```
📚 Importing Libraries 📚
```

```
# Library for Data Manipulation.
import pandas as pd
import numpy as np

#Library for Data Visualization
import seaborn as sns
import matplotlib.pyplot as plt
import altair as alt
sns.set(style="white",font_scale=1.5)
sns.set(rc={"axes.facecolor":"#FFFAF0","figure.facecolor":"#FFFAF0"})
sns.set_context("poster",font_scale = .7)


# Library for perform  Statical Analysis.
from scipy import stats
from scipy.stats import chi2
from scipy.stats import chi2_contingency
```

Import Dataset:

```
from google.colab import files
data_to_load = files.upload()
```

Choose Files WA_Fn-Us…-Attrition.csv

- **WA_Fn-UseC_-HR-Employee-Attrition.csv**(text/csv) - 227977 bytes, last modified: 7/10/2023 - 100% done

```
Saving WA_Fn-UseC_-HR-Employee-Attrition.csv to WA_Fn-UseC_-HR-Employee-Attrition.csv
```

```
df = pd.read_csv('WA_Fn-UseC_-HR-Employee-Attrition.csv')
```

```
df.head().style.set_properties(**{'background-color': '#E9F6E2','color': 'black','border-color': '#8b8c8c'})
```

	Age	Attrition	BusinessTravel	DailyRate	Department	DistanceFromHome	Education	EducationField	EmployeeCount	EmployeeNumber	En
0	41	Yes	Travel_Rarely	1102	Sales	1	2	Life Sciences	1	1	
1	49	No	Travel_Frequently	279	Research & Development	8	1	Life Sciences	1	2	
2	37	Yes	Travel_Rarely	1373	Research & Development	2	2	Other	1	4	
3	33	No	Travel_Frequently	1392	Research & Development	3	4	Life Sciences	1	5	
4	27	No	Travel_Rarely	591	Research & Development	2	1	Medical	1	7	

Calculate the Dimension of Dataset.

```
df.shape
```

```
(1470, 35)
```

1. There are total 1470 rows in the dataset.
2. There are 35 columns in the dataset.

```
#Generating  Basic Information of Attribute.

df.info(verbose=False)
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1470 entries, 0 to 1469
Columns: 35 entries, Age to YearsWithCurrManager
dtypes: int64(26), object(9)
memory usage: 402.1+ KB
```

1.There are 26 Numerical Attributes in the dataset.

2. On the Other hand we have 9 Categorical Attributes.*italicized text*

Display the random ssmple of dataset with only numerical values.

```
df.select_dtypes(np.number).sample(5).style.set_properties(**{'background-color': '#E9F6E2',
                                                              'color': 'black','border-color': '#8b8c8c'})
```

	Age	DailyRate	DistanceFromHome	Education	EmployeeCount	EmployeeNumber	EnvironmentSatisfaction	HourlyRate	JobInvolvement	Jo
973	35	817	1	3	1	1369	4	60	2	
1326	32	414	2	4	1	1862	3	82	2	
1008	54	971	1	3	1	1422	4	54	3	
1390	28	1404	17	3	1	1960	3	32	2	
926	43	531	4	4	1	1293	4	56	2	

1. Some of the numerical features are storing cateegories lalled in numbers.
2. So for better analysis se will replace those lebelled numerical values with appropriate categorical values.

Labelling of Categories in Numerical Features.

```
df["Education"] = df["Education"].replace({1:"Below College",2:"College",3:"Bachelor",4:"Master",5:"Doctor"})
```

```
df["EnvironmentSatisfaction"] = df["EnvironmentSatisfaction"].replace({1:"Low",2:"Medium",3:"High",4:"Very High"})
```

```
df["JobInvolvement"] = df["JobInvolvement"].replace({1:"Low",2:"Medium",3:"High",4:"Very High"})
```

```
df["JobLevel"] = df["JobLevel"].replace({1:"Entry Level",2:"Junior Level",3:"Mid Level",4:"Senior Level", 5:"Executive Level"})
```

```
df["JobSatisfaction"] = df["JobSatisfaction"].replace({1:"Low",2:"Medium",3:"High",4:"Very High"})
```

```
df["PerformanceRating"] = df["PerformanceRating"].replace({1:"Low",2:"Good",3:"Excellent",4:"Outstanding"})
```

```
df["RelationshipSatisfaction"] = df["RelationshipSatisfaction"].replace({1:"Low",2:"Medium",3:"High",4:"Very High"})
```

```
df["WorkLifeBalance"] = df["WorkLifeBalance"].replace({1:"Bad",2:"Good",3:"Better",4:"Best"})
```

Display a random sample of dataset with only categorical values.

```
df.select_dtypes(include="O").sample(5).style.set_properties(**{'background-color': '#  #FFD39B', 'color': 'black','border-color': '#8b8c8c'}
```

	Attrition	BusinessTravel	Department	Education	EducationField	EnvironmentSatisfaction	Gender	JobInvolvement	JobLevel	
307	No	Travel_Rarely	Research & Development	College	Life Sciences	Low	Female	Medium	Mid Level	
1057	Yes	Travel_Frequently	Sales	Bachelor	Technical Degree	Low	Female	High	Junior Level	
1102	No	Travel_Rarely	Sales	Master	Life Sciences	High	Male	High	Entry Level	Repr
1287	No	Travel_Rarely	Research & Development	Bachelor	Medical	Medium	Male	Very High	Junior Level	H Repr

Checking if there is any duplicaate records.

```
have_duplicate_rows = df.duplicated().any()
have_duplicate_rows
```

```
False
```

Output is false so we can say that there is no duplicate records present in the dataset.

Calculate the total number of missing values and the percentage of missing values.

```
missing_df = df.isnull().sum().to_frame().rename(columns={0:"Total No. of Missing Values"})
missing_df["% of Missing Values"] = round((missing_df["Total No. of Missing Values"]/len(df))*100,2)
missing_df
```

	Total No. of Missing Values	% of Missing Values
Age	0	0.0
Attrition	0	0.0
BusinessTravel	0	0.0
DailyRate	0	0.0
Department	0	0.0
DistanceFromHome	0	0.0
Education	0	0.0
EducationField	0	0.0
EmployeeCount	0	0.0
EmployeeNumber	0	0.0
EnvironmentSatisfaction	0	0.0

None of the Attribute are having missing values.

```
df.describe()
```

	Age	DailyRate	DistanceFromHome	EmployeeCount	EmployeeNumber	HourlyRate	MonthlyIncome	MonthlyRate	NumCompaniesWo
count	1470.000000	1470.000000	1470.000000	1470.0	1470.000000	1470.000000	1470.000000	1470.000000	1470.00
mean	36.923810	802.485714	9.192517	1.0	1024.865306	65.891156	6502.931293	14313.103401	2.69
std	9.135373	403.509100	8.106864	0.0	602.024335	20.329428	4707.956783	7117.786044	2.49
min	18.000000	102.000000	1.000000	1.0	1.000000	30.000000	1009.000000	2094.000000	0.00
25%	30.000000	465.000000	2.000000	1.0	491.250000	48.000000	2911.000000	8047.000000	1.00
50%	36.000000	802.000000	7.000000	1.0	1020.500000	66.000000	4919.000000	14235.500000	2.00
75%	43.000000	1157.000000	14.000000	1.0	1555.750000	83.750000	8379.000000	20461.500000	4.00
max	60.000000	1499.000000	29.000000	1.0	2068.000000	100.000000	19999.000000	26999.000000	9.00

```
cols = ["Over18","EmployeeCount","EmployeeNumber","StandardHours"]
```

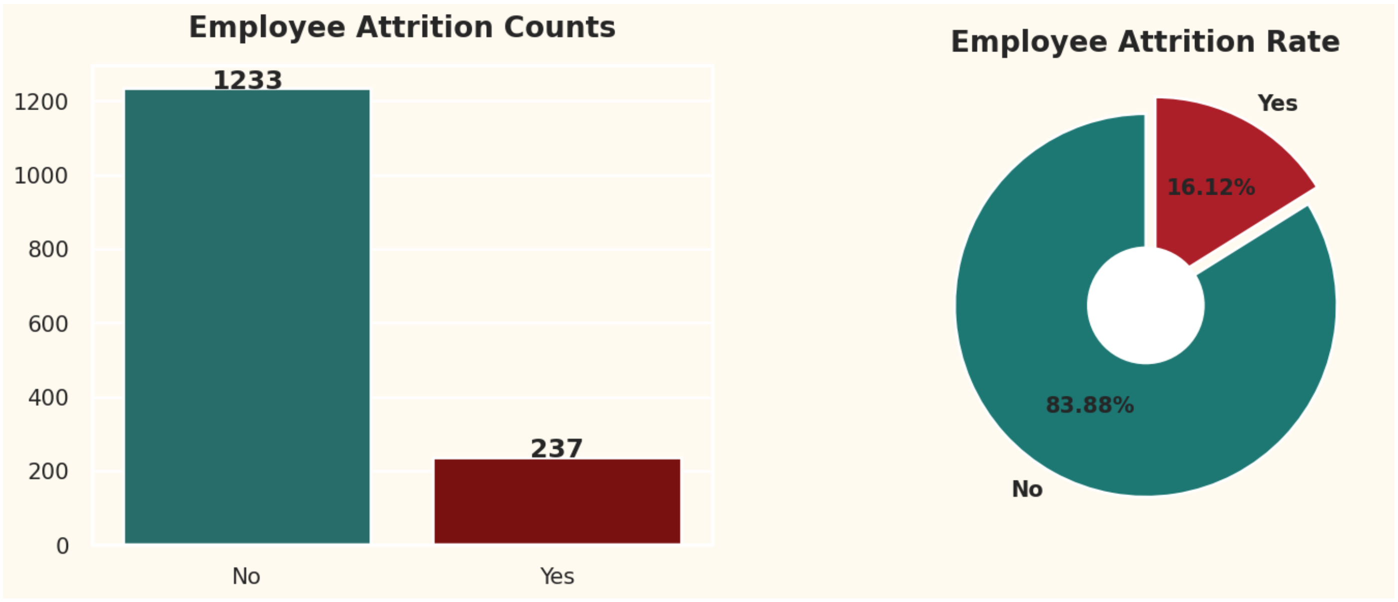
```
df.describe()
```

	Age	DailyRate	DistanceFromHome	EmployeeCount	EmployeeNumber	HourlyRate	MonthlyIncome	MonthlyRate	NumCompaniesWo
count	1470.000000	1470.000000	1470.000000	1470.0	1470.000000	1470.000000	1470.000000	1470.000000	1470.00
mean	36.923810	802.485714	9.192517	1.0	1024.865306	65.891156	6502.931293	14313.103401	2.69
std	9.135373	403.509100	8.106864	0.0	602.024335	20.329428	4707.956783	7117.786044	2.49
min	18.000000	102.000000	1.000000	1.0	1.000000	30.000000	1009.000000	2094.000000	0.00
25%	30.000000	465.000000	2.000000	1.0	491.250000	48.000000	2911.000000	8047.000000	1.00
50%	36.000000	802.000000	7.000000	1.0	1020.500000	66.000000	4919.000000	14235.500000	2.00
75%	43.000000	1157.000000	14.000000	1.0	1555.750000	83.750000	8379.000000	20461.500000	4.00
max	60.000000	1499.000000	29.000000	1.0	2068.000000	100.000000	19999.000000	26999.000000	9.00

```
#cat_cols = df.select_dtypes(include="O").columns

#for column in cat_cols:
 #   print('Unique values of ', column, set(df[column]))
  #  print("-"*127)
```

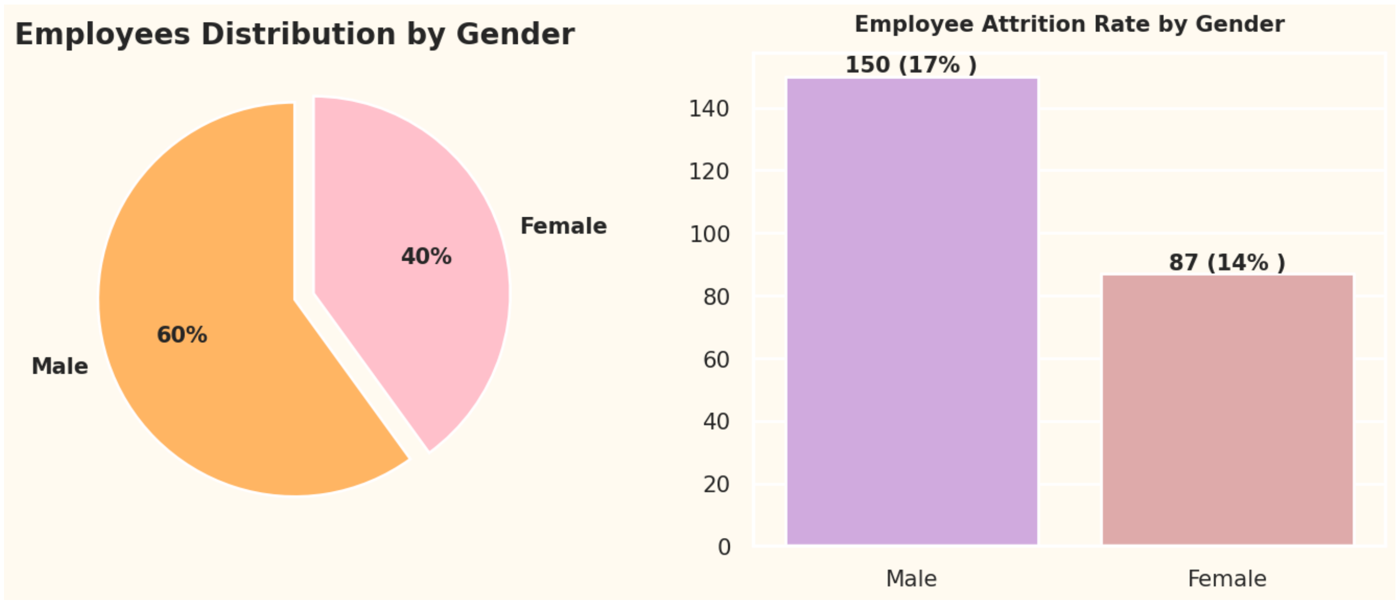
```
#Visualization to show Employee Attrition in Counts.
plt.figure(figsize=(17,6))
plt.subplot(1,2,1)
```

```
attrition_rate = df["Attrition"].value_counts()
sns.barplot(x=attrition_rate.index,y=attrition_rate.values,palette=["#1d7874","#8B0000"])
plt.title("Employee Attrition Counts",fontweight="black",size=20,pad=20)
for i, v in enumerate(attrition_rate.values):
    plt.text(i, v, v,ha="center", fontweight='black', fontsize=18)


#Visualization to show Employee Attrition in Percentage.
plt.subplot(1,2,2)
plt.pie(attrition_rate, labels=["No","Yes"], autopct="%.2f%%", textprops={"fontweight":"black","size":15},
        colors = ["#1d7874","#AC1F29"],explode=[0,0.1],startangle=90)
center_circle = plt.Circle((0, 0), 0.3, fc='white')
fig = plt.gcf()
fig.gca().add_artist(center_circle)
plt.title("Employee Attrition Rate",fontweight="black",size=20,pad=10)
plt.show()
```



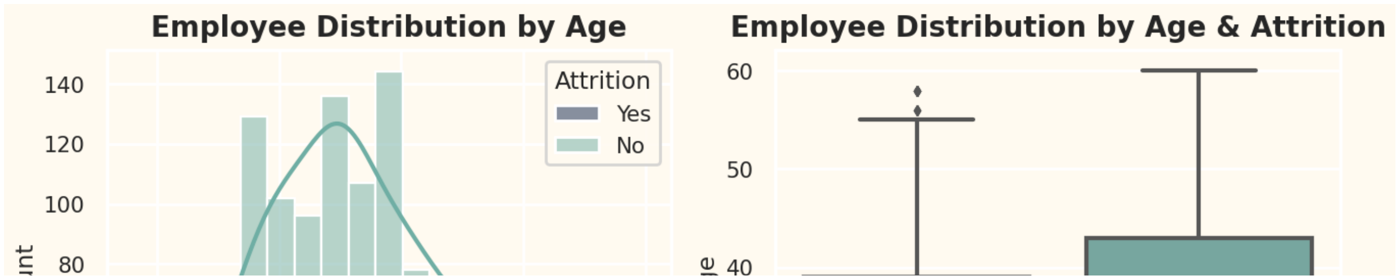
1. The employee attrition rate of this organization is 16.12%.
2. According to experts in the field of Human Resources, says that the attrition rate 4% to 6% is normal in organization.
3. So we can say the attrition rate of the organization is at a dangerous level.
4. Therefore the organization should take measures to reduce the attrition rate.

```
#Visualization to show Total Employees by Gender.
plt.figure(figsize=(14,6))
plt.subplot(1,2,1)
gender_attrition = df["Gender"].value_counts()
plt.title("Employees Distribution by Gender",fontweight="black",size=20)
plt.pie(gender_attrition, autopct="%.0f%%",labels=gender_attrition.index,textprops=({"fontweight":"black","size":15}),
        explode=[0,0.1],startangle=90,colors= ["#ffb563","#FFC0CB"])



#Visualization to show Employee Attrition by Gender.
plt.subplot(1,2,2)
new_df = df[df["Attrition"]=="Yes"]
value_1 = df["Gender"].value_counts()
value_2 = new_df["Gender"].value_counts()
attrition_rate = np.floor((value_2/value_1)*100).values
sns.barplot(x=value_2.index, y=value_2.values,palette=["#D4A1E7","#E7A1A1"])
plt.title("Employee Attrition Rate by Gender",fontweight="black",size=15,pad=15)
for index,value in enumerate(value_2):
    plt.text(index,value,str(value)+" ("+str(int(attrition_rate[index]))+"% )",ha="center",va="bottom",
             size=15,fontweight="black")
plt.tight_layout()
plt.show()
```

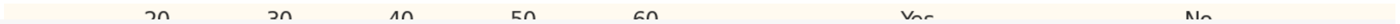


1. The number of male employees in the organization accounts for a higher proportion than female employees by more than 20%.
2. Male employees are leaving more from the organization compared to female employees.

```
#Visualization to show Employee Distribution by Age.
plt.figure(figsize=(13.5,6))
plt.subplot(1,2,1)
sns.histplot(x="Age",hue="Attrition",data=df,kde=True,palette=["#11264e","#6faea4"])
plt.title("Employee Distribution by Age",fontweight="black",size=20,pad=10)


#Visualization to show Employee Distribution by Age & Attrition.
plt.subplot(1,2,2)
sns.boxplot(x="Attrition",y="Age",data=df,palette=["#D4A1E7","#6faea4"])
plt.title("Employee Distribution by Age & Attrition",fontweight="black",size=20,pad=10)
plt.tight_layout()
plt.show()
```



1. Most of the emloyees are between age 30 to 40.
2. We can clearly observe a trend that as the age is increasing the attrition is decreasing.
3. From the boxplot we can also observe that the medain age of employee who left the organization is less than the employees who are working in the organization.
4. Employees with young age leaves the company more compared to elder employees.

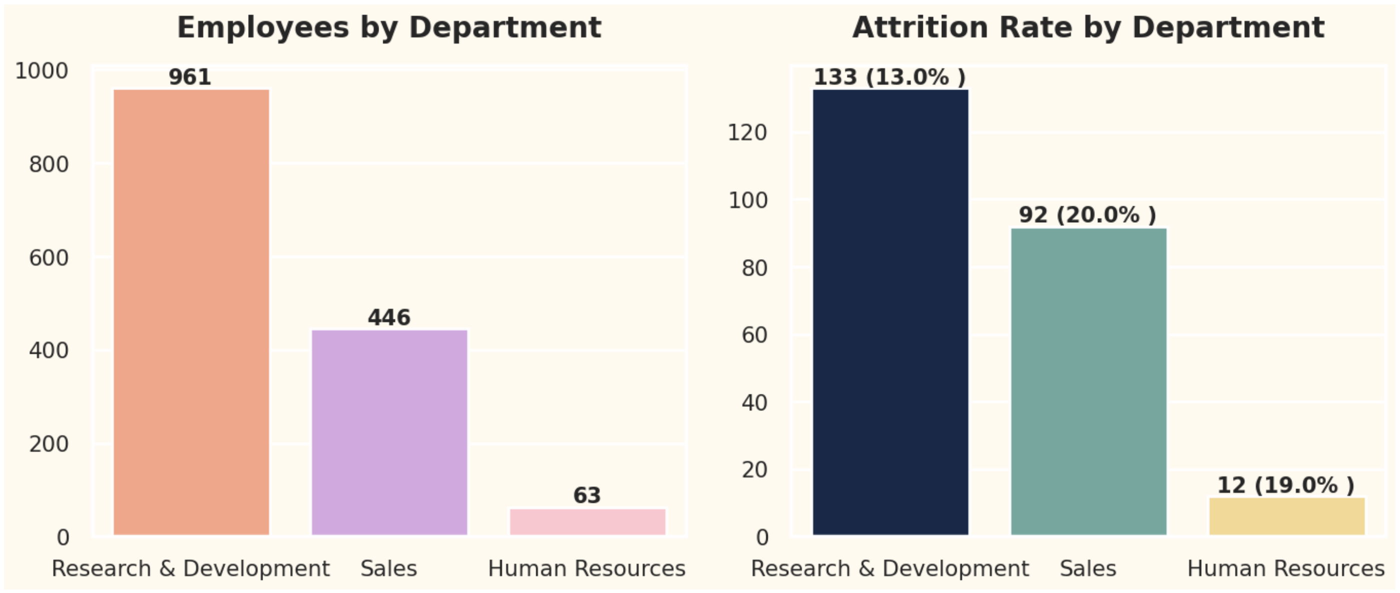


```
#Visualization to show Total Employees by Businees Travel.

plt.figure(figsize=(14,6))
plt.subplot(1,2,1)
value_1 = df["BusinessTravel"].value_counts()
plt.title("Employees by Business Travel", fontweight="black", size=20, pad=20)
plt.pie(value_1.values, labels=value_1.index, autopct="%.1f%%",pctdistance=0.75,startangle=90,
        colors=['#E84040', '#E96060', '#E88181'],textprops={"fontweight":"black","size":15})
center_circle = plt.Circle((0, 0), 0.4, fc='white')
fig = plt.gcf()
fig.gca().add_artist(center_circle)



#Visualization to show Attrition Rate by Businees Travel.
plt.subplot(1,2,2)
new_df = df[df["Attrition"]=="Yes"]
value_2 = new_df["BusinessTravel"].value_counts()
attrition_rate = np.floor((value_2/value_1)*100).values
sns.barplot(x=value_2.index,y=value_2.values,palette=["#11264e","#6faea4","#FEE08B"])
plt.title("Attrition Rate by Businees Travel",fontweight="black",size=20,pad=20)
for index,value in enumerate(value_2):
    plt.text(index,value,str(value)+" ("+str(int(attrition_rate[index]))+"% )",ha="center",va="bottom",
             size=15,fontweight="black")
plt.tight_layout()
plt.show()
```

Employees by Business Travel
Attrition Rate by Businees Travel

1. Most of the employees in the organization Travel Rarely.
2. Highest employee attrition can be observed by those employees who Travels Frequently.
3. Lowest employee attrition can be observed by those employees who are Non-Travel.

```
     #Visualization to show Total Employees by Department.

plt.figure(figsize=(14,6))
plt.subplot(1,2,1)
value_1 = df["Department"].value_counts()
sns.barplot(x=value_1.index, y=value_1.values,palette = ["#FFA07A", "#D4A1E7", "#FFC0CB"])
plt.title("Employees by Department",fontweight="black",size=20,pad=20)
for index,value in enumerate(value_1.values):
    plt.text(index,value,value,ha="center",va="bottom",fontweight="black",size=15,)



    #Visualization to show Employee Attrition Rate by Department.

plt.subplot(1,2,2)
new_df = df[df["Attrition"]=="Yes"]
value_2 = new_df["Department"].value_counts()
attrition_rate = np.floor((value_2/value_1)*100).values
sns.barplot(x=value_2.index, y=value_2.values,palette=["#11264e","#6faea4","#FEE08B"])
plt.title("Attrition Rate by Department",fontweight="black",size=20,pad=20)
for index,value in enumerate(value_2):
    plt.text(index,value,str(value)+" ("+str(attrition_rate[index])+"% )",ha="center",va="bottom",
             size=15,fontweight="black")
plt.tight_layout()
plt.show()
```



1. Most of the employees are from Research & Development Department.
2. Highest Attrition is in the Sales Department.
3. Human Resources Department Attrition rate is also very high.

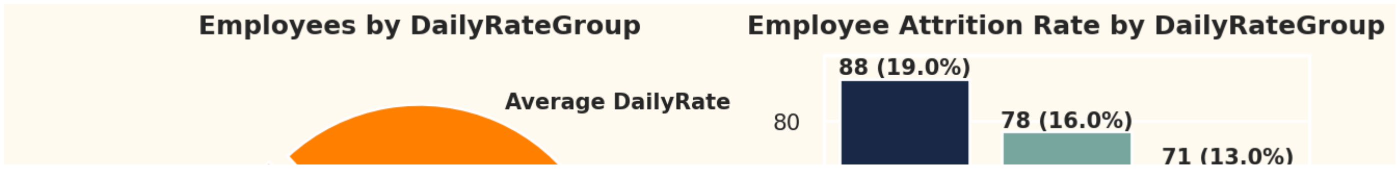
```
df["DailyRate"].describe().to_frame()
```

	DailyRate
count	1470.000000
mean	802.485714
std	403.509100
min	102.000000
25%	465.000000
50%	802.000000
75%	1157.000000
max	1499.000000

DailyRate shows the Daily salary rate for employees. To generate meaningful insights we can cut Daily Rates into three groups for meaningful analysis.

```
      # Define the bin edges for the groups
bin_edges = [0, 500, 1000, 1500]

# Define the labels for the groups
bin_labels = ['Low DailyRate', 'Average DailyRate', 'High DailyRate']

# Cut the DailyRate column into groups
df['DailyRateGroup'] = pd.cut(df['DailyRate'], bins=bin_edges, labels=bin_labels)
```

```
   ##Visualization to show Total Employees by DailyRateGroup.

plt.figure(figsize=(13,6))
plt.subplot(1,2,1)
value_1 = df["DailyRateGroup"].value_counts()
plt.pie(value_1.values, labels=value_1.index,autopct="%.2f%%",textprops={"fontweight":"black","size":15},
        explode=[0,0.1,0.1],colors= ['#FF8000', '#FF9933', '#FFB366', '#FFCC99'])
plt.title("Employees by DailyRateGroup",fontweight="black",pad=15,size=18)

    #Visualization to show Attrition Rate by DailyRateGroup.

plt.subplot(1,2,2)
new_df = df[df["Attrition"]=="Yes"]
value_2 = new_df["DailyRateGroup"].value_counts()
attrition_rate = np.floor((value_2/value_1)*100).values
sns.barplot(x=value_2.index.tolist(),y= value_2.values,palette=["#11264e","#6faea4","#FEE08B"])
plt.title("Employee Attrition Rate by DailyRateGroup",fontweight="black",pad=15,size=18)
for index,value in enumerate(value_2.values):
    plt.text(index,value, str(value)+" ("+str(attrition_rate[index])+"%)",ha="center",va="bottom",fontweight="black",size=15)

plt.tight_layout()
plt.show()
```



1. Employees with Average DailyRate & High Daily Rate are approxiamately equal.
2. But the attrition rate is very high of employees with average Daily Rate compared to the employees with High DailyRate. 3. The attrition rate is also high of employees with low DailyRate. 4. Employees which are not getting High Daily Rate are mostly leaving the organization.

```
#Analyzing Employee Attrition by Distance From Home.


print("Total Unique Values in Attribute is =>",df["DistanceFromHome"].nunique())
```

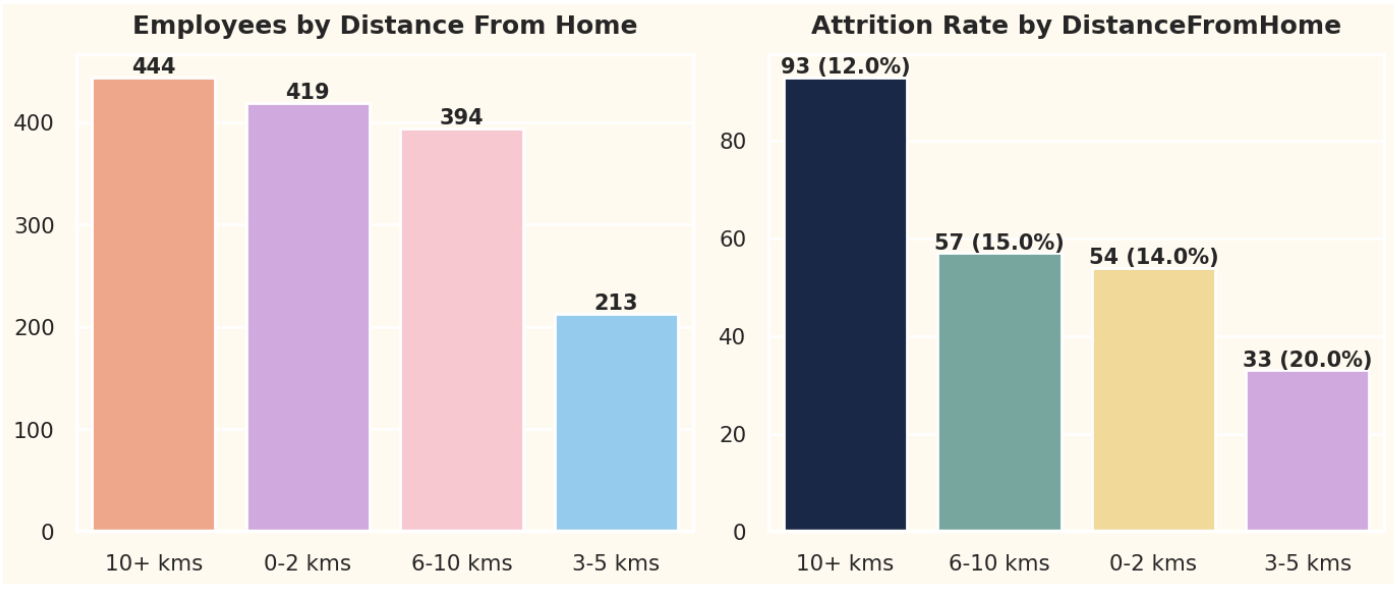
```
Total Unique Values in Attribute is => 29
```

```
df["DistanceFromHome"].describe().to_frame()
```

	DistanceFromHome
count	1470.000000
mean	9.192517
std	8.106864
min	1.000000
25%	2.000000
50%	7.000000
75%	14.000000
max	29.000000

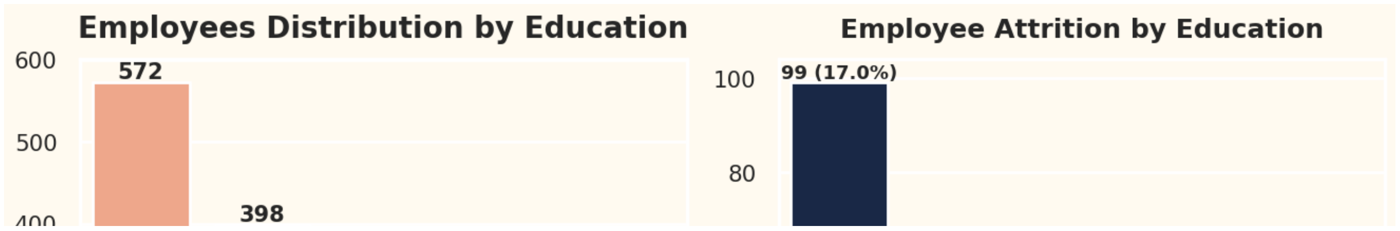
```
# Define the bin edges for the groups
bin_edges = [0,2,5,10,30]

# Define the labels for the groups
bin_labels = ['0-2 kms', '3-5 kms', '6-10 kms',"10+ kms"]

# Cuttinf the DistaanceFromHome column into groups
df['DistanceGroup'] = pd.cut(df['DistanceFromHome'], bins=bin_edges, labels=bin_labels)
```

```
##Visualization to show Total Employees by DistnaceFromHome.
plt.figure(figsize=(14,6))
plt.subplot(1,2,1)
value_1 = df["DistanceGroup"].value_counts()
sns.barplot(x=value_1.index.tolist(), y=value_1.values,palette = ["#FFA07A", "#D4A1E7", "#FFC0CB","#87CEFA"])
plt.title("Employees by Distance From Home",fontweight="black",pad=15,size=18)
for index, value in enumerate(value_1.values):
    plt.text(index,value,value,ha="center",va="bottom",fontweight="black",size=15)

    #Visualization to show Attrition Rate by DistanceFromHome.
plt.subplot(1,2,2)
new_df = df[df["Attrition"]=="Yes"]
value_2 = new_df["DistanceGroup"].value_counts()
attrition_rate = np.floor((value_2/value_1)*100).values
sns.barplot(x=value_2.index.tolist(),y= value_2.values,palette=["#11264e","#6faea4","#FEE08B","#D4A1E7","#E7A1A1"])
plt.title("Attrition Rate by DistanceFromHome",fontweight="black",pad=15,size=18)
for index,value in enumerate(value_2.values):
    plt.text(index,value, str(value)+" ("+str(attrition_rate[index])+"%)",ha="center",va="bottom",fontweight="black",size=15)

plt.tight_layout()
plt.show()
```



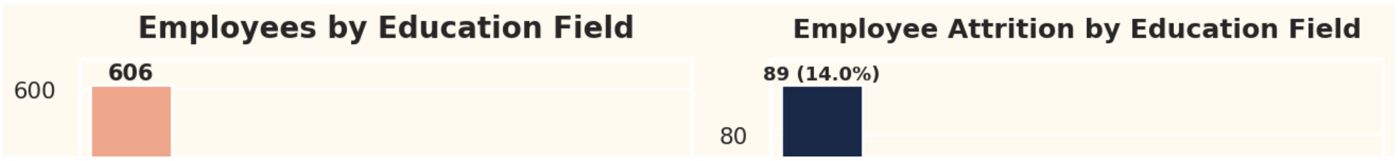
1. In the organization there are all kind of employees staying close or staying far from the office.
2. The feature Distance From Home doesn't follows any trend in attrition rate.
3. Employees staying close to the organization are mostly leaving compared to employees staing far from the oragnization.

```
#Visualization to show Total Employees by Education.
plt.figure(figsize=(13.5,6))
plt.subplot(1,2,1)
value_1 = df["Education"].value_counts()
sns.barplot(x=value_1.index,y=value_1.values,order=value_1.index,palette = ["#FFA07A", "#D4A1E7", "#FFC0CB","#87CEFA"])
plt.title("Employees Distribution by Education",fontweight="black",size=20,pad=15)
for index,value in enumerate(value_1.values):
    plt.text(index,value,value,ha="center",va="bottom",fontweight="black",size=15)

    #Visualization to show Employee Attrition by Education.
plt.subplot(1,2,2)
value_2 = new_df["Education"].value_counts()
attrition_rate = np.floor((value_2/value_1)*100).values
sns.barplot(x=value_2.index,y=value_2.values,order=value_2.index,palette=["#11264e","#6faea4","#FEE08B","#D4A1E7","#E7A1A1"])
plt.title("Employee Attrition by Education",fontweight="black",size=18,pad=15)
for index,value in enumerate(value_2.values):
    plt.text(index,value,str(value)+" ("+str(attrition_rate[index])+"%)",ha="center",va="bottom",
             fontweight="black",size=13)
plt.tight_layout()
plt.show()
```



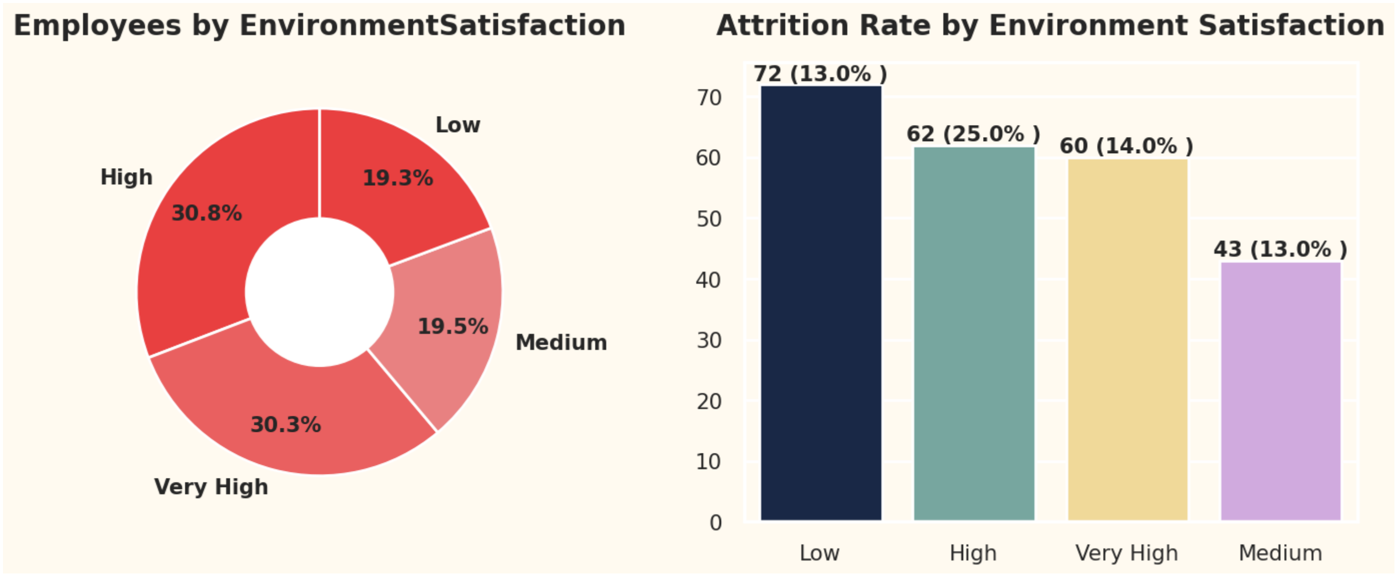
1. Most of the employees in the organization have completed Bachelors or Masters as their education qualification.
2. Very few employees in the organization have completed Doctorate degree as their education qualification.
3. We can observe a trend of decreasisng in attrition rate as the education qualification increases.

```
#Visualization to show Total Employees by Education Field.
plt.figure(figsize=(13.5,8))
plt.subplot(1,2,1)
value_1 = df["EducationField"].value_counts()
sns.barplot(x=value_1.index, y=value_1.values,order=value_1.index,palette = ["#FFA07A", "#D4A1E7", "#FFC0CB","#87CEFA"])
plt.title("Employees by Education Field",fontweight="black",size=20,pad=15)
for index,value in enumerate(value_1.values):
    plt.text(index,value,value,ha="center",va="bottom",fontweight="black",size=15)
plt.xticks(rotation=90)


    #Visualization to show Employee Attrition by Education Field.

plt.subplot(1,2,2)
value_2 = new_df["EducationField"].value_counts()
attrition_rate = np.floor((value_2/value_1)*100).values
sns.barplot(x=value_2.index,y=value_2.values,order=value_2.index,palette=["#11264e","#6faea4","#FEE08B","#D4A1E7"])
plt.title("Employee Attrition by Education Field",fontweight="black",size=18,pad=15)
for index,value in enumerate(value_2.values):
    plt.text(index,value,str(value)+" ("+str(attrition_rate[index])+"%)",ha="center",va="bottom",
             fontweight="black",size=13)
plt.xticks(rotation=90)
plt.tight_layout()
plt.show()
```



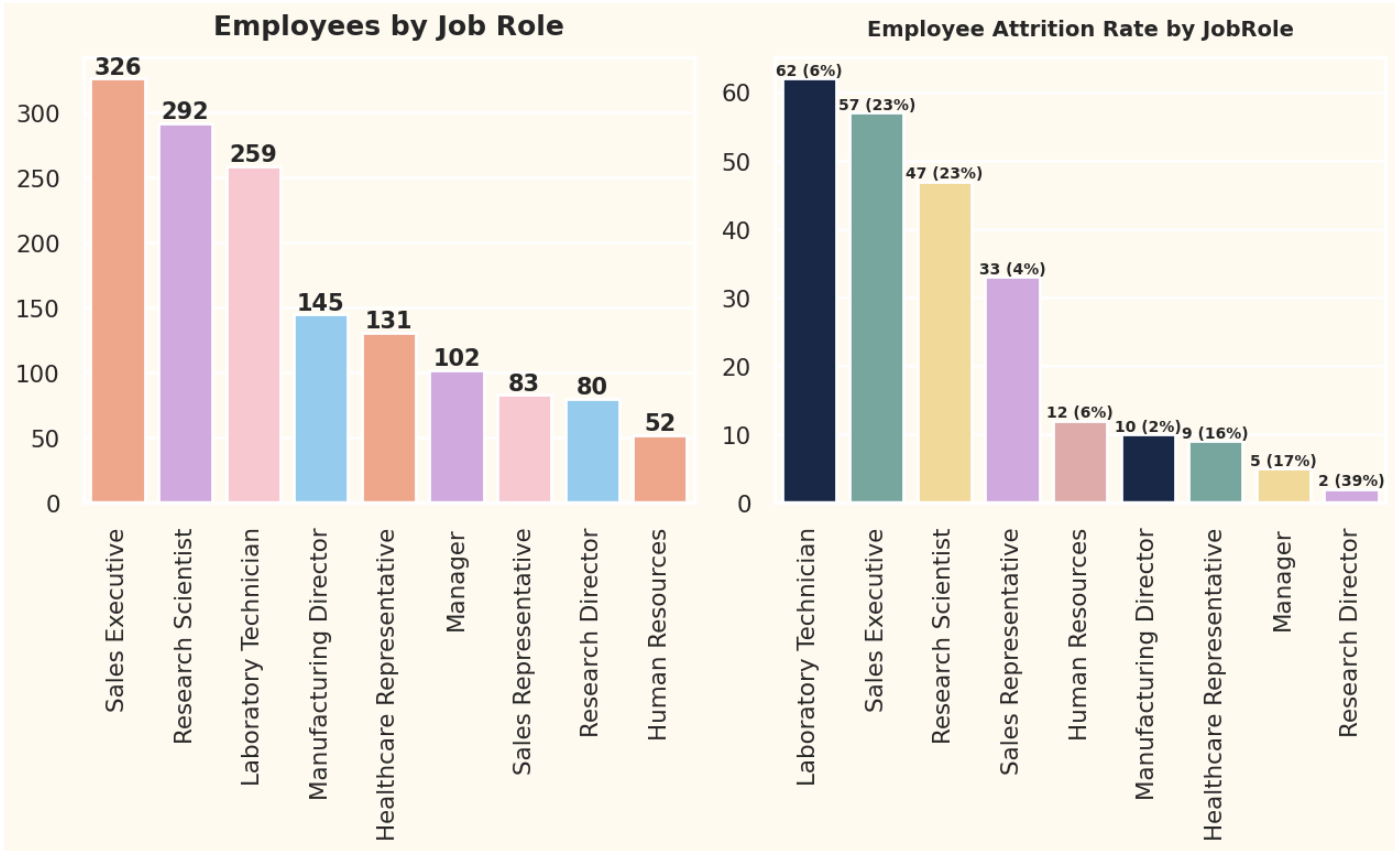
1. Most of the employees are either from Life Science or Medical Education Field.
2. Very few employees are from Human Resources Education Field.
3. Education Fields like Human Resources, Marketing, Technical is having very high attrition rate.
4. This may be because of work load becuase there are very few employees in these education fields compared to education field with less attrition rate.

```
#Visualization to show Total Employees by EnvironmentSatisfaction.
plt.figure(figsize=(14,6))
plt.subplot(1,2,1)
value_1 = df["EnvironmentSatisfaction"].value_counts()
plt.title("Employees by EnvironmentSatisfaction", fontweight="black", size=20, pad=20)
plt.pie(value_1.values, labels=value_1.index, autopct="%.1f%%",pctdistance=0.75,startangle=90,
        colors=['#E84040', '#E96060', '#E88181'],textprops={"fontweight":"black","size":15})
center_circle = plt.Circle((0, 0), 0.4, fc='white')
fig = plt.gcf()
fig.gca().add_artist(center_circle)


#Visualization to show Attrition Rate by EnvironmentSatisfaction.
plt.subplot(1,2,2)
new_df = df[df["Attrition"]=="Yes"]
value_2 = new_df["EnvironmentSatisfaction"].value_counts()
attrition_rate = np.floor((value_2/value_1)*100).values
sns.barplot(x=value_2.index,y=value_2.values,order=value_2.index,palette=["#11264e","#6faea4","#FEE08B","#D4A1E7","#E7A1A1"])
plt.title("Attrition Rate by Environment Satisfaction",fontweight="black",size=20,pad=20)
for index,value in enumerate(value_2):
    plt.text(index,value,str(value)+" ("+str(attrition_rate[index])+"% )",ha="center",va="bottom",
            size=15,fontweight="black")
plt.tight_layout()
plt.show()
```



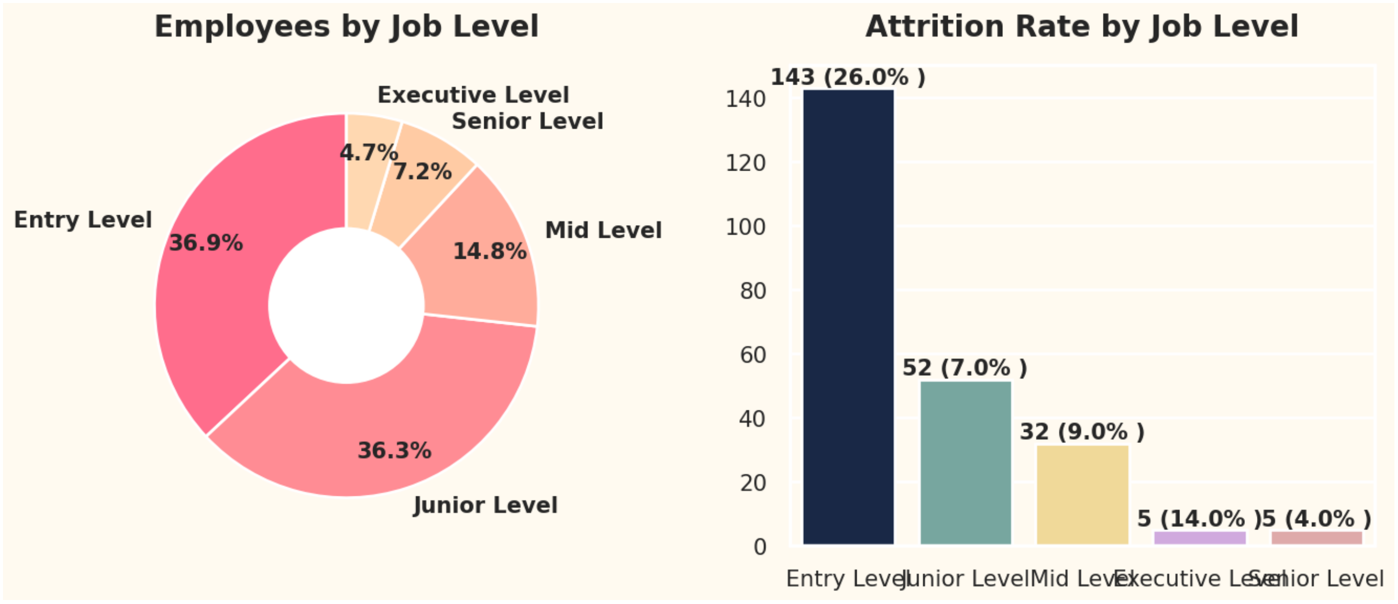
1. Most of the employees have rated the organization environment satisfaction High & Very High.
2. Though the organization environment satisfaction is high still there's very high attriton in this envirnoment.
3. Attrition Rate increases with increase in level of environment satisfication.

```
##Visualization to show Total Employees by JobRole.

plt.figure(figsize=(13,8))
plt.subplot(1,2,1)
value_1 = df["JobRole"].value_counts()
sns.barplot(x=value_1.index.tolist(), y=value_1.values,palette = ["#FFA07A", "#D4A1E7", "#FFC0CB","#87CEFA"])
plt.title("Employees by Job Role",fontweight="black",pad=15,size=18)
plt.xticks(rotation=90)
for index, value in enumerate(value_1.values):
    plt.text(index,value,value,ha="center",va="bottom",fontweight="black",size=15)


    #Visualization to show Attrition Rate by JobRole.

plt.subplot(1,2,2)
new_df = df[df["Attrition"]=="Yes"]
value_2 = new_df["JobRole"].value_counts()
attrition_rate = np.floor((value_2/value_1)*100).values
sns.barplot(x=value_2.index.tolist(), y=value_2.values,palette=["#11264e","#6faea4","#FEE08B","#D4A1E7","#E7A1A1"])
plt.title("Employee Attrition Rate by JobRole",fontweight="black",pad=14,size=14)
plt.xticks(rotation=90)
for index,value in enumerate(value_2.values):
    plt.text(index,value, str(value)+" ("+str(int(attrition_rate[index]))+"%)",ha="center",va="bottom",
             fontweight="black",size=10)

plt.tight_layout()
plt.show()
```



1. Most employees is working as Sales executive, Research Scientist or Laboratory Technician. in this organization.
2. Highest attrition rates are in sector of Research Director, Sales Executive, Research Scientist.

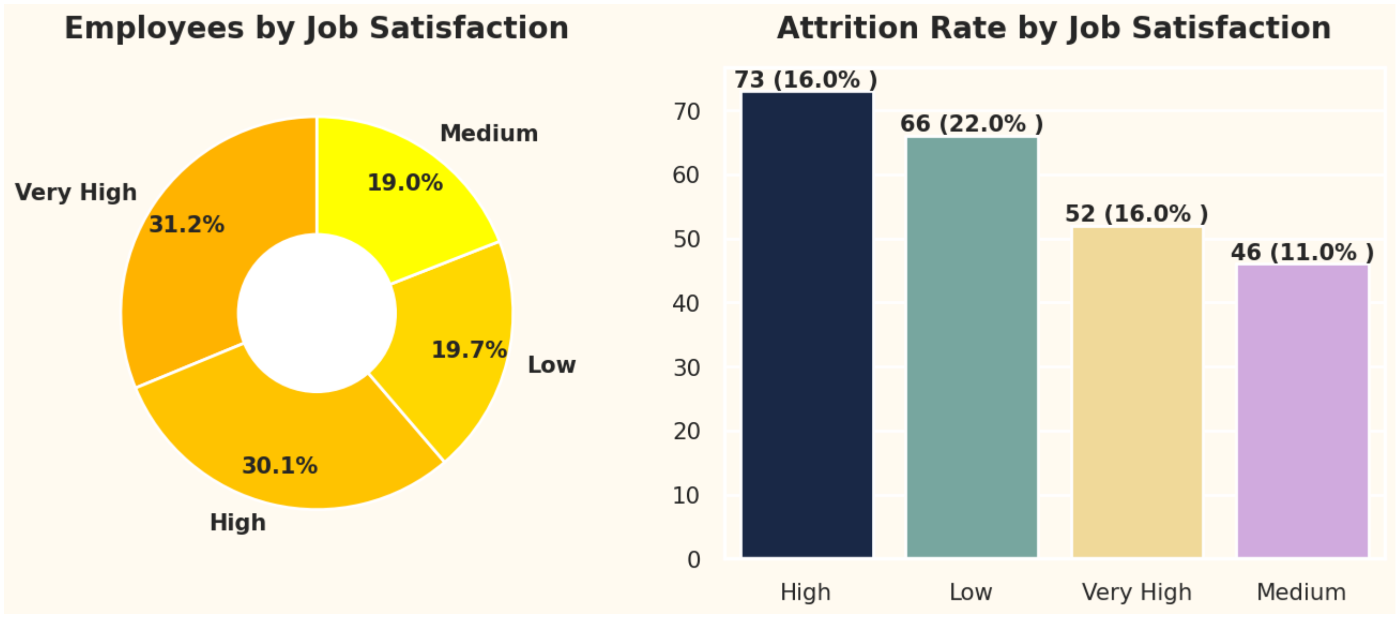
```
     #Visualization to show Total Employees by Job Level.

plt.figure(figsize=(14,6))
plt.subplot(1,2,1)
value_1 = df["JobLevel"].value_counts()
plt.title("Employees by Job Level", fontweight="black", size=20, pad=20)
plt.pie(value_1.values, labels=value_1.index, autopct="%.1f%%",pctdistance=0.8,startangle=90,
        colors=['#FF6D8C', '#FF8C94', '#FFAC9B', '#FFCBA4',"#FFD8B1"],textprops={"fontweight":"black","size":15})
center_circle = plt.Circle((0, 0), 0.4, fc='white')
fig = plt.gcf()
fig.gca().add_artist(center_circle)


     #Visualization to show Attrition Rate by JobLevel.

plt.subplot(1,2,2)
new_df = df[df["Attrition"]=="Yes"]
value_2 = new_df["JobLevel"].value_counts()
attrition_rate = np.floor((value_2/value_1)*100).values
sns.barplot(x=value_2.index,y=value_2.values,order=value_2.index,palette=["#11264e","#6faea4","#FEE08B","#D4A1E7","#E7A1A1"])
plt.title("Attrition Rate by Job Level",fontweight="black",size=20,pad=20)
for index,value in enumerate(value_2):
    plt.text(index,value,str(value)+" ("+str(attrition_rate[index])+"% )",ha="center",va="bottom",
             size=15,fontweight="black")
plt.tight_layout()
plt.show()
```



1. Most of the employees in the organization are at Entry Level or Junior Level.
2. Highest Attrition is at the Entry Level.
3. As the level increases the attrition rate decreases.

```
   #Visualization to show Total Employees by Job Satisfaction.

plt.figure(figsize=(14,6))
plt.subplot(1,2,1)
value_1 = df["JobSatisfaction"].value_counts()
plt.title("Employees by Job Satisfaction", fontweight="black", size=20, pad=20)
plt.pie(value_1.values, labels=value_1.index, autopct="%.1f%%",pctdistance=0.8,startangle=90,
        colors=['#FFB300', '#FFC300', '#FFD700', '#FFFF00'],textprops={"fontweight":"black","size":15})
center_circle = plt.Circle((0, 0), 0.4, fc='white')
```

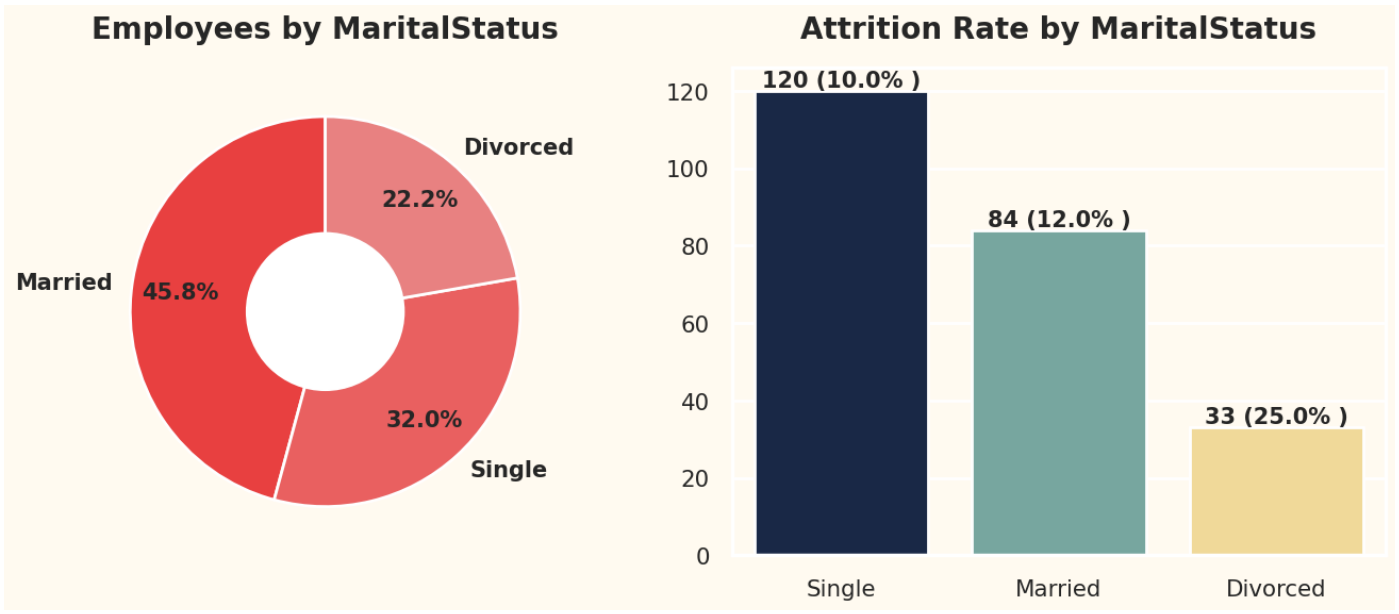
```
fig = plt.gcf()
fig.gca().add_artist(center_circle)

      #Visualization to show Attrition Rate by Job Satisfaction.

plt.subplot(1,2,2)
new_df = df[df["Attrition"]=="Yes"]
value_2 = new_df["JobSatisfaction"].value_counts()
attrition_rate = np.floor((value_2/value_1)*100).values
sns.barplot(x=value_2.index,y=value_2.values,order=value_2.index,palette=["#11264e","#6faea4","#FEE08B","#D4A1E7","#E7A1A1"])
plt.title("Attrition Rate by Job Satisfaction",fontweight="black",size=20,pad=20)
for index,value in enumerate(value_2):
    plt.text(index,value,str(value)+" ("+str(attrition_rate[index])+"% )",ha="center",va="bottom",
             size=15,fontweight="black")
plt.tight_layout()
plt.show()
```



1. Most of the employees have rated their job satisfaction as high or very high.
2. Employees who rated their job satisfaction low are mostly leaving the organization.
3. All the categories in job satisfaction is having high attrition rate.

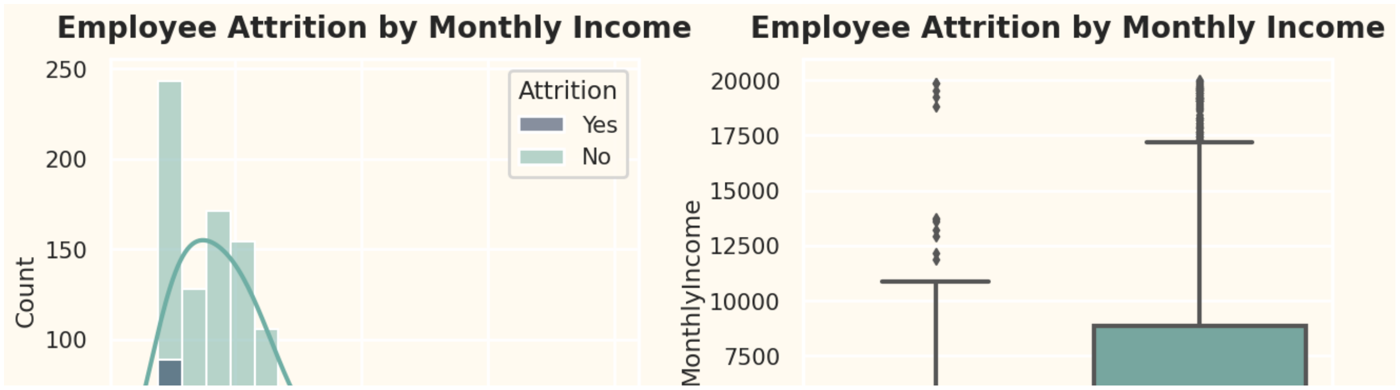
```
     #Visualization to show Total Employees by MaritalStatus.

plt.figure(figsize=(14,6))
plt.subplot(1,2,1)
value_1 = df["MaritalStatus"].value_counts()
plt.title("Employees by MaritalStatus", fontweight="black", size=20, pad=20)
plt.pie(value_1.values, labels=value_1.index, autopct="%.1f%%",pctdistance=0.75,startangle=90,
        colors=['#E84040', '#E96060', '#E88181', '#E7A1A1'],textprops={"fontweight":"black","size":15})
center_circle = plt.Circle((0, 0), 0.4, fc='white')
fig = plt.gcf()
fig.gca().add_artist(center_circle)


    #Visualization to show Attrition Rate by MaritalStatus.

plt.subplot(1,2,2)
new_df = df[df["Attrition"]=="Yes"]
value_2 = new_df["MaritalStatus"].value_counts()
attrition_rate = np.floor((value_2/value_1)*100).values
sns.barplot(x=value_2.index, y=value_2.values,palette=["#11264e","#6faea4","#FEE08B","#D4A1E7","#E7A1A1"])
```

```
plt.title("Attrition Rate by MaritalStatus",
          fontweight="black",
          size=20,pad=20)
for index,value in enumerate(value_2):
    plt.text(index,value,str(value)+" ("+str(attrition_rate[index])+"% )",ha="center",va="bottom",
             size=15,fontweight="black")
plt.tight_layout()
plt.show()
```



1. Most of the employees are Married in the organization.
2. The attrition rate is very high of employees who are divorced.
3. The attrition rate is low for employees who are single.

```
  #Visualization to show Employee Distribution by MonthlyIncome.

plt.figure(figsize=(13,6))
plt.subplot(1,2,1)
sns.histplot(x="MonthlyIncome", hue="Attrition", kde=True ,data=df,palette=["#11264e","#6faea4"])
plt.title("Employee Attrition by Monthly Income",fontweight="black",size=20,pad=15)

  #Visualization to show Employee Attrition by Monthly Income.
plt.subplot(1,2,2)
sns.boxplot(x="Attrition",y="MonthlyIncome",data=df,palette=["#D4A1E7","#6faea4"])
plt.title("Employee Attrition by Monthly Income",fontweight="black",size=20,pad=15)
plt.tight_layout()
plt.show()
```



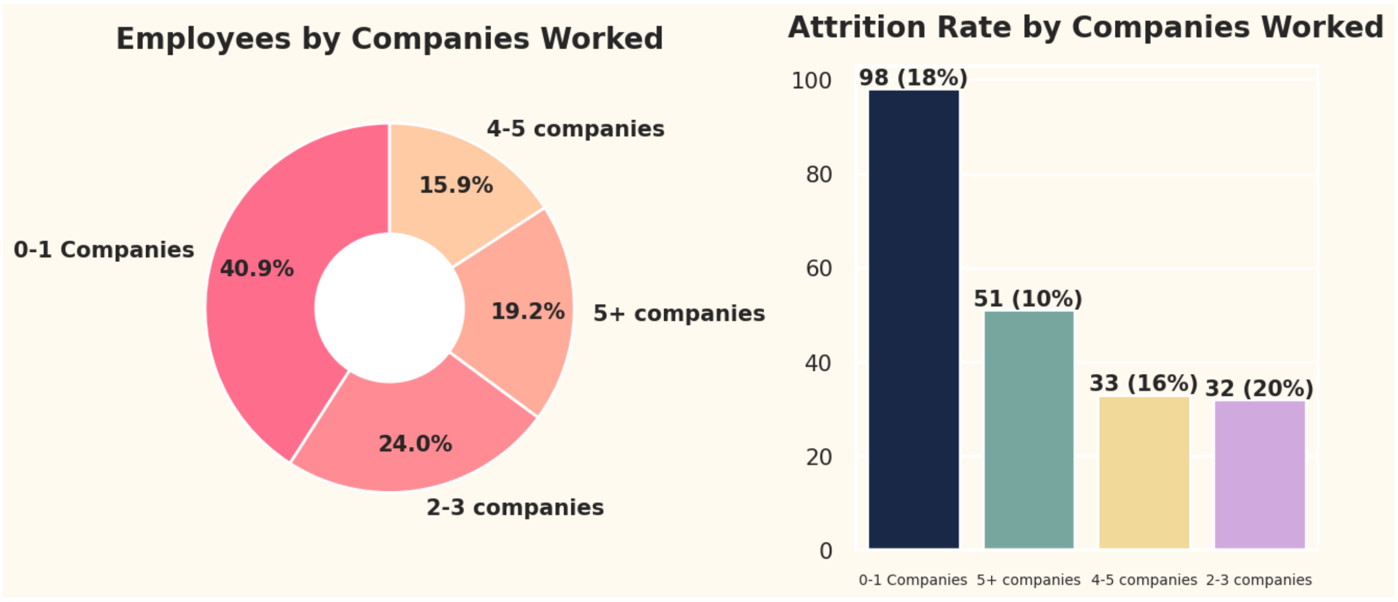
1. Most of the employees are getting paid less than 10000 in the organiation.
2. The average monthly income of employee who have left is comparatively low with employee who are still working.
3. As the Monthly Income increases the attrition decreases.

```
    #Employee Attrition by Number of Companies Worked.

df["NumCompaniesWorked"].describe().to_frame()
```

	NumCompaniesWorked
count	1470.000000
mean	2.693197
std	2.498009
min	0.000000
25%	1.000000
50%	2.000000
75%	4.000000
max	9.000000

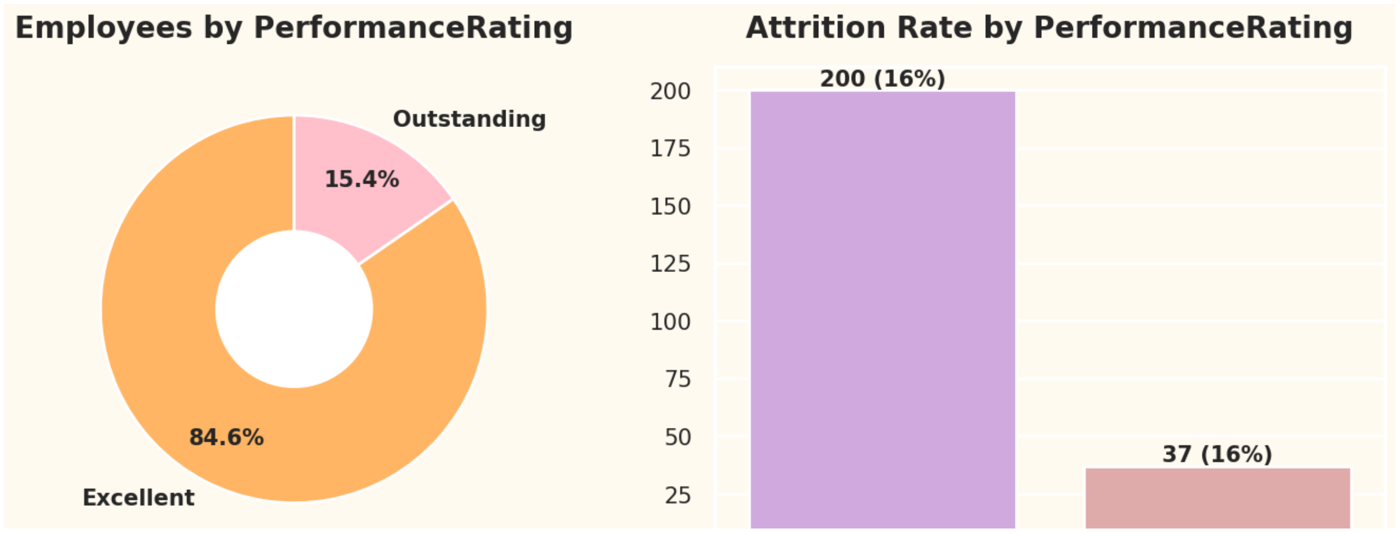
```
# Define the bin edges for the groups
bin_edges = [0, 1, 3, 5, 10]

# Define the labels for the groups
bin_labels = ['0-1 Companies', '2-3 companies', '4-5 companies', "5+ companies"]

# Cut the DailyRate column into groups
df["NumCompaniesWorkedGroup"] = pd.cut(df['NumCompaniesWorked'], bins=bin_edges, labels=bin_labels)
```

```
#Visualization to show Total Employees by NumCompaniesWorked.

plt.figure(figsize=(13,6))
plt.subplot(1,2,1)
value_1 = df["NumCompaniesWorkedGroup"].value_counts()
plt.title("Employees by Companies Worked", fontweight="black", size=20, pad=20)
plt.pie(value_1.values, labels=value_1.index, autopct="%.1f%%",pctdistance=0.75,startangle=90,
        colors=['#FF6D8C', '#FF8C94', '#FFAC9B', '#FFCBA4'],textprops={"fontweight":"black","size":15})
center_circle = plt.Circle((0, 0), 0.4, fc='white')
fig = plt.gcf()
fig.gca().add_artist(center_circle)

#Visualization to show Attrition Rate by NumCompaniesWorked.

plt.subplot(1,2,2)
new_df = df[df["Attrition"]=="Yes"]
value_2 = new_df["NumCompaniesWorkedGroup"].value_counts()
attrition_rate = np.floor((value_2/value_1)*100).values
sns.barplot(x=value_2.index.tolist(), y=value_2.values,palette=["#11264e","#6faea4","#FEE08B","#D4A1E7","#E7A1A1"])
plt.title("Attrition Rate by Companies Worked",fontweight="black",size=20,pad=20)
for index,value in enumerate(value_2):
    plt.text(index,value,str(value)+" ("+str(int(attrition_rate[index]))+"%)",ha="center",va="bottom",
             size=15,fontweight="black")
plt.xticks(size=10)
plt.tight_layout()
plt.show()
```



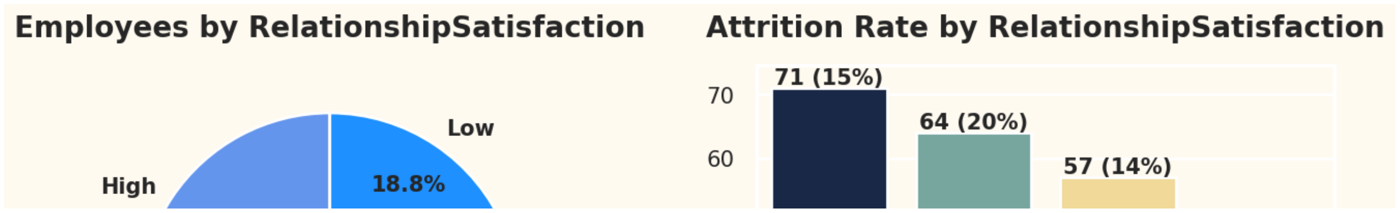
1. Most of the employees have worked for less than 2 companies.
2. There's a high attrition rate of employees who haved for less than 5 companies.

```
#Visualization to show Total Employees by PerformanceRating.

plt.figure(figsize=(14,6))
plt.subplot(1,2,1)
value_1 = df["PerformanceRating"].value_counts()
plt.title("Employees by PerformanceRating", fontweight="black", size=20, pad=20)
plt.pie(value_1.values, labels=value_1.index, autopct="%.1f%%",pctdistance=0.75,startangle=90,
        colors=["#ffb563","#FFC0CB"],textprops={"fontweight":"black","size":15})
center_circle = plt.Circle((0, 0), 0.4, fc='white')
fig = plt.gcf()
fig.gca().add_artist(center_circle)


#Visualization to show Attrition Rate by PerformanceRating.

plt.subplot(1,2,2)
new_df = df[df["Attrition"]=="Yes"]
value_2 = new_df["PerformanceRating"].value_counts()
attrition_rate = np.floor((value_2/value_1)*100).values
sns.barplot(x=value_2.index.tolist(),y= value_2.values,palette=["#D4A1E7","#E7A1A1"])
plt.title("Attrition Rate by PerformanceRating",fontweight="black",size=20,pad=20)
for index,value in enumerate(value_2):
    plt.text(index,value,str(value)+" ("+str(int(attrition_rate[index]))+"%)",ha="center",va="bottom",
             size=15,fontweight="black")
plt.tight_layout()
plt.show()
```

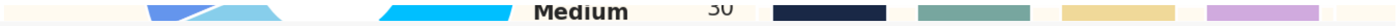


1. Most of the employees are having excellent performance rating.
2. Both the categories in this field is having same attriton rate.
3. That's why we can't generate any meaningful inisghts

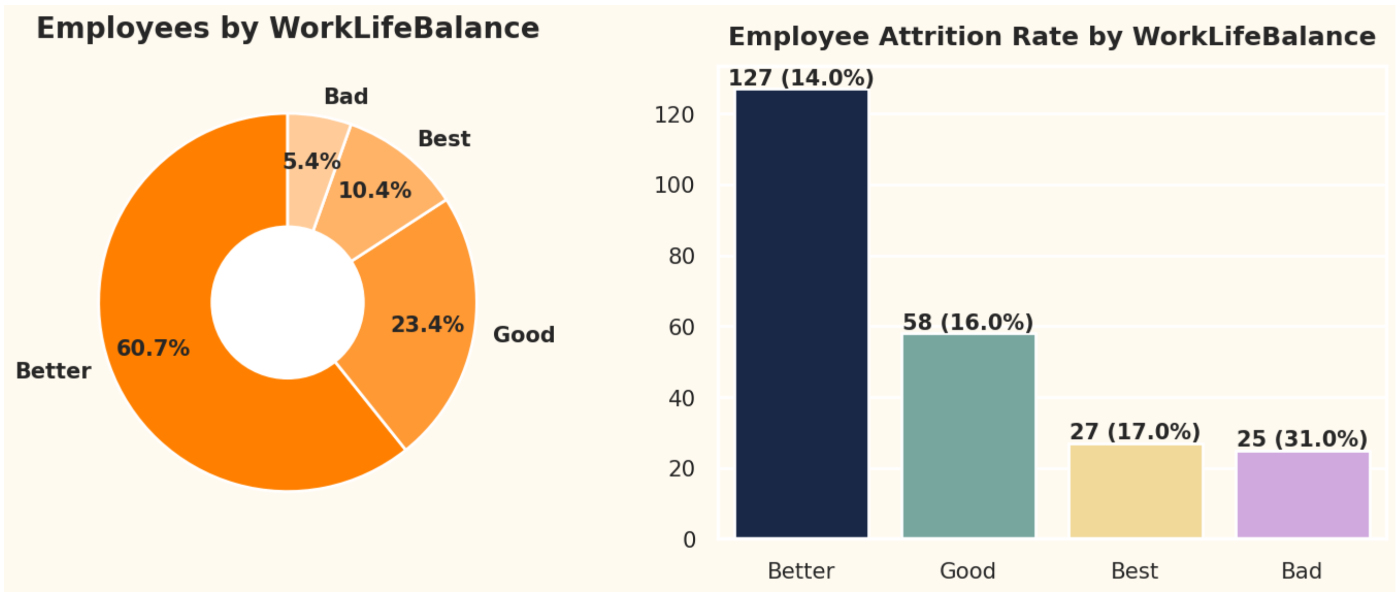
```
#Visualization to show Total Employees by RelationshipSatisfaction.

plt.figure(figsize=(13,6))
plt.subplot(1,2,1)
value_1 = df["RelationshipSatisfaction"].value_counts()
plt.title("Employees by RelationshipSatisfaction", fontweight="black", size=20, pad=20)
plt.pie(value_1.values, labels=value_1.index, autopct="%.1f%%",pctdistance=0.75,startangle=90,
        colors=['#6495ED', '#87CEEB', '#00BFFF', '#1E90FF'],textprops={"fontweight":"black","size":15})
center_circle = plt.Circle((0, 0), 0.4, fc='white')
fig = plt.gcf()
fig.gca().add_artist(center_circle)

#Visualization to show Attrition Rate by RelationshipSatisfaction.

plt.subplot(1,2,2)
new_df = df[df["Attrition"]=="Yes"]
value_2 = new_df["RelationshipSatisfaction"].value_counts()
attrition_rate = np.floor((value_2/value_1)*100).values
sns.barplot(x=value_2.index, y=value_2.values,order=value_2.index,palette=["#11264e","#6faea4","#FEE08B","#D4A1E7","#E7A1A1"])
plt.title("Attrition Rate by RelationshipSatisfaction",fontweight="black",size=20,pad=20)
for index,value in enumerate(value_2):
    plt.text(index,value,str(value)+" ("+str(int(attrition_rate[index]))+"%)",ha="center",va="bottom",
             size=15,fontweight="black")
plt.tight_layout()
plt.show()
```



1. Most of the employees are having high or very high relationship satisfaction.
2. Though the relationship satification is high there's a high attrition rate.
3. All the categories in this feature is having a high attriton rate.



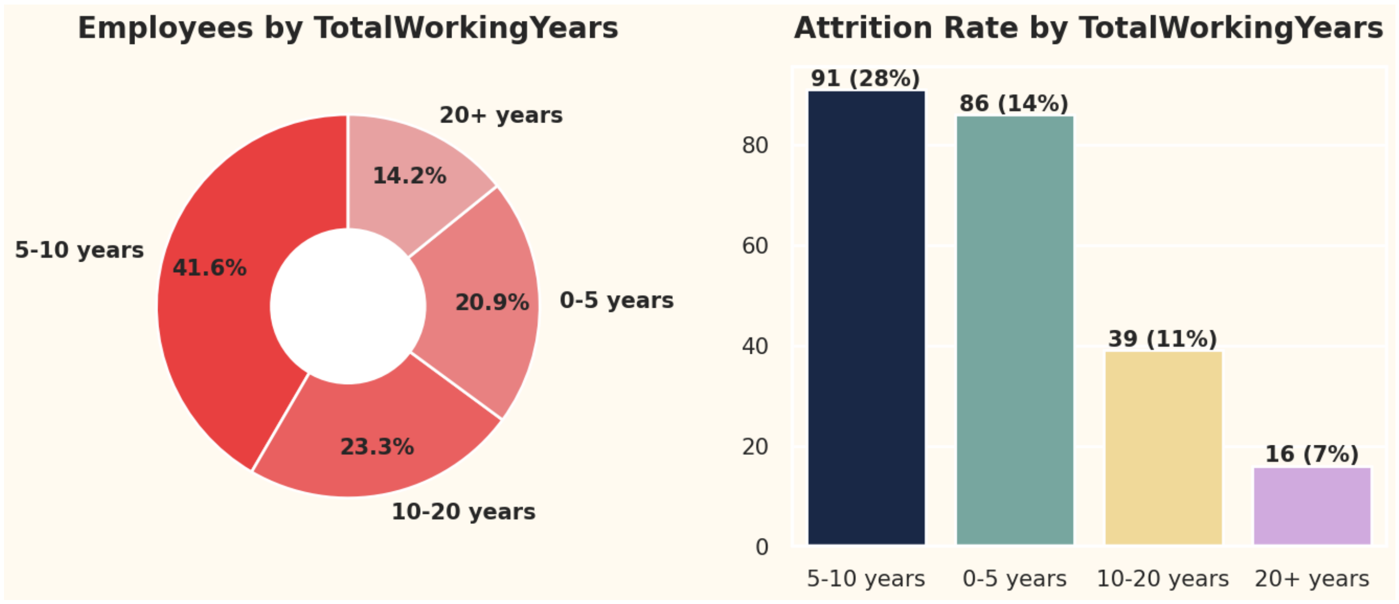
```
#Visualization to show Total Employees by WorkLifeBalance.

plt.figure(figsize=(14.5,6))
plt.subplot(1,2,1)
value_1 = df["WorkLifeBalance"].value_counts()
plt.title("Employees by WorkLifeBalance", fontweight="black", size=20, pad=20)
plt.pie(value_1.values, labels=value_1.index, autopct="%.1f%%",pctdistance=0.75,startangle=90,
        colors= ['#FF8000', '#FF9933', '#FFB366', '#FFCC99'],textprops={"fontweight":"black","size":15})
center_circle = plt.Circle((0, 0), 0.4, fc='white')
fig = plt.gcf()
fig.gca().add_artist(center_circle)

#Visualization to show Attrition Rate by WorkLifeBalance.

plt.subplot(1,2,2)
new_df = df[df["Attrition"]=="Yes"]
value_2 = new_df["WorkLifeBalance"].value_counts()
attrition_rate = np.floor((value_2/value_1)*100).values
sns.barplot(x=value_2.index, y=value_2.values,order=value_2.index,palette=["#11264e","#6faea4","#FEE08B","#D4A1E7","#E7A1A1"])
plt.title("Employee Attrition Rate by WorkLifeBalance",fontweight="black",pad=15,size=18)
for index,value in enumerate(value_2.values):
    plt.text(index,value, str(value)+" ("+str(attrition_rate[index])+"%)",ha="center",va="bottom",
             fontweight="black",size=15)
plt.tight_layout()
plt.show()
```



1. More than 60% of employees are having a better work life balance.
2. Employees with Bad Work Life Balance is having Very High Attrition Rate.
3. Other Categories is also having High attriton Rate.

```
# Define the bin edges for the groups
bin_edges = [0, 5, 10, 20, 50]

# Define the labels for the groups
bin_labels = ['0-5 years', '5-10 years', '10-20 years', "20+ years"]

# Cut the DailyRate column into groups
df["TotalWorkingYearsGroup"] = pd.cut(df['TotalWorkingYears'], bins=bin_edges, labels=bin_labels)
```

```
#Visualization to show Total Employees by TotalWorkingYearsGroup.

plt.figure(figsize=(14,6))
plt.subplot(1,2,1)
value_1 = df["TotalWorkingYearsGroup"].value_counts()
plt.title("Employees by TotalWorkingYears", fontweight="black", size=20, pad=20)
plt.pie(value_1.values, labels=value_1.index, autopct="%.1f%%",pctdistance=0.75,startangle=90,
        colors=['#E84040', '#E96060', '#E88181', '#E7A1A1'],textprops={"fontweight":"black","size":15})
center_circle = plt.Circle((0, 0), 0.4, fc='white')
fig = plt.gcf()
fig.gca().add_artist(center_circle)

#Visualization to show Attrition Rate by TotalWorkingYearsGroup.

plt.subplot(1,2,2)
new_df = df[df["Attrition"]=="Yes"]
value_2 = new_df["TotalWorkingYearsGroup"].value_counts()
attrition_rate = np.floor((value_2/value_1)*100).values
sns.barplot(x=value_2.index.tolist(), y=value_2.values,palette=["#11264e","#6faea4","#FEE08B","#D4A1E7","#E7A1A1"])
plt.title("Attrition Rate by TotalWorkingYears",fontweight="black",size=20,pad=20)
for index,value in enumerate(value_2):
    plt.text(index,value,str(value)+" ("+str(int(attrition_rate[index]))+"%)",ha="center",va="bottom",
             size=15,fontweight="black")
plt.tight_layout()
plt.show()
```

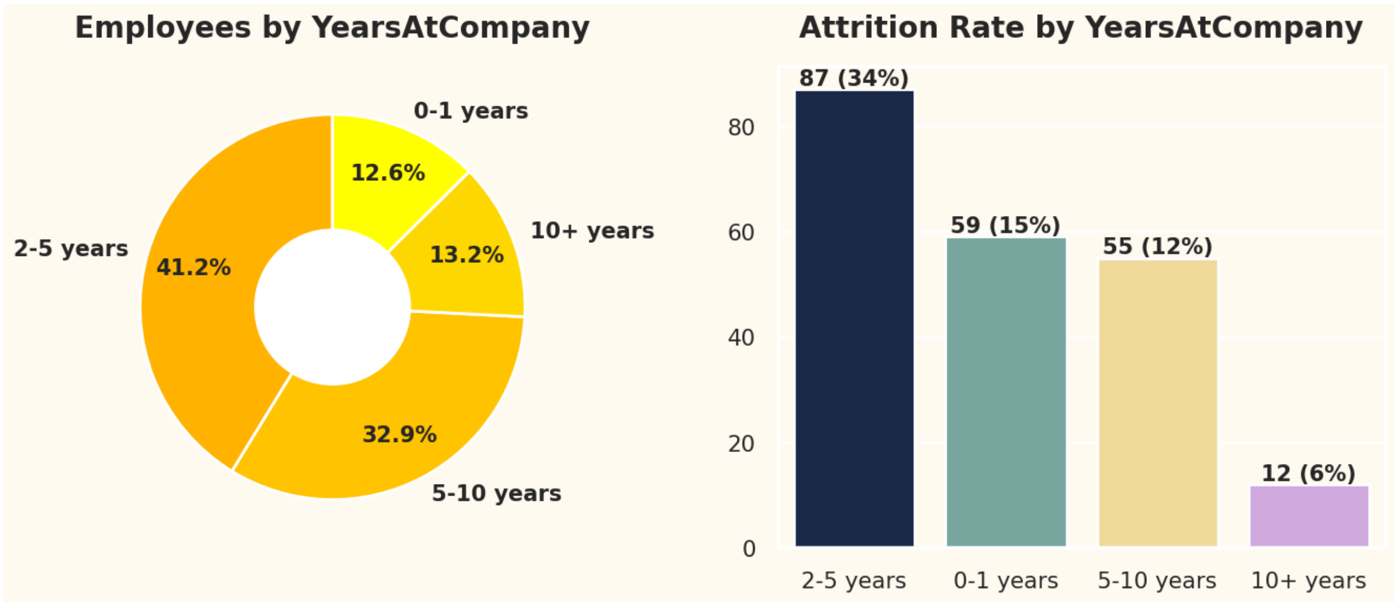


1. Most of the employees are having a total of 5 to 10 years of working experience. But their Attrition Rate is also **very high.
2. Employee with working experience of less than 10 years are having High Attrition Rate.
3. Employee with working experience of more than 10 years are having Less Attrition Rate.

```
# Define the bin edges for the groups
bin_edges = [0, 1, 5, 10, 20]
```

```
# Define the labels for the groups
bin_labels = ['0-1 years', '2-5 years', '5-10 years', "10+ years"]

# Cut the DailyRate column into groups
df["YearsAtCompanyGroup"] = pd.cut(df['YearsAtCompany'], bins=bin_edges, labels=bin_labels)
```

```
#Visualization to show Total Employees by YearsAtCompanyGroup.

plt.figure(figsize=(14,6))
plt.subplot(1,2,1)
value_1 = df["YearsAtCompanyGroup"].value_counts()
plt.title("Employees by YearsAtCompany", fontweight="black", size=20, pad=20)
plt.pie(value_1.values, labels=value_1.index, autopct="%.1f%%",pctdistance=0.75,startangle=90,
        colors=['#FFB300', '#FFC300', '#FFD700', '#FFFF00'],textprops={"fontweight":"black","size":15})
center_circle = plt.Circle((0, 0), 0.4, fc='white')
fig = plt.gcf()
fig.gca().add_artist(center_circle)


#Visualization to show Attrition Rate by YearsAtCompanyGroup.

plt.subplot(1,2,2)
new_df = df[df["Attrition"]=="Yes"]
value_2 = new_df["YearsAtCompanyGroup"].value_counts()
attrition_rate = np.floor((value_2/value_1)*100).values
sns.barplot(x=value_2.index.tolist(), y=value_2.values,palette=["#11264e","#6faea4","#FEE08B","#D4A1E7","#E7A1A1"])
plt.title("Attrition Rate by YearsAtCompany",fontweight="black",size=20,pad=20)
for index,value in enumerate(value_2):
    plt.text(index,value,str(value)+" ("+str(int(attrition_rate[index]))+"%)",ha="center",va="bottom",
             size=15,fontweight="black")
plt.tight_layout()
plt.show()
```

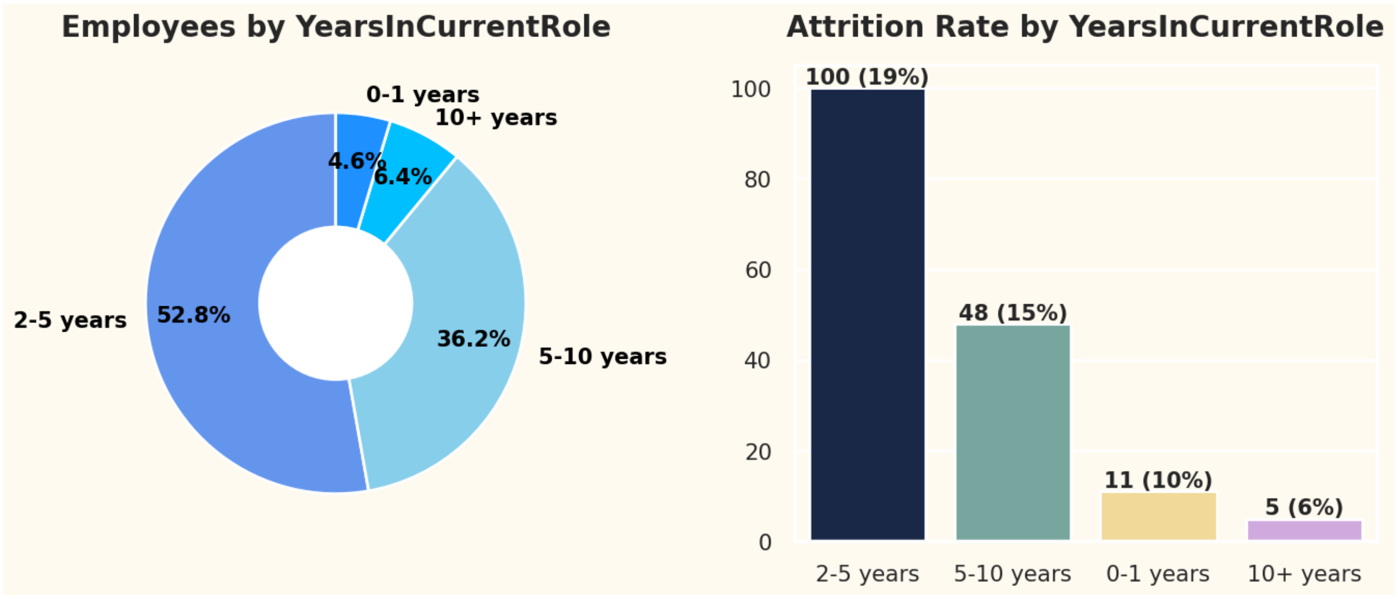


1. Most employees has worked for 2 to 10 years in the organization.
2. Very few employees has working for less than 1 year or more than 10 years.
3. Employee who have worked for 2-5 years are having very high attrition rate.
4. Employee who have worked for 10+ years are having low attrition rate.

```
# Define the bin edges for the groups
bin_edges = [0, 1, 5, 10, 20]
```

```
# Define the labels for the groups
bin_labels = ['0-1 years', '2-5 years', '5-10 years', "10+ years"]

# Cut the DailyRate column into groups
df["YearsInCurrentRoleGroup"] = pd.cut(df['YearsInCurrentRole'], bins=bin_edges, labels=bin_labels)
```

```
#Visualization to show Total Employees by YearsInCurrentRoleGroup.

plt.figure(figsize=(14,6))
plt.subplot(1,2,1)
value_1 = df["YearsInCurrentRoleGroup"].value_counts()
plt.title("Employees by YearsInCurrentRole", fontweight="black", size=20, pad=20)
plt.pie(value_1.values, labels=value_1.index, autopct="%.1f%%",pctdistance=0.75,startangle=90,
        colors=['#6495ED', '#87CEEB', '#00BFFF', '#1E90FF'],textprops={"fontweight":"black","size":15,"color":"black"})
center_circle = plt.Circle((0, 0), 0.4, fc='white')
fig = plt.gcf()
fig.gca().add_artist(center_circle)

#Visualization to show Attrition Rate by YearsInCurrentRoleGroup.

plt.subplot(1,2,2)
new_df = df[df["Attrition"]=="Yes"]
value_2 = new_df["YearsInCurrentRoleGroup"].value_counts()
attrition_rate = np.floor((value_2/value_1)*100).values
sns.barplot(x=value_2.index.tolist(), y=value_2.values,palette= ["#11264e","#6faea4","#FEE08B","#D4A1E7","#E7A1A1"])
plt.title("Attrition Rate by YearsInCurrentRole",fontweight="black",size=20,pad=20)
for index,value in enumerate(value_2):
    plt.text(index,value,str(value)+" ("+str(int(attrition_rate[index]))+"%)",ha="center",va="bottom",
             size=15,fontweight="black")
plt.tight_layout()
plt.show()
```

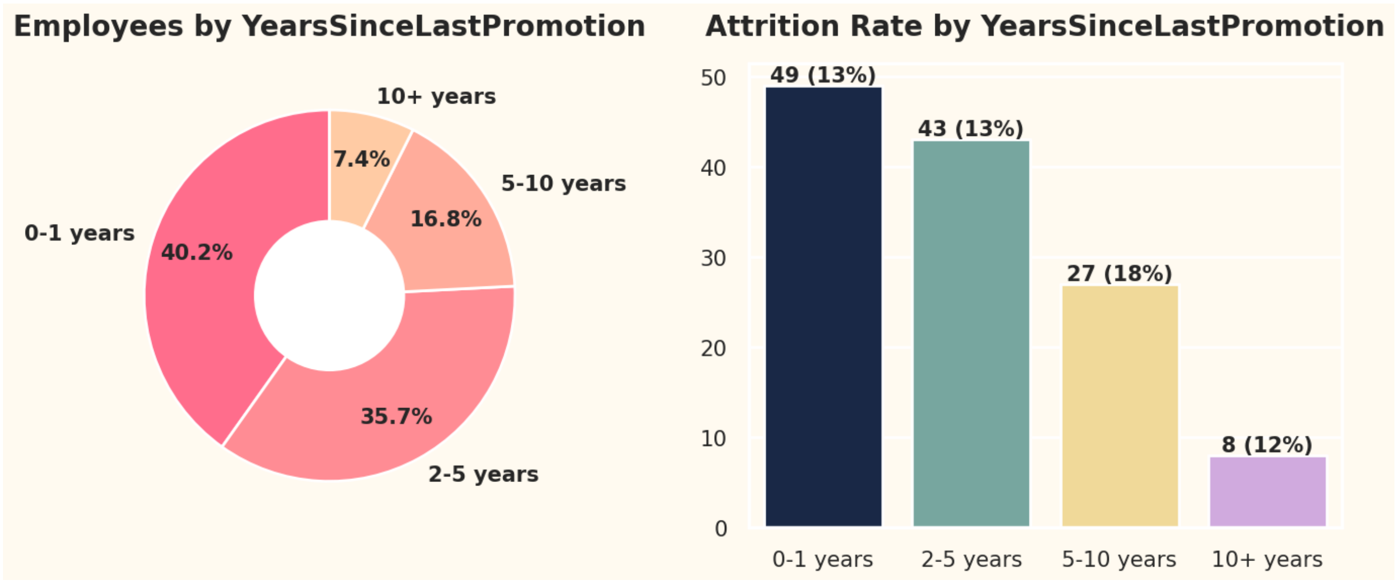


1. Most employees has worked for 2 to 10 years for the same role in the organization.
2. Very few employees has worked for less than 1 year or more than 10 years in the same role.
3. Employee who has worked for 2 to 0 years in the same role are having very high attrition rate.
4. Employee who has worked for 10+ years in the same role are having low attrition rate.

```
# Define the bin edges for the groups
bin_edges = [0, 1, 5, 10, 20]

# Define the labels for the groups
```

```
bin_labels = ['0-1 years', '2-5 years', '5-10 years', "10+ years"]

# Cut the DailyRate column into groups
df["YearsSinceLastPromotionGroup"] = pd.cut(df['YearsSinceLastPromotion'], bins=bin_edges, labels=bin_labels)
```

```
#Visualization to show Total Employees by YearsSinceLastPromotionGroup.

plt.figure(figsize=(14,6))
plt.subplot(1,2,1)
value_1 = df["YearsSinceLastPromotionGroup"].value_counts()
plt.title("Employees by YearsSinceLastPromotion", fontweight="black", size=20, pad=20)
plt.pie(value_1.values, labels=value_1.index, autopct="%.1f%%",pctdistance=0.75,startangle=90,
        colors=['#FF6D8C', '#FF8C94', '#FFAC9B', '#FFCBA4'],textprops={"fontweight":"black","size":15})
center_circle = plt.Circle((0, 0), 0.4, fc='white')
fig = plt.gcf()
fig.gca().add_artist(center_circle)

#Visualization to show Attrition Rate by YearsSinceLastPromotionGroup.

plt.subplot(1,2,2)
new_df = df[df["Attrition"]=="Yes"]
value_2 = new_df["YearsSinceLastPromotionGroup"].value_counts()
attrition_rate = np.floor((value_2/value_1)*100).values
sns.barplot(x=value_2.index.tolist(), y=value_2.values,palette=["#11264e","#6faea4","#FEE08B","#D4A1E7","#E7A1A1"])

plt.title("Attrition Rate by YearsSinceLastPromotion",fontweight="black",size=20,pad=20)
for index,value in enumerate(value_2):
    plt.text(index,value,str(value)+" ("+str(int(attrition_rate[index]))+"%)",ha="center",va="bottom",
             size=15,fontweight="black")
plt.tight_layout()
plt.show()
```

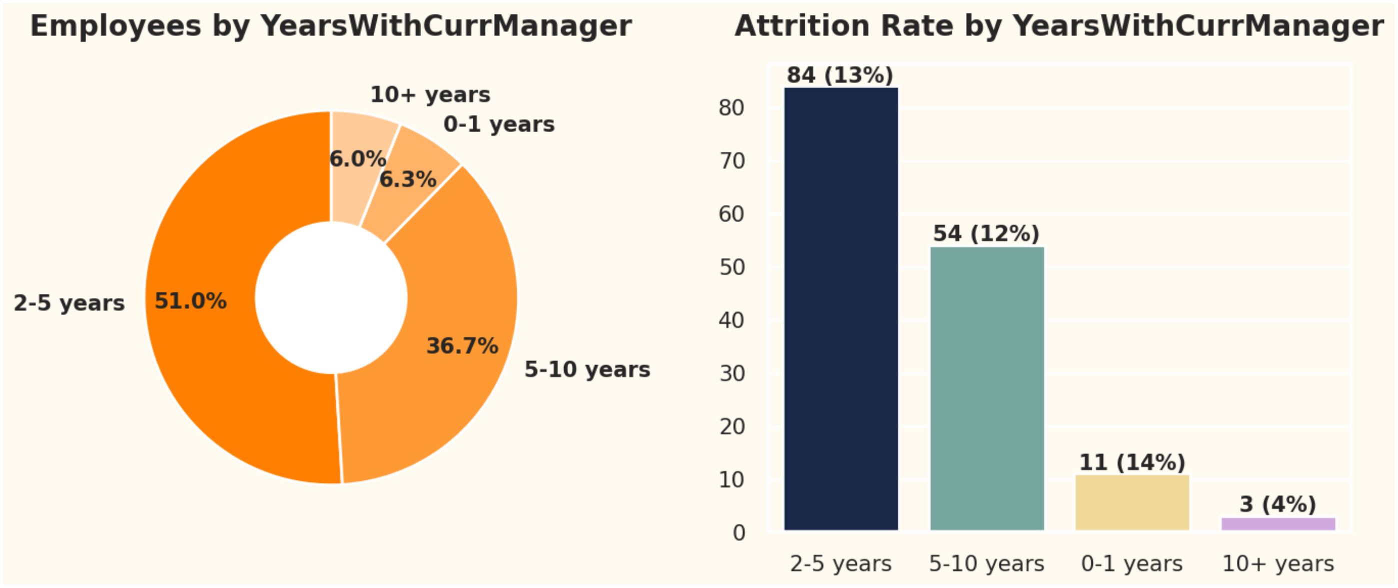


1. Almost 36% of employee has not been promoted since 2 to 5 years.
2. Almost 8% of employees has not been promoted since 10+ years.
3. All the categories in this feature is having high attrition rate specially employee who has not been promoted since 5+ years.

```
# Define the bin edges for the groups
bin_edges = [0, 1, 5, 10, 20]

# Define the labels for the groups
bin_labels = ['0-1 years', '2-5 years', '5-10 years', "10+ years"]
```

```
# Cut the DailyRate column into groups
df["YearsWithCurrManagerGroup"] = pd.cut(df['YearsWithCurrManager'], bins=bin_edges, labels=bin_labels)
```

```
#Visualization to show Total Employees by YearsWithCurrManagerGroup.

plt.figure(figsize=(14,6))
plt.subplot(1,2,1)
value_1 = df["YearsWithCurrManagerGroup"].value_counts()
plt.title("Employees by YearsWithCurrManager", fontweight="black", size=20, pad=20)
plt.pie(value_1.values, labels=value_1.index, autopct="%.1f%%",pctdistance=0.75,startangle=90,
        colors= ['#FF8000', '#FF9933', '#FFB366', '#FFCC99'],textprops={"fontweight":"black","size":15})
center_circle = plt.Circle((0, 0), 0.4, fc='white')
fig = plt.gcf()
fig.gca().add_artist(center_circle)

#Visualization to show Attrition Rate by YearsWithCurrManagerGroup.

plt.subplot(1,2,2)
new_df = df[df["Attrition"]=="Yes"]
value_2 = new_df["YearsWithCurrManagerGroup"].value_counts()
attrition_rate = np.floor((value_2/value_1)*100).values
sns.barplot(x=value_2.index.tolist(), y=value_2.values,palette=["#11264e","#6faea4","#FEE08B","#D4A1E7","#E7A1A1"])
plt.title("Attrition Rate by YearsWithCurrManager",fontweight="black",size=20,pad=20)
for index,value in enumerate(value_2):
    plt.text(index,value,str(value)+" ("+str(int(attrition_rate[index]))+"%)",ha="center",va="bottom",
             size=15,fontweight="black")
plt.tight_layout()
plt.show()
```

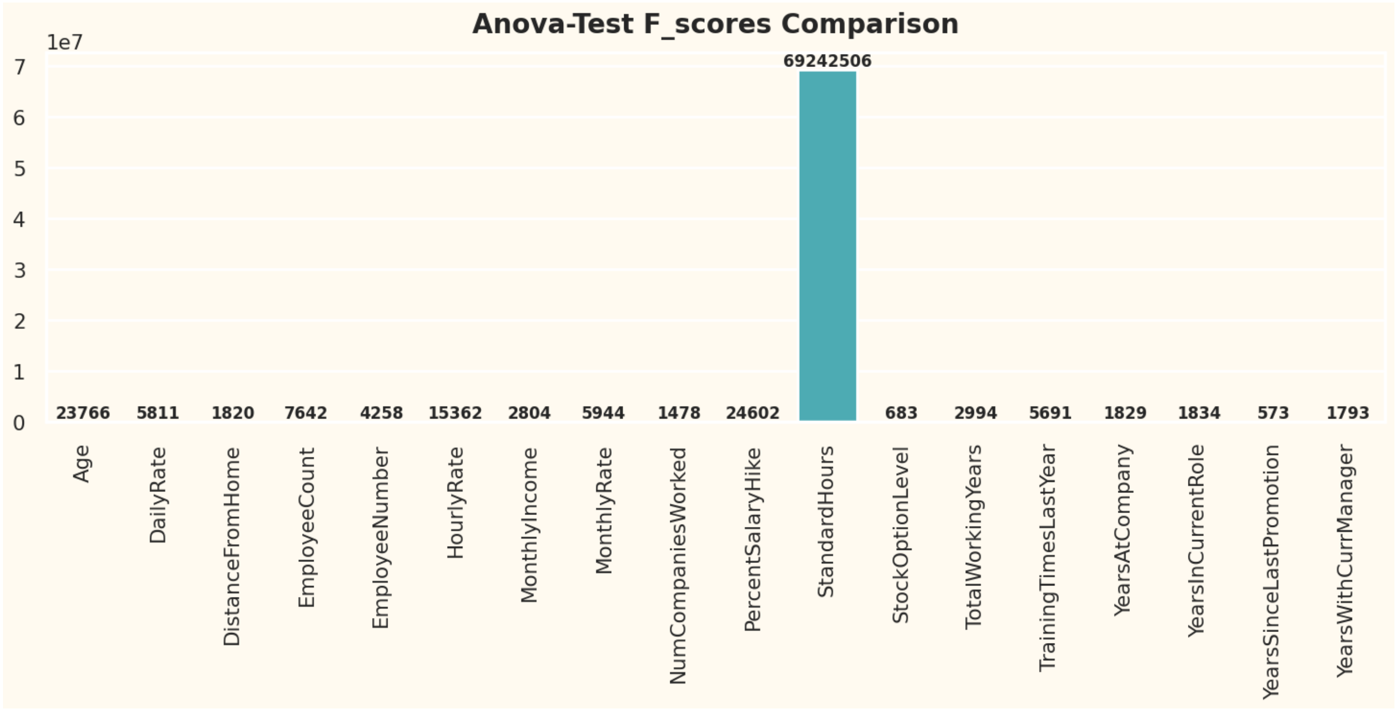


1. Almost 51% of employees has worked for 2-5 years with the same manager.
2. Almost 38% of employees has worked for 5-10 years with the same manager.
3. Employee who has worked for 10+ year with the same manager are having very low attrition rate.
4. Other Categories is having high attrition rate.

```
# Anova test is used to Analyzing the impact of different numerical features on a response categorical feature.
# Anova test returns two statistical values f_score and p_value.
# A larger F-score indicates a stronger association between the independent variable(s) and the dependent variable.
```

```
#Performing ANOVA Test to Analyze the Numerical Features Importance in Employee Attrition.




num_cols = df.select_dtypes(np.number).columns
new_df = df.copy()

new_df["Attrition"] = new_df["Attrition"].replace({"No":0,"Yes":1})
f_scores = {}
p_values = {}

for column in num_cols:
    f_score, p_value = stats.f_oneway(new_df[column],new_df["Attrition"])

    f_scores[column] = f_score
    p_values[column] = p_value
```

```
#Visualizing the F_Score of ANOVA Test of Each Numerical features.

plt.figure(figsize=(18,5))
keys = list(f_scores.keys())
values = list(f_scores.values())

sns.barplot(x=keys, y=values)
plt.title("Anova-Test F_scores Comparison",fontweight="black",size=20,pad=15)
plt.xticks(rotation=90)

for index,value in enumerate(values):
    plt.text(index,value,int(value), ha="center", va="bottom",fontweight="black",size=12)
plt.show()
```



```
#Comparing F_Score and P_value of ANOVA Test.


test_df = pd.DataFrame({"Features":keys,"F_Score":values})
```

```
test_df["P_value"] = [format(p, '.20f') for p in list(p_values.values())]
test_df
```

	Features	F_Score	P_value
0	Age	2.376693e+04	0.00000000000000000000
1	DailyRate	5.811797e+03	0.00000000000000000000
2	DistanceFromHome	1.820615e+03	0.00000000000000000000
3	EmployeeCount	7.642519e+03	0.00000000000000000000
4	EmployeeNumber	4.258788e+03	0.00000000000000000000
5	HourlyRate	1.536212e+04	0.00000000000000000000
6	MonthlyIncome	2.804460e+03	0.00000000000000000000
7	MonthlyRate	5.944089e+03	0.00000000000000000000
8	NumCompaniesWorked	1.478189e+03	0.00000000000000000000
9	PercentSalaryHike	2.460251e+04	0.00000000000000000000
10	StandardHours	6.924251e+07	0.00000000000000000000
11	StockOptionLevel	6.830696e+02	0.00000000000000000000
12	TotalWorkingYears	2.994906e+03	0.00000000000000000000
13	TrainingTimesLastYear	5.691402e+03	0.00000000000000000000
14	YearsAtCompany	1.829443e+03	0.00000000000000000000
15	YearsInCurrentRole	1.834262e+03	0.00000000000000000000
16	YearsSinceLastPromotion	5.738964e+02	0.00000000000000000000
17	YearsWithCurrManager	1.793291e+03	0.00000000000000000000

The following features shows a strong association with attrition, as indicated by their high F-scores and very low p-values.

1. Age
2. DailyRate
3. HourlyRate
4. MonthlyIncome
5. MonthlyRate
6. NumCompaniesWorked
7. PercentSalaryHike
8. TotalWorkingYears
9. TrainingTimesLastYear
10. YearsAtCompany
11. YearsWithCurrManager

- The following features doesn't shows significant relationship with attrition because of their moderate F-scores and extremely high p-values.

1. DistanceFromHome
2. StockOptionLevel
3. YearsInCurrentRole
4. YearsSinceLastPromotion

- It is important for the organization to pay attention to the identified significant features and consider them when implementing strategies to reduce attrition rates.

```
# Visualizing the Chi-Square Statistic Values of Each Categorical Features.


columns = list(chi2_statistic.keys())
values = list(chi2_statistic.values())

plt.figure(figsize=(18,6))
sns.barplot(x=columns, y=values)
plt.xticks(rotation=90)
plt.title("Chi2 Statistic Value of each Categorical Columns",fontweight="black",size=20,pad=15)
```

```
for index,value in enumerate(values):
    plt.text(index,value,round(value,2),ha="center",va="bottom",fontweight="black",size=15)

plt.show()
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-62-9758a8b8fdcc> in <cell line: 4>()
      2
      3
----> 4 columns = list(chi2_statistic.keys())
      5 values = list(chi2_statistic.values())
      6

NameError: name 'chi2_statistic' is not defined
```

SEARCH STACK OVERFLOW

```
test_df = pd.DataFrame({"Features":columns,"Chi_2 Statistic":values})
test_df["P_value"] =  [format(p, '.20f') for p in list(p_values.values())]
test_df
```

The following features showed statistically significant associations with employee attrition:

1. Department
2. EducationField
3. EnvironmentSatisfaction
4. JobLevel
5. JobRole
6. JobSatisfaction
7. MaritalStatus
8. WorkLifeBalance

- The following features did not show statistically significant associations with attrition.

1. Gender
2. PerformanceRating
3. RelationshipSatisfaction

- It is important for the organization to pay attention to the identified significant features and consider them when implementing strategies to reduce attrition rates.