```
import pandas as pd
import matplotlib.pyplot as plt

df = pd.read_csv("evp.csv")

df.head(1)
```

```
   VIN (1-10) County     City State  Postal Code  Model Year   Make
Model  \
0  5YJ3E1EA5L   King  Seattle    WA      98133.0        2020  TESLA
MODEL 3

            Electric Vehicle Type  \
0  Battery Electric Vehicle (BEV)

  Clean Alternative Fuel Vehicle (CAFV) Eligibility  Electric Range  \
0           Clean Alternative Fuel Vehicle Eligible             266

   Base MSRP  Legislative District  DOL Vehicle ID  \
0          0                 46.0       249903111

              Vehicle Location                                Electric
Utility  \
0  POINT (-122.35029 47.71871)  CITY OF SEATTLE - (WA)|CITY OF TACOMA
- (WA)

   2020 Census Tract
0       5.303300e+10
```

```
df.columns
```

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

```
df.isnull().sum()
```

```
VIN (1-10)                                           0
County                                               3
City                                                 3
State                                                0
Postal Code                                          3
Model Year                                           0
Make                                                 0
Model                                                0
Electric Vehicle Type                                0
```

```
Clean Alternative Fuel Vehicle (CAFV) Eligibility      0
Electric Range                                         0
Base MSRP                                              0
Legislative District                                 463
DOL Vehicle ID                                         0
Vehicle Location                                       9
Electric Utility                                       3
2020 Census Tract                                      3
dtype: int64
```

How many unique electric vehicle (EV) models are available in the dataset?

```
df.describe()
```

```
         Postal Code     Model Year  Electric Range      Base MSRP  \
count  220222.000000  220225.000000   220225.000000  220225.000000
mean    98176.179355    2021.194242       48.724423     852.456874
std      2534.666722       2.981490       85.874160    7469.168138
min      1731.000000    1999.000000        0.000000       0.000000
25%     98052.000000    2020.000000        0.000000       0.000000
50%     98125.000000    2022.000000        0.000000       0.000000
75%     98374.000000    2023.000000       42.000000       0.000000
max     99577.000000    2025.000000      337.000000  845000.000000

       Legislative District  DOL Vehicle ID  2020 Census Tract
count         219762.000000    2.202250e+05       2.202220e+05
mean              28.907909    2.315541e+08       5.298061e+10
std               14.911386    6.983360e+07       1.524348e+09
min                1.000000    4.385000e+03       1.001020e+09
25%               17.000000    1.980609e+08       5.303301e+10
50%               32.000000    2.448443e+08       5.303303e+10
75%               42.000000    2.659792e+08       5.305307e+10
max               49.000000    4.792548e+08       5.602100e+10
```

```
df["Electric Vehicle Type"].unique()
```

```
array(['Battery Electric Vehicle (BEV)',
       'Plug-in Hybrid Electric Vehicle (PHEV)'], dtype=object)
```

```
df["Model"].unique()
```

```
array(['MODEL 3', 'X5', 'VOLT', 'ID.4', 'EV9', 'NIRO', 'R1T', 'MODEL
X',
       'PRIUS PRIME', 'LEAF', 'I3', 'WRANGLER', 'MODEL Y', 'BZ4X',
       'MODEL S', '500', 'C-MAX', 'X3', 'XC90', 'PACIFICA', 'E-TRON',
       'NX', '330E', 'SORENTO', 'XC60', 'CYBERTRUCK', 'TONALE',
       'GRAND CHEROKEE', 'RAV4 PRIME', 'I-PACE', 'A3', 'FUSION',
       'BOLT EV', 'EV6', 'CAYENNE', 'PS2', 'HARDTOP', 'CX-90',
'HORNET',
       'COUNTRYMAN', 'I4', 'CLARITY', 'CX-70', 'I8', 'Q5', 'RX',
```

```
'530E',
       'Q5 E', 'SOUL', 'IONIQ 6', 'TUCSON', 'XM', 'SPARK', 'V60',
       'PRIUS PLUG-IN', 'EQE-CLASS SUV', 'LYRIQ', 'EQS-CLASS SEDAN',
       'IONIQ 5', 'F-150', 'XC40', 'ROADSTER', 'TAYCAN', 'MUSTANG
MACH-E',
       'KONA ELECTRIC', 'ARIYA', 'EQB-CLASS', 'BOLT EUV', 'SOLTERRA',
       'IONIQ', 'R1S', 'E-TRON GT', 'EQUINOX EV', 'SANTA FE', 'BLAZER
EV',
       'EQS-CLASS SUV', 'IX', 'GLE-CLASS', 'B-CLASS', 'E-GOLF', 'RZ',
       'FOCUS', 'Q8', 'AVIATOR', 'PROLOGUE', 'TRANSIT', 'GLC-CLASS',
       'EQE-CLASS SEDAN', 'ESCAPE', 'OUTLANDER', 'S60', 'SPORTAGE',
       'PANAMERA', 'C40', 'KONA', 'FORTWO ELECTRIC DRIVE',
       'RANGE ROVER SPORT', 'AIR', 'SONATA', 'SOUL EV', 'GV60', 'Q4',
       'E-TRON SPORTBACK', 'CORSAIR', 'I5', 'HUMMER EV SUV', 'OCEAN',
       'ZDX', '500E', 'RAV4', 'EQ FORTWO', 'RS E-TRON GT', 'OPTIMA',
'I7',
       'CROSSTREK', 'I-MIEV', 'ELR', 'SILVERADO EV', 'RANGE ROVER',
       'POLESTAR 3', 'FORTWO', 'SQ8', 'MX-30', 'HUMMER EV PICKUP',
       'C-CLASS', 'GV70', 'S-CLASS', 'EDV', 'ACCORD', 'TX', 'IONIQ 5
N',
       '740E', 'ESPRINTER', 'CITY', 'S90', 'MACAN', 'G80', '745E', 'VF
8',
       'A7 E', 'EX40', 'ZEVO', 'ONE', 'RANGER', 'KARMA', 'CR-V',
'918',
       '750E', 'TRANSIT CONNECT ELECTRIC', 'SPECTRE', 'FLYING SPUR',
       'BENTAYGA', 'CT6', 'WHEEGO', 'A8 E', 'PROMASTER 3500', 'G-
CLASS',
       '745LE', 'REVUELTO'], dtype=object)
```

```
df["Model"].nunique()
```

```
159
```

There are 159 Electric Vehicle Models in this data set

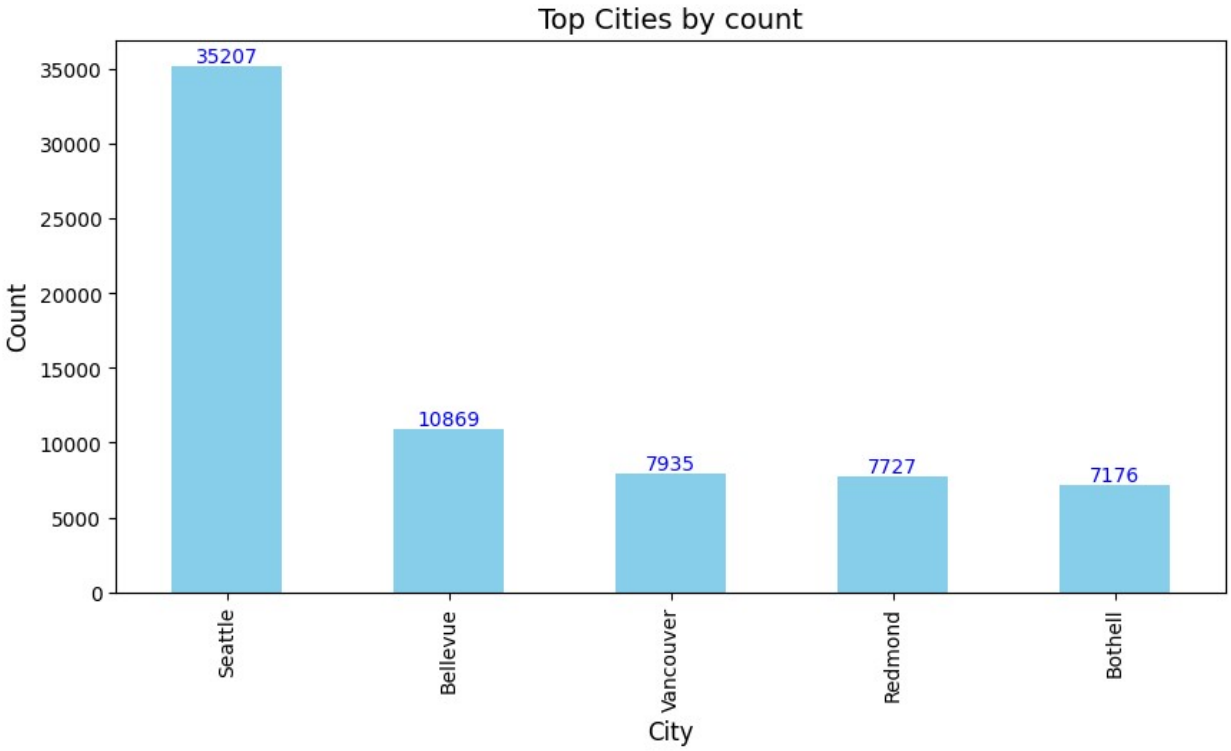
Which cities and counties have the highest concentration of electric vehicles?

```
df.columns
```

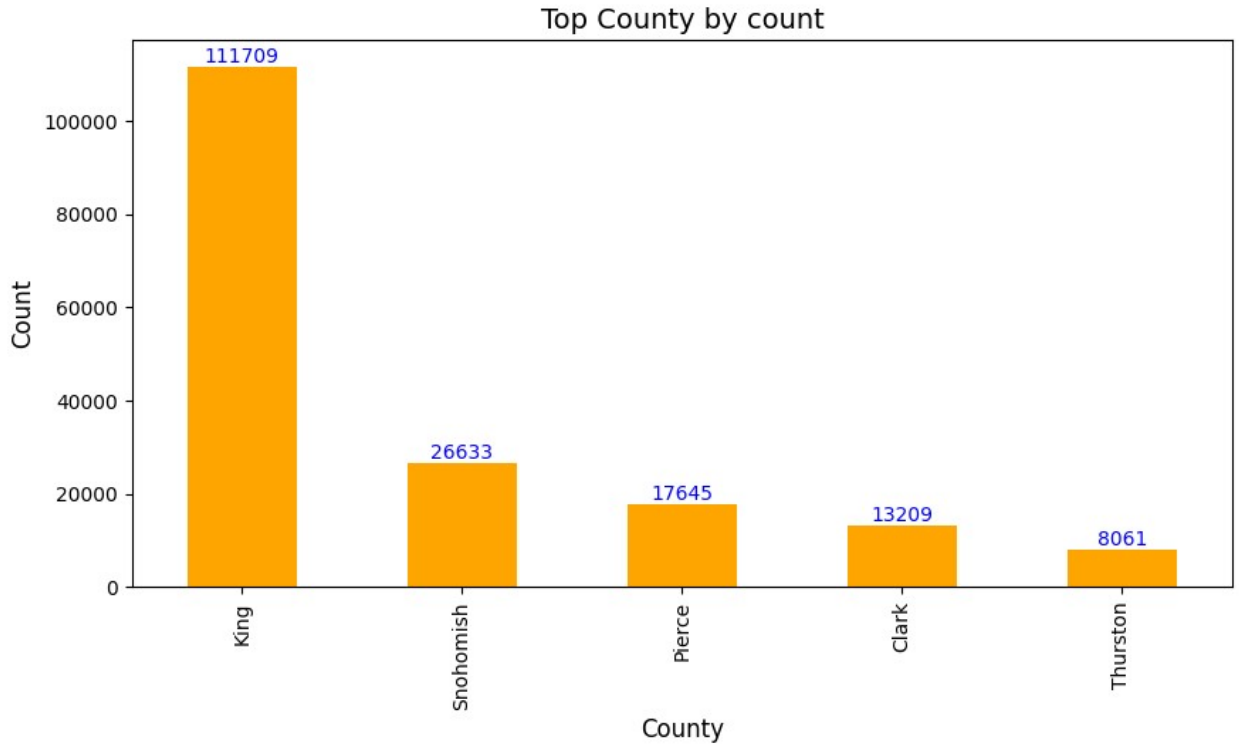
```
Index(['VIN (1-10)', 'County', 'City', 'State', 'Postal Code', 'Model
Year',
       'Make', 'Model', 'Electric Vehicle Type',
       'Clean Alternative Fuel Vehicle (CAFV) Eligibility', 'Electric
Range',
       'Base MSRP', 'Legislative District', 'DOL Vehicle ID',
       'Vehicle Location', 'Electric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

```
top_cities = df["City"].value_counts().head()
```

```
plt.figure(figsize=(10, 5))
bars = top_cities.plot(kind='bar', color='skyblue')

# Adding data labels
for bar in bars.patches:
    plt.text(bar.get_x() + bar.get_width() / 2,  # x-coordinate
             bar.get_height() - 0.1 if bar.get_height() > 0 else 0.1,
# y-coordinate
             f'{int(bar.get_height())}',  # Label text
             ha='center', va='bottom', fontsize=10, color='blue')

# bars.patches : Contains each bar object in plot
# plt.text : adds text to the plot at specified coordinates.
# bar.get_x() : retrievs the x-coordinates (start) of the bar.
# bar.get_width() : Width of the bar
# bar.get_x() + bar.get_width() / 2 : Calculates the center of the bar
for placing the label horizontally
# bar.height() : retrieve the height of bar, which represent actual
count
# The general form of a ternary conditional statement in Python is:
# value_if_true if condition else value_if_false
# bar.get_height() - 0.1 if bar.get_height() > 0 else 0.1
# bar.get_height() > 0 : checks height of bar is positive
# if height is positive then
# bar.get_height() - 0.1 - Places the label slightly below the top of
the bar (subtracting 0.1 ensures some space)
# if height is negetive or 0
# Positions the label slightly above the baseline of the bar.

plt.title("Top Cities by count", fontsize=14)
plt.xlabel("City",fontsize=12)
plt.ylabel("Count",fontsize=12)
plt.show()
```




```
top_county = df["County"].value_counts().head()
plt.figure(figsize=(10, 5))
bars = top_county.plot(kind='bar', color='orange')
for bar in bars.patches:
    plt.text(bar.get_x() + bar.get_width() / 2, # x coordinate
             bar.get_height() - 0.1 if bar.get_height() > 0 else 0.1,
# y coordinate
             f'{int(bar.get_height())}', # label text
             ha="center", va="bottom", fontsize = 10, color="blue")

plt.title("Top County by count", fontsize=14)
plt.xlabel("County",fontsize=12)
plt.ylabel("Count",fontsize=12)
plt.show()
```




```
df.columns
```

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

```
df["State"].value_counts(normalize=True) * 100
```

```
State
WA    99.789760
CA     0.050403
VA     0.027245
MD     0.016801
TX     0.013168
NC     0.009082
CO     0.007719
FL     0.005903
GA     0.005903
CT     0.004541
IL     0.004541
AZ     0.004087
NY     0.004087
```

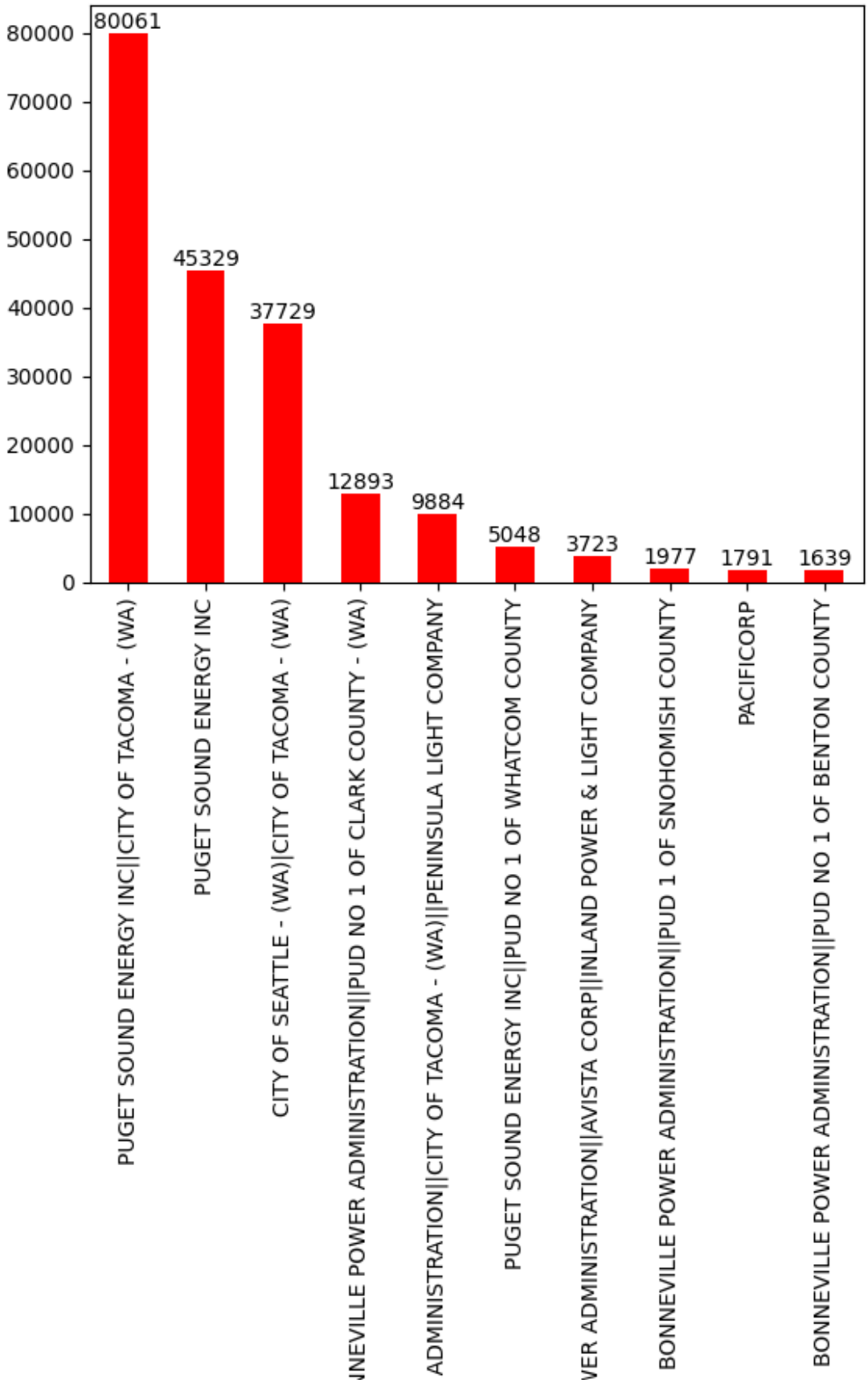
```
MA     0.003633
NJ     0.003633
AL     0.003633
OR     0.003633
NV     0.003633
SC     0.003179
MI     0.002724
MO     0.002724
PA     0.002724
HI     0.002270
KS     0.002270
ID     0.002270
KY     0.002270
DC     0.001816
OH     0.001816
RI     0.001816
TN     0.001362
NM     0.000908
DE     0.000908
UT     0.000908
ME     0.000908
NE     0.000908
IN     0.000908
AR     0.000908
MN     0.000454
OK     0.000454
NS     0.000454
AE     0.000454
WI     0.000454
LA     0.000454
WY     0.000454
BC     0.000454
MS     0.000454
AK     0.000454
NH     0.000454
Name: proportion, dtype: float64
```

Which electric utility companies serve the most EV owners?

```
df.columns
```

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

```
top_electric_vehicle = df["Electric Utility"].value_counts().head(10)
bars = top_electric_vehicle.plot(kind="bar", color="red")

for bar in bars.patches:
    plt.text(bar.get_x() + bar.get_width() / 2,
    bar.get_height() - 0.1 if bar.get_height() > 0 else 0.1,
    f'{int(bar.get_height())}', ha="center", va="bottom", fontsize=10)
plt.show()
```


80061
45329
37729
12893
9884
5048
3723
1977
1791
1639
80000
70000
60000
50000
40000
30000
20000
10000
0
PUGET SOUND ENERGY INC||CITY OF TACOMA - (WA)
PUGET SOUND ENERGY INC
CITY OF SEATTLE - (WA)|CITY OF TACOMA - (WA)
INEVILLE POWER ADMINISTRATION||PUD NO 1 OF CLARK COUNTY - (WA)
ADMINISTRATION||CITY OF TACOMA - (WA)||PENINSULA LIGHT COMPANY
PUGET SOUND ENERGY INC||PUD NO 1 OF WHATCOM COUNTY
VER ADMINISTRATION||AVISTA CORP||INLAND POWER & LIGHT COMPANY
BONNEVILLE POWER ADMINISTRATION||PUD 1 OF SNOHOMISH COUNTY
PACIFICORP
BONNEVILLE POWER ADMINISTRATION||PUD NO 1 OF BENTON COUNTY

```
df["Electric Utility"].value_counts(normalize=True) * 100
```

```
Electric Utility
PUGET SOUND ENERGY INC||CITY OF TACOMA - (WA)
36.354678
PUGET SOUND ENERGY INC
20.583320
CITY OF SEATTLE - (WA)|CITY OF TACOMA - (WA)
17.132257
BONNEVILLE POWER ADMINISTRATION||PUD NO 1 OF CLARK COUNTY - (WA)
5.854547
BONNEVILLE POWER ADMINISTRATION||CITY OF TACOMA - (WA)||PENINSULA
LIGHT COMPANY                                    4.488198

...
CITY OF SEATTLE - (WA)
0.000908
BONNEVILLE POWER ADMINISTRATION||PUD NO 1 OF JEFFERSON COUNTY
0.000908
BONNEVILLE POWER ADMINISTRATION||CITY OF TACOMA - (WA)||ALDER MUTUAL
LIGHT CO, INC|PENINSULA LIGHT COMPANY     0.000908
BONNEVILLE POWER ADMINISTRATION||PUD NO 1 OF ASOTIN COUNTY
0.000454
BONNEVILLE POWER ADMINISTRATION||PENINSULA LIGHT COMPANY
0.000454
Name: proportion, Length: 75, dtype: float64
```

How many EVs are located in each Census Tract?

```
df.columns
```

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

```
df["2020 Census Tract"].value_counts().head()
```

```
2020 Census Tract
5.303303e+10    2631
5.303303e+10    1171
5.303303e+10     931
5.303301e+10     864
5.306701e+10     820
Name: count, dtype: int64
```

```
df["2020 Census Tract"].value_counts()
```

```
2020 Census Tract
5.303303e+10    2631
5.303303e+10    1171
5.303303e+10     931
5.303301e+10     864
5.306701e+10     820
                ...
6.073001e+09       1
2.403380e+10       1
3.200300e+10       1
2.116397e+10       1
6.073011e+09       1
Name: count, Length: 2184, dtype: int64
```

5.303303e+10 have most EVs

```
df.columns
```

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

```
df.groupby("Make")["Electric
Range"].mean().sort_values(ascending=False).head()
```

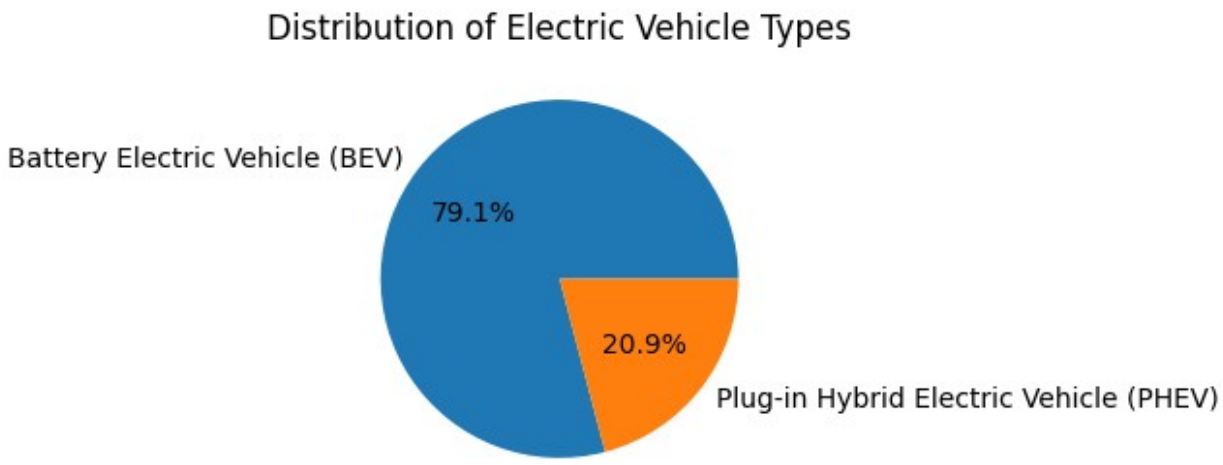
```
Make
JAGUAR                  199.296610
WHEEGO ELECTRIC CARS    100.000000
TH!NK                   100.000000
CHEVROLET                86.357826
FIAT                     80.165000
Name: Electric Range, dtype: float64
```

What is the most common type of electric vehicle?

```
df.columns
```

```
Index(['VIN (1-10)', 'County', 'City', 'State', 'Postal Code', 'Model
Year',
       'Make', 'Model', 'Electric Vehicle Type',
       'Clean Alternative Fuel Vehicle (CAFV) Eligibility', 'Electric
Range',
       'Base MSRP', 'Legislative District', 'DOL Vehicle ID',
```

```
       'Vehicle Location', 'Electric Utility', '2020 Census Tract'],
      dtype='object')

df["Electric Vehicle Type"].value_counts()

Electric Vehicle Type
Battery Electric Vehicle (BEV)            174212
Plug-in Hybrid Electric Vehicle (PHEV)     46013
Name: count, dtype: int64

et_type = df["Electric Vehicle Type"].value_counts(normalize=True) *
100
plt.figure(figsize=(3,3))
plt.pie(et_type, autopct='%1.1f%%',  labels=et_type.index)
plt.title("Distribution of Electric Vehicle Types")
plt.show()
```



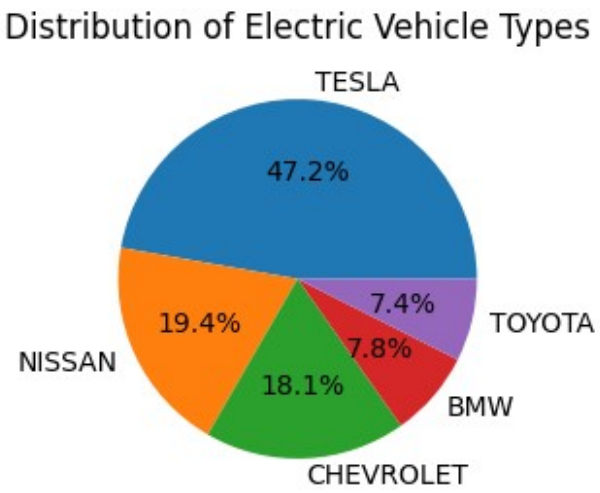
Top Manufacturers of CAFV-Eligible Vehicles

```
df.columns

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

top_cafv_makes = df[df["Clean Alternative Fuel Vehicle (CAFV)
Eligibility"] == "Clean Alternative Fuel Vehicle Eligible"]
["Make"].value_counts().head()
```

```
top_cafv_makes
```

```
Make
TESLA        25304
NISSAN       10414
CHEVROLET     9714
BMW           4175
TOYOTA        3977
Name: count, dtype: int64
```

```
plt.figure(figsize=(3,3))
plt.pie(top_cafv_makes, autopct='%1.1f%%',
labels=top_cafv_makes.index)
plt.title("Distribution of Electric Vehicle Types")
plt.show()
```

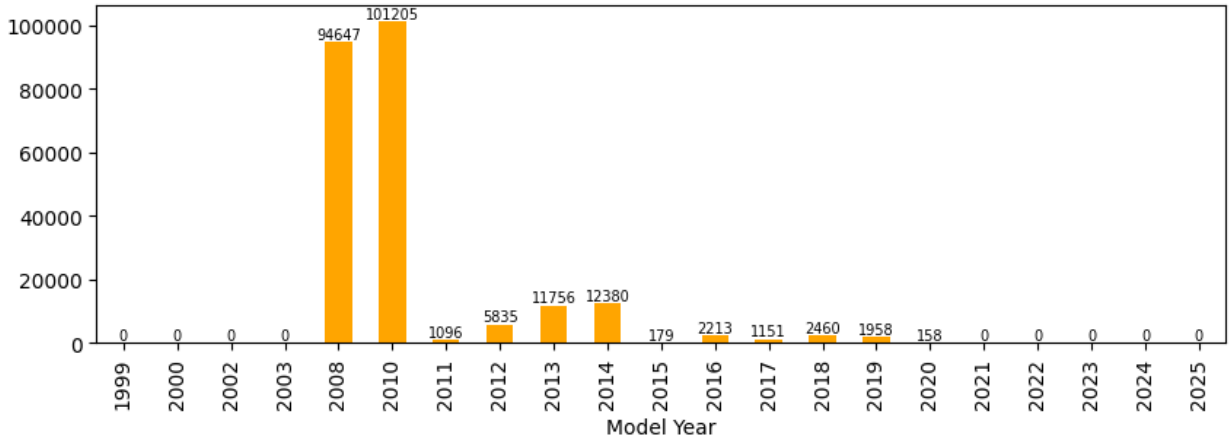



```
df.columns
```

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

```
avg_base_msrp = df.groupby("Model Year")["Base MSRP"].mean().round(2)
plt.figure(figsize=(10,3))
bars = avg_base_msrp.plot(kind="bar", color="orange")
for bar in bars.patches:
    plt.text(bar.get_x() + bar.get_width() / 2,
             bar.get_height() - 0.1 if bar.get_height() > 0 else 0.1,
```

```
            f"{int(bar.get_height())}",
            ha="center", va="bottom", fontsize=7)
plt.show()
```



In 2008 (94,647) and 2010 (1,01,205) the base MSRP is outlier

```
df[df['Model Year'].isin([2008,2010])][['Model Year', 'Base
MSRP',"Make"]]
```

	Model Year	Base MSRP	Make
311	2010	110950	TESLA
2782	2008	98950	TESLA
11627	2010	110950	TESLA
11678	2008	98950	TESLA
15084	2008	98950	TESLA
16296	2008	98950	TESLA
16529	2008	98950	TESLA
23506	2010	110950	TESLA
24171	2010	110950	TESLA
32894	2010	110950	TESLA
36088	2010	110950	TESLA
44602	2010	110950	TESLA
52197	2010	110950	TESLA
54712	2008	98950	TESLA
65046	2008	98950	TESLA
66277	2008	98950	TESLA
68745	2010	110950	TESLA
71010	2008	98950	TESLA
72043	2010	110950	TESLA
72155	2010	110950	TESLA
79781	2010	110950	TESLA
83270	2010	110950	TESLA
86293	2008	98950	TESLA
89515	2010	110950	TESLA
92329	2008	98950	TESLA

```
111329        2008      98950                 TESLA
111383        2008      98950                 TESLA
112124        2010      32995  WHEEGO ELECTRIC CARS
119098        2010      32995  WHEEGO ELECTRIC CARS
120486        2008      98950                 TESLA
121310        2008          0                 TESLA
132083        2010     110950                 TESLA
136735        2008      98950                 TESLA
149702        2008      98950                 TESLA
153317        2008      98950                 TESLA
158303        2008      98950                 TESLA
162142        2010     110950                 TESLA
166954        2008      98950                 TESLA
169273        2010     110950                 TESLA
169338        2008      98950                 TESLA
173209        2010     110950                 TESLA
188403        2010      32995  WHEEGO ELECTRIC CARS
189187        2010     110950                 TESLA
197400        2008      98950                 TESLA
203400        2008      98950                 TESLA
215166        2010     110950                 TESLA
217042        2010     110950                 TESLA
```

```
df[df["Make"].isin(["TESLA"])]["Model Year"]
```

```
0         2020
3         2018
7         2020
11        2018
15        2020
          ...
220217    2024
220219    2023
220221    2018
220222    2019
220224    2018
Name: Model Year, Length: 95378, dtype: int64
```

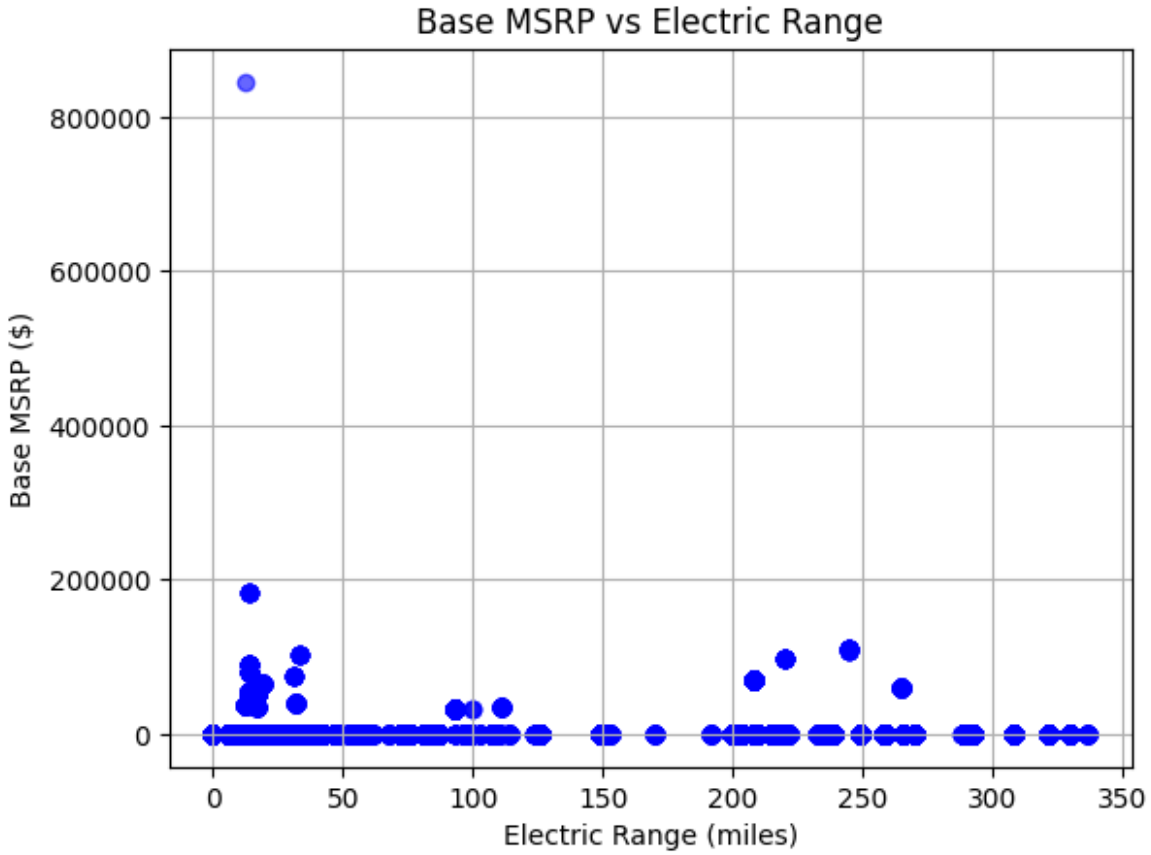
```
df.columns
```

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

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

```
                Base MSRP  Electric Range
Base MSRP        1.000000        0.115324
Electric Range   0.115324        1.000000
```

```
plt.scatter(df['Electric Range'], df['Base MSRP'], alpha=0.6,
color='blue')
plt.title('Base MSRP vs Electric Range')
plt.xlabel('Electric Range (miles)')
plt.ylabel('Base MSRP ($)')
plt.grid()
plt.show()
```




```
df['Price Range'] = pd.cut(df['Base MSRP'], bins=[0, 40000, 80000,
120000, 200000], labels=['Low', 'Mid', 'High', 'Luxury'])
df.groupby('Price Range', observed=False)['Electric Range'].mean()
```

```
Price Range
Low        67.561346
Mid       143.715367
High      113.362832
Luxury     14.000000
Name: Electric Range, dtype: float64
```

```
filtered_df = df[df['Base MSRP'] < 120000]
filtered_df[['Base MSRP', 'Electric Range']].corr()
```

```
                Base MSRP  Electric Range
Base MSRP        1.000000        0.121521
Electric Range   0.121521        1.000000
```

The value 0.115 indicates a very weak positive correlation. The electric range does not heavily influence the base MSRP.Some manufacturers may price vehicles based on brand perception or additional features rather than the electric range alone.

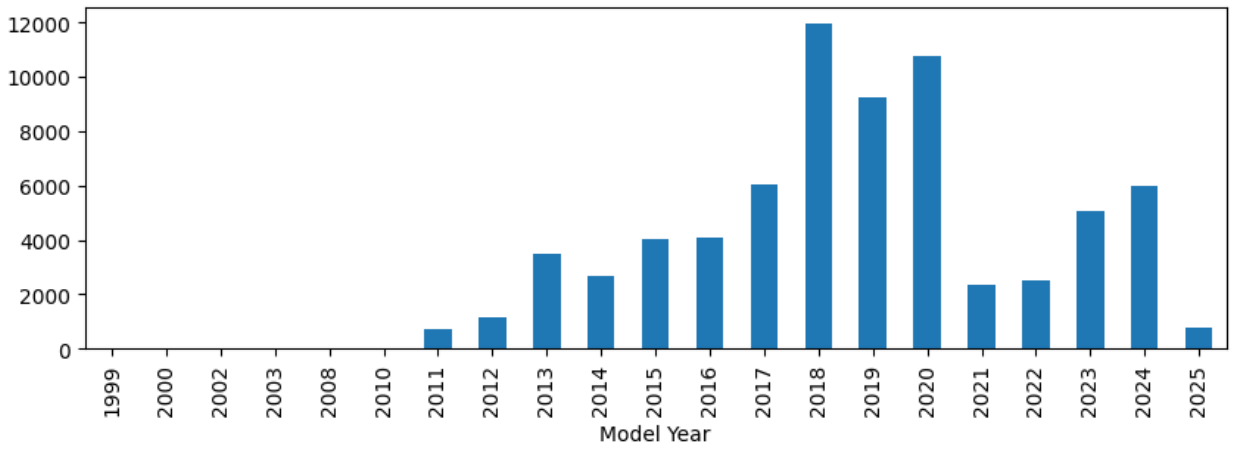
```
df.columns
```

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

```
cafv_trend = df[df['Clean Alternative Fuel Vehicle (CAFV)
Eligibility'] == 'Clean Alternative Fuel Vehicle Eligible']
cafv_count_by_year = cafv_trend['Model
Year'].value_counts().sort_index()
cafv_count_by_year
```

```
Model Year
1999        2
2000        7
2002        2
2003        1
2008       22
2010       24
2011      696
2012     1162
2013     3472
2014     2680
2015     4038
2016     4106
2017     6068
2018    11955
2019     9265
2020    10764
2021     2363
2022     2529
2023     5063
2024     6010
```

```
2025      787
Name: count, dtype: int64

plt.figure(figsize=(10,3))
cafv_count_by_year.plot(kind="bar")
plt.show()
```



Which counties have the highest average electric range for vehicles?

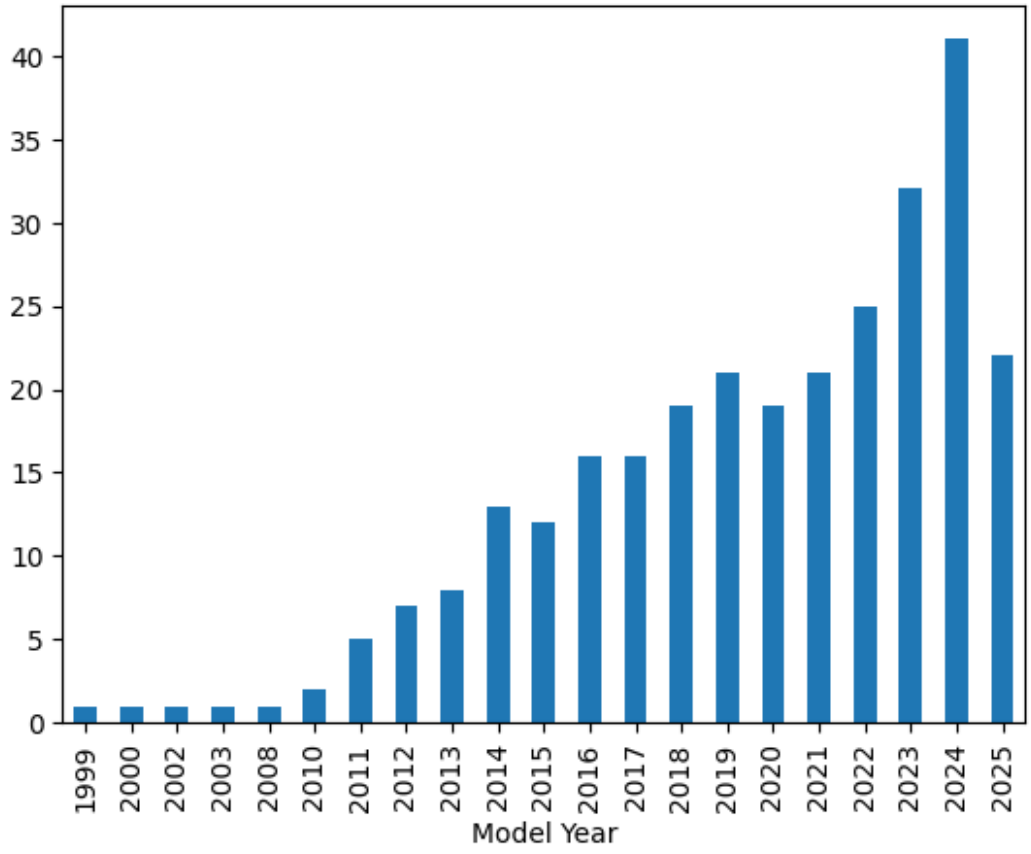
```
df.groupby('County')['Electric
Range'].mean().sort_values(ascending=False).head(10)

County
Chatham       322.0
St. Louis     322.0
Denton        322.0
Rockingham    308.0
Nassau        293.0
Palm Beach    291.0
Travis        265.0
Chesapeake    259.0
Arlington     220.0
Lee           220.0
Name: Electric Range, dtype: float64
```

How has the diversity of makes and models evolved over time?

```
df.groupby("Model Year")["Make"].nunique()

Model Year
1999     1
2000     1
2002     1
2003     1
2008     1
```

```
2010     2
2011     5
2012     7
2013     8
2014    13
2015    12
2016    16
2017    16
2018    19
2019    21
2020    19
2021    21
2022    25
2023    32
2024    41
2025    22
Name: Make, dtype: int64
```

```
diverity_over_time = df.groupby("Model Year")["Make"].nunique()
diverity_over_time.plot(kind="bar")
plt.show()
```



```
df.groupby("Model Year")["Make"].apply(set)
```

```
Model Year
1999                                                    {FORD}
2000                                                    {FORD}
2002                                                  {TOYOTA}
2003                                                  {TOYOTA}
2008                                                   {TESLA}
2010                            {TESLA, WHEEGO ELECTRIC CARS}
2011    {TESLA, CHEVROLET, AZURE DYNAMICS, TH!NK, NISSAN}
2012    {TESLA, CHEVROLET, TOYOTA, MITSUBISHI, FISKER,...
2013    {CHEVROLET, TESLA, HONDA, SMART, TOYOTA, FORD,...
2014    {BMW, CHEVROLET, TESLA, HONDA, CADILLAC, MERCE...
2015    {BMW, TESLA, CHEVROLET, MERCEDES-BENZ, SMART, ...
2016    {BMW, CHEVROLET, TESLA, AUDI, CADILLAC, MERCED...
2017    {BMW, CHEVROLET, TESLA, AUDI, CADILLAC, MERCED...
2018    {TOYOTA, PORSCHE, VOLVO, TESLA, MERCEDES-BENZ,...
2019    {JAGUAR, TOYOTA, PORSCHE, VOLVO, TESLA, MERCED...
2020    {LINCOLN, TOYOTA, JAGUAR, PORSCHE, VOLVO, TESL...
2021    {LINCOLN, TOYOTA, PORSCHE, VOLVO, TESLA, KIA, ...
2022    {LINCOLN, RIVIAN, TOYOTA, JAGUAR, PORSCHE, VOL...
2023    {LINCOLN, RIVIAN, TOYOTA, GENESIS, JAGUAR, FIS...
2024    {LINCOLN, RIVIAN, TOYOTA, JAGUAR, GENESIS, RAM...
2025    {LINCOLN, RIVIAN, GENESIS, PORSCHE, VOLVO, TES...
Name: Make, dtype: object
```

```
df.columns
```

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

```
eligible = df[df["Clean Alternative Fuel Vehicle (CAFV) Eligibility"]
== "Clean Alternative Fuel Vehicle Eligible"]
```

```
eligible[["Model","Make"]].value_counts().head()
```

```
Model    Make
MODEL 3  TESLA        13723
LEAF     NISSAN       10414
MODEL S  TESLA         5916
VOLT     CHEVROLET     4760
BOLT EV  CHEVROLET     4710
Name: count, dtype: int64
```