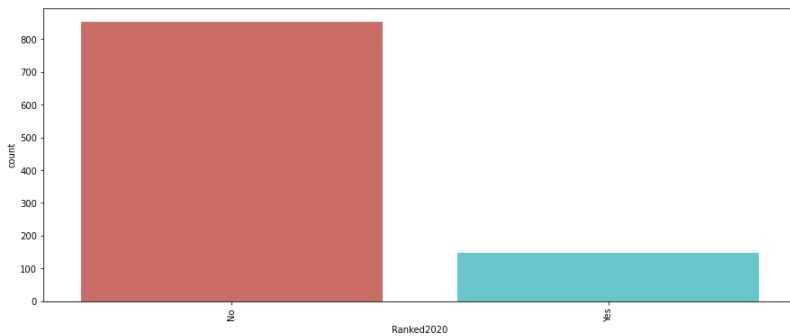
```
df['Ranked2020'].value_counts()
```

Out[18]:

```
No      852
Yes     148
Name: Ranked2020, dtype: int64
```

In [19]:

```
plt.figure(figsize=(15,6))
sns.countplot('Ranked2020', data = df, palette = 'hls')
plt.xticks(rotation = 90)
plt.show()
```





In [20]:

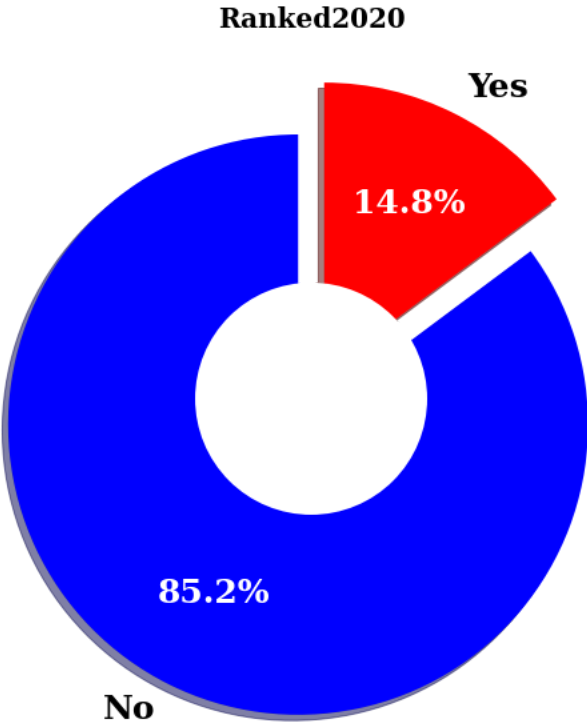
```
label_data = df['Ranked2020'].value_counts()

explode = (0.1, 0.1)
plt.figure(figsize=(14, 10))
patches, texts, pcts = plt.pie(label_data,
                                labels = label_data.index,
                                colors = ['blue', 'red'],
                                pctdistance = 0.65,
                                shadow = True,
                                startangle = 90,
                                explode = explode,
                                autopct = '%1.1f%%',
                                textprops={ 'fontsize': 25,
                                              'color': 'black',
                                              'weight': 'bold',
                                              'family': 'serif' })

plt.setp(pcts, color='white')

hfont = {'fontname': 'serif', 'weight': 'bold'}
plt.title('Ranked2020', size=20, **hfont)

centre_circle = plt.Circle((0,0),0.40,fc='white')
fig = plt.gcf()
fig.gca().add_artist(centre_circle)
plt.show()
```



In [21]:

```
df['Country'].unique()
```

Out[21]:

```
array(['Finland', 'Lithuania', 'UK', 'Germany', 'France', 'Italy',  
      'Sweden', 'Austria', 'Switzerland', 'Norway', 'Poland', 'Belgium',  
      'Latvia', 'The Netherlands', 'Luxembourg', 'Spain', 'Greece',  
      'Portugal', 'Denmark', 'Cyprus', 'Hungary', 'Ireland', 'Slovakia',  
      'Estonia', 'Czech Republic', 'Croatia', 'Romania', 'Bulgaria',  
      'Slovenia', 'Liechtenstein'], dtype=object)
```

In [22]:

```
df['Country'].value_counts()
```

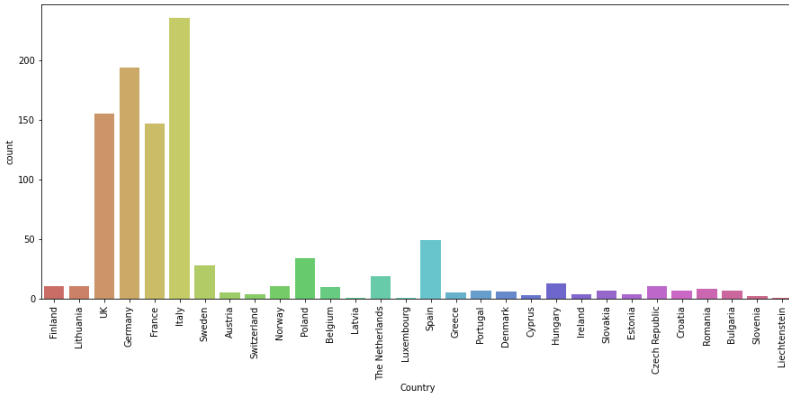
Out[22]:

Italy	235
Germany	194
UK	155
France	147
Spain	49
Poland	34
Sweden	28
The Netherlands	19
Hungary	13
Czech Republic	11
Lithuania	11
Finland	11
Norway	11
Belgium	10
Romania	8
Bulgaria	7
Portugal	7
Slovakia	7
Croatia	7
Denmark	6
Greece	5
Austria	5
Ireland	4
Switzerland	4
Estonia	4
Cyprus	3
Slovenia	2
Latvia	1
Luxembourg	1
Liechtenstein	1

Name: Country, dtype: int64

In [23]:

```
plt.figure(figsize=(15,6))
sns.countplot('Country', data = df, palette = 'hls')
plt.xticks(rotation = 90)
plt.show()
```



In [24]:

```
df['Sector'].unique()
```

Out[24]:

```
array(['Technology', 'Health', 'Financial Services', 'Energy',
      'Waste management & recycling', 'Industrial Goods', 'Ecommerce',
      'Food & Beverage', 'Support Services', 'Fintech', 'Transport',
      'Fashion', 'Retail', 'Games industry', 'Education', 'Property',
      'Management Consulting', 'Advertising', 'Interiors',
      'Pharmaceuticals', 'Personal & Household Goods', 'Automobiles',
      'Cyber Security', 'Construction', 'Batteries', 'Beauty',
      'Chemicals & Pharmaceuticals', 'Media', 'Aerospace & Defence',
      'Sales & Marketing', 'Telecoms', 'Agricultural Commodities',
      'Travel & Leisure', 'Restaurants', 'Insurance', 'Architecture',
      'Law', 'Precious metals', 'Sales and Marketing'], dtype=object)
```

```
In [25]:  
df['Sector'].value_counts()
```

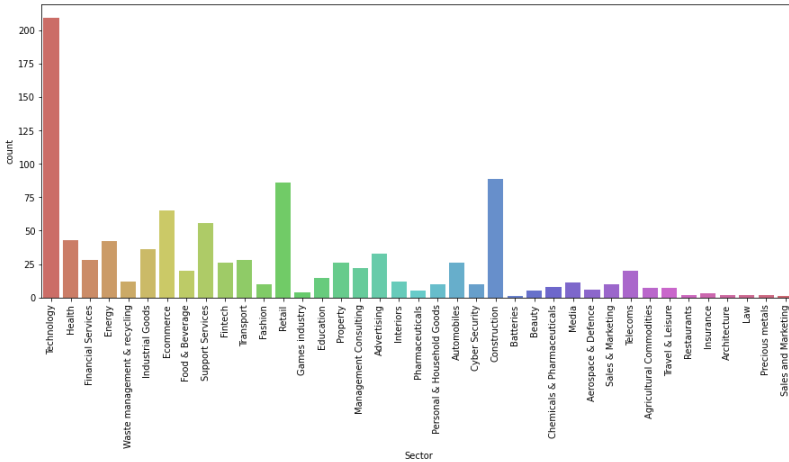
Out[25]:

Technology	209
Construction	89
Retail	86
Ecommerce	65
Support Services	56
Health	43
Energy	42
Industrial Goods	36
Advertising	33
Transport	28
Financial Services	28
Fintech	26
Automobiles	26
Property	26
Management Consulting	22
Telecoms	20
Food & Beverage	20
Education	15
Interiors	12
Waste management & recycling	12
Media	11
Sales & Marketing	10
Personal & Household Goods	10
Cyber Security	10
Fashion	10
Chemicals & Pharmaceuticals	8
Travel & Leisure	7
Agricultural Commodities	7
Aerospace & Defence	6
Pharmaceuticals	5
Beauty	5
Games industry	4
Insurance	3
Restaurants	2
Architecture	2
Law	2
Precious metals	2
Batteries	1
Sales and Marketing	1

Name: Sector, dtype: int64

In [26]:

```
plt.figure(figsize=(15,6))
sns.countplot('Sector', data = df, palette = 'hls')
plt.xticks(rotation = 90)
plt.show()
```



In [27]:

```
df['FoundingYear'].unique()
```

Out[27]:

```
array([2016, 2013, 2012, 2017, 2015, 2014, 2011, 2000, 2006, 2004, 2001,
       2010, 2008, 2009, 2005, 2007, 2003, 1989, 1996, 1976, 1999, 1990,
       1982, 2002, 1986, 1984, 1994, 1993, 1998, 1979, 1952, 1997, 1983,
       1977, 1965, 1975, 1956, 1995, 1991, 1938, 1992, 1898], dtype=int64)
```

In [28]:

```
df['FoundingYear'].value_counts()
```

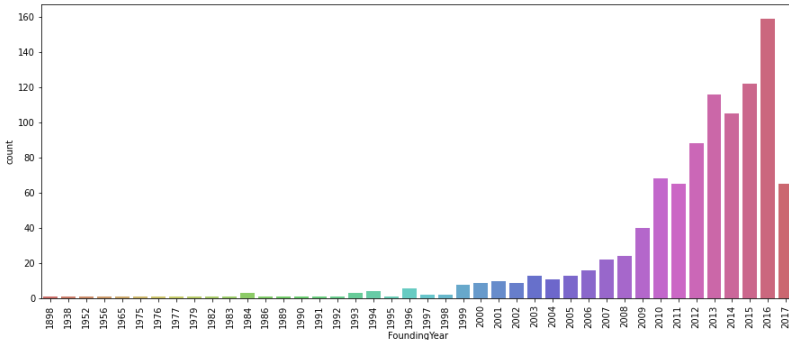
Out[28]:

2016	159
2015	122
2013	116
2014	105
2012	88
2010	68
2017	65
2011	65
2009	40
2008	24
2007	22
2006	16
2005	13
2003	13
2004	11
2001	10
2000	9
2002	9
1999	8
1996	6
1994	4
1984	3
1993	3
1998	2
1997	2
1938	1
1992	1
1991	1
1977	1
1995	1
1956	1
1975	1
1965	1
1990	1
1983	1
1952	1
1979	1
1986	1
1982	1
1976	1
1989	1
1898	1

Name: FoundingYear, dtype: int64

In [29]:

```
plt.figure(figsize=(15,6))
sns.countplot('FoundingYear', data = df, palette = 'hls')
plt.xticks(rotation = 90)
plt.show()
```

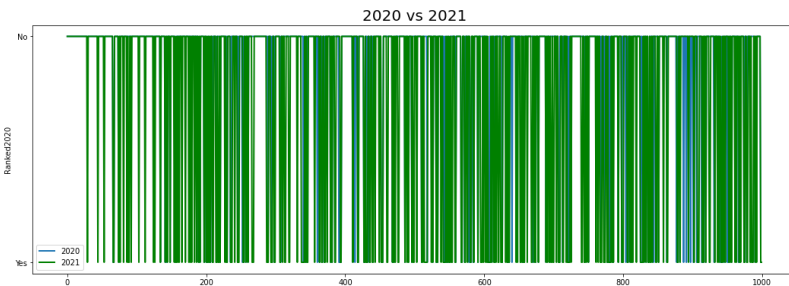


In [30]:

```
import plotly.express as px
import plotly.graph_objects as go
from plotly import tools
from plotly.subplots import make_subplots
from plotly.offline import iplot, init_notebook_mode
```

In [31]:

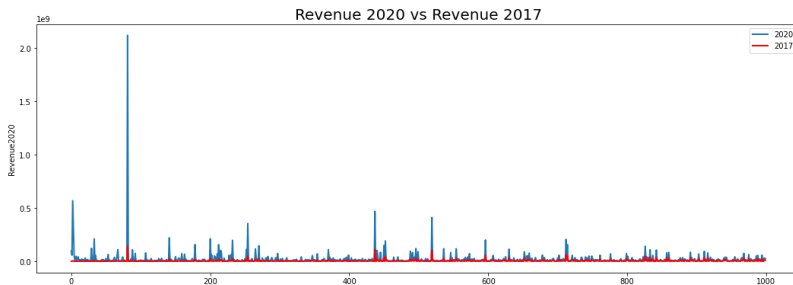
```
plt.figure(figsize=(18,6))
plt.title('2020 vs 2021',fontsize = 20)
sns.lineplot(data=df['Ranked2020'],linewidth = 2, label = '2020')
sns.lineplot(data=df['Ranked2021'],linewidth = 2, label = '2021', color='green')
plt.show()
```





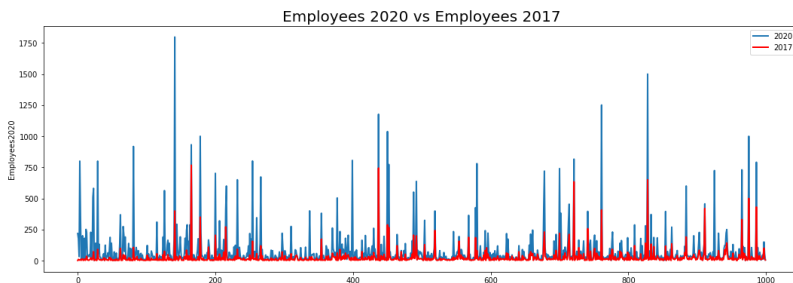
In [32]:

```
plt.figure(figsize=(18,6))
plt.title('Revenue 2020 vs Revenue 2017',fontsize = 20)
sns.lineplot(data=df['Revenue2020'],linewidth = 2, label = '2020')
sns.lineplot(data=df['Revenue2017'],linewidth = 2, label = '2017', color='r')
plt.show()
```



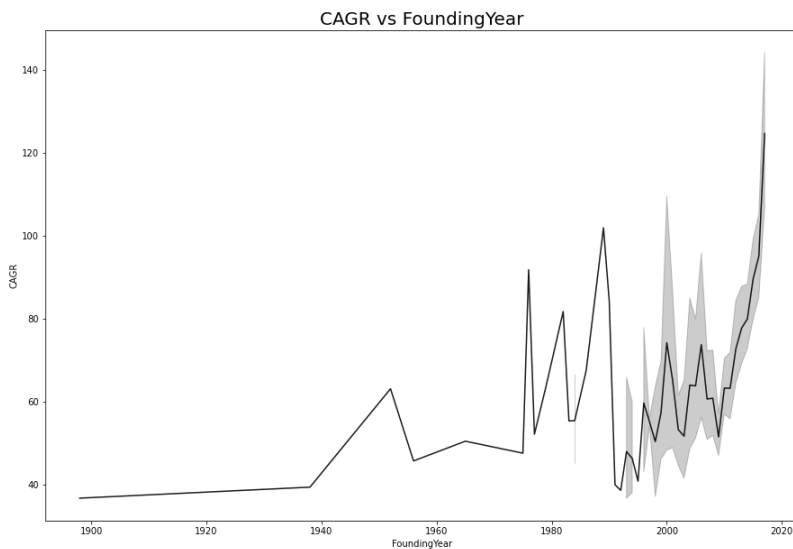
In [33]:

```
plt.figure(figsize=(18,6))
plt.title('Employees 2020 vs Employees 2017',fontsize = 20)
sns.lineplot(data=df['Employees2020'],linewidth = 2, label = '2020')
sns.lineplot(data=df['Employees2017'],linewidth = 2, label = '2017', color='r')
plt.show()
```



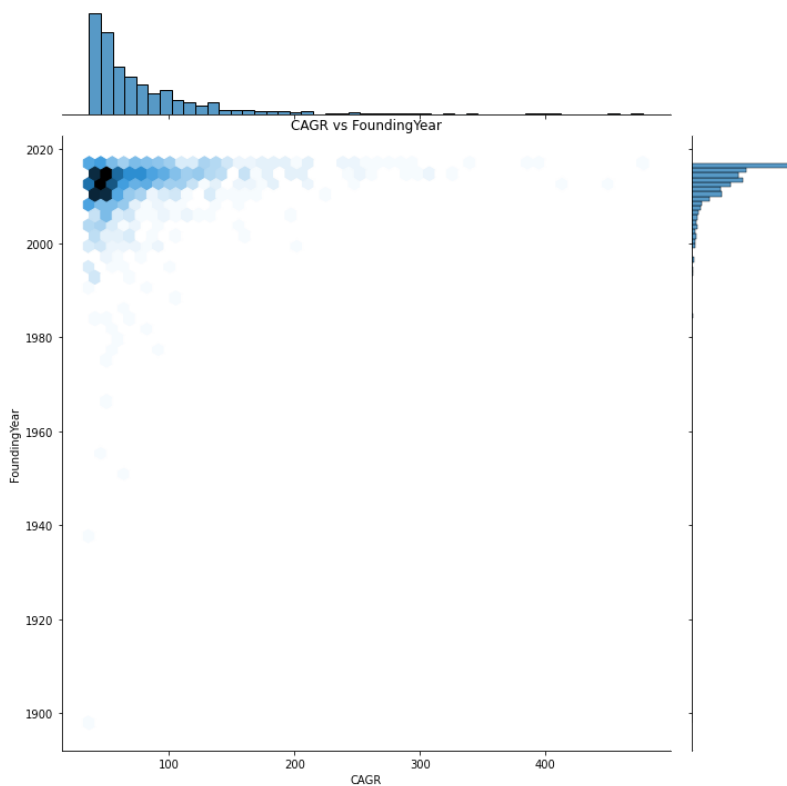
In [34]:

```
plt.figure(figsize=(15,10))
sns.lineplot(data=df, x='FoundingYear', y='CAGR', color='black', alpha=0.9)
plt.title('CAGR vs FoundingYear',fontsize = 20)
plt.show()
```



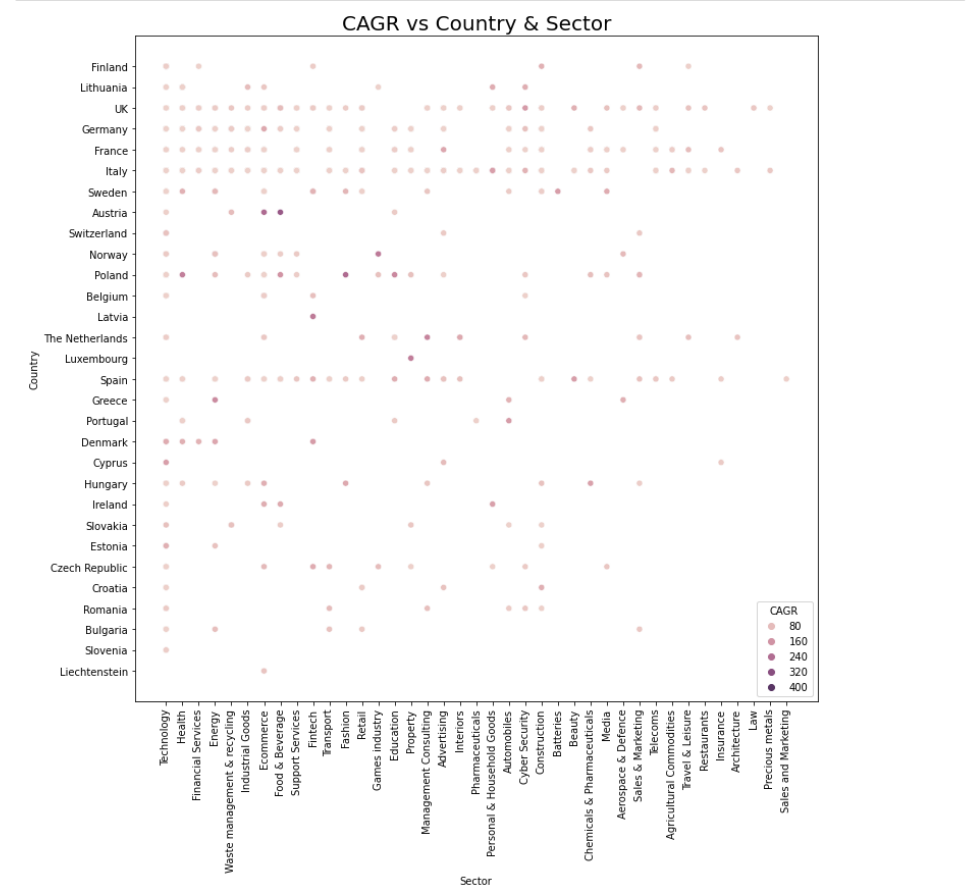
In [35]:

```
sns.jointplot(x='CAGR', y='FoundingYear', data=df , height = 10 , kind='hex')  
plt.title('CAGR vs FoundingYear',fontsize = 12)  
plt.show()
```



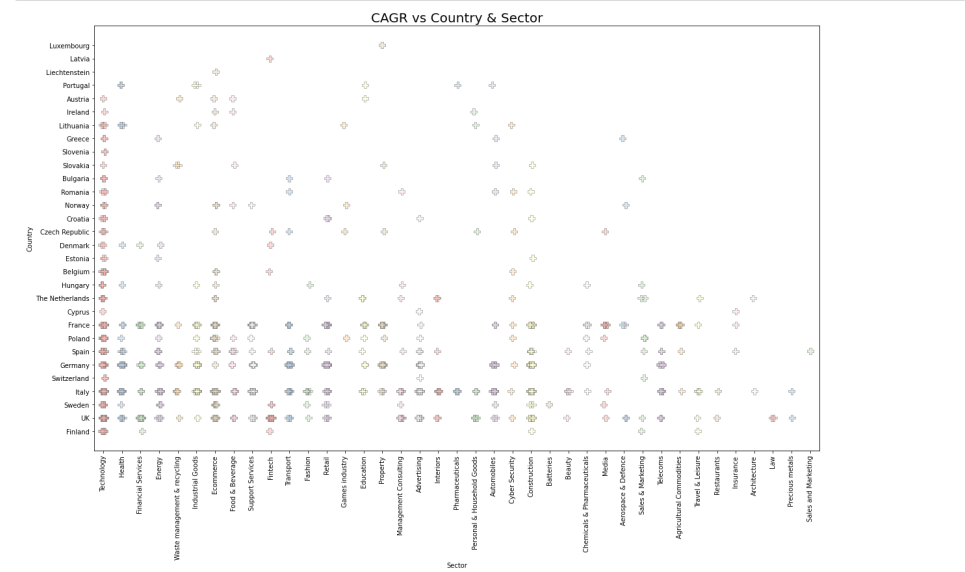
In [36]:

```
plt.figure(figsize=(12,12))
sns.scatterplot(x='Sector',y='Country',hue='CAGR',data=df)
plt.xticks(rotation=90)
plt.title('CAGR vs Country & Sector',fontsize = 20)
plt.show()
```



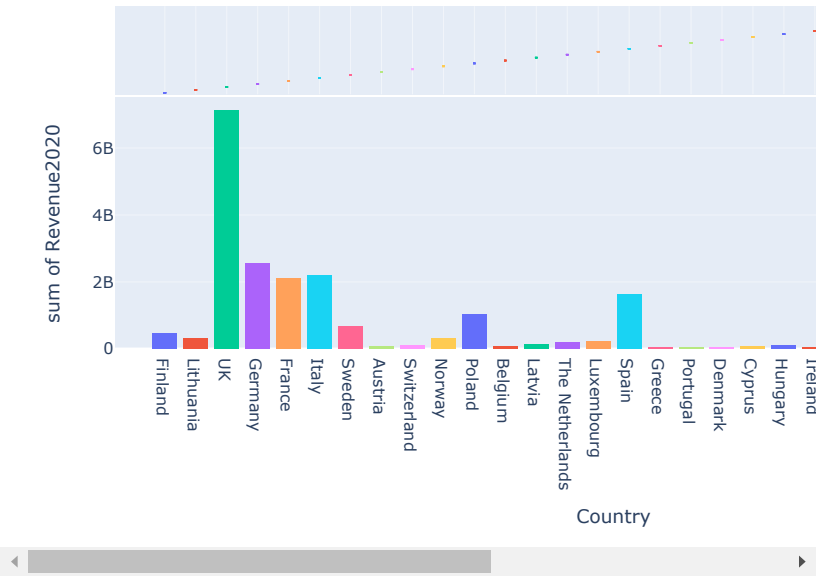
In [37]:

```
fig, ax = plt.subplots(figsize=(20,12))
plt.xticks(rotation=90)
ax = sns.stripplot(y='Country', x='Sector', data=df, palette='Pastel1', s=10, marker='P', li
plt.title('CAGR vs Country & Sector',fontsize = 20)
plt.show()
```



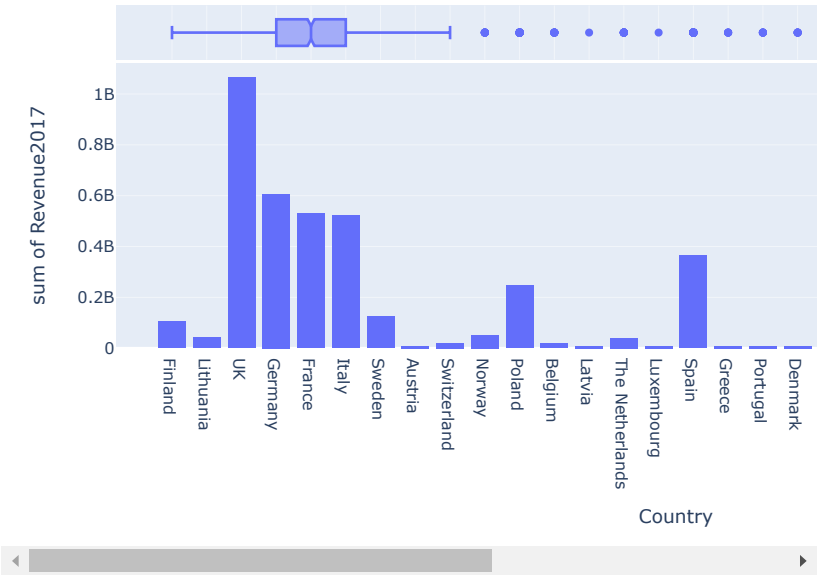
```
In [38]:  
fig= px.histogram(df, x='Country',color='Country', y='Revenue2020',marginal='box'  
                ,hover_data=df.columns,title= 'Revenue distribution relative to countries'  
                width=1000, height=500)  
fig.show()
```

Revenue distribution relative to countries for 2020



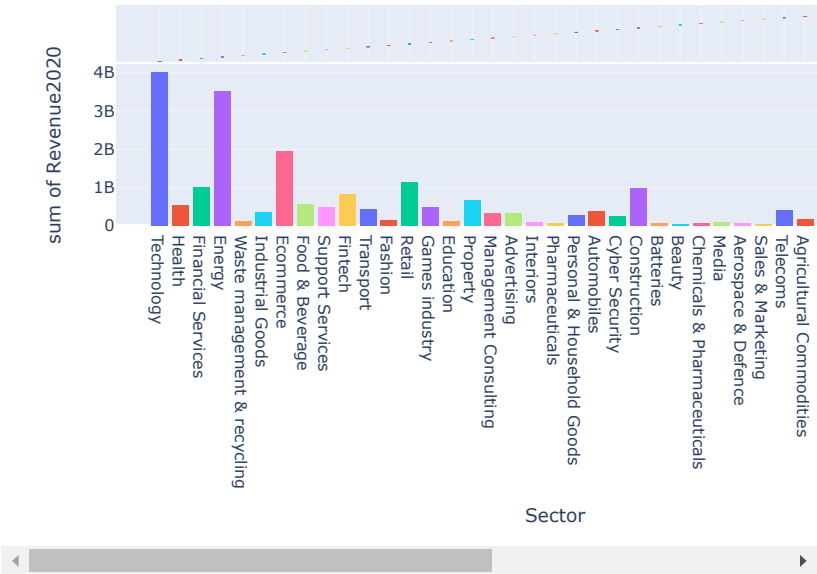
```
In [39]:  
fig= px.histogram(df, x='Country', y='Revenue2017',marginal='box',hover_data=df.columns,  
                  title= 'Revenue distribution relative to countries for 2017',  
                  width=1000, height=500)  
fig.show()
```

Revenue distribution relative to countries for 2017



```
In [40]:  
  
fig= px.histogram(df, x='Sector', color='Sector',y='Revenue2020',  
                  marginal='box',hover_data=df.columns,  
                  title= 'Sector revenues for 2020', width=1000, height=500)  
  
fig.show()
```

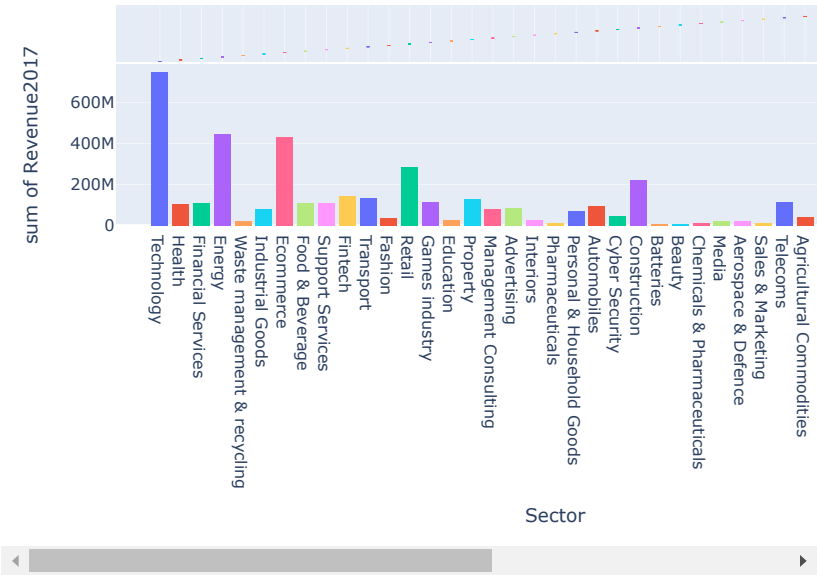
Sector revenues for 2020





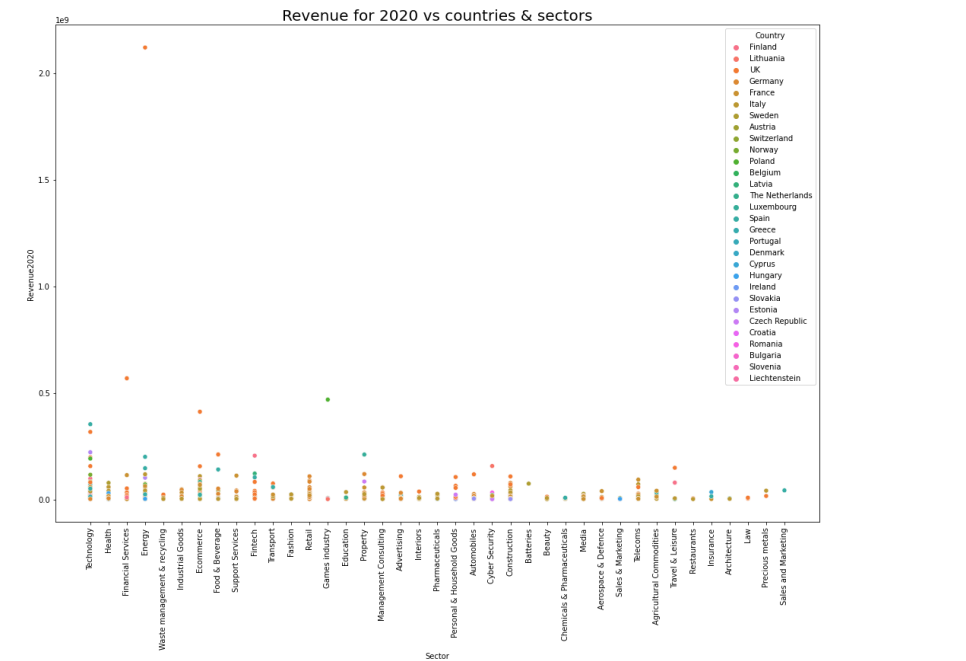
```
In [41]:  
  
fig = px.histogram(df, x='Sector', color='Sector',y='Revenue2017',marginal='box',  
                  hover_data=df.columns,title= 'Sector revenues for 2017',  
                  width=1000, height=500)  
fig.show()
```

Sector revenues for 2017

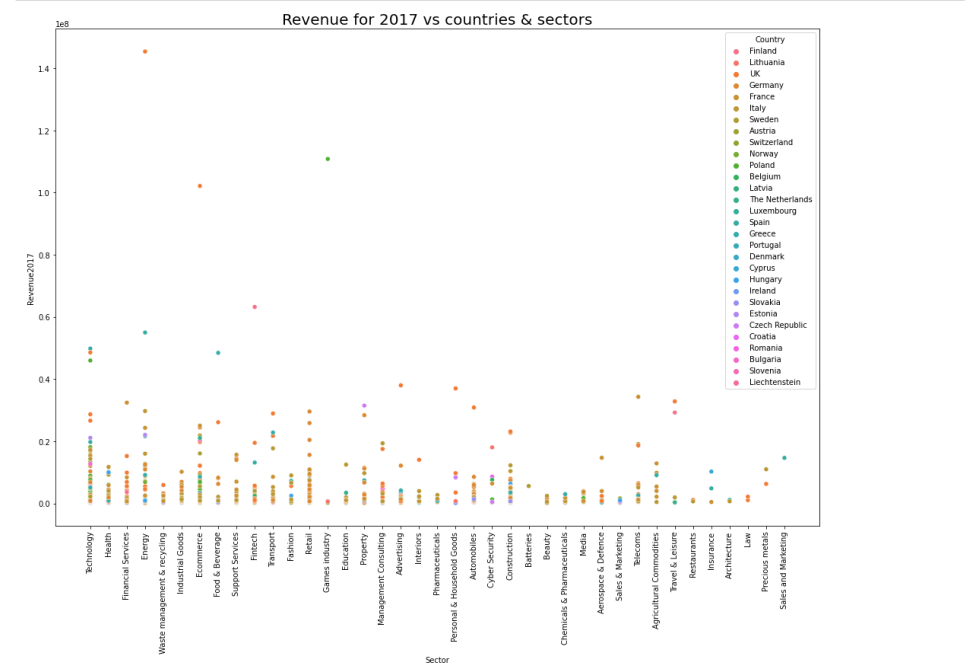


In [42]:

```
plt.figure(figsize=(18,12))
sns.scatterplot(data=df, x='Sector', y='Revenue2020', hue='Country', legend=True)
plt.xticks(rotation=90)
plt.title('Revenue for 2020 vs countries & sectors',fontsize = 20)
plt.show()
```

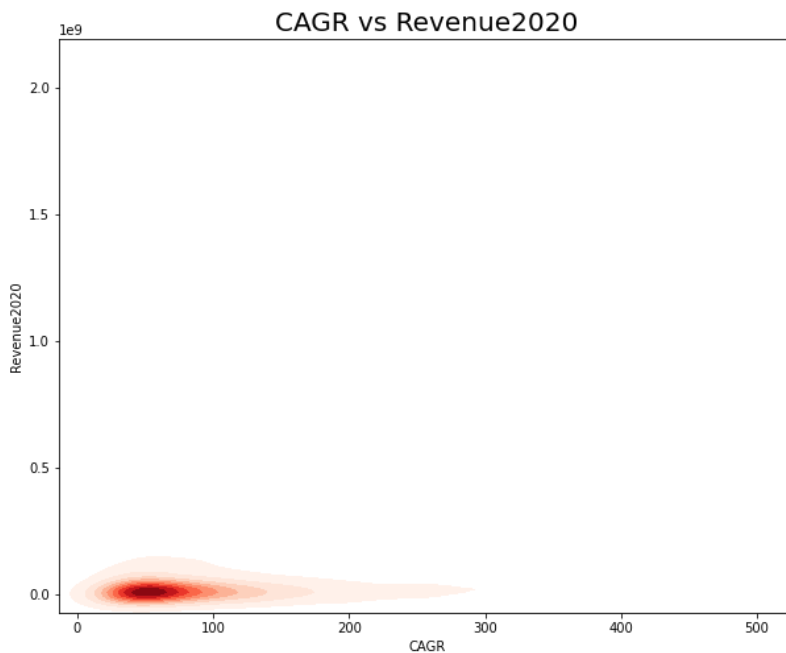


```
In [43]:  
plt.figure(figsize=(18,12))  
sns.scatterplot(data=df, x='Sector', y='Revenue2017', hue='Country', legend=True)  
plt.xticks(rotation=90)  
plt.title('Revenue for 2017 vs countries & sectors',fontsize = 20)  
plt.show()
```



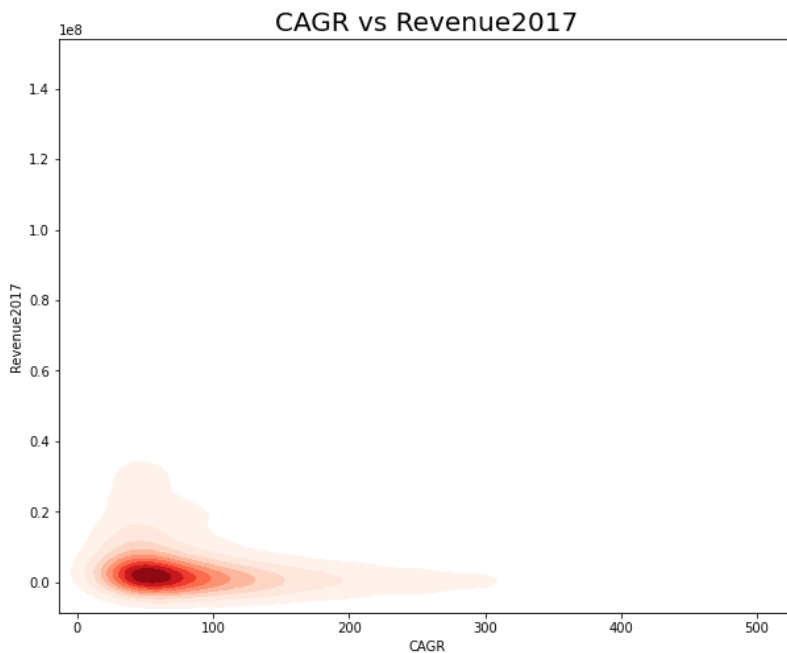
In [44]:

```
plt.figure(figsize=(10,8))  
sns.kdeplot(df['CAGR'],df['Revenue2020'],shade=True,cmap='Reds', shade_lowest=False)  
plt.title('CAGR vs Revenue2020',fontsize = 20)  
plt.show()
```



In [45]:

```
plt.figure(figsize=(10,8))
sns.kdeplot(df['CAGR'],df['Revenue2017'],shade=True,cmap='Reds', shade_lowest=False)
plt.title('CAGR vs Revenue2017',fontsize = 20)
plt.show()
```



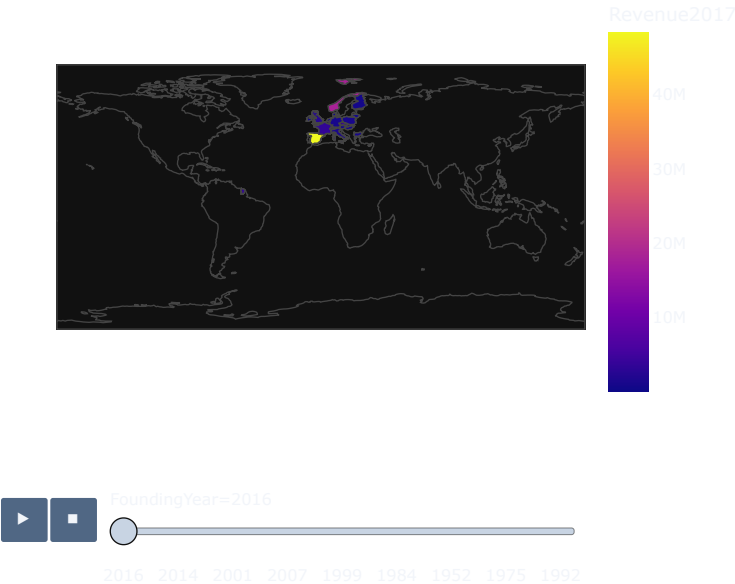
```
In [46]:  
fig=px.choropleth(data_frame=df,locations=df['Country'],locationmode='country names',color=d  
fig.update_layout(dict1={'title':'Revenue distribution relative to countries for 2020 on the  
fig.show()
```

Revenue distribution relative to countries for 2020 on the map

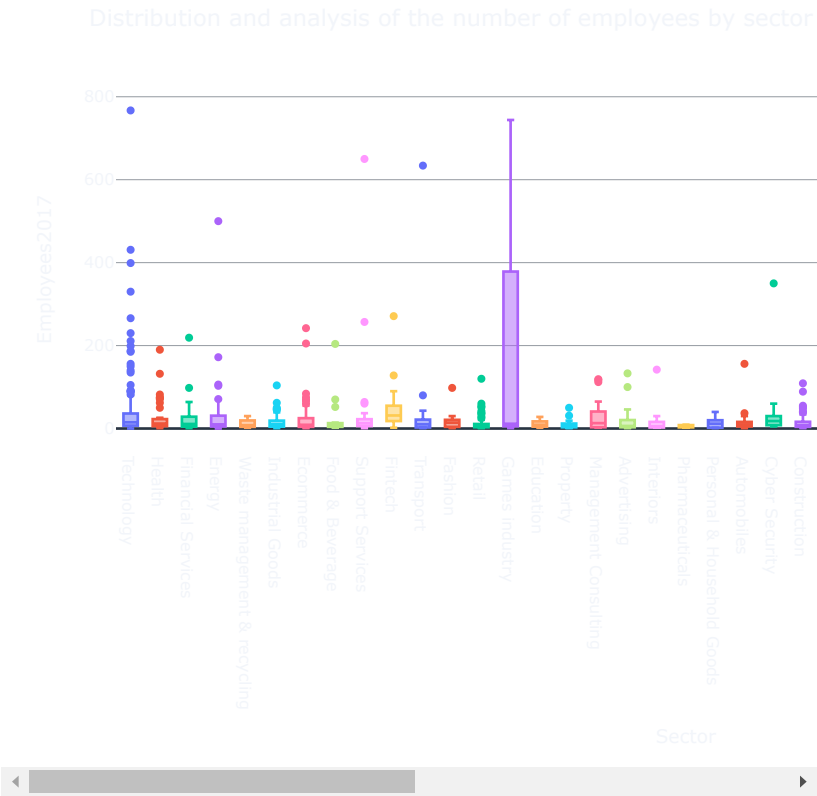


```
In [47]:  
fig=px.choropleth(data_frame=df,locations=df['Country'],locationmode='country names',color=d  
fig.update_layout(dict1={'title':'Revenue distribution relative to countries for 2017 on the  
fig.show()
```

Revenue distribution relative to countries for 2017 on the map



```
In [48]:  
fig = px.box(df, x='Sector', color='Sector', y='Employees2017',  
             title='Distribution and analysis of the number of employees by sector in 2017',  
             fig.show())
```





```
In [49]:  
fig = px.box(df, x='Sector', color='Sector', y='Employees2020',  
            title= 'Distribution and analysis of the number of employees by sector in 2020')  
fig.show()
```

