In [2]:
```
!pip install scikit-learn
```

```
Collecting scikit-learn
  Downloading scikit_learn-1.5.2-cp312-cp312-win_amd64.whl.metadata (13 kB)
Requirement already satisfied: numpy>=1.19.5 in c:\users\dell\appdata\local
\programs\python\python312\lib\site-packages (from scikit-learn) (2.1.1)
Requirement already satisfied: scipy>=1.6.0 in c:\users\dell\appdata\local\p
rograms\python\python312\lib\site-packages (from scikit-learn) (1.14.1)
Collecting joblib>=1.2.0 (from scikit-learn)
  Downloading joblib-1.4.2-py3-none-any.whl.metadata (5.4 kB)
Collecting threadpoolctl>=3.1.0 (from scikit-learn)
  Downloading threadpoolctl-3.5.0-py3-none-any.whl.metadata (13 kB)
Downloading scikit_learn-1.5.2-cp312-cp312-win_amd64.whl (11.0 MB)
   ---------------------------------------- 0.0/11.0 MB ? eta -:--:--
   ---------------------------------------- 0.0/11.0 MB ? eta -:--:--
   --- ------------------------------------ 1.0/11.0 MB 4.2 MB/s eta 0:00:03
   ------ --------------------------------- 1.8/11.0 MB 4.2 MB/s eta 0:00:03
   --------- ------------------------------ 2.6/11.0 MB 4.1 MB/s eta 0:00:03
   ----------- ---------------------------- 3.1/11.0 MB 3.7 MB/s eta 0:00:03
   ------------ --------------------------- 3.4/11.0 MB 3.6 MB/s eta 0:00:03
   -------------- ------------------------- 3.9/11.0 MB 3.1 MB/s eta 0:00:03
   ---------------- ----------------------- 4.5/11.0 MB 2.9 MB/s eta 0:00:03
   ------------------- -------------------- 5.2/11.0 MB 3.0 MB/s eta 0:00:02
   -------------------- ------------------- 5.8/11.0 MB 3.0 MB/s eta 0:00:02
   ----------------------- ---------------- 6.6/11.0 MB 3.1 MB/s eta 0:00:02
   -------------------------- ------------- 7.3/11.0 MB 3.1 MB/s eta 0:00:02
   --------------------------- ------------ 7.6/11.0 MB 3.1 MB/s eta 0:00:02
   ----------------------------- ---------- 8.1/11.0 MB 3.0 MB/s eta 0:00:01
   -------------------------------- ------- 8.9/11.0 MB 3.0 MB/s eta 0:00:01
   ----------------------------------- ---- 9.7/11.0 MB 3.1 MB/s eta 0:00:01
   ------------------------------------- -- 10.2/11.0 MB 3.0 MB/s eta 0:00:0
1
   ---------------------------------------  10.7/11.0 MB 3.0 MB/s eta 0:00:0
1
   ---------------------------------------- 11.0/11.0 MB 3.0 MB/s eta 0:00:0
0
Downloading joblib-1.4.2-py3-none-any.whl (301 kB)
Downloading threadpoolctl-3.5.0-py3-none-any.whl (18 kB)
Installing collected packages: threadpoolctl, joblib, scikit-learn
Successfully installed joblib-1.4.2 scikit-learn-1.5.2 threadpoolctl-3.5.0
```

In [6]:
```
!pip install plotly
!pip install bar-chart-race
```

```
Requirement already satisfied: plotly in c:\users\dell\appdata\local\program
s\python\python312\lib\site-packages (5.24.1)
Requirement already satisfied: tenacity>=6.2.0 in c:\users\dell\appdata\loca
l\programs\python\python312\lib\site-packages (from plotly) (9.0.0)
Requirement already satisfied: packaging in c:\users\dell\appdata\local\prog
rams\python\python312\lib\site-packages (from plotly) (24.1)
Collecting bar-chart-race
  Downloading bar_chart_race-0.1.0-py3-none-any.whl.metadata (4.2 kB)
Requirement already satisfied: pandas>=0.24 in c:\users\dell\appdata\local\p
rograms\python\python312\lib\site-packages (from bar-chart-race) (2.2.3)
Requirement already satisfied: matplotlib>=3.1 in c:\users\dell\appdata\loca
l\programs\python\python312\lib\site-packages (from bar-chart-race) (3.9.2)
Requirement already satisfied: contourpy>=1.0.1 in c:\users\dell\appdata\loc
al\programs\python\python312\lib\site-packages (from matplotlib>=3.1->bar-ch
art-race) (1.3.0)
Requirement already satisfied: cycler>=0.10 in c:\users\dell\appdata\local\p
rograms\python\python312\lib\site-packages (from matplotlib>=3.1->bar-chart-
race) (0.12.1)
Requirement already satisfied: fonttools>=4.22.0 in c:\users\dell\appdata\lo
cal\programs\python\python312\lib\site-packages (from matplotlib>=3.1->bar-c
hart-race) (4.54.1)
Requirement already satisfied: kiwisolver>=1.3.1 in c:\users\dell\appdata\lo
cal\programs\python\python312\lib\site-packages (from matplotlib>=3.1->bar-c
hart-race) (1.4.7)
Requirement already satisfied: numpy>=1.23 in c:\users\dell\appdata\local\pr
ograms\python\python312\lib\site-packages (from matplotlib>=3.1->bar-chart-r
ace) (2.1.1)
Requirement already satisfied: packaging>=20.0 in c:\users\dell\appdata\loca
l\programs\python\python312\lib\site-packages (from matplotlib>=3.1->bar-cha
rt-race) (24.1)
Requirement already satisfied: pillow>=8 in c:\users\dell\appdata\local\prog
rams\python\python312\lib\site-packages (from matplotlib>=3.1->bar-chart-rac
e) (10.4.0)
Requirement already satisfied: pyparsing>=2.3.1 in c:\users\dell\appdata\loc
al\programs\python\python312\lib\site-packages (from matplotlib>=3.1->bar-ch
art-race) (3.1.4)
Requirement already satisfied: python-dateutil>=2.7 in c:\users\dell\appdata
\local\programs\python\python312\lib\site-packages (from matplotlib>=3.1->ba
r-chart-race) (2.9.0.post0)
Requirement already satisfied: pytz>=2020.1 in c:\users\dell\appdata\local\p
rograms\python\python312\lib\site-packages (from pandas>=0.24->bar-chart-rac
e) (2024.2)
Requirement already satisfied: tzdata>=2022.7 in c:\users\dell\appdata\local
\programs\python\python312\lib\site-packages (from pandas>=0.24->bar-chart-r
ace) (2024.2)
Requirement already satisfied: six>=1.5 in c:\users\dell\appdata\local\progr
ams\python\python312\lib\site-packages (from python-dateutil>=2.7->matplotli
b>=3.1->bar-chart-race) (1.16.0)
Downloading bar_chart_race-0.1.0-py3-none-any.whl (156 kB)
Installing collected packages: bar-chart-race
Successfully installed bar-chart-race-0.1.0
```

In [7]:

```python
#Import required libraries
import pandas as pd
import numpy as np
import seaborn as sns
```

```python
import matplotlib.pyplot as plt
from sklearn.impute import SimpleImputer
import plotly.express as px
```

In [8]:
```python
#Reading the .csv file
df=pd.read_csv(r"C:\Users\DELL\OneDrive\Desktop\Innomatics\dataset.csv")
df.head()
```

Out[8]:

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

In [9]:
```python
#shape of the data
shape=df.shape
print("The Number of rows : {}".format(shape[0]))
print("The Number of columns : {}".format(shape[1]))
```

```
The Number of rows : 112634
The Number of columns : 17
```

In [4]:
```python
# Information about the data
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 112634 entries, 0 to 112633
Data columns (total 17 columns):
 #   Column                                             Non-Null Count   Dty
pe
---  ------                                             --------------   ---
--
 0   VIN (1-10)                                         112634 non-null  obj
ect
 1   County                                             112634 non-null  obj
ect
 2   City                                               112634 non-null  obj
ect
 3   State                                              112634 non-null  obj
ect
 4   Postal Code                                        112634 non-null  int
64
 5   Model Year                                         112634 non-null  int
64
 6   Make                                               112634 non-null  obj
ect
 7   Model                                              112614 non-null  obj
ect
 8   Electric Vehicle Type                              112634 non-null  obj
ect
 9   Clean Alternative Fuel Vehicle (CAFV) Eligibility  112634 non-null  obj
ect
 10  Electric Range                                     112634 non-null  int
64
 11  Base MSRP                                          112634 non-null  int
64
 12  Legislative District                               112348 non-null  flo
at64
 13  DOL Vehicle ID                                     112634 non-null  int
64
 14  Vehicle Location                                   112610 non-null  obj
ect
 15  Electric Utility                                   112191 non-null  obj
ect
 16  2020 Census Tract                                  112634 non-null  int
64
dtypes: float64(1), int64(6), object(10)
memory usage: 14.6+ MB
```

Exploratory data Analysis

Getting the insights from the data which includes

- Missing values.
- Duplicated Values.
- Outliers.
- Relationships.
- Distributions.

In [5]:
```python
# Checking the Missing values
df.isna().sum()
```

Out[5]:
```
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
```

Insights

- There are 20 missing values in Model column.
- 286 missing values in Legislative District.
- 443 Missing values in Electric Utility.

In [6]:
```python
# Checking the Duplicated values
df.duplicated().sum()
```

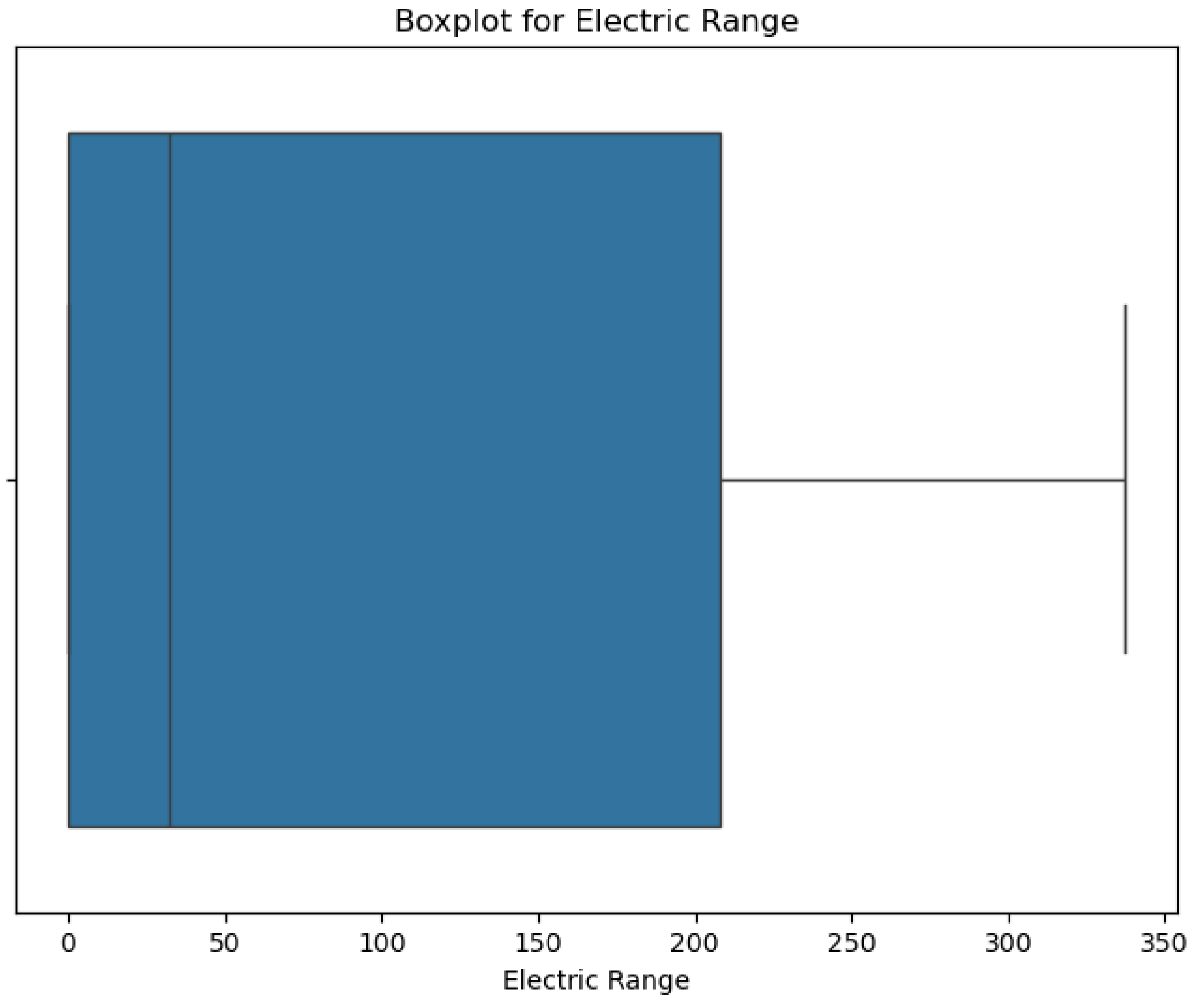
Out[6]:
```
0
```

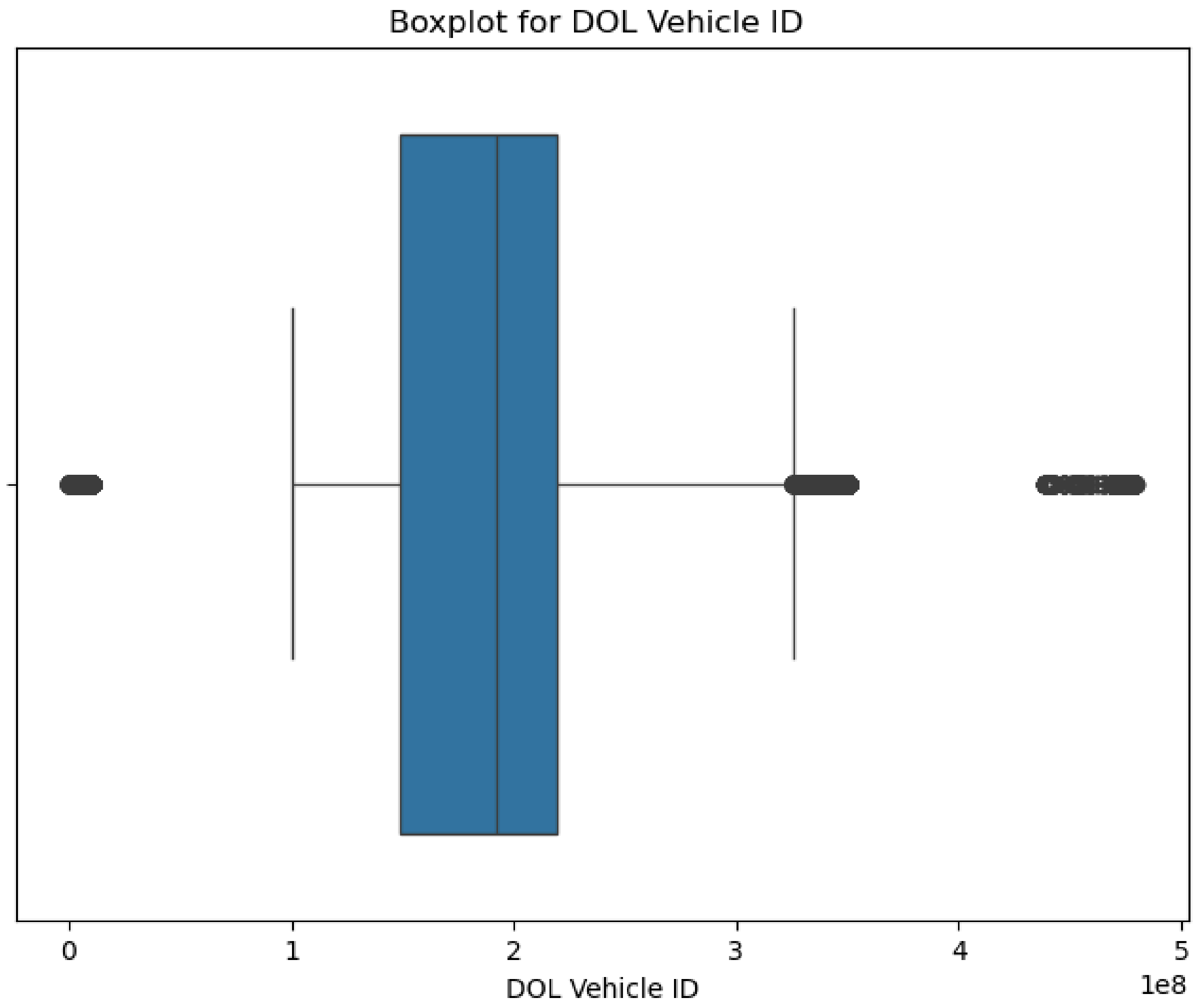
Insights

- There are no duplicated values in the data.

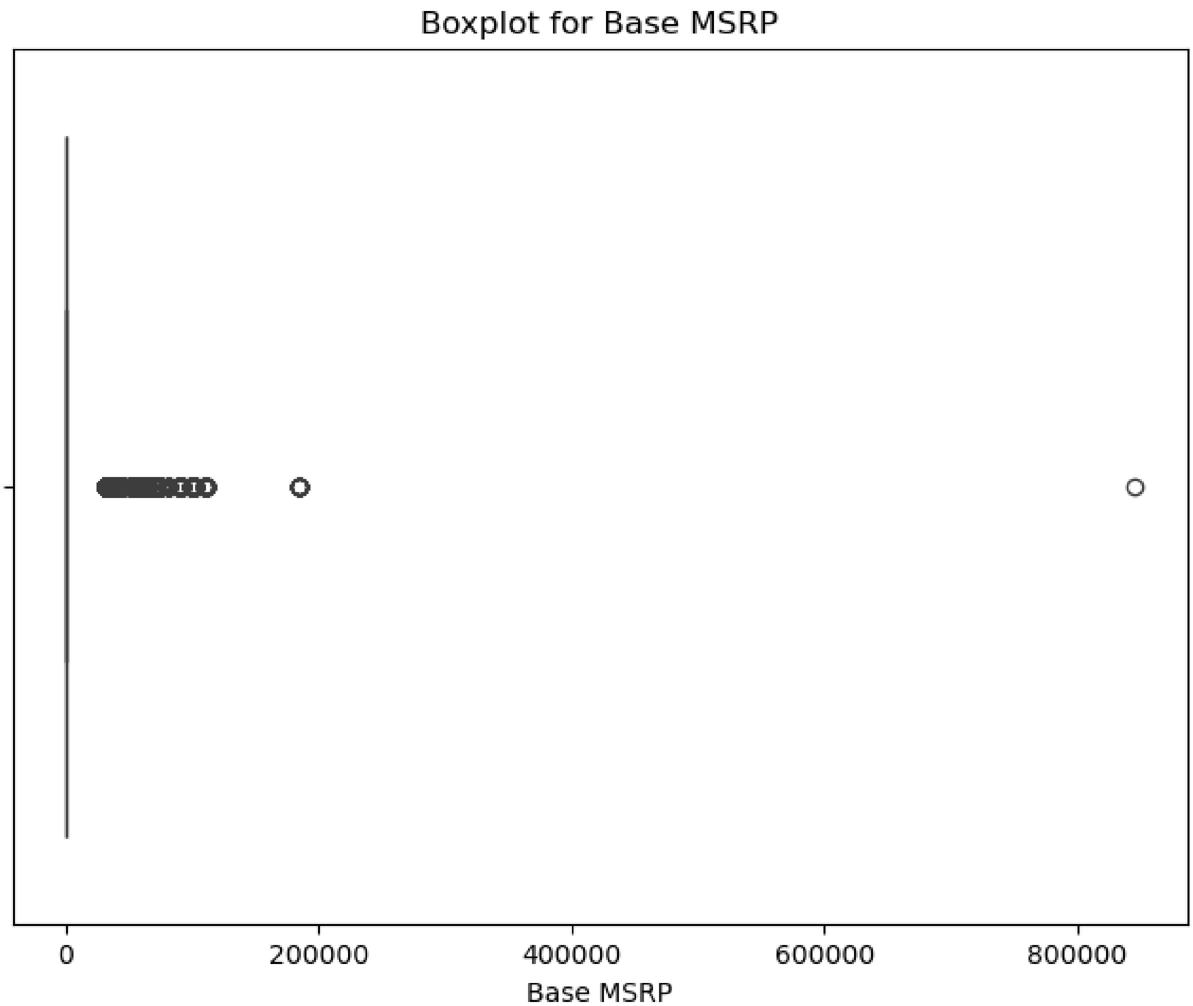
In [7]:
```python
# Checking the outliers
plt.figure(figsize=(8,6))
sns.boxplot(x=df["Electric Range"])
plt.title("Boxplot for Electric Range")
plt.show()

plt.figure(figsize=(8,6))
sns.boxplot(x=df["DOL Vehicle ID"])
plt.title("Boxplot for DOL Vehicle ID ")
plt.show

plt.figure(figsize=(8,6))
```

```
sns.boxplot(x=df["Base MSRP"])
plt.title("Boxplot for Base MSRP")
plt.show()
```





Boxplot for DOL Vehicle ID
0
1
2
3
4
5
1e8
DOL Vehicle ID



Imputing the missing values

In [8]:
```
Missing_columns=["Model","Legislative District","2020 Census Tract"]
```

In [9]:
```
SIM=SimpleImputer(strategy="most_frequent")
SIM
```

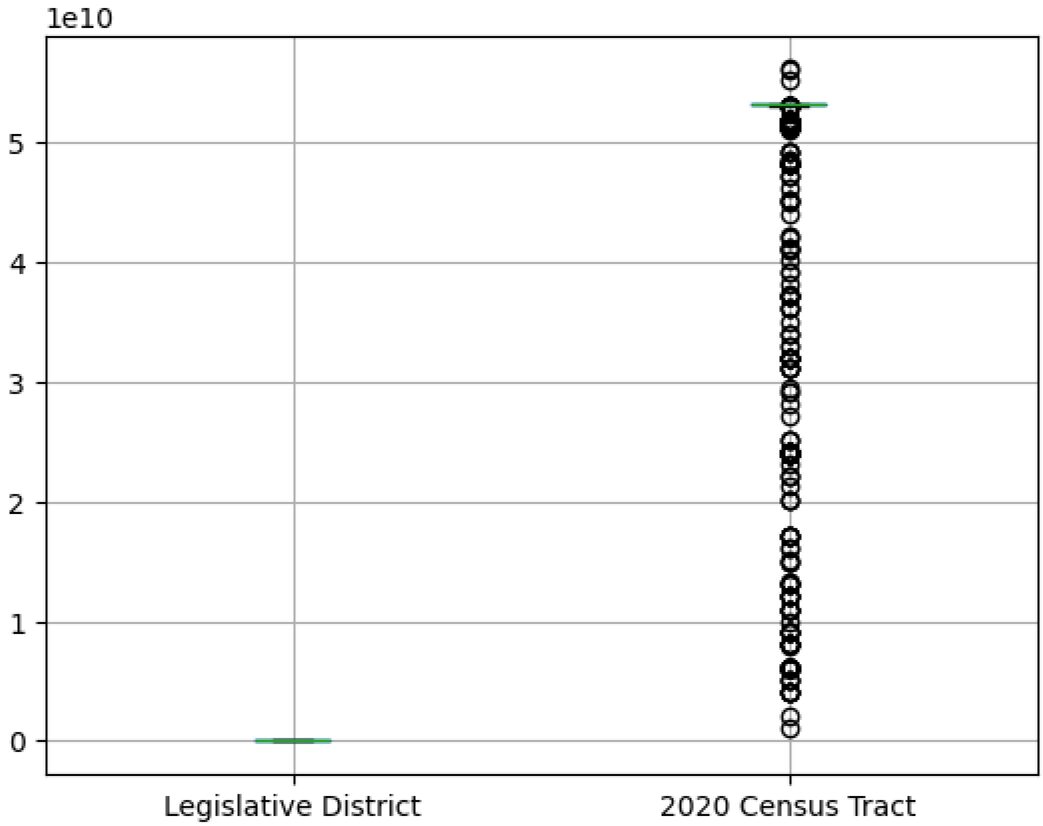
Out[9]:
```
SimpleImputer
SimpleImputer(strategy='most_frequent')
```

In [10]:
```
df[["Model"]]=SIM.fit_transform(df[["Model"]])
```

In [11]:
```
df["Model"].isna().sum()
```

Out[11]:
```
0
```

In [12]:
```
df[["Legislative District","2020 Census Tract"]].boxplot()
plt.show()
```



In [13]:
```
SIM=SimpleImputer(strategy="mean")
df[["2020 Census Tract"]]=SIM.fit_transform(df[["2020 Census Tract"]])
df["2020 Census Tract"].isna().sum()
```

Out[13]: 0

In [14]:
```
SIM=SimpleImputer(strategy="median")
df[["Legislative District"]]=SIM.fit_transform(df[["Legislative District"]])
df["Legislative District"].isna().sum()
```

Out[14]: 0

Univariate Analysis

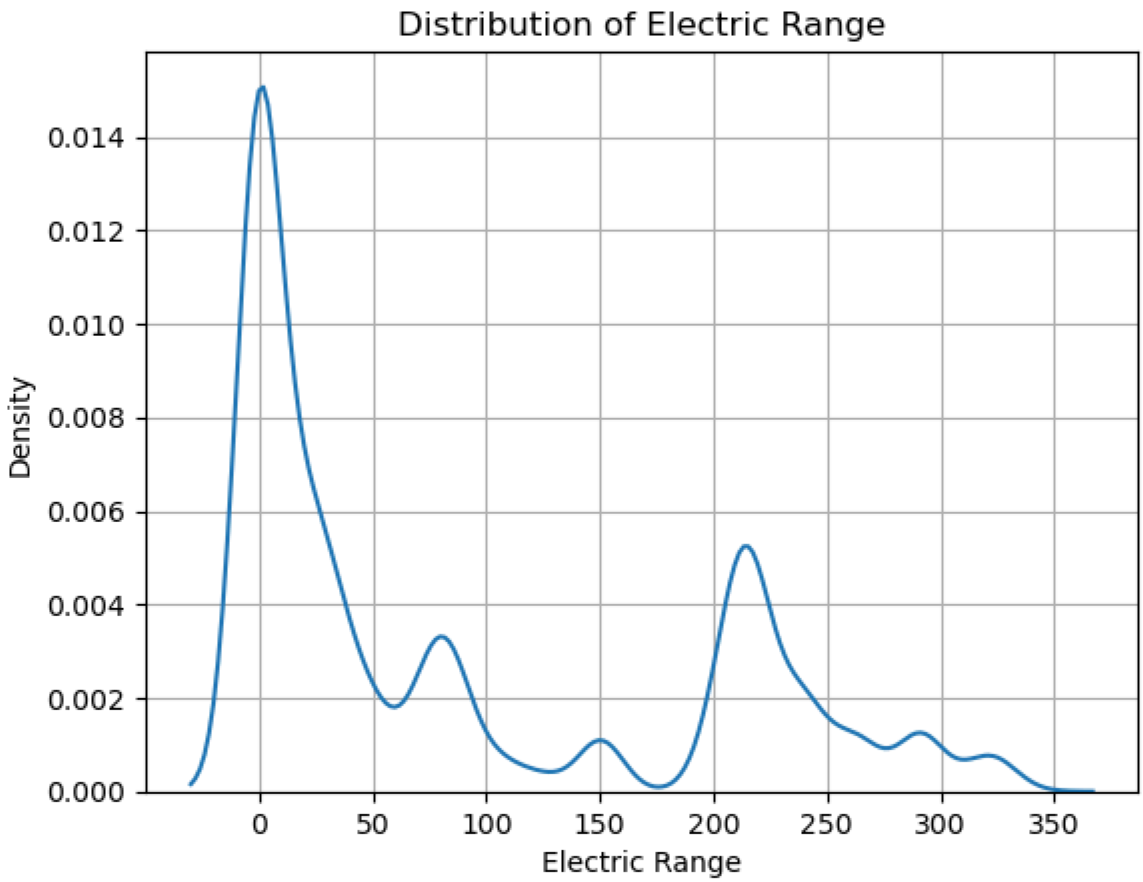
- Analysing the data using single feature.

In [15]:
```
df.columns
```

Out[15]:
```
Index(['VIN (1-10)', 'County', 'City', 'State', 'Postal Code', 'Model Yea
r',
       'Make', 'Model', 'Electric Vehicle Type',
       'Clean Alternative Fuel Vehicle (CAFV) Eligibility', 'Electric Rang
e',
       'Base MSRP', 'Legislative District', 'DOL Vehicle ID',
       'Vehicle Location', 'Electric Utility', '2020 Census Tract'],
      dtype='object')
```

What is the distribution of Electric Range?

In [16]:
```
sns.kdeplot(x=df["Electric Range"])
plt.title("Distribution of Electric Range")
plt.grid()
plt.show()
```



Insights

- In between 0 to 45 the electric range density is more compared to 5 to 100.
- Above 350 the electric range is decreasing.

Distribution of City?

In [17]:
```
d1=pd.DataFrame(df["City"].value_counts())
d1
```

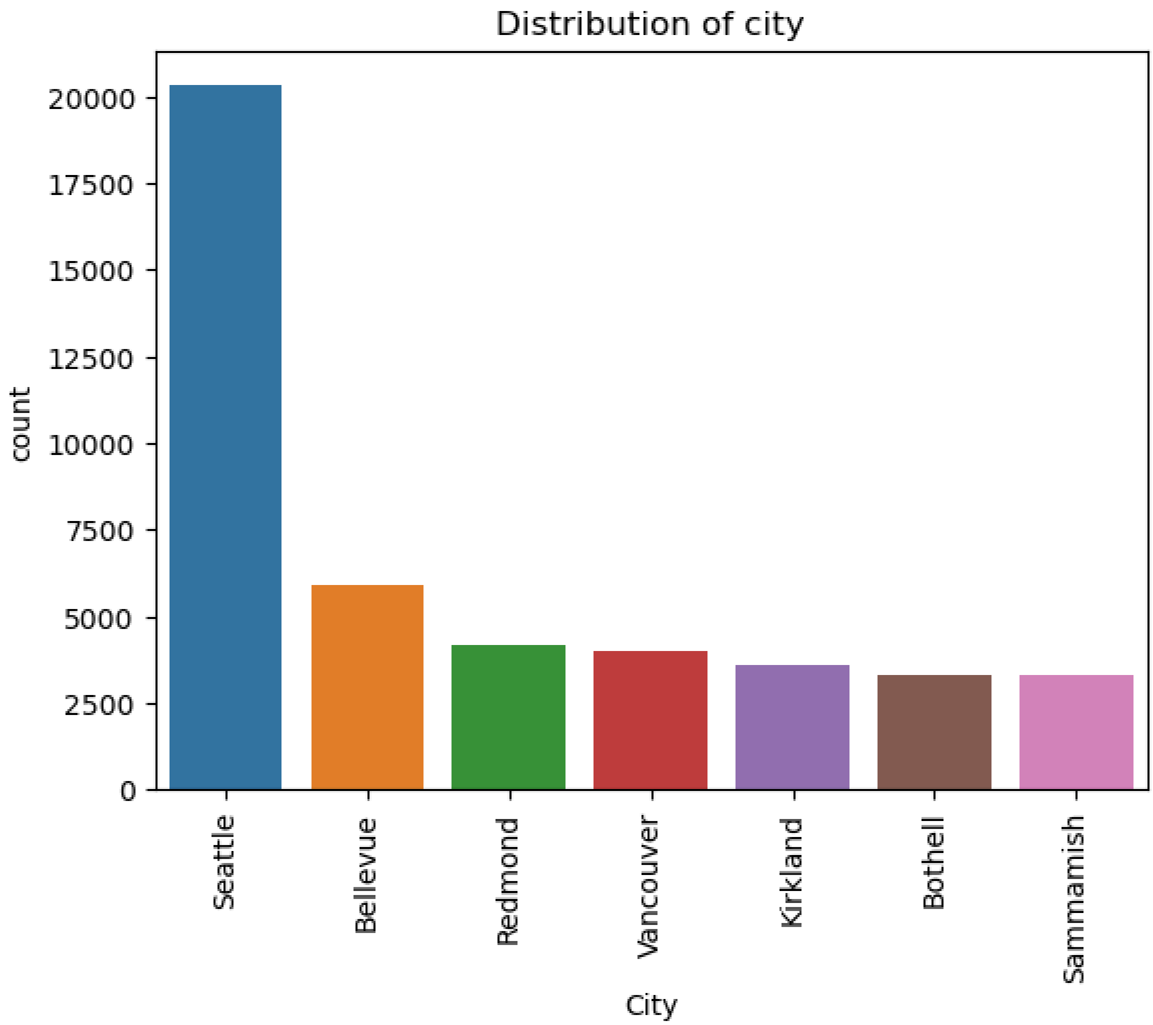
Out[17]:

City	count
Seattle	20305
Bellevue	5921
Redmond	4201
Vancouver	4013
Kirkland	3598
...	...
Hartline	1
Gaithersburg	1
El Paso	1
Klickitat	1
Worley	1

629 rows × 1 columns

In [18]:

```
sns.barplot(x=d1.index[:7],y=d1["count"][:7],hue=d1.index[:7])
plt.title("Distribution of city")
plt.xticks(rotation=90)
plt.show()
```



Insights

- Seattle is ranked more in distribution of cities.
- Worley is less compared to other cities.

Dstribution of Make?

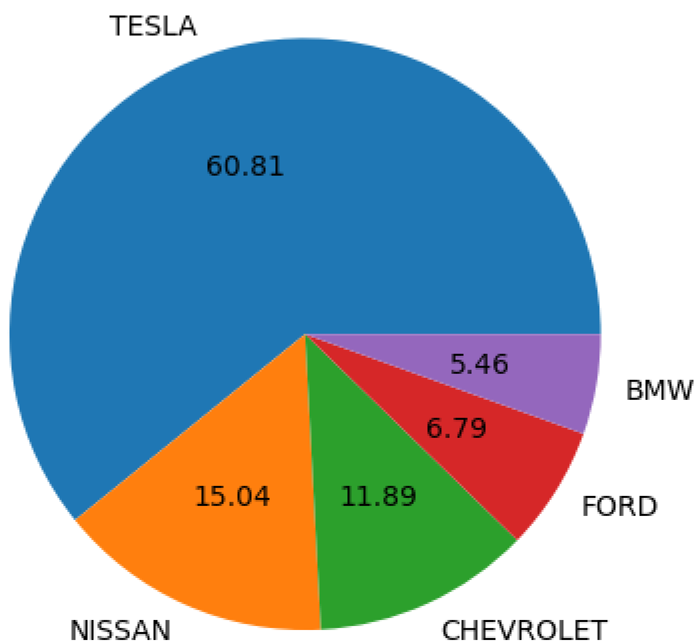
In [19]:
```
d2=pd.DataFrame(df["Make"].value_counts())
d2
```

Out[19]:

Make	count
TESLA	52078
NISSAN	12880
CHEVROLET	10182
FORD	5819
BMW	4680
KIA	4483
TOYOTA	4405
VOLKSWAGEN	2514
AUDI	2332
VOLVO	2288
CHRYSLER	1794
HYUNDAI	1412
JEEP	1152
RIVIAN	885
FIAT	822
PORSCHE	818
HONDA	792
MINI	632
MITSUBISHI	588
POLESTAR	558
MERCEDES-BENZ	506
SMART	273
JAGUAR	219
LINCOLN	168
CADILLAC	108
LUCID MOTORS	65
SUBARU	59
LAND ROVER	38
LEXUS	33
FISKER	20
GENESIS	18
AZURE DYNAMICS	7

	count
Make	
TH!NK	3
BENTLEY	3

In [20]:
```
plt.pie(x=d2["count"][:5],labels=d2.index[:5],autopct="%0.2f")
plt.show()
```



Insights

- Tesla has the highest propotion in the make compared to others.

Distribution of State?

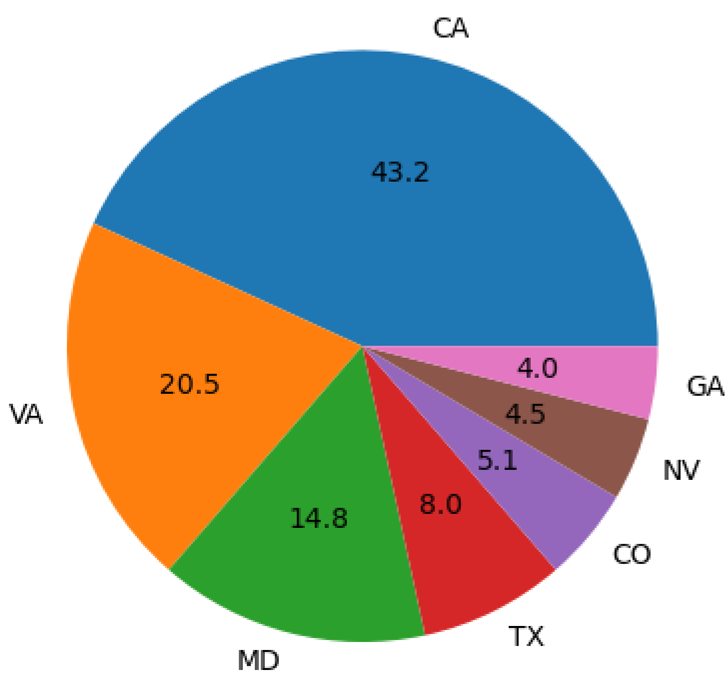
In [21]:
```
d3=pd.DataFrame(df["State"].value_counts())
d3
```

Out[21]:

State	count
WA	112348
CA	76
VA	36
MD	26
TX	14
CO	9
NV	8
GA	7
NC	7
CT	6
DC	6
FL	6
AZ	6
IL	6
SC	5
OR	5
NE	5
HI	4
UT	4
AR	4
NY	4
TN	3
KS	3
MO	3
PA	3
MA	3
LA	3
NJ	3
NH	2
OH	2
WY	2
ID	2

	count
State	
KY	1
RI	1
ME	1
MN	1
SD	1
WI	1
NM	1
AK	1
MS	1
AL	1
DE	1
OK	1
ND	1

In [22]:
```
plt.pie(x=d3["count"][1:8],labels=d3.index[1:8],autopct="%0.1f")
plt.show()
```



Bivariate Analysis

- Analysing the data using two features.

Which state has more Battery and least plug-in-hybrid electric type vehicles?

In [23]:
```
df.columns
```

Out[23]:
```
Index(['VIN (1-10)', 'County', 'City', 'State', 'Postal Code', 'Model Yea
r',
       'Make', 'Model', 'Electric Vehicle Type',
       'Clean Alternative Fuel Vehicle (CAFV) Eligibility', 'Electric Rang
e',
       'Base MSRP', 'Legislative District', 'DOL Vehicle ID',
       'Vehicle Location', 'Electric Utility', '2020 Census Tract'],
      dtype='object')
```

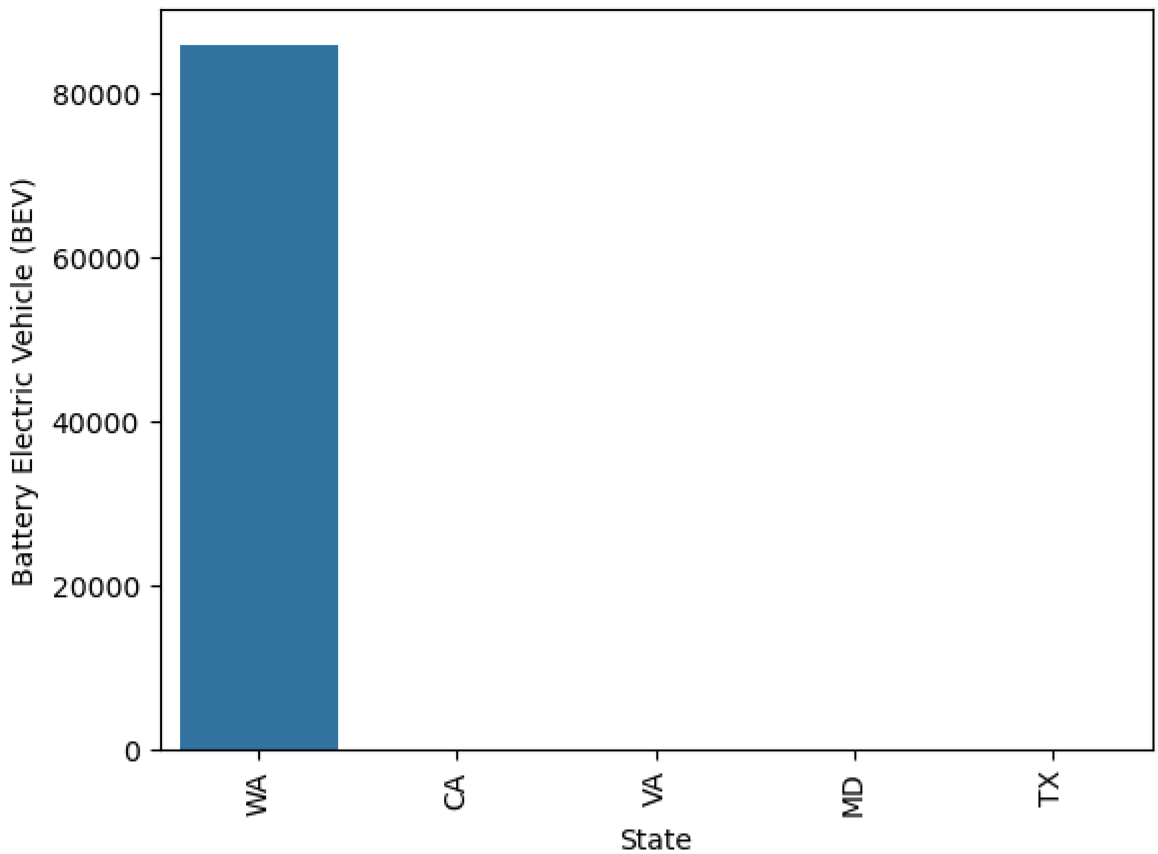
In [24]:
```
g1=pd.crosstab(index=df["State"],columns=df["Electric Vehicle Type"]).sort_v
g1.head()
len(g1)
```

Out[24]:
```
45
```

In [25]:
```
g1.index
```

Out[25]:
```
Index(['WA', 'CA', 'VA', 'MD', 'TX', 'CO', 'NV', 'IL', 'AZ', 'DC', 'SC', 'G
A',
       'NC', 'FL', 'NE', 'AR', 'NY', 'PA', 'TN', 'OR', 'HI', 'UT', 'KS', 'L
A',
       'MA', 'MO', 'ID', 'OH', 'WY', 'CT', 'NH', 'DE', 'MN', 'MS', 'NM', 'R
I',
       'SD', 'WI', 'NJ', 'AK', 'AL', 'KY', 'ME', 'ND', 'OK'],
      dtype='object', name='State')
```

In [26]:
```
sns.barplot(x=g1.index[:5],y=g1["Battery Electric Vehicle (BEV)"][:5],hue=g1
plt.xticks(rotation=90)
plt.show()
```

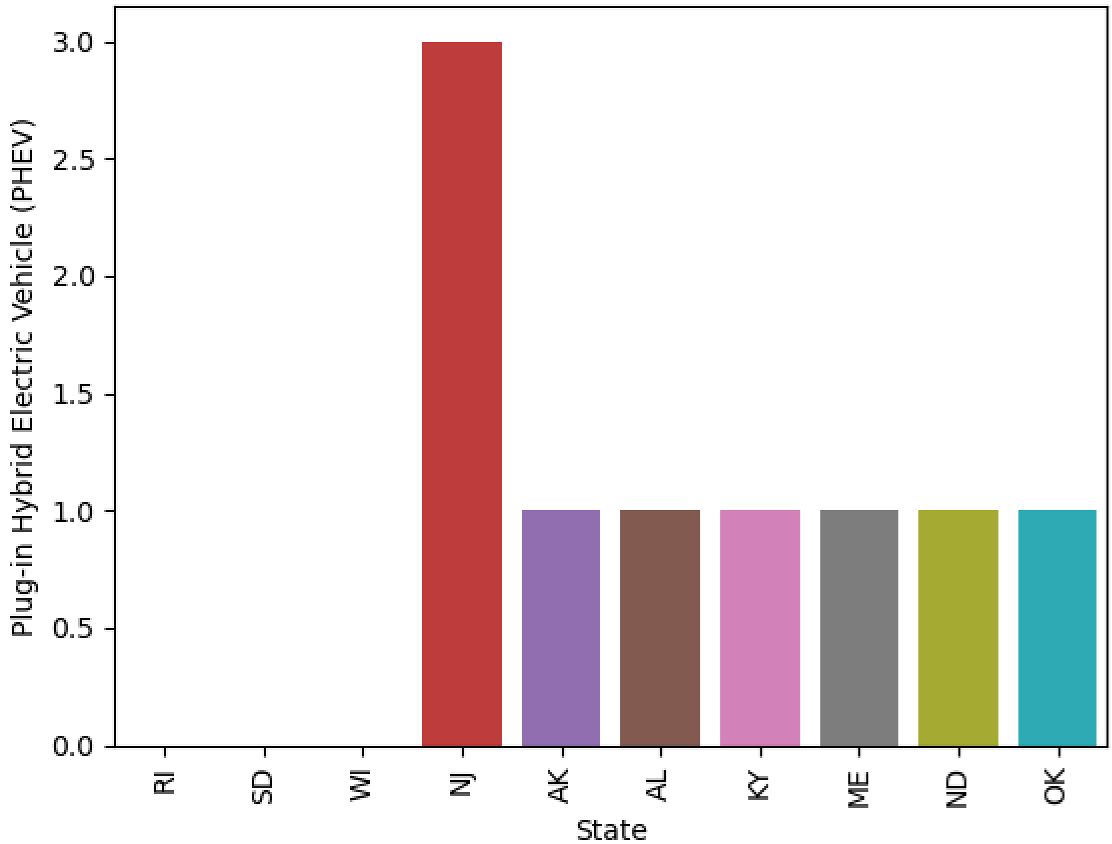


Insights

- WA has more Battery Electric vehicles compared to other states.

In [27]:

```
sns.barplot(x=g1.index[35:45],y=g1["Plug-in Hybrid Electric Vehicle (PHEV)"]
plt.xticks(rotation=90)
plt.show()
```



Insights

- OK,ND has less plug-in-hybrid electric vehicles.

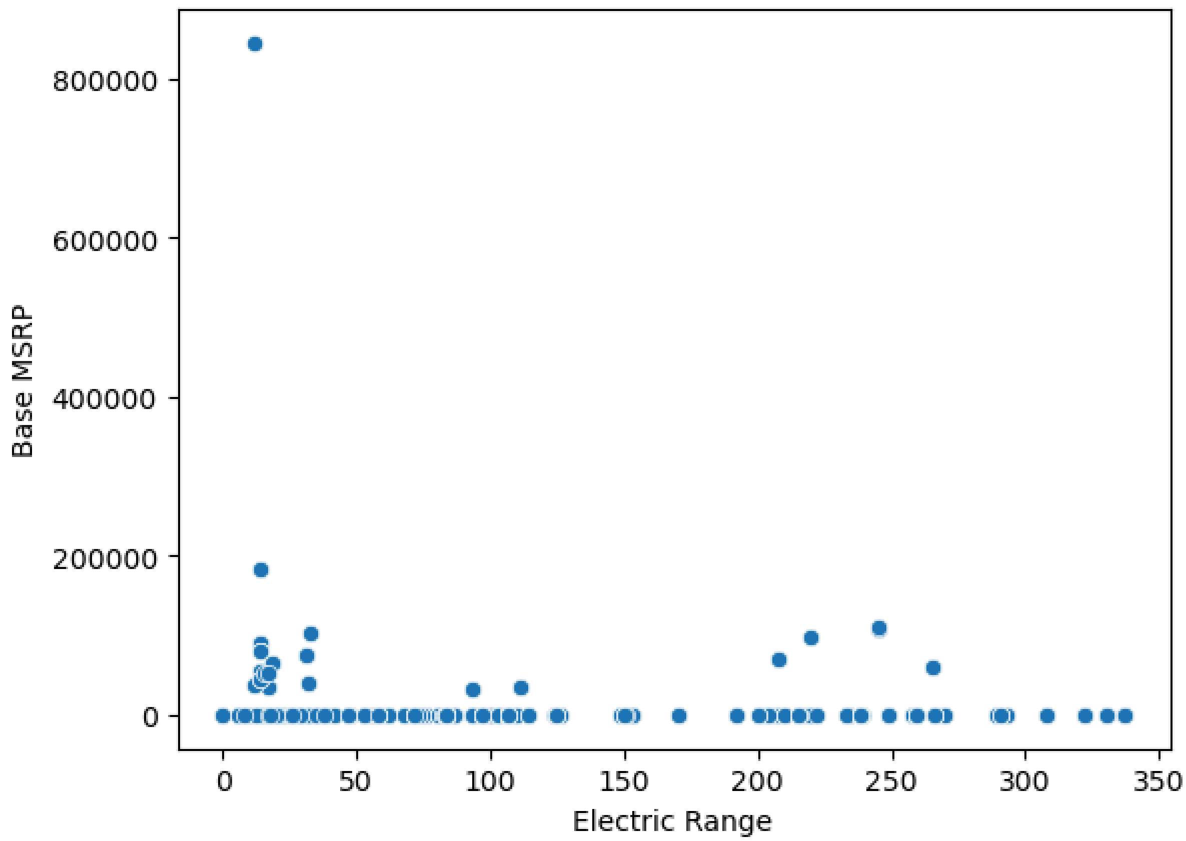
What is the relationship between the Electric Range and Base MSRP of electric vehicles?

In [28]:
```
df[["Electric Range","Base MSRP"]].corr()
```

Out[28]:

	Electric Range	Base MSRP
Electric Range	1.000000	0.085025
Base MSRP	0.085025	1.000000

In [29]:
```
sns.scatterplot(x=df["Electric Range"],y=df["Base MSRP"])
plt.show()
```

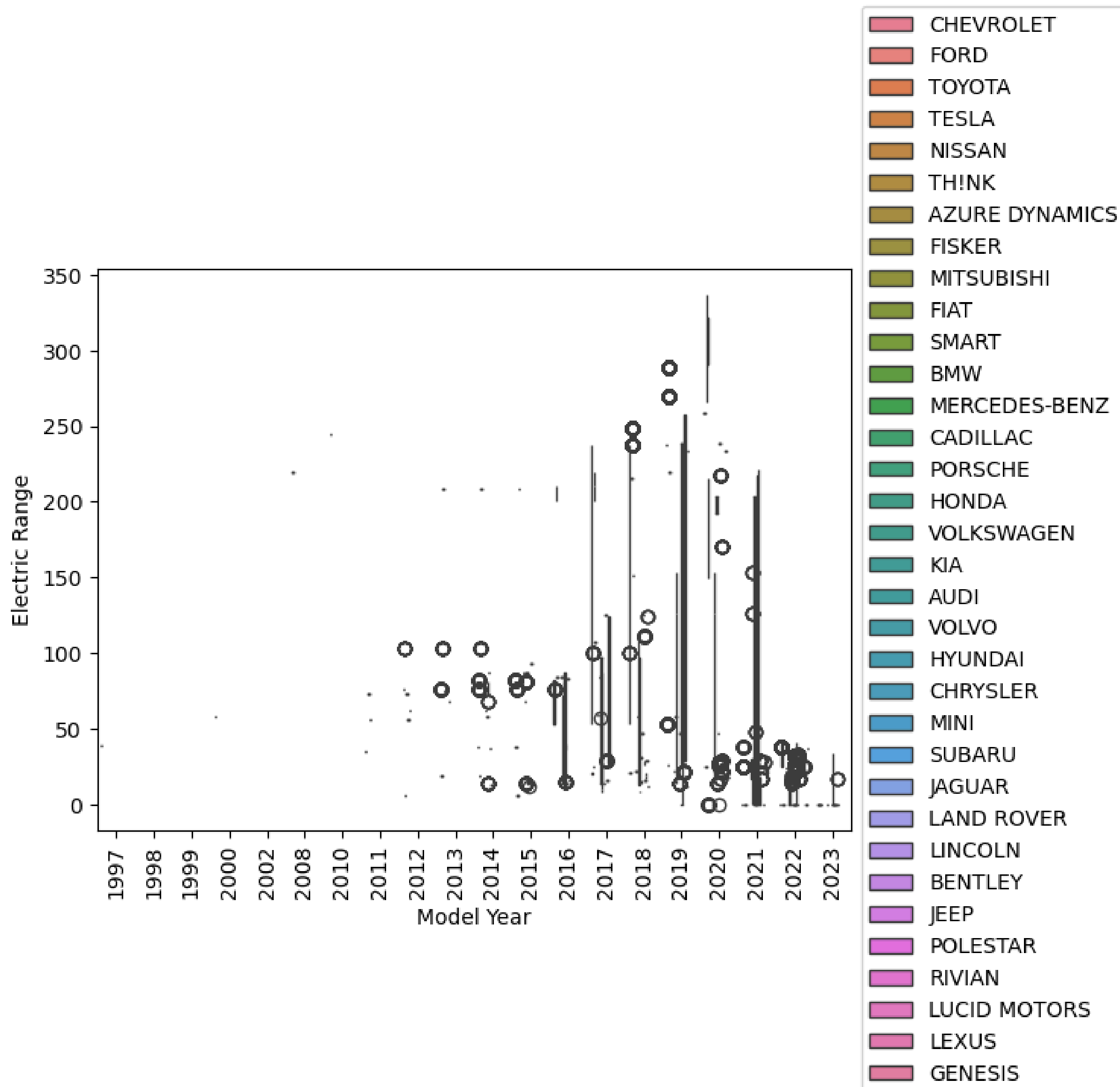


Insights

- Since the correlation is minimal, Electric Range is not a reliable predictor of the Base MSRP

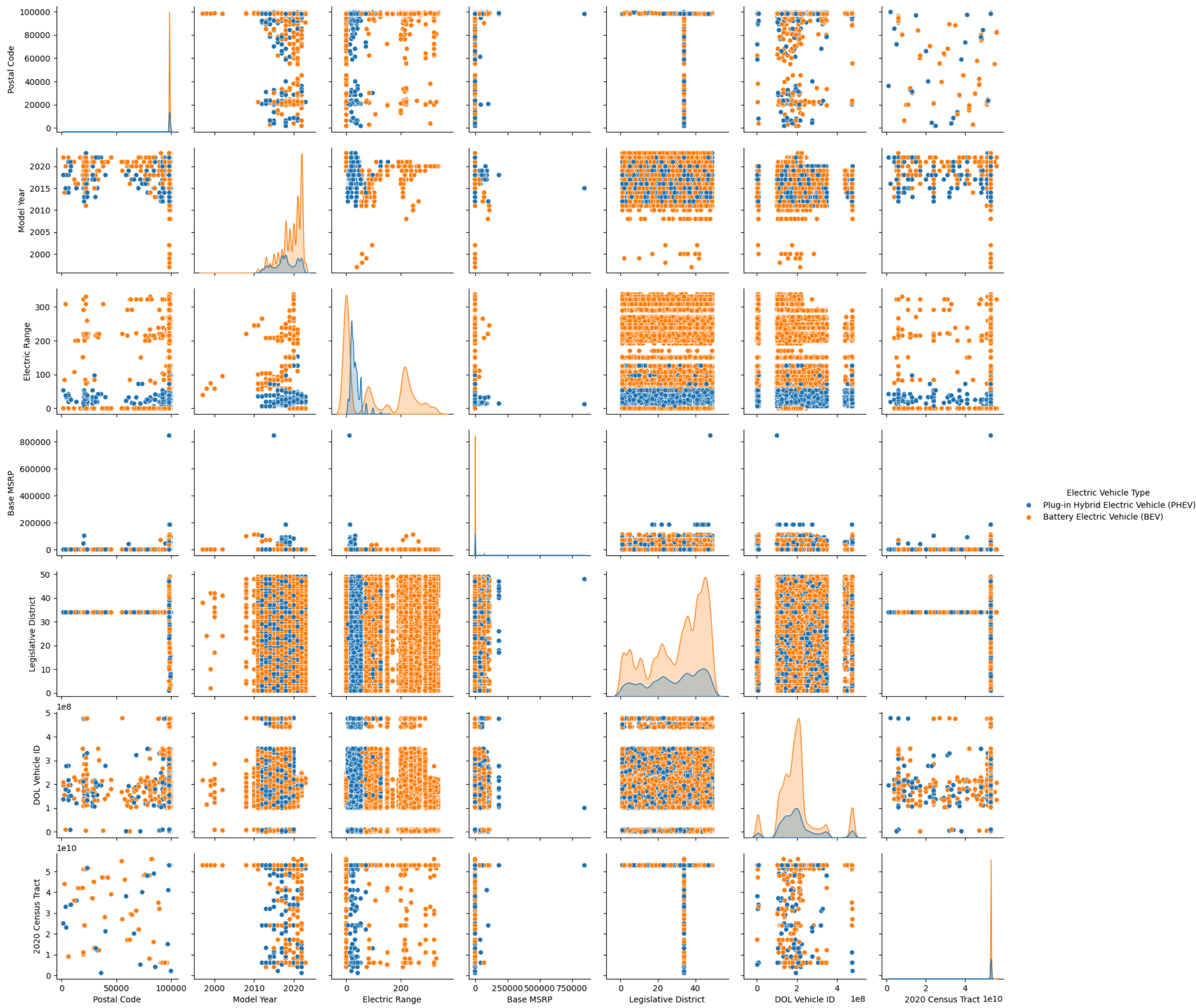
How does Model Year influence the Electric Range across different Make

In [30]:
```
sns.boxplot(x=df["Model Year"],y=df["Electric Range"],hue=df["Make"])
plt.legend(loc='center left', bbox_to_anchor=(1, 0.5))
plt.xticks(rotation=90)
plt.show()
```



How do various numerical features (e.g., Electric Range, Base MSRP) interact with each other for different Electric Vehicle Type categories (BEV vs. PHEV)?

In [51]:
```
sns.pairplot(df, hue='Electric Vehicle Type', diag_kind='kde')
plt.show()
```



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

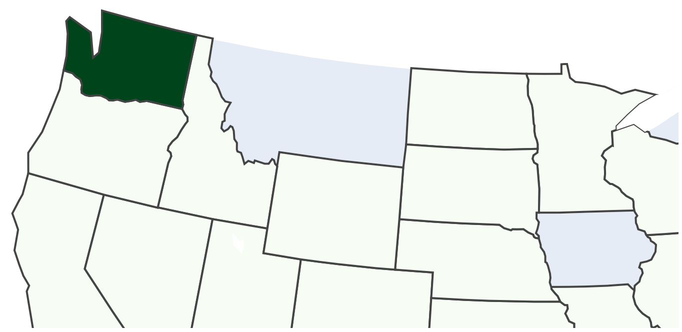
In [7]:

```
import pandas as pd
import plotly.express as px

# Check if 'State' column has valid state codes
print(state_data['State'].unique())

# If the state data is valid, proceed with the plot
fig = px.choropleth(
    state_data,
    locations='State',
    locationmode='USA-states',
    color='EV Count',
    color_continuous_scale='greens',
    scope='usa',
    labels={'EV Count': 'Number of EV Vehicles'},
    title='Number of Electric Vehicles by State'
)
```

```python
# Show the plot
fig.show()
```

```
['AK' 'AL' 'AR' 'AZ' 'CA' 'CO' 'CT' 'DC' 'DE' 'FL' 'GA' 'HI' 'ID' 'IL'
 'KS' 'KY' 'LA' 'MA' 'MD' 'ME' 'MN' 'MO' 'MS' 'NC' 'ND' 'NE' 'NH' 'NJ'
 'NM' 'NV' 'NY' 'OH' 'OK' 'OR' 'PA' 'RI' 'SC' 'SD' 'TN' 'TX' 'UT' 'VA'
 'WA' 'WI' 'WY']
```

Number of Electric Vehicles by State



In [ ]:

```python

```

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

In [107…

```python
df.columns
```

Out[107…

```
Index(['State', 'VIN (1-10)'], dtype='object')
```

In [109…

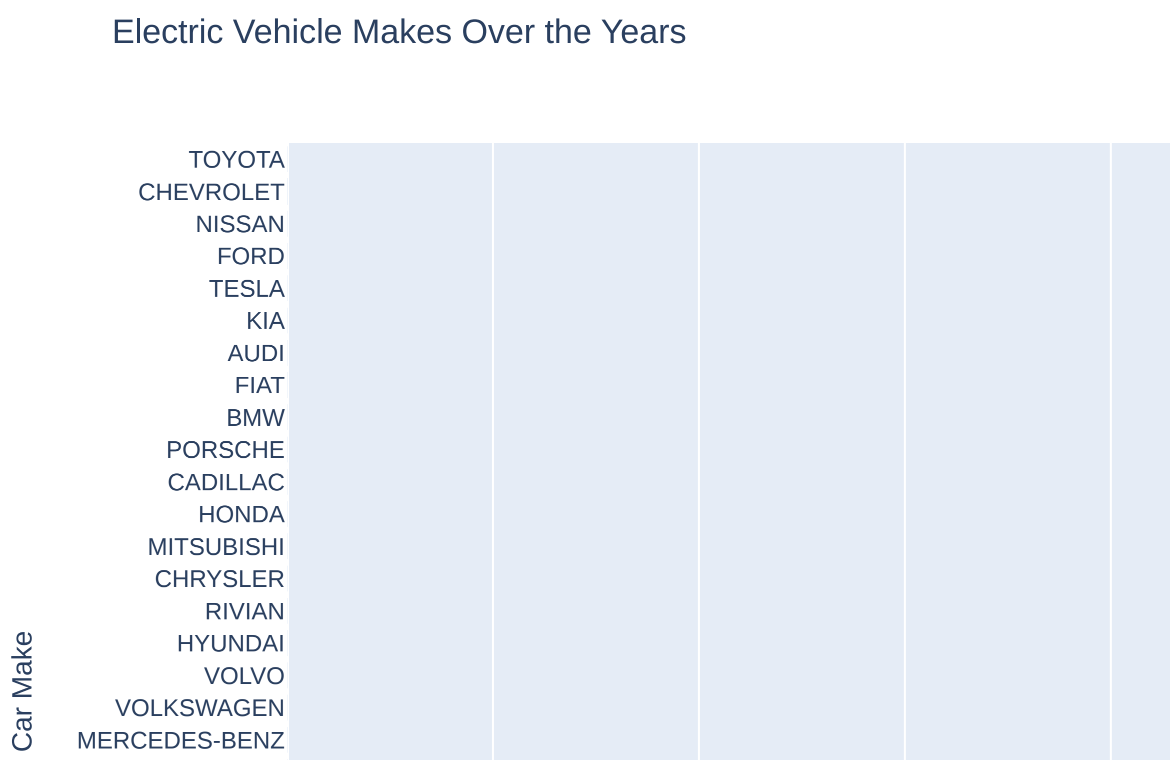
```python
df
```

Out[109…

	State	VIN (1-10)
0	CA	15000
1	TX	7000
2	NY	6000
3	FL	8000
4	IL	5000

In [4]:

```
import pandas as pd
import plotly.express as px

# Load the dataset into a DataFrame
df = pd.read_csv('C:/Users/DELL/OneDrive/Desktop/Innomatics/dataset.csv')

# Step 1: Group data by 'Model Year' and 'Make' to get EV count
ev_make_by_year = df.groupby(['Model Year', 'Make']).size().reset_index(name

# Step 2: Create a list of all unique makes
unique_makes = df['Make'].unique()

# Step 3: Ensure all makes appear in every year by filling missing combinati
all_years = pd.DataFrame({'Model Year': sorted(df['Model Year'].unique())})
all_combinations = all_years.assign(key=1).merge(pd.DataFrame({'Make': uniqu
ev_make_by_year_full = all_combinations.merge(ev_make_by_year, on=['Model Ye

# Step 4: Convert EV Count to integer (since it was NaN before)
ev_make_by_year_full['EV Count'] = ev_make_by_year_full['EV Count'].astype(i

# Step 5: Create the animated racing bar plot with increased height
fig = px.bar(
    ev_make_by_year_full,  # Data
    x='EV Count',  # X-axis shows the count of EVs
    y='Make',  # Y-axis shows the car Make
    color='Make',  # Color by car Make
    animation_frame='Model Year',  # Animation by year
    orientation='h',  # Horizontal bar chart
    title='Electric Vehicle Makes Over the Years',
    labels={'EV Count':'Number of EVs', 'Make':'Car Make'},  # Axis labels
    range_x=[0, ev_make_by_year_full['EV Count'].max() * 1.1],  # Dynamicall
    height=800  # Increased height for better visibility
)

# Step 6: Show the plot
fig.show()
```


Electric Vehicle Makes Over the Years
Car Make
TOYOTA
CHEVROLET
NISSAN
FORD
TESLA
KIA
AUDI
FIAT
BMW
PORSCHE
CADILLAC
HONDA
MITSUBISHI
CHRYSLER
RIVIAN
HYUNDAI
VOLVO
VOLKSWAGEN
MERCEDES-BENZ

In [ ]: