In [1]:
```
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
import warnings

warnings.filterwarnings("ignore")
```

In [2]:
```
data=pd.read_csv("C:\\Users\\Dell\\OneDrive\\Desktop\\excel books\\EV-data\\Electri
```

In [3]:
```
data
```

Out[3]:

	VIN (1-10)	County	City	State	Postal Code	Model Year	Make	Model	El Ve
0	5YJYGDEE1L	King	Seattle	WA	98122.0	2020	TESLA	MODEL Y	B El V
1	7SAYGDEE9P	Snohomish	Bothell	WA	98021.0	2023	TESLA	MODEL Y	B El V
2	5YJSA1E4XK	King	Seattle	WA	98109.0	2019	TESLA	MODEL S	B El V
3	5YJSA1E27G	King	Issaquah	WA	98027.0	2016	TESLA	MODEL S	B El V
4	5YJYGDEE5M	Kitsap	Suquamish	WA	98392.0	2021	TESLA	MODEL Y	B El V
...	...	...	...	...	...	...	...	...	
177861	7SAYGDEE3N	Pierce	Bonney Lake	WA	98391.0	2022	TESLA	MODEL Y	B El V
177862	KM8K23AG1P	Mason	Shelton	WA	98584.0	2023	HYUNDAI	KONA ELECTRIC	B El V
177863	5YJYGDEE6M	Grant	Quincy	WA	98848.0	2021	TESLA	MODEL Y	B El V
177864	WVGKMPE27M	King	Black Diamond	WA	98010.0	2021	VOLKSWAGEN	ID.4	B El V
177865	5YJ3E1EA8M	Pierce	Tacoma	WA	98422.0	2021	TESLA	MODEL 3	B El V

	VIN (1-10)	County	City	State	Postal Code	Model Year	Make	Model	El Ve

In [4]:
```
data.head(5)
```

Out[4]:

	VIN (1-10)	County	City	State	Postal Code	Model Year	Make	Model	Electric Vehicle Type	Clea Alternativ Fue Vehicl (CAFV Eligibilit
0	5YJYGDEE1L	King	Seattle	WA	98122.0	2020	TESLA	MODEL Y	Battery Electric Vehicle (BEV)	Clear Alternative Fue Vehicle Eligible
1	7SAYGDEE9P	Snohomish	Bothell	WA	98021.0	2023	TESLA	MODEL Y	Battery Electric Vehicle (BEV)	Eligibility unknown as battery range has not b..
2	5YJSA1E4XK	King	Seattle	WA	98109.0	2019	TESLA	MODEL S	Battery Electric Vehicle (BEV)	Clear Alternative Fue Vehicle Eligible
3	5YJSA1E27G	King	Issaquah	WA	98027.0	2016	TESLA	MODEL S	Battery Electric Vehicle (BEV)	Clear Alternative Fue Vehicle Eligible
4	5YJYGDEE5M	Kitsap	Suquamish	WA	98392.0	2021	TESLA	MODEL Y	Battery Electric Vehicle (BEV)	Eligibility unknown as battery range has not b..

In [5]:
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 177866 entries, 0 to 177865
Data columns (total 17 columns):
 #   Column                                             Non-Null Count   Dtype  
---  ------                                             --------------   -----  
 0   VIN (1-10)                                         177866 non-null  object 
 1   County                                             177861 non-null  object 
 2   City                                               177861 non-null  object 
 3   State                                              177866 non-null  object 
 4   Postal Code                                        177861 non-null  float64
 5   Model Year                                         177866 non-null  int64  
 6   Make                                               177866 non-null  object 
 7   Model                                              177866 non-null  object 
 8   Electric Vehicle Type                              177866 non-null  object 
 9   Clean Alternative Fuel Vehicle (CAFV) Eligibility  177866 non-null  object 
 10  Electric Range                                     177866 non-null  int64  
 11  Base MSRP                                          177866 non-null  int64  
 12  Legislative District                               177477 non-null  float64
 13  DOL Vehicle ID                                     177866 non-null  int64  
 14  Vehicle Location                                   177857 non-null  object 
 15  Electric Utility                                   177861 non-null  object 
 16  2020 Census Tract                                  177861 non-null  float64
dtypes: float64(3), int64(4), object(10)
memory usage: 23.1+ MB
```

In [5]:
```
data.isnull().sum()
```

Out[5]:
```
VIN (1-10)                                             0
County                                                 5
City                                                   5
State                                                  0
Postal Code                                            5
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                                 389
DOL Vehicle ID                                         0
Vehicle Location                                       9
Electric Utility                                       5
2020 Census Tract                                      5
dtype: int64
```

In [6]:
```
data = data.dropna()
```

In [7]:
```
data.isnull().sum()
```

Out[7]:
```
VIN (1-10)                                           0
County                                               0
City                                                 0
State                                                0
Postal Code                                          0
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                                 0
DOL Vehicle ID                                       0
Vehicle Location                                     0
Electric Utility                                     0
2020 Census Tract                                    0
dtype: int64
```

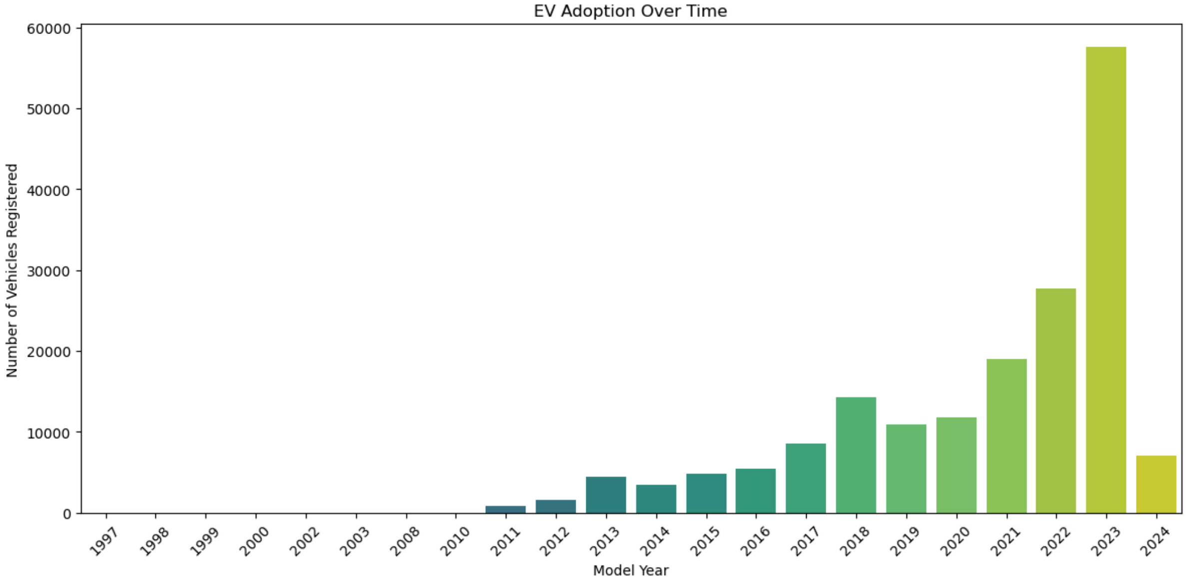
EV Adoption Over Time

In [8]:
```
import seaborn as sns
```

In [11]:
```
ev_adoption_by_year = data['Model Year'].value_counts().sort_index()
ev_adoption_by_year
```

Out[11]:
```
1997        1
1998        1
1999        5
2000        7
2002        2
2003        1
2008       19
2010       23
2011      775
2012     1614
2013     4399
2014     3496
2015     4826
2016     5469
2017     8534
2018    14286
2019    10913
2020    11740
2021    19063
2022    27708
2023    57519
2024     7072
Name: Model Year, dtype: int64
```

In [12]:
```
plt.figure(figsize=(12, 6))
sns.barplot(x=ev_adoption_by_year.index, y=ev_adoption_by_year.values, palette="vi
plt.title('EV Adoption Over Time')
plt.xlabel('Model Year')
plt.ylabel('Number of Vehicles Registered')
plt.xticks(rotation=45)
plt.tight_layout()
plt.show()
```



Geographical Distribution

In [43]:
```
data.head(2)
```

Out[43]:

	VIN (1-10)	County	City	State	Postal Code	Model Year	Make	Model	Electric Vehicle Type	Clean Alternative Fuel Vehicle (CAFV) Eligibility	E
0	5YJYGDEE1L	King	Seattle	WA	98122.0	2020	TESLA	MODEL Y	Battery Electric Vehicle (BEV)	Clean Alternative Fuel Vehicle Eligible	
1	7SAYGDEE9P	Snohomish	Bothell	WA	98021.0	2023	TESLA	MODEL Y	Battery Electric Vehicle (BEV)	Eligibility unknown as battery range has not b...	

In [14]:
```
ev_county_distribution = data['County'].value_counts()
```

In [15]:
```
ev_county_distribution=ev_county_distribution.head(3)
```

In [16]:
```
top_counties = ev_county_distribution.head(3).index
top_counties
```

Out[16]:
```
Index(['King', 'Snohomish', 'Pierce'], dtype='object')
```

In [17]:
```
top_counties_data = data[data['County'].isin(top_counties)]
```

In [18]:
```
top_counties_data
```

Out[18]:

	VIN (1-10)	County	City	State	Postal Code	Model Year	Make	Model	Ele Vel .
0	5YJYGDEE1L	King	Seattle	WA	98122.0	2020	TESLA	MODEL Y	Ba Ele Ve (
1	7SAYGDEE9P	Snohomish	Bothell	WA	98021.0	2023	TESLA	MODEL Y	Ba Ele Ve (
2	5YJSA1E4XK	King	Seattle	WA	98109.0	2019	TESLA	MODEL S	Ba Ele Ve (
3	5YJSA1E27G	King	Issaquah	WA	98027.0	2016	TESLA	MODEL S	Ba Ele Ve (
7	KNAGV4LD9J	Snohomish	Bothell	WA	98012.0	2018	KIA	OPTIMA	Plu Hy Ele Ve (P
...	...	...	...	...	...	...	...	...	
177858	5YJ3E1EB8N	Snohomish	Snohomish	WA	98296.0	2022	TESLA	MODEL 3	Ba Ele Ve (
177859	1N4BZ1DV7M	King	Redmond	WA	98053.0	2021	NISSAN	LEAF	Ba Ele Ve (
177861	7SAYGDEE3N	Pierce	Bonney Lake	WA	98391.0	2022	TESLA	MODEL Y	Ba Ele Ve (
177864	WVGKMPE27M	King	Black Diamond	WA	98010.0	2021	VOLKSWAGEN	ID.4	Ba Ele Ve (
177865	5YJ3E1EA8M	Pierce	Tacoma	WA	98422.0	2021	TESLA	MODEL 3	Ba Ele Ve (

	VIN (1-10)	County	City	State	Postal Code	Model Year	Make	Model	Ele Vel

In [19]:
```
ev_city_distribution_top_counties = top_counties_data.groupby(['County', 'City']).s
ev_city_distribution_top_counties
```

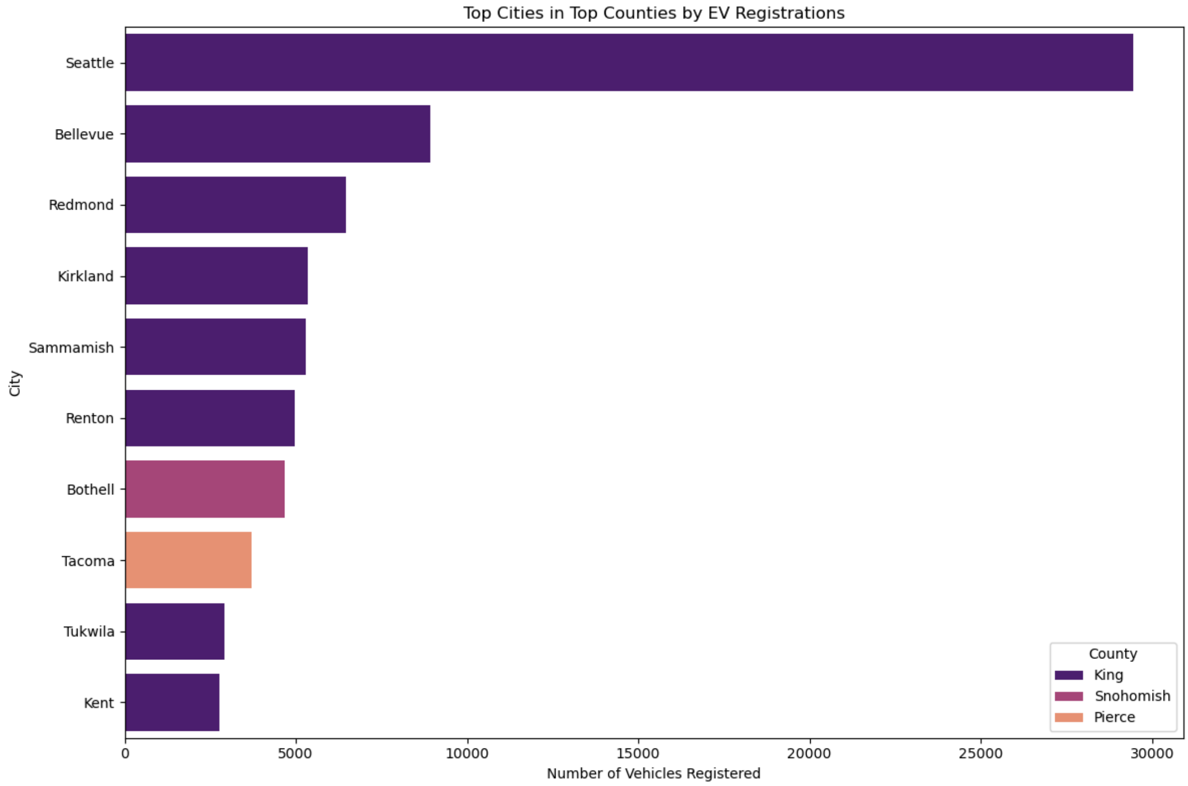
Out[19]:

	County	City	Number of Vehicles
0	King	Seattle	29447
1	King	Bellevue	8930
2	King	Redmond	6478
3	King	Kirkland	5362
4	King	Sammamish	5280
...	...	...	...
108	Snohomish	Alderwood Manor	1
109	Snohomish	Startup	1
110	King	Gold Bar	1
111	Pierce	Kapowsin	1
112	Pierce	Prairie Ridge	1

113 rows × 3 columns

In [21]:
```
top_cities = ev_city_distribution_top_counties.head(10)
```

In [24]:
```
plt.figure(figsize=(12, 8))
sns.barplot(x='Number of Vehicles', y='City', hue='County', data=top_cities, palett
plt.title('Top Cities in Top Counties by EV Registrations')
plt.xlabel('Number of Vehicles Registered')
plt.ylabel('City')
plt.legend(title='County')
plt.tight_layout()
plt.show()
```

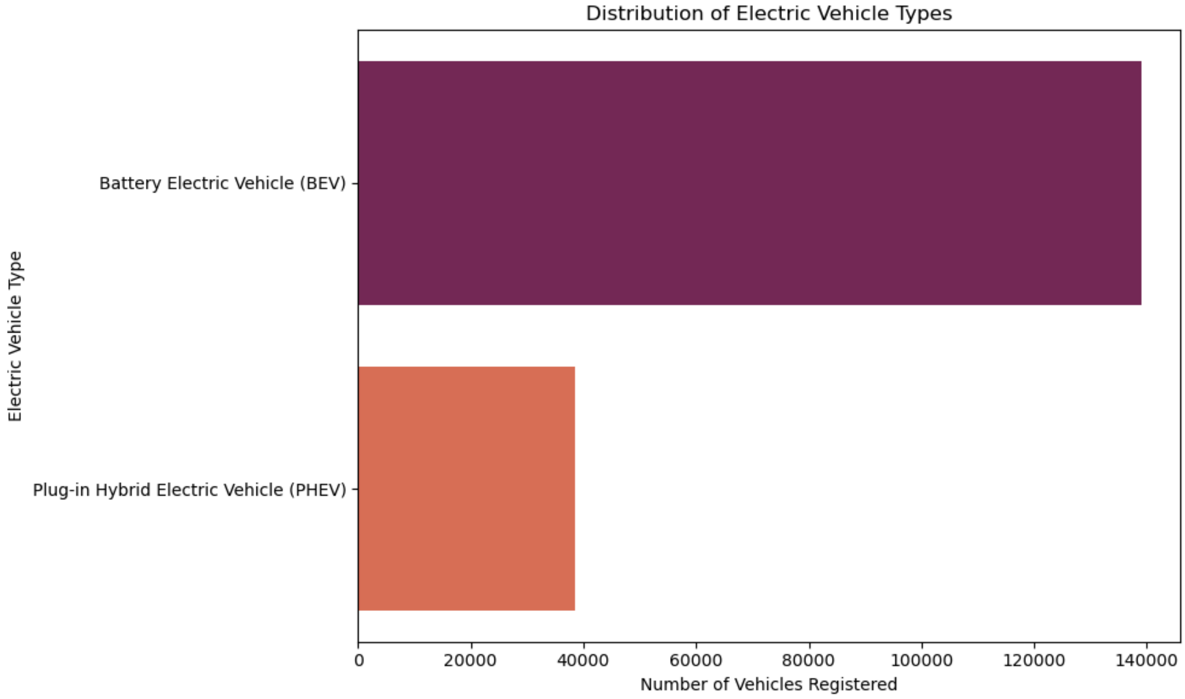


types of electric vehicles

In [25]:
```
ev_type_distribution = data['Electric Vehicle Type'].value_counts()
ev_type_distribution
```

Out[25]:
```
Battery Electric Vehicle (BEV)            138947
Plug-in Hybrid Electric Vehicle (PHEV)     38526
Name: Electric Vehicle Type, dtype: int64
```

In [27]:
```
plt.figure(figsize=(10, 6))
sns.barplot(x=ev_type_distribution.values, y=ev_type_distribution.index, palette="r
plt.title('Distribution of Electric Vehicle Types')
plt.xlabel('Number of Vehicles Registered')
plt.ylabel('Electric Vehicle Type')
plt.tight_layout()
plt.show()
```

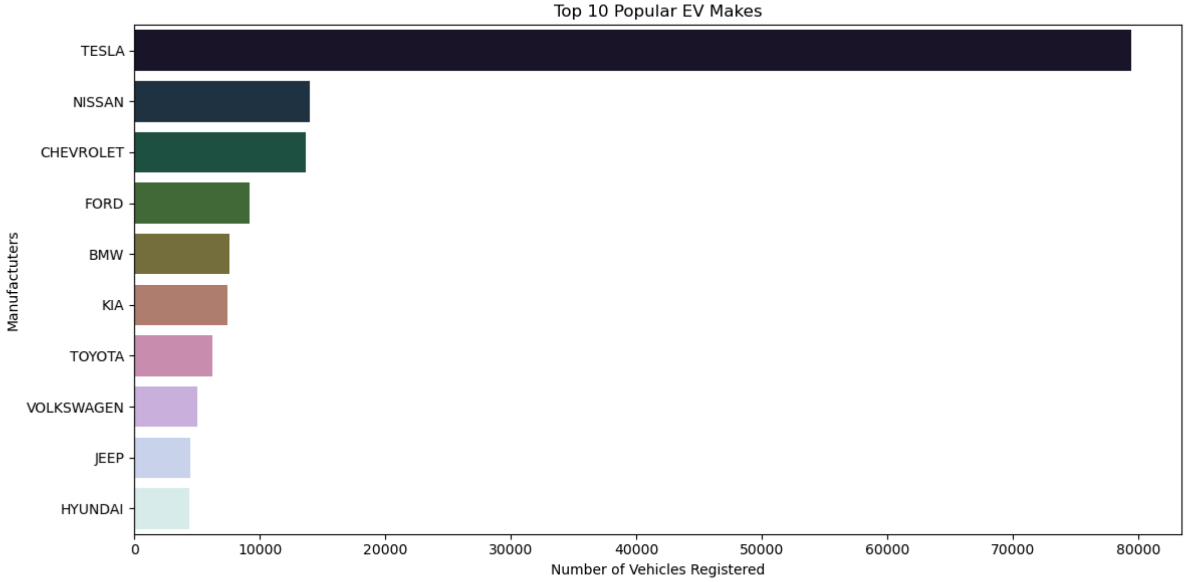


most popular manufacturers

In [31]:
```
ev_make_distribution = data['Make'].value_counts().head(10)
ev_make_distribution
```

Out[31]:
```
TESLA         79471
NISSAN        13984
CHEVROLET     13651
FORD           9177
BMW            7556
KIA            7423
TOYOTA         6254
VOLKSWAGEN     4993
JEEP           4468
HYUNDAI        4398
Name: Make, dtype: int64
```

In [32]:
```
plt.figure(figsize=(12, 6))
sns.barplot(x=ev_make_distribution.values, y=ev_make_distribution.index, palette="c
plt.title('Top 10 Popular EV Makes')
plt.xlabel('Number of Vehicles Registered')
plt.ylabel('Manufactuters')
plt.tight_layout()
plt.show()
```



In [34]:

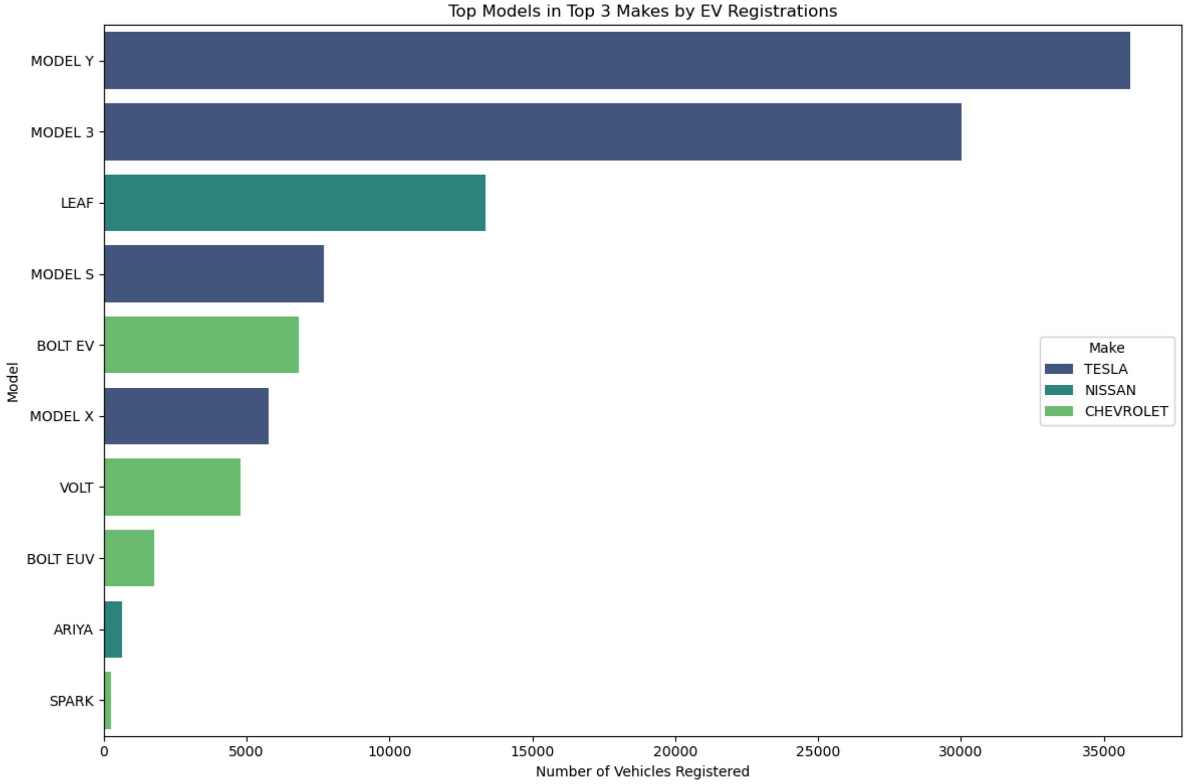
```
top_3_makes = ev_make_distribution.head(3).index

top_makes_data = data[data['Make'].isin(top_3_makes)]


ev_model_distribution_top_makes = top_makes_data.groupby(['Make', 'Model']).size().

top_models = ev_model_distribution_top_makes.head(10)

top_models
```

Out[34]:

	Make	Model	Number of Vehicles
0	TESLA	MODEL Y	35921
1	TESLA	MODEL 3	30009
2	NISSAN	LEAF	13352
3	TESLA	MODEL S	7711
4	CHEVROLET	BOLT EV	6811
5	TESLA	MODEL X	5784
6	CHEVROLET	VOLT	4782
7	CHEVROLET	BOLT EUV	1770
8	NISSAN	ARIYA	632
9	CHEVROLET	SPARK	240

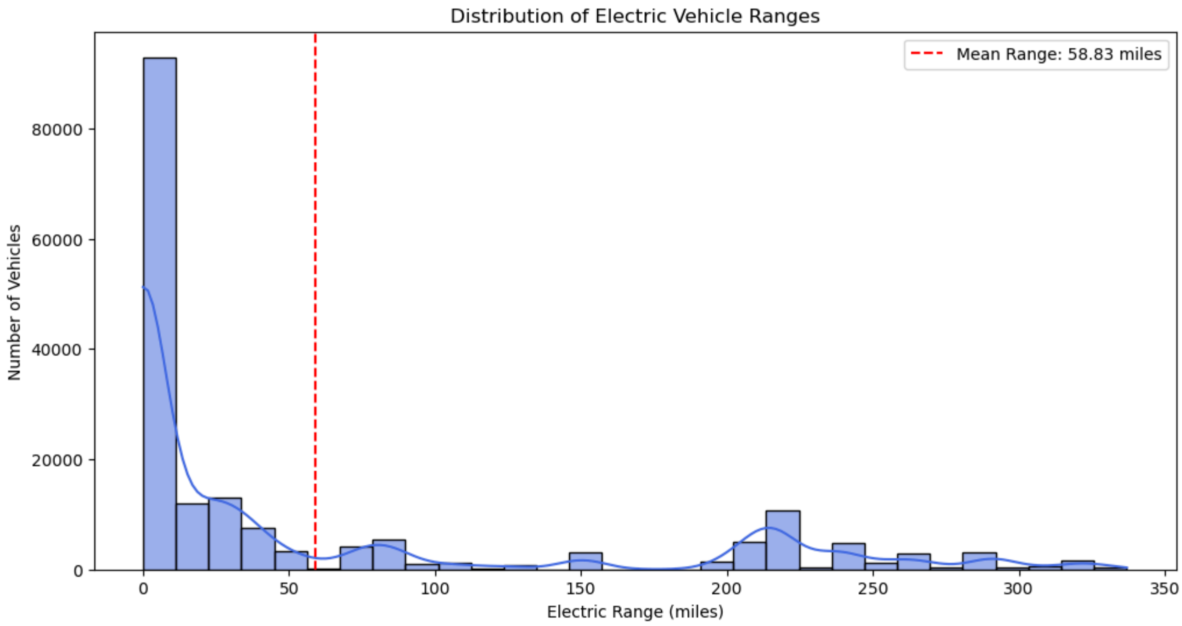
In [42]:

```
plt.figure(figsize=(12, 8))
sns.barplot(x='Number of Vehicles', y='Model', hue='Make', data=top_models, palette
plt.title('Top Models in Top 3 Makes by EV Registrations')
plt.xlabel('Number of Vehicles Registered')
plt.ylabel('Model')
plt.legend(title='Make', loc='center right')
plt.tight_layout()
plt.show()
```



Distribution of electric range

In [92]:
```
plt.figure(figsize=(12, 6))
sns.histplot(data['Electric Range'], bins=30, kde=True, color='royalblue')
plt.title('Distribution of Electric Vehicle Ranges')
plt.xlabel('Electric Range (miles)')
plt.ylabel('Number of Vehicles')
plt.axvline(data['Electric Range'].mean(), color='red', linestyle='--', label=f'Mea
plt.legend()
plt.show()
```



Electric range by model year

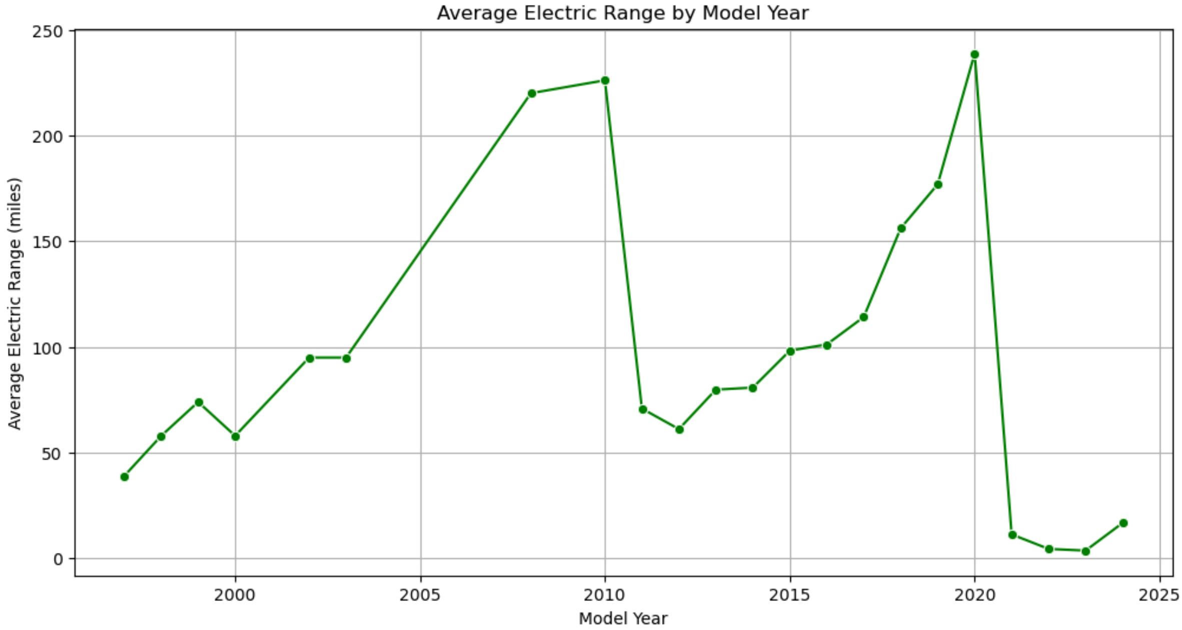
In [47]:
```
average_range_by_year = data.groupby('Model Year')['Electric Range'].mean().reset_i
average_range_by_year
```

Out[47]:

	Model Year	Electric Range
0	1997	39.000000
1	1998	58.000000
2	1999	74.000000
3	2000	58.000000
4	2002	95.000000
5	2003	95.000000
6	2008	220.000000
7	2010	226.086957
8	2011	70.891613
9	2012	61.172243
10	2013	79.822232
11	2014	80.798341
12	2015	98.254869
13	2016	101.197111
14	2017	114.162292
15	2018	156.165967
16	2019	176.918904
17	2020	238.748978
18	2021	11.402665
19	2022	4.518045
20	2023	3.729168
21	2024	16.791431

In [49]:

```
plt.figure(figsize=(12, 6))
sns.lineplot(x='Model Year', y='Electric Range', data=average_range_by_year, marker
plt.title('Average Electric Range by Model Year')
plt.xlabel('Model Year')
plt.ylabel('Average Electric Range (miles)')
plt.grid(True)
plt.show()
```

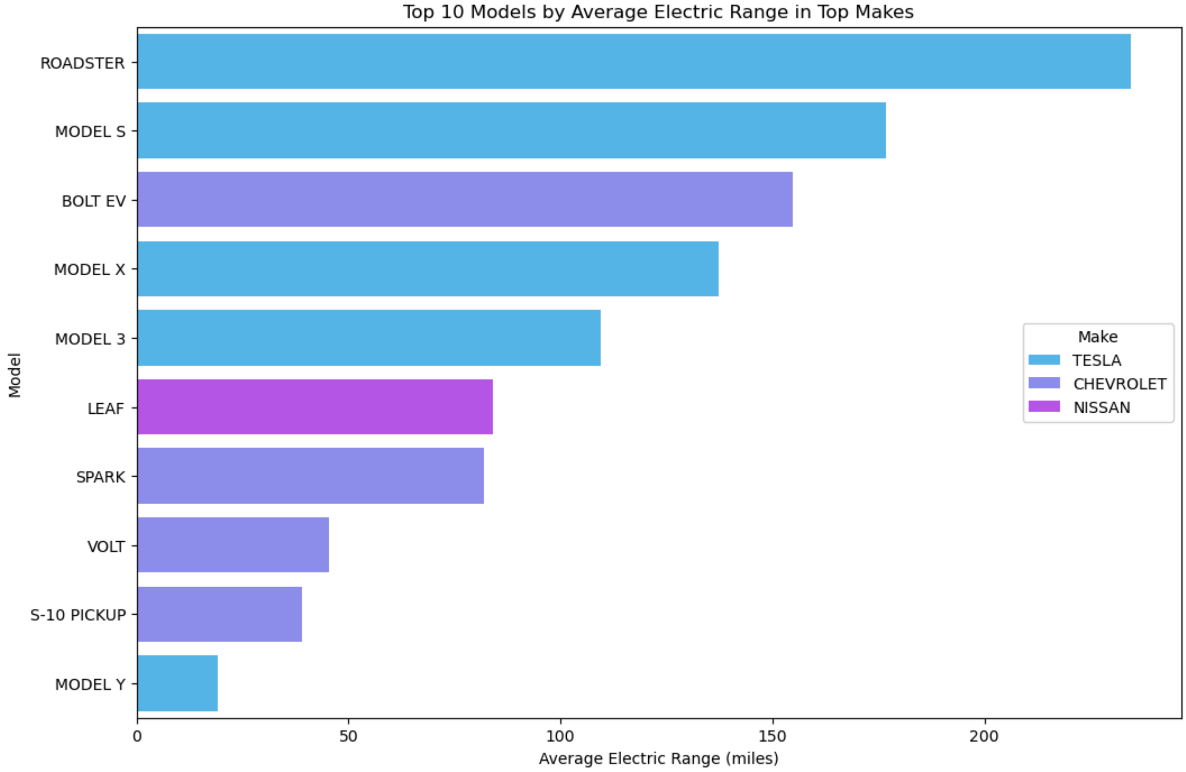


In [51]:
```
average_range_by_model = top_makes_data.groupby(['Make', 'Model'])['Electric Range'


top_range_models = average_range_by_model.head(10)
top_range_models
```

Out[51]:

	Make	Model	Electric Range
0	TESLA	ROADSTER	234.673913
1	TESLA	MODEL S	176.794449
2	CHEVROLET	BOLT EV	154.857143
3	TESLA	MODEL X	137.192600
4	TESLA	MODEL 3	109.463028
5	NISSAN	LEAF	84.148742
6	CHEVROLET	SPARK	82.000000
7	CHEVROLET	VOLT	45.365119
8	CHEVROLET	S-10 PICKUP	39.000000
9	TESLA	MODEL Y	19.191531

In [101…
```
plt.figure(figsize=(12, 8))
barplot = sns.barplot(x='Electric Range', y='Model', hue='Make', data=top_range_moc
plt.title('Top 10 Models by Average Electric Range in Top Makes')
plt.xlabel('Average Electric Range (miles)')
plt.ylabel('Model')
plt.legend(title='Make', loc='center right')
plt.show()
```



Estimation of market Size

In [59]:
```
ev_registration_counts = data['Model Year'].value_counts().sort_index()
ev_registration_counts
```

Out[59]:
```
1997        1
1998        1
1999        5
2000        7
2002        2
2003        1
2008       19
2010       23
2011      775
2012     1614
2013     4399
2014     3496
2015     4826
2016     5469
2017     8534
2018    14286
2019    10913
2020    11740
2021    19063
2022    27708
2023    57519
2024     7072
Name: Model Year, dtype: int64
```

In [60]:
```
from scipy.optimize import curve_fit
```

In [61]:
```
filtered_years = ev_registration_counts[ev_registration_counts.index <= 2023]
```

In [62]:
```
filtered_years
```

Out[62]:
```
1997        1
1998        1
1999        5
2000        7
2002        2
2003        1
2008       19
2010       23
2011      775
2012     1614
2013     4399
2014     3496
2015     4826
2016     5469
2017     8534
2018    14286
2019    10913
2020    11740
2021    19063
2022    27708
2023    57519
Name: Model Year, dtype: int64
```

In [63]:
```python
def exp_growth(x, a, b):
    return  a * np.exp(b * x)
```

In [67]:
```python
x_data = filtered_years.index - filtered_years.index.min()
x_data
```

Out[67]:
```
Int64Index([0, 1, 2, 3, 5, 6, 11, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23,
            24, 25, 26],
           dtype='int64')
```

In [68]:
```python
y_data = filtered_years.values
```

In [69]:
```python
y_data
```

Out[69]:
```
array([    1,     1,     5,     7,     2,     1,    19,    23,   775,
        1614,  4399,  3496,  4826,  5469,  8534, 14286, 10913, 11740,
       19063, 27708, 57519], dtype=int64)
```

In [70]:
```python
params, covariance = curve_fit(exp_growth, x_data, y_data)
```

the curve_fit function finds the optimal values of a and b that minimize the difference between the actual data and the model.

In [79]:
```python
forecast_years = np.arange(2024, 2024 + 7) - filtered_years.index.min()
forecasted_values = exp_growth(forecast_years, *params)
```

In [ ]:
```python

```

In [80]:
```python
forecasted_values
```

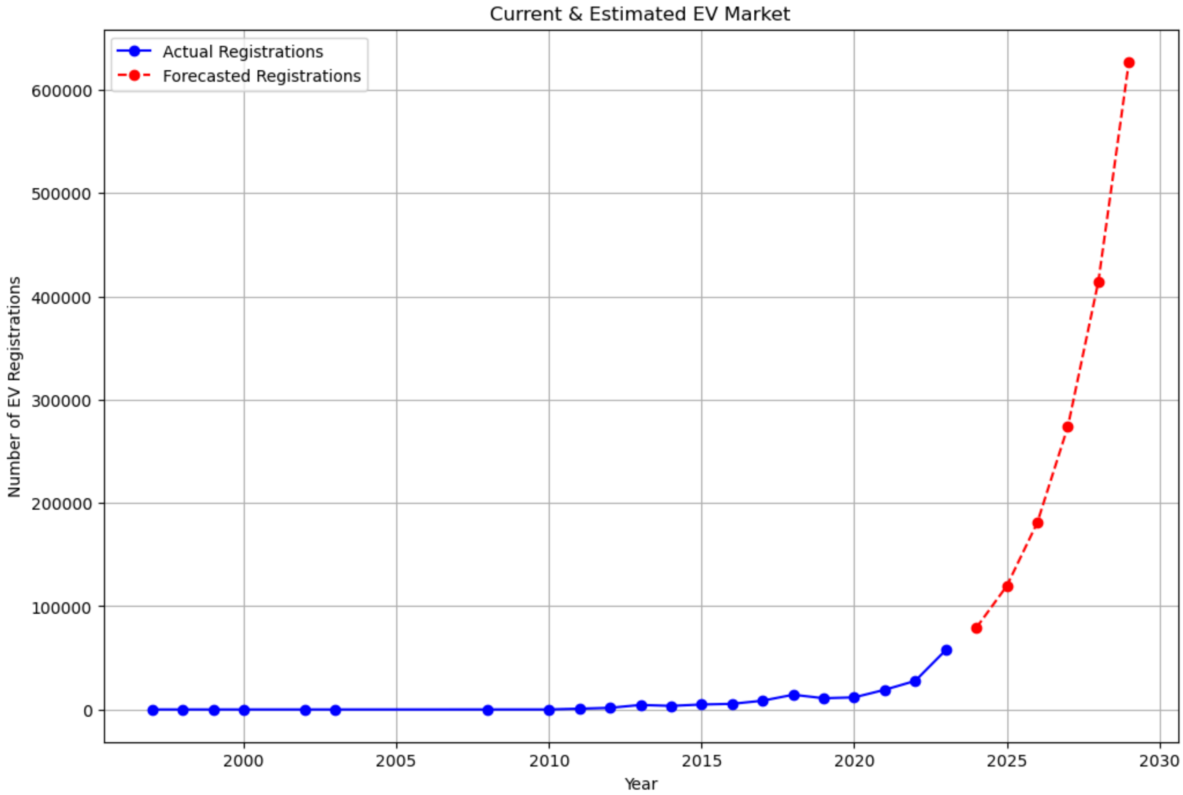
Out[80]:
```
array([ 79079.20808939, 119653.96274429, 181047.22020266, 273940.74706209,
       414497.01805382, 627171.31284077, 948966.6716959 ])
```

In [81]:
```python
forecasted_evs = dict(zip(forecast_years + filtered_years.index.min(), forecasted_v

print(forecasted_evs)
```

```
{2024: 79079.20808938889, 2025: 119653.96274428742, 2026: 181047.22020265696, 202
7: 273940.74706208805, 2028: 414497.01805382164, 2029: 627171.3128407666, 2030: 94
8966.6716959006}
```

In [82]:
```
years = np.arange(filtered_years.index.min(), 2030)
actual_years = filtered_years.index
forecast_years_full = np.arange(2024, 2030)
```

In [83]:
```
actual_values = filtered_years.values
forecasted_values_full = [forecasted_evs[year] for year in forecast_years_full]

plt.figure(figsize=(12, 8))
plt.plot(actual_years, actual_values, 'bo-', label='Actual Registrations')
plt.plot(forecast_years_full, forecasted_values_full, 'ro--', label='Forecasted Reg

plt.title('Current & Estimated EV Market')
plt.xlabel('Year')
plt.ylabel('Number of EV Registrations')
plt.legend()
plt.grid(True)
```



In [ ]: