Topic- EDA

Preprocessing:

Import the libraries

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

In [2]:
```
# Import the data to csv file
data=pd.read_csv("C:/Users/PWORLD/Downloads/myexcel - myexcel.csv.csv")
data
```

Out[2]:

	Name	Team	Number	Position	Age	Height	Weight	College	Salary
0	Avery Bradley	Boston Celtics	0	PG	25	06-Feb	180	Texas	7730337.0
1	Jae Crowder	Boston Celtics	99	SF	25	06-Jun	235	Marquette	6796117.0
2	John Holland	Boston Celtics	30	SG	27	06-May	205	Boston University	NaN
3	R.J. Hunter	Boston Celtics	28	SG	22	06-May	185	Georgia State	1148640.0
4	Jonas Jerebko	Boston Celtics	8	PF	29	06-Oct	231	NaN	5000000.0
...	...	...	...	...	...	...	...	...	...
453	Shelvin Mack	Utah Jazz	8	PG	26	06-Mar	203	Butler	2433333.0
454	Raul Neto	Utah Jazz	25	PG	24	06-Jan	179	NaN	900000.0
455	Tibor Pleiss	Utah Jazz	21	C	26	07-Mar	256	NaN	2900000.0
456	Jeff Withey	Utah Jazz	24	C	26	7-0	231	Kansas	947276.0
457	Priyanka	Utah Jazz	34	C	25	07-Mar	231	Kansas	947276.0

458 rows × 9 columns

Cleaning the data.

In [4]:
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 458 entries, 0 to 457
Data columns (total 9 columns):
 #   Column    Non-Null Count  Dtype  
---  ------    --------------  -----  
 0   Name      458 non-null    object 
 1   Team      458 non-null    object 
 2   Number    458 non-null    int64  
 3   Position  458 non-null    object 
 4   Age       458 non-null    int64  
 5   Height    458 non-null    object 
 6   Weight    458 non-null    int64  
 7   College   374 non-null    object 
 8   Salary    447 non-null    float64
dtypes: float64(1), int64(3), object(5)
memory usage: 32.3+ KB
```

In [13]:
```
data.isnull()
```

Out[13]:

	Name	Team	Number	Position	Age	Height	Weight	College	Salary
0	False	False	False	False	False	False	False	False	False
1	False	False	False	False	False	False	False	False	False
2	False	False	False	False	False	False	False	False	True
3	False	False	False	False	False	False	False	False	False
4	False	False	False	False	False	False	False	True	False
...	...	...	...	...	...	...	...	...	...
453	False	False	False	False	False	False	False	False	False
454	False	False	False	False	False	False	False	True	False
455	False	False	False	False	False	False	False	True	False
456	False	False	False	False	False	False	False	False	False
457	False	False	False	False	False	False	False	False	False

458 rows × 9 columns

In [14]:
```
data.isnull().sum()
```

Out[14]:
```
Name         0
Team         0
Number       0
Position     0
Age          0
Height       0
Weight       0
College     84
Salary      11
dtype: int64
```

In [16]:
```
data = data.dropna()
data
```

Out[16]:

	Name	Team	Number	Position	Age	Height	Weight	College	Salary
0	Avery Bradley	Boston Celtics	0	PG	25	06-Feb	180	Texas	7730337.0
1	Jae Crowder	Boston Celtics	99	SF	25	06-Jun	235	Marquette	6796117.0
3	R.J. Hunter	Boston Celtics	28	SG	22	06-May	185	Georgia State	1148640.0
6	Jordan Mickey	Boston Celtics	55	PF	21	06-Aug	235	LSU	1170960.0
7	Kelly Olynyk	Boston Celtics	41	C	25	7-0	238	Gonzaga	2165160.0
...	...	...	...	...	...	...	...	...	...
451	Chris Johnson	Utah Jazz	23	SF	26	06-Jun	206	Dayton	981348.0
452	Trey Lyles	Utah Jazz	41	PF	20	06-Oct	234	Kentucky	2239800.0
453	Shelvin Mack	Utah Jazz	8	PG	26	06-Mar	203	Butler	2433333.0
456	Jeff Withey	Utah Jazz	24	C	26	7-0	231	Kansas	947276.0
457	Priyanka	Utah Jazz	34	C	25	07-Mar	231	Kansas	947276.0

365 rows × 9 columns

In [18]:
```
data.fillna(0)
data
```

Out[18]:

	Name	Team	Number	Position	Age	Height	Weight	College	Salary
0	Avery Bradley	Boston Celtics	0	PG	25	06-Feb	180	Texas	7730337.0
1	Jae Crowder	Boston Celtics	99	SF	25	06-Jun	235	Marquette	6796117.0
3	R.J. Hunter	Boston Celtics	28	SG	22	06-May	185	Georgia State	1148640.0
6	Jordan Mickey	Boston Celtics	55	PF	21	06-Aug	235	LSU	1170960.0
7	Kelly Olynyk	Boston Celtics	41	C	25	7-0	238	Gonzaga	2165160.0
...	...	...	...	...	...	...	...	...	...
451	Chris Johnson	Utah Jazz	23	SF	26	06-Jun	206	Dayton	981348.0
452	Trey Lyles	Utah Jazz	41	PF	20	06-Oct	234	Kentucky	2239800.0
453	Shelvin Mack	Utah Jazz	8	PG	26	06-Mar	203	Butler	2433333.0
456	Jeff Withey	Utah Jazz	24	C	26	7-0	231	Kansas	947276.0
457	Priyanka	Utah Jazz	34	C	25	07-Mar	231	Kansas	947276.0

365 rows × 9 columns

In [19]:
```
data.duplicated()
```

Out[19]:
```
0      False
1      False
3      False
6      False
7      False
       ...
451    False
452    False
453    False
456    False
457    False
Length: 365, dtype: bool
```

In [3]:
```
df=data.drop_duplicates()
df
```

Out[3]:

	Name	Team	Number	Position	Age	Height	Weight	College	Salary
0	Avery Bradley	Boston Celtics	0	PG	25	06-Feb	180	Texas	7730337.0
1	Jae Crowder	Boston Celtics	99	SF	25	06-Jun	235	Marquette	6796117.0
2	John Holland	Boston Celtics	30	SG	27	06-May	205	Boston University	NaN
3	R.J. Hunter	Boston Celtics	28	SG	22	06-May	185	Georgia State	1148640.0
4	Jonas Jerebko	Boston Celtics	8	PF	29	06-Oct	231	NaN	5000000.0
...	...	...	...	...	...	...	...	...	...
453	Shelvin Mack	Utah Jazz	8	PG	26	06-Mar	203	Butler	2433333.0
454	Raul Neto	Utah Jazz	25	PG	24	06-Jan	179	NaN	900000.0
455	Tibor Pleiss	Utah Jazz	21	C	26	07-Mar	256	NaN	2900000.0
456	Jeff Withey	Utah Jazz	24	C	26	7-0	231	Kansas	947276.0
457	Priyanka	Utah Jazz	34	C	25	07-Mar	231	Kansas	947276.0

458 rows × 9 columns

Correct the data in the "height" column by replacing it with random numbers between 150 and 180.

In [8]:
```
# Generate random heights between 150 and 180
random_heights = np.random.randint(150, 181, size=len(df))

# Replace the existing 'height' column with the random values
df['Height'] = random_heights

df['Height']
```

Out[8]:
```
0      161
1      174
2      150
3      154
4      163
      ...
453    155
454    158
455    154
456    178
457    170
Name: Height, Length: 458, dtype: int32
```

In [4]:
```
# check the updated dataframe
df.head(20)
```

Out[4]:

	Name	Team	Number	Position	Age	Height	Weight	College	Salary
0	Avery Bradley	Boston Celtics	0	PG	25	06-Feb	180	Texas	7730337.0
1	Jae Crowder	Boston Celtics	99	SF	25	06-Jun	235	Marquette	6796117.0
2	John Holland	Boston Celtics	30	SG	27	06-May	205	Boston University	NaN
3	R.J. Hunter	Boston Celtics	28	SG	22	06-May	185	Georgia State	1148640.0
4	Jonas Jerebko	Boston Celtics	8	PF	29	06-Oct	231	NaN	5000000.0
5	Amir Johnson	Boston Celtics	90	PF	29	06-Sep	240	NaN	12000000.0
6	Jordan Mickey	Boston Celtics	55	PF	21	06-Aug	235	LSU	1170960.0
7	Kelly Olynyk	Boston Celtics	41	C	25	7-0	238	Gonzaga	2165160.0
8	Terry Rozier	Boston Celtics	12	PG	22	06-Feb	190	Louisville	1824360.0
9	Marcus Smart	Boston Celtics	36	PG	22	06-Apr	220	Oklahoma State	3431040.0
10	Jared Sullinger	Boston Celtics	7	C	24	06-Sep	260	Ohio State	2569260.0
11	Isaiah Thomas	Boston Celtics	4	PG	27	05-Sep	185	Washington	6912869.0
12	Evan Turner	Boston Celtics	11	SG	27	06-Jul	220	Ohio State	3425510.0
13	James Young	Boston Celtics	13	SG	20	06-Jun	215	Kentucky	1749840.0
14	Tyler Zeller	Boston Celtics	44	C	26	7-0	253	North Carolina	2616975.0
15	Bojan Bogdanovic	Brooklyn Nets	44	SG	27	06-Aug	216	NaN	3425510.0
16	Markel Brown	Brooklyn Nets	22	SG	24	06-Mar	190	Oklahoma State	845059.0
17	Wayne Ellington	Brooklyn Nets	21	SG	28	06-Apr	200	North Carolina	1500000.0
18	Rondae Hollis-Jefferson	Brooklyn Nets	24	SG	21	06-Jul	220	Arizona	1335480.0
19	Jarrett Jack	Brooklyn Nets	2	PG	32	06-Mar	200	Georgia Tech	6300000.0

1. Determine the distribution of employees across each team and calculate the percentage split relative to the total number of employees.

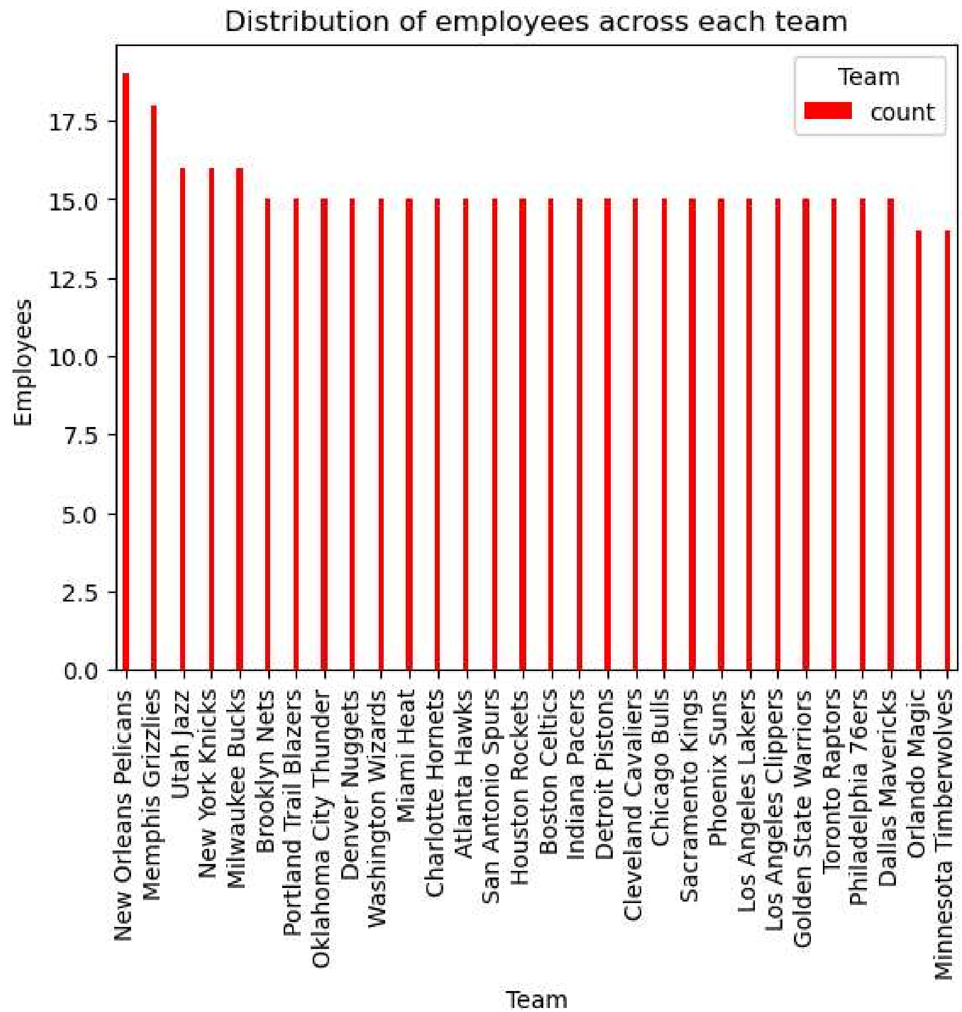
In [10]:
```
# Calculate the number of employees in each team
team_counts = df['Team'].value_counts()
print("Distribution of employees across each team:")
print(team_counts)
```

```
Distribution of employees across each team:
Team
New Orleans Pelicans      19
Memphis Grizzlies         18
Utah Jazz                 16
New York Knicks           16
Milwaukee Bucks           16
Brooklyn Nets             15
Portland Trail Blazers    15
Oklahoma City Thunder     15
Denver Nuggets            15
Washington Wizards        15
Miami Heat                15
Charlotte Hornets         15
Atlanta Hawks             15
San Antonio Spurs         15
Houston Rockets           15
Boston Celtics            15
Indiana Pacers            15
Detroit Pistons           15
Cleveland Cavaliers       15
Chicago Bulls             15
Sacramento Kings          15
Phoenix Suns              15
Los Angeles Lakers        15
Los Angeles Clippers      15
Golden State Warriors     15
Toronto Raptors           15
Philadelphia 76ers        15
Dallas Mavericks          15
Orlando Magic             14
Minnesota Timberwolves    14
Name: count, dtype: int64
```

In [12]:
```
# Calculate the percentage split relative to the total number of employees
total_employees = len(df)
team_percentages = (team_counts / total_employees) * 100
print("\nPercentage split relative to total number of employees:")
print(team_percentages)
```

```
Percentage split relative to total number of employees:
Team
New Orleans Pelicans      4.148472
Memphis Grizzlies         3.930131
Utah Jazz                 3.493450
New York Knicks           3.493450
Milwaukee Bucks           3.493450
Brooklyn Nets             3.275109
Portland Trail Blazers    3.275109
Oklahoma City Thunder     3.275109
Denver Nuggets            3.275109
Washington Wizards        3.275109
Miami Heat                3.275109
Charlotte Hornets         3.275109
Atlanta Hawks             3.275109
San Antonio Spurs         3.275109
Houston Rockets           3.275109
Boston Celtics            3.275109
Indiana Pacers            3.275109
Detroit Pistons           3.275109
Cleveland Cavaliers       3.275109
Chicago Bulls             3.275109
Sacramento Kings          3.275109
Phoenix Suns              3.275109
Los Angeles Lakers        3.275109
Los Angeles Clippers      3.275109
Golden State Warriors     3.275109
Toronto Raptors           3.275109
Philadelphia 76ers        3.275109
Dallas Mavericks          3.275109
Orlando Magic             3.056769
Minnesota Timberwolves    3.056769
Name: count, dtype: float64
```

In [16]:
```
# plot a bar diagram for Distribution of employees across each team:

team_counts.plot(kind='bar',color=[ 'red'],width=0.2)
plt.xlabel('Team')
plt.ylabel('Employees ')
plt.title('Distribution of employees across each team')
plt.legend(title='Team')
plt.show()
```

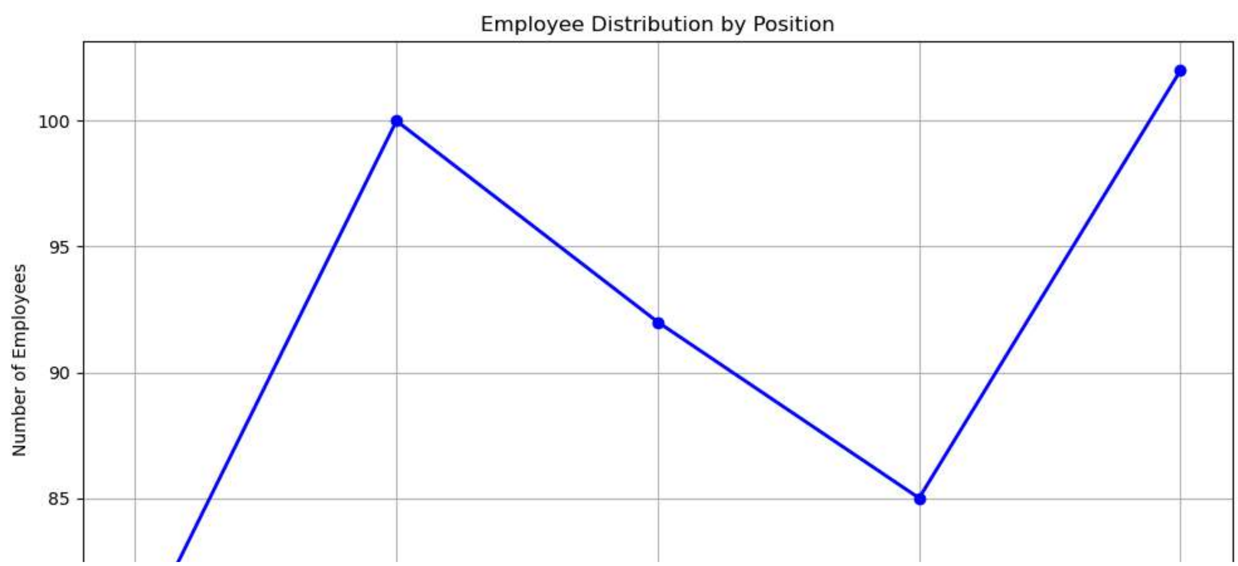


2. Segregate employees based on their positions within the company.

In [5]:
```
# Group employees by their positions
position_groups = df.groupby('Position')['Name'].apply(list)

print(position_groups)
```

```
Position
C     [Kelly Olynyk, Jared Sullinger, Tyler Zeller, ...
PF    [Jonas Jerebko, Amir Johnson, Jordan Mickey, C...
PG    [Avery Bradley, Terry Rozier, Marcus Smart, Is...
SF    [Jae Crowder, Thanasis Antetokounmpo, Carmelo ...
SG    [John Holland, R.J. Hunter, Evan Turner, James...
Name: Name, dtype: object
```

In [36]:
```
import matplotlib.pyplot as plt
import pandas as pd



# Aggregate data to count employees by position
position_counts = df['Position'].value_counts()

# Extract labels (positions) and values (employee counts)
labels = position_counts.index.tolist()
values = position_counts.tolist()

# Sort labels and values to ensure proper order in the line plot
labels, values = zip(*sorted(zip(labels, values)))

# Plotting the line diagram
plt.figure(figsize=(10, 6))
plt.plot(labels, values, marker='o', linestyle='-', color='b', linewidth=2)

# Adding labels and title
plt.xlabel('Position')
plt.ylabel('Number of Employees')
plt.title('Employee Distribution by Position')

# Display the plot
plt.grid(True)
plt.tight_layout()
plt.show()
```



3. Identify the predominant age group among employees.

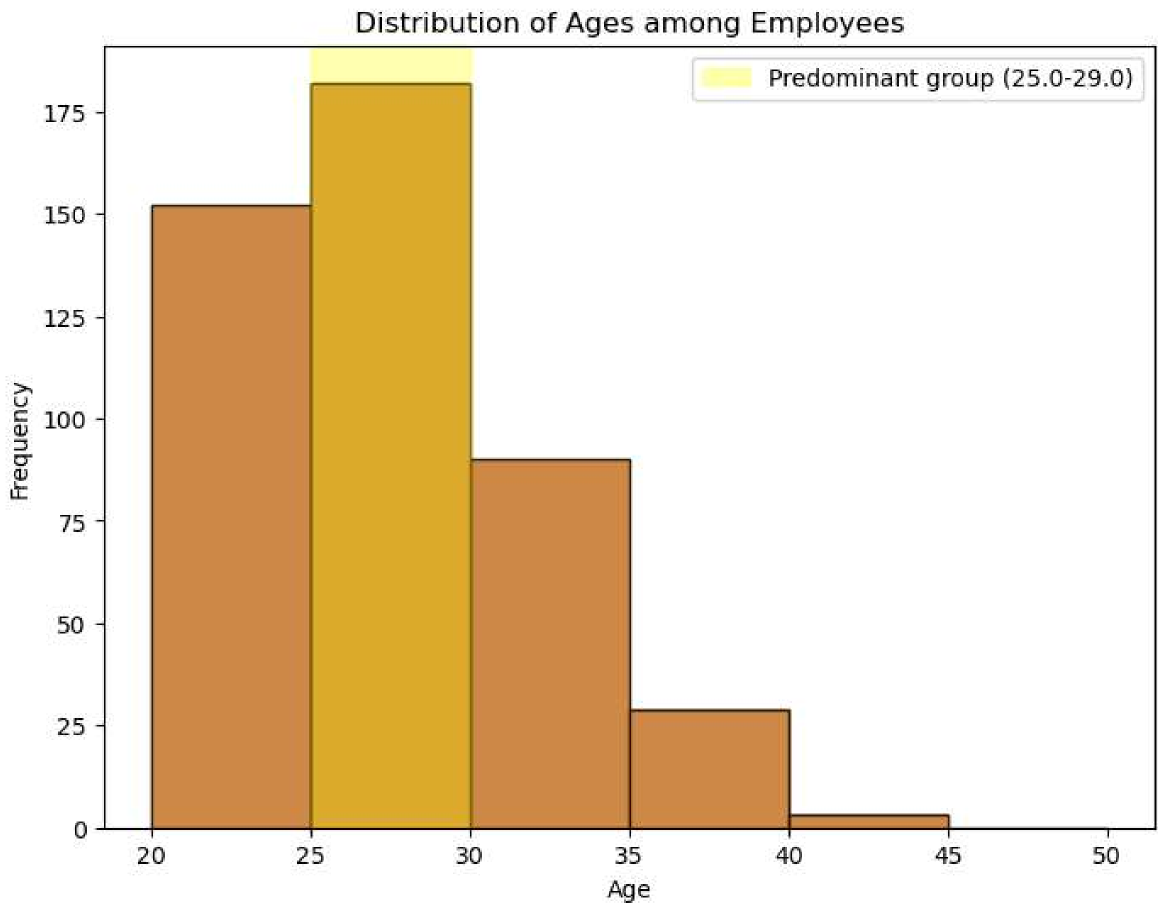
In [88]:
```
# Define age groups (adjust as per "myexcel.csv" dataset and requirements)
age_bins = [20, 30, 40, 50]  # Example age bins: 20-29, 30-39, 40-49
age_labels = ['20-29', '30-39', '40-49']
# Categorize employees into age groups
df['AgeGroup'] = pd.cut(df['Age'], bins=age_bins, labels=age_labels, right=False)

# Count the number of employees in each age group
age_group_counts = df['AgeGroup'].value_counts()

# Determine the predominant age group
predominant_age_group = age_group_counts.idxmax()
```

In [87]:
```
# Display the results
print("Age group counts:")
print(age_group_counts)
print("\nPredominant age group among employees:", predominant_age_group)
```

```
Age group counts:
AgeGroup
20-29    334
30-39    119
40-49      3
Name: count, dtype: int64

Predominant age group among employees: 20-29
```

In [6]:

```
# Plotting the histogram
plt.figure(figsize=(8, 6))
plt.hist(df['Age'], bins=range(20, 51, 5), edgecolor='black', alpha=0.7)
plt.xlabel('Age')
plt.ylabel('Frequency')
plt.title('Distribution of Ages among Employees')

# Find the predominant age group
age_counts, bins, _ = plt.hist(df['Age'], bins=range(20, 51, 5), edgecolor='black', alpha=0.7)
max_count_index = age_counts.argmax()
predominant_age_group = f'{bins[max_count_index]}-{bins[max_count_index + 1] - 1}'

# Highlight the predominant age group
plt.axvspan(bins[max_count_index], bins[max_count_index + 1], color='yellow', alpha=0.3, label=f'P
plt.legend()

plt.show()
```

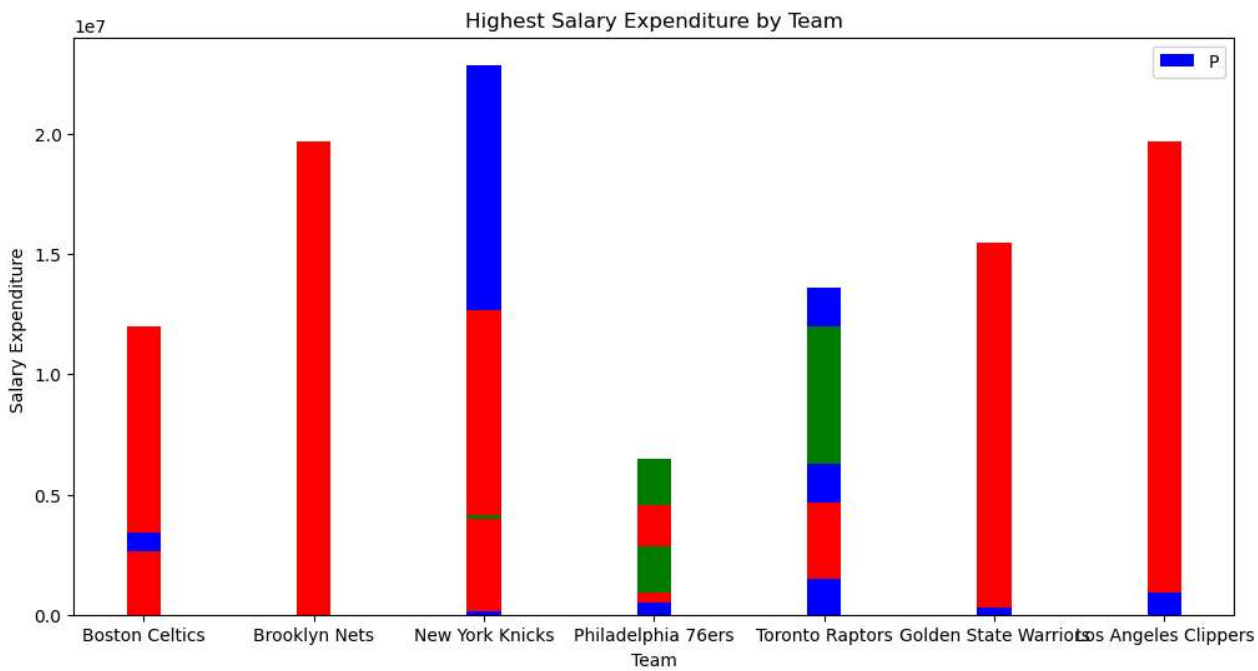


4. Discover which team and position have the highest salary expenditure.

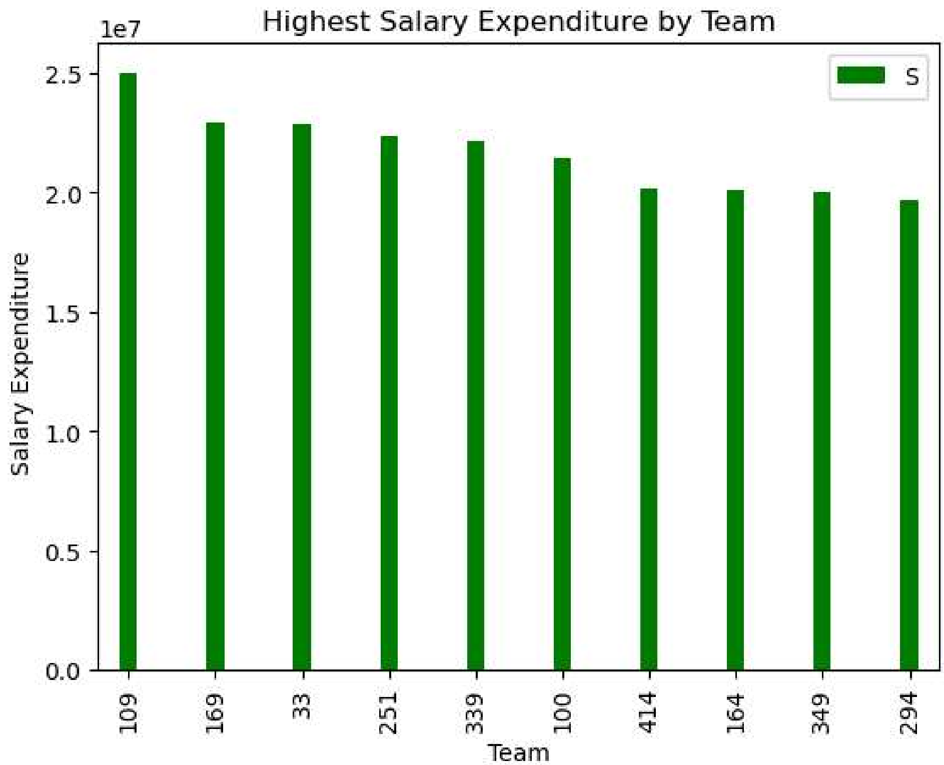
In [20]:
```
highest_salary=df.sort_values(by = 'Salary',ascending = False)[["Team","Position","Salary"]].head(

print(highest_salary.head())
```

```
                   Team Position      Salary
109    Los Angeles Lakers       SF  25000000.0
169   Cleveland Cavaliers       SF  22970500.0
33        New York Knicks       SF  22875000.0
251       Houston Rockets        C  22359364.0
339            Miami Heat       PF  22192730.0
```

In [33]:
```
# Data for plotting
x=df['Team'].head(100)
y=df['Salary'].head(100)
# Plotting the bar chart
plt.figure(figsize=(12, 6))
plt.bar(x,y, color=['blue', 'green', 'red'],width=0.2)

# Adding labels and title
plt.xlabel('Team')
plt.ylabel('Salary Expenditure ')
plt.title('Highest Salary Expenditure by Team')
plt.legend("Position")
# Adding the values on top of the bars
plt.show()
```



In [23]:
```
highest_salary.plot(kind='bar',color=[ 'green', 'red'],width=0.2)
plt.xlabel('Team')
plt.ylabel('Salary Expenditure ')
plt.title('Highest Salary Expenditure by Team')
plt.legend('Salary')
plt.show()
```

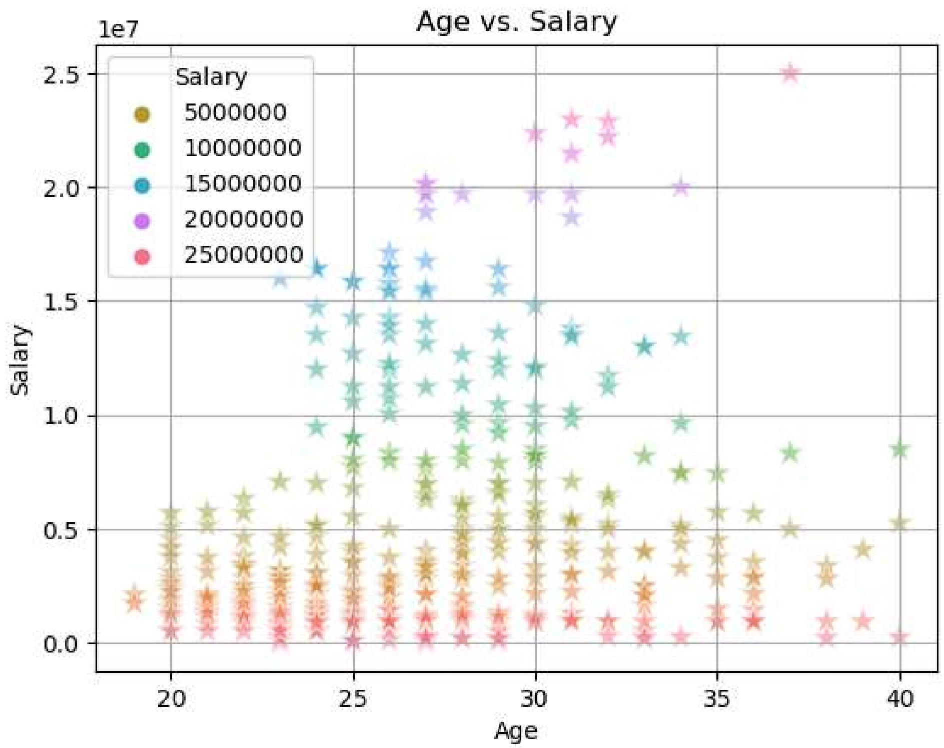


5. Investigate if there's any correlation between age and salary, and represent it visually.

In [18]:
```
correlation = df['Age'].corr(df['Salary'])
print("Correlation between age and salary:", correlation)
```

```
Correlation between age and salary: 0.21400941226570974
```

In [43]:
```
# plot a scatterplot for correlation between age and salary.
import seaborn as sns
import matplotlib.pyplot as plt

x=df['Age']
y=df['Salary']
sns.scatterplot(x=x, y=y, alpha=0.5,color="c",hue=df['Salary'],palette='husl',marker='*',s=150)
plt.title('Age vs. Salary')
plt.xlabel('Age')
plt.ylabel('Salary')
plt.grid(True)
plt.legend(title="Salary")
plt.show()
```



DATA STORY:

```
- We have embarked data of the company ,The resulting bar chart visually represents the distribution of employees
  across different teams within the company. Each bar represents a team (e.g New Orleans Pelicans) with its height indicating
  the percentage of employees in that team relative to the total workforce.
- Understanding how employees are segmented by their positions provides valuable insights into organizational dynamics,
  efficiency, and potential areas for improvement. By analyzing these data points, companies can better tailor their strategies
  for talent management, succession planning, and fostering a balanced organizational structure.
- Understanding the age demographics of employees is crucial for workforce planning, employee engagement strategies, and
  diversity initiatives. This data story delves into the age distribution within the company, highlighting the predominant
  age group and its implications.
- Understanding which teams and positions have the highest salary expenditure provides actionable insights for organizational
  decision-making. By pinpointing areas of significant investment in talent, companies can strategically allocate resources,
  foster competitive advantage, and ensure alignment with financial goals.
```

```
- Understanding the relationship between age and salary is crucial for both employees and
employers. It provides insights into
  career trajectories, earning potential, and possibly highlights any age-related biases or
opportunities for skill development
  and compensation adjustments.
```

In [ ]: