In [12]:
```
import pandas as pd

import seaborn as sns
import matplotlib.pyplot as plt
```

In [11]:
```
p=pd.read_csv("G:/My Drive/INNO_INTERN/DATASETS/project2.csv")
```

In [3]:
```
p.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 112634 entries, 0 to 112633
Data columns (total 17 columns):
 #   Column                                             Non-Null Count   D
type
---  ------                                             --------------   -
----
 0   VIN (1-10)                                         112634 non-null  o
bject
 1   County                                             112634 non-null  o
bject
 2   City                                               112634 non-null  o
bject
 3   State                                              112634 non-null  o
bject
 4   Postal Code                                        112634 non-null  i
nt64
 5   Model Year                                         112634 non-null  i
nt64
 6   Make                                               112634 non-null  o
bject
 7   Model                                              112614 non-null  o
bject
 8   Electric Vehicle Type                              112634 non-null  o
bject
 9   Clean Alternative Fuel Vehicle (CAFV) Eligibility  112634 non-null  o
bject
 10  Electric Range                                     112634 non-null  i
nt64
 11  Base MSRP                                          112634 non-null  i
nt64
 12  Legislative District                               112348 non-null  f
loat64
 13  DOL Vehicle ID                                     112634 non-null  i
nt64
 14  Vehicle Location                                   112610 non-null  o
bject
 15  Electric Utility                                   112191 non-null  o
bject
 16  2020 Census Tract                                  112634 non-null  i
nt64
dtypes: float64(1), int64(6), object(10)
memory usage: 14.6+ MB
```

In [4]:
```
p.head()
```

Out[4]:

	VIN (1-10)	County	City	State	Postal Code	Model Year	Make	Model	Electric Vehicle Type
0	JTMEB3FV6N	Monroe	Key West	FL	33040	2022	TOYOTA	RAV4 PRIME	Plug-in Hybrid Electric Vehicle (PHEV)
1	1G1RD6E45D	Clark	Laughlin	NV	89029	2013	CHEVROLET	VOLT	Plug-in Hybrid Electric Vehicle (PHEV)
2	JN1AZ0CP8B	Yakima	Yakima	WA	98901	2011	NISSAN	LEAF	Battery Electric Vehicle (BEV)
3	1G1FW6S08H	Skagit	Concrete	WA	98237	2017	CHEVROLET	BOLT EV	Battery Electric Vehicle (BEV)
4	3FA6P0SU1K	Snohomish	Everett	WA	98201	2019	FORD	FUSION	Plug-in Hybrid Electric Vehicle (PHEV)

In [5]:
```
p.tail()
```

Out[5]:

	VIN (1-10)	County	City	State	Postal Code	Model Year	Make	Model	Electric Vehicle Type
112629	7SAYGDEF2N	King	Duvall	WA	98019	2022	TESLA	MODEL Y	Battery Electric Vehicle (BEV)
112630	1N4BZ1CP7K	San Juan	Friday Harbor	WA	98250	2019	NISSAN	LEAF	Battery Electric Vehicle (BEV)
112631	1FMCU0KZ4N	King	Vashon	WA	98070	2022	FORD	ESCAPE	Plug-in Hybrid Electric Vehicle (PHEV)
112632	KNDCD3LD4J	King	Covington	WA	98042	2018	KIA	NIRO	Plug-in Hybrid Electric Vehicle (PHEV)
112633	YV4BR0CL8N	King	Covington	WA	98042	2022	VOLVO	XC90	Plug-in Hybrid Electric Vehicle (PHEV)

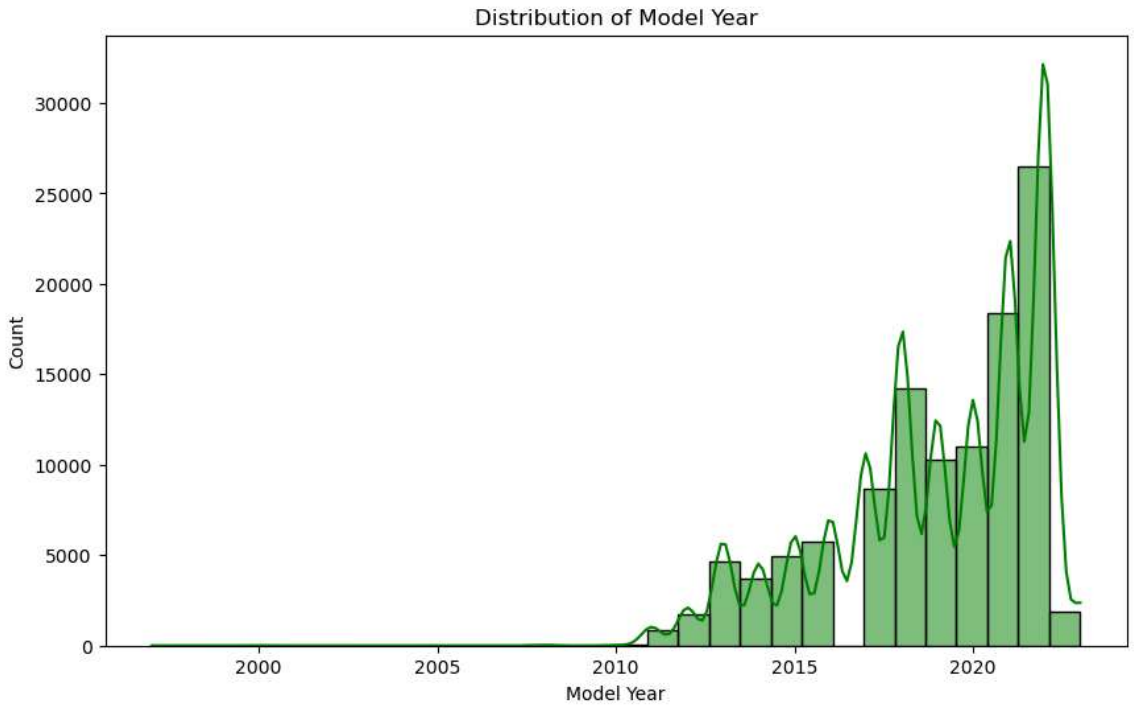
In [7]:
```
p.describe()
```

Out[7]:

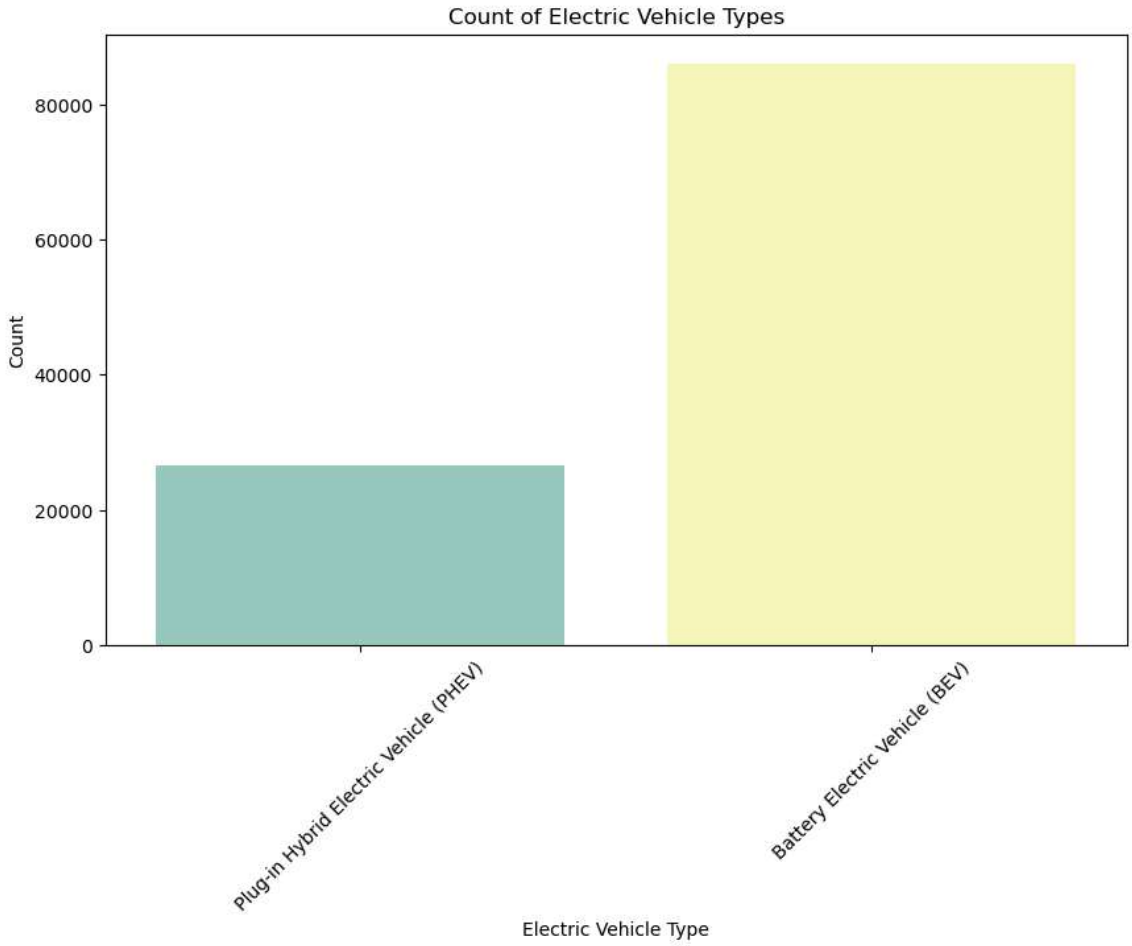
	Postal Code	Model Year	Electric Range	Base MSRP	Legislative District	DOL Vel
count	112634.000000	112634.000000	112634.000000	112634.000000	112348.000000	1.126340e
mean	98156.226850	2019.003365	87.812987	1793.439681	29.805604	1.994567e
std	2648.733064	2.892364	102.334216	10783.753486	14.700545	9.398427e
min	1730.000000	1997.000000	0.000000	0.000000	1.000000	4.777000e
25%	98052.000000	2017.000000	0.000000	0.000000	18.000000	1.484142e
50%	98119.000000	2020.000000	32.000000	0.000000	34.000000	1.923896e
75%	98370.000000	2022.000000	208.000000	0.000000	43.000000	2.191899e
max	99701.000000	2023.000000	337.000000	845000.000000	49.000000	4.792548e

In [14]:
```
# ------------- Univariate Analysis -------------
# 1. Distribution of Model Year
plt.figure(figsize=(10, 6))
sns.histplot(p['Model Year'], kde=True, bins=30, color='green')
plt.title('Distribution of Model Year')
plt.xlabel('Model Year')
plt.ylabel('Count')
plt.show()
```

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

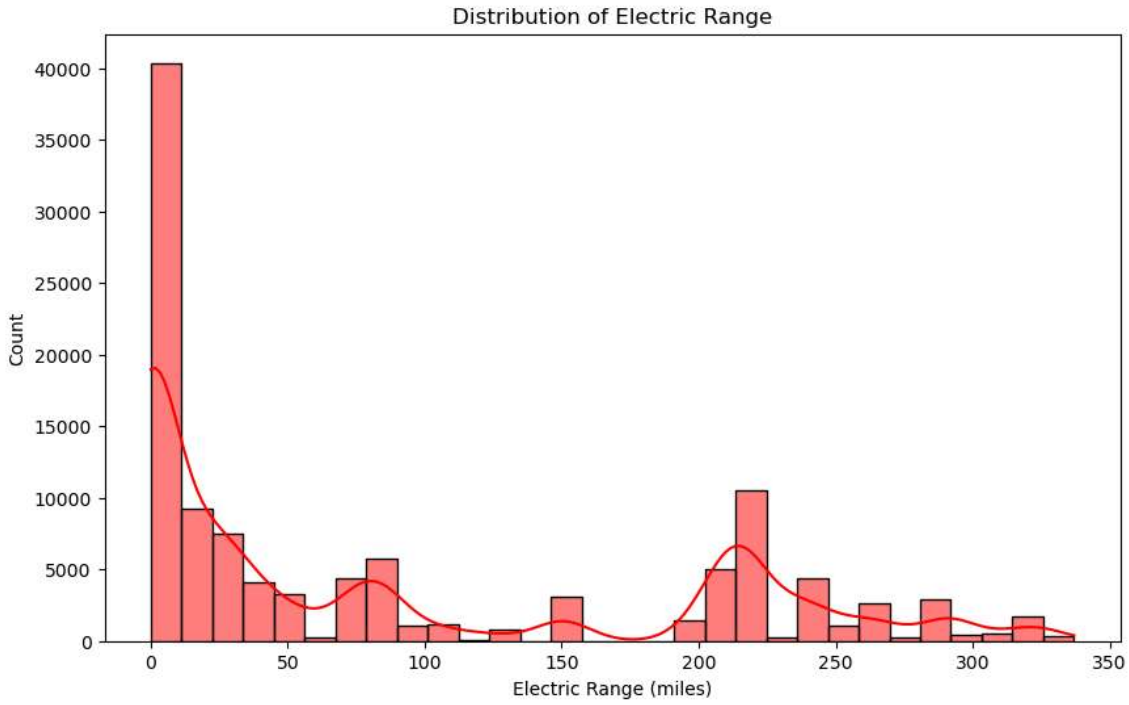


In [17]:
```
# 2. Distribution of Electric Vehicle Type
plt.figure(figsize=(10, 6))
sns.countplot(data=p, x='Electric Vehicle Type', palette='Set3')
plt.title('Count of Electric Vehicle Types')
plt.xlabel('Electric Vehicle Type')
plt.ylabel('Count')
plt.xticks(rotation=45)
plt.show()
```

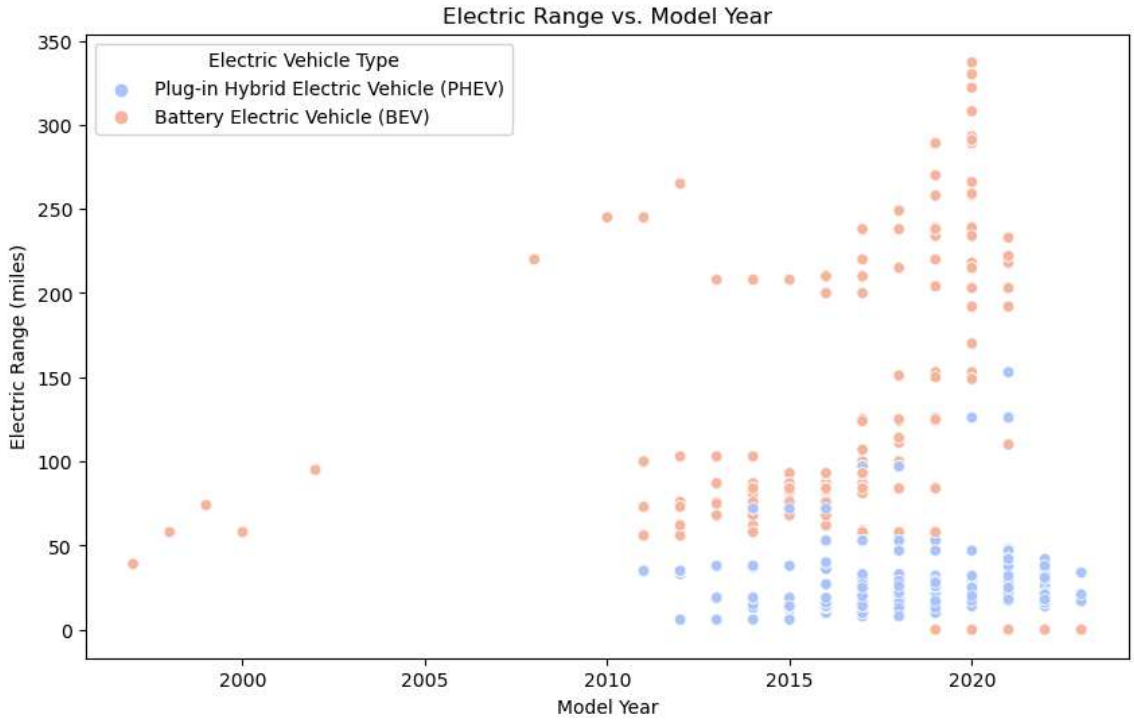


In [18]:
```
# 3. Distribution of Electric Range
plt.figure(figsize=(10, 6))
sns.histplot(p['Electric Range'], kde=True, bins=30, color='red')
plt.title('Distribution of Electric Range')
plt.xlabel('Electric Range (miles)')
plt.ylabel('Count')
plt.show()
```

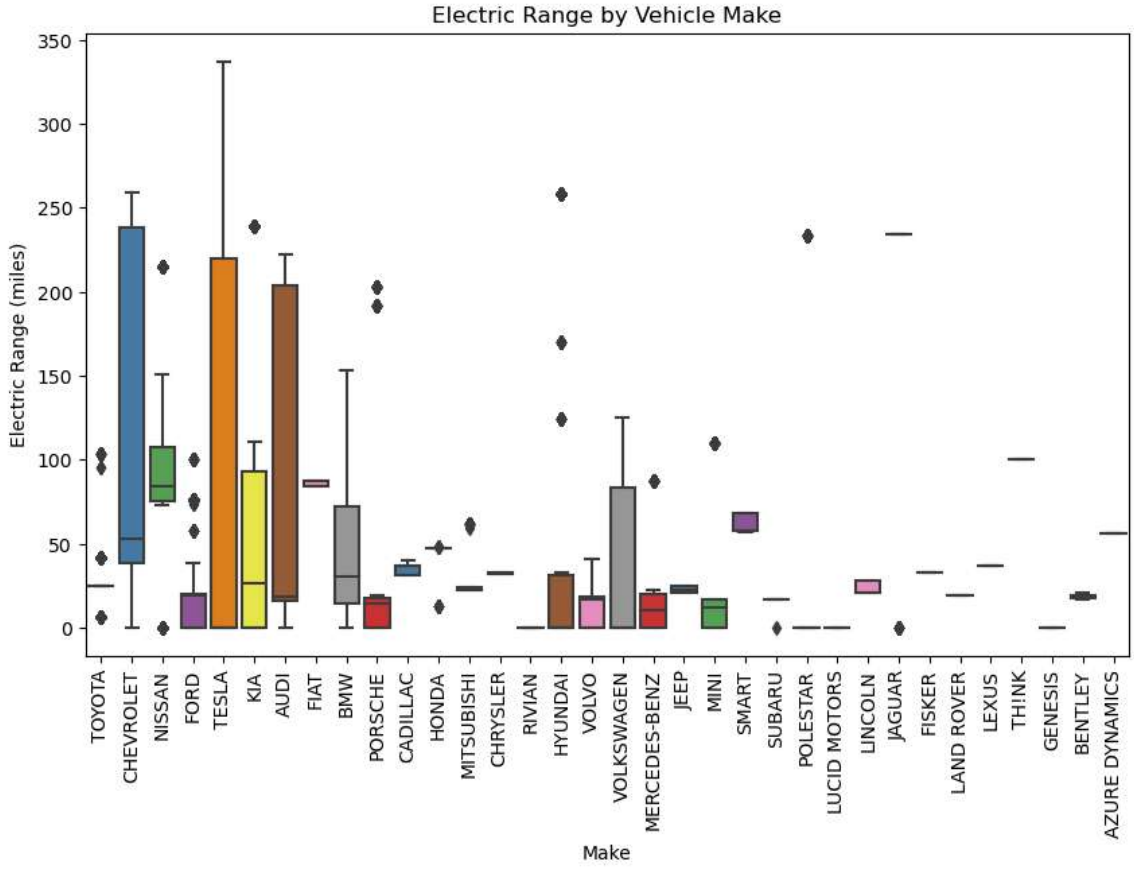
```
C:\Users\sadgu\anaconda3\Lib\site-packages\seaborn\_oldcore.py:1119: FutureWarning: use_inf_as_na option is deprecated and will be removed in a future version. Convert inf values to NaN before operating instead.
  with pd.option_context('mode.use_inf_as_na', True):
```



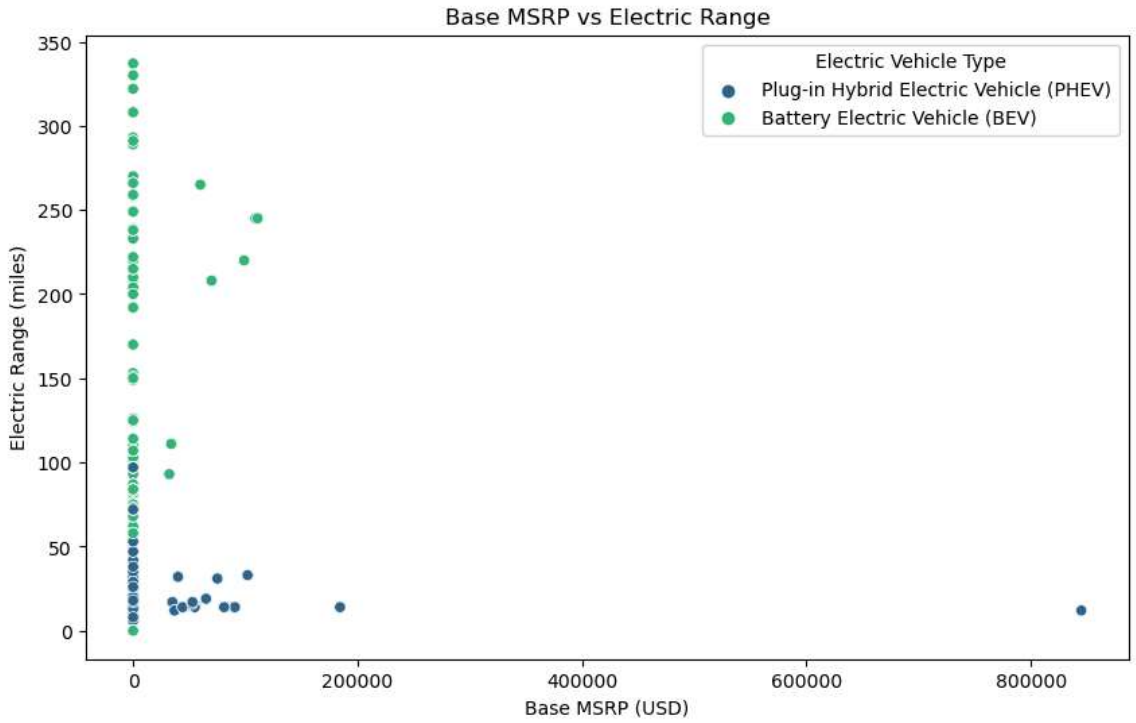
In [19]:
```
# ------------- Bivariate Analysis -------------
# 4. Electric Range vs. Model Year
plt.figure(figsize=(10, 6))
sns.scatterplot(data=p, x='Model Year', y='Electric Range', hue='Electric V
plt.title('Electric Range vs. Model Year')
plt.xlabel('Model Year')
plt.ylabel('Electric Range (miles)')
plt.legend(title='Electric Vehicle Type')
plt.show()
```



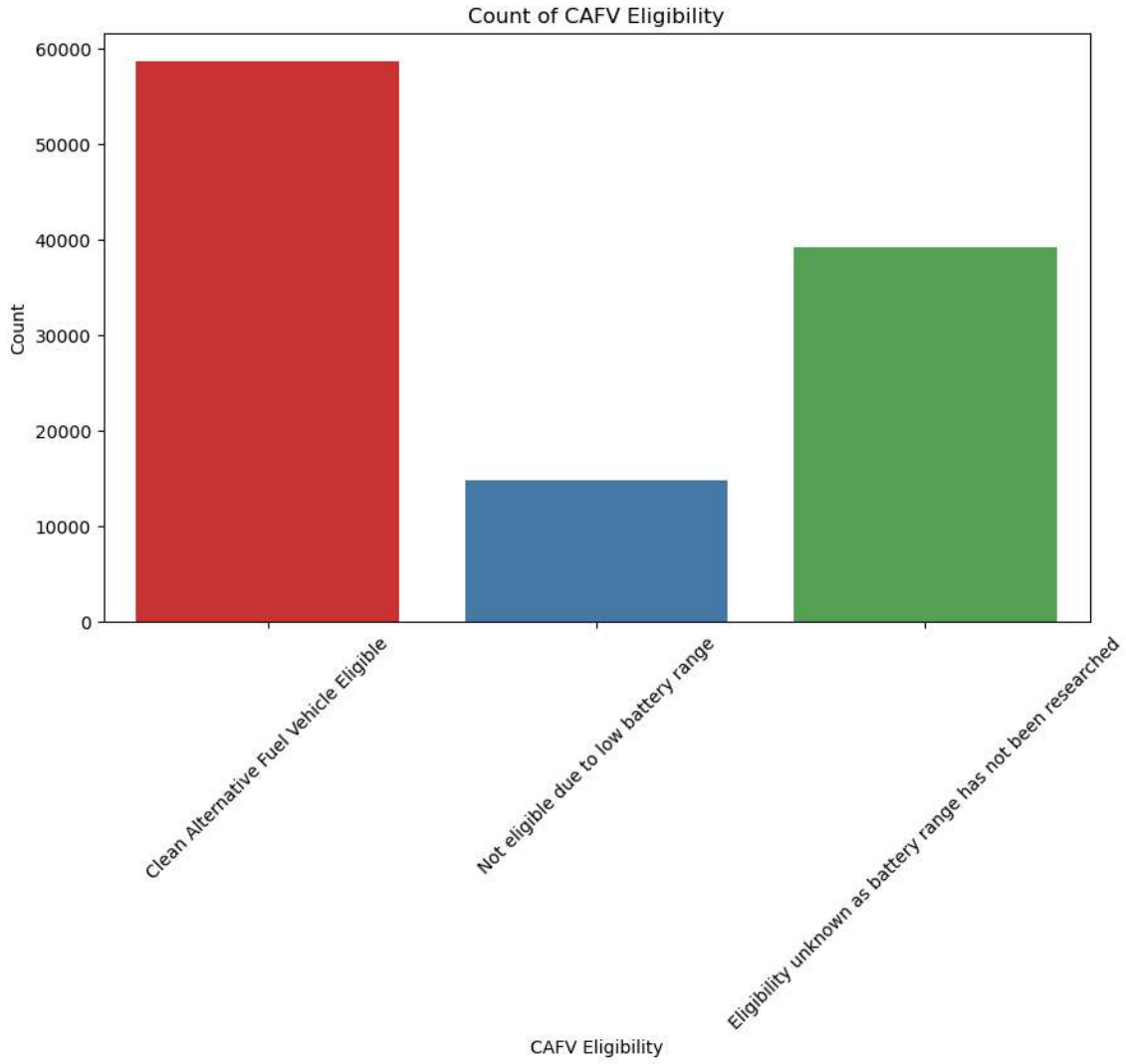
In [22]:
```
# 5. Electric Range by Make
plt.figure(figsize=(10, 6))
sns.boxplot(data=p, x='Make', y='Electric Range', palette='Set1')
plt.title('Electric Range by Vehicle Make')
plt.xlabel('Make')
plt.ylabel('Electric Range (miles)')
plt.xticks(rotation=90)
plt.show()
```



In [23]:
```
# 6. Base MSRP vs Electric Range
plt.figure(figsize=(10, 6))
sns.scatterplot(data=p, x='Base MSRP', y='Electric Range', hue='Electric Ve
plt.title('Base MSRP vs Electric Range')
plt.xlabel('Base MSRP (USD)')
plt.ylabel('Electric Range (miles)')
plt.legend(title='Electric Vehicle Type')
plt.show()
```



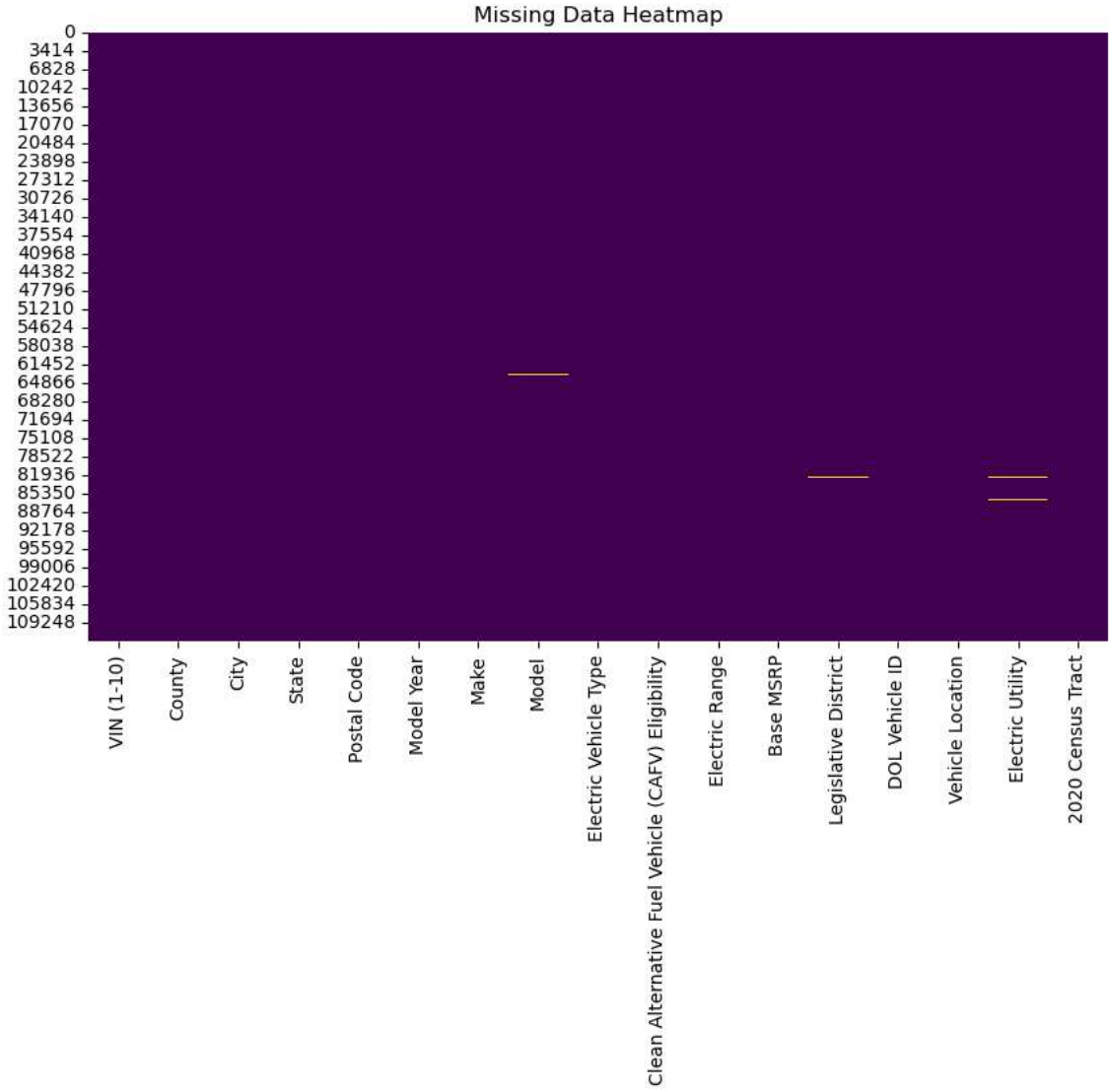
In [24]:
```
# 7. Count of Clean Alternative Fuel Vehicle Eligibility
plt.figure(figsize=(10, 6))
sns.countplot(data=p, x='Clean Alternative Fuel Vehicle (CAFV) Eligibility'
plt.title('Count of CAFV Eligibility')
plt.xlabel('CAFV Eligibility')
plt.ylabel('Count')
plt.xticks(rotation=45)
plt.show()
```



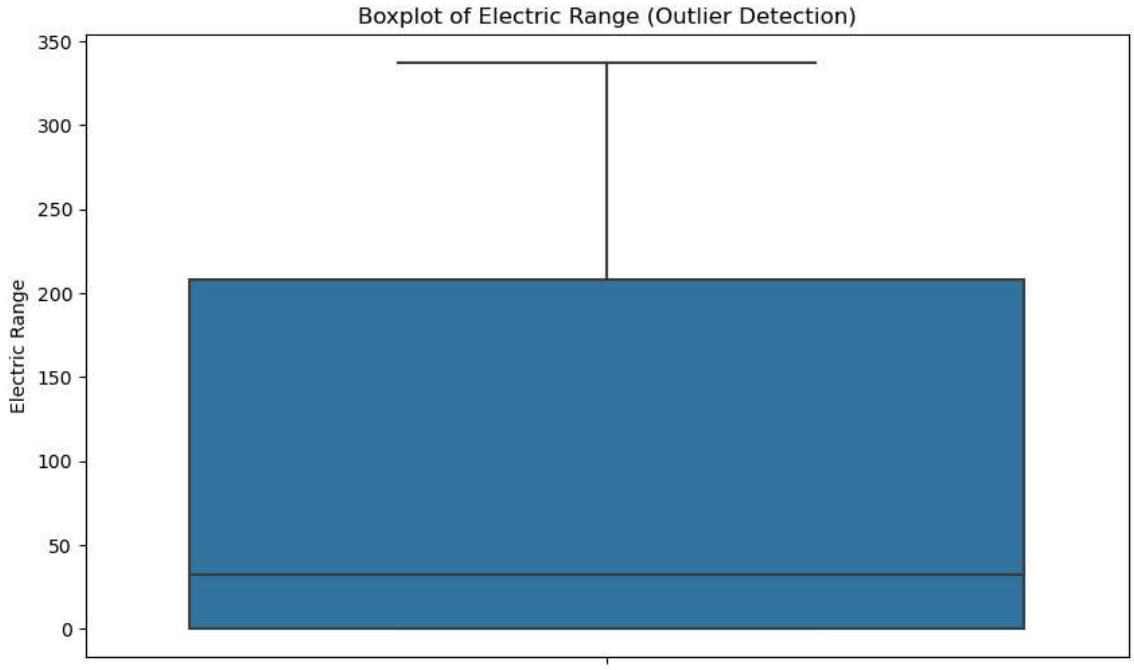
In [26]:
```
# Missing Data Check
print(p.isnull().sum())
```

```
VIN (1-10)                                             0
County                                                 0
City                                                   0
State                                                  0
Postal Code                                            0
Model Year                                             0
Make                                                   0
Model                                                 20
Electric Vehicle Type                                  0
Clean Alternative Fuel Vehicle (CAFV) Eligibility      0
Electric Range                                         0
Base MSRP                                              0
Legislative District                                 286
DOL Vehicle ID                                         0
Vehicle Location                                      24
Electric Utility                                     443
2020 Census Tract                                      0
dtype: int64
```

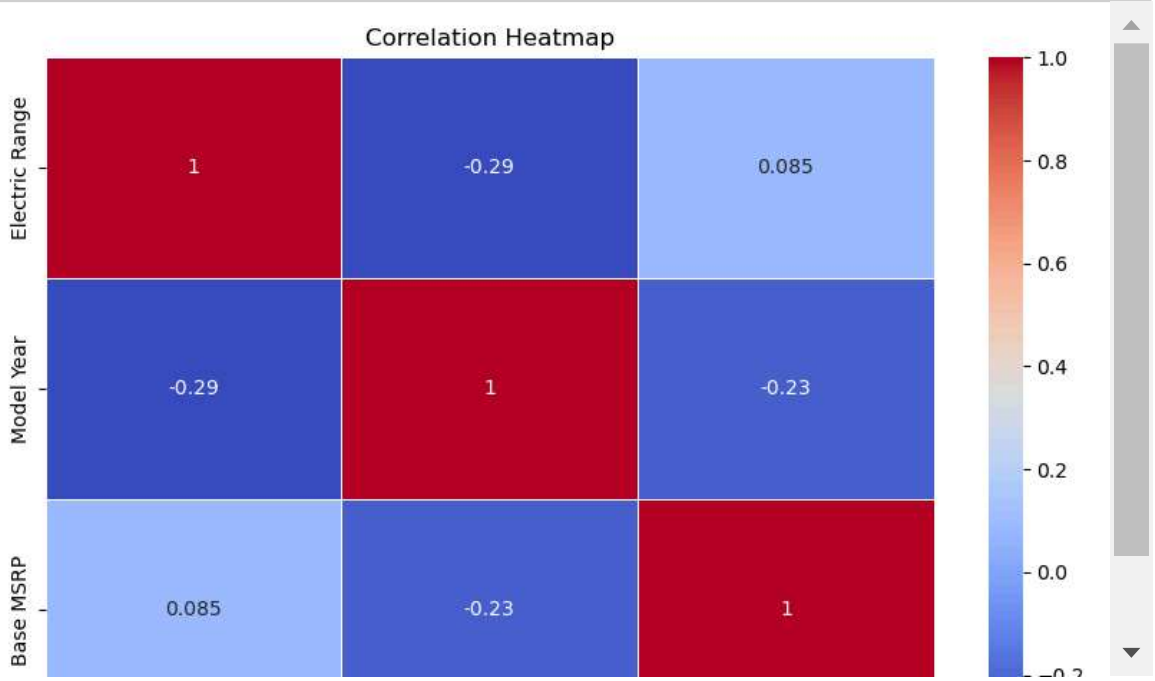
In [27]:
```
import seaborn as sns
import matplotlib.pyplot as plt

# Heatmap to visualize missing data
plt.figure(figsize=(10, 6))
sns.heatmap(p.isnull(), cbar=False, cmap='viridis')
plt.title('Missing Data Heatmap')
plt.show()
```



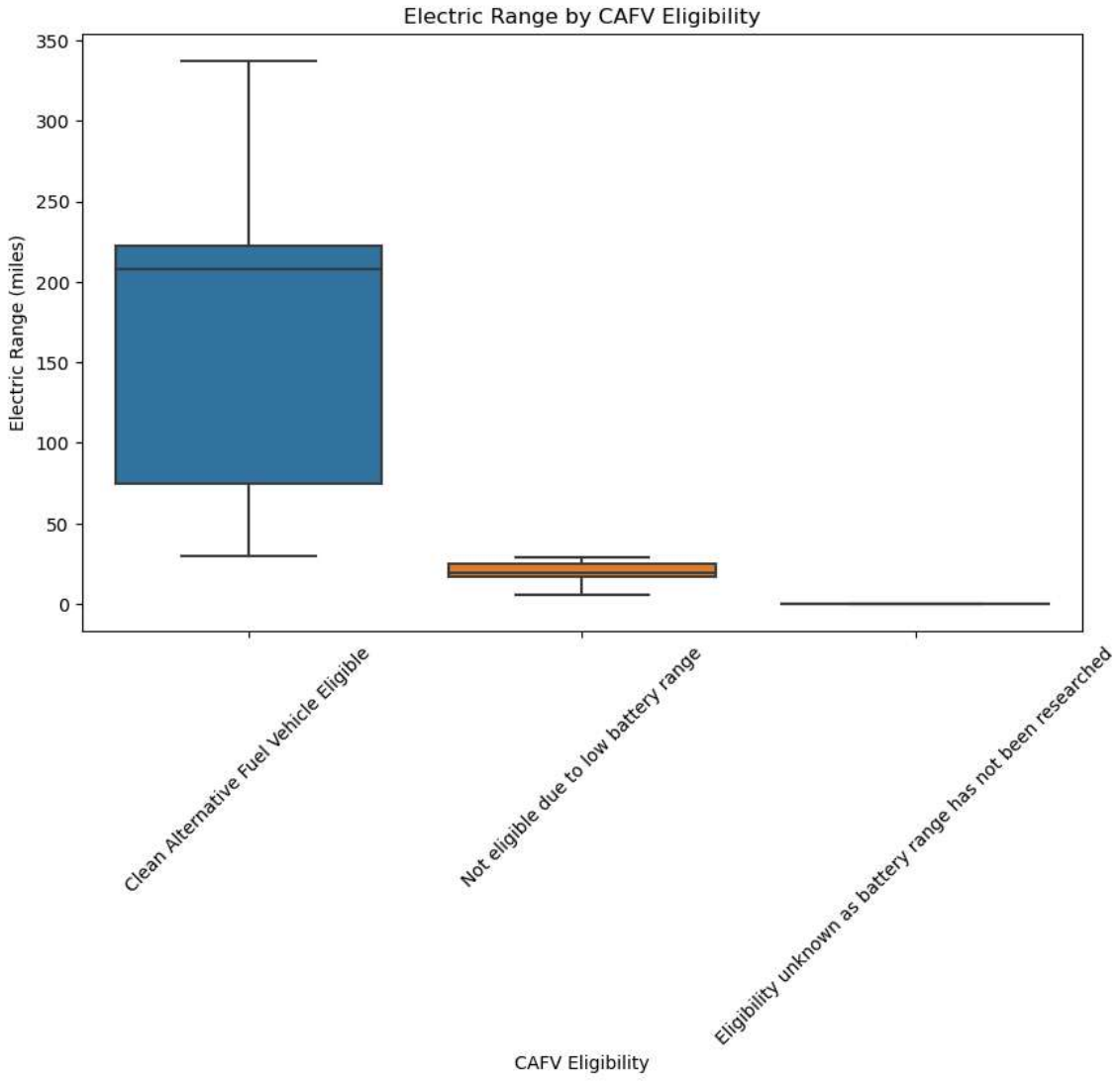
In [29]:
```
# Boxplot for outlier detection in Electric Range
plt.figure(figsize=(10, 6))
sns.boxplot(data=p, y='Electric Range')
plt.title('Boxplot of Electric Range (Outlier Detection)')
plt.show()

# Boxplot for outlier detection in Base MSRP
plt.figure(figsize=(10, 6))
sns.boxplot(data=p, y='Base MSRP')
plt.title('Boxplot of Base MSRP (Outlier Detection)')
plt.show()
```



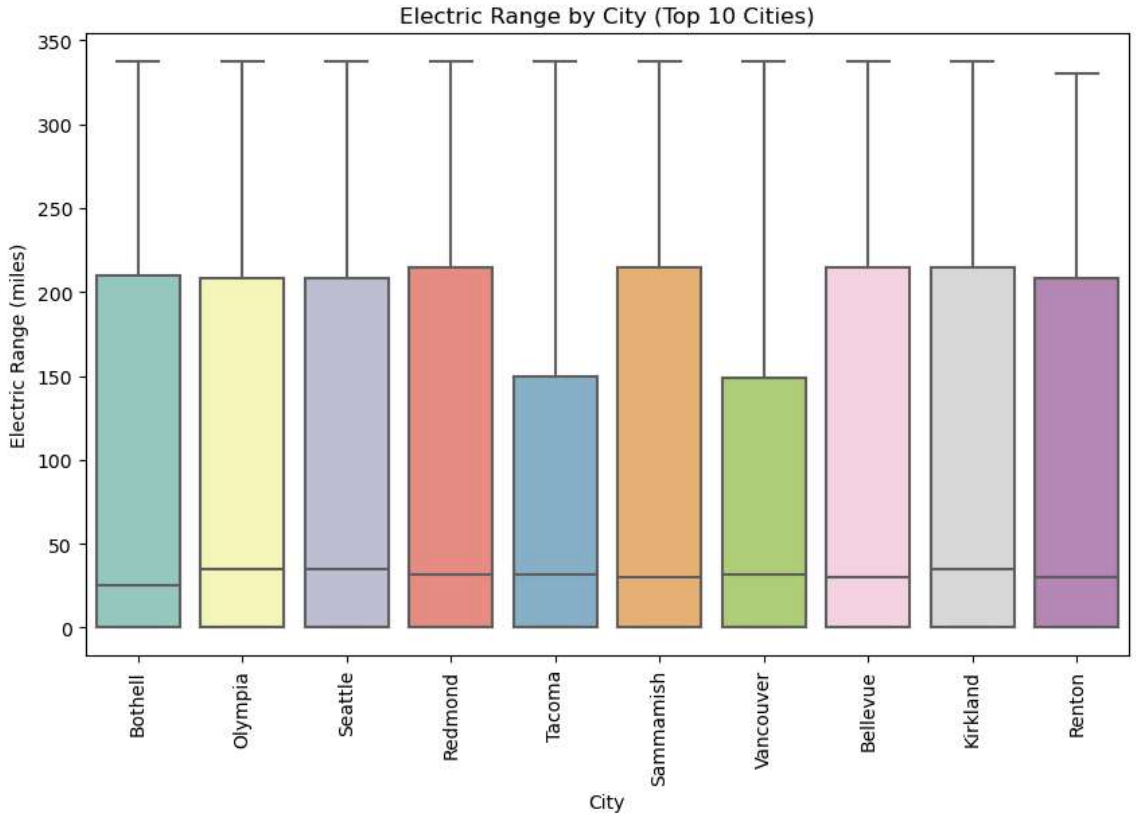
In [31]:
```
# Correlation matrix and heatmap
plt.figure(figsize=(10, 6))
corr_matrix = p[['Electric Range', 'Model Year', 'Base MSRP']].corr()
sns.heatmap(corr_matrix, annot=True, cmap='coolwarm', linewidths=0.5)
plt.title('Correlation Heatmap')
plt.show()
```



In [32]:
```
# Boxplot for Electric Range by CAFV Eligibility
plt.figure(figsize=(10, 6))
sns.boxplot(data=p, x='Clean Alternative Fuel Vehicle (CAFV) Eligibility',
plt.title('Electric Range by CAFV Eligibility')
plt.xlabel('CAFV Eligibility')
plt.ylabel('Electric Range (miles)')
plt.xticks(rotation=45)
plt.show()
```



In [34]:
```
# Electric Range by City (Top 10 Cities)
top_cities = p['City'].value_counts().nlargest(10).index
filtered_data = p[p['City'].isin(top_cities)]

plt.figure(figsize=(10, 6))
sns.boxplot(data=filtered_data, x='City', y='Electric Range', palette='Set3
plt.title('Electric Range by City (Top 10 Cities)')
plt.xlabel('City')
plt.ylabel('Electric Range (miles)')
plt.xticks(rotation=90)
plt.show()
```



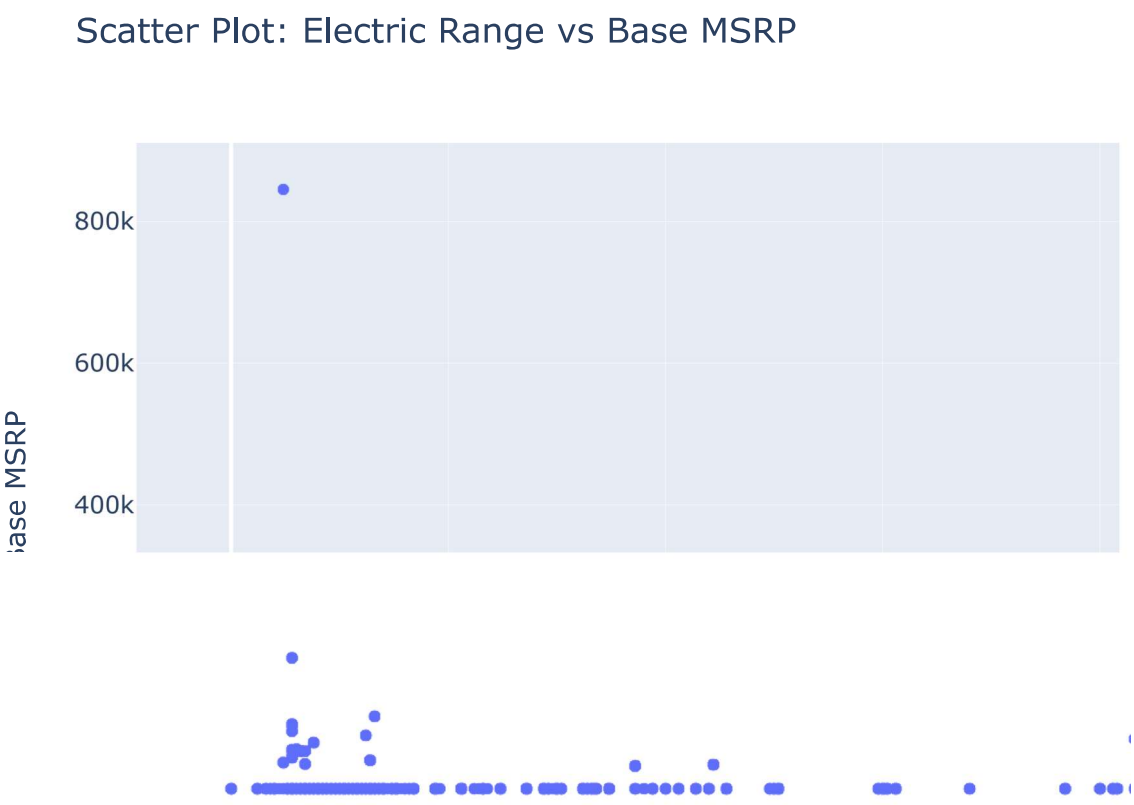
Task 2: Create a Choropleth using plotly.express to display the number of EV vehicles based on location.

In [35]:
```
!pip install plotly
```

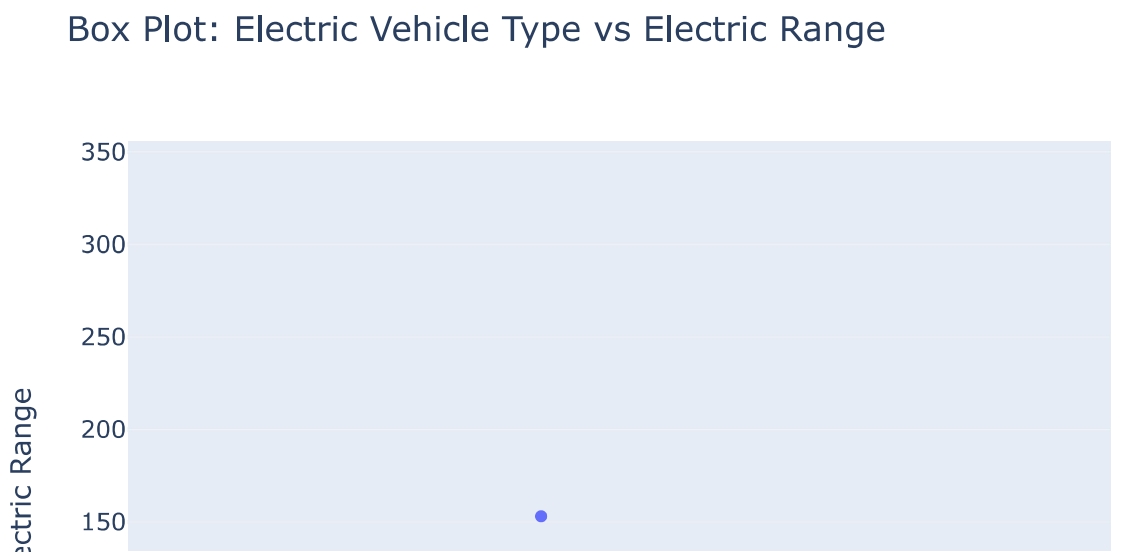
```
Requirement already satisfied: plotly in c:\users\sadgu\anaconda3\lib\site
-packages (5.9.0)
Requirement already satisfied: tenacity>=6.2.0 in c:\users\sadgu\anaconda3
\lib\site-packages (from plotly) (8.2.2)
```

In [36]:
```
import plotly.express as px
```

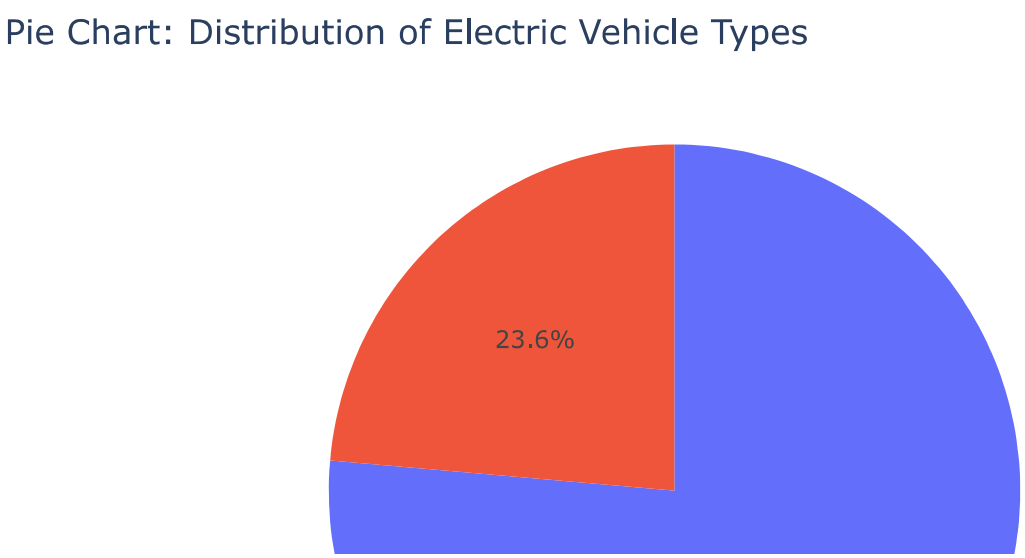
In [37]:
```
scatter_plot = px.scatter(p, x="Electric Range", y="Base MSRP", title="Scat
scatter_plot.show()
```



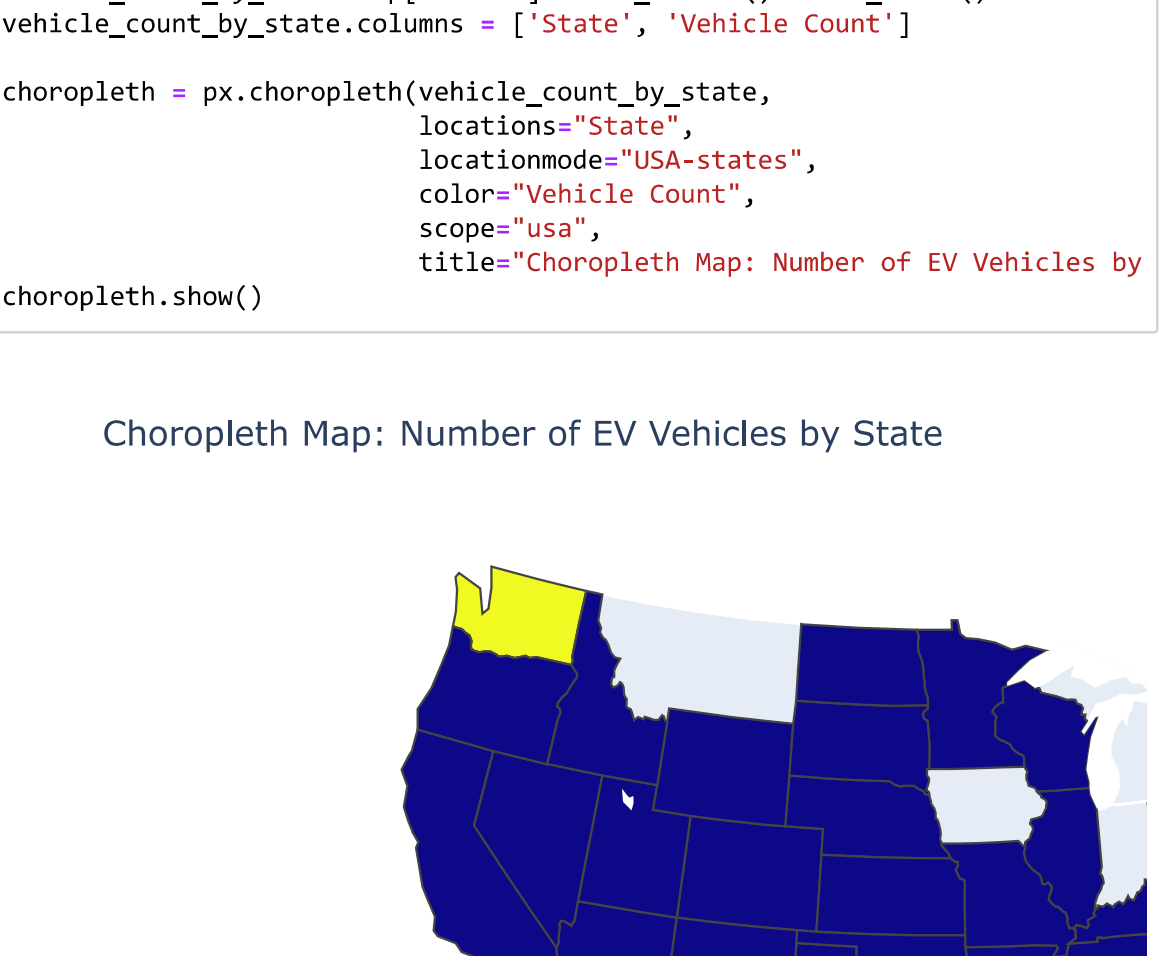
In [39]:
```
box_plot = px.box(p, x="Electric Vehicle Type", y="Electric Range", title="
box_plot.show()
```



In [40]:
```
vehicle_type_count = p['Electric Vehicle Type'].value_counts().reset_index(
vehicle_type_count.columns = ['Electric Vehicle Type', 'Count']
pie_chart = px.pie(vehicle_type_count, names='Electric Vehicle Type', value
pie_chart.show()
```

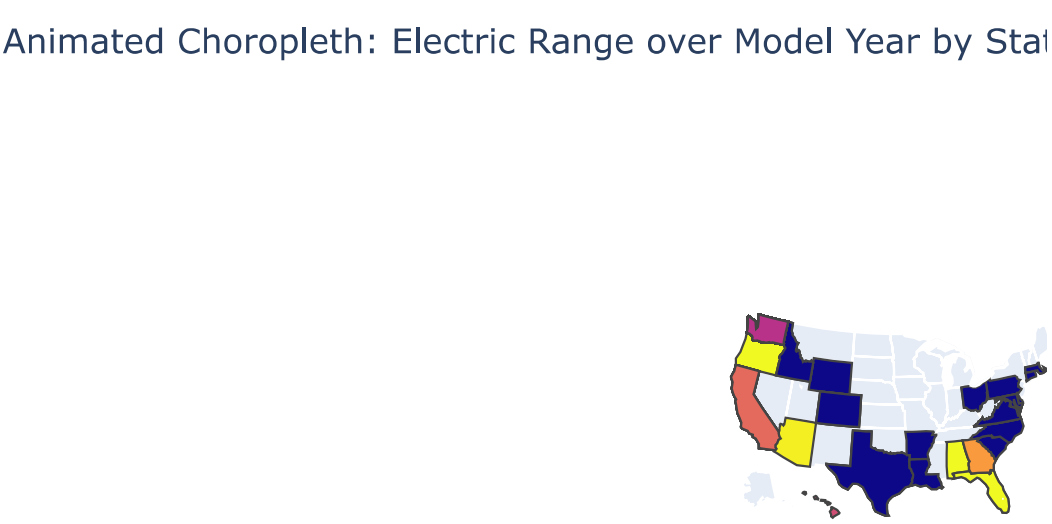


In [41]:
```
vehicle_count_by_state = p['State'].value_counts().reset_index()
vehicle_count_by_state.columns = ['State', 'Vehicle Count']

choropleth = px.choropleth(vehicle_count_by_state,
                           locations="State",
                           locationmode="USA-states",
                           color="Vehicle Count",
                           scope="usa",
                           title="Choropleth Map: Number of EV Vehicles by
choropleth.show()
```



In [43]:
```
animated_choropleth = px.choropleth(p,
                                    locations="State",
                                    locationmode="USA-states",
                                    color="Electric Range",
                                    animation_frame="Model Year",
                                    scope="usa",
                                    title="Animated Choropleth: Electric Ra
animated_choropleth.show()
```

```
C:\Users\sadgu\anaconda3\Lib\site-packages\plotly\express\_core.py:1979: F
utureWarning:

When grouping with a length-1 list-like, you will need to pass a length-1
tuple to get_group in a future version of pandas. Pass `(name,)` instead o
f `name` to silence this warning.
```



Task 3: Create a Racing Bar Plot to display the animation of EV Make and its count each year.

In [44]:
```
!pip install bar_chart_race
```

```
Collecting bar_chart_race
  Obtaining dependency information for bar_chart_race from https://files.p
ythonhosted.org/packages/09/01/f6d1a1a0978b39560843c54be7349804d7d2faef0a8
69acd7c8a6fc920b0/bar_chart_race-0.1.0-py3-none-any.whl.metadata (https://
files.pythonhosted.org/packages/09/01/f6d1a1a0978b39560843c54be7349804d7d2
faef0a869acd7c8a6fc920b0/bar_chart_race-0.1.0-py3-none-any.whl.metadata)
  Downloading bar_chart_race-0.1.0-py3-none-any.whl.metadata (4.2 kB)
Requirement already satisfied: pandas>=0.24 in c:\users\sadgu\anaconda3\li
b\site-packages (from bar_chart_race) (2.2.3)
Requirement already satisfied: matplotlib>=3.1 in c:\users\sadgu\anaconda3
\lib\site-packages (from bar_chart_race) (3.7.2)
Requirement already satisfied: contourpy>=1.0.1 in c:\users\sadgu\anaconda
3\lib\site-packages (from matplotlib>=3.1->bar_chart_race) (1.2.1)
Requirement already satisfied: cycler>=0.10 in c:\users\sadgu\anaconda3\li
b\site-packages (from matplotlib>=3.1->bar_chart_race) (0.11.0)
Requirement already satisfied: fonttools>=4.22.0 in c:\users\sadgu\anacond
a3\lib\site-packages (from matplotlib>=3.1->bar_chart_race) (4.25.0)
Requirement already satisfied: kiwisolver>=1.0.1 in c:\users\sadgu\anacond
a3\lib\site-packages (from matplotlib>=3.1->bar_chart_race) (1.4.4)
Requirement already satisfied: numpy>=1.20 in c:\users\sadgu\anaconda3\lib
\site-packages (from matplotlib>=3.1->bar_chart_race) (1.24.3)
Requirement already satisfied: packaging>=20.0 in c:\users\sadgu\anaconda3
\lib\site-packages (from matplotlib>=3.1->bar_chart_race) (23.1)
Requirement already satisfied: pillow>=6.2.0 in c:\users\sadgu\anaconda3\l
ib\site-packages (from matplotlib>=3.1->bar_chart_race) (9.4.0)
Requirement already satisfied: pyparsing<3.1,>=2.3.1 in c:\users\sadgu\ana
conda3\lib\site-packages (from matplotlib>=3.1->bar_chart_race) (3.0.9)
Requirement already satisfied: python-dateutil>=2.7 in c:\users\sadgu\anac
onda3\lib\site-packages (from matplotlib>=3.1->bar_chart_race) (2.8.2)
Requirement already satisfied: pytz>=2020.1 in c:\users\sadgu\anaconda3\li
b\site-packages (from pandas>=0.24->bar_chart_race) (2023.3.post1)
Requirement already satisfied: tzdata>=2022.7 in c:\users\sadgu\anaconda3
\lib\site-packages (from pandas>=0.24->bar_chart_race) (2023.3)
Requirement already satisfied: six>=1.5 in c:\users\sadgu\anaconda3\lib\si
te-packages (from python-dateutil>=2.7->matplotlib>=3.1->bar_chart_race)
(1.16.0)
Downloading bar_chart_race-0.1.0-py3-none-any.whl (156 kB)
   ---------------------------------------- 0.0/156.8 kB ? eta -:--:--
   ------------------ --------------------- 71.7/156.8 kB 4.1 MB/s eta 0:0
0:01
   ------------------ --------------------- 71.7/156.8 kB 4.1 MB/s eta 0:0
0:01
   ------------------ --------------------- 71.7/156.8 kB 4.1 MB/s eta 0:0
0:01
   ---------------------- ---------------- 92.2/156.8 kB 655.4 kB/s eta 0:
00:01
   ---------------------- ---------------- 92.2/156.8 kB 655.4 kB/s eta 0:
00:01
   ---------------------------------------- 156.8/156.8 kB 672.4 kB/s eta 0:
00:00
Installing collected packages: bar_chart_race
Successfully installed bar_chart_race-0.1.0
```

In [47]:
```
# Assuming you have already loaded your dataset
# Create a pivot table with counts of vehicles by 'Make' and 'Model Year'
pivot_data = p.pivot_table(index="Model Year", columns="Make", aggfunc="siz

# Sort the columns by sum of vehicle counts
pivot_data = pivot_data.loc[:, pivot_data.sum(axis=0).sort_values(ascending
```

In [46]:
```
import bar_chart_race as bcr
```

In [49]:

```
# Create a pivot table with counts of vehicles by 'Make' and 'Model Year'
pivot_data = p.pivot_table(index="Model Year", columns="Make", aggfunc="siz

# Reset index to make 'Model Year' a column
pivot_data.reset_index(inplace=True)
melted_data = pivot_data.melt(id_vars=["Model Year"], var_name="Make", valu

# Create an animated bar plot
fig = px.bar(melted_data,
             x='Count',
             y='Make',
             color='Make',
             animation_frame='Model Year',
             range_x=[0, melted_data['Count'].max() + 10],  # Adjust range
             title='Year-wise EV Make Sales Animation',
             orientation='h')


fig.update_layout(
    title_font=dict(size=30),
    xaxis_title_font=dict(size=20),
    yaxis_title_font=dict(size=20),
    width=1000,
    height=600,
    bargap=0.1,
)
```

Year-wise EV Make Sales Animatior