In [1]:
```
import pandas as pd
import matplotlib.pyplot as plt
import numpy as np
import seaborn as sns
```

In [2]:
```
df = pd.read_csv('data-sciencefile.csv')
```

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

Out[3]:

	Unnamed: 0	work_year	experience_level	employment_type	job_title	salary	salary_currency	salary_in_usd	employee_residence	remote_ratio	company_location	company_size
0	0	2020	MI	FT	Data Scientist	70000	EUR	79833	DE	0	DE	L
1	1	2020	SE	FT	Machine Learning Scientist	260000	USD	260000	JP	0	JP	S
2	2	2020	SE	FT	Big Data Engineer	85000	GBP	109024	GB	50	GB	M
3	3	2020	MI	FT	Product Data Analyst	20000	USD	20000	HN	0	HN	S
4	4	2020	SE	FT	Machine Learning Engineer	150000	USD	150000	US	50	US	L
...	...	...	...	...	...	...	...	...	...	...	...	...
602	602	2022	SE	FT	Data Engineer	154000	USD	154000	US	100	US	M
603	603	2022	SE	FT	Data Engineer	126000	USD	126000	US	100	US	M
604	604	2022	SE	FT	Data Analyst	129000	USD	129000	US	0	US	M
605	605	2022	SE	FT	Data Analyst	150000	USD	150000	US	100	US	M
606	606	2022	MI	FT	AI Scientist	200000	USD	200000	IN	100	US	L

607 rows × 12 columns

In [ ]:

In [4]:
```
df.shape
```

Out[4]:
```
(607, 12)
```

In [8]:
```
df.drop(['Unnamed: 0', 'salary'], axis = 1, inplace = True)
```

In [9]:
```
df
```

Out[9]:

	work_year	experience_level	employment_type	job_title	salary_currency	salary_in_usd	employee_residence	remote_ratio	company_location	company_size
0	2020	MI	FT	Data Scientist	EUR	79833	DE	0	DE	L
1	2020	SE	FT	Machine Learning Scientist	USD	260000	JP	0	JP	S
2	2020	SE	FT	Big Data Engineer	GBP	109024	GB	50	GB	M
3	2020	MI	FT	Product Data Analyst	USD	20000	HN	0	HN	S
4	2020	SE	FT	Machine Learning Engineer	USD	150000	US	50	US	L
...	...	...	...	...	...	...	...	...	...	...
602	2022	SE	FT	Data Engineer	USD	154000	US	100	US	M
603	2022	SE	FT	Data Engineer	USD	126000	US	100	US	M
604	2022	SE	FT	Data Analyst	USD	129000	US	0	US	M
605	2022	SE	FT	Data Analyst	USD	150000	US	100	US	M
606	2022	MI	FT	AI Scientist	USD	200000	IN	100	US	L

607 rows × 10 columns

In [ ]:

In [15]:
```
df1 = df.groupby('work_year')['salary_in_usd'].mean().round(2)
df1
```

Out[15]:
```
work_year
2020     95813.00
2021     99853.79
2022    124522.01
Name: salary_in_usd, dtype: float64
```

In [17]:
```
df1.index
```

Out[17]:
```
Index([2020, 2021, 2022], dtype='int64', name='work_year')
```

In [32]:
```
df1.values
```

Out[32]:
```
array([[2020.  ,   95.81],
       [2021.  ,   99.85],
       [2022.  ,  124.52]])
```

In [19]:
```
data = {
    'work_year':df1.index,
    'average_salary':df1.values
}

df1 = pd.DataFrame(data)
df1
```

Out[19]:

	work_year	average_salary
0	2020	95813.00
1	2021	99853.79
2	2022	124522.01

In [21]:
```
df1['average_salary'] = (df1['average_salary']/1000).round(2)
```

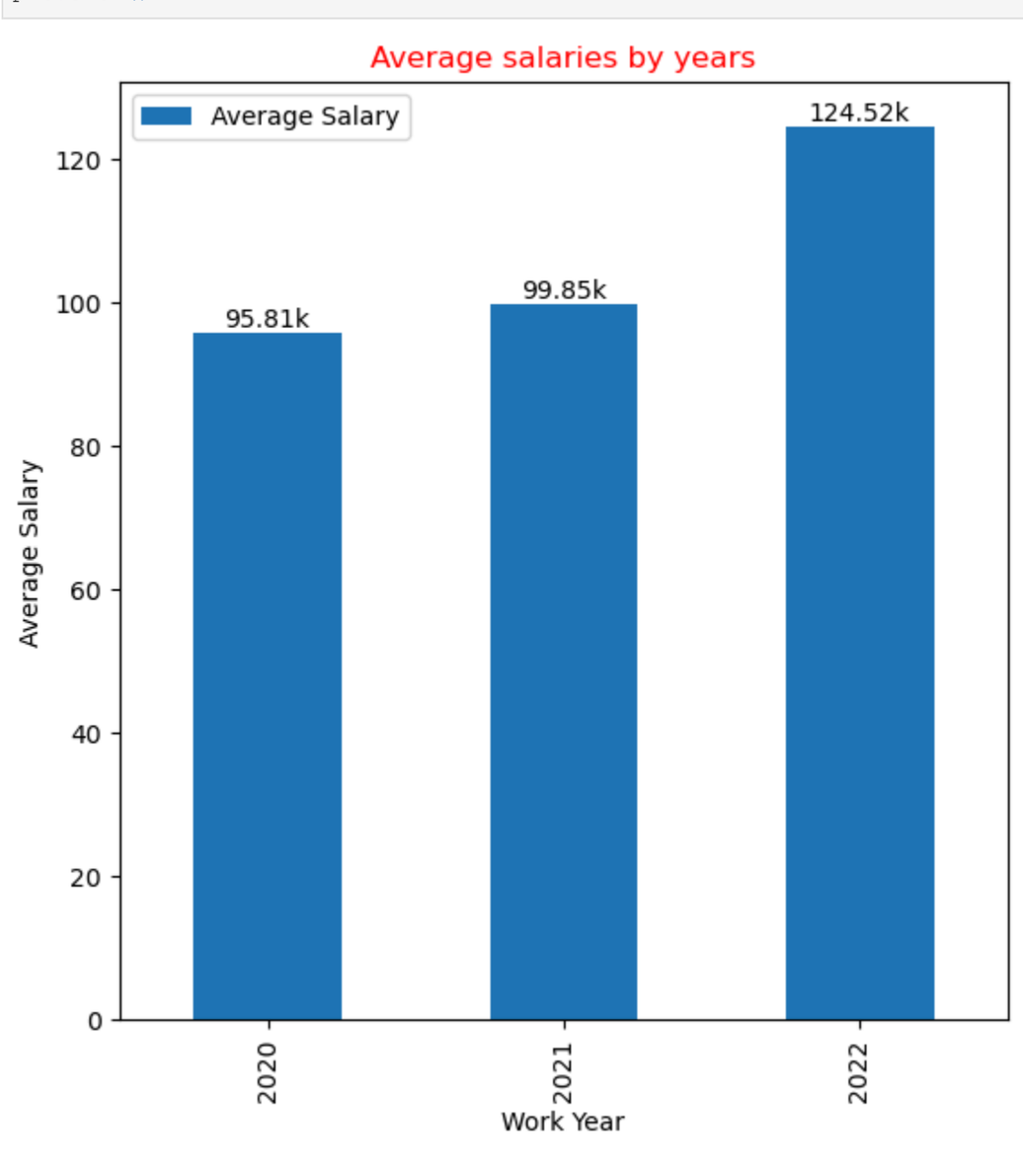
In [22]:
```
df1
```

Out[22]:

	work_year	average_salary
0	2020	95.81
1	2021	99.85
2	2022	124.52

In [34]:
```
ax = df1.plot(kind = 'bar', x = 'work_year', y ='average_salary', legend = True)
ax.bar_label(ax.containers[0], labels = df1['average_salary'].map('{:.2f}k'.format))
plt.subplots_adjust(top = 1.2)
ax.legend(['Average Salary'], loc = 'upper left')

plt.xlabel('Work Year')
plt.ylabel('Average Salary')
plt.title('Average salaries by years', color = 'red')
plt.show()
```



In [ ]:

In [35]:
```
df
```

Out[35]:

	work_year	experience_level	employment_type	job_title	salary_currency	salary_in_usd	employee_residence	remote_ratio	company_location	company_size
0	2020	MI	FT	Data Scientist	EUR	79833	DE	0	DE	L
1	2020	SE	FT	Machine Learning Scientist	USD	260000	JP	0	JP	S
2	2020	SE	FT	Big Data Engineer	GBP	109024	GB	50	GB	M
3	2020	MI	FT	Product Data Analyst	USD	20000	HN	0	HN	S
4	2020	SE	FT	Machine Learning Engineer	USD	150000	US	50	US	L
...	...	...	...	...	...	...	...	...	...	...
602	2022	SE	FT	Data Engineer	USD	154000	US	100	US	M
603	2022	SE	FT	Data Engineer	USD	126000	US	100	US	M
604	2022	SE	FT	Data Analyst	USD	129000	US	0	US	M
605	2022	SE	FT	Data Analyst	USD	150000	US	100	US	M
606	2022	MI	FT	AI Scientist	USD	200000	IN	100	US	L

607 rows × 10 columns

In [ ]:

In [38]:
```
df2 = df.remote_ratio.value_counts()
df2
```

Out[38]:
```
remote_ratio
100    381
0      127
50      99
Name: count, dtype: int64
```

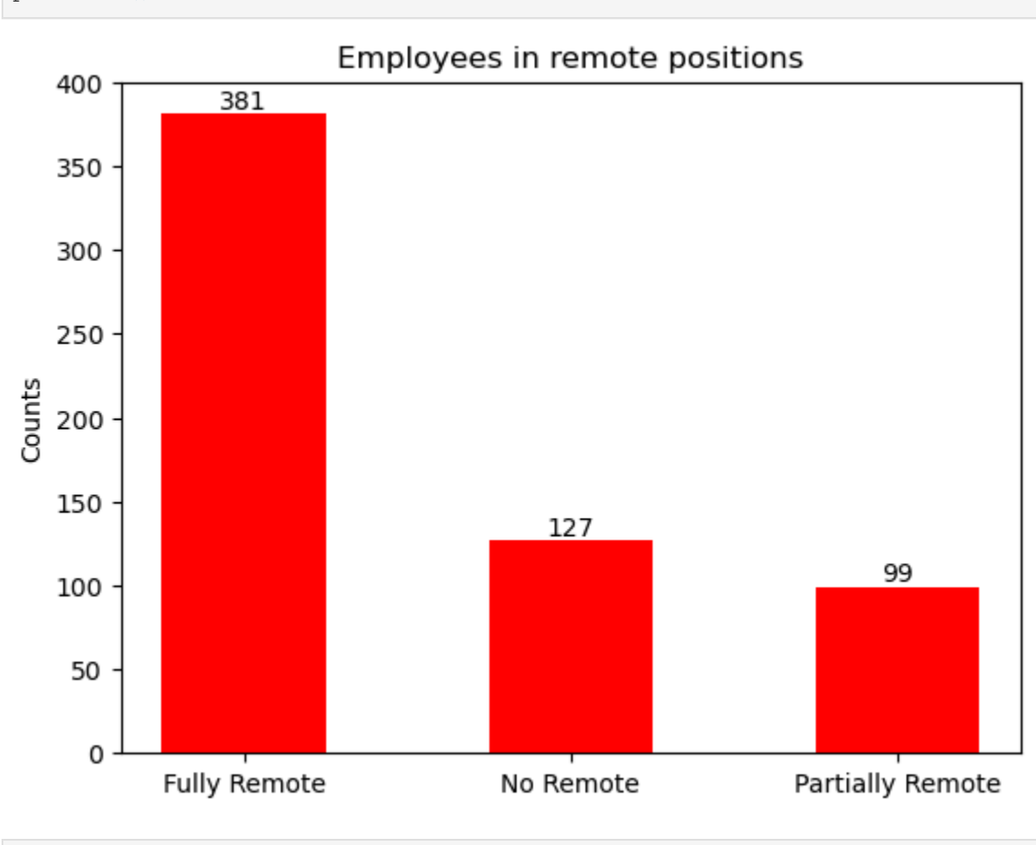
In [40]:
```
values = df2.to_list()
```

In [46]:
```
values
```

Out[46]:
```
[381, 127, 99]
```

In [43]:
```
labels = ['Fully Remote', 'No Remote', 'Partially Remote']
```

In [48]:
```
z = plt.bar(labels,values, width = 0.5, color = 'red')
plt.bar_label(z, labels = values)
plt.title('Employees in remote positions')
plt.ylabel('Counts')
plt.show()
```



In [ ]:

In [49]:
```
df
```

Out[49]:

	work_year	experience_level	employment_type	job_title	salary_currency	salary_in_usd	employee_residence	remote_ratio	company_location	company_size
0	2020	MI	FT	Data Scientist	EUR	79833	DE	0	DE	L
1	2020	SE	FT	Machine Learning Scientist	USD	260000	JP	0	JP	S
2	2020	SE	FT	Big Data Engineer	GBP	109024	GB	50	GB	M
3	2020	MI	FT	Product Data Analyst	USD	20000	HN	0	HN	S
4	2020	SE	FT	Machine Learning Engineer	USD	150000	US	50	US	L
...	...	...	...	...	...	...	...	...	...	...
602	2022	SE	FT	Data Engineer	USD	154000	US	100	US	M
603	2022	SE	FT	Data Engineer	USD	126000	US	100	US	M
604	2022	SE	FT	Data Analyst	USD	129000	US	0	US	M
605	2022	SE	FT	Data Analyst	USD	150000	US	100	US	M
606	2022	MI	FT	AI Scientist	USD	200000	IN	100	US	L

607 rows × 10 columns

In [ ]:

In [52]:
```
df3 = df.company_size.value_counts()
```

In [53]:
```
df3
```

Out[53]:
```
company_size
M    326
L    198
S     83
Name: count, dtype: int64
```

In [55]:
```
df3.index.to_list()
```

Out[55]:
```
['M', 'L', 'S']
```

In [56]:
```
df3 .to_list()
```

Out[56]:
```
[326, 198, 83]
```

In [58]:
```
values2 = df3.values
```

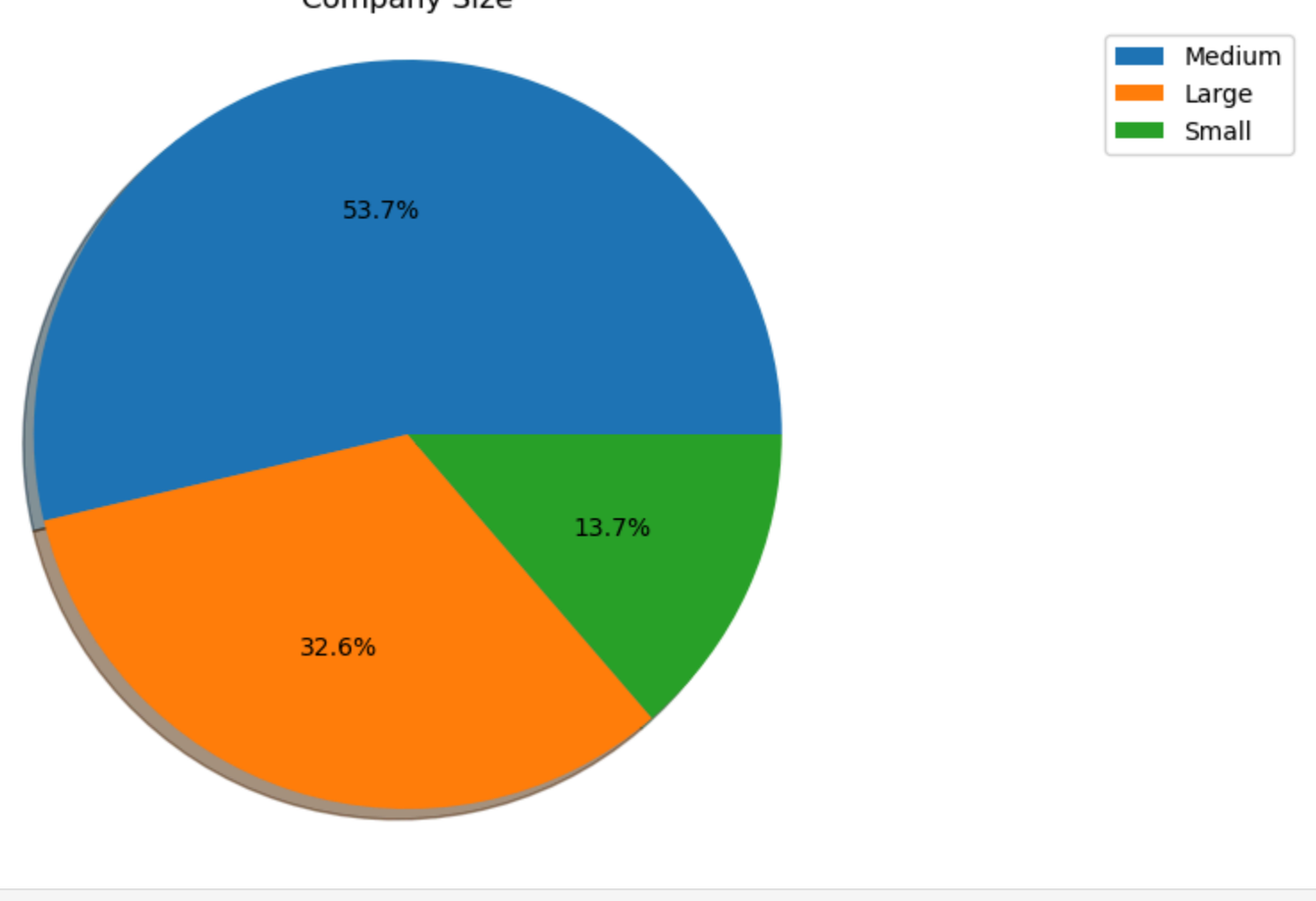
In [59]:
```
values2
```

Out[59]:
```
array([326, 198,  83], dtype=int64)
```

In [ ]:

In [63]:
```
labels_for_company = ['Medium', 'Large', 'Small']
```

In [67]:
```
plt.figure(figsize = (13,6))
plt.pie(x = values2, labels = None, autopct = '%1.1f%%', shadow = True)
plt.axis ('equal')
plt.legend(labels = labels_for_company, loc = 'upper right')
plt.title('Company Size')
plt.show()
```



In [70]:
```
df4 = df.job_title.value_counts()[:5]
```

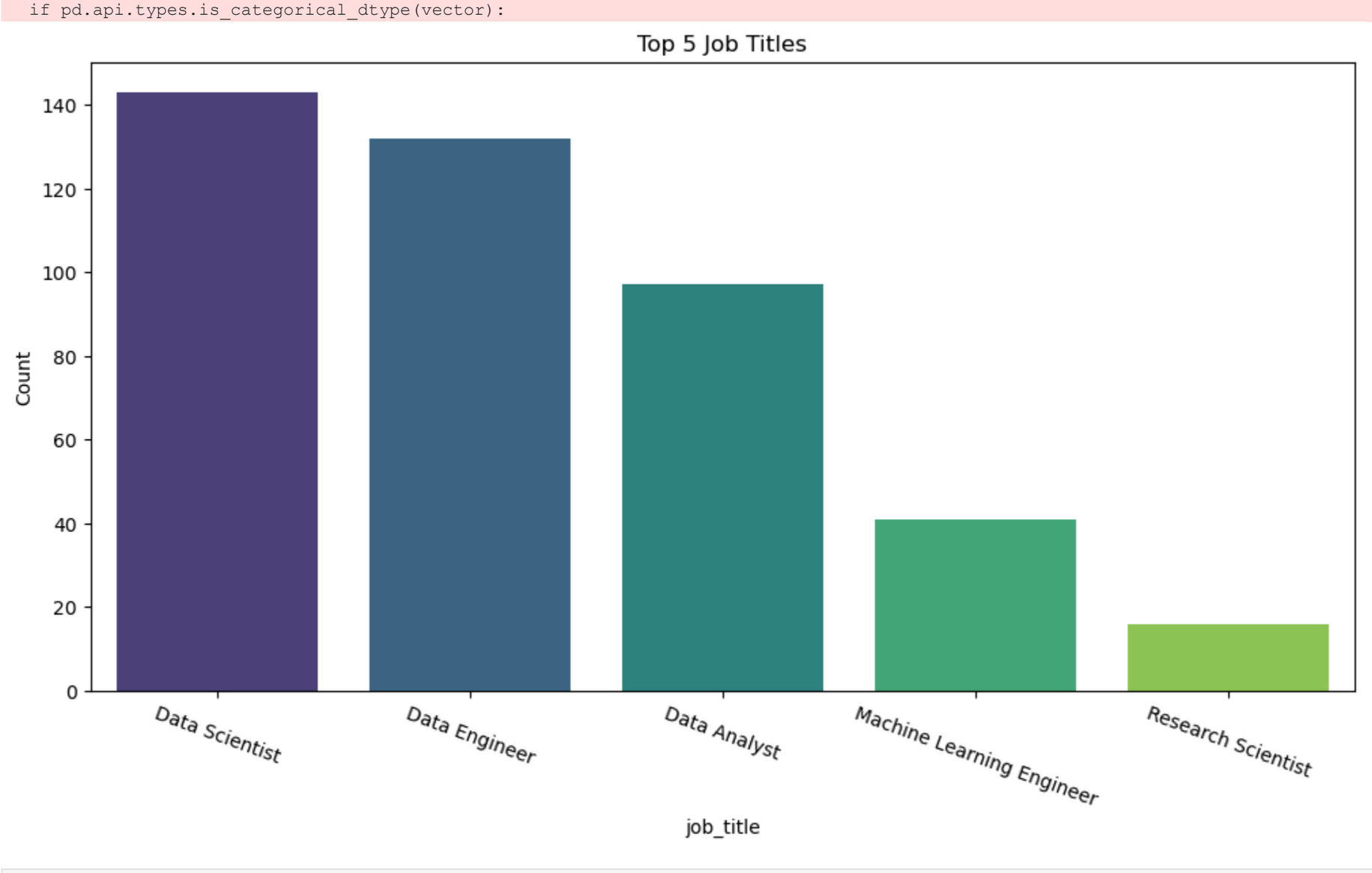
In [71]:
```
df4
```

Out[71]:
```
job_title
Data Scientist               143
Data Engineer                132
Data Analyst                  97
Machine Learning Engineer     41
Research Scientist            16
Name: count, dtype: int64
```

In [ ]:

In [78]:
```
plt.figure(figsize = (12,6))
#plt.bar(x = df4.index, height = df4.values)
sns.barplot(x = df4.index, y = df4.values, palette = 'viridis' )
plt.title('Top 5 Job Titles')
plt.ylabel('Count')
plt.xticks(rotation = -20)
plt.show()
```

```
C:\Users\Emenike favour\anaconda\Lib\site-packages\seaborn\_oldcore.py:1498: FutureWarning: is_categorical_dtype is deprecated and will be removed in a future version. Use isinstance(dtype, CategoricalDtype) instead
  if pd.api.types.is_categorical_dtype(vector):
C:\Users\Emenike favour\anaconda\Lib\site-packages\seaborn\_oldcore.py:1498: FutureWarning: is_categorical_dtype is deprecated and will be removed in a future version. Use isinstance(dtype, CategoricalDtype) instead
  if pd.api.types.is_categorical_dtype(vector):
C:\Users\Emenike favour\anaconda\Lib\site-packages\seaborn\_oldcore.py:1498: FutureWarning: is_categorical_dtype is deprecated and will be removed in a future version. Use isinstance(dtype, CategoricalDtype) instead
  if pd.api.types.is_categorical_dtype(vector):
```



In [ ]:

In [79]:
```
df
```

Out[79]:

	work_year	experience_level	employment_type	job_title	salary_currency	salary_in_usd	employee_residence	remote_ratio	company_location	company_size
0	2020	MI	FT	Data Scientist	EUR	79833	DE	0	DE	L
1	2020	SE	FT	Machine Learning Scientist	USD	260000	JP	0	JP	S
2	2020	SE	FT	Big Data Engineer	GBP	109024	GB	50	GB	M
3	2020	MI	FT	Product Data Analyst	USD	20000	HN	0	HN	S
4	2020	SE	FT	Machine Learning Engineer	USD	150000	US	50	US	L
...	...	...	...	...	...	...	...	...	...	...
602	2022	SE	FT	Data Engineer	USD	154000	US	100	US	M
603	2022	SE	FT	Data Engineer	USD	126000	US	100	US	M
604	2022	SE	FT	Data Analyst	USD	129000	US	0	US	M
605	2022	SE	FT	Data Analyst	USD	150000	US	100	US	M
606	2022	MI	FT	AI Scientist	USD	200000	IN	100	US	L

607 rows × 10 columns

In [ ]:

In [80]:
```
df5 = df[['salary_in_usd', 'company_size']]
df5
```

Out[80]:

	salary_in_usd	company_size
0	79833	L
1	260000	S
2	109024	M
3	20000	S
4	150000	L
...	...	...
602	154000	M
603	126000	M
604	129000	M
605	150000	M
606	200000	L

607 rows × 2 columns

In [83]:
```
S = df5[df5['company_size'] == 'S']
M = df5[df5['company_size'] == 'M']
L = df5[df5['company_size'] == 'L']

labels = ['Medium', 'Large', 'Small']

sal_mean = [S['salary_in_usd'].mean(), M['salary_in_usd'].mean(), L['salary_in_usd'].mean()]
```

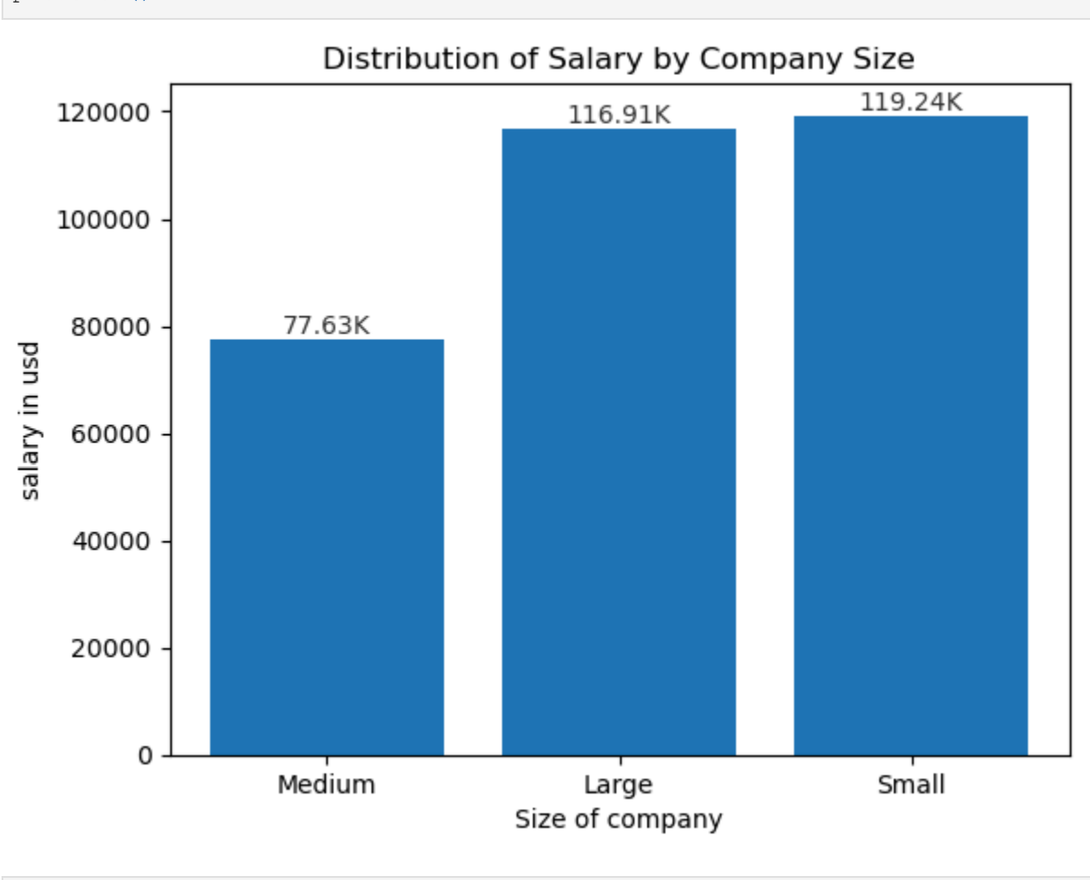
In [84]:
```
sal_mean
```

Out[84]:
```
[77632.67469879518, 116905.46625766871, 119242.99494949495]
```

In [86]:
```
label_change = np.round([x/1000 for x in sal_mean], 2)
label_change = list(map(str, label_change))
label_change = [x + 'K' for x in label_change]
label_change
```

Out[86]:
```
['77.63K', '116.91K', '119.24K']
```

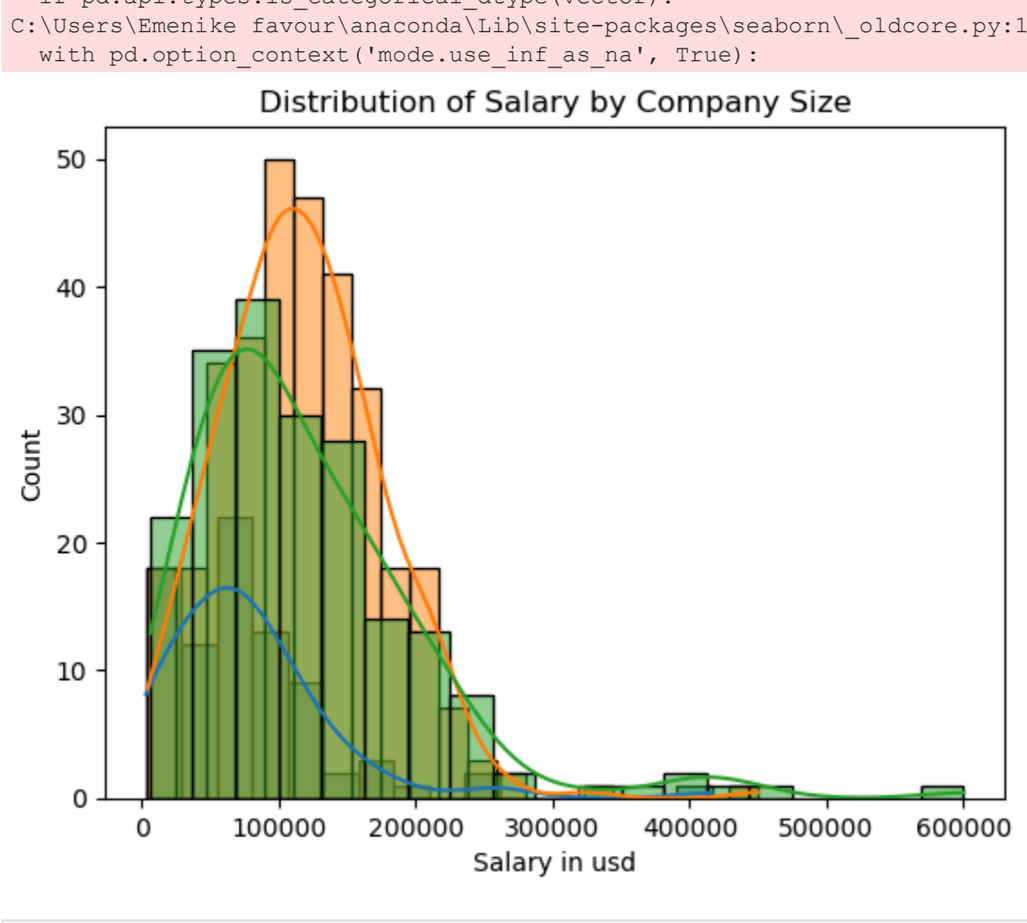
In [91]:
```
z = plt.bar(labels, sal_mean)
plt.bar_label(z,label_change, alpha = 0.8)
plt.title('Distribution of Salary by Company Size')
plt.xlabel('Size of company')
plt.ylabel('salary in usd')
plt.show()
```



In [ ]:

In [95]:
```
sns.histplot(S['salary_in_usd'], label = "Small", kde = True)
sns.histplot(M['salary_in_usd'], label = "Medium", kde = True)
sns.histplot(L['salary_in_usd'], label = "Large", kde = True)
plt.title('Distribution of Salary by Company Size')
plt.xlabel('Salary in usd')
plt.show()
```

```
C:\Users\Emenike favour\anaconda\Lib\site-packages\seaborn\_oldcore.py:1498: FutureWarning: is_categorical_dtype is deprecated and will be removed in a future version. Use isinstance(dtype, CategoricalDtype) instead
  if pd.api.types.is_categorical_dtype(vector):
C:\Users\Emenike favour\anaconda\Lib\site-packages\seaborn\_oldcore.py:1119: FutureWarning: use_inf_as_na option is deprecated and will be removed in a future version. Convert inf values to NaN before operating instead.
  with pd.option_context('mode.use_inf_as_na', True):
C:\Users\Emenike favour\anaconda\Lib\site-packages\seaborn\_oldcore.py:1498: FutureWarning: is_categorical_dtype is deprecated and will be removed in a future version. Use isinstance(dtype, CategoricalDtype) instead
  if pd.api.types.is_categorical_dtype(vector):
C:\Users\Emenike favour\anaconda\Lib\site-packages\seaborn\_oldcore.py:1119: FutureWarning: use_inf_as_na option is deprecated and will be removed in a future version. Convert inf values to NaN before operating instead.
  with pd.option_context('mode.use_inf_as_na', True):
C:\Users\Emenike favour\anaconda\Lib\site-packages\seaborn\_oldcore.py:1498: FutureWarning: is_categorical_dtype is deprecated and will be removed in a future version. Use isinstance(dtype, CategoricalDtype) instead
  if pd.api.types.is_categorical_dtype(vector):
C:\Users\Emenike favour\anaconda\Lib\site-packages\seaborn\_oldcore.py:1119: FutureWarning: use_inf_as_na option is deprecated and will be removed in a future version. Convert inf values to NaN before operating instead.
  with pd.option_context('mode.use_inf_as_na', True):
```



In [ ]:

In [96]:
```
df
```

Out[96]:

	work_year	experience_level	employment_type	job_title	salary_currency	salary_in_usd	employee_residence	remote_ratio	company_location	company_size
0	2020	MI	FT	Data Scientist	EUR	79833	DE	0	DE	L
1	2020	SE	FT	Machine Learning Scientist	USD	260000	JP	0	JP	S
2	2020	SE	FT	Big Data Engineer	GBP	109024	GB	50	GB	M
3	2020	MI	FT	Product Data Analyst	USD	20000	HN	0	HN	S
4	2020	SE	FT	Machine Learning Engineer	USD	150000	US	50	US	L
...	...	...	...	...	...	...	...	...	...	...
602	2022	SE	FT	Data Engineer	USD	154000	US	100	US	M
603	2022	SE	FT	Data Engineer	USD	126000	US	100	US	M
604	2022	SE	FT	Data Analyst	USD	129000	US	0	US	M
605	2022	SE	FT	Data Analyst	USD	150000	US	100	US	M
606	2022	MI	FT	AI Scientist	USD	200000	IN	100	US	L

607 rows × 10 columns

In [ ]:

In [97]:
```
df6 = df.experience_level.value_counts()
```

In [100…
```
df6
```

Out[100]:
```
experience_level
SE    280
MI    213
EN     88
EX     26
Name: count, dtype: int64
```

In [103…
```
df6.index
```

Out[103]:
```
Index(['SE', 'MI', 'EN', 'EX'], dtype='object', name='experience_level')
```

In [101…
```
df6.index.to_list()
```

Out[101]:
```
['SE', 'MI', 'EN', 'EX']
```

In [ ]:

In [102…
```
#df['experience_level'].replace('SE', 'Senior', inplace = True)
```

In [107…
```
exp_map ={
    'SE':'Seniours',
    'MI':'Middle',
    'EN':'Entry level',
    'EX':'Executive'
}
```

In [108…
```
df['experience_level'].replace(exp_map, inplace = True)
```

In [ ]:

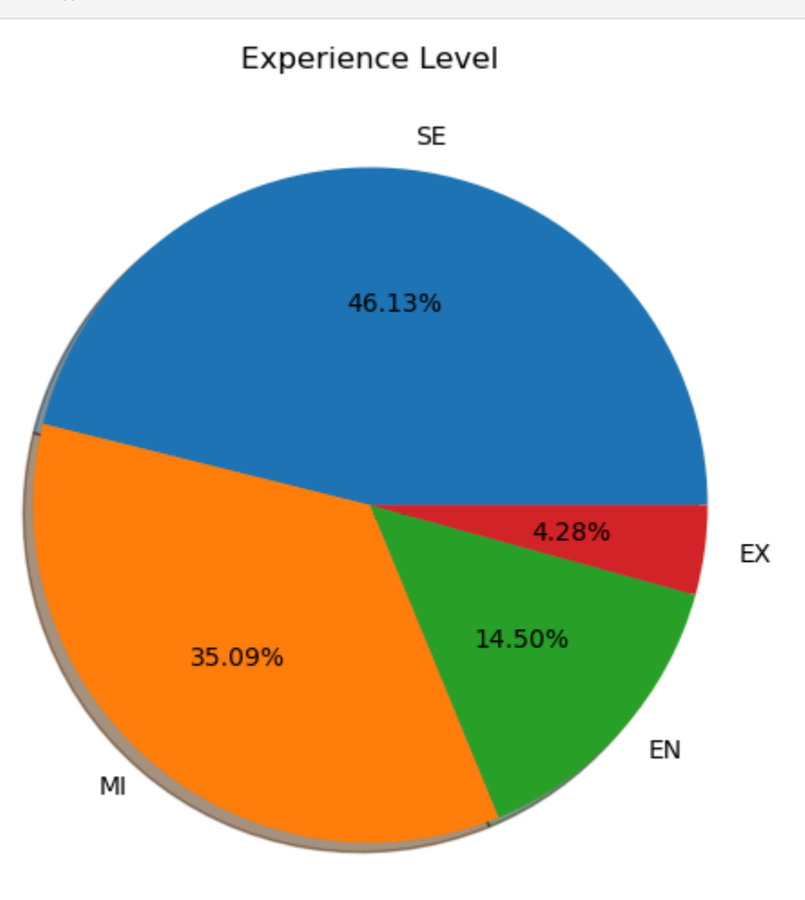
In [109…
```
labels = df6.index.to_list()
labels
```

Out[109]:
```
['SE', 'MI', 'EN', 'EX']
```

In [115…
```
values = df6.values
values
```

Out[115]:
```
array([280, 213,  88,  26], dtype=int64)
```

In [121…
```
plt.figure(figsize= (6,12))
plt.pie(x = values, labels = labels, autopct='%1.2f%%', shadow = 2)
plt.title('Experience Level')
plt.show()
```



In [ ]: