Problem Statement

What is a competitive salary range for a full-time data scientist, and is there a difference in salary between U.S. and non-U.S. positions?

Alternative ways to ask the question

How does full-time data scientist salary vary by experience level and location?

How does full-time data scientist salary vary by company size and location?

Load and Prepare the Data

In [51]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns

import os
```

In [52]:
```
# Load the dataset from CSV
infile = "/Users/evebarr20/Documents/DSE_5002/Project1/project_1_data.csv"

# Read the CSV file into a DataFrame
# The column "Unnamed: 0" is treated as the index rather than a data column
ds_df = pd.read_csv(infile, index_col = "Unnamed: 0")

# Preview the first few rows of the dataset
ds_df.head()
```

Out[52]:

	work_year	experience_level	employment_type	job_title	salary	salary_currency
0	2020	MI	FT	Data Scientist	70000	EUR
1	2020	SE	FT	Machine Learning Scientist	260000	USD
2	2020	SE	FT	Big Data Engineer	85000	GBP
3	2020	MI	FT	Product Data Analyst	20000	USD
4	2020	SE	FT	Machine Learning Engineer	150000	USD

In [53]:

```
# shape of the data set
ds_df.shape
```

Out[53]: (607, 11)

In [54]:

```
# data types
ds_df.dtypes
```

Out[54]:

```
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
```

In [55]:

```
# convert appropriate variables to type Categorical
ds_df["work_year"] = pd.Categorical(ds_df.work_year)
ds_df["experience_level"] = pd.Categorical(ds_df.experience_level)
ds_df["employment_type"] = pd.Categorical(ds_df.employment_type)
ds_df["job_title"] = pd.Categorical(ds_df.job_title)
ds_df["salary_currency"] = pd.Categorical(ds_df.salary_currency)
ds_df["employee_residence"] = pd.Categorical(ds_df.employee_residence)
ds_df["remote_ratio"] = pd.Categorical(ds_df.remote_ratio)
ds_df["company_location"] = pd.Categorical(ds_df.company_location)
ds_df["company_size"] = pd.Categorical(ds_df.company_size)

ds_df.dtypes
```

Out[55]:
```
work_year             category
experience_level      category
employment_type       category
job_title             category
salary                   int64
salary_currency       category
salary_in_usd            int64
employee_residence    category
remote_ratio          category
company_location      category
company_size          category
dtype: object
```

In [56]:
```
# check for missing values
missing_counts = ds_df.isnull().sum()
print(missing_counts)
```

```
work_year             0
experience_level      0
employment_type       0
job_title             0
salary                0
salary_currency       0
salary_in_usd         0
employee_residence    0
remote_ratio          0
company_location      0
company_size          0
dtype: int64
```

Exploratory Data Analysis

Summary

In [57]:
```
# statistic summary for numeric columns
ds_df.describe()
```

Out[57]:

	salary	salary_in_usd
count	6.070000e+02	607.000000
mean	3.240001e+05	112297.869852
std	1.544357e+06	70957.259411
min	4.000000e+03	2859.000000
25%	7.000000e+04	62726.000000
50%	1.150000e+05	101570.000000
75%	1.650000e+05	150000.000000
max	3.040000e+07	600000.000000

In the salary column, the mean is larger than the median, indicating right skew. Also, the standard deviation is high, indicating a substantial difference between values, which makes sense because the salary variable mixes many currencies, thereby inflating the range. Raw salaries would not be ideal for cross-country comparisons.

In the salary_in_usd column, the mean is larger than the median, indicating right-skewness, but the mean and median are closer together than in the salary variable. The quartiles form a reasonable salary band, and the max is high but not absurdly higher relative to the mean, unlike the salary variable. Converting to USD reduces distortion and better represents actual pay differences, making it appropriate for comparison and recommendations.

In [58]:
```
# Get the list of unique years
ds_df["work_year"].unique()
```

Out[58]:
```
[2020, 2021, 2022]
Categories (3, int64): [2020, 2021, 2022]
```

In [59]:
```
# Count of each unique year
ds_df["work_year"].value_counts(normalize=True)
```

Out[59]:
```
work_year
2022    0.523888
2021    0.357496
2020    0.118616
Name: proportion, dtype: float64
```

There are three unique work years in the dataset, with approximately 52% of observations occurring in 2022.

In [60]:
```
# Get the list of unique experience_level
ds_df["experience_level"].unique()
```

Out[60]:
```
['MI', 'SE', 'EN', 'EX']
Categories (4, object): ['EN', 'EX', 'MI', 'SE']
```

In [61]:
```
# Count of each unique experience_level
ds_df["experience_level"].value_counts(normalize=True)
```

Out[61]:
```
experience_level
SE    0.461285
MI    0.350906
EN    0.144975
EX    0.042834
Name: proportion, dtype: float64
```

There are four unique experience levels in the dataset, with the majority of observations falling into intermediate/senior and junior/mid levels. Significantly fewer observations fall into the expert executive-level/director category.

In [62]:
```
# Get the list of unique employment_type
ds_df["employment_type"].unique()
```

Out[62]:
```
['FT', 'CT', 'PT', 'FL']
Categories (4, object): ['CT', 'FL', 'FT', 'PT']
```

In [63]:
```
# Count of each unique employment_type
ds_df["employment_type"].value_counts(normalize=True)
```

Out[63]:
```
employment_type
FT    0.968699
PT    0.016474
CT    0.008237
FL    0.006590
Name: proportion, dtype: float64
```

There are four unique employment types in the dataset; a large proportion of observations falls into the full-time category (approximately 96%), and the remaining 4% fall into the other categories.

In [64]:
```
# Get the list of unique job_title
ds_df["job_title"].unique()
```

Out[64]:
```
['Data Scientist', 'Machine Learning Scientist', 'Big Data Engineer', 'Prod
uct Data Analyst', 'Machine Learning Engineer', ..., 'ETL Developer', 'Head
of Machine Learning', 'NLP Engineer', 'Lead Machine Learning Engineer', 'Da
ta Analytics Lead']
Length: 50
Categories (50, object): ['3D Computer Vision Researcher', 'AI Scientist',
'Analytics Engineer', 'Applied Data Scientist', ..., 'Principal Data Scient
ist', 'Product Data Analyst', 'Research Scientist', 'Staff Data Scientist']
```

In [65]:
```
# Count of each unique job_title
ds_df["job_title"].value_counts(normalize=True)
```

Out[65]:
```
job_title
Data Scientist                                  0.235585
Data Engineer                                   0.217463
Data Analyst                                    0.159802
Machine Learning Engineer                       0.067545
Research Scientist                              0.026359
Data Science Manager                            0.019769
Data Architect                                  0.018122
Big Data Engineer                               0.013180
Machine Learning Scientist                      0.013180
Director of Data Science                        0.011532
AI Scientist                                    0.011532
Principal Data Scientist                        0.011532
Data Science Consultant                         0.011532
Data Analytics Manager                          0.011532
Computer Vision Engineer                        0.009885
BI Data Analyst                                 0.009885
ML Engineer                                     0.009885
Lead Data Engineer                              0.009885
Data Engineering Manager                        0.008237
Business Data Analyst                           0.008237
Applied Data Scientist                          0.008237
Head of Data                                    0.008237
Head of Data Science                            0.006590
Data Analytics Engineer                         0.006590
Applied Machine Learning Scientist              0.006590
Analytics Engineer                              0.006590
Machine Learning Developer                      0.004942
Machine Learning Infrastructure Engineer        0.004942
Lead Data Scientist                             0.004942
Lead Data Analyst                               0.004942
Data Science Engineer                           0.004942
Principal Data Engineer                         0.004942
Computer Vision Software Engineer               0.004942
Principal Data Analyst                          0.003295
Financial Data Analyst                          0.003295
ETL Developer                                   0.003295
Director of Data Engineering                    0.003295
Product Data Analyst                            0.003295
Cloud Data Engineer                             0.003295
NLP Engineer                                    0.001647
Marketing Data Analyst                          0.001647
3D Computer Vision Researcher                   0.001647
Machine Learning Manager                        0.001647
Lead Machine Learning Engineer                  0.001647
Head of Machine Learning                        0.001647
Finance Data Analyst                            0.001647
Data Specialist                                 0.001647
Data Analytics Lead                             0.001647
Big Data Architect                              0.001647
Staff Data Scientist                            0.001647
Name: proportion, dtype: float64
```

There are fifty unique job titles in the dataset. The range across the categories is relatively balanced. The most common job title is data scientist at about 23%, and the *least common job title is staff data scientist at 0.16%*

In [66]:
```
# Get the list of unique Salary currency
ds_df["salary_currency"].unique()
```

Out[66]:
```
['EUR', 'USD', 'GBP', 'HUF', 'INR', ..., 'CLP', 'BRL', 'TRY', 'AUD', 'CHF']
Length: 17
Categories (17, object): ['AUD', 'BRL', 'CAD', 'CHF', ..., 'PLN', 'SGD', 'T
RY', 'USD']
```

In [67]:
```
# Count of each unique Salary currency
ds_df["salary_currency"].value_counts(normalize=True)
```

Out[67]:
```
salary_currency
USD    0.655684
EUR    0.156507
GBP    0.072488
INR    0.044481
CAD    0.029654
JPY    0.004942
PLN    0.004942
TRY    0.004942
CNY    0.003295
DKK    0.003295
BRL    0.003295
HUF    0.003295
MXN    0.003295
SGD    0.003295
AUD    0.003295
CHF    0.001647
CLP    0.001647
Name: proportion, dtype: float64
```

There are seventeen unique salary currencies in the dataset, with approximately 65% of observations occurring in the United States and 15% in Europe.

In [68]:
```
# Get the list of unique employee residence
ds_df["employee_residence"].unique()
```

Out[68]:
```
['DE', 'JP', 'GB', 'HN', 'US', ..., 'EE', 'AU', 'BO', 'IE', 'CH']
Length: 57
Categories (57, object): ['AE', 'AR', 'AT', 'AU', ..., 'TR', 'UA', 'US', 'V
N']
```

In [69]:
```
# Count of each unique employee residence
ds_df["employee_residence"].value_counts(normalize=True)
```

Out[69]:
```
employee_residence
US    0.546952
GB    0.072488
IN    0.049423
CA    0.047776
DE    0.041186
FR    0.029654
ES    0.024712
GR    0.021417
JP    0.011532
PK    0.009885
BR    0.009885
PT    0.009885
NL    0.008237
IT    0.006590
PL    0.006590
RU    0.006590
TR    0.004942
AE    0.004942
VN    0.004942
AT    0.004942
AU    0.004942
BE    0.003295
SI    0.003295
MX    0.003295
RO    0.003295
SG    0.003295
NG    0.003295
HU    0.003295
DK    0.003295
TN    0.001647
CL    0.001647
RS    0.001647
UA    0.001647
BG    0.001647
PR    0.001647
BO    0.001647
CH    0.001647
PH    0.001647
NZ    0.001647
EE    0.001647
MY    0.001647
DZ    0.001647
MT    0.001647
MD    0.001647
LU    0.001647
KE    0.001647
CN    0.001647
JE    0.001647
CO    0.001647
IR    0.001647
AR    0.001647
CZ    0.001647
IE    0.001647
HR    0.001647
HN    0.001647
```

```
HK    0.001647
IQ    0.001647
Name: proportion, dtype: float64
```

There are fifty-seven unique employee residences in the dataset, with the majority of employees living in the United States, approximately 54%

In [70]:
```
# Get the list of unique remote ratio
ds_df["remote_ratio"].unique()
```

Out[70]:
```
[0, 50, 100]
Categories (3, int64): [0, 50, 100]
```

In [71]:
```
# Count of each unique remote ratio
ds_df["remote_ratio"].value_counts(normalize=True)
```

Out[71]:
```
remote_ratio
100    0.627677
0      0.209226
50     0.163097
Name: proportion, dtype: float64
```

There are three unique remote ratios in the dataset. The range across the categories is relatively balanced, and a large proportion of employees get more than 80% of their work done remotely.

In [72]:
```
# Get the list of unique company location
ds_df["company_location"].unique()
```

Out[72]:
```
['DE', 'JP', 'GB', 'HN', 'US', ..., 'DZ', 'EE', 'MY', 'AU', 'IE']
Length: 50
Categories (50, object): ['AE', 'AS', 'AT', 'AU', ..., 'TR', 'UA', 'US', 'V
N']
```

In [73]:
```
# Count of each unique company location
ds_df["company_location"].value_counts(normalize=True)
```

Out[73]:
```
company_location
US    0.584843
GB    0.077430
CA    0.049423
DE    0.046129
IN    0.039539
FR    0.024712
ES    0.023064
GR    0.018122
JP    0.009885
PL    0.006590
PT    0.006590
NL    0.006590
AT    0.006590
MX    0.004942
LU    0.004942
TR    0.004942
PK    0.004942
AE    0.004942
AU    0.004942
BR    0.004942
DK    0.004942
CN    0.003295
CZ    0.003295
BE    0.003295
SI    0.003295
RU    0.003295
NG    0.003295
IT    0.003295
CH    0.003295
NZ    0.001647
CL    0.001647
EE    0.001647
SG    0.001647
UA    0.001647
RO    0.001647
CO    0.001647
MY    0.001647
DZ    0.001647
MT    0.001647
MD    0.001647
KE    0.001647
IR    0.001647
IQ    0.001647
AS    0.001647
IL    0.001647
IE    0.001647
HU    0.001647
HR    0.001647
HN    0.001647
VN    0.001647
Name: proportion, dtype: float64
```

There are fifty unique company locations in the dataset, with the majority of companies located in the United States at about 58%.

In [74]:
```
# Get the list of unique company size
ds_df["company_size"].unique()
```

Out[74]:
```
['L', 'S', 'M']
Categories (3, object): ['L', 'M', 'S']
```

In [75]:
```
# Count of each unique company size
ds_df["company_size"].value_counts(normalize=True)
```

Out[75]:
```
company_size
M    0.537068
L    0.326194
S    0.136738
Name: proportion, dtype: float64
```

There are three distinct company sizes in the dataset, with the majority of observations falling into the medium and large categories. Significantly fewer observations fall into the small category.

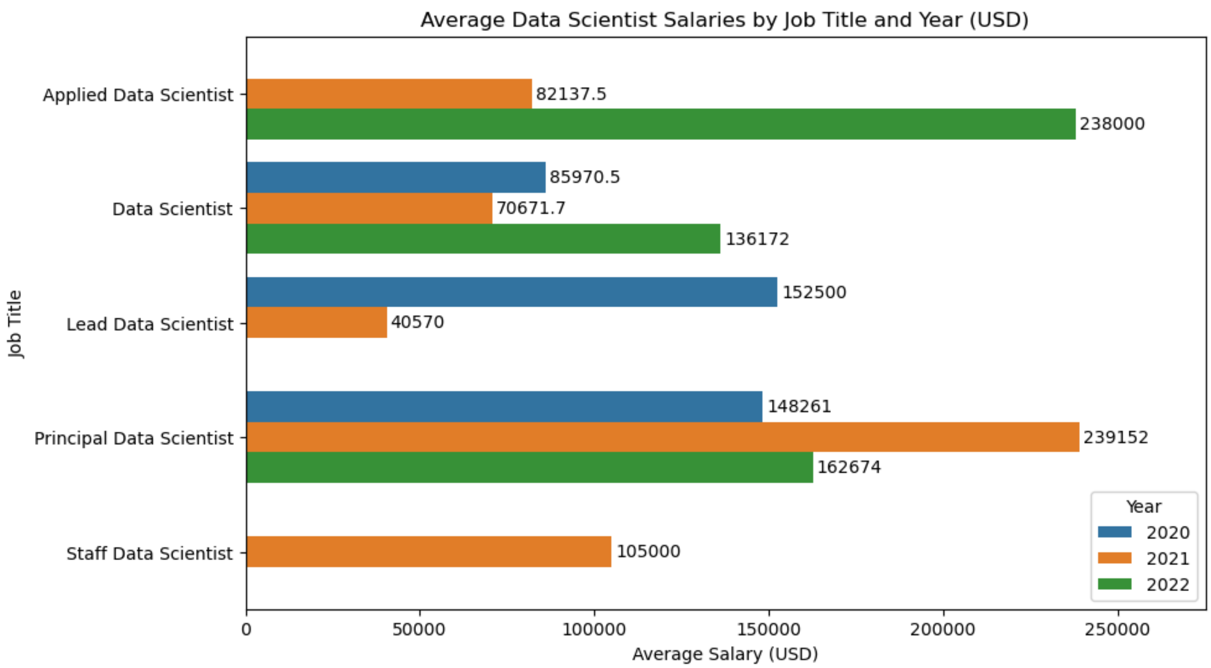
Plots

In [76]:
```
# --------------------------------
# Prepare data for salary trend plots
# --------------------------------

# Create a copy of the original dataset to avoid modifying the raw data
ds_roles_df = ds_df.copy()

# Filter the dataset to include only job titles containing "Data Scientist"
# (e.g., Data Scientist, Applied Data Scientist, Lead Data Scientist, etc.)
ds_roles_df = ds_roles_df[ds_roles_df["job_title"].str.contains("Data Scient

# Remove unused job title categories after filtering
# This prevents plotting categories with no data
ds_roles_df["job_title"] = ds_roles_df["job_title"].cat.remove_unused_catego

# Calculate the average salary (USD) by work year and job title
# observed = True ensures only existing category combinations are included
avg_salary =(ds_roles_df
    .groupby(["work_year", "job_title"],observed=True)["salary_in_usd"]
    .mean()
    .reset_index()
             )
```

In [77]:
```
# --------------------------------------
# Plot 1: Average Salary by job title and year
# --------------------------------------

# Create a wider figure for better readability of horizontal bars
plt.figure(figsize = (10, 6))

# Bar plot showing average salary (USD) by job title,
# with separate bars for each work year
```

```
ax = sns.barplot(
    data = avg_salary,
    x = "salary_in_usd",
    y = "job_title",
    hue = "work_year",
    orient = "y",
    errorbar=None
)

# Add numeric salary labels to each bar
for container in ax.containers:
    ax.bar_label(container, fontsize = 10, padding = 2)

# Add extra horizontal space to value labels are not cut off
ax.margins(x = 0.15)

# Add titles and axis labels
plt.title("Average Data Scientist Salaries by Job Title and Year (USD)")
plt.xlabel("Average Salary (USD)")
plt.ylabel("Job Title")

# Add legend title for clarity
plt.legend(title = "Year")
```

Out[77]: <matplotlib.legend.Legend at 0x1675342d0>



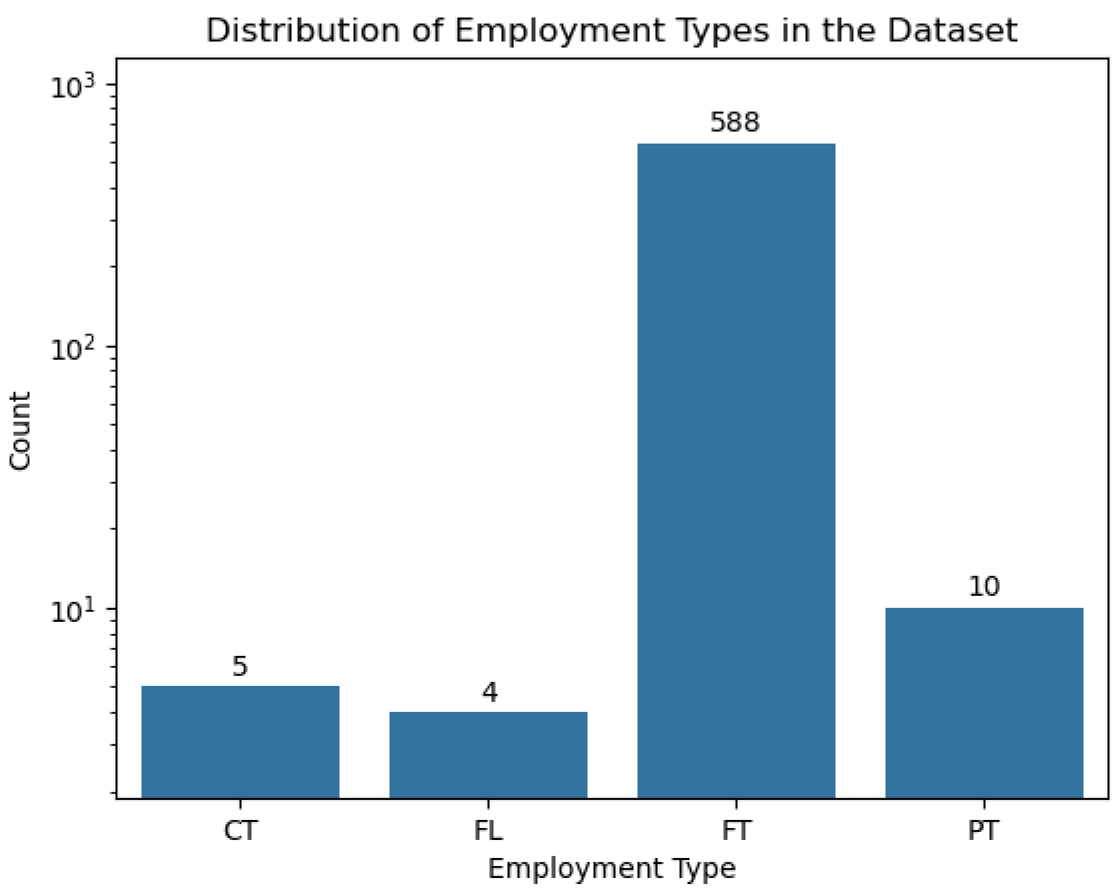
Average salaries vary by both job title and year. Applied Data Scientist shows the highest average in 2022, while Principal Data Scientist has the highest average in 2021. Some roles increase over time while others decrease, showing that salary trends are not consistent across job titles.

In [78]:

```
# ---------------------------------------
# Plot 2: Distribution of Employment Types
# ---------------------------------------
```

```python
# Bar plot showing the distribution of employment types in the dataset
ax = sns.countplot(ds_df, x = "employment_type")

# Use a log scale to better visualize categories with very different counts
ax.set_yscale("log")

# Add numeric labels to each bar
for container in ax.containers:
    ax.bar_label(container, fontsize = 10, padding = 2)

# Add extra vertical space so bar labels are not clipped at the top
ax.margins(y = 0.15)

# Add plot title and axis labels
plt.title("Distribution of Employment Types in the Dataset")
plt.xlabel("Employment Type")
plt.ylabel("Count")
```

Out[78]: `Text(0, 0.5, 'Count')`



FT(Full-time) positions dominate the dataset, while PT(Part-time), CT(Contract), and FL(Freelance) roles are relatively rare

In [79]:
```python
# ---------------------------------------
# Plot 3: Salary Distribution by Experience Level
# ---------------------------------------
```

```
# Box plot showing the distribution of salaries across experience levels
sns.boxplot(ds_df, x = "salary_in_usd", y = "experience_level")

# Add plot title and axis labels
plt.title("Salary Distribution by Experience Level (USD)")
plt.xlabel("Salary in USD")
plt.ylabel("Experience Level")
```

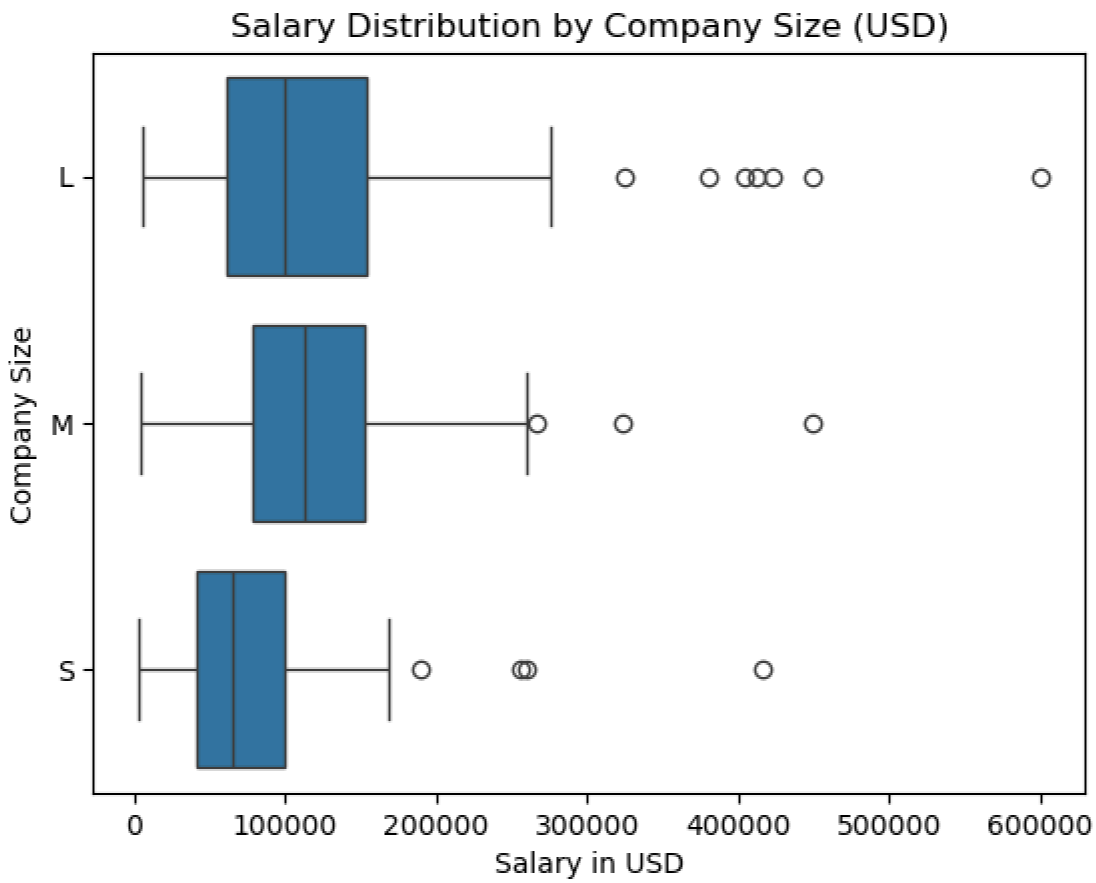
Out[79]: Text(0, 0.5, 'Experience Level')

This boxplot shows that expert-level roles have the highest median salary and the highest extreme high-salary outliers. Entry-level positions have the lowest median salaries. Across all experience levels, the salary distributions are right-skewed, indicating the presence of higher-end outliers. Overall, the plot suggests a clear positive relationship between experience level and salary, with higher experience associated with higher pay.

In [80]:
```
# ---------------------------------------
# Plot 4: Salary Distribution by Company Size
# ---------------------------------------

# Box plot showing the distribution of salaries across company sizes
sns.boxplot(ds_df, x = "salary_in_usd", y = "company_size")

# Add plot title and axis labels
plt.title("Salary Distribution by Company Size (USD)")
```

```
plt.xlabel("Salary in USD")
plt.ylabel("Company Size")
```

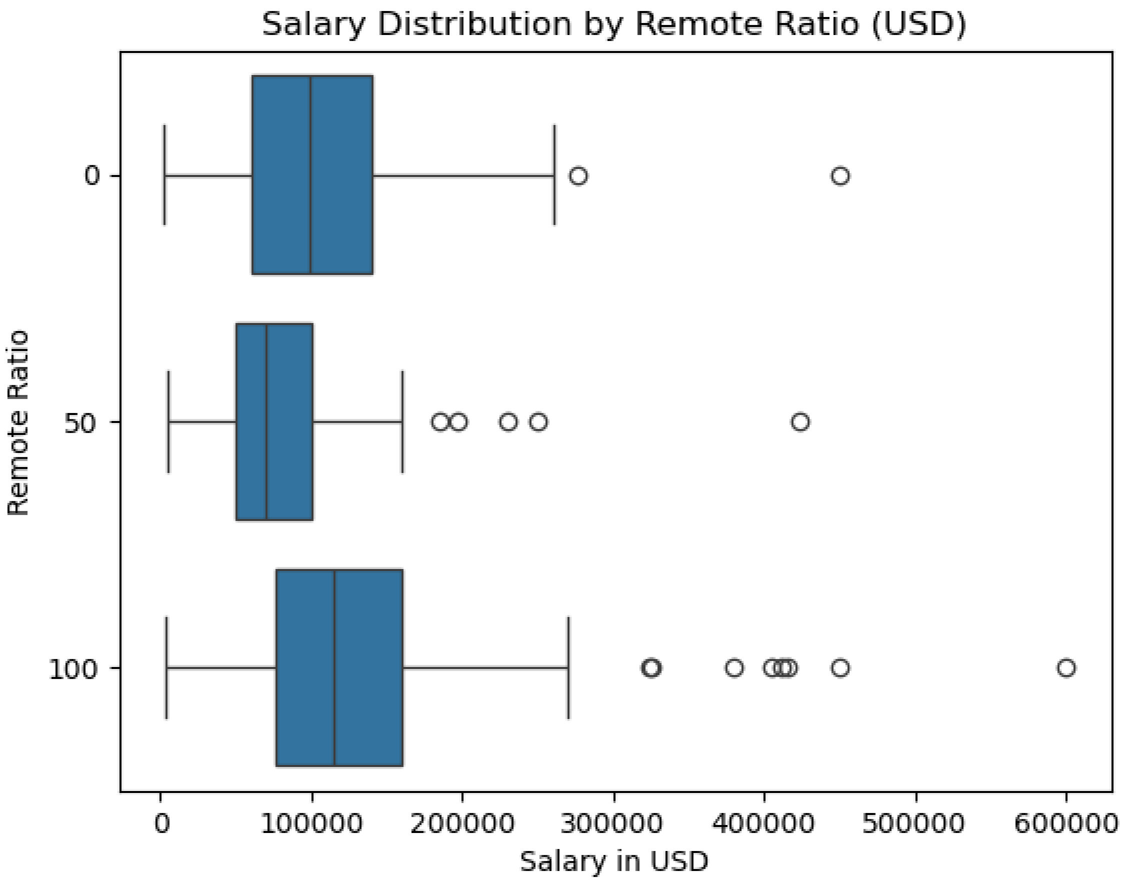
Out[80]: Text(0, 0.5, 'Company Size')



This boxplot shows that medium-sized companies have a slightly higher median salary than large companies. Small companies have the lowest median salaries and the narrowest distribution. All company sizes display right-skewed salary distributions, with larger companies showing more extreme high-salary outliers. Overall, company size appears to influence salary levels.

In [81]:
```
# --------------------------------------
# Plot 5: Salary Distribution by Remote Ratio
# --------------------------------------

# Box plot showing the distribution of salaries across remote ratio
# Note: Remote ratio indicates the percentage of time an employee works remo
sns.boxplot(ds_df, x = "salary_in_usd", y = "remote_ratio")

# Add plot title and axis labels
plt.title("Salary Distribution by Remote Ratio (USD)")
plt.xlabel("Salary in USD")
plt.ylabel("Remote Ratio")
```

Out[81]: Text(0, 0.5, 'Remote Ratio')



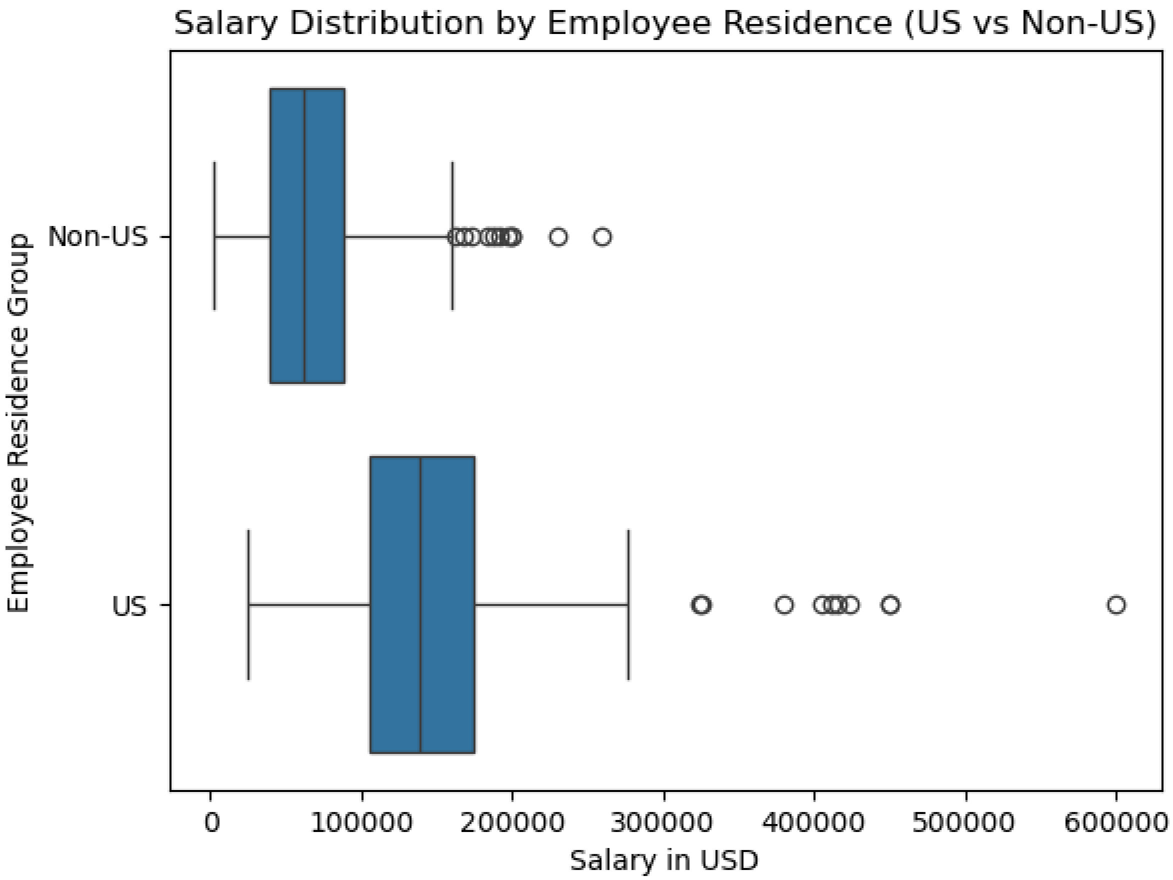
This boxplot shows that fully remote roles (100%) have the highest median salary, followed by on-site roles (0%), while hybrid roles (50%) have the lowest median salary. All three remote categories exhibit right-skewed salary distributions. Fully remote positions exhibit the greatest variability and the most extreme high-end outliers.

In [82]:
```
# Create a copy of the original dataset so we don't modify ds_df
salary_by_residence_df = ds_df.copy()

# Create a simple grouping variable for employee residence:
# - "US" if the employee resides in the United States
# - "Non-US" otherwise
salary_by_residence_df["residence_group"] = np.where(
    salary_by_residence_df["employee_residence"] == "US",
    "US",
    "Non-US"
)
```

In [83]:
```
# -------------------------------------
# Plot 6: Salary Distribution by Employee Residence (US vs Non-US)
# -------------------------------------

# Boxplot comparing salary distributions for US vs Non-US employee residence
sns.boxplot(data = salary_by_residence_df, x = "salary_in_usd", y = "resider

# Add plot title and axis labels
plt.title("Salary Distribution by Employee Residence (US vs Non-US)")
```

```
plt.xlabel("Salary in USD")
plt.ylabel("Employee Residence Group")
```

Out[83]: Text(0, 0.5, 'Employee Residence Group')



US-based employees have higher median salaries and more extreme high-end outliers than employees residing outside the US. The salary distribution among US employees is also wider, indicating greater compensation variability. Overall, this suggests that employees living in the US tend to earn higher salaries than those living in non-US locations

In [84]:
```
# Create a copy of the original dataset so we don't modify ds_df
salary_by_location_df = ds_df.copy()

# Create a simple grouping variable for company location:
# - "US" if the company resides in the United States
# - "Non-US" otherwise
salary_by_location_df["location_group"] = np.where(
    salary_by_location_df["company_location"] == "US",
    "US",
    "Non-US"
)
```

In [85]:
```
# ---------------------------------------
# Plot 7: Salary Distribution by Company Location (US vs Non-US)
# ---------------------------------------
```

```
# Boxplot comparing salary distributions for US vs Non-US company locations
sns.boxplot(data = salary_by_location_df, x = "salary_in_usd", y = "locatior

# Add plot title and axis labels
plt.title("Salary Distribution by Company Location (US vs Non-US)")
plt.xlabel("Salary in USD")
plt.ylabel("Company Location Group")
```

Out[85]: Text(0, 0.5, 'Company Location Group')

The salary distribution by company location (US vs Non-US) closely mirrors the distribution by employee residence. This suggests that, in this dataset, company location and employee residence are highly correlated, leading to similar salary patterns across both variables.

In [86]:
```
# Count how many times each salary currency appears in the dataset
# value_counts() returns the frequency of each unique currency
salary_currency_counts_df = (
    ds_df["salary_currency"]
    .value_counts()
    .reset_index()
)

# Keep only the top 5 most common salary currencies
salary_currency_counts_df = salary_currency_counts_df.head(5)

# Remove unused category levels so only the top currencies appear in plots
salary_currency_counts_df["salary_currency"] = (
```

```
    salary_currency_counts_df["salary_currency"].cat.remove_unused_categorie
)

salary_currency_counts_df
```

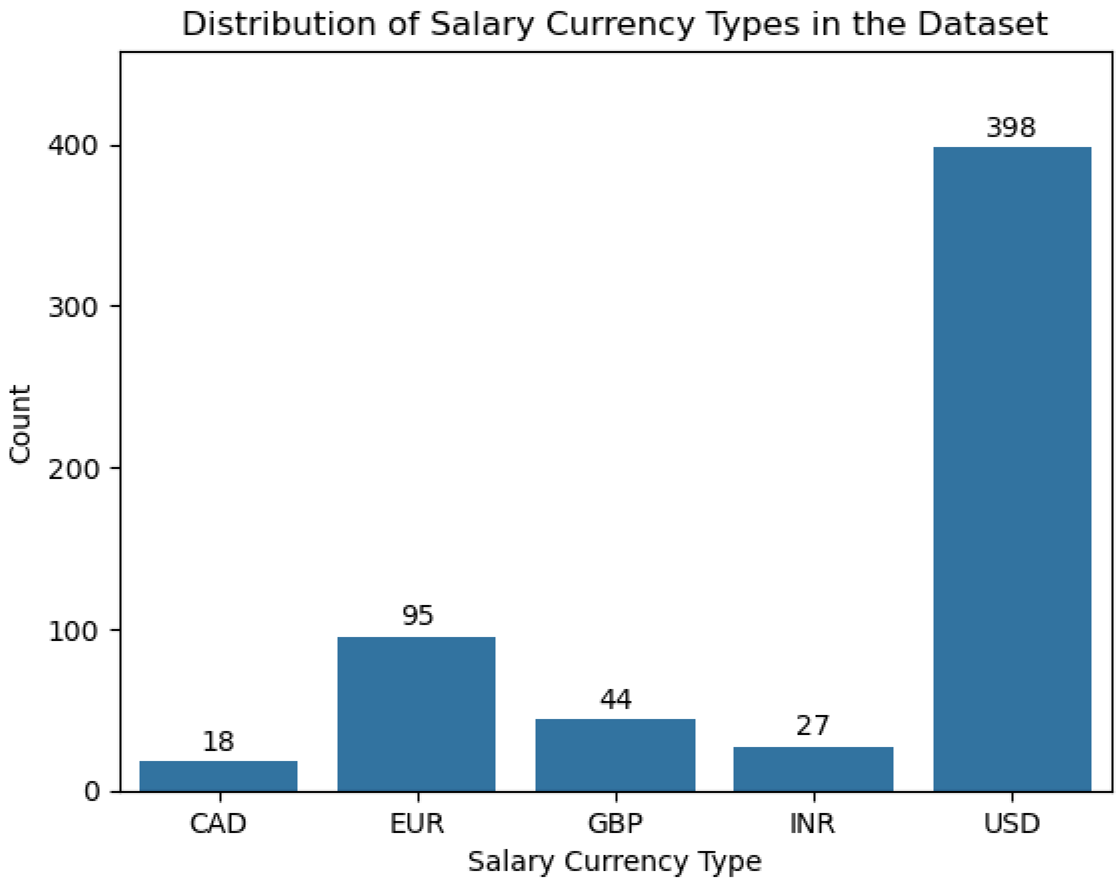
Out[86]:

	salary_currency	count
0	USD	398
1	EUR	95
2	GBP	44
3	INR	27
4	CAD	18

In [87]:

```
# ---------------------------------------
# Plot 8: Distribution of Salary Currency Types (Top 5)
# ---------------------------------------

# Create a bar plot showing how frequently each salary currency appears
currency_bar_ax = sns.barplot(
    data = salary_currency_counts_df,
    x = "salary_currency",
    y = "count"
)


# Add numeric labels on top of each bar
for container in currency_bar_ax.containers:
    currency_bar_ax.bar_label(container, fontsize = 10, padding = 2)

# Add extra vertical space so value labels are not cut off
currency_bar_ax.margins(y = 0.15)

# Add plot title and axis labels
plt.title("Distribution of Salary Currency Types in the Dataset")
plt.xlabel("Salary Currency Type")
plt.ylabel("Count")
```

Out[87]: `Text(0, 0.5, 'Count')`

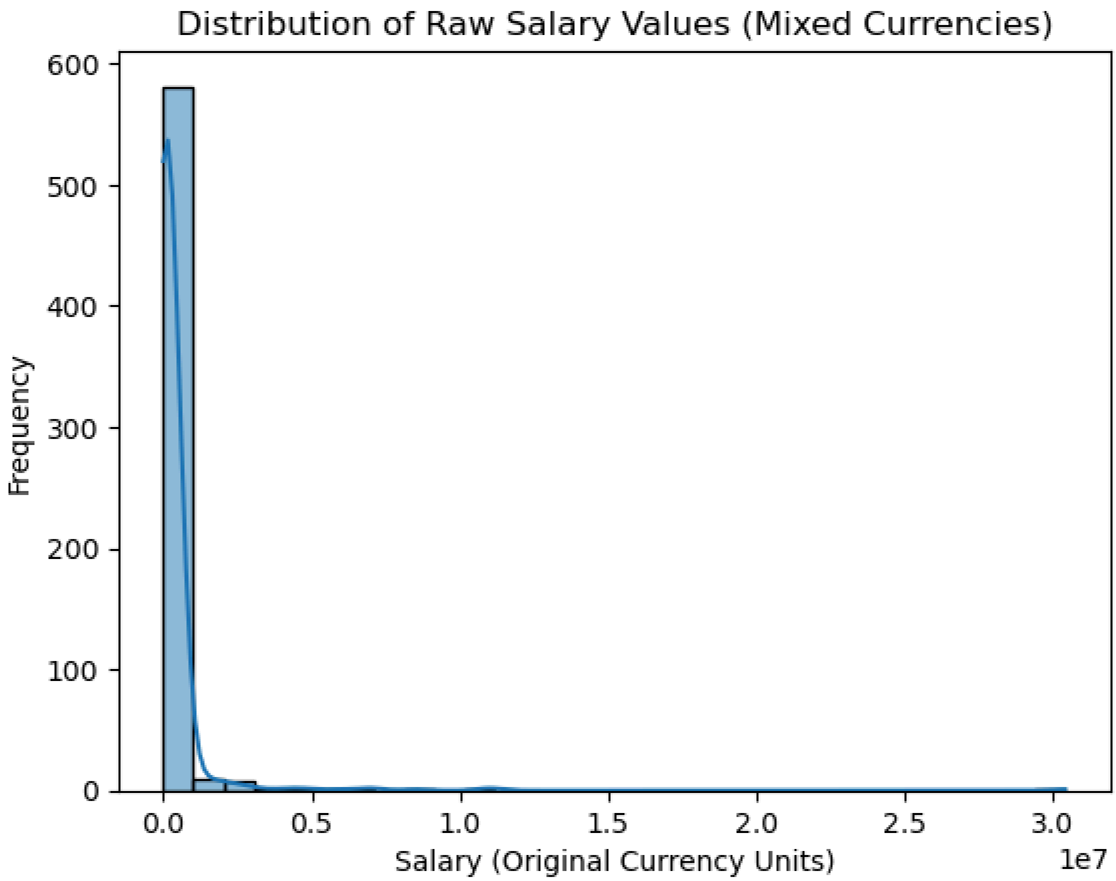


The majority of salary records in the dataset are reported in USD, accounting for 398 observations. Other currencies such as EUR (95), GBP (44), INR (27), and CAD (18) appear far less frequently. This indicates that the dataset is heavily U.S.-centric, which helps explain why converting salaries to USD is necessary for meaningful comparisons across countries.

In [88]:

```
# --------------------------------------
# Plot 9: Distribution of Raw Salary Values
# --------------------------------------
# This histogram visualizes the raw salary values as they appear in the data
# Salaries are shown in their original currencies (not converted to USD).
# kde=True adds a smooth density curve to show the overall shape of the sala
sns.histplot(ds_df["salary"], bins=30, kde=True)

# Add plot title and axis labels
plt.title("Distribution of Raw Salary Values (Mixed Currencies)")
plt.xlabel("Salary (Original Currency Units)")
plt.ylabel("Frequency")
```

Out[88]: `Text(0, 0.5, 'Frequency')`



The distribution of raw salary values is highly skewed and difficult to interpret due to salaries being reported in multiple currencies. Large numeric values do not necessarily indicate higher pay, as they may reflect differences in currency rather than compensation. This highlights the importance of using salary_in_usd for cross-country salary comparisons.

Answer CEO Question

Table 1

In [89]:
```
# ------------------------------------------------------------
# Prepare data for Plot 1:
# Average full-time Data Scientist salaries by
# job title, experience level, and company size
# ------------------------------------------------------------

# Create a copy of the original dataset to avoid modifying it
filtered_ds_df = ds_df.copy()

# Keep only job titles that include "Data Scientist"
filtered_ds_df = filtered_ds_df[
    filtered_ds_df ["job_title"].str.contains("Data Scientist")
]
```

```
# Keep only full-time employees
filtered_ds_df = filtered_ds_df[
    filtered_ds_df["employment_type"] == "FT"
]

# Calculate the average salary (USD) grouped by:
# job title, experience level, and company size
avg_salary_by_role = (
    filtered_ds_df
    .groupby(["job_title", "experience_level", "company_size"],
             observed = True)
        ["salary_in_usd"]
    .mean()
    .reset_index()
)
# Remove unused categories to prevent empty bars/labels in plots
for col in ["job_title", "experience_level", "company_size"]:
    avg_salary_by_role[col] = avg_salary_by_role[col].cat.remove_unused_cate

avg_salary_by_role
```

Out[89]:

	job_title	experience_level	company_size	salary_in_usd
0	Applied Data Scientist	EN	L	110037.000000
1	Applied Data Scientist	MI	L	105619.000000
2	Applied Data Scientist	SE	L	278500.000000
3	Data Scientist	EN	L	38365.000000
4	Data Scientist	EN	M	50888.250000
5	Data Scientist	EN	S	76385.833333
6	Data Scientist	MI	L	85501.260870
7	Data Scientist	MI	M	101014.041667
8	Data Scientist	MI	S	35956.833333
9	Data Scientist	SE	L	153273.875000
10	Data Scientist	SE	M	155811.046512
11	Data Scientist	SE	S	89487.500000
12	Lead Data Scientist	MI	L	115000.000000
13	Lead Data Scientist	SE	L	40570.000000
14	Lead Data Scientist	SE	S	190000.000000
15	Principal Data Scientist	MI	L	151000.000000
16	Principal Data Scientist	SE	L	227500.000000
17	Principal Data Scientist	SE	M	161565.666667

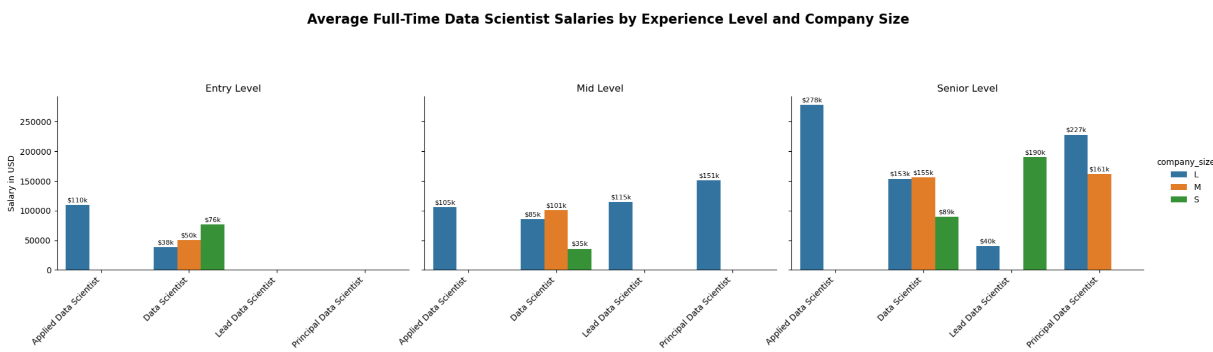
This table summarizes the average salary (USD) for full-time Data Scientist roles by job title, experience level, and company size. It provides a baseline for understanding what salary ranges may be considered competitive across different experience levels and organizational sizes.

In [46]:
```
# export table as a csv for my presentation
avg_salary_by_role.to_csv("avg_salary_by_role.csv", index=False)
```

Plot 1

In [90]:
```
# Map experience level codes to readable labels
experience_level_labels = {
    "EN": "Entry Level",
    "MI": "Mid Level",
    "SE": "Senior Level",
}

# ----------------------------------------------------------
# Create faceted bart chat
# - Average salary by job title
# - Faceted by experience level
# - Colored by company size
# -----------------------------------------------------------
salary_facet_grid = sns.catplot(
    data = avg_salary_by_role,
    kind = "bar",
    x = "job_title",
    y = "salary_in_usd",
    col = "experience_level",
    hue = "company_size",
    aspect = 1.3, #  ?
)

# Rotate job title labels for readability
salary_facet_grid.set_xticklabels(rotation = 45, ha = "right")

#-------------------------------------------------------------
# Add value labels to each bar
# rename facet titles using the readable experience labels
#-------------------------------------------------------------
for axis in salary_facet_grid.axes.flat:
    for container in axis.containers:
        labels = [
            f"${int(v/1000)}k" if v > 0 else ""
            for v in container.datavalues
        ]
        axis.bar_label(
            container,
            labels=labels,
            fontsize=8,
            padding=2
        )
```

```
    # Replace facet titles like "experience_level = EN"
    # with readable titles like "Entry Level"
    title = axis.get_title()
    experience_code = title.split(" = ")[-1]
    axis.set_title(experience_level_labels.get(experience_code))

# Add y-axis label
salary_facet_grid.set_axis_labels("", "Salary in USD")

# Add an overall title for the entire figure

salary_facet_grid.fig.suptitle(
    "Average Full-Time Data Scientist Salaries by Experience Level and Compa
    fontsize = 16,
    fontweight = "bold"
)

# Adjust spacing so the title doesn't overlap the facets
salary_facet_grid.fig.subplots_adjust(top=0.70)
```



This plot shows average full-time data scientist salaries by job title, experience level, and company size. The Data Scientist role appears consistently across experience levels and company sizes, making it the most reliable basis for identifying a competitive salary range. More senior titles (Lead and Principal Data Scientist) primarily appear at higher experience levels, which explains the absence of entry-level values for those roles.

In [48]:
```
# save plot as an image
salary_facet_grid.savefig("plot1.png", dpi=300, bbox_inches="tight")
```

Table 2

In [91]:
```
# create copy of original dataset
ds_location_analysis = ds_df.copy()

# Filter to Data Scientist-related job titles only
ds_location_analysis = ds_location_analysis[
    ds_location_analysis['job_title'].str.contains('Data Scientist')
]

# Filter to full-time positions only
ds_location_analysis = ds_location_analysis[
    ds_location_analysis['employment_type'] == 'FT'
```

```
]

# ------------------------------------------------------------
# Create a simplified location group based on employee residence
# - US: employee resides in the United States
# - Non-US: employee resides outside the United States
# ------------------------------------------------------------
ds_location_analysis["location_group"] = np.where(
    ds_location_analysis["employee_residence"] == "US",
    "US",
    "Non-US"
)

# ------------------------------------------------------------
# Calculate average salary (USD) by job title and residence group
# ------------------------------------------------------------
avg_salary_by_residence = (ds_location_analysis
    .groupby(["job_title", "location_group"], observed = True)["salary_in_us
    .mean()
    .reset_index())

# Remove unused job title categories after filtering
ds_location_analysis["job_title"] = (
    ds_location_analysis["job_title"].cat.remove_unused_categories()
)

avg_salary_by_residence
```

Out[91]:

	job_title	location_group	salary_in_usd
0	Applied Data Scientist	Non-US	82137.500000
1	Applied Data Scientist	US	238000.000000
2	Data Scientist	Non-US	57989.370968
3	Data Scientist	US	149408.333333
4	Lead Data Scientist	Non-US	77785.000000
5	Lead Data Scientist	US	190000.000000
6	Principal Data Scientist	Non-US	161565.666667
7	Principal Data Scientist	US	202000.000000

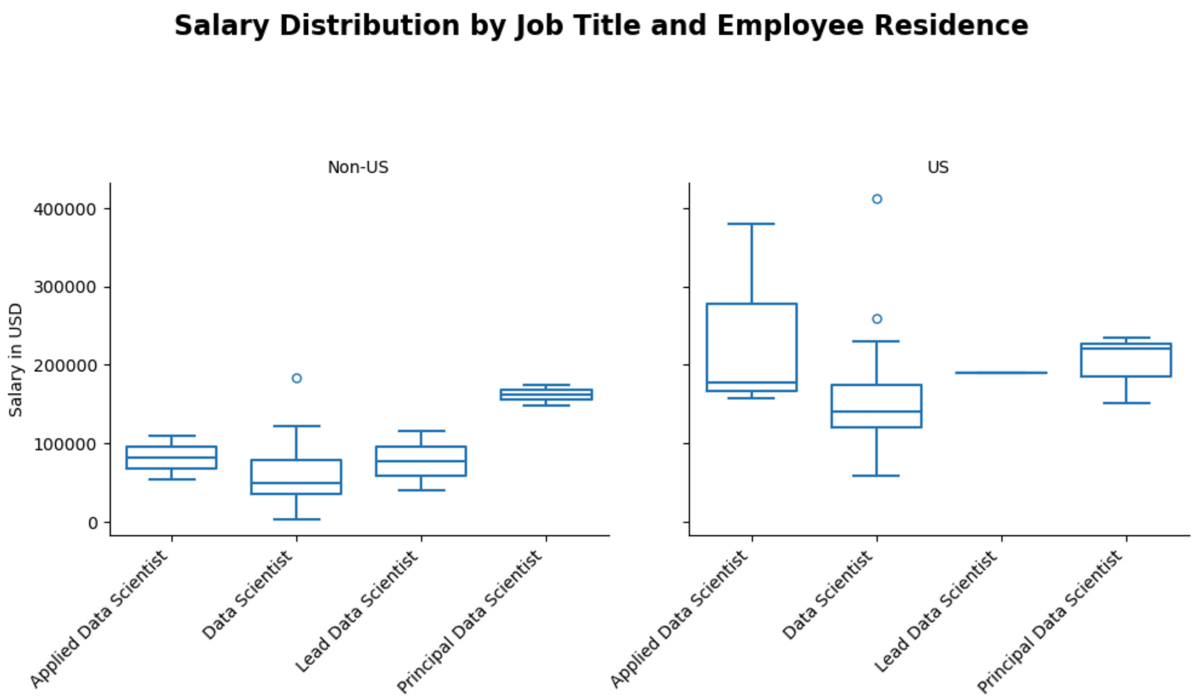
This table shows the average salary (USD) for full-time Data Scientist roles, grouped by job title and company location (US vs Non-US). Across all job titles, average salaries are higher for U.S.-based positions compared to non-U.S. positions

In [50]:

```
# export table as a csv for my presentation
avg_salary_by_residence.to_csv("avg_salary_by_residence.csv", index=False)
```

Plot 2

In [92]:
```
# -----------------------------------------------------------
# Plot: Salary distribution by job title and employee residence
# (US vs Non-US) using boxplots
# -----------------------------------------------------------

salary_by_residence_plot = sns.catplot(
    data = ds_location_analysis,
    x = "job_title",
    y = "salary_in_usd",
    col = "location_group", # Separate plots for US vs Non-US
    kind = "box",
    col_wrap = 2, # Arrange facets in two columns
    fill = False, # Outline-only boxes for clarity
    gap = .1      # Small spacing between categories
)

# Rotate job title labels for readability
salary_by_residence_plot.set_xticklabels(rotation = 45, ha = "right")

# Set axis labels and facet titles
salary_by_residence_plot.set_axis_labels("", "Salary in USD")
salary_by_residence_plot.set_titles("{col_name}")

# Add an overall title above all facets
salary_by_residence_plot.fig.suptitle(
    "Salary Distribution by Job Title and Employee Residence",
    fontsize=16,
    fontweight="bold"
)

# Adjust spacing so the title doesn't overlap the plots
salary_by_residence_plot.fig.subplots_adjust(top=0.70)
```



Across all data scientist job titles, employees residing in the U.S. earn higher salaries than those working outside the U.S. U.S.-based roles show higher median salaries, along with greater variability and higher upper-end values. Non-U.S. salaries tend to be lower and more concentrated

In [ ]:
```
# save plot as an image
salary_by_residence_plot.savefig("plot2.png", dpi=300, bbox_inches="tight")
```

Summary

This analysis examined what constitutes a competitive salary range for full-time data scientists and whether compensation differs between U.S. and non-U.S. positions. The results show that competitive salaries vary substantially based on experience level, company size, job title, and location. For core Data Scientist roles, wages in the dataset generally range from approximately `$35,000` to `$155,000`, with higher experience levels associated with higher pay. Senior-level and specialized roles, such as Applied or Principal Data Scientist positions, can command significantly higher salaries, reaching up to $278,000 in some cases, particularly at larger companies. Across all job titles, U.S.-based positions have higher median compensation than non-U.S. roles

In [ ]:
```
```