Analyzing Data Science Salaries 2023 with Python

```
Data Science Job Salaries Dataset contains 11 columns, each are:

work_year: The year the salary was paid.
experience_level: The experience level in the job during the year
employment_type: The type of employment for the role
job_title: The role worked in during the year.
salary: The total gross salary amount paid.
salary_currency: The currency of the salary paid as an ISO 4217 currency
code.
salaryinusd: The salary in USD
employee_residence: Employee's primary country of residence in during the
work year as an ISO 3166 country code.
remote_ratio: The overall amount of work done remotely
company_location: The country of the employer's main office or contracting
branch
company_size: The median number of people that worked for the company during
the year
```

Import Library

In [2]:
```
import pandas as pd
```

In [18]:
```
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
import seaborn as sns
import plotly.express as px
```

Uploading Csv fle

In [19]:
```
df = pd.read_csv(r"C:\Users\Syed Arif\Downloads\ds_salaries.csv")
```

Data Preprocessing

.head()

head is used show to the By default = 5 rows in the dataset

In [20]:
```
df.head()
```

Out[20]:

	work_year	experience_level	employment_type	job_title	salary	salary_currency	salary_in_us
0	2023	SE	FT	Principal Data Scientist	80000	EUR	858
1	2023	MI	CT	ML Engineer	30000	USD	300
2	2023	MI	CT	ML Engineer	25500	USD	255
3	2023	SE	FT	Data Scientist	175000	USD	1750
4	2023	SE	FT	Data Scientist	120000	USD	1200

.tail()

tail is used to show rows by Descending order

In [21]:
```
df.tail()
```

Out[21]:

	work_year	experience_level	employment_type	job_title	salary	salary_currency	salary_
3750	2020	SE	FT	Data Scientist	412000	USD	
3751	2021	MI	FT	Principal Data Scientist	151000	USD	
3752	2020	EN	FT	Data Scientist	105000	USD	
3753	2020	EN	CT	Business Data Analyst	100000	USD	
3754	2021	SE	FT	Data Science Manager	7000000	INR	

.shape

It show the total no of rows & Column in the dataset

In [22]:
```
df.shape
```

Out[22]:
```
(3755, 11)
```

.Columns

It show the no of each Column

In [23]:
```
df.columns
```

Out[23]:
```
Index(['work_year', 'experience_level', 'employment_type', 'job_title',
       'salary', 'salary_currency', 'salary_in_usd', 'employee_residence',
       'remote_ratio', 'company_location', 'company_size'],
      dtype='object')
```

.dtypes

This Attribute show the data type of each column

In [24]:
```
df.dtypes
```

Out[24]:
```
work_year              int64
experience_level      object
employment_type       object
job_title             object
salary                 int64
salary_currency       object
salary_in_usd          int64
employee_residence    object
remote_ratio           int64
company_location      object
company_size          object
dtype: object
```

.unique()

In a column, It show the unique value of specific column.

In [25]:
```
df["employee_residence"].unique()
```

Out[25]:
```
array(['ES', 'US', 'CA', 'DE', 'GB', 'NG', 'IN', 'HK', 'PT', 'NL', 'CH',
       'CF', 'FR', 'AU', 'FI', 'UA', 'IE', 'IL', 'GH', 'AT', 'CO', 'SG',
       'SE', 'SI', 'MX', 'UZ', 'BR', 'TH', 'HR', 'PL', 'KW', 'VN', 'CY',
       'AR', 'AM', 'BA', 'KE', 'GR', 'MK', 'LV', 'RO', 'PK', 'IT', 'MA',
       'LT', 'BE', 'AS', 'IR', 'HU', 'SK', 'CN', 'CZ', 'CR', 'TR', 'CL',
       'PR', 'DK', 'BO', 'PH', 'DO', 'EG', 'ID', 'AE', 'MY', 'JP', 'EE',
       'HN', 'TN', 'RU', 'DZ', 'IQ', 'BG', 'JE', 'RS', 'NZ', 'MD', 'LU',
       'MT'], dtype=object)
```

.nuique()

It will show the total no of unque value from whole data frame

In [26]:
```
df.nunique()
```

Out[26]:
```
work_year               4
experience_level        4
employment_type         4
job_title              93
salary                815
salary_currency        20
salary_in_usd        1035
employee_residence     78
remote_ratio            3
company_location       72
company_size            3
dtype: int64
```

.describe()

It show the Count, mean , median etc

In [27]:
```
df.describe()
```

Out[27]:

	work_year	salary	salary_in_usd	remote_ratio
count	3755.000000	3.755000e+03	3755.000000	3755.000000
mean	2022.373635	1.906956e+05	137570.389880	46.271638
std	0.691448	6.716765e+05	63055.625278	48.589050
min	2020.000000	6.000000e+03	5132.000000	0.000000
25%	2022.000000	1.000000e+05	95000.000000	0.000000
50%	2022.000000	1.380000e+05	135000.000000	0.000000
75%	2023.000000	1.800000e+05	175000.000000	100.000000
max	2023.000000	3.040000e+07	450000.000000	100.000000

.value_counts

It Shows all the unique values with their count

In [28]:
```
df["employee_residence"].value_counts()
```

Out[28]:
```
US    3004
GB     167
CA      85
ES      80
IN      71
      ...
BA       1
AM       1
CY       1
KW       1
MT       1
Name: employee_residence, Length: 78, dtype: int64
```

.isnull()

It shows the how many null values

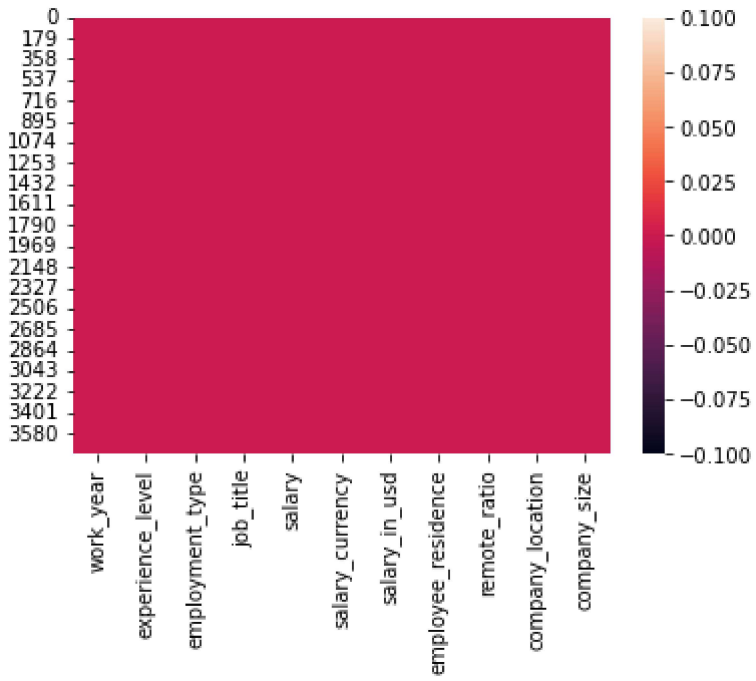
In [29]:
```
df.isnull()
```

Out[29]:

	work_year	experience_level	employment_type	job_title	salary	salary_currency	salary_in_
0	False	False	False	False	False	False	F
1	False	False	False	False	False	False	F
2	False	False	False	False	False	False	F
3	False	False	False	False	False	False	F
4	False	False	False	False	False	False	F
...	...	...	...	...	...	...	
3750	False	False	False	False	False	False	F
3751	False	False	False	False	False	False	F
3752	False	False	False	False	False	False	F
3753	False	False	False	False	False	False	F
3754	False	False	False	False	False	False	F

3755 rows × 11 columns

In [30]:
```
sns.heatmap(df.isnull())
plt.show()
```



Dealing with Categorical features

There's 4 categorical values in column 'Experience Level', each are:

EN, which refers to Entry-level / Junior.

MI, which refers to Mid-level / Intermediate.

SE, which refers to Senior-level / Expert.

EX, which refers to Executive-level / Director.

In [31]:
```
df['experience_level'] = df['experience_level'].replace('EN','Entry-level')
df['experience_level'] = df['experience_level'].replace('MI','Mid-level')
df['experience_level'] = df['experience_level'].replace('SE','Senior-level')
df['experience_level'] = df['experience_level'].replace('EX','Executive-level'

ex_level = df['experience_level'].value_counts()
fig = px.treemap(ex_level, path = [ex_level.index], values = ex_level.values,
                 title = 'Experience Level')
fig.show()
```

```
C:\Users\Syed Arif\anaconda3\lib\site-packages\plotly\express\_core.py:1637:
FutureWarning: The frame.append method is deprecated and will be removed from
pandas in a future version. Use pandas.concat instead.
  df_all_trees = df_all_trees.append(df_tree, ignore_index=True)
```

Employment Type

There are 4 employment types here :

PT : Part-time

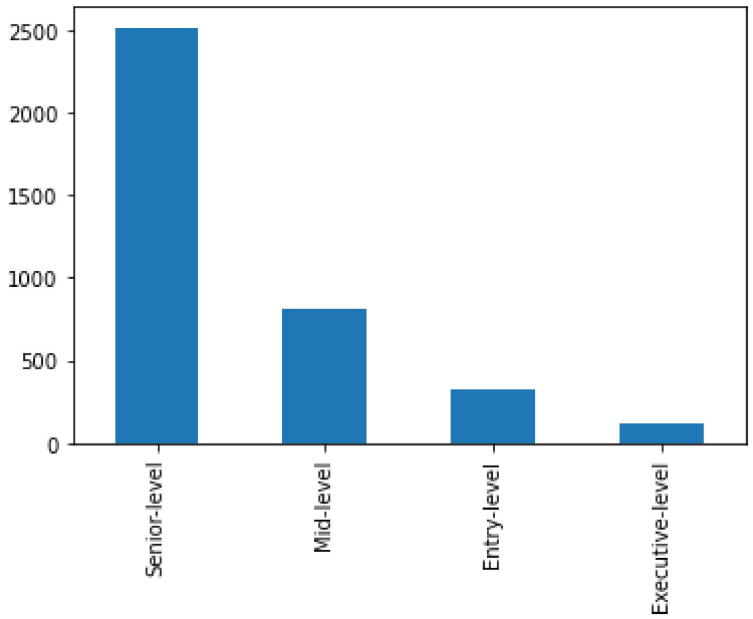
FT : Full-time

CT : Contract

FL : Freelance

In [33]:
```
df['employment_type'] = df['employment_type'].replace('PT','Part-time')
df['employment_type'] = df['employment_type'].replace('FT','Full-time')
df['employment_type'] = df['employment_type'].replace('CT','Contract')
df['employment_type'] = df['employment_type'].replace('FL','Freelance')

df['Experience'] = df['experience_level'].replace('experience_level','Experien

Experience = df['experience_level'].value_counts()

df.Experience.value_counts().plot(kind = "bar")
```

Out[33]: <AxesSubplot:>



Remote Ratio Distribution

In [36]:
```
remote_type = ['Fully Remote', 'Partially Remote', 'No Remote Work']

fig = px.bar(x = remote_type, y = df['remote_ratio'].value_counts().values,
       color = remote_type, text = df['remote_ratio'].value_counts().values,
       title = 'Remote Ratio Distribution')

fig.show()
```

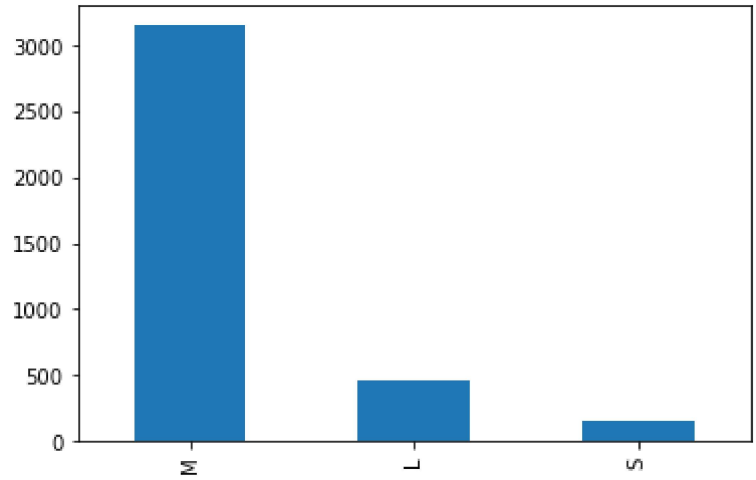
Top 5 Job Titles Based on Experience Level

In [54]:
```
# Group the data by job title and experience level and count the occurrences
experience_counts = df.groupby(['job_title', 'experience_level']).size().reset_

# Sort the counts in descending order and get the top 5 designations
top_5_designations = experience_counts.sort_values(by='count', ascending=False

# Create a bar chart using Plotly Express
fig = px.bar(top_5_designations, x='job_title', y='count', color='experience_l
             labels={'job_title': 'Job Title', 'count': 'Count'},
             title='Top 5 Job Titles Based on Experience Level')

# Show the plot
fig.show()
```

Show the Bar Plot Between Company Size and their Counts

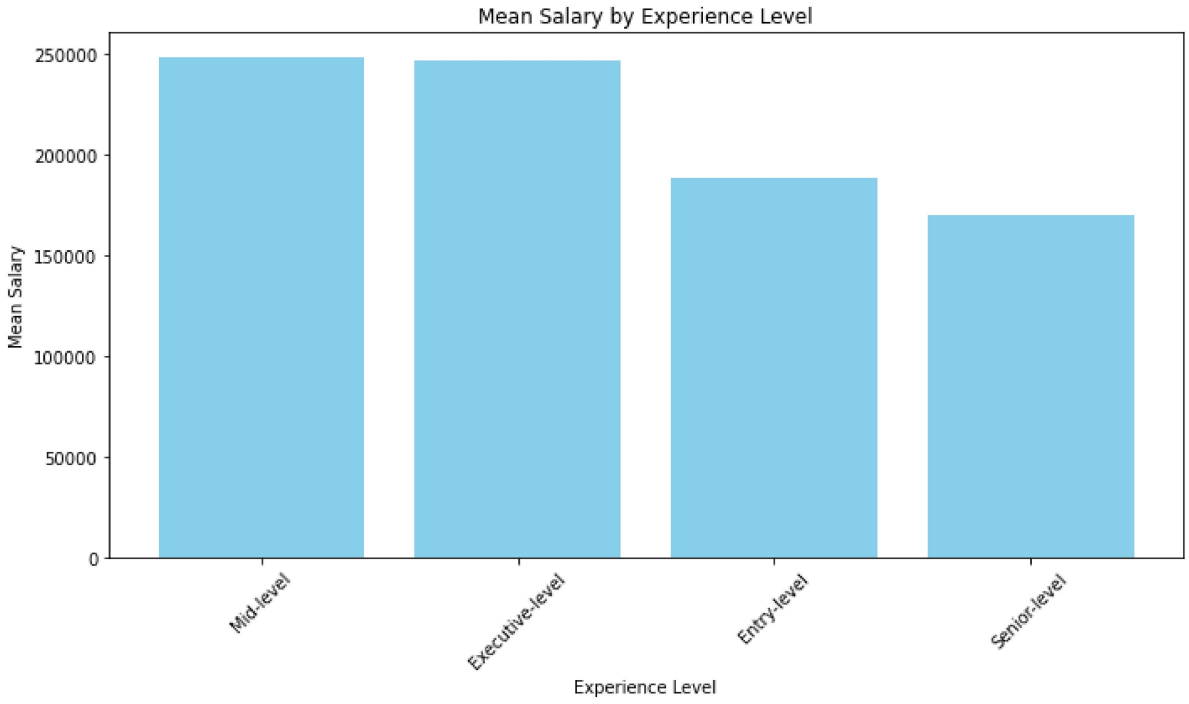
In [56]:
```
df['company_size'] = df['company_size'].replace('PT','Part-time')
df['Size'] = df['company_size'].replace('company_size','Size')

df.Size.value_counts().plot(kind = "bar")
```

Out[56]: <AxesSubplot:>



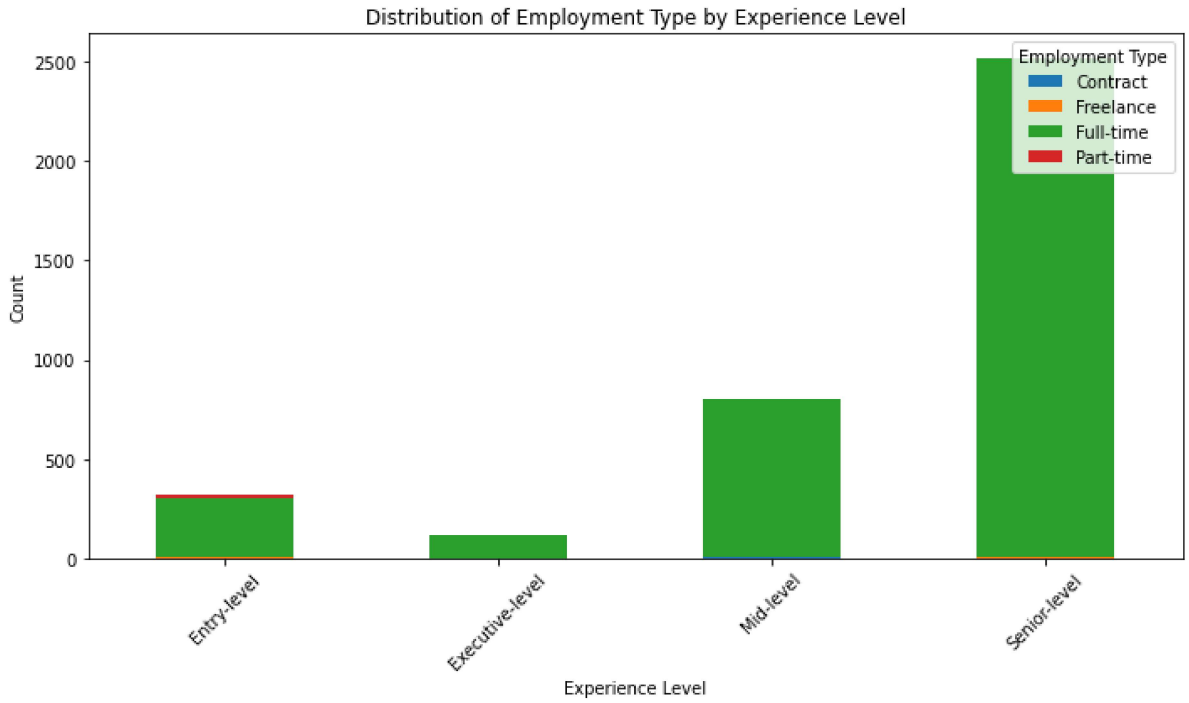
Mean Salary by Experience Level

In [61]:
```
# Group the data by experience level and calculate the mean salary for each gr
experience_salary = df.groupby('experience_level')['salary'].mean().reset_inde

# Sort the data by mean salary in descending order
experience_salary = experience_salary.sort_values(by='salary', ascending=False

# Create a bar graph
plt.figure(figsize=(10, 6))
plt.bar(experience_salary['experience_level'], experience_salary['salary'], co
plt.xlabel('Experience Level')
plt.ylabel('Mean Salary')
plt.title('Mean Salary by Experience Level')
plt.xticks(rotation=45)
plt.tight_layout()

# Show the plot
plt.show()
```



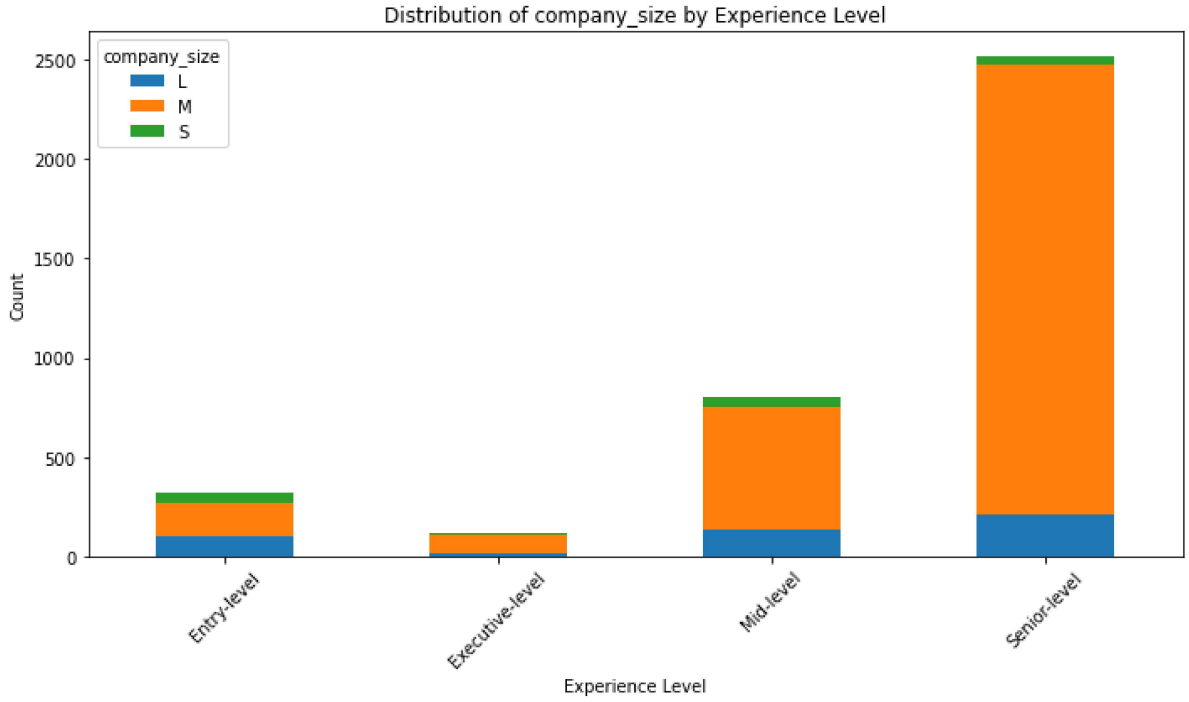
Show the Distribution of Employment Type by Experience Level

In [63]:
```
# Create a crosstab between experience_level and employment_type
crosstab = pd.crosstab(df['experience_level'], df['employment_type'])

# Create a grouped bar chart
crosstab.plot(kind='bar', stacked=True, figsize=(10, 6))
plt.xlabel('Experience Level')
plt.ylabel('Count')
plt.title('Distribution of Employment Type by Experience Level')
plt.xticks(rotation=45)
plt.legend(title='Employment Type', loc='upper right')
plt.tight_layout()

# Show the plot
plt.show()
```



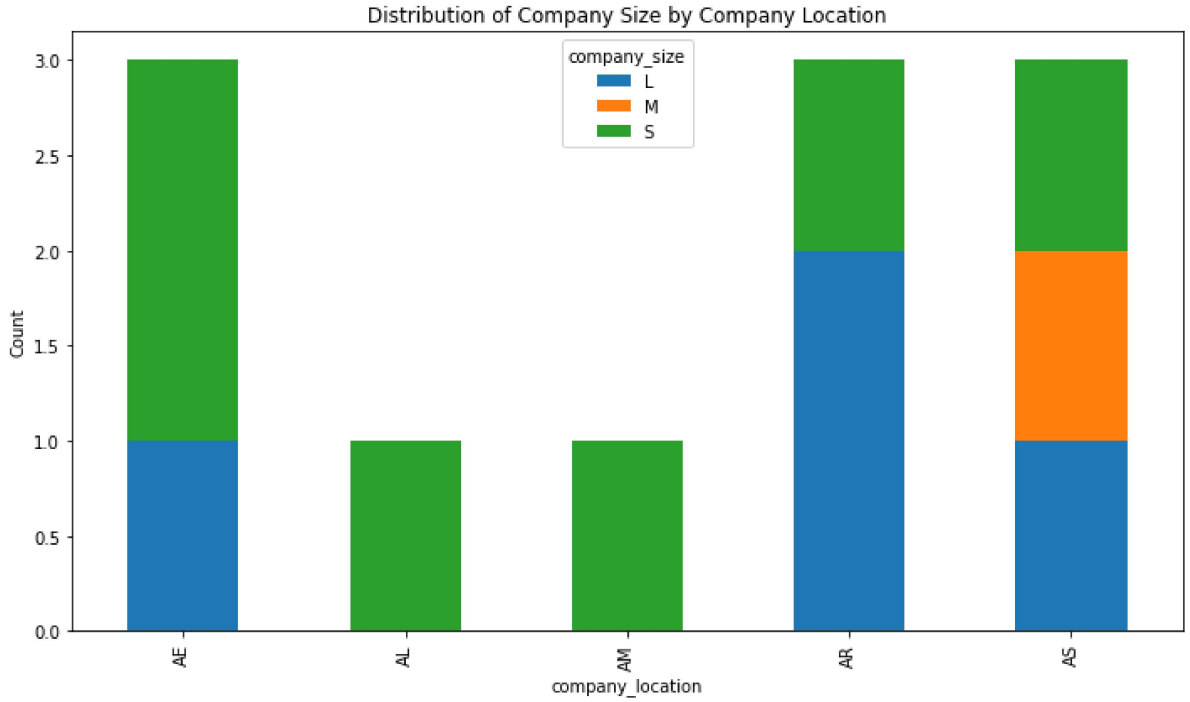
Show the Distribution of CompanySize by Experience Level

In [65]:
```
# Create a crosstab between experience_level and company_size
crosstab = pd.crosstab(df['experience_level'], df['company_size'])

# Create a grouped bar chart
crosstab.plot(kind='bar', stacked=True, figsize=(10, 6))
plt.xlabel('Experience Level')
plt.ylabel('Count')
plt.title('Distribution of company_size by Experience Level')
plt.xticks(rotation=45)
plt.tight_layout()

# Show the plot
plt.show()
```



Show the Distribution between experience_level and company_size

In [68]:
```
# Create a crosstab between experience_level and company_size
crosstab = pd.crosstab(df['company_location'], df['company_size']).head()

# Create a grouped bar chart
crosstab.plot(kind='bar', stacked=True, figsize=(10, 6))
plt.xlabel('company_location')
plt.ylabel('Count')
plt.title('Distribution of Company Size by Company Location')
plt.xticks(rotation=90)
plt.tight_layout()

# Show the plot
plt.show()
```



In [76]:
```
df.columns
```

Out[76]:
```
Index(['work_year', 'experience_level', 'employment_type', 'job_title',
       'salary', 'salary_currency', 'salary_in_usd', 'employee_residence',
       'remote_ratio', 'company_location', 'company_size', 'Experience',
       'Size'],
      dtype='object')
```

Count the occurrences of each currency

In [88]:
```
# Count the occurrences of each currency
currency_counts = df['salary_currency'].value_counts()

# Create a bar plot
plt.figure(figsize=(10, 6))
currency_counts.plot(kind='bar')
plt.title('Salary Currency Distribution')
plt.xlabel('Currency')
plt.ylabel('Count')
plt.xticks(rotation=45)  # Rotate x-axis labels for better readability
plt.tight_layout()

# Show the plot
plt.show()
```