HR DATA ANALYSIS

Importing necessary libraries for data manipulation, visualization, and handling warnings:

In [4]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import warnings
```

Reading the CSV file into a DataFrame using Pandas:

In [5]:
```
df=pd.read_csv(r"C:\Users\masir\Downloads\HR Data.csv")
pd.set_option('display.max_columns',None)
df.head()
```

Out[5]:

	Age	Attrition	BusinessTravel	DailyRate	Department	DistanceFromHome	Education	E
0	41	Yes	Travel_Rarely	1102	Sales	1	2	
1	49	No	Travel_Frequently	279	Research & Development	8	1	
2	37	Yes	Travel_Rarely	1373	Research & Development	2	2	
3	33	No	Travel_Frequently	1392	Research & Development	3	4	
4	27	No	Travel_Rarely	591	Research & Development	2	1	

Displaying a concise summary of the DataFrame using the info() method:

In [5]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1470 entries, 0 to 1469
Data columns (total 35 columns):
 #   Column                    Non-Null Count  Dtype 
---  ------                    --------------  ----- 
 0   Age                       1470 non-null   int64 
 1   Attrition                 1470 non-null   int64 
 2   BusinessTravel            1470 non-null   object
 3   DailyRate                 1470 non-null   int64 
 4   Department                1470 non-null   object
 5   DistanceFromHome          1470 non-null   int64 
 6   Education                 1470 non-null   int64 
 7   EducationField            1470 non-null   object
 8   EmployeeCount             1470 non-null   int64 
 9   EmployeeNumber            1470 non-null   int64 
 10  EnvironmentSatisfaction   1470 non-null   int64 
 11  Gender                    1470 non-null   object
 12  HourlyRate                1470 non-null   int64 
 13  JobInvolvement            1470 non-null   int64 
 14  JobLevel                  1470 non-null   int64 
 15  JobRole                   1470 non-null   object
 16  JobSatisfaction           1470 non-null   int64 
 17  MaritalStatus             1470 non-null   object
 18  MonthlyIncome             1470 non-null   int64 
 19  MonthlyRate               1470 non-null   int64 
 20  NumCompaniesWorked        1470 non-null   int64 
 21  Over18                    1470 non-null   int64 
 22  OverTime                  1470 non-null   int64 
 23  PercentSalaryHike         1470 non-null   int64 
 24  PerformanceRating         1470 non-null   int64 
 25  RelationshipSatisfaction  1470 non-null   int64 
 26  StandardHours             1470 non-null   int64 
 27  StockOptionLevel          1470 non-null   int64 
 28  TotalWorkingYears         1470 non-null   int64 
 29  TrainingTimesLastYear     1470 non-null   int64 
 30  WorkLifeBalance           1470 non-null   int64 
 31  YearsAtCompany            1470 non-null   int64 
 32  YearsInCurrentRole        1470 non-null   int64 
 33  YearsSinceLastPromotion   1470 non-null   int64 
 34  YearsWithCurrManager      1470 non-null   int64 
dtypes: int64(29), object(6)
memory usage: 402.1+ KB
```

Generating descriptive statistics for the DataFrame using the describe() method:

In [6]:
```
df.describe()
```

Out[6]:

	Age	Attrition	DailyRate	DistanceFromHome	Education	EmployeeCo
count	1470.000000	1470.000000	1470.000000	1470.000000	1470.000000	147
mean	36.923810	0.161224	802.485714	9.192517	2.912925	
std	9.135373	0.367863	403.509100	8.106864	1.024165	
min	18.000000	0.000000	102.000000	1.000000	1.000000	
25%	30.000000	0.000000	465.000000	2.000000	2.000000	
50%	36.000000	0.000000	802.000000	7.000000	3.000000	
75%	43.000000	0.000000	1157.000000	14.000000	4.000000	
max	60.000000	1.000000	1499.000000	29.000000	5.000000	

Dropping unnecessary columns from the DataFrame

In [9]:
```
df.drop(['EmployeeCount','Over18','StandardHours'],axis=1,inplace=True)
```

In [10]:
```
# Calculating the absolute value of the correlation matrix for numerical co
corr=df.corr(numeric_only=True).abs()
corr.style.background_gradient(cmap='Blues')
```

	Age	DailyRate	DistanceFromHome	Education	EmployeeNumber	EnvironmentSa
Age	1.000000	0.010661	0.001686	0.208034	0.010145	
DailyRate	0.010661	1.000000	0.004985	0.016806	0.050990	
DistanceFromHome	0.001686	0.004985	1.000000	0.021042	0.032916	
Education	0.208034	0.016806	0.021042	1.000000	0.042070	
EmployeeNumber	0.010145	0.050990	0.032916	0.042070	1.000000	
vironmentSatisfaction	0.010146	0.018355	0.016075	0.027128	0.017621	
HourlyRate	0.024287	0.023381	0.031131	0.016775	0.035179	
JobInvolvement	0.029820	0.046135	0.008783	0.042438	0.006888	
JobLevel	0.509604	0.002966	0.005303	0.101589	0.018519	
JobSatisfaction	0.004892	0.030571	0.003669	0.011296	0.046247	

Dropping duplicates and NA values

In [11]:
```
df.drop_duplicates()
df.dropna()
```

Out[11]:

	Age	Attrition	BusinessTravel	DailyRate	Department	DistanceFromHome	Education
0	41	Yes	Travel_Rarely	1102	Sales	1	2
1	49	No	Travel_Frequently	279	Research & Development	8	1
2	37	Yes	Travel_Rarely	1373	Research & Development	2	2
3	33	No	Travel_Frequently	1392	Research & Development	3	4
4	27	No	Travel_Rarely	591	Research & Development	2	1
...	...	...	...	...	...	...	...
1465	36	No	Travel_Frequently	884	Research & Development	23	2
1466	39	No	Travel_Rarely	613	Research & Development	6	1
1467	27	No	Travel_Rarely	155	Research & Development	4	3
1468	49	No	Travel_Frequently	1023	Sales	2	3
1469	34	No	Travel_Rarely	628	Research & Development	8	3

1470 rows × 32 columns

Display the first five rows of the DataFrame

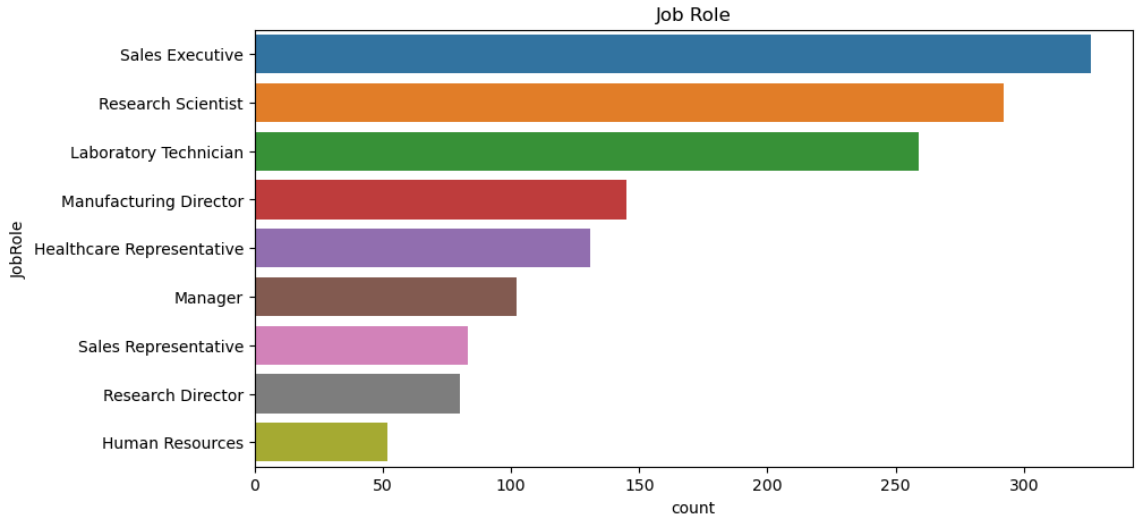
In [14]:
```
df.head()
```

Out[14]:

	Age	Attrition	BusinessTravel	DailyRate	Department	DistanceFromHome	Education	Ed
0	51	0	Travel_Rarely	684	Research & Development	6	3	
1	52	0	Travel_Rarely	699	Research & Development	1	4	
2	42	0	Travel_Rarely	532	Research & Development	4	2	
3	47	0	Travel_Rarely	359	Research & Development	2	4	
4	46	0	Travel_Rarely	1319	Sales	3	3	

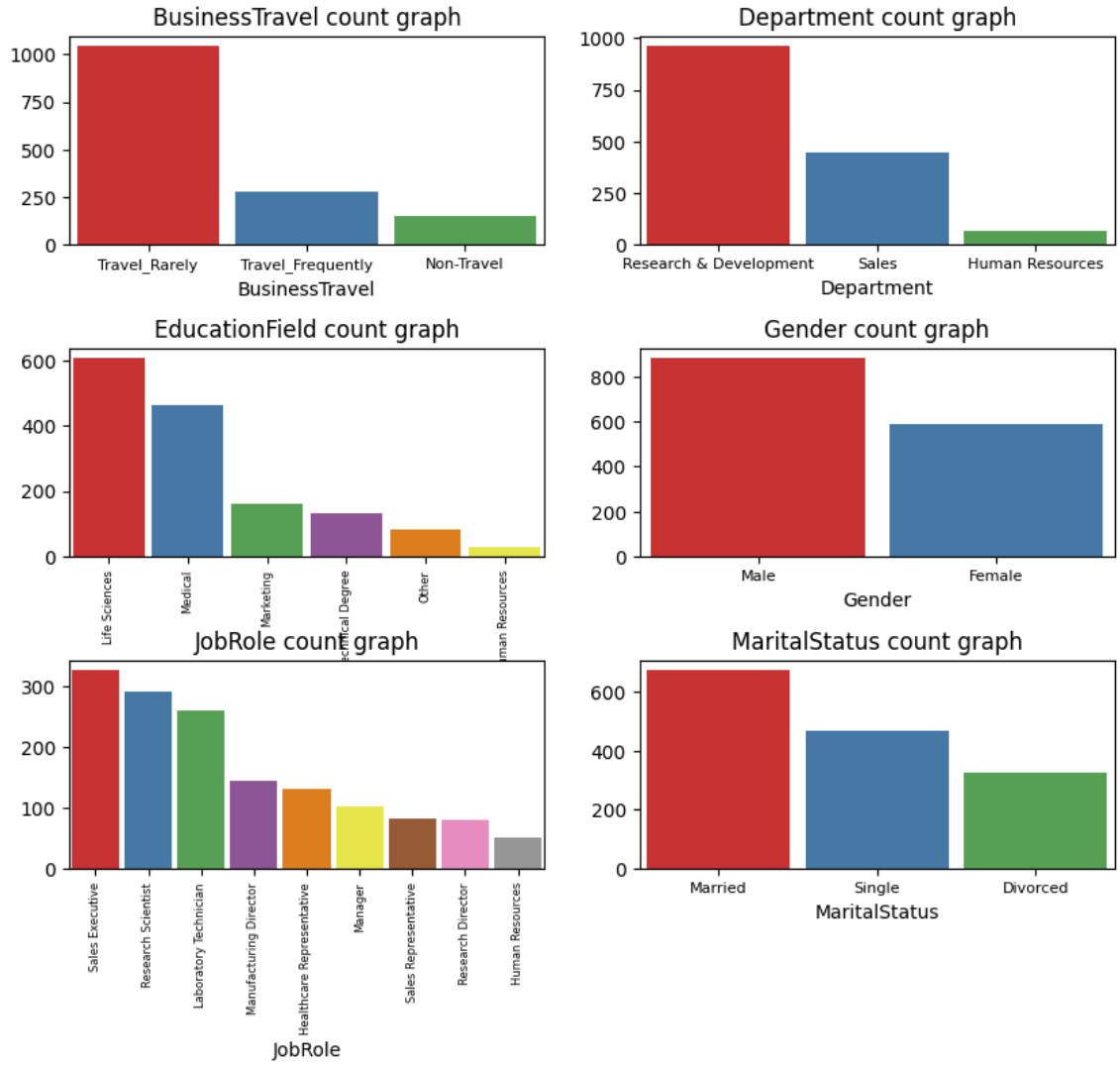
Creating some visualizations.

In [14]:
```
plt.figure(figsize=(10,5))
sns.countplot(data=df,y='JobRole')
plt.title("Job Role")
plt.show()
```



List of categorical columns to be visualized:

In [13]:
```
categorical_cols=['BusinessTravel','Department','EducationField','Gender','
# Creating subplots grid to display count plots for each categorical column
fig,ax=plt.subplots(3,2,figsize=(10,8))
# Adjusting the spacing between subplots vertically
fig.subplots_adjust(hspace=0.5)
# create count plots for each categorical column
def count_plotter(ax,col,data=df):
    # Counting the occurrences of each category in the column
    counted=df[col].value_counts()
    # Creating a bar plot to visualize the count of each category
    sns.barplot(ax=ax,x=counted.index,y=counted.values,width=0.9,palette='S
    # Setting title for the subplot based on the column name
    ax.set_title(f"{col} count graph")
    # Rotating x-axis labels for better readability in case of 'JobRole' or
    if col=='JobRole' or col=='EducationField':
       ax.set_xticklabels(labels=counted.index,rotation=90,fontsize=6)
    else:
        ax.set_xticklabels(labels=counted.index,fontsize=8)
# Flattening the axes array to simplify indexing during iteration
axes=[ax[0,0],ax[0,1],ax[1,0],ax[1,1],ax[2,0],ax[2,1]]
# Iterating over each categorical column to create count plots
for category in categorical_cols:
    count_plotter(axes[categorical_cols.index(category)],category)
```



Iterate over specified column names and corresponding x-axis values, creating a FacetGrid plot for each combination:

In [13]:
```
df_heatmap1=df.groupby(['JobRole','JobSatisfaction']).MonthlyIncome.mean().
df_heatmap2=df.groupby(['JobInvolvement','JobLevel']).MonthlyRate.mean().un
df_heatmap2
```

Out[13]:

JobLevel	1	2	3	4	5
JobInvolvement					
1	12661.666667	15916.314286	14240.900000	21197.000000	14848.000000
2	13853.708029	15410.187500	14519.954545	13940.370370	15923.117647
3	13488.798742	14447.791798	14817.562500	14592.854839	13255.860465
4	14905.051724	14445.703704	13781.285714	14024.857143	18149.250000

Create a grid of subplots with various types of plots to visualize different aspects of the data:

In [19]:
```
#plt.figure(figsize=(8,4))
fig,ax=plt.subplots(3,2,figsize=(15,13))
fig.suptitle('General Statistics')
fig.subplots_adjust(wspace=0.4,hspace=0.5)
sns.boxplot(ax=ax[0,0],data=df,y='Department',x='TotalWorkingYears',hue='Ge
ax[0,0].set_title('Ages by Department',fontsize=14)
sns.boxplot(ax=ax[0,1],data=df,y='EducationField',x='Age',hue='Gender',pale
ax[0,1].set_title('Ages by Education Field',fontsize=14)
sns.heatmap(ax=ax[1,0],data=df_heatmap1,square=True,linewidth=1,cmap='Reds'
ax[1,0].set_title('Job Role-Satisfaction Mapping',fontsize=14)
sns.heatmap(ax=ax[1,1],data=df_heatmap2,square=True,linewidth=1,cmap='Blues
ax[1,1].set_title('Job Level-Involvement Mapping',fontsize=14)
sns.histplot(ax=ax[2,0],data=df,x='PercentSalaryHike',hue='Gender',multiple
ax[2,0].set_title('Distribution of Salary Percent Hike',fontsize=14)
sns.histplot(ax=ax[2,1],data=df,x='YearsAtCompany',hue='Gender',multiple='s
ax[2,1].set_title('Distribution of Worked Year in Company',fontsize=14)
plt.show()
```

```
C:\Users\sherr\anaconda3\Lib\site-packages\seaborn\_oldcore.py:1119: Futur
eWarning: use_inf_as_na option is deprecated and will be removed in a futu
re version. Convert inf values to NaN before operating instead.
  with pd.option_context('mode.use_inf_as_na', True):
C:\Users\sherr\anaconda3\Lib\site-packages\seaborn\_oldcore.py:1119: Futur
eWarning: use_inf_as_na option is deprecated and will be removed in a futu
re version. Convert inf values to NaN before operating instead.
  with pd.option_context('mode.use_inf_as_na', True):
```