Data Science job postings on Glassdoor - EDA

GOAL

Conduct comprehensive data cleaning and exploratory data analysis (EDA) to enhance the quality and understand the inherent patterns within the dataset, facilitating informed decision-making and future analysis.

PROJECT DURATION

Project duration varies between 2 and 3 days. In order to carry out the project as quickly as possible, it is important that the relevant data is available, complete and clean.

Importing Libraries

In [6]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns

#2 lines below for html export
import plotly.io as pio
pio.renderers.default = 'notebook'

# 2 lines below for PDF export
!pip install Pyppeteer
!pyppeteer-install
```

```
Requirement already satisfied: Pyppeteer in c:\users\shubh\anaconda3\lib\site-packages
(1.0.2)
Requirement already satisfied: pyee<9.0.0,>=8.1.0 in c:\users\shubh\anaconda3\lib\site-p
ackages (from Pyppeteer) (8.2.2)
Requirement already satisfied: websockets<11.0,>=10.0 in c:\users\shubh\anaconda3\lib\si
te-packages (from Pyppeteer) (10.4)
Requirement already satisfied: appdirs<2.0.0,>=1.4.3 in c:\users\shubh\anaconda3\lib\sit
e-packages (from Pyppeteer) (1.4.4)
Requirement already satisfied: tqdm<5.0.0,>=4.42.1 in c:\users\shubh\anaconda3\lib\site-
packages (from Pyppeteer) (4.64.1)
Requirement already satisfied: urllib3<2.0.0,>=1.25.8 in c:\users\shubh\anaconda3\lib\si
te-packages (from Pyppeteer) (1.26.14)
Requirement already satisfied: certifi>=2021 in c:\users\shubh\anaconda3\lib\site-packag
es (from Pyppeteer) (2022.12.7)
Requirement already satisfied: importlib-metadata>=1.4 in c:\users\shubh\anaconda3\lib\s
ite-packages (from Pyppeteer) (4.11.3)
Requirement already satisfied: zipp>=0.5 in c:\users\shubh\anaconda3\lib\site-packages
(from importlib-metadata>=1.4->Pyppeteer) (3.11.0)
Requirement already satisfied: colorama in c:\users\shubh\anaconda3\lib\site-packages (f
rom tqdm<5.0.0,>=4.42.1->Pyppeteer) (0.4.6)
chromium is already installed.
```

Reading the file

In [2]:
```
df = pd.read_csv('Uncleaned_DS_jobs.csv')
```

In [3]:
```
df.head(3)
```

Out[3]:

	index	Job Title	Salary Estimate	Job Description	Rating	Company Name	Location	Headquarters	Size
0	0	Sr Data Scientist	$137K-171K (Glassdoor est.)	Description\n\nThe Senior Data Scientist is re...	3.1	Healthfirst\n3.1	New York, NY	New York, NY	1001 to 5000 employees
1	1	Data Scientist	$137K-171K (Glassdoor est.)	Secure our Nation, Ignite your Future\n\nJoin ...	4.2	ManTech\n4.2	Chantilly, VA	Herndon, VA	5001 to 10000 employees
2	2	Data Scientist	$137K-171K (Glassdoor est.)	Overview\n\n\nAnalysis Group is one of the lar...	3.8	Analysis Group\n3.8	Boston, MA	Boston, MA	1001 to 5000 employees

In [4]:
```
df.info()
#provides a concise summary of the DataFrame.
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 672 entries, 0 to 671
Data columns (total 15 columns):
 #   Column             Non-Null Count  Dtype  
---  ------             --------------  -----  
 0   index              672 non-null    int64  
 1   Job Title          672 non-null    object 
 2   Salary Estimate    672 non-null    object 
 3   Job Description    672 non-null    object 
 4   Rating             672 non-null    float64
 5   Company Name       672 non-null    object 
 6   Location           672 non-null    object 
 7   Headquarters       672 non-null    object 
 8   Size               672 non-null    object 
 9   Founded            672 non-null    int64  
 10  Type of ownership  672 non-null    object 
 11  Industry           672 non-null    object 
 12  Sector             672 non-null    object 
 13  Revenue            672 non-null    object 
 14  Competitors        672 non-null    object 
dtypes: float64(1), int64(2), object(12)
memory usage: 78.9+ KB
```

Finding if there is any missing value in the dataset

In [5]:
```
any_null_columns = df.isnull().any()
print(any_null_columns)
```

```
index                False
Job Title            False
Salary Estimate      False
Job Description      False
Rating               False
Company Name         False
Location             False
Headquarters         False
Size                 False
Founded              False
Type of ownership    False
Industry             False
Sector               False
Revenue              False
Competitors          False
dtype: bool
```

Result: There are not any missing value

Index Column

In [6]:
```
#dropping this column as it used as serial number. (can affect the analysis)
df.drop(columns=['index'], inplace=True)
```

In [7]:
```
df.head(3)
```

Out[7]:

	Job Title	Salary Estimate	Job Description	Rating	Company Name	Location	Headquarters	Size	Founde
0	Sr Data Scientist	$137K–$171K (Glassdoor est.)	Description\n\nThe Senior Data Scientist is re...	3.1	Healthfirst\n3.1	New York, NY	New York, NY	1001 to 5000 employees	19
1	Data Scientist	$137K–$171K (Glassdoor est.)	Secure our Nation, Ignite your Future\n\nJoin ...	4.2	ManTech\n4.2	Chantilly, VA	Herndon, VA	5001 to 10000 employees	196
2	Data Scientist	$137K–$171K (Glassdoor est.)	Overview\n\n\nAnalysis Group is one of the lar...	3.8	Analysis Group\n3.8	Boston, MA	Boston, MA	1001 to 5000 employees	198

Job Title Column

In [8]:
```
df['Job Title'].unique()
```

Out[8]:
```
array(['Sr Data Scientist', 'Data Scientist',
       'Data Scientist / Machine Learning Expert',
       'Staff Data Scientist - Analytics',
       'Data Scientist - Statistics, Early Career', 'Data Modeler',
       'Experienced Data Scientist', 'Data Scientist - Contract',
       'Data Analyst II', 'Medical Lab Scientist',
       'Data Scientist/Machine Learning', 'Human Factors Scientist',
       'Business Intelligence Analyst I- Data Insights',
       'Data Scientist - Risk', 'Data Scientist-Human Resources',
       'Senior Research Statistician- Data Scientist', 'Data Engineer',
       'Associate Data Scientist', 'Business Intelligence Analyst',
       'Senior Analyst/Data Scientist', 'Data Analyst',
       'Machine Learning Engineer', 'Data Analyst I',
       'Scientist - Molecular Biology',
       'Computational Scientist, Machine Learning',
       'Senior Data Scientist', 'Jr. Data Engineer',
       'E-Commerce Data Analyst', 'Data Analytics Engineer',
       'Product Data Scientist - Ads Data Science',
       'Data Scientist - Intermediate', 'Global Data Analyst',
       'Data & Machine Learning Scientist',
       'Data Scientist - Machine Learning', 'Data Engineer (Remote)',
       'Data Scientist, Applied Machine Learning - Bay Area',
       'Principal Data Scientist', 'Business Data Analyst',
       'Purification Scientist', 'Data Engineer, Enterprise Analytics',
       'Data Scientist 3 (718)', 'Real World Science, Data Scientist',
       'Data Scientist - Image and Video Analytics',
       'Data Science Manager, Payment Acceptance - USA',
       'Data Scientist / Applied Mathematician',
       'Patient Safety- Associate Data Scientist',
       '(Sr.) Data Scientist -', 'Data Scientist, Kinship - NYC/Portland',
```

```
        'Applied Technology Researcher / Data Scientist',
        'Health Data Scientist - Biomedical/Biostats',
        'Staff Data Scientist', 'Sr Data Engineer (Sr BI Developer)',
        'Lead Data Scientist', 'RFP Data Analyst',
        'Data Scientist (TS/SCI)', 'Software Engineer - Data Science',
        'Data Analyst/Engineer', 'NGS Scientist', 'Senior Data Engineer',
        'Sr. ML/Data Scientist - AI/NLP/Chatbot',
        'Data Integration and Modeling Engineer',
        'Tableau Data Engineer 20-0117', 'AI Data Scientist',
        'Research Scientist Patient Preferences (Remote)',
        'Scientist - Biomarker and Flow Cytometry', 'Analytics Manager',
        'Staff Scientist- Upstream PD',
        'Sr Scientist - Extractables & Leachables',
        'ELISA RESEARCH SCIENTIST (CV-15)', 'Say Business Data Analyst',
        'Geospatial Data Scientist', 'Computational Scientist',
        'Senior Data Analyst', 'Sr Data Analyst',
        'Machine Learning Scientist - Bay Area, CA',
        'Senior Data Scientist - Algorithms',
        'Senior Data & Machine Learning Scientist',
        'Research Scientist - Patient-Centered Research (Remote)',
        'Jr. Business Data Analyst (position added 6/12/2020)',
        'Sr. Data Scientist II',
        'Production Engineer - Statistics/Data Analysis',
        'Statistical Scientist', 'Computational Behavioral Scientist',
        'Principal Data Scientist - Machine Learning',
        'Principal Machine Learning Scientist',
        'Senior Data Scientist - R&D Oncology',
        'Health Plan Data Analyst, Sr',
        'Principal Scientist/Associate Director, Quality Control and Analytical Technolog
ies',
        'Analytics - Business Assurance Data Analyst',
        'Senior Data Scientist – Image Analytics, Novartis AI Innovation Lab',
        'Data Science Instructor', 'Senior Business Intelligence Analyst',
        'In-Line Inspection Data Analyst',
        'Data Scientist - TS/SCI FSP or CI Required',
        'Data Scientist - TS/SCI Required',
        'Data Science Software Engineer',
        'ENGINEER - COMPUTER SCIENTIST - RESEARCH COMPUTER SCIENTIST - SIGNAL PROCESSING
- SAN ANTONIO OR',
        'AI Ops Data Scientist', 'Intelligence Data Analyst, Senior',
        'Analytics Manager - Data Mart',
        'Data Modeler (Analytical Systems)',
        'Senior Machine Learning Scientist - Bay Area, CA',
        'Report Writer-Data Analyst', 'Staff Data Scientist - Pricing',
        'Equity Data Insights Analyst - Quantitative Analyst',
        'Operations Data Analyst', 'Software Data Engineer',
        'Real World Evidence (RWE) Scientist', 'Computer Scientist 1',
        'Environmental Data Science', 'Staff BI and Data Engineer',
        'Data Scientist - Statistics, Mid-Career',
        'Director of Data Science',
        'Data Engineer, Digital & Comp Pathology',
        'Manager / Lead, Data Science & Analytics',
        'Diversity and Inclusion Data Analyst',
        'Data Scientist Machine Learning', 'Chief Scientist',
        'Development Scientist, Voltaren',
        'Principal Data & Analytics Platform Engineer',
        'Machine Learning Engineer/Scientist',
        'Data Analyst - Unilever Prestige', 'VP, Data Science',
        'Data Engineer - Kafka', 'Decision Scientist',
        'Data Science All Star Program - Data Engineer Track',
        'Scientist - Machine Learning', 'Sr. Data Scientist',
        'Applied AI Scientist / Engineer',
        'Data Engineer (Analytics, SQL, Python, AWS)',
        'Senior Data Analyst - Finance & Platform Analytics',
        'Market Research Data Scientist',
        'IT Partner Digital Health Technology and Data Science',
```

```
       'Software Engineer (Data Scientist, C,C++,Linux,Unix) - SISW - MG',
       'Senior Clinical Data Scientist Programmer',
       'Computer Vision / Deep Learning Scientist',
       'Data Solutions Engineer - Data Modeler',
       'Data Scientist (TS/SCI w/ Poly)',
       'Weapons and Sensors Engineer/Scientist',
       'Applied Computer Scientist', 'Cloud Data Engineer (Azure)',
       'Lead Certified Clinical Laboratory Scientist - Saturday - Tuesday, 8:00pm - 6:30
am shift',
       'Sr. Data Analyst',
       'Senior Scientist - Toxicologist - Product Integrity (Stewardship)',
       'Senior Machine Learning Engineer',
       'Data Scientist- Industrial Discrete Sector Industry',
       'Senior Principal Data Scientist (Python/R)',
       'Data Scientist(s)/Machine Learning Engineer',
       'Scientist / Group Lead, Cancer Biology',
       'Manager, Field Application Scientist, Southeast',
       'COMPUTER SCIENTIST - ENGINEER - RESEARCH COMPUTER SCIENTIST - SIGNAL PROCESSIN
G',
       'Machine Learning Scientist / Engineer', 'Data Science Analyst',
       'COMPUTER SCIENTIST - ENGINEER - RESEARCH COMPUTER SCIENTIST - TRANSPORTATION TEC
HNOLOGY',
       'Software Engineer - Machine Learning & Data Science (Applied Intelligence Servic
es Team)',
       'Clinical Data Analyst', 'Data Scientist Technical Specialist',
       'Data Science Manager', 'Big Data Engineer', 'Data Architect',
       'Aviation AI/ML Data Scientist', 'Machine Learning Engineer, Sr.',
       'Information Systems Engineering Specialist (Engineering Scientist)',
       'Scientist/Research Associate-Metabolic Engineering',
       'Vice President, Biometrics and Clinical Data Management',
       'Enterprise Data Analyst (Enterprise Portfolio Management Office)',
       'Lead Data Scientist - Network Analysis and Control',
       'Sr. Research Associate/ Scientist, NGS prep & Molecular Genomics',
       'Developer III - Data Science',
       'Hydrogen/Tritium Materials Scientist (Experienced)',
       'Data Scientist/Data Analytics Practitioner',
       'AI/ML - Machine Learning Scientist, Siri Understanding'],
      dtype=object)
```

Salary Estimate Column

In [9]:
```
df['Salary Estimate'].unique()
```

Out[9]:
```
array(['$137K-$171K (Glassdoor est.)', '$75K-$131K (Glassdoor est.)',
       '$79K-$131K (Glassdoor est.)', '$99K-$132K (Glassdoor est.)',
       '$90K-$109K (Glassdoor est.)', '$101K-$165K (Glassdoor est.)',
       '$56K-$97K (Glassdoor est.)', '$79K-$106K (Glassdoor est.)',
       '$71K-$123K (Glassdoor est.)', '$90K-$124K (Glassdoor est.)',
       '$91K-$150K (Glassdoor est.)', '$141K-$225K (Glassdoor est.)',
       '$145K-$225K(Employer est.)', '$79K-$147K (Glassdoor est.)',
       '$122K-$146K (Glassdoor est.)', '$112K-$116K (Glassdoor est.)',
       '$110K-$163K (Glassdoor est.)', '$124K-$198K (Glassdoor est.)',
       '$79K-$133K (Glassdoor est.)', '$69K-$116K (Glassdoor est.)',
       '$31K-$56K (Glassdoor est.)', '$95K-$119K (Glassdoor est.)',
       '$212K-$331K (Glassdoor est.)', '$66K-$112K (Glassdoor est.)',
       '$128K-$201K (Glassdoor est.)', '$138K-$158K (Glassdoor est.)',
       '$80K-$132K (Glassdoor est.)', '$87K-$141K (Glassdoor est.)',
       '$92K-$155K (Glassdoor est.)', '$105K-$167K (Glassdoor est.)'],
      dtype=object)
```

In [10]:
```
# function to separate the upper limit and lower limit of the salary
def salary_to_numeric(salary_range):
    salary_range = salary_range.replace('K', '')  # Remove 'K'
    salary_range = salary_range.split('-')  # Split into low and high values
    low_salary = int(salary_range[0][1:]) * 1000  # Convert to numeric value in thousand
```

```
    high_salary = salary_range[1].split()[0]
    if high_salary.isdigit():  # Check if the high salary is a valid integer
        high_salary = int(high_salary) * 1000  # Convert to numeric value in thousands
    else:
        high_salary = low_salary  # If the high salary is not a valid integer, use the l

    return low_salary, high_salary

# Apply the function to the 'Salary Estimate' column and create two new columns for low
df[['Low_Salary_in_dollar', 'High_Salary_in_dollar']] = df['Salary Estimate'].apply(sala
```

In [11]:
```
df[['Low_Salary_in_dollar', 'High_Salary_in_dollar']]
```

Out[11]:

	Low_Salary_in_dollar	High_Salary_in_dollar
0	137000	137000
1	137000	137000
2	137000	137000
3	137000	137000
4	137000	137000
...	...	...
667	105000	105000
668	105000	105000
669	105000	105000
670	105000	105000
671	105000	105000

672 rows × 2 columns

Note:- we will be using the upper limit(High_Salary_in_dollar) in this analysis.

Job Description Column

In [12]:
```
#dropping this column
df.drop(columns=['Job Description'], inplace=True)
```

Rating Column

In [13]:
```
df['Rating'].unique()
```

Out[13]:
```
array([ 3.1,  4.2,  3.8,  3.5,  2.9,  3.9,  4.4,  3.6,  4.5,  4.7,  3.7,
        3.4,  4.1,  3.2,  4.3,  2.8,  5. ,  4.8,  3.3,  2.7,  2.2,  2.6,
        4. ,  2.5,  4.9,  2.4, -1. ,  2.3,  4.6,  3. ,  2.1,  2. ])
```

In [15]:
```
# the Scale for rating is 0 to 5

# Replace '-1' with NaN
df['Rating'] = df['Rating'].replace(-1.0, np.nan).round(1)

# Printing the unique values in the 'Rating' column
unique_ratings = df['Rating'].unique()
df['Rating'].unique()
```

Out[15]:
```
array([3.1, 4.2, 3.8, 3.5, 2.9, 3.9, 4.4, 3.6, 4.5, 4.7, 3.7, 3.4, 4.1,
       3.2, 4.3, 2.8, 5. , 4.8, 3.3, 2.7, 2.2, 2.6, 4. , 2.5, 4.9, 2.4,
       nan, 2.3, 4.6, 3. , 2.1, 2. ])
```

Company Name Column

In [16]:
```
df['Company Name'].unique()
```

Out[16]:
```
array(['Healthfirst\n3.1', 'ManTech\n4.2', 'Analysis Group\n3.8',
       'INFICON\n3.5', 'Affinity Solutions\n2.9', 'HG Insights\n4.2',
       'Novartis\n3.9', 'iRobot\n3.5', 'Intuit - Data\n4.4',
       'XSELL Technologies\n3.6', 'Novetta\n4.5', '1904labs\n4.7',
       'PNNL\n3.7', 'Old World Industries\n3.1',
       'Mathematica Policy Research\n3.4',
       'Guzman & Griffin Technologies (GGTI)\n4.4',
       'Upside Business Travel\n4.1', 'Buckman\n3.5',
       'Insight Enterprises, Inc.\n4.2', 'Tower Health\n3.5',
       'Triplebyte\n3.2', 'PulsePoint\n4.3', 'Exponent\n3.5',
       'Guardian Life\n3.5',
       'Spectrum Communications and Consulting\n3.4',
       'Oversight Systems\n4.7', 'LSQ\n4.2',
       'MIT Lincoln Laboratory\n3.8', 'Kingfisher Systems\n4.5',
       'Formation\n2.8', 'Cohere Health\n5.0', 'Acuity Insurance\n4.8',
       'Chef\n3.6', 'Puget Sound Energy\n3.3', 'Sandhills Global\n2.7',
       'A Place for Mom\n2.7', 'Great-Circle Technologies\n2.2',
       'Edmunds.com\n3.4', 'Cambridge Associates, LLC\n3.1',
       'Liberty Mutual Insurance\n3.4', 'Cenlar\n2.6',
       'Arsenal Biosciences\n5.0', 'Eversight\n4.2', 'Pfizer\n4.1',
       'Klaviyo\n4.8', 'Intellectual Ventures\n3.3', 'GovTech\n3.7',
       'Quick Base\n4.3', 'Giving Assistant\n4.8', 'Takeda\n3.7',
       'Netskope\n4.0', 'IT Concepts\n4.8', 'iSeatz\n3.5',
       'Summa Health System\n3.7', 'Benson Hill\n3.5', 'Twitter\n4.1',
       'Postmates - Corporate HQ\n3.2', 'Envision LLC\n4.5',
       'Swiss Re\n3.8', 'Systems & Technology Research\n4.5',
       'Dermalogica\n3.8', 'Bayview Asset Management\n3.7',
       'Via Transportation\n3.7', 'Grid Dynamics\n4.0',
       'Tempus Labs\n3.3', 'CareDx\n2.5', 'IZEA\n4.2', 'Autodesk\n4.0',
       'Caterpillar\n3.7', 'New England Biolabs\n4.9',
       'Allied Solutions\n3.4', 'The Knot Worldwide\n3.5',
       'IFG Companies\n2.9', 'Amyris\n3.3', 'AstraZeneca\n4.0',
       'Powertek\n3.6', 'Object Partners\n4.7', 'The Mom Project\n4.9',
       'Lightspeed Systems\n4.3', 'Stripe\n4.0',
       'Comprehensive Healthcare\n2.6',
       'Fullpower Technologies, Inc.\n4.5', 'Mars\n3.9',
       'NuWave Solutions\n4.4', 'Merrick Bank\n3.6', 'QOMPLX\n3.5',
       'GutCheck\n3.8', 'Inter-American Development Bank\n3.5',
       'Avlino\n4.9', 'Stratagem Group\n4.4', 'Evidation\n4.1',
       'Tecolote Research\n3.8', 'Tivity Health\n3.2', 'hc1\n2.9',
       'HP Inc.\n4.1', 'SAIC\n3.7', 'AllianceBernstein\n3.2',
       'Big Huge Games\n4.9', 'Maxar Technologies\n3.5',
       'Phantom AI\n5.0', 'Noblis\n4.0', 'Spring Health\n3.6',
       'ClearEdge\n4.0', 'GetWellNetwork\n4.8', 'TACG Solutions\n4.5',
       'Scoop\n4.7', 'Montway Inc\n3.4', 'Juniper Networks\n3.8',
       'Notion Labs\n5.0', 'Lendio\n4.9', 'Direct Agents\n4.4',
       'NAVEX Global\n3.3', 'Upstart\n4.2', 'AppLovin\n4.8',
       'ISO New England\n3.8', 'Relativity\n3.7', 'Tempo Automation\n3.3',
       'MITRE\n3.3', 'Expedition Technology, Inc.\n5.0', 'Evidera\n3.8',
       'Plymouth Rock Assurance\n3.4', 'Crown Bioscience\n2.4',
       'GNS Healthcare\n2.9', 'OneMagnify\n4.4', 'SPECTRUM\n2.9',
       'Advanced BioScience Laboratories\n2.7',
       'Procore Technologies\n4.2', 'Ritedose\n3.5',
       'Covid-19 Search Partners', 'bioMérieux\n4.2',
       'Radical Convergence', 'Leidos\n3.5', 'Demandbase\n4.5',
       'Shelter Insurance\n4.1', 'USAC\n2.7',
       'General Dynamics Information Technology\n3.4', 'Offerpad\n4.4',
```

'Magna International Inc.\n3.5', 'United BioSource\n2.3',
'Kelly\n3.4', 'C3.ai\n4.7', 'Quartet Health\n3.9',
'Midland Credit Management\n3.3',
'Resurgent Capital Services\n4.4', 'webfx.com\n4.7',
'Argo Group US\n3.5', 'BWX Technologies\n3.3', 'Life360\n3.9',
'MassMutual\n3.7', 'Natera\n3.9', 'Genentech\n4.0', 'Ntrepid\n4.2',
'Constant Contact\n3.6', 'Sage Intacct\n4.7',
'Shape Security\n4.1', 'SkillSoniq\n5.0', 'Joby Aviation\n4.3',
"Cook Children's Health Care System\n3.8",
'Rubius Therapeutics\n3.8', 'GreatAmerica Financial Services\n4.6',
'Coverent\n4.1', 'Mteq\n3.7', 'Rocket Lawyer\n4.4',
'Alion Science & Technology\n3.6', 'Protolabs\n3.7',
'Quest Integrity\n2.9', 'Phoenix Operations Group\n5.0',
'Dice.com\n3.4', 'Southwest Research Institute\n3.9',
'The Buffalo Group\n4.3',
'Central California Alliance for Health\n3.5',
'Security Finance Corporation of Spartanburg\n3.1',
'Opendoor\n3.6', 'Global Data Management Inc\n4.5',
'Photon Infotech\n3.0', 'REE\n5.0',
'Riverside Research Institute\n3.6', 'T. Rowe Price\n3.6',
'Encode, Inc.', 'Brighthouse Financial\n3.8',
'II-VI Incorporated\n3.3', 'Surya Systems\n4.6', 'PayPal\n3.8',
'Predictive Research Inc\n3.9', '1010data\n3.1', 'Gigya\n3.6',
'Genesis Research\n5.0', 'Sanofi\n3.7', 'XPO Logistics\n3.7',
'Trace Data\n3.9', 'Descript\n4.3',
'Rincon Research Corporation\n4.2', 'Better Hire\n4.0',
'Parker Hannifin\n3.3', 'Gallup\n4.1', 'Insider Inc\n3.3',
'Rapid Value Solutions\n3.9', 'Battelle\n3.1',
'The Drive Media, Inc.\n5.0',
'Pacific Northwest National Laboratory\n3.7',
'US Pharmacopeia\n3.2', 'Itlize Global\n4.6', 'eBay\n3.5',
'Paige\n5.0', 'ABIOMED\n4.1', 'Comcast\n3.5',
'Metronome, LLC\n3.2', 'Lawrence Livermore National Lab\n4.7',
'FHLBank Pittsburgh\n3.8', 'Jacobs\n3.6',
'Underwriters Laboratories\n3.3', 'Altus Group\n3.7', 'Jobot\n5.0',
'Trovetechs Inc', 'Oshkosh Corporation\n4.2', 'Mackin\n3.4',
'PETADATA', 'VBeyond Corporation\n4.4', 'Take-Two\n3.7',
'Colony Brands\n3.7', 'Capio Group\n4.1', 'SleePare\n3.4',
'ShorePoint\n4.5', 'Dolphin\n3.5', 'TE Connectivity\n3.6',
'State of Virginia\n3.2', 'TA Digital\n3.7',
'Market America Inc\n4.0', 'TrueAccord\n3.4',
'ALTA IT Services\n3.9', 'Kollasoft Inc.\n3.2',
'ASRC Federal Holding Company\n3.4', 'Adwait Algorithm\n4.4',
'Cambridge FX\n3.5', 'Metromile\n3.8', 'Criteo\n3.9',
'Advance Sourcing Concepts\n3.4', 'Enterprise Solutions Inc\n3.8',
'Microagility', 'Conch Technologies, Inc\n4.6', 'GSK\n3.9',
'Rainmaker Resources, LLC', '22nd Century Technologies\n3.7',
'Huxley\n3.3', 'FM Systems\n3.4', 'B4Corp',
'Blue Cross and Blue Shield of North Carolina\n3.7',
'Jane Street\n4.8', 'SSATI\n5.0',
'Solving IT International Inc\n3.4',
'The Davey Tree Expert Company\n3.3', 'Centauri\n4.6',
'Stride Search', 'Software Engineering Institute\n2.6',
'TechProjects\n4.8', '7Park Data\n3.9',
'Ameritas Life Insurance Corp\n3.0', 'Western Digital\n3.5',
'Shimento, Inc.\n2.9', 'Averity\n5.0', 'Praxis Engineering\n4.7',
'Point72 Ventures',
'Johns Hopkins University Applied Physics Laboratory\n4.5',
'Cambridge Mobile Telematics\n4.9', 'Blend360\n4.6',
'Nolij Consulting\n3.9', 'Hatch Data Inc',
'Compass Consulting Group\n4.7', 'SolutionIT, Inc.\n4.4',
'Perspecta\n3.2', 'Smith Hanley Associates\n4.5',
'Allen Institute\n3.5', 'Eliassen Group\n4.4',
'Bayside Solutions\n3.1', 'Evolve Vacation Rental\n3.5',
'AgreeYa Solutions\n3.8', 'Carolina Power & Light Co\n3.7',
'New Iron Group, Inc.\n5.0', 'Travelers\n4.0', 'Twitch\n3.6',

'Biogen\n3.6', 'HireAi', 'Mentor Graphics\n4.1',
'WCG (WIRB-Copernicus Group)\n3.6',
'Visionary Integration Professionals\n4.3', 'Dynetics\n4.0',
'Navy Federal Credit Union\n3.9',
'Exact Sciences Corporation\n4.0',
'Community Behavioral Health\n3.6', 'Reynolds American\n3.3',
'LifeOmic\n5.0', 'Visionist, Inc.\n4.9', 'Navio',
'Concerto HealthAI\n3.3', 'Evolvinc', 'PROPRIUS\n5.0',
'TECHNOCRAFT Solutions\n3.4', 'Latitude, Inc.\n4.1',
'Royce Geospatial\n5.0', 'CyberCoders\n4.2',
'Booz Allen Hamilton Inc.\n3.7', 'Burns & McDonnell\n3.8',
'InvenTech Info\n4.8', 'Robert Half\n3.5',
'Conflux Systems Inc.\n4.5', 'Voice\n3.4',
'Falcon IT & Staffing Solutions', 'DataLab USA\n3.6',
'Werner Enterprises Inc\n3.1', 'PeopleCom\n5.0',
'VANTA Partners\n5.0', 'Blue Icy Water, LLC',
"Farmer's Business Network, Inc.\n3.5", 'Sonde Health',
'Maxiom\n5.0', 'Change Healthcare\n2.7', 'DCS Corp\n4.1',
'Hive (CA)\n2.1', 'Hackensack Meridian Health\n3.3',
'Net2Source Inc.\n3.2', 'The Trade Desk\n3.2', 'IBM\n3.7',
'Knowesis Inc.\n4.4', 'MoTek Technologies\n3.1', 'HPOne\n3.5',
'Blue Cloak LLC', 'TBWA\\Chiat\\Day\n2.7',
'ThreeBridge Solutions\n3.5', 'Numeric, LLC\n3.2',
'Centraprise\n4.2', 'DW Simpson\n4.2', 'LinQuest\n3.9',
'Trexquant Investment\n4.0', 'Fleetcor\n3.7',
'Radiant Digital\n4.5', 'Child Care Aware of America\n2.8',
'IntelliPro Group Inc.\n4.1', 'USI\n3.4', 'Apex Systems\n3.9',
'Pragmatics, Inc.\n2.9', 'Crossover Health\n3.5',
'Lorven Technologies Inc\n4.0', 'Gap Inc.\n3.5',
'Tygart Technology, Inc\n4.7', 'Murray Resources\n4.6',
'New York Technology Partners\n4.0',
'Two95 International Inc.\n4.0', 'Sophinea', 'CRS Group\n4.7',
'Blackstone Talent Group\n3.5', 'Roche\n4.1',
'Creative Circle\n3.6', 'Blue Horizon Tek Solutions\n5.0',
'Sharpedge Solutions Inc\n4.7',
'Alaka`ina Foundation Family of Companies\n3.6',
'Hexagon US Federal\n2.7', 'Monte Rosa Therapeutics',
'Comtech Global Inc\n4.0', 'Aveshka, Inc.\n3.8',
'10x Genomics\n4.2', 'CompuForce', '1-800-Flowers\n2.7',
'Aptive\n3.5', 'JCD Staffing\n5.0', 'Thumbtack\n3.9',
'MILVETS Systems Technology, Inc.\n3.4', 'Apple\n4.1',
'Kforce\n4.1', 'OppLoans\n4.4', 'Brillient\n3.9',
'Xator Corporation\n2.9', 'HAN IT Staffing Inc.\n4.6',
'Infinitive Inc\n3.4', 'New Relic\n4.7', 'ICW Group\n3.3',
'NYSTEC\n3.8', 'E3 Federal Solutions\n4.5', 'Peraton\n3.4',
'Group O\n3.1', 'CaptiveAire\n4.1', 'Temboo\n3.9', 'Kibo',
'AeroVironment\n4.2', 'Applied Research Laboratories\n3.8',
'Conagen\n2.0', 'Alector\n4.8', 'Homology Medicines, Inc.\n4.4',
'Inland Empire Health Plan\n3.3',
'Sandia National Laboratories\n3.8', 'CIA\n3.8',
'Maven Wave Partners\n4.5', 'Ovative Group\n4.3', 'Kognetics\n3.6',
'Envision Healthcare\n2.9', 'ConsumerTrack\n3.2',
'Meridian Knowledge Solutions\n4.4', 'UST Global\n4.2',
'IMG Systems\n3.2', 'Trident Systems Inc\n3.4', 'GrainBridge, LLC',
'First Health Group\n3.2', 'Sprezzatura Management Consulting',
'Progress Rail, A Caterpillar Company\n2.8',
'Axiologic Solutions\n4.5', 'Indigo Slate\n3.0', 'Cubic\n3.3',
'Advanced Bio-Logic Solutions Corp\n4.0',
'Alignment Healthcare\n3.5', 'WGSN\n3.5',
'ISYS Technologies, Inc.\n3.6', 'TransVoyant\n3.0', 'Geotab\n4.3',
'EGlobalTech\n3.7', 'Central Business Solutions, Inc\n3.0',
'KeHE Distributors\n2.5', 'Moxie Software\n3.0',
'Unicom Technologies INC\n4.7', 'Americo Life\n3.3',
'Tokio Marine HCC\n3.3', 'CACI International\n3.5',
'Berico Technologies', 'Kehe Food Distributors', 'Pactera Edge',
'Qurate Retail Group\n3.6', 'A-Line Staffing Solutions\n4.1',

```
       'Clear Ridge Defense', 'Criterion Systems, Inc.\n3.8',
       'Foundation Medicine\n4.0', 'TRANZACT\n3.6', 'JKGT', 'AccessHope',
       'ChaTeck Incorporated\n5.0'], dtype=object)
```

In [17]:
```
# removing '\n' and number from company name
def remove_newline_and_number(value):
    return value.split('\n')[0]

# Apply the function to each element in the data array
df['Company Name'] = list(map(remove_newline_and_number, df['Company Name']))
df['Company Name']
```

Out[17]:
```
0              Healthfirst
1                  ManTech
2          Analysis Group
3                  INFICON
4       Affinity Solutions
              ...
667               TRANZACT
668                   JKGT
669             AccessHope
670    ChaTeck Incorporated
671          1-800-Flowers
Name: Company Name, Length: 672, dtype: object
```

Location Column

In [18]:
```
df['Location'].unique()
```

Out[18]:
```
array(['New York, NY', 'Chantilly, VA', 'Boston, MA', 'Newton, MA',
       'Santa Barbara, CA', 'Cambridge, MA', 'Bedford, MA',
       'San Diego, CA', 'Chicago, IL', 'Herndon, VA', 'Saint Louis, MO',
       'Richland, WA', 'Northbrook, IL', 'Washington, DC', 'Remote',
       'Memphis, TN', 'Plano, TX', 'West Grove, PA', 'Phoenix, AZ',
       'Appleton, WI', 'Atlanta, GA', 'Orlando, FL', 'Lexington, MA',
       'McLean, VA', 'San Francisco, CA', 'Sheboygan, WI',
       'United States', 'Bothell, WA', 'Lincoln, NE', 'Overland Park, KS',
       'Santa Monica, CA', 'Portsmouth, NH', 'Ewing, NJ',
       'South San Francisco, CA', 'Palo Alto, CA', 'Bellevue, WA',
       'New Orleans, LA', 'Akron, OH', 'Fort Wayne, IN', 'Woburn, MA',
       'Carson, CA', 'Coral Gables, FL', 'Santa Clara, CA',
       'Brisbane, CA', 'Winter Park, FL', 'Redwood City, CA',
       'Peoria, IL', 'Ipswich, MA', 'Carmel, IN', 'Emeryville, CA',
       'Gaithersburg, MD', 'Longmont, CO', 'Austin, TX', 'Yakima, WA',
       'Santa Cruz, CA', 'Springfield, VA', 'Alexandria, VA', 'Utah',
       'Reston, VA', 'Denver, CO', 'New Jersey', 'Aurora, CO',
       'Hill AFB, UT', 'Chandler, AZ', 'Indianapolis, IN',
       'Nashville, TN', 'Timonium, MD', 'Burlingame, CA',
       'Annapolis Junction, MD', 'Bethesda, MD', 'Dayton, OH',
       'Schaumburg, IL', 'Cupertino, CA', 'Lehi, UT', 'Culver City, CA',
       'Lake Oswego, OR', 'San Mateo, CA', 'Holyoke, MA',
       'Woodbridge, NJ', 'Dearborn, MI', 'Maryland Heights, MO',
       'Rockville, MD', 'Carpinteria, CA', 'Columbia, SC',
       'Hauppauge, NY', 'Fort Meade, MD', 'Columbia, MO', 'Vicksburg, MS',
       'Birmingham, AL', 'Blue Bell, PA', 'Cincinnati, OH',
       'Harrisburg, PA', 'Oak Ridge, TN', 'San Carlos, CA', 'Waltham, MA',
       'Fort Worth, TX', 'Smithfield, RI', 'Cedar Rapids, IA',
       'Fort Belvoir, VA', 'Linthicum Heights, MD', 'Maple Plain, MN',
       'Tulsa, OK', 'Baltimore, MD', 'Oklahoma City, OK',
       'Scotts Valley, CA', 'Spartanburg, SC', 'Hartford, CT',
       'Beavercreek, OH', 'Norfolk, VA', 'Charlotte, NC', 'Champaign, IL',
       'Texas', 'Hoboken, NJ', 'Lebanon, IN', 'Oakland, CA',
       'Melbourne, FL', 'Cleveland, OH', 'Norwell, MA', 'San Jose, CA',
       'Piscataway, NJ', 'Danvers, MA', 'Vienna, VA', 'Livermore, CA',
       'Pittsburgh, PA', 'Irvine, CA', 'Oshkosh, WI', 'Menlo Park, CA',
```

```
       'Dallas, TX', 'Arlington, VA', 'Monroe, WI', 'Sacramento, CA',
       'Hampton, VA', 'Richmond, VA', 'Monterey, CA', 'Woodlawn, MD',
       'Ann Arbor, MI', 'Concord, CA', 'Durham, NC', 'Kent, OH',
       'Laurel, MD', 'Columbia, MD', 'Falls Church, VA',
       'Thousand Oaks, CA', 'Edison, NJ', 'Adelphi, MD', 'Seattle, WA',
       'Sunnyvale, CA', 'Fremont, CA', 'Hamilton, NJ', 'Huntsville, AL',
       'Merrifield, VA', 'Madison, WI', 'Philadelphia, PA',
       'Winston-Salem, NC', 'Raleigh, NC', 'Burbank, CA', 'San Ramon, CA',
       'Oxnard, CA', 'Kansas City, MO', 'Jersey City, NJ',
       'Manchester, NH', 'Winters, TX', 'Brooklyn, NY', 'Germantown, MD',
       'Omaha, NE', 'Open Fork, VA', 'Ashburn, VA', 'Lombard, IL',
       'Alpharetta, GA', 'Boulder, CO', 'Mountain View, CA',
       'Trumbull, CT', 'Sterling, VA', 'Foster City, CA', 'Frederick, MD',
       'Colorado Springs, CO', 'Southfield, MI', 'San Clemente, CA',
       'The Woodlands, TX', 'Pleasanton, CA', 'Wilmington, DE',
       'Fort Sam Houston, TX', 'Lexington Park, MD',
       'Patuxent, Anne Arundel, MD', 'Fairfax, VA', 'San Antonio, TX',
       'Silver Spring, MD', 'Portland, OR', 'Simi Valley, CA',
       'New Bedford, MA', 'Rancho Cucamonga, CA', 'Collegeville, PA',
       'Minneapolis, MN', 'Gahanna, OH', 'California', 'Wellesley, MA',
       'Washington, VA', 'Orange, CA', 'Bridgeport, WV', 'Oakville, CA',
       'Naperville, IL', 'Houston, TX', 'Redmond, WA', 'West Chester, PA',
       'Quantico, VA', 'Fort Lee, NJ', 'Irwindale, CA'], dtype=object)
```

Headquarters Column

In [20]:
```
#dropping this column
df.drop(columns=['Headquarters'], inplace=True)
```

Size Column

In [21]:
```
df['Size'].unique()
```

Out[21]:
```
array(['1001 to 5000 employees', '5001 to 10000 employees',
       '501 to 1000 employees', '51 to 200 employees', '10000+ employees',
       '201 to 500 employees', '1 to 50 employees', '-1', 'Unknown'],
      dtype=object)
```

In [22]:
```
#Replacing '-1' and 'Unknown' with NaN
df['Size'].replace(['-1', 'Unknown'], np.nan, inplace=True)
df['Size'].unique()
```

Out[22]:
```
array(['1001 to 5000 employees', '5001 to 10000 employees',
       '501 to 1000 employees', '51 to 200 employees', '10000+ employees',
       '201 to 500 employees', '1 to 50 employees', nan], dtype=object)
```

Founded Column

In [23]:
```
df['Founded'].unique()
```

Out[23]:
```
array([1993, 1968, 1981, 2000, 1998, 2010, 1996, 1990, 1983, 2014, 2012,
       2016, 1965, 1973, 1986, 1997, 2015, 1945, 1988, 2017, 2011, 1967,
       1860, 1992, 2003, 1951, 2005, 2019, 1925, 2008, 1999, 1978, 1966,
       1912, 1958, 2013, 1849, 1781, 1926, 2006, 1994, 1863, 1995,   -1,
       1982, 1974, 2001, 1985, 1913, 1971, 1911, 2009, 1959, 2007, 1939,
       2002, 1961, 1963, 1969, 1946, 1957, 1953, 1948, 1850, 1851, 2004,
       1976, 1918, 1954, 1947, 1955, 2018, 1937, 1917, 1935, 1929, 1820,
       1952, 1932, 1894, 1960, 1788, 1830, 1984, 1933, 1880, 1887, 1970,
       1942, 1980, 1989, 1908, 1853, 1875, 1914, 1898, 1956, 1977, 1987,
       1896, 1972, 1949, 1962], dtype=int64)
```

Type of ownership Column

In [24]:
```
df['Type of ownership'].unique()
```

Out[24]:
```
array(['Nonprofit Organization', 'Company - Public',
       'Private Practice / Firm', 'Company - Private', 'Government',
       'Subsidiary or Business Segment', 'Other Organization', '-1',
       'Unknown', 'Hospital', 'Self-employed', 'College / University',
       'Contract'], dtype=object)
```

In [25]:
```
#Replacing '-1' with NaN
df['Type of ownership'].replace(['-1', 'Unknown'], np.nan, inplace=True)
df['Type of ownership'].unique()
```

Out[25]:
```
array(['Nonprofit Organization', 'Company - Public',
       'Private Practice / Firm', 'Company - Private', 'Government',
       'Subsidiary or Business Segment', 'Other Organization', nan,
       'Hospital', 'Self-employed', 'College / University', 'Contract'],
      dtype=object)
```

Industry Column

In [26]:
```
df['Industry'].unique()
```

Out[26]:
```
array(['Insurance Carriers', 'Research & Development', 'Consulting',
       'Electrical & Electronic Manufacturing', 'Advertising & Marketing',
       'Computer Hardware & Software', 'Biotech & Pharmaceuticals',
       'Consumer Electronics & Appliances Stores',
       'Enterprise Software & Network Solutions', 'IT Services', 'Energy',
       'Chemical Manufacturing', 'Federal Agencies', 'Internet',
       'Health Care Services & Hospitals',
       'Investment Banking & Asset Management', 'Aerospace & Defense',
       'Utilities', '-1', 'Express Delivery Services',
       'Staffing & Outsourcing', 'Insurance Agencies & Brokerages',
       'Consumer Products Manufacturing', 'Industrial Manufacturing',
       'Food & Beverage Manufacturing', 'Banks & Credit Unions',
       'Video Games', 'Shipping', 'Telecommunications Services',
       'Lending', 'Cable, Internet & Telephone Providers', 'Real Estate',
       'Venture Capital & Private Equity', 'Miscellaneous Manufacturing',
       'Oil & Gas Services', 'Transportation Equipment Manufacturing',
       'Telecommunications Manufacturing', 'Transportation Management',
       'News Outlet', 'Architectural & Engineering Services',
       'Food & Beverage Stores', 'Other Retail Stores',
       'Hotels, Motels, & Resorts', 'State & Regional Agencies',
       'Financial Transaction Processing', 'Timber Operations',
       'Colleges & Universities', 'Travel Agencies', 'Accounting',
       'Logistics & Supply Chain', 'Farm Support Services',
       'Social Assistance', 'Construction',
       'Department, Clothing, & Shoe Stores', 'Publishing',
       'Health, Beauty, & Fitness', 'Wholesale', 'Rail'], dtype=object)
```

In [27]:
```
#Replacing '-1' with NaN
df['Industry'].replace(['-1'], np.nan, inplace=True)
df['Industry'].unique()
```

Out[27]:
```
array(['Insurance Carriers', 'Research & Development', 'Consulting',
       'Electrical & Electronic Manufacturing', 'Advertising & Marketing',
       'Computer Hardware & Software', 'Biotech & Pharmaceuticals',
       'Consumer Electronics & Appliances Stores',
       'Enterprise Software & Network Solutions', 'IT Services', 'Energy',
       'Chemical Manufacturing', 'Federal Agencies', 'Internet',
       'Health Care Services & Hospitals',
       'Investment Banking & Asset Management', 'Aerospace & Defense',
       'Utilities', nan, 'Express Delivery Services',
       'Staffing & Outsourcing', 'Insurance Agencies & Brokerages',
       'Consumer Products Manufacturing', 'Industrial Manufacturing',
```

```
       'Food & Beverage Manufacturing', 'Banks & Credit Unions',
       'Video Games', 'Shipping', 'Telecommunications Services',
       'Lending', 'Cable, Internet & Telephone Providers', 'Real Estate',
       'Venture Capital & Private Equity', 'Miscellaneous Manufacturing',
       'Oil & Gas Services', 'Transportation Equipment Manufacturing',
       'Telecommunications Manufacturing', 'Transportation Management',
       'News Outlet', 'Architectural & Engineering Services',
       'Food & Beverage Stores', 'Other Retail Stores',
       'Hotels, Motels, & Resorts', 'State & Regional Agencies',
       'Financial Transaction Processing', 'Timber Operations',
       'Colleges & Universities', 'Travel Agencies', 'Accounting',
       'Logistics & Supply Chain', 'Farm Support Services',
       'Social Assistance', 'Construction',
       'Department, Clothing, & Shoe Stores', 'Publishing',
       'Health, Beauty, & Fitness', 'Wholesale', 'Rail'], dtype=object)
```

Sector Column

In [28]:
```
df['Sector'].unique()
```

Out[28]:
```
array(['Insurance', 'Business Services', 'Manufacturing',
       'Information Technology', 'Biotech & Pharmaceuticals', 'Retail',
       'Oil, Gas, Energy & Utilities', 'Government', 'Health Care',
       'Finance', 'Aerospace & Defense', '-1',
       'Transportation & Logistics', 'Media', 'Telecommunications',
       'Real Estate', 'Travel & Tourism', 'Agriculture & Forestry',
       'Education', 'Accounting & Legal', 'Non-Profit',
       'Construction, Repair & Maintenance', 'Consumer Services'],
      dtype=object)
```

In [29]:
```
#Replacing '-1' with NaN
df['Sector'].replace(['-1'], np.nan, inplace=True)
df['Sector'].unique()
```

Out[29]:
```
array(['Insurance', 'Business Services', 'Manufacturing',
       'Information Technology', 'Biotech & Pharmaceuticals', 'Retail',
       'Oil, Gas, Energy & Utilities', 'Government', 'Health Care',
       'Finance', 'Aerospace & Defense', nan,
       'Transportation & Logistics', 'Media', 'Telecommunications',
       'Real Estate', 'Travel & Tourism', 'Agriculture & Forestry',
       'Education', 'Accounting & Legal', 'Non-Profit',
       'Construction, Repair & Maintenance', 'Consumer Services'],
      dtype=object)
```

Revenue Column

In [30]:
```
df['Revenue'].unique()
```

Out[30]:
```
array(['Unknown / Non-Applicable', '$1 to $2 billion (USD)',
       '$100 to $500 million (USD)', '$10+ billion (USD)',
       '$2 to $5 billion (USD)', '$500 million to $1 billion (USD)',
       '$5 to $10 billion (USD)', '$10 to $25 million (USD)',
       '$25 to $50 million (USD)', '$50 to $100 million (USD)',
       '$1 to $5 million (USD)', '$5 to $10 million (USD)',
       'Less than $1 million (USD)', '-1'], dtype=object)
```

In [31]:
```
#Replacing '-1' with NaN
df['Revenue'].replace(['-1','Unknown / Non-Applicable'], np.nan, inplace=True)
df['Revenue'].unique()
```

Out[31]:
```
array([nan, '$1 to $2 billion (USD)', '$100 to $500 million (USD)',
       '$10+ billion (USD)', '$2 to $5 billion (USD)',
       '$500 million to $1 billion (USD)', '$5 to $10 billion (USD)',
       '$10 to $25 million (USD)', '$25 to $50 million (USD)',
```

```
       '$50 to $100 million (USD)', '$1 to $5 million (USD)',
       '$5 to $10 million (USD)', 'Less than $1 million (USD)'],
      dtype=object)
```

Exploratory Data Analysis(EDA)

In [32]:
```
# Get basic statistics of numerical variables (mean, median, min, max)
df.describe()
```

Out[32]:

	Rating	Founded	Low_Salary_in_dollar	High_Salary_in_dollar
count	622.000000	672.000000	672.000000	672.000000
mean	3.881833	1635.529762	99196.428571	99196.428571
std	0.610805	756.746640	33009.958111	33009.958111
min	2.000000	-1.000000	31000.000000	31000.000000
25%	3.500000	1917.750000	79000.000000	79000.000000
50%	3.800000	1995.000000	91000.000000	91000.000000
75%	4.400000	2009.000000	122000.000000	122000.000000
max	5.000000	2019.000000	212000.000000	212000.000000

Identifying the categorical data types

In [34]:
```
#Get the data types of each column in the dataset
data_types = df.dtypes

#Identify columns with object or categorical data type
categorical_columns = data_types[data_types == 'object'].index.tolist()

print("Categorical Columns:")
print(categorical_columns)
```

```
Categorical Columns:
['Job Title', 'Salary Estimate', 'Company Name', 'Location', 'Size', 'Type of ownershi
p', 'Industry', 'Sector', 'Revenue', 'Competitors']
```

Finding the value count of categorical columns

In [44]:
```
#Select categorical columns for analysis
categorical_columns = ['Size', 'Type of ownership', 'Industry', 'Sector', 'Revenue', 'Lo

#Get unique values and their counts in each categorical column
for col in categorical_columns:

    value_counts = df[col].value_counts()
    print(f"\nValue counts in '{col}':")
    print(value_counts)
```

```
Value counts in 'Size':
51 to 200 employees        135
1001 to 5000 employees     104
1 to 50 employees           86
201 to 500 employees        85
10000+ employees            80
501 to 1000 employees       77
5001 to 10000 employees     61
Name: Size, dtype: int64
```

```
Value counts in 'Type of ownership':
Company - Private                 397
Company - Public                  153
Nonprofit Organization             36
Subsidiary or Business Segment     28
Government                         10
Other Organization                  5
Private Practice / Firm             4
College / University                3
Self-employed                       2
Contract                            2
Hospital                            1
Name: Type of ownership, dtype: int64

Value counts in 'Industry':
Biotech & Pharmaceuticals                    66
IT Services                                  61
Computer Hardware & Software                 57
Aerospace & Defense                          46
Enterprise Software & Network Solutions      43
Consulting                                   38
Staffing & Outsourcing                       36
Insurance Carriers                           28
Internet                                     27
Advertising & Marketing                      23
Health Care Services & Hospitals             21
Research & Development                       17
Federal Agencies                             16
Investment Banking & Asset Management        13
Banks & Credit Unions                         8
Lending                                       8
Energy                                        5
Consumer Products Manufacturing               5
Telecommunications Services                   5
Insurance Agencies & Brokerages               4
Food & Beverage Manufacturing                 4
Chemical Manufacturing                        3
Electrical & Electronic Manufacturing         3
Colleges & Universities                       3
Other Retail Stores                           3
Architectural & Engineering Services          3
Utilities                                     3
Real Estate                                   3
Miscellaneous Manufacturing                   3
Accounting                                    3
Wholesale                                     3
Industrial Manufacturing                      3
Video Games                                   3
Travel Agencies                               2
Express Delivery Services                     2
Timber Operations                             2
Financial Transaction Processing              2
Construction                                  2
Health, Beauty, & Fitness                     2
Transportation Equipment Manufacturing        2
Oil & Gas Services                            2
Venture Capital & Private Equity              2
Consumer Electronics & Appliances Stores      2
Department, Clothing, & Shoe Stores           1
Publishing                                    1
Social Assistance                             1
Farm Support Services                         1
Logistics & Supply Chain                      1
Transportation Management                     1
State & Regional Agencies                     1
```

```
Hotels, Motels, & Resorts                       1
Food & Beverage Stores                          1
News Outlet                                     1
Telecommunications Manufacturing                1
Cable, Internet & Telephone Providers           1
Shipping                                        1
Rail                                            1
Name: Industry, dtype: int64

Value counts in 'Sector':
Information Technology                 188
Business Services                      120
Biotech & Pharmaceuticals               66
Aerospace & Defense                     46
Finance                                 33
Insurance                               32
Manufacturing                           23
Health Care                             21
Government                              17
Oil, Gas, Energy & Utilities            10
Telecommunications                       7
Retail                                   7
Transportation & Logistics               6
Media                                    5
Real Estate                              3
Travel & Tourism                         3
Agriculture & Forestry                   3
Education                                3
Accounting & Legal                       3
Construction, Repair & Maintenance       2
Consumer Services                        2
Non-Profit                               1
Name: Sector, dtype: int64

Value counts in 'Revenue':
$100 to $500 million (USD)          94
$10+ billion (USD)                  63
$2 to $5 billion (USD)              45
$10 to $25 million (USD)            41
$1 to $2 billion (USD)              36
$25 to $50 million (USD)            36
$50 to $100 million (USD)           31
$1 to $5 million (USD)              31
$500 million to $1 billion (USD)    19
$5 to $10 million (USD)             14
Less than $1 million (USD)          14
$5 to $10 billion (USD)              8
Name: Revenue, dtype: int64

Value counts in 'Location':
San Francisco, CA    69
New York, NY         50
Washington, DC       26
Boston, MA           24
Chicago, IL          22
                     ..
Oshkosh, WI           1
Culver City, CA       1
Lake Oswego, OR       1
New Orleans, LA       1
Irwindale, CA         1
Name: Location, Length: 207, dtype: int64
```

OBSERVATION

1. Size of Companies:

- The majority of companies in the dataset have between 51 to 200 employees, with 135 companies falling into this category.
- Companies with 1001 to 5000 employees and 1 to 50 employees are also relatively common, with 104 and 86 companies, respectively.
- The least common size category is companies with 5001 to 10,000 employees, with only 61 companies falling into this group.

1. Type of Ownership:

- The most common type of ownership is "Company - Private," with 397 companies falling into this category.
- "Company - Public" is the second most common type of ownership, with 153 companies.
- Other types of ownership, such as "Nonprofit Organization" and "Subsidiary or Business Segment," are less common.

1. Industry:

- The dataset covers a wide range of industries, with the top three being Biotech & Pharmaceuticals (66 companies), IT Services (61 companies), and Computer Hardware & Software (57 companies).
- Many other industries are represented, including Aerospace & Defense, Enterprise Software & Network Solutions, and Consulting.

1. Sector:

- The most common sector is Information Technology, with 188 companies falling into this category.
- Other significant sectors include Business Services (120 companies), Biotech & Pharmaceuticals (66 companies), and Aerospace & Defense (46 companies).

1. Revenue:

- The dataset includes companies with a diverse range of revenue levels.
- The most common revenue range is 100 to 500 million (USD) with 94 companies.
- Some companies have very high revenue levels, such as 10+ billion (USD), while others have lower revenues, such as Less than 1 million (USD).

1. Location:

- The dataset includes companies from various locations, with San Francisco, CA having the highest representation (69 companies).
- Other notable locations include New York, NY (50 companies), Washington, DC (26 companies), and Boston, MA (24 companies).

Identifying the Numerical data types

In [41]:
```
#Get the data types of each column in the dataset
data_types = df.dtypes

#Identify columns with int64 or float64 data type
numerical_columns = df.select_dtypes(include=['float64', 'int64'])
```

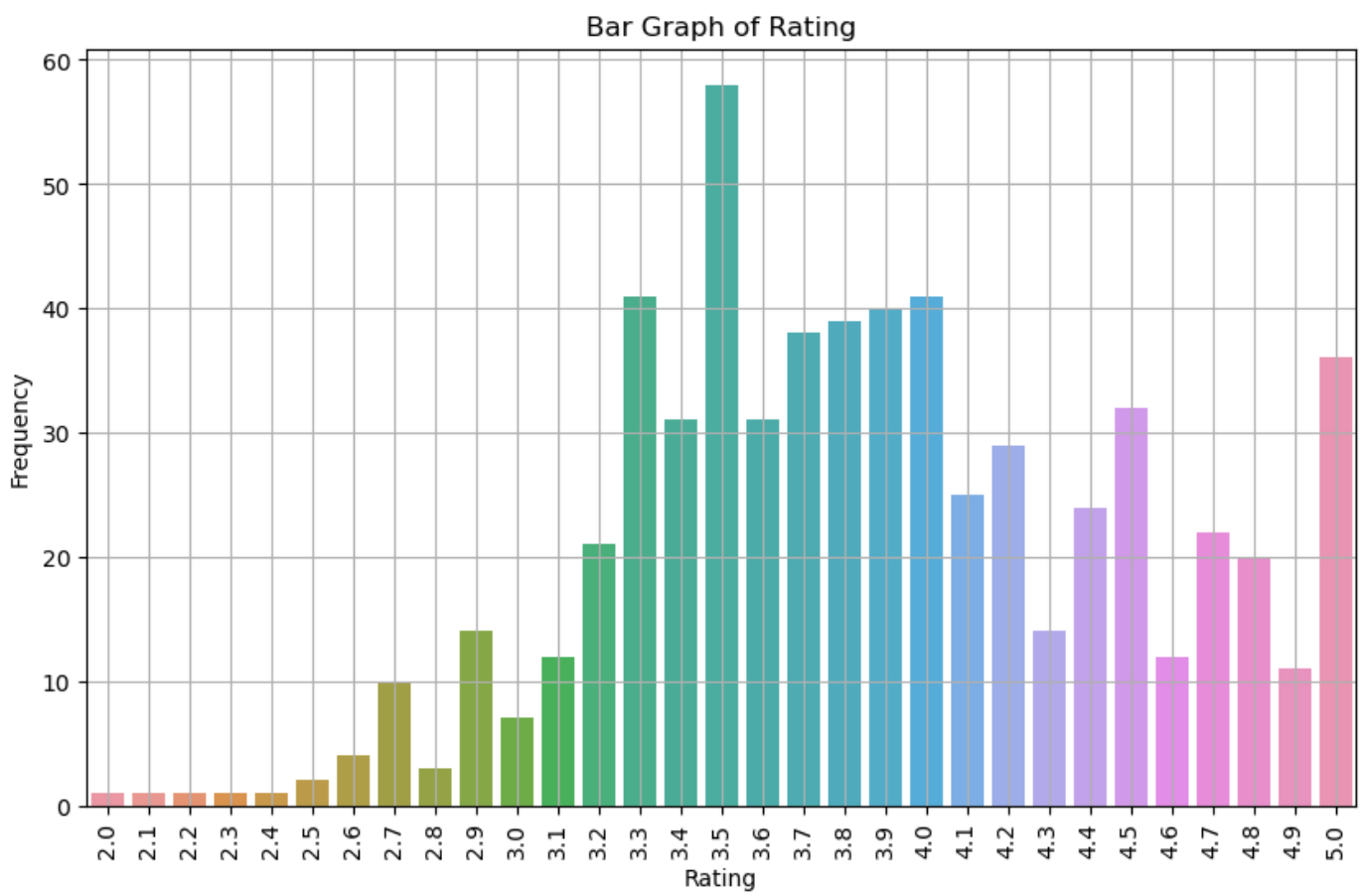
```
print("Numerical Columns:")
print(numerical_columns)
```

```
Numerical Columns:
     Rating  Founded  Low_Salary_in_dollar  High_Salary_in_dollar
0       3.1     1993                137000                 137000
1       4.2     1968                137000                 137000
2       3.8     1981                137000                 137000
3       3.5     2000                137000                 137000
4       2.9     1998                137000                 137000
..      ...      ...                   ...                    ...
667     3.6     1989                105000                 105000
668     NaN       -1                105000                 105000
669     NaN       -1                105000                 105000
670     5.0       -1                105000                 105000
671     2.7     1976                105000                 105000

[672 rows x 4 columns]
```

Finding skewness and kurtosis for Rating

In [53]:
```
#Calculate skewness and kurtosis for each numerical column
skewness = df['Rating'].skew()
kurtosis = df['Rating'].kurt()

#Count the frequency of each unique value in the column
value_counts = df['Rating'].value_counts()

#Plot the bar graph
plt.figure(figsize=(10, 6))
sns.barplot(x=value_counts.index, y=value_counts.values)
plt.xlabel('Rating')
plt.ylabel('Frequency')
plt.title(f'Bar Graph of Rating')
plt.xticks(rotation=90)
plt.grid(True)

#Calculate and print skewness and kurtosis
skewness = df['Rating'].skew()
kurtosis = df['Rating'].kurtosis()
print(f"Skewness: {skewness}")
print(f"Kurtosis: {kurtosis}")

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

```
Skewness: 0.018729142314406803
Kurtosis: -0.443772323456598
```



OBSERVATION

1. Skewness:

- The skewness value is approximately 0.0187, which is very close to 0. This suggests that the distribution of ratings is nearly symmetric, with a slight right (positive) skew. This means that while the majority of ratings are clustered around the mean, there may be some higher ratings that are pulling the distribution slightly to the right.

1. Kurtosis:

- The kurtosis value is approximately -0.44, which is less than 3. This indicates that the distribution of ratings is platykurtic, meaning it has thinner tails and is less peaked than a normal distribution.

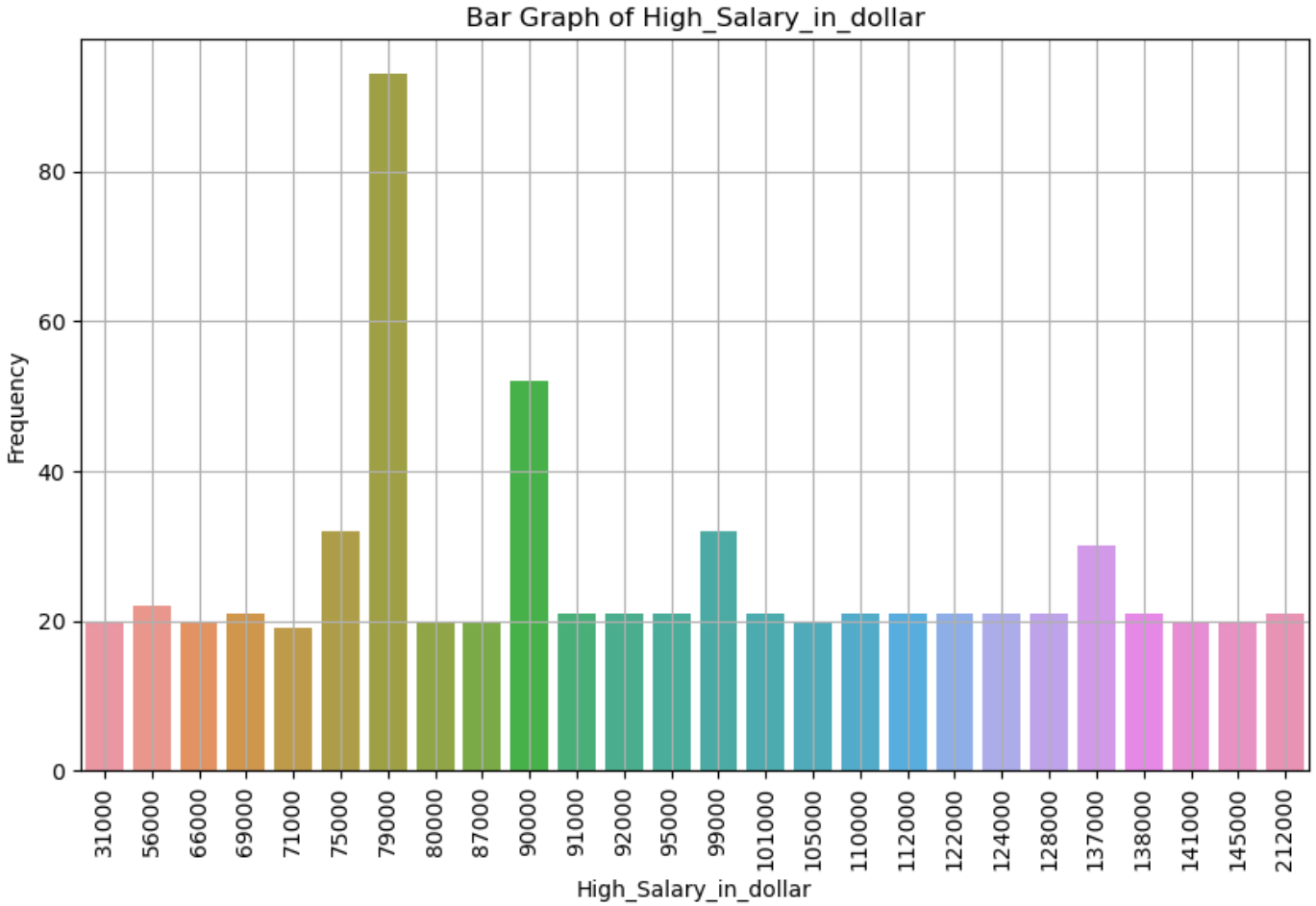
1. '3.5' is the rating by employees mostly.

Finding skewness and kurtosis for High_Salary_in_dollar(Upper limit of salary)

In [55]:
```
#Calculate skewness and kurtosis
skewness = df['High_Salary_in_dollar'].skew()
kurtosis = df['High_Salary_in_dollar'].kurt()

#Count the frequency of each unique value in the column
value_counts = df['High_Salary_in_dollar'].value_counts()

#Plot the bar graph
plt.figure(figsize=(10, 6))
sns.barplot(x=value_counts.index, y=value_counts.values)
plt.xlabel('High_Salary_in_dollar')
plt.ylabel('Frequency')
```

```
plt.title(f'Bar Graph of High_Salary_in_dollar')
plt.xticks(rotation=90)
plt.grid(True)

#Calculate and print skewness and kurtosis
skewness = df['High_Salary_in_dollar'].skew()
kurtosis = df['High_Salary_in_dollar'].kurtosis()
print(f"Skewness: {skewness}")
print(f"Kurtosis: {kurtosis}")

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

```
Skewness: 1.0891390200125406
Kurtosis: 2.4071969532297586
```



OBSERVATION

1. Skewness:

- Skewness value of approximately 1.09, indicating a moderate right (positive) skew in the distribution of salaries. This suggests that there may be a few companies with relatively high salaries that are causing the distribution to be skewed to the right.

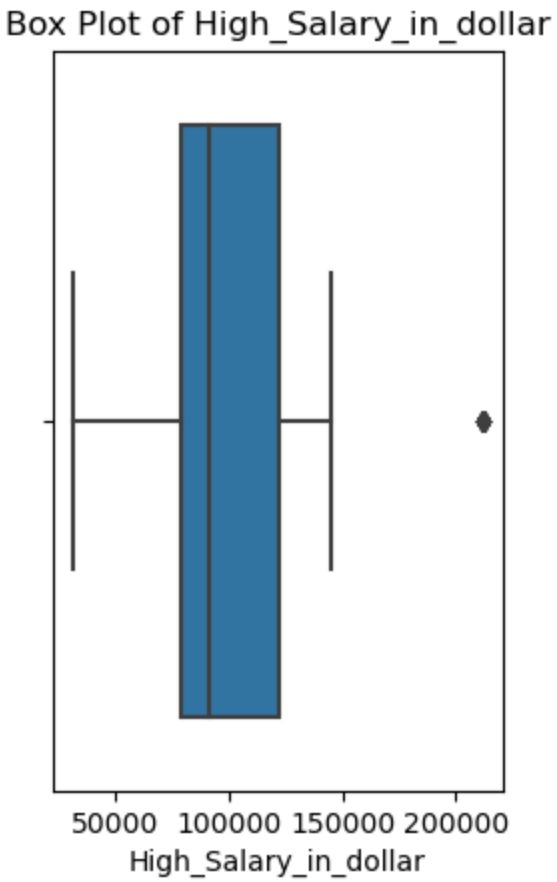
1. Kurtosis:

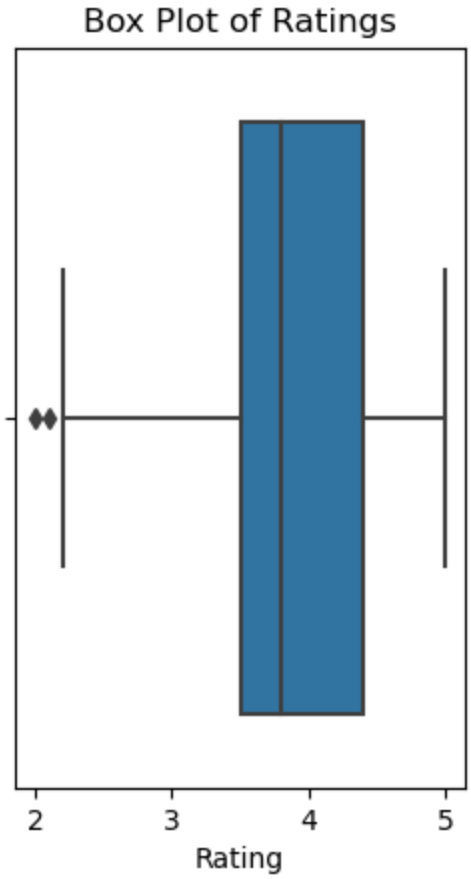
- Kurtosis value of approximately 2.41, which is greater than 3. This suggests that the distributions of salaries are leptokurtic, meaning they have heavier tails and are more peaked than a normal distribution. This indicates that there may be some outliers with very high salaries.

1. 79000 is the most common salary.

Finding Outliers for High_Salary_in_dollar(Upper limit of salary) and Rating

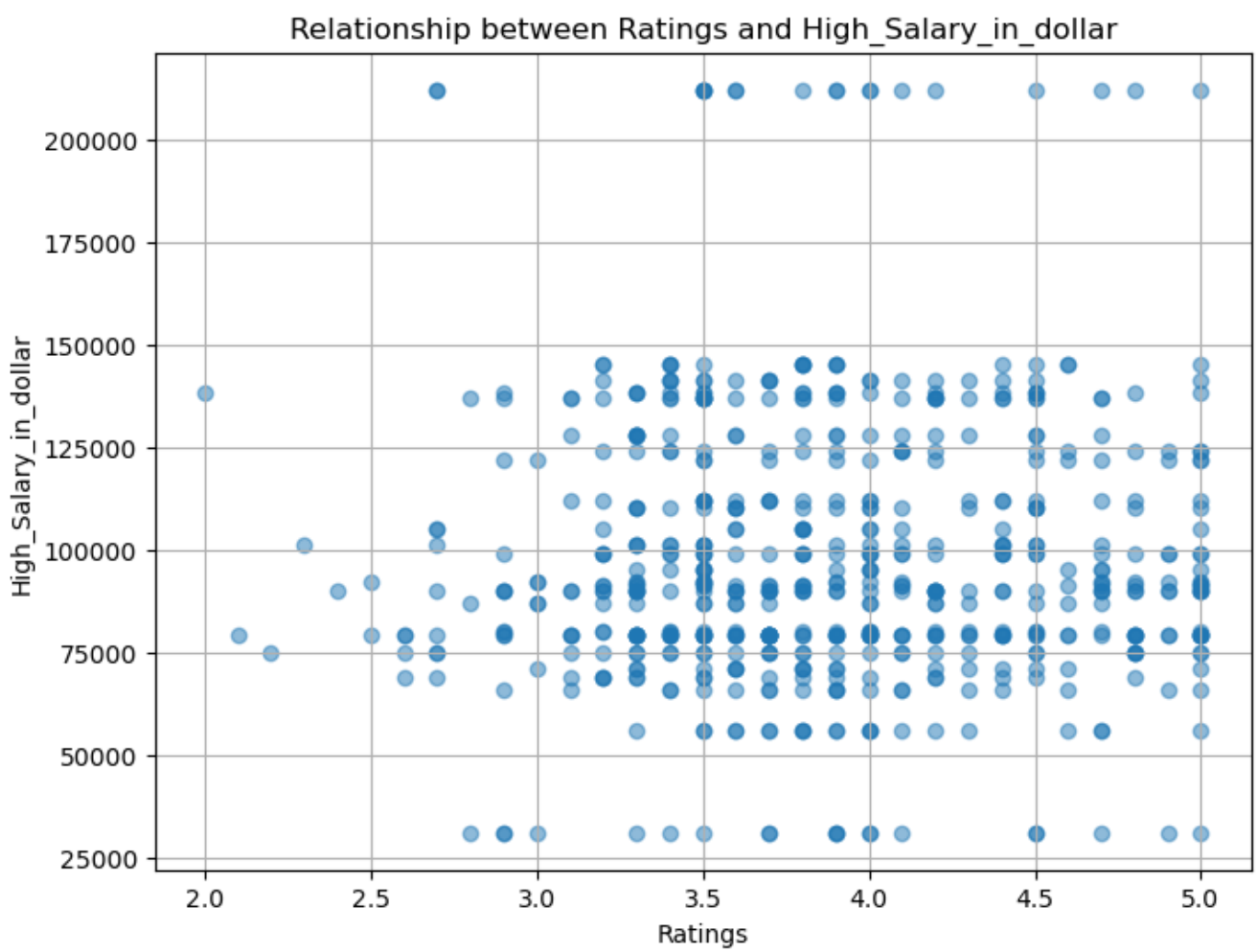
In [58]:
```
#Visualize the distribution of 'High-Salary-in-dollar' box plot
plt.subplot(1, 2, 2)
sns.boxplot(x=df['High_Salary_in_dollar'])
plt.title('Box Plot of High_Salary_in_dollar')
plt.xlabel('High_Salary_in_dollar')

plt.show()

#Visualize the distribution of 'Rating' using a histogram and box plot
plt.subplot(1, 2, 2)
sns.boxplot(x=df['Rating'])
plt.title('Box Plot of Ratings')
plt.xlabel('Rating')

plt.show()
```





Finding relationship between rating and salary paid

In [59]:
```
ratings = df['Rating']
salaries = df['High_Salary_in_dollar']

#Create a scatter plot to visualize the relationship between ratings and salaries
plt.figure(figsize=(8, 6))
plt.scatter(ratings, salaries, alpha=0.5)
plt.title('Relationship between Ratings and High_Salary_in_dollar')
plt.xlabel('Ratings')
plt.ylabel('High_Salary_in_dollar')
plt.grid(True)
plt.show()

#Calculate the correlation coefficient between ratings and salaries
correlation_coefficient = ratings.corr(salaries)
print("Correlation Coefficient:", correlation_coefficient)
```




```
Correlation Coefficient: 0.009875247968025304
```

OBSERVATION

a correlation coefficient of 0.0099 indicates a very weak positive relationship between the Ratings and High-Salary-in-dollar.

How does the salary vary across different locations?

In [61]:
```
salary_by_location = df.groupby('Location')['High_Salary_in_dollar'].agg(['mean', 'media

#Set the option to display all rows and columns
pd.set_option('display.max_rows', None)
pd.set_option('display.max_columns', None)

print("\nSalary Variation by Location:")
print(salary_by_location.to_string())
```

```
Salary Variation by Location:
                                    mean     median     min     max
Location
Adelphi, MD                 108500.000000  108500.0  105000  112000
Akron, OH                    75000.000000   75000.0   75000   75000
Alexandria, VA               85000.000000   87000.0   79000   87000
Alpharetta, GA               69000.000000   69000.0   69000   69000
Ann Arbor, MI               134500.000000  134500.0  124000  145000
Annapolis Junction, MD       82200.000000   80000.0   31000  122000
Appleton, WI                137000.000000  137000.0  137000  137000
Arlington, VA                86333.333333   87000.0   31000  141000
Ashburn, VA                  79000.000000   79000.0   79000   79000
Atlanta, GA                  91285.714286   87000.0   31000  145000
```

Aurora, CO	99000.000000	99000.0	99000	99000
Austin, TX	115000.000000	128000.0	79000	138000
Baltimore, MD	84800.000000	87000.0	71000	95000
Beavercreek, OH	71000.000000	71000.0	71000	71000
Bedford, MA	118000.000000	137000.0	79000	138000
Bellevue, WA	83250.000000	83000.0	75000	92000
Bethesda, MD	79000.000000	80500.0	56000	99000
Birmingham, AL	95500.000000	95500.0	90000	101000
Blue Bell, PA	101000.000000	101000.0	101000	101000
Boston, MA	97750.000000	90500.0	56000	212000
Bothell, WA	83000.000000	83000.0	75000	91000
Boulder, CO	69000.000000	69000.0	69000	69000
Bridgeport, WV	87000.000000	87000.0	87000	87000
Brisbane, CA	79000.000000	79000.0	79000	79000
Brooklyn, NY	124000.000000	124000.0	124000	124000
Burbank, CA	124000.000000	124000.0	124000	124000
Burlingame, CA	99000.000000	99000.0	99000	99000
California	80000.000000	80000.0	80000	80000
Cambridge, MA	97944.444444	90000.0	31000	212000
Carmel, IN	79000.000000	79000.0	79000	79000
Carpinteria, CA	90000.000000	90000.0	90000	90000
Carson, CA	75666.666667	79000.0	69000	79000
Cedar Rapids, IA	108333.333333	124000.0	56000	145000
Champaign, IL	71000.000000	71000.0	71000	71000
Chandler, AZ	100000.000000	100000.0	99000	101000
Chantilly, VA	100000.000000	99000.0	31000	137000
Charlotte, NC	71000.000000	71000.0	71000	71000
Chicago, IL	98136.363636	93500.0	31000	145000
Cincinnati, OH	91500.000000	90000.0	31000	145000
Cleveland, OH	90000.000000	90000.0	90000	90000
Collegeville, PA	80000.000000	80000.0	80000	80000
Colorado Springs, CO	31000.000000	31000.0	31000	31000
Columbia, MD	123000.000000	123000.0	122000	124000
Columbia, MO	101000.000000	101000.0	101000	101000
Columbia, SC	90000.000000	90000.0	90000	90000
Concord, CA	112000.000000	112000.0	79000	145000
Coral Gables, FL	79000.000000	79000.0	79000	79000
Culver City, CA	90000.000000	90000.0	90000	90000
Cupertino, CA	98000.000000	99000.0	90000	105000
Dallas, TX	141000.000000	141000.0	141000	141000
Danvers, MA	91000.000000	91000.0	91000	91000
Dayton, OH	155500.000000	155500.0	99000	212000
Dearborn, MI	90000.000000	90000.0	90000	90000
Denver, CO	98400.000000	105000.0	66000	112000
Durham, NC	92000.000000	92000.0	79000	105000
Edison, NJ	95333.333333	105000.0	69000	112000
Emeryville, CA	103500.000000	103500.0	79000	128000
Ewing, NJ	75000.000000	75000.0	75000	75000
Fairfax, VA	73000.000000	73000.0	66000	80000
Falls Church, VA	104800.000000	110000.0	69000	128000
Fort Belvoir, VA	81166.666667	79000.0	66000	105000
Fort Lee, NJ	105000.000000	105000.0	105000	105000
Fort Meade, MD	101000.000000	101000.0	101000	101000
Fort Sam Houston, TX	212000.000000	212000.0	212000	212000
Fort Wayne, IN	75000.000000	75000.0	75000	75000
Fort Worth, TX	96000.000000	87000.0	56000	145000
Foster City, CA	69000.000000	69000.0	69000	69000
Frederick, MD	69000.000000	69000.0	69000	69000
Fremont, CA	110000.000000	110000.0	110000	110000
Gahanna, OH	80000.000000	80000.0	80000	80000
Gaithersburg, MD	82428.571429	79000.0	56000	112000
Germantown, MD	79000.000000	79000.0	79000	79000
Hamilton, NJ	110000.000000	110000.0	110000	110000
Hampton, VA	110000.000000	110000.0	79000	141000
Harrisburg, PA	56000.000000	56000.0	56000	56000
Hartford, CT	83000.000000	71000.0	66000	112000

Hauppauge, NY	90000.000000	90000.0	90000	90000
Herndon, VA	111714.285714	99000.0	56000	212000
Hill AFB, UT	99000.000000	99000.0	99000	99000
Hoboken, NJ	90000.000000	90000.0	90000	90000
Holyoke, MA	90000.000000	90000.0	90000	90000
Houston, TX	92000.000000	92000.0	92000	92000
Huntsville, AL	124000.000000	124000.0	110000	138000
Indianapolis, IN	104500.000000	104500.0	99000	110000
Ipswich, MA	79000.000000	79000.0	79000	79000
Irvine, CA	91000.000000	91000.0	91000	91000
Irwindale, CA	105000.000000	105000.0	105000	105000
Jersey City, NJ	124000.000000	124000.0	124000	124000
Kansas City, MO	108000.000000	108000.0	92000	124000
Kent, OH	74000.000000	74000.0	69000	79000
Lake Oswego, OR	90000.000000	90000.0	90000	90000
Laurel, MD	77666.666667	80000.0	31000	122000
Lebanon, IN	90000.000000	90000.0	90000	90000
Lehi, UT	90000.000000	90000.0	90000	90000
Lexington Park, MD	158500.000000	158500.0	105000	212000
Lexington, MA	114000.000000	114000.0	91000	137000
Lincoln, NE	75000.000000	75000.0	75000	75000
Linthicum Heights, MD	79000.000000	79000.0	79000	79000
Livermore, CA	114500.000000	114500.0	91000	138000
Lombard, IL	79000.000000	79000.0	79000	79000
Longmont, CO	72500.000000	72500.0	66000	79000
Madison, WI	110000.000000	110000.0	110000	110000
Manchester, NH	124000.000000	124000.0	124000	124000
Maple Plain, MN	77333.333333	79000.0	31000	122000
Maryland Heights, MO	90000.000000	90000.0	90000	90000
McLean, VA	103583.333333	101000.0	66000	145000
Melbourne, FL	90000.000000	90000.0	90000	90000
Memphis, TN	83000.000000	56000.0	56000	137000
Menlo Park, CA	141000.000000	141000.0	141000	141000
Merrifield, VA	128666.666667	138000.0	110000	138000
Minneapolis, MN	80000.000000	80000.0	80000	80000
Monroe, WI	141000.000000	141000.0	141000	141000
Monterey, CA	141000.000000	141000.0	141000	141000
Mountain View, CA	69000.000000	69000.0	69000	69000
Naperville, IL	98500.000000	98500.0	92000	105000
Nashville, TN	92666.666667	99000.0	80000	99000
New Bedford, MA	138000.000000	138000.0	138000	138000
New Jersey	95500.000000	95500.0	92000	99000
New Orleans, LA	75000.000000	75000.0	75000	75000
New York, NY	107880.000000	93500.0	31000	212000
Newton, MA	103666.666667	95000.0	79000	137000
Norfolk, VA	71000.000000	71000.0	71000	71000
Northbrook, IL	114000.000000	114000.0	91000	137000
Norwell, MA	90000.000000	90000.0	90000	90000
Oak Ridge, TN	63500.000000	63500.0	56000	71000
Oakland, CA	84500.000000	84500.0	79000	90000
Oakville, CA	87000.000000	87000.0	87000	87000
Oklahoma City, OK	123750.000000	102000.0	79000	212000
Omaha, NE	83000.000000	83000.0	79000	87000
Open Fork, VA	79000.000000	79000.0	79000	79000
Orange, CA	87000.000000	87000.0	87000	87000
Orlando, FL	112000.000000	128000.0	71000	137000
Oshkosh, WI	141000.000000	141000.0	141000	141000
Overland Park, KS	75000.000000	75000.0	75000	75000
Oxnard, CA	124000.000000	124000.0	124000	124000
Palo Alto, CA	99800.000000	90000.0	75000	122000
Patuxent, Anne Arundel, MD	66000.000000	66000.0	66000	66000
Peoria, IL	79000.000000	79000.0	79000	79000
Philadelphia, PA	102333.333333	110000.0	69000	128000
Phoenix, AZ	130500.000000	130500.0	124000	137000
Piscataway, NJ	91000.000000	91000.0	91000	91000
Pittsburgh, PA	96000.000000	85000.0	69000	145000

Plano, TX	100000.000000	96000.0	71000	137000
Pleasanton, CA	212000.000000	212000.0	212000	212000
Portland, OR	128000.000000	128000.0	128000	128000
Portsmouth, NH	75000.000000	75000.0	75000	75000
Quantico, VA	105000.000000	105000.0	105000	105000
Raleigh, NC	147000.000000	133000.0	110000	212000
Rancho Cucamonga, CA	138000.000000	138000.0	138000	138000
Redmond, WA	105000.000000	105000.0	105000	105000
Redwood City, CA	99000.000000	101000.0	79000	128000
Remote	102666.666667	90000.0	75000	145000
Reston, VA	102750.000000	99000.0	79000	145000
Richland, WA	94666.666667	91000.0	56000	137000
Richmond, VA	143000.000000	143000.0	141000	145000
Rockville, MD	90500.000000	90500.0	90000	91000
Sacramento, CA	141000.000000	141000.0	141000	141000
Saint Louis, MO	86750.000000	79000.0	69000	137000
San Antonio, TX	66000.000000	66000.0	66000	66000
San Carlos, CA	83000.000000	67500.0	56000	141000
San Clemente, CA	87500.000000	87500.0	80000	95000
San Diego, CA	107571.428571	99000.0	90000	137000
San Francisco, CA	95594.202899	95000.0	31000	145000
San Jose, CA	83500.000000	90500.0	31000	122000
San Mateo, CA	107000.000000	107000.0	90000	124000
San Ramon, CA	124000.000000	124000.0	124000	124000
Santa Barbara, CA	105000.000000	99000.0	79000	137000
Santa Clara, CA	76666.666667	79000.0	31000	122000
Santa Cruz, CA	108500.000000	108500.0	79000	138000
Santa Monica, CA	108000.000000	108000.0	75000	141000
Schaumburg, IL	99000.000000	99000.0	99000	99000
Scotts Valley, CA	95500.000000	95500.0	79000	112000
Seattle, WA	144600.000000	112000.0	92000	212000
Sheboygan, WI	82500.000000	82500.0	75000	90000
Silver Spring, MD	66000.000000	66000.0	66000	66000
Simi Valley, CA	138000.000000	138000.0	138000	138000
Smithfield, RI	100500.000000	100500.0	56000	145000
South San Francisco, CA	92000.000000	87000.0	56000	138000
Southfield, MI	55500.000000	55500.0	31000	80000
Spartanburg, SC	79000.000000	79000.0	79000	79000
Springfield, VA	101500.000000	101500.0	79000	124000
Sterling, VA	69000.000000	69000.0	69000	69000
Sunnyvale, CA	100666.666667	112000.0	66000	124000
Texas	71000.000000	71000.0	71000	71000
The Woodlands, TX	91000.000000	91000.0	87000	95000
Thousand Oaks, CA	93800.000000	92000.0	66000	112000
Timonium, MD	99000.000000	99000.0	99000	99000
Trumbull, CT	69000.000000	69000.0	69000	69000
Tulsa, OK	55000.000000	55000.0	31000	79000
United States	93454.545455	79000.0	31000	212000
Utah	83000.000000	83000.0	79000	87000
Vicksburg, MS	101000.000000	101000.0	101000	101000
Vienna, VA	98000.000000	98000.0	91000	105000
Waltham, MA	73000.000000	73000.0	56000	90000
Washington, DC	111346.153846	100000.0	31000	212000
Washington, VA	87000.000000	87000.0	87000	87000
Wellesley, MA	87000.000000	87000.0	87000	87000
West Chester, PA	105000.000000	105000.0	105000	105000
West Grove, PA	137000.000000	137000.0	137000	137000
Wilmington, DE	212000.000000	212000.0	212000	212000
Winston-Salem, NC	124000.000000	124000.0	110000	138000
Winter Park, FL	89333.333333	90000.0	79000	99000
Winters, TX	124000.000000	124000.0	124000	124000
Woburn, MA	107666.666667	110000.0	75000	138000
Woodbridge, NJ	90000.000000	90000.0	90000	90000
Woodlawn, MD	145000.000000	145000.0	145000	145000
Yakima, WA	79000.000000	79000.0	79000	79000

OBSERVATION

1. Salary Range Variation:

- The salary range varies significantly across different locations.
- For example, salaries in "San Francisco, CA" have a wide range, from 31,000 to 145,000, indicating a high level of income disparity within the city. Conversely, some locations like "Akron, OH" and "Columbia, SC" have a consistent salary of 75,0000.

1. High-Paying Locations:

- Some locations stand out as having higher average and maximum salaries.
- Locations like "Palo Alto, CA," "Mountain View, CA," and "Menlo Park, CA" have high median salaries, suggesting that they are tech hubs with well-paying jobs. Similarly, "Wilmington, DE" has the highest maximum salary of 212,000.

1. Low-Paying Locations:

- Locations such as "Colorado Springs, CO," "San Antonio, TX," and "Tulsa, OK" have relatively low salaries, with minimum salaries of 31,000, 66,000, and 31,000, respectively.

1. Mid-Range Salaries:

- Many locations have median salaries in the 70,000 to 100,000 range. This includes cities like "Chicago, IL," "Dallas, TX," and "Seattle, WA."

1. Outliers:

- Some locations have outliers with significantly higher salaries, such as "Fort Sam Houston, TX" with a maximum salary of 212,000 and "$NewYork,NY$"$with\ a\ maximum\ salary\ of$ 212,000. These outliers could be due to specific high-paying industries or roles.

1. Regional Differences:

- There are noticeable variations in salaries within the same state or region. For example, in Virginia, you have "McLean, VA" with a median of 101,000 and "$Arlington,VA$" $with\ a\ median\ of$ 87,000. This suggests that factors like proximity to major cities or industries can impact salaries.

1. Uniform Salaries:

- Some locations have uniform salaries with no variation. For instance, "Fort Belvoir, VA," "Chantilly, VA," and "Foster City, CA" all have the same salary for mean, median, minimum, and maximum.

1. Lack of Data:

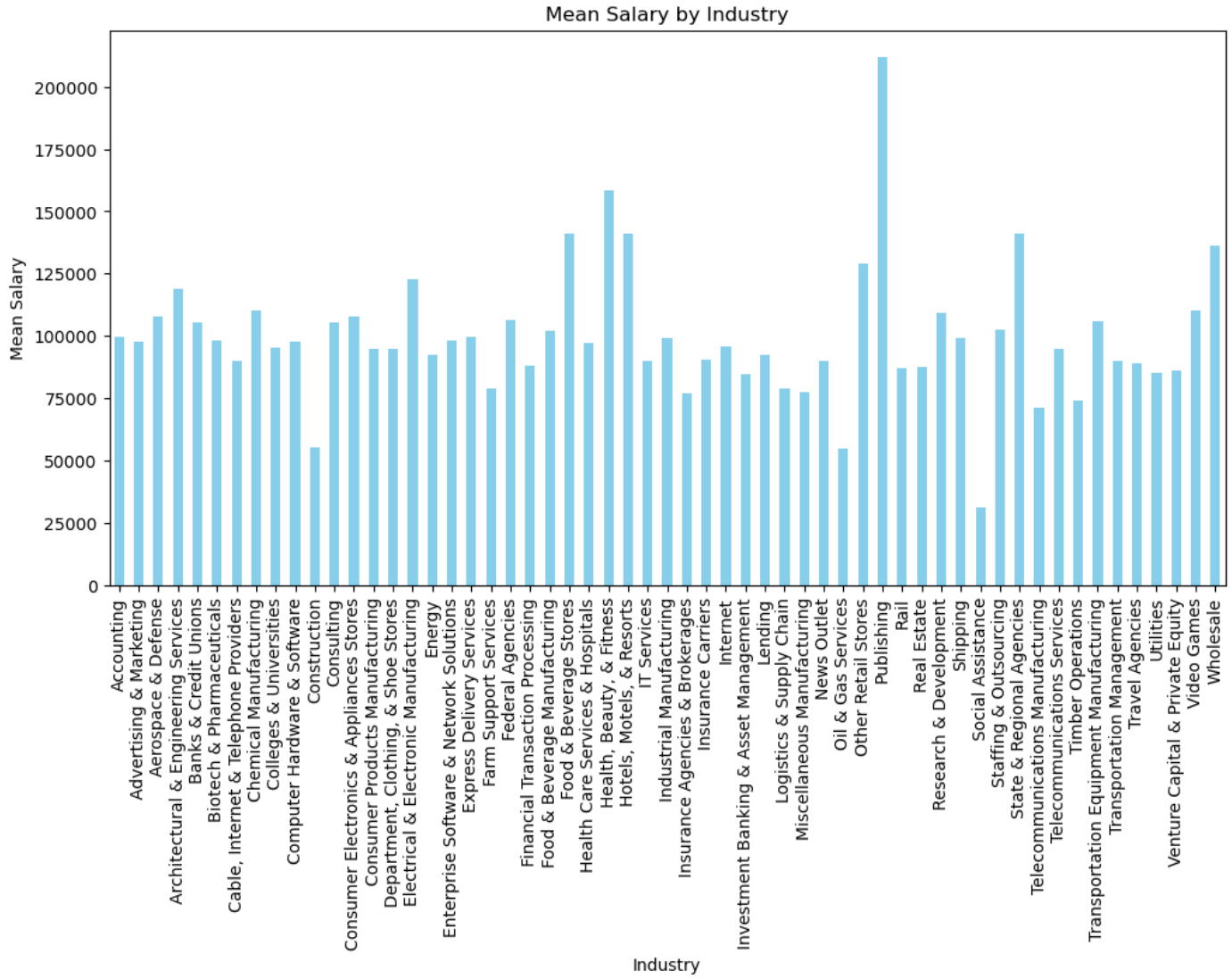
- Some locations have limited data points, which could affect the accuracy of the statistics. For instance, "California" and "Remote" each have only one data point.

1. Influence of Industry:

- It's clear that the type of industries and job markets in each location play a significant role in determining salary levels. Tech-heavy areas tend to have higher salaries, while other areas may have lower salaries due to different economic factors.

Finding salary with respect to the industry

In [63]:
```
#Group the data by 'Industry' and calculate summary statistics of salary for each indust
salary_by_industry = df.groupby('Industry')['High_Salary_in_dollar'].agg(['mean'])

#Create a bar chart for mean salary by industry
plt.figure(figsize=(12, 6))  # Adjust the figure size as needed
salary_by_industry['mean'].plot(kind='bar', color='skyblue')
plt.xlabel('Industry')
plt.ylabel('Mean Salary')
plt.title('Mean Salary by Industry')
plt.xticks(rotation=90)  # Rotate x-axis labels for better readability
plt.show()
```



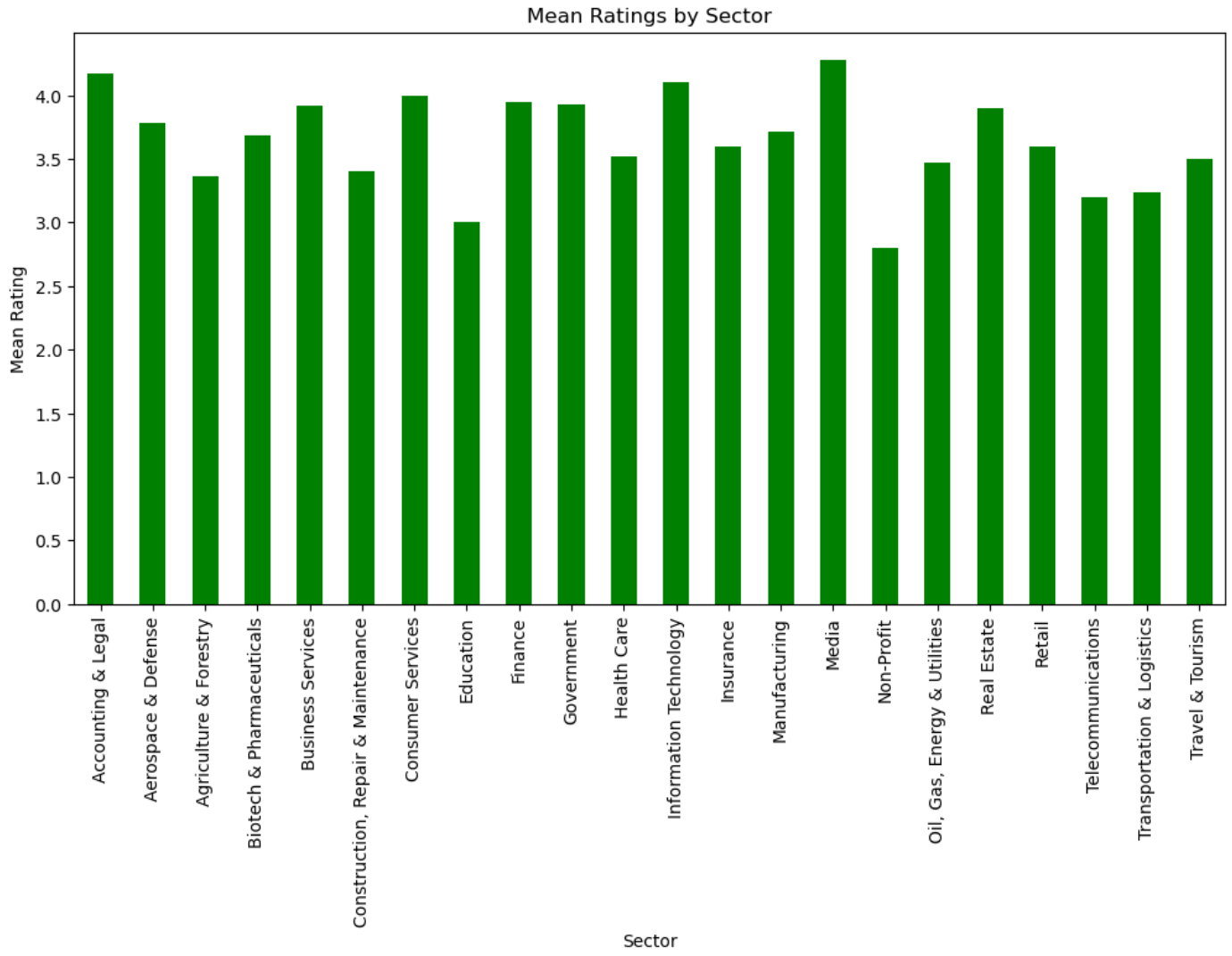
OBSERVATION

Industry plays a crucial role in determining salary levels. Certain industries, such as "Health, Beauty, & Fitness" and "Publishing," have higher mean and median salaries compared to industries like "Social Assistance" and "Oil & Gas Services."

How do ratings vary between different sectors?

In [65]:
```
#Group the data by 'Sector' and calculate summary statistics of ratings for each sector
ratings_by_sector = df.groupby('Sector')['Rating'].agg(['mean', 'median', 'min', 'max'])

#Create a bar chart for mean ratings by sector
plt.figure(figsize=(12, 6))  # Adjust the figure size as needed
```

```
ratings_by_sector['mean'].plot(kind='bar', color='green')
plt.xlabel('Sector')
plt.ylabel('Mean Rating')
plt.title('Mean Ratings by Sector')
plt.xticks(rotation=90)  # Rotate x-axis labels for better readability
plt.show()
```

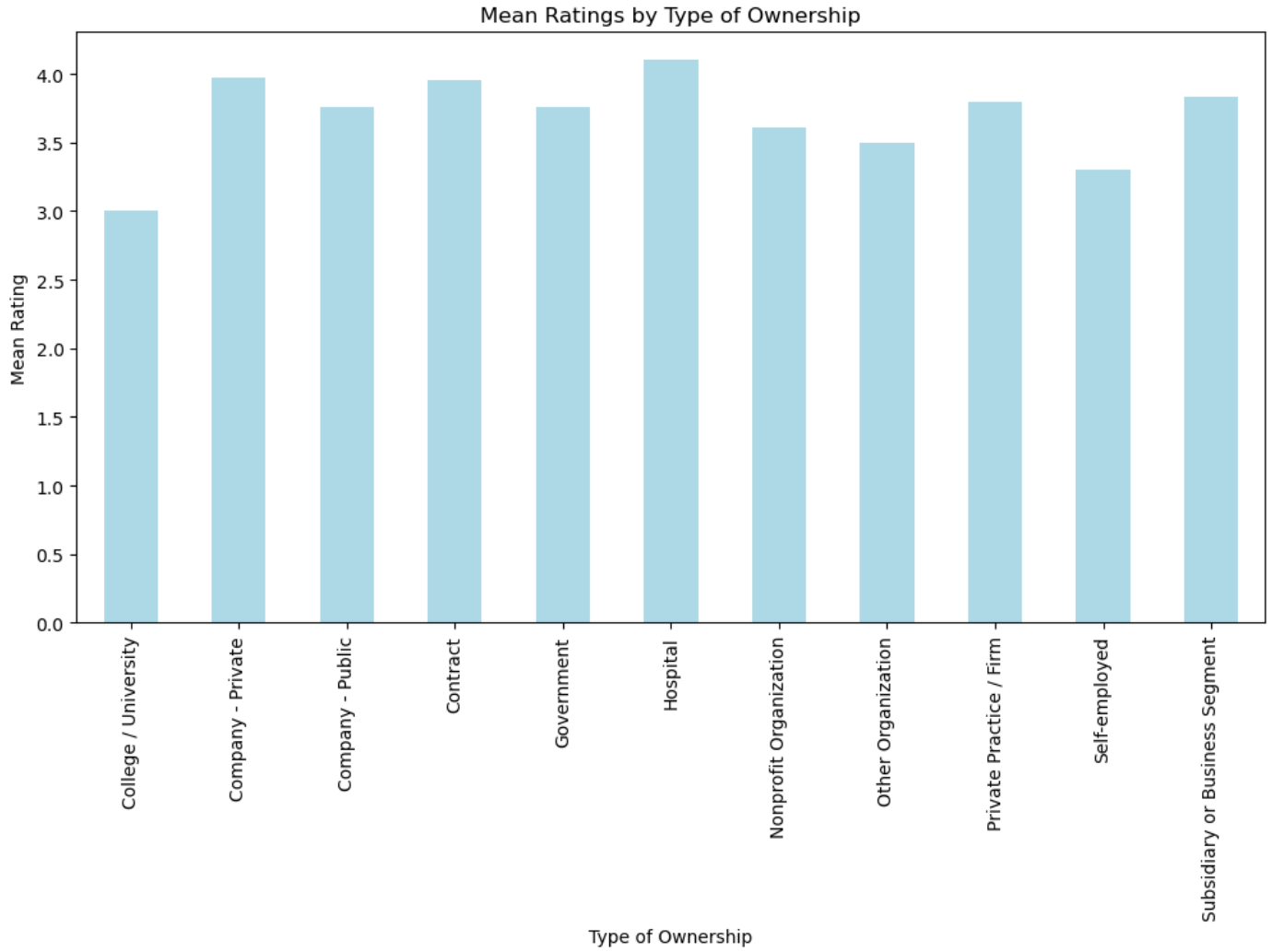


OBSERVATION

- The Media sector tends to have the highest average ratings, indicating generally positive employee sentiment.
- Employees in the Education sector and Non-Profit organizations tend to give lower ratings on average compared to other sectors and ownership types

Finding relationship rating and type of ownership

In [66]:
```
#Group the data by 'Type of ownership' and calculate summary statistics of ratings for e
ratings_by_ownership = df.groupby('Type of ownership')['Rating'].agg(['mean', 'median',

#Create a bar chart for mean ratings by type of ownership
plt.figure(figsize=(12, 6))  # Adjust the figure size as needed
ratings_by_ownership['mean'].plot(kind='bar', color='lightblue')
plt.xlabel('Type of Ownership')
plt.ylabel('Mean Rating')
plt.title('Mean Ratings by Type of Ownership')
plt.xticks(rotation=90)  # Rotate x-axis labels for better readability
plt.show()
```



OBSERVATION

- Hospital ownership types having the highest average ratings.

Which companies have the most job postings in the dataset?

In [67]:
```
#Perform a frequency count of each unique company name
company_job_postings_count = df['Company Name'].value_counts()

#Get the companies with the highest job posting counts
companies_with_most_job_postings = company_job_postings_count.head()

print("Companies with the Most Job Postings:")
print(companies_with_most_job_postings)
```

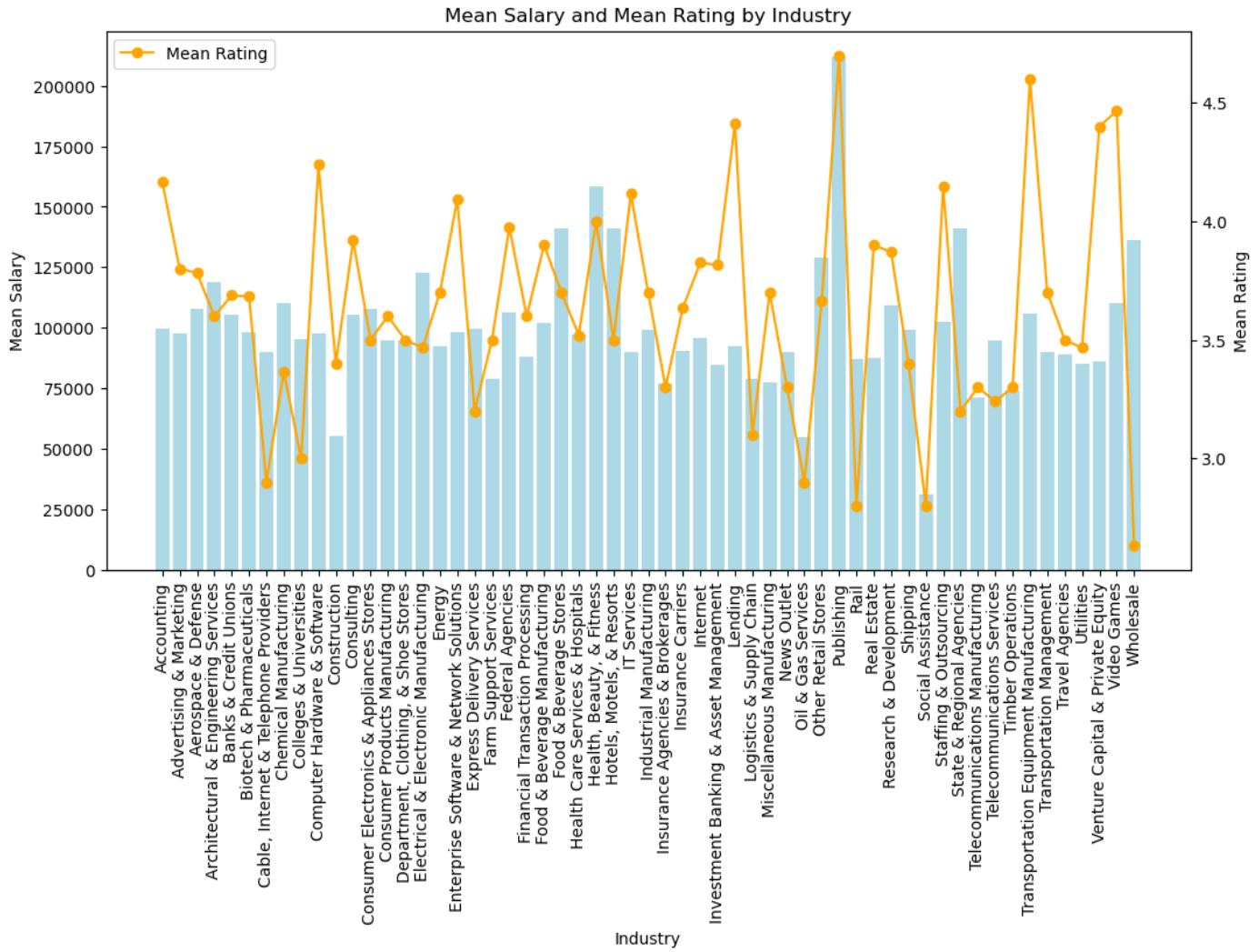
```
Companies with the Most Job Postings:
Hatch Data Inc        12
Maxar Technologies    12
Tempus Labs           11
AstraZeneca           10
Klaviyo                8
Name: Company Name, dtype: int64
```

OBSERVATION

The companies listed with the most job postings are actively recruiting and may present various job opportunities for individuals seeking employment. Job seekers interested in these companies should explore the specific job listings to identify positions that align with their skills and career goals.

Compare Industry with respect to salary and rating

In [71]:
```
#Group the data by 'Industry' and calculate metrics within each group
grouped_data = df.groupby('Industry')
metrics_within_groups = grouped_data.agg({
    'High_Salary_in_dollar': 'mean',
    'Rating': 'mean'
    })

#Create a bar chart to visualize the mean salary and mean rating within each industry
plt.figure(figsize=(12, 6))

#Plot mean salary
plt.bar(metrics_within_groups.index, metrics_within_groups['High_Salary_in_dollar'], col
plt.xlabel('Industry')
plt.ylabel('Mean Salary')
plt.xticks(rotation=90)

#Create a secondary y-axis for mean rating
plt.twinx()
plt.plot(metrics_within_groups.index, metrics_within_groups['Rating'], marker='o', color
plt.ylabel('Mean Rating')

plt.title('Mean Salary and Mean Rating by Industry')
plt.legend(loc='upper left')
plt.show()
```



OBSERVATION

1. Industries that offer the highest average salaries include Publishing, Health, Beauty, & Fitness, Hotels, Motels, & Resorts, Food & Beverage Stores, and State & Regional Agencies.

2. Industries with the highest average ratings from employees include Health, Beauty, & Fitness, Publishing, Video Games, Staffing & Outsourcing, Computer Hardware & Software, and Lending.