In [2]:
```python
import pandas as pd
import matplotlib.pyplot as plt
import numpy as np
import seaborn as sns
```

In [3]:
```python
df = pd.read_csv("Documents/Animal Dataset.csv")
```

In [6]:
```python
df.head()
```

Out[6]:

	Animal	Height (cm)	Weight (kg)	Color	Lifespan (years)	Diet	Habitat	Predators	Average Speed (km/h)	Countries Found	Conservation Status	Family	Gestation Period (days)	Top Speed (km/h)	Social Structure	Offspring per Birth
0	Aardvark	105-130	40-65	Grey	20-30	Insectivore	Savannas, Grasslands	Lions, Hyenas	40	Africa	Least Concern	Orycteropodidae	210-240	40	Solitary	1
1	Aardwolf	40-50	8-14	Yellow-brown	10-12	Insectivore	Grasslands, Savannas	Lions, Leopards	24-30	Eastern and Southern Africa	Least Concern	Hyaenidae	90	40	Solitary	2-5
2	African Elephant	270-310	2700-6000	Grey	60-70	Herbivore	Savannah, Forest	Lions, Hyenas	25	Africa	Vulnerable	Elephantidae	640-660	40	Herd-based	1
3	African Lion	80-110	120-250	Tan	10-14	Carnivore	Grasslands, Savannas	Hyenas, Crocodiles	58	Africa	Vulnerable	Felidae	98-105	80	Group-based	2-4 (usually)
4	African Wild Dog	75-80	18-36	Multicolored	10-12	Carnivore	Savannahs	Lions, Hyenas	56	Sub-Saharan Africa	Endangered	Canidae	70	56	Group-based	10-12

In [7]:
```python
df.shape
```

Out[7]: (205, 16)

In [8]:
```python
df.dtypes
```

Out[8]:
```
Animal                     object
Height (cm)                object
Weight (kg)                object
Color                      object
Lifespan (years)           object
Diet                       object
Habitat                    object
Predators                  object
Average Speed (km/h)       object
Countries Found            object
Conservation Status        object
Family                     object
Gestation Period (days)    object
Top Speed (km/h)           object
Social Structure           object
Offspring per Birth        object
dtype: object
```

In [27]:
```python
# creating a new dataframe with the columns i need.
selected_columns = ['Gestation Period (days)', 'Conservation Status',]
df_reproductive = df[selected_columns]
```

In [28]:
```python
print(df_reproductive)
df_reproductive.shape
```

```
   Gestation Period (days) Conservation Status
0                  210-240       Least Concern
1                       90       Least Concern
2                  640-660          Vulnerable
3                   98-105          Vulnerable
4                       70          Endangered
..                     ...                 ...
200                215-280       Least Concern
201                  80-90          Endangered
202         Not Applicable        Not Evaluated
203                180-365       Least Concern
204                  10-25          Endangered

[205 rows x 2 columns]
```

Out[28]: (205, 2)

In [29]:
```python
# Removing rows with any "Not Applicable"
df_reproductive = df_reproductive[~df_reproductive.isin(['Not Applicable']).any(axis=1)]
df_reproductive = df_reproductive.reset_index(drop=True)
df_reproductive.shape
```

Out[29]: (184, 2)

In [30]:
```python
pattern = r'^\d+-\d+$'
# Keep only rows where Gestation Period matches the x-y pattern
df_reproductive = df_reproductive[df_reproductive['Gestation Period (days)'].str.match(pattern)]

# Reset index after filtering
df_reproductive = df_reproductive.reset_index(drop=True)
df_reproductive.shape
```

Out[30]: (148, 2)

In [31]:
```python
# Display the first few rows and basic information about the dataframe
print("First few rows of the dataframe:")
print(df_reproductive.head())

print("\nDataframe information:")
print(df_reproductive.info())

print("\nSummary statistics:")
print(df_reproductive.describe())
```

```
First few rows of the dataframe:
  Gestation Period (days) Conservation Status
0                 210-240       Least Concern
1                 640-660          Vulnerable
2                  98-105          Vulnerable
3                   10-15       Not Evaluated
4                 270-290     Near Threatened

Dataframe information:
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 148 entries, 0 to 147
Data columns (total 2 columns):
 #   Column                   Non-Null Count  Dtype 
---  ------                   --------------  ----- 
 0   Gestation Period (days)  148 non-null    object
 1   Conservation Status      148 non-null    object
dtypes: object(2)
memory usage: 2.4+ KB
None

Summary statistics:
       Gestation Period (days) Conservation Status
count                      148                 148
unique                     130                   9
top                      60-64       Least Concern
freq                         3                  54
```

In [34]:
```python
# Function to calculate average from range
def calculate_average(value):
    if isinstance(value, str) and '-' in value:
        x, y = map(float, value.split('-'))
        return (x + y) / 2
    return value

# Apply the average calculation to Gestation Period column
df_reproductive['Gestation Period (days)'] = df_reproductive['Gestation Period (days)'].apply(calculate_average)
print("First few rows of the dataframe:")
print(df_reproductive.head())
```

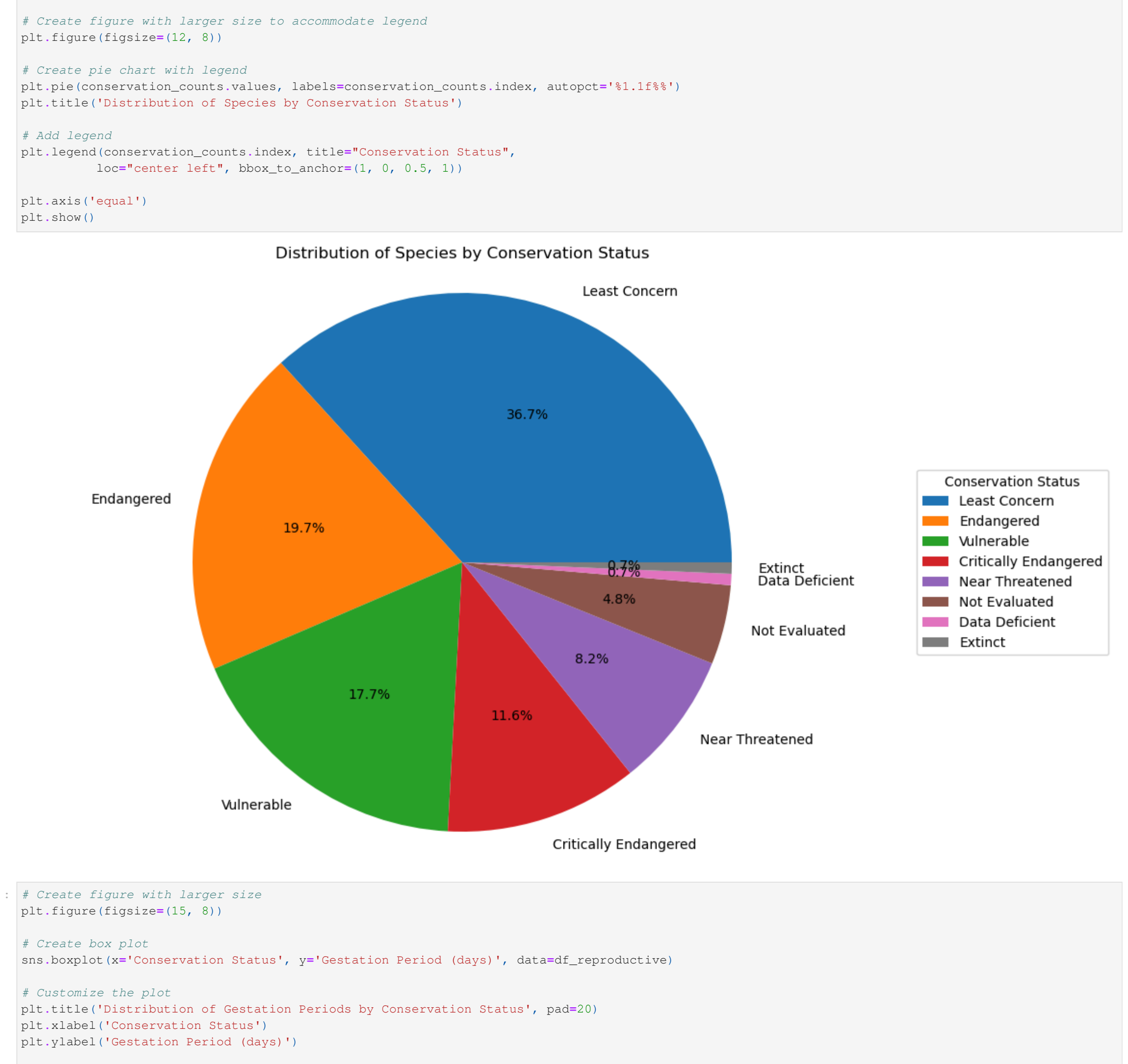
```
First few rows of the dataframe:
   Gestation Period (days) Conservation Status
0                    225.0       Least Concern
1                    650.0          Vulnerable
2                    101.5          Vulnerable
3                     12.5       Not Evaluated
4                    280.0     Near Threatened
```

In [39]:
```python
# Remove rows with "Extinct (around 58 million years ago)"
df_reproductive = df_reproductive[df_reproductive['Conservation Status'] != 'Extinct (around 58 million years ago)']

# Reset the index
df_reproductive = df_reproductive.reset_index(drop=True)
```

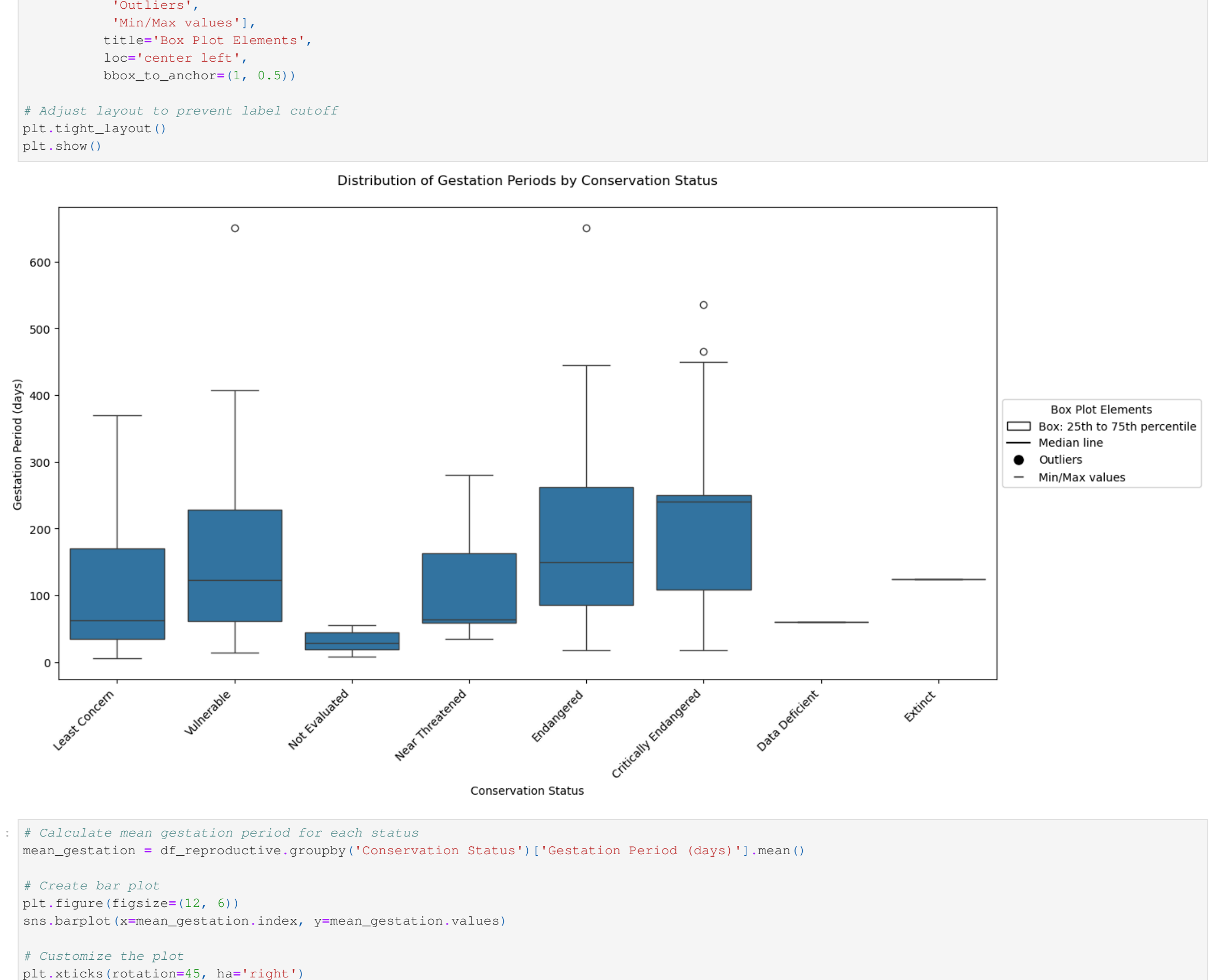
In [40]:
```python
# Display summary statistics grouped by Conservation Status
summary_stats = df_reproductive.groupby('Conservation Status')['Gestation Period (days)'].describe()
print("\nSummary Statistics by Conservation Status:")
print(summary_stats)

# Additional insights
print("\nOverall Statistics for Gestation Period:")
print(df_reproductive['Gestation Period (days)'].describe())

# Count number of species in each conservation status
print("\nNumber of Species per Conservation Status:")
print(df_reproductive['Conservation Status'].value_counts())
```

```
Summary Statistics by Conservation Status:
                      count        mean         std    min      25%    50%  \
Conservation Status                                                          
Critically Endangered  17.0  215.647059  151.879657   17.5  108.500  240.0   
Data Deficient          1.0   60.000000         NaN   60.0   60.000   60.0   
Endangered             29.0  192.896552  150.040136   17.5   85.000  150.0   
Extinct                 1.0  124.500000         NaN  124.5  124.500  124.5   
Least Concern          54.0  110.944444   97.019579    6.0   35.000   62.5   
Near Threatened        12.0  112.291667   88.799431   35.0   59.250   63.0   
Not Evaluated           7.0   30.857143   18.121613    8.0   18.750   28.0   
Vulnerable             26.0  162.403846  147.632044   13.5   60.625  122.5   

                          75%    max  
Conservation Status                   
Critically Endangered  250.00  535.0  
Data Deficient          60.00   60.0  
Endangered            262.50  650.0  
Extinct               124.50  124.5  
Least Concern         170.00  370.0  
Near Threatened       162.50  280.0  
Not Evaluated          43.75   55.0  
Vulnerable            228.75  650.0  

Overall Statistics for Gestation Period:
count    147.000000
mean     144.363946
std      129.946954
min        6.000000
25%       48.500000
50%      108.500000
75%      225.000000
max      650.000000
Name: Gestation Period (days), dtype: float64

Number of Species per Conservation Status:
Conservation Status
Least Concern            54
Endangered               29
Vulnerable               26
Critically Endangered    17
Near Threatened          12
Not Evaluated             7
Data Deficient            1
Extinct                   1
Name: count, dtype: int64
```

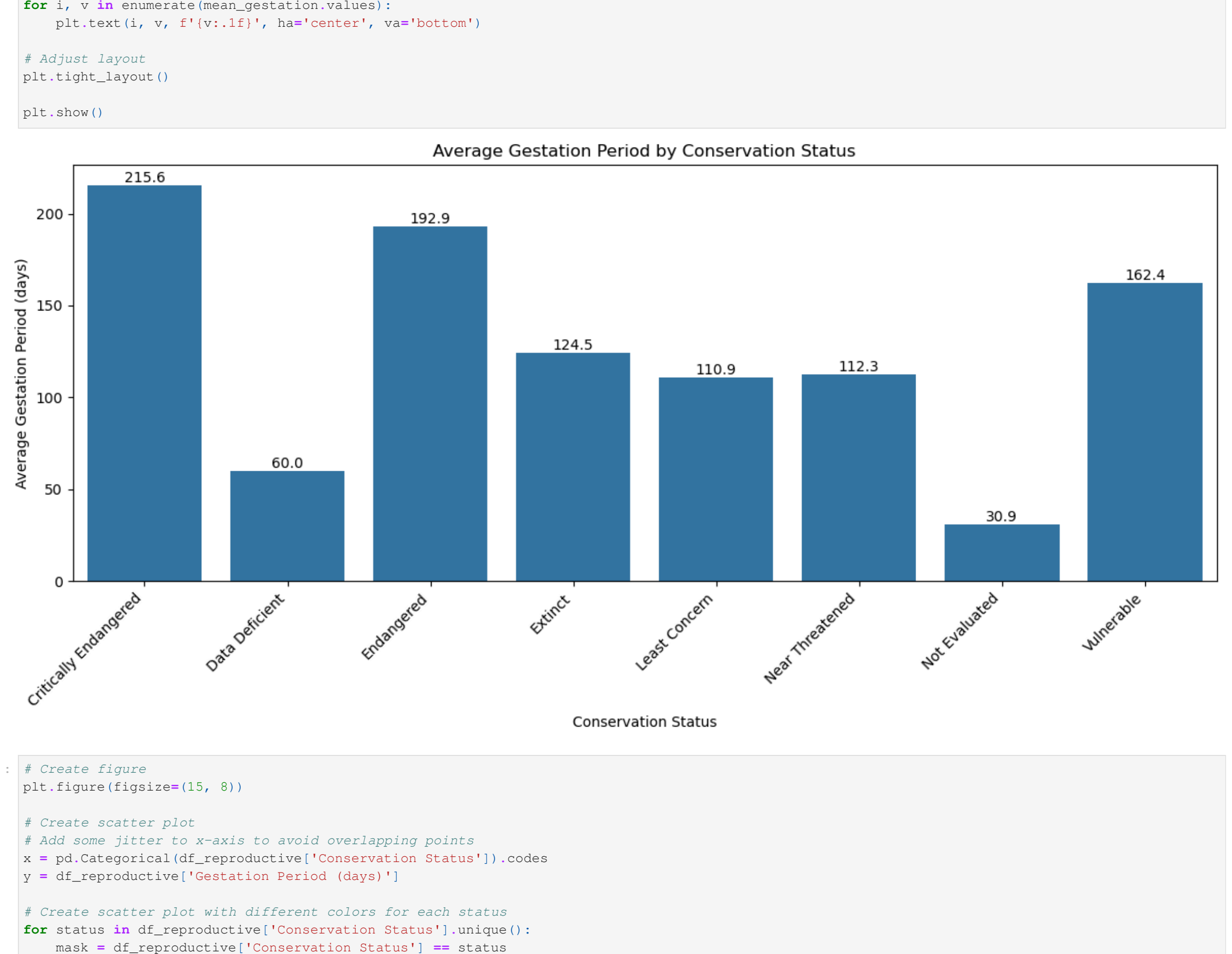
In [43]:
```python
# Calculate counts
conservation_counts = df_reproductive['Conservation Status'].value_counts()

# Create figure with larger size to accommodate legend
plt.figure(figsize=(12, 8))

# Create pie chart with legend
plt.pie(conservation_counts.values, labels=conservation_counts.index, autopct='%1.1f%%')
plt.title('Distribution of Species by Conservation Status')

# Add legend
plt.legend(conservation_counts.index, title="Conservation Status",
          loc="center left", bbox_to_anchor=(1, 0, 0.5, 1))

plt.axis('equal')
plt.show()
```



In [46]:
```python
# Create figure with larger size
plt.figure(figsize=(15, 8))

# Create box plot
sns.boxplot(x='Conservation Status', y='Gestation Period (days)', data=df_reproductive)

# Customize the plot
plt.title('Distribution of Gestation Periods by Conservation Status', pad=20)
plt.xlabel('Conservation Status')
plt.ylabel('Gestation Period (days)')

# Rotate x-axis labels for better readability
plt.xticks(rotation=45, ha='right')

# Add a legend explaining box plot elements
box_plot_elements = [
    plt.Rectangle((0,0),1,1, facecolor='white', edgecolor='black'),
    plt.Line2D([0], [0], color='black', linewidth=1.5),
    plt.Line2D([0], [0], marker='o', color='black', markersize=8, linestyle='None'),
    plt.Line2D([0], [0], marker='_', color='black', markersize=8, linestyle='None')
]

plt.legend(box_plot_elements,
          ['Box: 25th to 75th percentile',
           'Median line',
           'Outliers',
           'Min/Max values'],
          title='Box Plot Elements',
          loc='center left',
          bbox_to_anchor=(1, 0.5))

# Adjust layout to prevent label cutoff
plt.tight_layout()
plt.show()
```



In [45]:
```python
# Calculate mean gestation period for each status
mean_gestation = df_reproductive.groupby('Conservation Status')['Gestation Period (days)'].mean()

# Create bar plot
plt.figure(figsize=(12, 6))
sns.barplot(x=mean_gestation.index, y=mean_gestation.values)

# Customize the plot
plt.xticks(rotation=45, ha='right')
plt.title('Average Gestation Period by Conservation Status')
plt.xlabel('Conservation Status')
plt.ylabel('Average Gestation Period (days)')

# Add value labels on top of each bar
for i, v in enumerate(mean_gestation.values):
    plt.text(i, v, f'{v:.1f}', ha='center', va='bottom')

# Adjust layout
plt.tight_layout()

plt.show()
```



In [47]:
```python
# Create figure
plt.figure(figsize=(15, 8))

# Create scatter plot
# Add some jitter to x-axis to avoid overlapping points
x = pd.Categorical(df_reproductive['Conservation Status']).codes
y = df_reproductive['Gestation Period (days)']

# Create scatter plot with different colors for each status
for status in df_reproductive['Conservation Status'].unique():
    mask = df_reproductive['Conservation Status'] == status
    plt.scatter(x[mask] + np.random.normal(0, 0.04, size=sum(mask)),
                y[mask],
                alpha=0.6,
                label=status)

# Customize the plot
plt.title('Individual Gestation Periods by Conservation Status', pad=20)
plt.xlabel('Conservation Status')
plt.ylabel('Gestation Period (days)')

# Set x-axis labels
plt.xticks(range(len(df_reproductive['Conservation Status'].unique())),
           df_reproductive['Conservation Status'].unique(),
           rotation=45,
           ha='right')

# Add legend
plt.legend(title='Conservation Status',
          loc='center left',
          bbox_to_anchor=(1, 0.5))

# Add grid for better readability
plt.grid(True, alpha=0.3)

# Adjust layout
plt.tight_layout()
plt.show()
```