In [1]:

```python
import numpy as np
import pandas as pd
import os
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

In [2]:

```python
df = pd.read_csv("EV_CARS _INDIA.csv")
```

In [3]:

```python
df.head()
```

Out[3]:

	Brand Name	Battery Capacity(kWh)	Acceleration(sec)	TopSpeed(km/h)	Range(km)	Max Power(kW)	Max Torque(Nm)	Transmission	No. of Seats	Charging T(h)	No. of Airbags	Drive Type	Price(Lh
0	Audi RS e-tron GT	93.4	3.3	250	480	500	830	Automatic	5	9	Yes	AWD	204
1	Audi e-tron GT	93.4	4.1	245	500	523	630	Automatic	5	9	Yes	AWD	179
2	Audi e-tron	95.0	5.7	200	484	300	664	Automatic	5	9	Yes	AWD	123
3	Tata Nexon EV	30.2	9.9	180	312	96	245	Automatic	5	9	Yes	FWD	17
4	Tata Tigor EV	26.0	5.7	120	306	55	170	Automatic	5	9	Yes	FWD	14

In [4]:

```python
df.describe()
```

Out[4]:

	Battery Capacity(kWh)	Acceleration(sec)	TopSpeed(km/h)	Range(km)	Max Power(kW)	Max Torque(Nm)	No. of Seats	Charging T(h)	Price(Lh)
count	11.000000	11.000000	11.000000	11.000000	11.000000	11.000000	11.000000	11.000000	11.000000
mean	56.634545	6.909091	165.090909	363.454545	212.181818	445.818182	4.909091	10.363636	74.272727
std	33.683686	2.653848	58.430223	139.407578	182.454826	280.207001	0.301511	3.931227	72.468049
min	10.080000	3.300000	80.000000	100.000000	19.000000	70.000000	4.000000	7.000000	9.000000
25%	28.100000	4.950000	120.000000	309.000000	75.500000	207.500000	5.000000	8.500000	15.500000
50%	44.500000	5.700000	180.000000	414.000000	107.000000	395.000000	5.000000	9.000000	25.000000
75%	91.700000	9.100000	200.000000	475.000000	302.000000	680.000000	5.000000	10.500000	117.500000
max	95.000000	11.200000	250.000000	500.000000	523.000000	830.000000	5.000000	21.000000	204.000000

In [5]:

```python
#check for null values
df.isnull().sum()
```

Out[5]:

```
Brand Name               0
Battery Capacity(kWh)    0
Acceleration(sec)        0
TopSpeed(km/h)           0
Range(km)                0
Max Power(kW)            0
Max Torque(Nm)           0
Transmission             0
No. of Seats             0
Charging T(h)            0
No. of Airbags           0
Drive Type               0
Price(Lh)                0
dtype: int64
```

In [6]:

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 11 entries, 0 to 10
Data columns (total 13 columns):
 #   Column                 Non-Null Count  Dtype  
---  ------                 --------------  -----  
 0   Brand Name             11 non-null     object 
 1   Battery Capacity(kWh)  11 non-null     float64
 2   Acceleration(sec)      11 non-null     float64
 3   TopSpeed(km/h)         11 non-null     int64  
 4   Range(km)              11 non-null     int64  
 5   Max Power(kW)          11 non-null     int64  
 6   Max Torque(Nm)         11 non-null     int64  
 7   Transmission           11 non-null     object 
 8   No. of Seats           11 non-null     int64  
 9   Charging T(h)          11 non-null     int64  
 10  No. of Airbags         11 non-null     object 
 11  Drive Type             11 non-null     object 
 12  Price(Lh)              11 non-null     int64  
dtypes: float64(2), int64(7), object(4)
memory usage: 1.2+ KB
```

In [7]:

```
df.columns
```

Out[7]:

```
Index(['Brand Name', 'Battery Capacity(kWh)', 'Acceleration(sec)',
       'TopSpeed(km/h)', 'Range(km)', 'Max Power(kW)', 'Max Torque(Nm)',
       'Transmission', 'No. of Seats', 'Charging T(h)', 'No. of Airbags',
       'Drive Type', 'Price(Lh)'],
      dtype='object')
```

In [8]:

```
#check for any hidden special characters
df["Brand Name"].unique()
```

Out[8]:

```
array(['Audi RS e-tron GT ', 'Audi e-tron GT ', 'Audi e-tron ',
       'Tata Nexon EV', 'Tata Tigor EV', 'Hyudai Kona Electric',
       'Jaguar I-Pace', 'Mahindra eVerito', 'MG ZS EV',
       'Mercedes Benz EQC', 'Mahindra e2op4/p6'], dtype=object)
```

In [9]:

```
df["Battery Capacity(kWh)"].unique()
```

Out[9]:

```
array([93.4 , 95.  , 30.2 , 26.  , 39.2 , 90.  , 21.2 , 44.5 , 80.  ,
       10.08])
```

In [10]:

```
df["Acceleration(sec)"].unique()
```

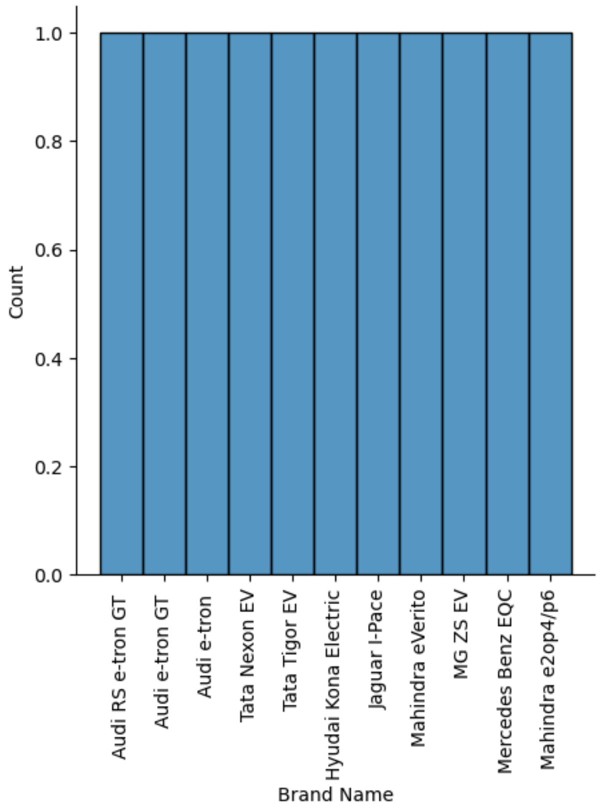
Out[10]:

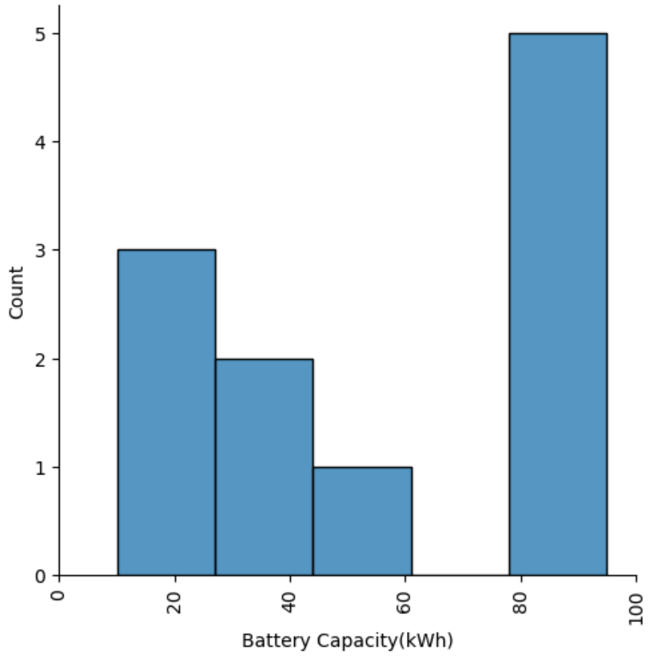
```
array([ 3.3,  4.1,  5.7,  9.9,  9.7,  4.8, 11.2,  8.5,  5.1,  8. ])
```

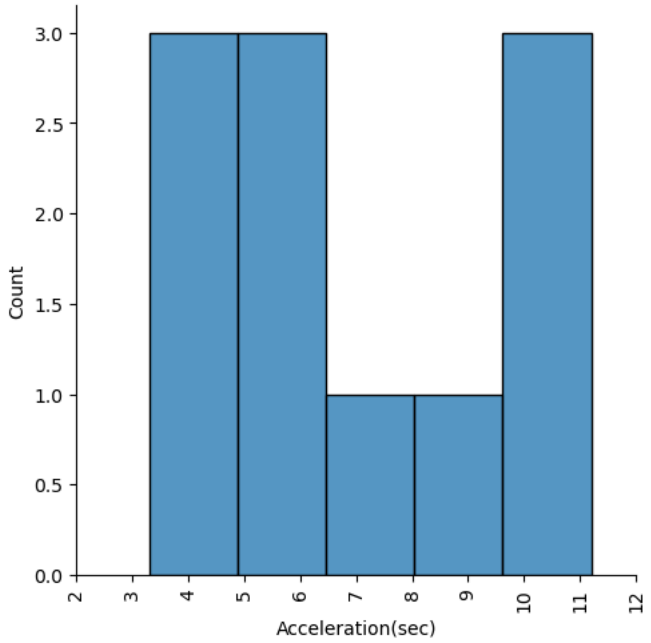
In [11]:

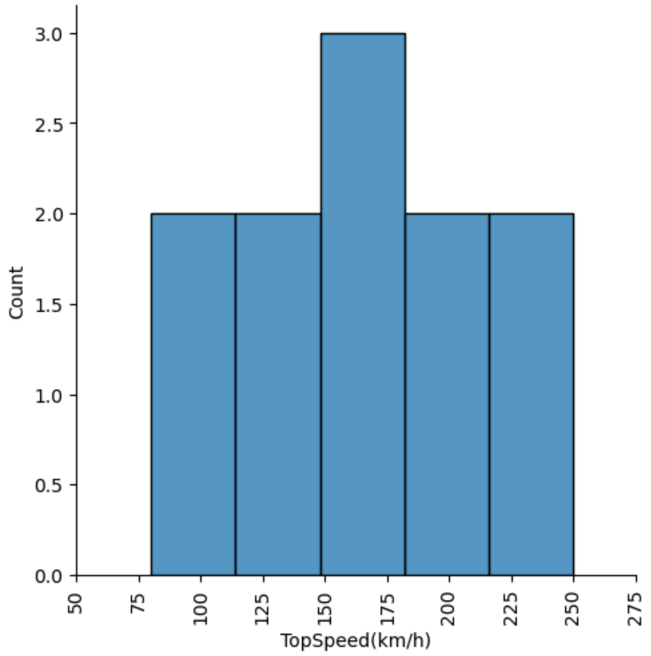
```python
#now data visualization
for col in df.columns:
    ax= sns.displot(df[col])
    ax.set_xticklabels(rotation=90)
```

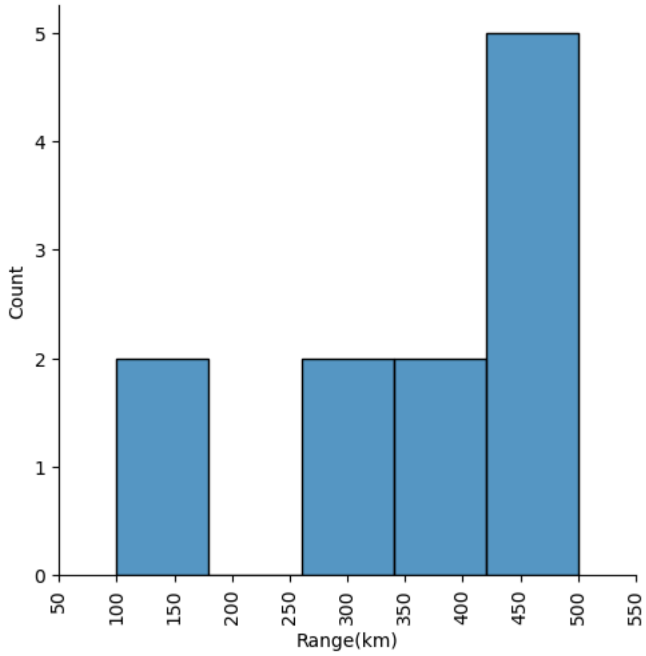
```
C:\ProgramData\anaconda3\Lib\site-packages\seaborn\axisgrid.py:118: UserWarning: The figure layout has changed to tight
  self._figure.tight_layout(*args, **kwargs)
C:\ProgramData\anaconda3\Lib\site-packages\seaborn\axisgrid.py:118: UserWarning: The figure layout has changed to tight
  self._figure.tight_layout(*args, **kwargs)
C:\ProgramData\anaconda3\Lib\site-packages\seaborn\axisgrid.py:118: UserWarning: The figure layout has changed to tight
  self._figure.tight_layout(*args, **kwargs)
C:\ProgramData\anaconda3\Lib\site-packages\seaborn\axisgrid.py:118: UserWarning: The figure layout has changed to tight
  self._figure.tight_layout(*args, **kwargs)
C:\ProgramData\anaconda3\Lib\site-packages\seaborn\axisgrid.py:118: UserWarning: The figure layout has changed to tight
  self._figure.tight_layout(*args, **kwargs)
C:\ProgramData\anaconda3\Lib\site-packages\seaborn\axisgrid.py:118: UserWarning: The figure layout has changed to tight
  self._figure.tight_layout(*args, **kwargs)
C:\ProgramData\anaconda3\Lib\site-packages\seaborn\axisgrid.py:118: UserWarning: The figure layout has changed to tight
  self._figure.tight_layout(*args, **kwargs)
C:\ProgramData\anaconda3\Lib\site-packages\seaborn\axisgrid.py:118: UserWarning: The figure layout has changed to tight
  self._figure.tight_layout(*args, **kwargs)
C:\ProgramData\anaconda3\Lib\site-packages\seaborn\axisgrid.py:118: UserWarning: The figure layout has changed to tight
  self._figure.tight_layout(*args, **kwargs)
C:\ProgramData\anaconda3\Lib\site-packages\seaborn\axisgrid.py:118: UserWarning: The figure layout has changed to tight
  self._figure.tight_layout(*args, **kwargs)
C:\ProgramData\anaconda3\Lib\site-packages\seaborn\axisgrid.py:118: UserWarning: The figure layout has changed to tight
  self._figure.tight_layout(*args, **kwargs)
C:\ProgramData\anaconda3\Lib\site-packages\seaborn\axisgrid.py:118: UserWarning: The figure layout has changed to tight
  self._figure.tight_layout(*args, **kwargs)
C:\ProgramData\anaconda3\Lib\site-packages\seaborn\axisgrid.py:118: UserWarning: The figure layout has changed to tight
  self._figure.tight_layout(*args, **kwargs)
```

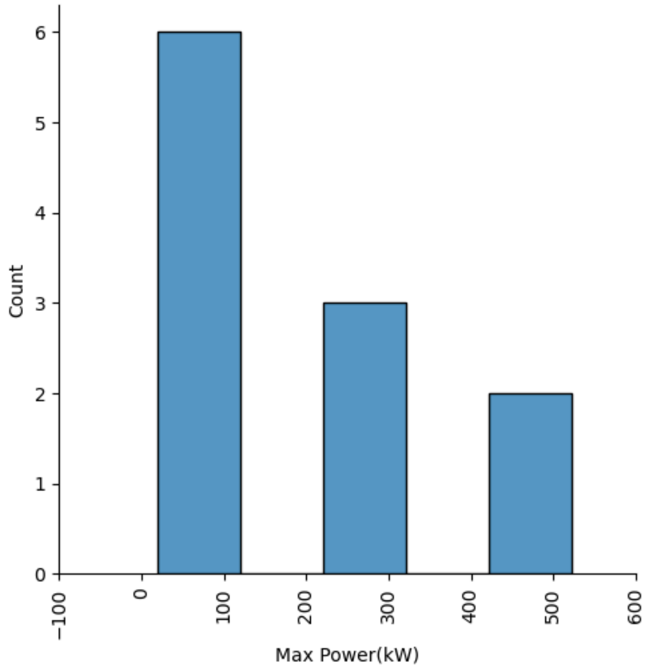


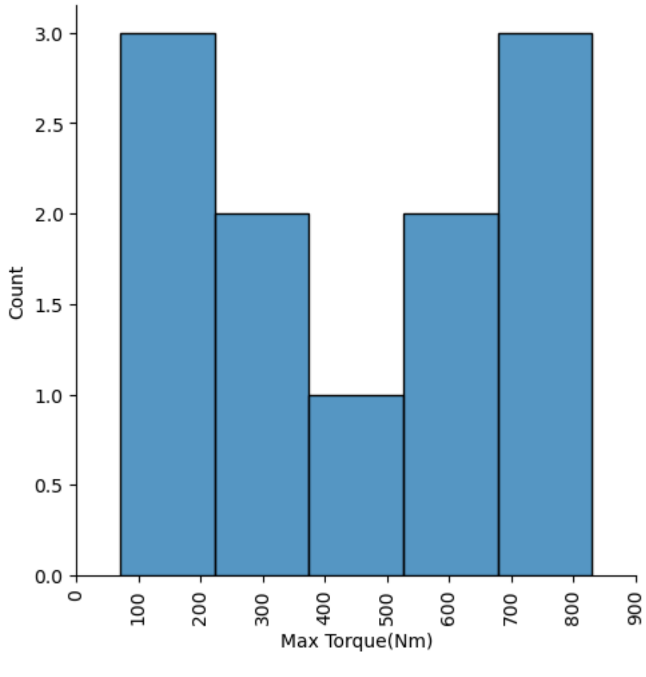

Count
0
1
2
3
4
5
0
20
40
60
80
100
Battery Capacity(kWh)

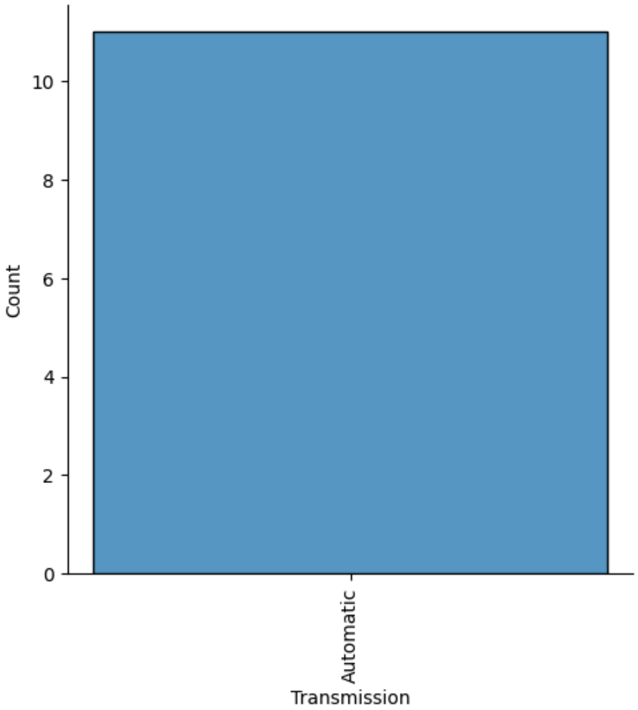

Count
0.0
0.5
1.0
1.5
2.0
2.5
3.0
2
3
4
5
6
7
8
9
10
11
12
Acceleration(sec)

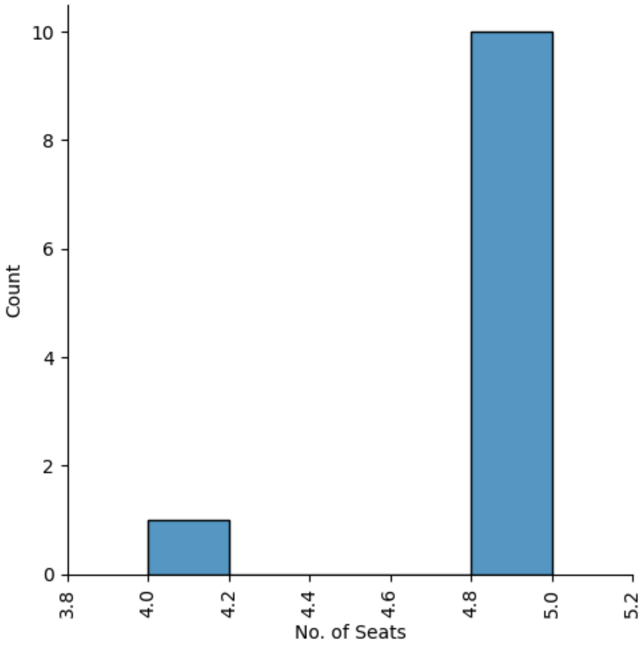

Count
0.0
0.5
1.0
1.5
2.0
2.5
3.0
50
75
100
125
150
175
200
225
250
275
TopSpeed(km/h)

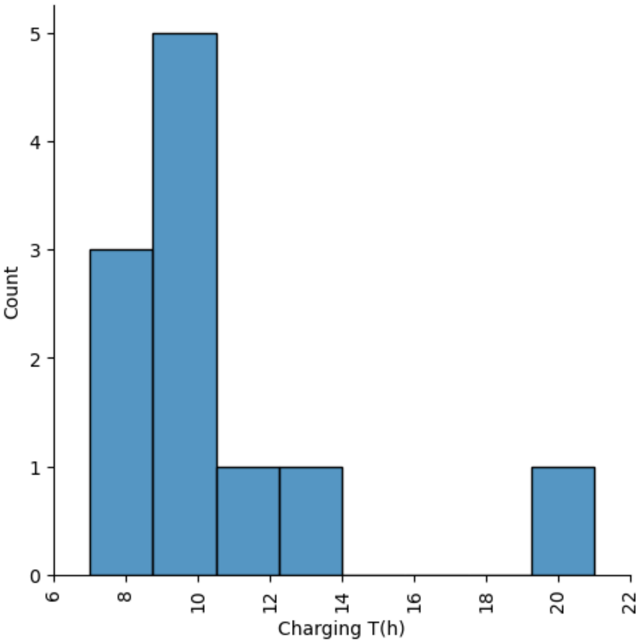

Count
0
1
2
3
4
5
50
100
150
200
250
300
350
400
450
500
550
Range(km)

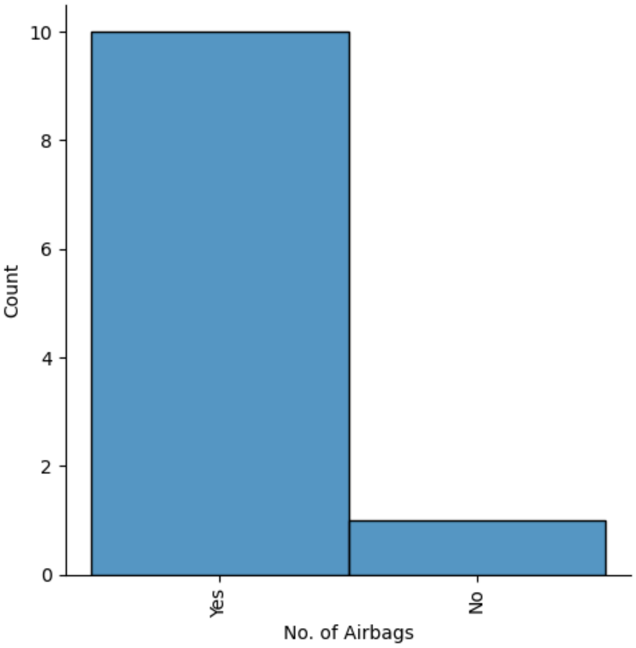

Count
0
1
2
3
4
5
6
−100
0
100
200
300
400
500
600
Max Power(kW)

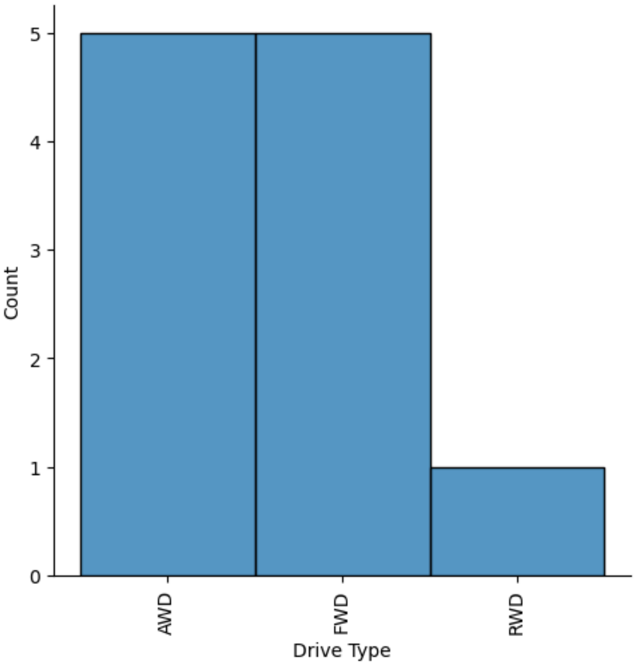

Count
0.0
0.5
1.0
1.5
2.0
2.5
3.0
0
100
200
300
400
500
600
700
800
900
Max Torque(Nm)

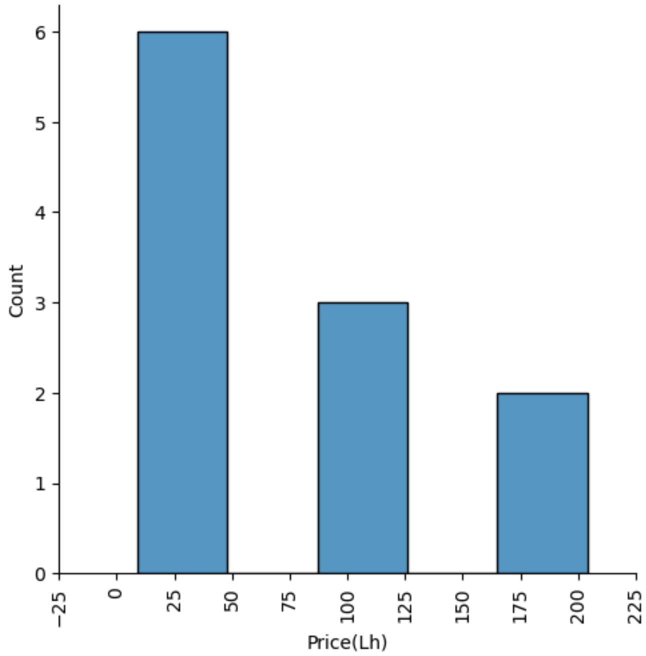

Count
0
2
4
6
8
10
Automatic
Transmission


Count
0
2
4
6
8
10
3.8
4.0
4.2
4.4
4.6
4.8
5.0
5.2
No. of Seats


Count
0
1
2
3
4
5
6
8
10
12
14
16
18
20
22
Charging T(h)

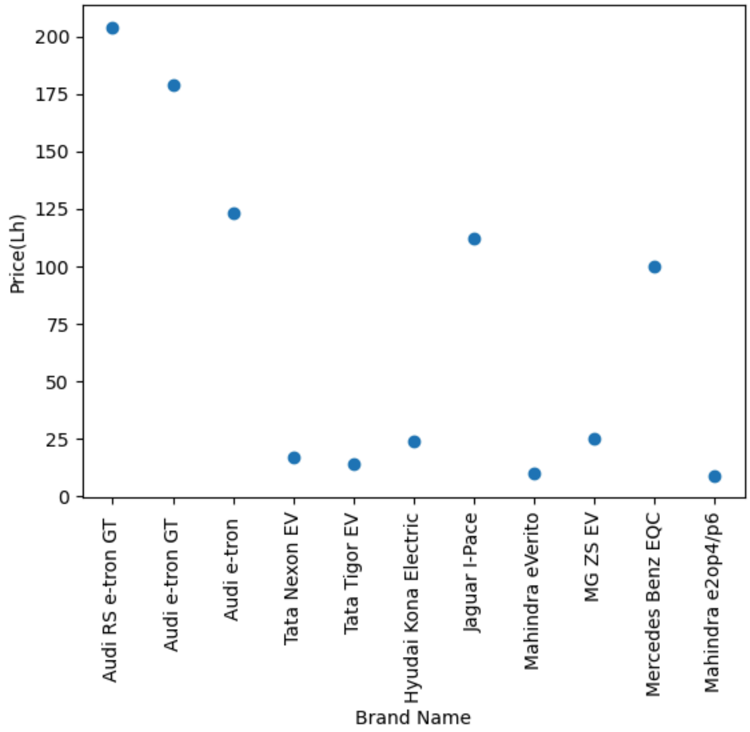

Count
0
2
4
6
8
10
Yes
No
No. of Airbags


Count
0
1
2
3
4
5
AWD
FWD
RWD
Drive Type



In [12]:
```
plt.xlabel('Brand Name')
plt.ylabel('Price(Lh)')
plt.scatter(df['Brand Name'],df['Price(Lh)'])
plt.xticks(rotation=90)
```

Out[12]:
```
([0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10],
 [Text(0, 0, 'Audi RS e-tron GT '),
  Text(1, 0, 'Audi e-tron GT '),
  Text(2, 0, 'Audi e-tron '),
  Text(3, 0, 'Tata Nexon EV'),
  Text(4, 0, 'Tata Tigor EV'),
  Text(5, 0, 'Hyudai Kona Electric'),
  Text(6, 0, 'Jaguar I-Pace'),
  Text(7, 0, 'Mahindra eVerito'),
  Text(8, 0, 'MG ZS EV'),
  Text(9, 0, 'Mercedes Benz EQC'),
  Text(10, 0, 'Mahindra e2op4/p6')])
```

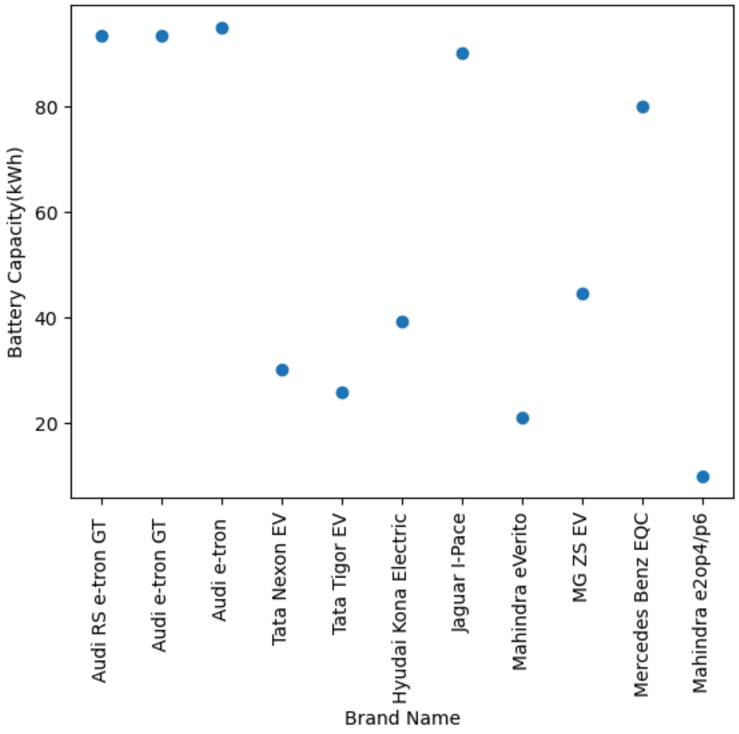


In [13]:

```
plt.xlabel('Brand Name')
plt.ylabel('Battery Capacity(kWh)')
plt.scatter(df['Brand Name'],df['Battery Capacity(kWh)'])
plt.xticks(rotation=90)
```

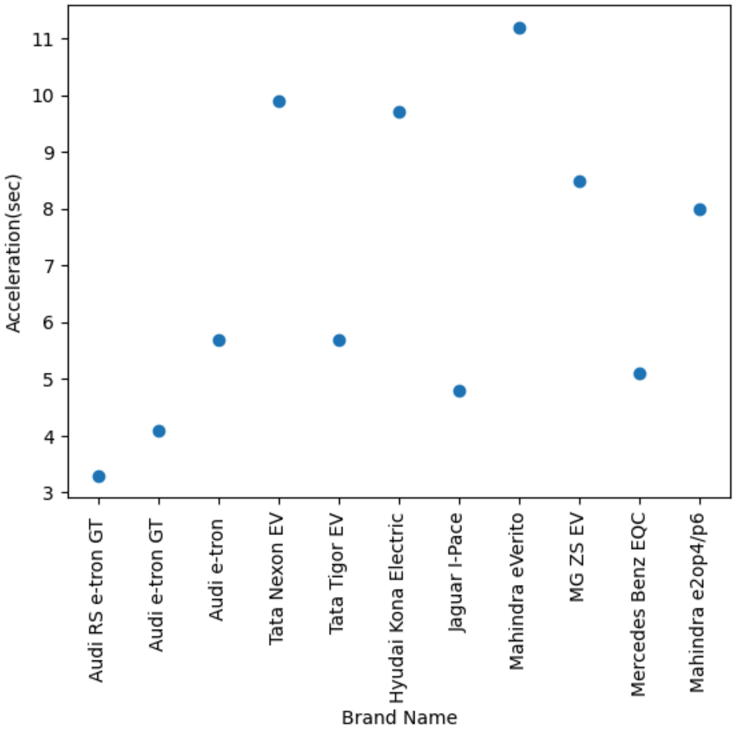
Out[13]:

```
([0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10],
 [Text(0, 0, 'Audi RS e-tron GT '),
  Text(1, 0, 'Audi e-tron GT '),
  Text(2, 0, 'Audi e-tron '),
  Text(3, 0, 'Tata Nexon EV'),
  Text(4, 0, 'Tata Tigor EV'),
  Text(5, 0, 'Hyudai Kona Electric'),
  Text(6, 0, 'Jaguar I-Pace'),
  Text(7, 0, 'Mahindra eVerito'),
  Text(8, 0, 'MG ZS EV'),
  Text(9, 0, 'Mercedes Benz EQC'),
  Text(10, 0, 'Mahindra e2op4/p6')])
```



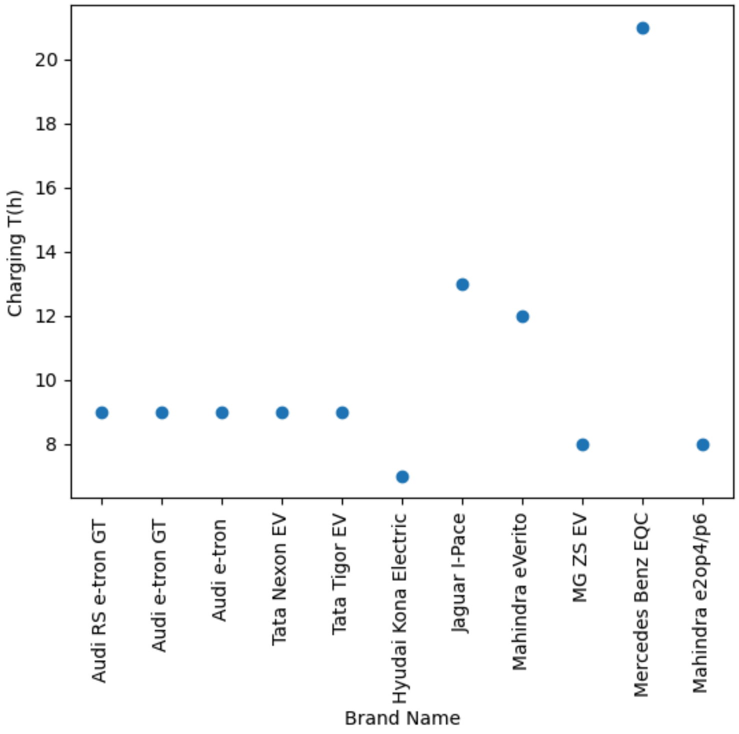
In [14]:
```
plt.xlabel('Brand Name')
plt.ylabel('Acceleration(sec)')
plt.scatter(df['Brand Name'],df['Acceleration(sec)'])
plt.xticks(rotation=90)
```

Out[14]:
```
([0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10],
 [Text(0, 0, 'Audi RS e-tron GT '),
  Text(1, 0, 'Audi e-tron GT '),
  Text(2, 0, 'Audi e-tron '),
  Text(3, 0, 'Tata Nexon EV'),
  Text(4, 0, 'Tata Tigor EV'),
  Text(5, 0, 'Hyudai Kona Electric'),
  Text(6, 0, 'Jaguar I-Pace'),
  Text(7, 0, 'Mahindra eVerito'),
  Text(8, 0, 'MG ZS EV'),
  Text(9, 0, 'Mercedes Benz EQC'),
  Text(10, 0, 'Mahindra e2op4/p6')])
```



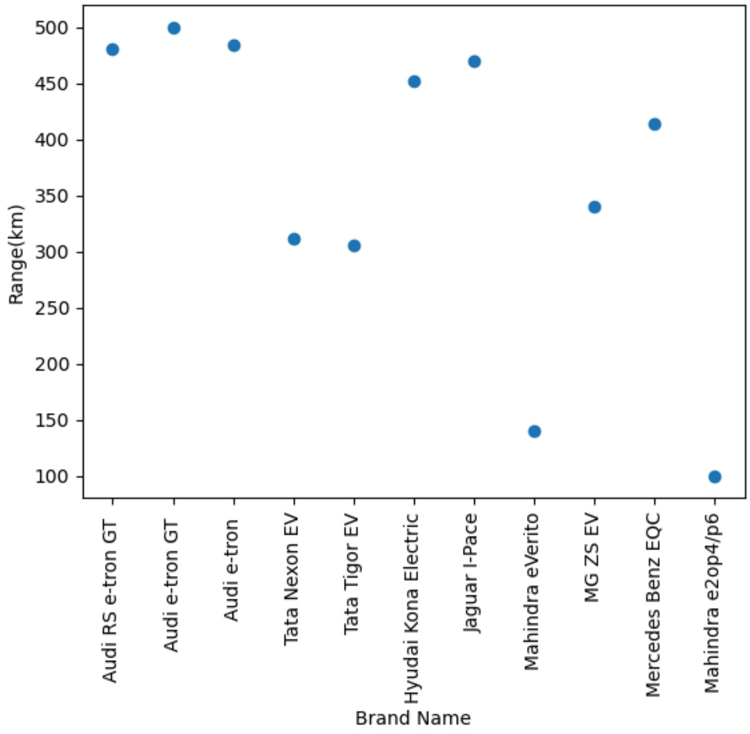
In [15]:
```
plt.xlabel('Brand Name')
plt.ylabel('Charging T(h)')
plt.scatter(df['Brand Name'],df['Charging T(h)'])
plt.xticks(rotation=90)
```

Out[15]:
```
([0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10],
 [Text(0, 0, 'Audi RS e-tron GT '),
  Text(1, 0, 'Audi e-tron GT '),
  Text(2, 0, 'Audi e-tron '),
  Text(3, 0, 'Tata Nexon EV'),
  Text(4, 0, 'Tata Tigor EV'),
  Text(5, 0, 'Hyudai Kona Electric'),
  Text(6, 0, 'Jaguar I-Pace'),
  Text(7, 0, 'Mahindra eVerito'),
  Text(8, 0, 'MG ZS EV'),
  Text(9, 0, 'Mercedes Benz EQC'),
  Text(10, 0, 'Mahindra e2op4/p6')])
```



In [16]:
```
plt.xlabel('Brand Name')
plt.ylabel('Range(km)')
plt.scatter(df['Brand Name'],df['Range(km)'])
plt.xticks(rotation=90)
```

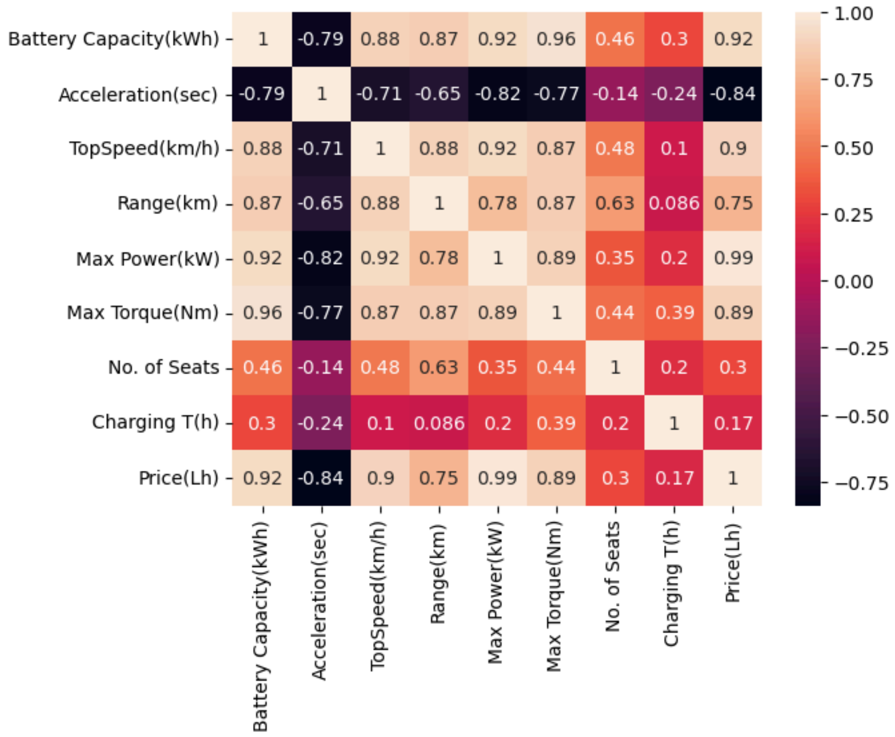
Out[16]:
```
([0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10],
 [Text(0, 0, 'Audi RS e-tron GT '),
  Text(1, 0, 'Audi e-tron GT '),
  Text(2, 0, 'Audi e-tron '),
  Text(3, 0, 'Tata Nexon EV'),
  Text(4, 0, 'Tata Tigor EV'),
  Text(5, 0, 'Hyudai Kona Electric'),
  Text(6, 0, 'Jaguar I-Pace'),
  Text(7, 0, 'Mahindra eVerito'),
  Text(8, 0, 'MG ZS EV'),
  Text(9, 0, 'Mercedes Benz EQC'),
  Text(10, 0, 'Mahindra e2op4/p6')])
```



In [17]:
```
#Based on the analysis of the plots, it's evident that the Jaguar I-Pace offers an excellent balance between price and f
```

In [18]:
```
import seaborn as sns
import pandas as pd

# Assuming 'df' is your DataFrame
# Select only numerical columns for correlation
numerical_df = df.select_dtypes(include=[float, int])

# Plot the heatmap
sns.heatmap(numerical_df.corr(), annot=True)
```

Out[18]:
```
<Axes: >
```



In [19]:
```
from the analysis. Similarly, the strong correlation between max power and price implies that changes in max power are closel
```

In [20]:
```
df.iloc[6]
```

Out[20]:
```
Brand Name               Jaguar I-Pace
Battery Capacity(kWh)             90.0
Acceleration(sec)                  4.8
TopSpeed(km/h)                     200
Range(km)                          470
Max Power(kW)                      294
Max Torque(Nm)                     696
Transmission                 Automatic
No. of Seats                         5
Charging T(h)                       13
No. of Airbags                     Yes
Drive Type                         AWD
Price(Lh)                          112
Name: 6, dtype: object
```

In [21]:
```
df['Price(Lh)']
```

Out[21]:
```
0     204
1     179
2     123
3      17
4      14
5      24
6     112
7      10
8      25
9     100
10      9
Name: Price(Lh), dtype: int64
```

In [22]:

```
df.iloc[9]
```

Out[22]:

```
Brand Name              Mercedes Benz EQC
Battery Capacity(kWh)                80.0
Acceleration(sec)                     5.1
TopSpeed(km/h)                        180
Range(km)                             414
Max Power(kW)                         304
Max Torque(Nm)                        760
Transmission                    Automatic
No. of Seats                            5
Charging T(h)                          21
No. of Airbags                        Yes
Drive Type                            AWD
Price(Lh)                             100
Name: 9, dtype: object
```

dataset2

In [83]:

```
df1 = pd.read_csv("EV Stats.csv")
df2 = pd.read_csv("ElectricCarData_Norm.csv")
df3 = pd.read_csv("Indian automoble buying behavour study 1.0.csv")
```

In [84]:

```
df1.tail()
```

Out[84]:

	Sl. No	State	Two Wheelers (Category L1 & L2 as per Central Motor Vehicles Rules	Two Wheelers (Category L2 (CMVR))	Two Wheelers (Max power not exceeding 250 Watts)	Three Wheelers (Category L5 slow speed as per CMVR)	Three Wheelers (Category L5 as per CMVR)	Passenger Cars (Category M1 as per CMVR)	Buses	Total in state
26	27	West Bengal	1451	65	10781	3	0	1840	0	14140
27	28	Andaman & Nicobar islands	0	0	0	0	0	82	0	82
28	29	Chandigarh	612	18	896	0	0	974	0	2500
29	30	Dadra and Nagar Haveli	4	0	9	0	0	803	0	816
30	31	Total	27549	14069	112538	389	720	105571	27	260863

In [85]:

```
df2.head()
```

Out[85]:

	Brand	Model	Accel	TopSpeed	Range	Efficiency	FastCharge	RapidCharge	PowerTrain	PlugType	BodyStyle	Segment	Seats	PriceEuro
0	Tesla	Model 3 Long Range Dual Motor	4.6 sec	233 km/h	450 km	161 Wh/km	940 km/h	Rapid charging possible	All Wheel Drive	Type 2 CCS	Sedan	D	5	55480
1	Volkswagen	ID.3 Pure	10.0 sec	160 km/h	270 km	167 Wh/km	250 km/h	Rapid charging possible	Rear Wheel Drive	Type 2 CCS	Hatchback	C	5	30000
2	Polestar	2	4.7 sec	210 km/h	400 km	181 Wh/km	620 km/h	Rapid charging possible	All Wheel Drive	Type 2 CCS	Liftback	D	5	56440
3	BMW	iX3	6.8 sec	180 km/h	360 km	206 Wh/km	560 km/h	Rapid charging possible	Rear Wheel Drive	Type 2 CCS	SUV	D	5	68040
4	Honda	e	9.5 sec	145 km/h	170 km	168 Wh/km	190 km/h	Rapid charging possible	Rear Wheel Drive	Type 2 CCS	Hatchback	B	4	32997

In [86]:

```
df3.head()
```

Out[86]:

	Age	Profession	Marrital Status	Education	No of Dependents	Personal loan	House Loan	Wife Working	Salary	Wife Salary	Total Salary	Make	Price
0	27	Salaried	Single	Post Graduate	0	Yes	No	No	800000	0	800000	i20	800000
1	35	Salaried	Married	Post Graduate	2	Yes	Yes	Yes	1400000	600000	2000000	Ciaz	1000000
2	45	Business	Married	Graduate	4	Yes	Yes	No	1800000	0	1800000	Duster	1200000
3	41	Business	Married	Post Graduate	3	No	No	Yes	1600000	600000	2200000	City	1200000
4	31	Salaried	Married	Post Graduate	2	Yes	No	Yes	1800000	800000	2600000	SUV	1600000

In [87]:

```
len(df1), len(df2), len(df3)
```

Out[87]:

```
(31, 103, 99)
```

In [88]:

```
# Plotting state v/S Total EVvehicles in state
df1["State"].dtype, df1["Total in state"].dtype

# x-axis = state
# y-axis = vehicles
```

Out[88]:

```
(dtype('O'), dtype('int64'))
```

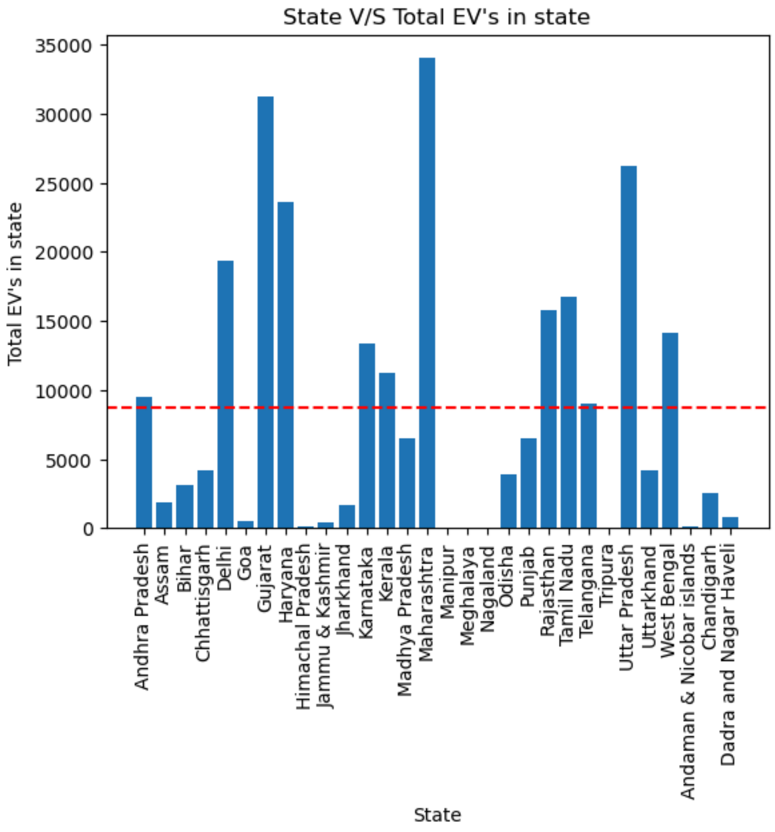
In [89]:

```
# Mean of total Sales from each state
column = df1["Total in state"][:-1]
average = np.mean(column)
print(f"Mean sales of all the States combined is : {average}")
```

```
Mean sales of all the States combined is : 8695.433333333332
```

In [90]:

```
y = df1["Total in state"][:30]
x = df1["State"][:30]

plt.bar(x, y)
plt.xlabel("State")
plt.ylabel("Total EV's in state")
plt.axhline(y.mean(), color='r', linestyle='--')
#plt.axhline(y.median(), color='g', linestyle='--')
plt.title("State V/S Total EV's in state")
plt.xticks(rotation=90)
plt.show()
```



Horizonal red-dotted line represents the mean....so the state having the sales above mean have higher chance of increased sales in the upcoming year as well

In [91]:

```
states_with_greater_sales = df1["State"][:30][df1["Total in state"] > average]
states_with_greater_sales
```

Out[91]:

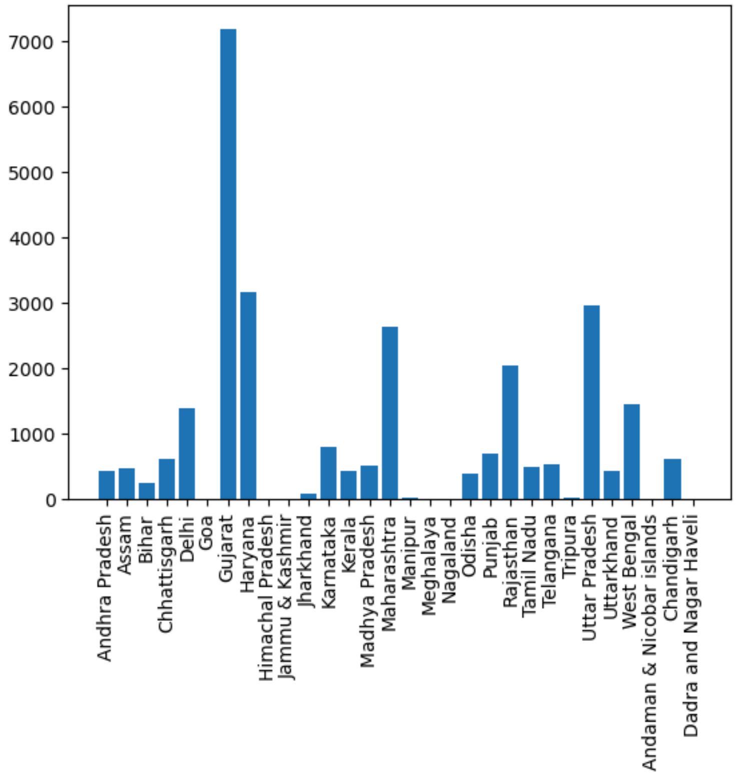
```
0      Andhra Pradesh
4               Delhi
6             Gujarat
7             Haryana
11          Karnataka
12             Kerala
14        Maharashtra
20          Rajasthan
21         Tamil Nadu
22          Telangana
24      Uttar Pradesh
26        West Bengal
Name: State, dtype: object
```

We can see Maharashtra, Gujrat and Uttar Pradesh has the highest registered EV sales and Manipur, Meghalaya and Himachal Pradesh has lowest EV registered

State V/S every ev vehicle category

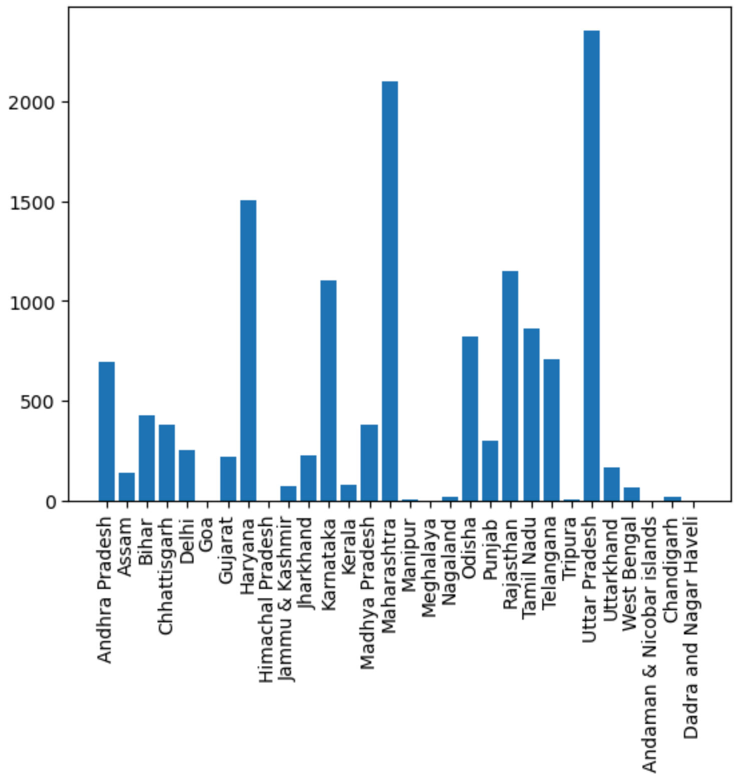
In [92]:

```
# Two Wheelers (Category L1 & L2 as per Central Motor Vehicles Rules

x = df1["State"][:30]
y = df1["Two Wheelers (Category L1 & L2 as per Central Motor Vehicles Rules"][:30]

plt.bar(x,y)
plt.xticks(rotation=90)
plt.show()
```



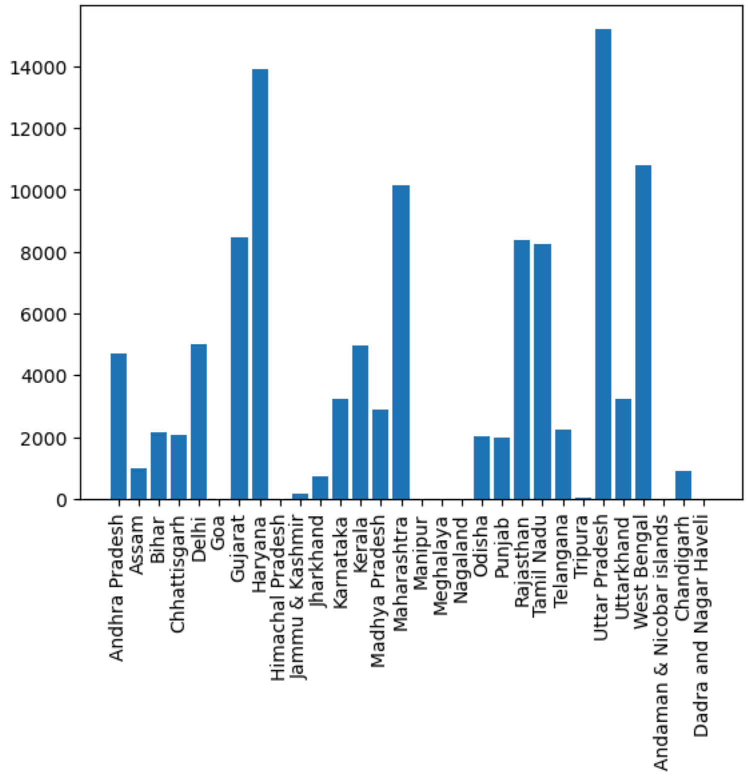
In [93]:

```
# Two Wheelers (Category L2 (CMVR))

x = df1["State"][:30]
y = df1["Two Wheelers (Category L2 (CMVR))"][:30]

plt.bar(x,y)
plt.xticks(rotation=90)
plt.show()
```



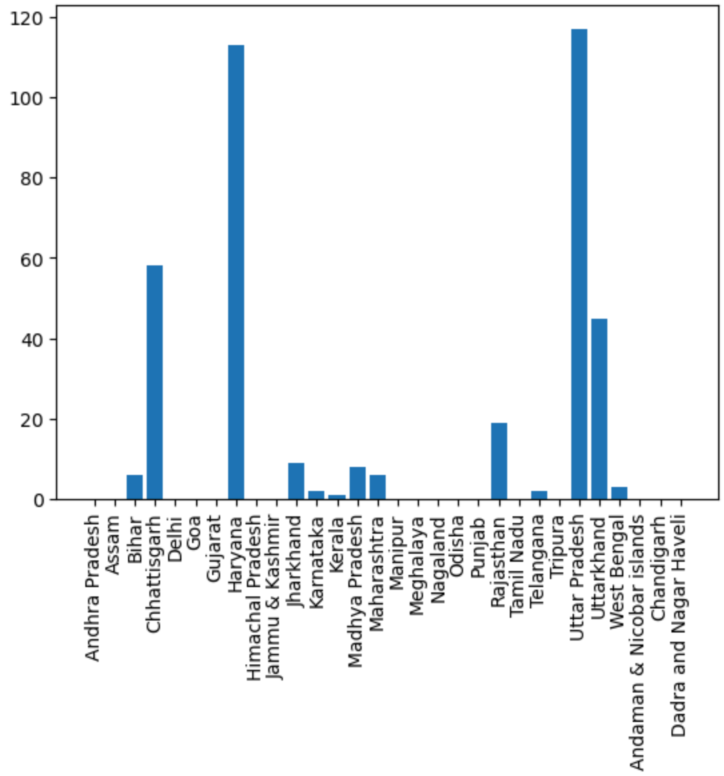
In [94]:

```
# Two Wheelers (Max power not exceeding 250 Watts)

x = df1["State"][:30]
y = df1["Two Wheelers (Max power not exceeding 250 Watts)"][:30]

plt.bar(x,y)
plt.xticks(rotation=90)
plt.show()
```



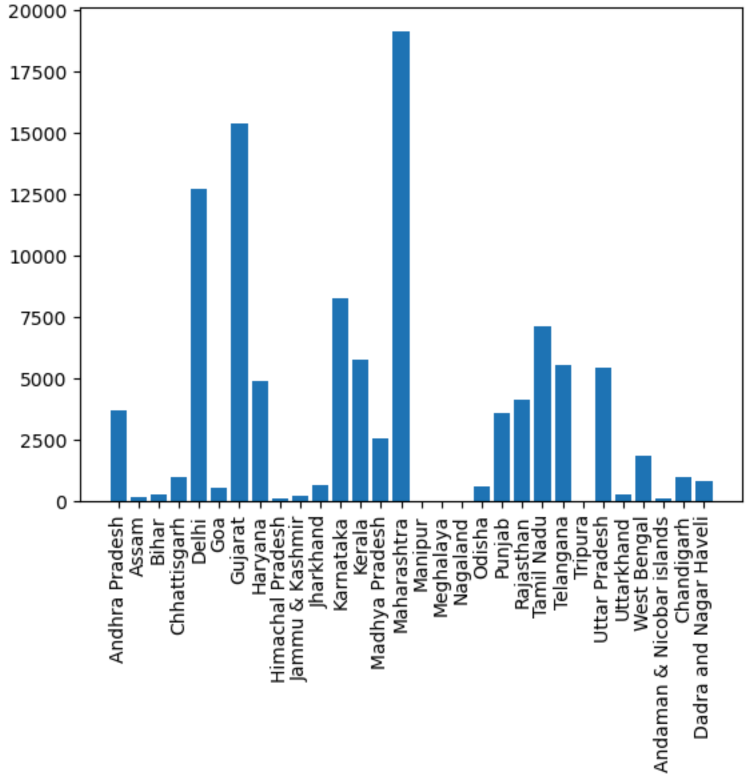
In [95]:

```python
# Three Wheelers (Category L5 slow speed as per CMVR)

x = df1["State"][:30]
y = df1["Three Wheelers (Category L5 slow speed as per CMVR)"][:30]

plt.bar(x,y)
plt.xticks(rotation=90)
plt.show()
```



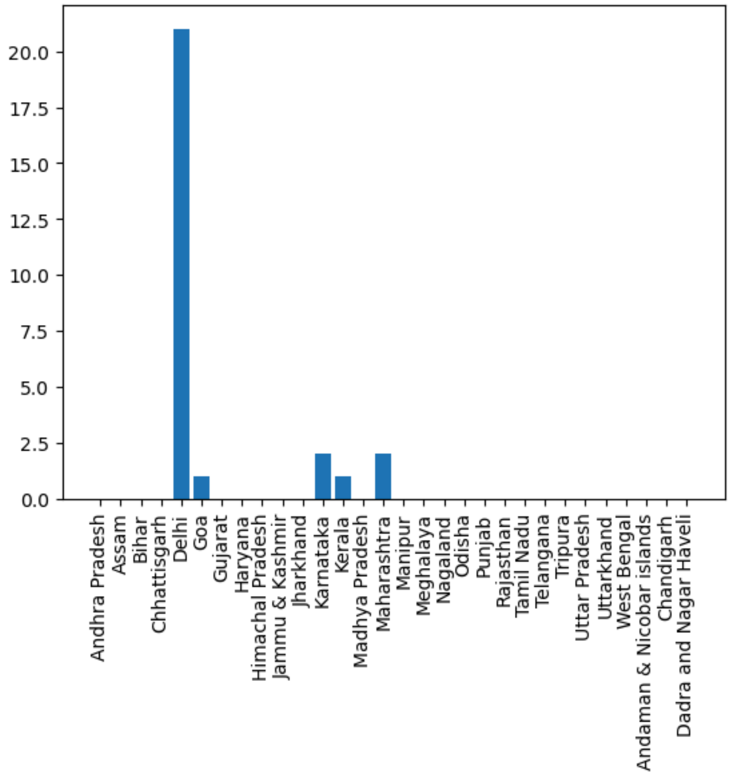
In [96]:

```python
# Passenger Cars (Category M1 as per CMVR)

x = df1["State"][:30]
y = df1["Passenger Cars (Category M1 as per CMVR)"][:30]

plt.bar(x,y)
plt.xticks(rotation=90)
plt.show()
```



In [97]:

```
# Buses

x = df1["State"][:30]
y = df1["Buses"][:30]

plt.bar(x,y)
plt.xticks(rotation=90)
plt.show()
```



In [98]:

```
df3.head(3)
```

Out[98]:

	Age	Profession	Marrital Status	Education	No of Dependents	Personal loan	House Loan	Wife Working	Salary	Wife Salary	Total Salary	Make	Price
0	27	Salaried	Single	Post Graduate	0	Yes	No	No	800000	0	800000	i20	800000
1	35	Salaried	Married	Post Graduate	2	Yes	Yes	Yes	1400000	600000	2000000	Ciaz	1000000
2	45	Business	Married	Graduate	4	Yes	Yes	No	1800000	0	1800000	Duster	1200000

In [99]:

```
# Columns of data
df3.columns
```

Out[99]:

```
Index(['Age', 'Profession', 'Marrital Status', 'Education', 'No of Dependents',
       'Personal loan', 'House Loan', 'Wife Working', 'Salary', 'Wife Salary',
       'Total Salary', 'Make', 'Price'],
      dtype='object')
```

In [100]:

```
# Additional Information about the data
df3.info()
```

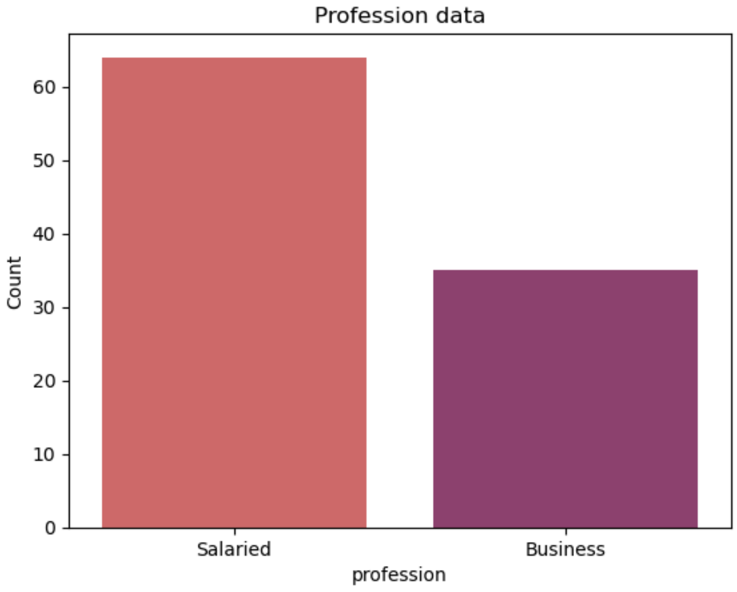
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 99 entries, 0 to 98
Data columns (total 13 columns):
 #   Column            Non-Null Count  Dtype 
---  ------            --------------  ----- 
 0   Age               99 non-null     int64 
 1   Profession        99 non-null     object
 2   Marrital Status   99 non-null     object
 3   Education         99 non-null     object
 4   No of Dependents  99 non-null     int64 
 5   Personal loan     99 non-null     object
 6   House Loan        99 non-null     object
 7   Wife Working      99 non-null     object
 8   Salary            99 non-null     int64 
 9   Wife Salary       99 non-null     int64 
 10  Total Salary      99 non-null     int64 
 11  Make              99 non-null     object
 12  Price             99 non-null     int64 
dtypes: int64(6), object(7)
memory usage: 10.2+ KB
```

In [101]:
```
# Checking Null values
df3.isnull().sum()
```

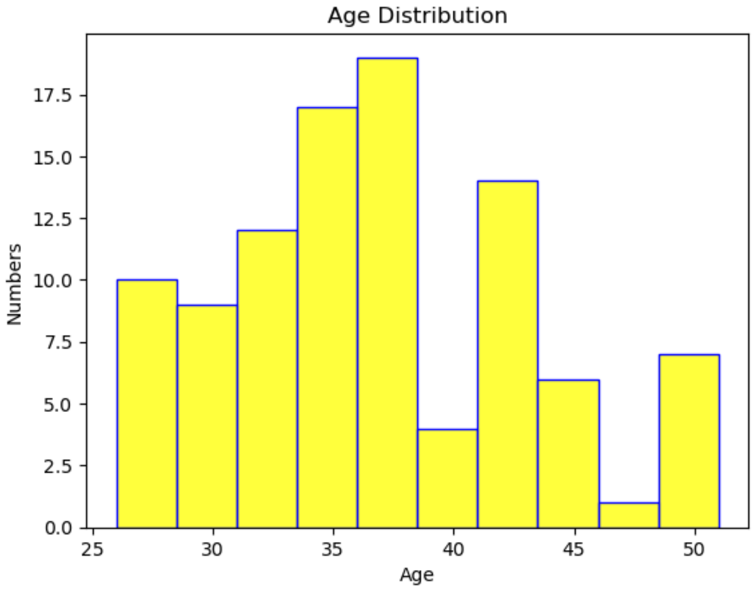
Out[101]:
```
Age                 0
Profession          0
Marrital Status     0
Education           0
No of Dependents    0
Personal loan       0
House Loan          0
Wife Working        0
Salary              0
Wife Salary         0
Total Salary        0
Make                0
Price               0
dtype: int64
```

Some Visualizations regarding the data

In [102]:
```
sns.countplot(data=df3, x="Profession", palette="flare")
plt.title("Profession data")
plt.xlabel("profession")
plt.ylabel("Count")
plt.show()
```

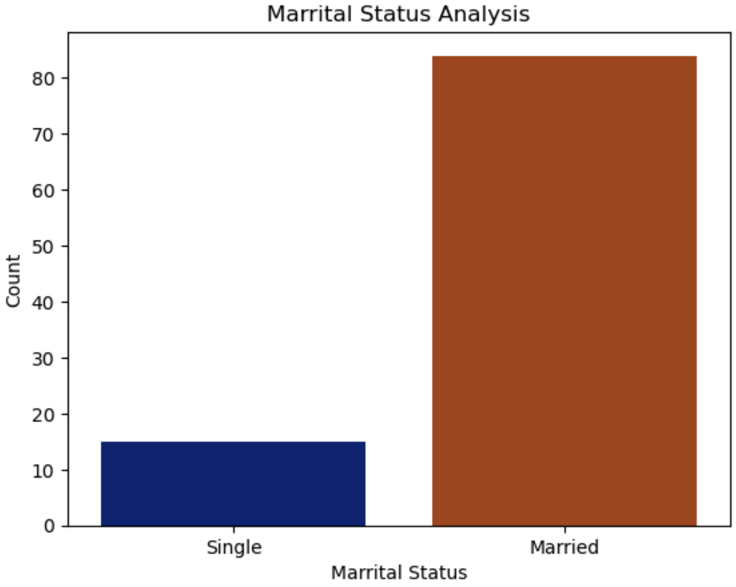


In [103]:
```
# Histogram of Ages
sns.histplot(data=df3, x="Age", bins=10, color="yellow", edgecolor="blue")
plt.title("Age Distribution")
plt.xlabel("Age")
plt.ylabel("Numbers")
plt.show()
```



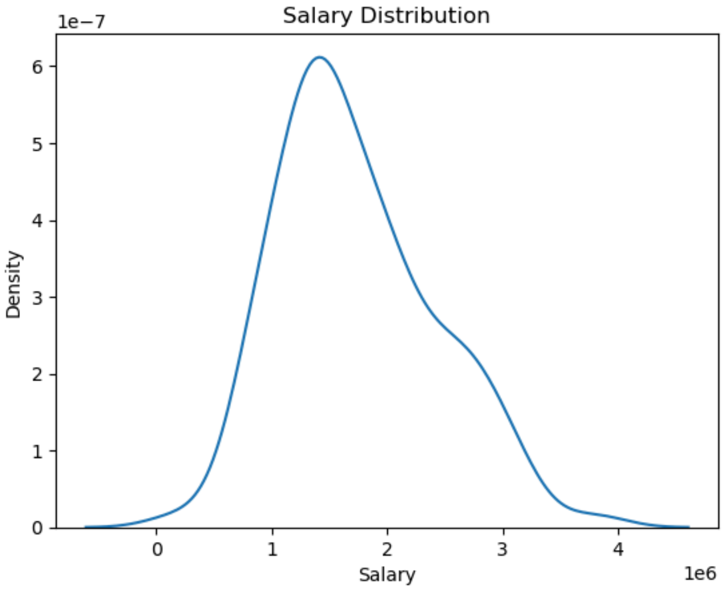
In [104]:

```
# Marrital Status
sns.countplot(data=df3, x="Marrital Status", palette="dark")
plt.title("Marrital Status Analysis")
plt.xlabel("Marrital Status")
plt.ylabel("Count")
plt.show()
```



In [105]:

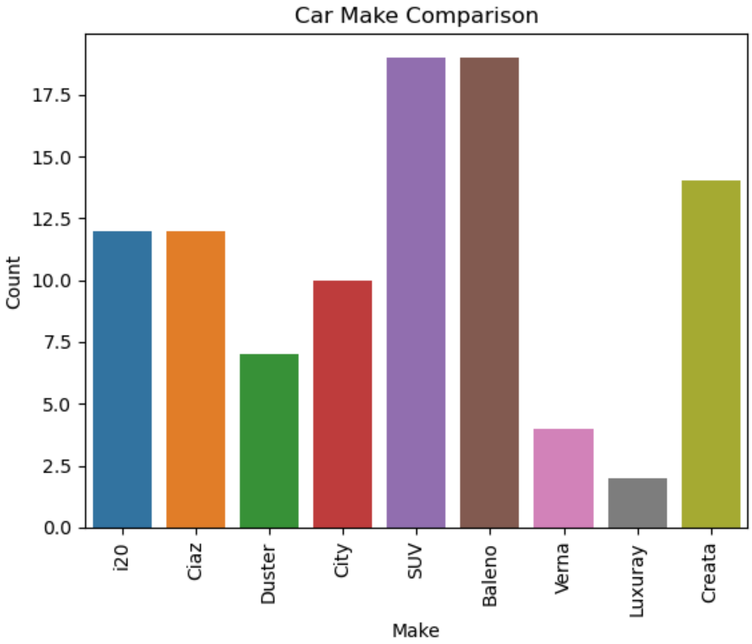
```
# Salary Analysis
sns.kdeplot(df3["Salary"])

plt.title("Salary Distribution")
plt.xlabel("Salary")
plt.ylabel("Density")

plt.show()
```



Car Make Comparison

In [106]:

```
sns.countplot(data=df3,x="Make")
plt.title("Car Make Comparison")
plt.xlabel("Make")
plt.ylabel("Count")
plt.xticks(rotation=90)
plt.show()
```



In [113]:

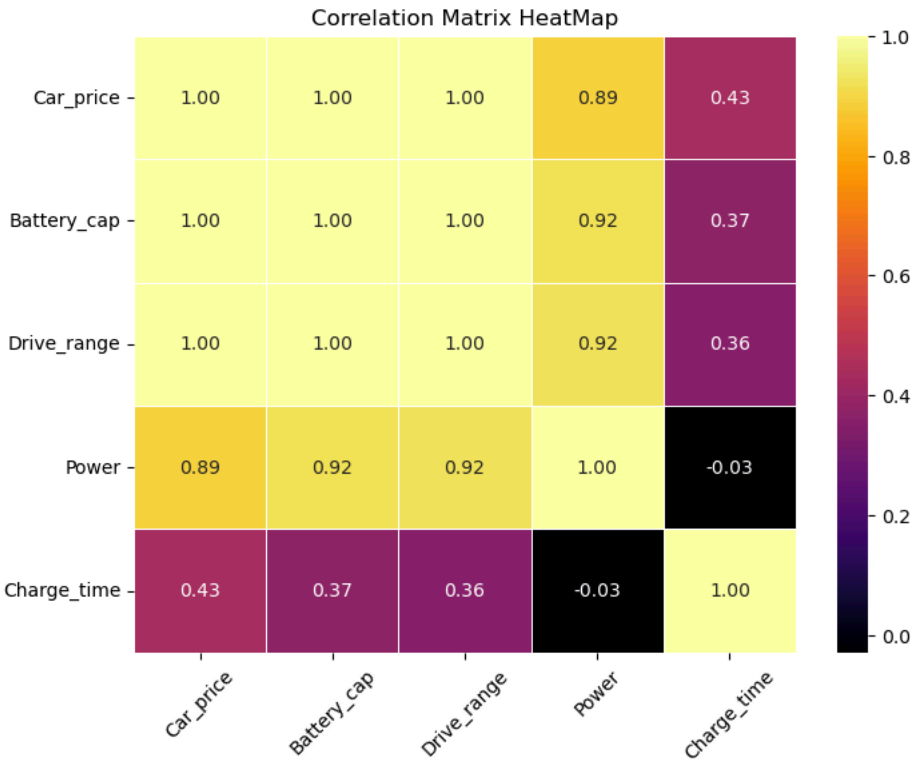
```
df3
```

Out[113]:

	Car_name	Car_price	Battery_cap	Drive_range	Power	Charge_time
0	MG Comet EV	7.98	17.3	230	41.42	7.00000
1	Tata Tiago EV	8.69	19.2	250	60.34	0.96667
2	Tata Tigor EV	12.49	26.0	315	73.75	7.50000

In [121]:

```python
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt

# Example DataFrame
data = {
    'Car_name': ['MG Comet EV', 'Tata Tiago EV', 'Tata Tigor EV'],
    'Car_price': [7.98, 8.69, 12.49],
    'Battery_cap': [17.3, 19.2, 26.0],
    'Drive_range': [230, 250, 315],
    'Power': [41.42, 60.34, 73.75],
    'Charge_time': [7.00000, 0.96667, 7.50000]
}
df3 = pd.DataFrame(data)

# Select only numeric columns for correlation calculation
df_numeric = df3.select_dtypes(include=[float, int])

# Calculate the correlation matrix
data_corr = df_numeric.corr()

# Plot the correlation matrix heatmap
plt.figure(figsize=(8, 6))
plt.title("Correlation Matrix HeatMap")

sns.heatmap(data_corr, annot=True, fmt=".2f", cmap="inferno", linewidth=0.5)

plt.xticks(rotation=45)
plt.yticks(rotation=0)

plt.show()

```



In [ ]:

```python

```