Milestone 5

Code:

```
# Julia Cuellar
# DSC 540
# Final project

import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import urllib.request as urllib2
from bs4 import BeautifulSoup
import urllib3
import requests
import sqlite3


# Read csv file
def read_csv_file():
    evcp = pd.read_csv('2019 EVCP use Q1 and Q2.csv')
    print("Csv data:\n", evcp)


# Drop 1st column from csv file
def csv_drop():
    evcp = pd.read_csv('2019 EVCP use Q1 and Q2.csv')
    evcp.drop('Charging event', axis=1, inplace=True)
    print("Remove 1st column from csv data:\n", evcp)


# Check, replace, and recheck the nulls from csv file
def csv_cpr_null():
    evcp = pd.read_csv('2019 EVCP use Q1 and Q2.csv')
    evcp.drop('Charging event', axis=1, inplace=True)
    print("Display csv data with null:\n", evcp.isnull())
    print("Display counts of null from csv data:\n", evcp.isnull().sum())
    evcp = evcp.fillna(" ")
    print("Display csv data with replaced nulls:\n", evcp)
    print("Display recounts of null from csv data:\n", evcp.isnull().sum())


# Rename Model column from csv file
def csv_rename_col():
    evcp = pd.read_csv('2019 EVCP use Q1 and Q2.csv')
    evcp.drop('Charging event', axis=1, inplace=True)
    evcp = evcp.fillna(" ")
    evcp.rename(columns={'Model': 'Charge'}, inplace=True)
    print("Rename Model column from csv data:\n", evcp)


# Display count plot of Total kWh column from csv file
def csv_showCountplot_kWh():
    evcp = pd.read_csv('2019 EVCP use Q1 and Q2.csv')
    evcp.drop('Charging event', axis=1, inplace=True)
    evcp = evcp.fillna(" ")
    evcp.rename(columns={'Model': 'Charge'}, inplace=True)
```

```
    sns.countplot(x='Total kWh', data=evcp)
    plt.title('kWh')
    plt.show()


# Display count plot of Site column from csv file
def csv_showCountplot_Site():
    evcp = pd.read_csv('2019 EVCP use Q1 and Q2.csv')
    evcp.drop('Charging event', axis=1, inplace=True)
    evcp = evcp.fillna(" ")
    evcp.rename(columns={'Model': 'Charge'}, inplace=True)
    sns.countplot(x='Site', data=evcp)
    plt.title('Site')
    plt.show()


# Display count plot of Charge column from csv file
def csv_showCountplot_Charge():
    evcp = pd.read_csv('2019 EVCP use Q1 and Q2.csv')
    evcp.drop('Charging event', axis=1, inplace=True)
    evcp = evcp.fillna(" ")
    evcp.rename(columns={'Model': 'Charge'}, inplace=True)
    sns.countplot(x='Charge', data=evcp)
    plt.title('Charge')
    plt.show()


# Display final format of csv file
def read_csv_2():
    evcp = pd.read_csv('2019 EVCP use Q1 and Q2.csv')
    evcp.drop('Charging event', axis=1, inplace=True)
    evcp = evcp.fillna(" ")
    evcp.rename(columns={'Model': 'Charge'}, inplace=True)
    evcp.to_csv('2019 EVCP use Q1-Q2.csv')


# Read web data
def read_web():
    response = urllib2.urlopen('https://data.wa.gov/Transportation/Electric-
Vehicle-Population-Data/f6w7-q2d2/data')
    html_doc = response.read()
    soup = BeautifulSoup(html_doc, 'html.parser')
    evcp = soup.prettify()
    print(evcp)
    evcp_web = pd.read_csv('Electric_Vehicle_Population_Data.csv')
    print("Web data:\n", evcp_web)


# Drop 1st column from web data
def web_drop():
    response = urllib2.urlopen('https://data.wa.gov/Transportation/Electric-
Vehicle-Population-Data/f6w7-q2d2/data')
    html_doc = response.read()
    soup = BeautifulSoup(html_doc, 'html.parser')
    evcp = soup.prettify()
    evcp_web = pd.read_csv('Electric_Vehicle_Population_Data.csv')
    evcp_web.drop('VIN (1-10)', axis=1, inplace=True)
```

```
    print("Remove 1st column from web data:\n", evcp_web)


# Check, replace, and recheck the nulls from web data
def web_cpr_null():
    response = urllib2.urlopen('https://data.wa.gov/Transportation/Electric-
Vehicle-Population-Data/f6w7-q2d2/data')
    html_doc = response.read()
    soup = BeautifulSoup(html_doc, 'html.parser')
    evcp = soup.prettify()
    evcp_web = pd.read_csv('Electric_Vehicle_Population_Data.csv')
    evcp_web.drop('VIN (1-10)', axis=1, inplace=True)
    print("Display web data with null:\n", evcp_web.isnull())
    print("Display counts of null from web data:\n", evcp_web.isnull().sum())
    evcp_web = evcp_web.fillna(" ")
    print("Display web data with replaced nulls:\n", evcp_web)
    print("Display recounts of null from web data:\n",
evcp_web.isnull().sum())


# Rename Electric Vehicle Type column from web data
def web_rename_col():
    response = urllib2.urlopen('https://data.wa.gov/Transportation/Electric-
Vehicle-Population-Data/f6w7-q2d2/data')
    html_doc = response.read()
    soup = BeautifulSoup(html_doc, 'html.parser')
    evcp = soup.prettify()
    evcp_web = pd.read_csv('Electric_Vehicle_Population_Data.csv')
    evcp_web.drop('VIN (1-10)', axis=1, inplace=True)
    evcp_web = evcp_web.fillna(" ")
    evcp_web.rename(columns={'Electric Vehicle Type': 'Charge'},
inplace=True)
    print("Rename Electric Vehicle Type column from web data:\n", evcp_web)


# Display count plot of Electric Range column from web data
def web_showCountplot_ER():
    response = urllib2.urlopen('https://data.wa.gov/Transportation/Electric-
Vehicle-Population-Data/f6w7-q2d2/data')
    html_doc = response.read()
    soup = BeautifulSoup(html_doc, 'html.parser')
    evcp = soup.prettify()
    evcp_web = pd.read_csv('Electric_Vehicle_Population_Data.csv')
    evcp_web.drop('VIN (1-10)', axis=1, inplace=True)
    evcp_web = evcp_web.fillna(" ")
    evcp_web.rename(columns={'Electric Vehicle Type': 'Charge'},
inplace=True)
    sns.countplot(x='Electric Range', data=evcp_web)
    plt.title('ER')
    plt.show()


# Display count plot of County column from web data
def web_showCountplot_County():
    response = urllib2.urlopen('https://data.wa.gov/Transportation/Electric-
Vehicle-Population-Data/f6w7-q2d2/data')
    html_doc = response.read()
```

```
    soup = BeautifulSoup(html_doc, 'html.parser')
    evcp = soup.prettify()
    evcp_web = pd.read_csv('Electric_Vehicle_Population_Data.csv')
    evcp_web.drop('VIN (1-10)', axis=1, inplace=True)
    evcp_web = evcp_web.fillna(" ")
    evcp_web.rename(columns={'Electric Vehicle Type': 'Charge'},
inplace=True)
    sns.countplot(x='County', data=evcp_web)
    plt.title('county')
    plt.show()


# Display count plot of Charge column from web data
def web_showCountplot_Charge():
    response = urllib2.urlopen('https://data.wa.gov/Transportation/Electric-
Vehicle-Population-Data/f6w7-q2d2/data')
    html_doc = response.read()
    soup = BeautifulSoup(html_doc, 'html.parser')
    evcp = soup.prettify()
    evcp_web = pd.read_csv('Electric_Vehicle_Population_Data.csv')
    evcp_web.drop('VIN (1-10)', axis=1, inplace=True)
    evcp_web = evcp_web.fillna(" ")
    evcp_web.rename(columns={'Electric Vehicle Type': 'Charge'},
inplace=True)
    sns.countplot(x='Charge', data=evcp_web)
    plt.title('charge')
    plt.show()


# Display final format of web data
def read_web_2():
    response = urllib2.urlopen('https://data.wa.gov/Transportation/Electric-
Vehicle-Population-Data/f6w7-q2d2/data')
    html_doc = response.read()
    soup = BeautifulSoup(html_doc, 'html.parser')
    evcp = soup.prettify()
    evcp_web = pd.read_csv('Electric_Vehicle_Population_Data.csv')
    evcp_web.drop('VIN (1-10)', axis=1, inplace=True)
    evcp_web = evcp_web.fillna(" ")
    evcp_web.rename(columns={'Electric Vehicle Type': 'Charge'},
inplace=True)
    evcp_web.to_csv('Electric Vehicle Pop Data.csv')


# Read api data
def read_api():
    res = requests.get('https://data.wa.gov/resource/rpr4-cgyd.json')
    print(res)
    evcp_api =
pd.read_csv('Electric_Vehicle_Title_and_Registration_Activity.csv')
    print("API data:\n", evcp_api)


# Drop multiple columns from api data
def api_drop():
    res = requests.get('https://data.wa.gov/resource/rpr4-cgyd.json')
    evcp_api =
```

```
pd.read_csv('Electric_Vehicle_Title_and_Registration_Activity.csv')
    evcp_api.drop(['VIN (1-10)', 'Sale Price', 'DOL Transaction Date', '2015
HB 2778 Exemption Eligibility',
                   'Sale Date', '2019 HB 2042 Clean Alternative Fuel Vehicle
(CAFV) Eligibility',
                   'Meets 2019 HB 2042 Electric Range Requirement', 'Meets
2019 HB 2042 Sale Date Requirement',
                   'Meets 2019 HB 2042 Sale Price/Value Requirement',
'Odometer Reading', 'Odometer Code'],
                  axis=1, inplace=True)
    print("Removal of multiple columns from api data:\n", evcp_api)


# Check, replace, and recheck the nulls from api data
def api_cpr_null():
    res = requests.get('https://data.wa.gov/resource/rpr4-cgyd.json')
    evcp_api =
pd.read_csv('Electric_Vehicle_Title_and_Registration_Activity.csv')
    evcp_api.drop(['VIN (1-10)', 'Sale Price', 'DOL Transaction Date', '2015
HB 2778 Exemption Eligibility',
                   'Sale Date', '2019 HB 2042 Clean Alternative Fuel Vehicle
(CAFV) Eligibility',
                   'Meets 2019 HB 2042 Electric Range Requirement', 'Meets
2019 HB 2042 Sale Date Requirement',
                   'Meets 2019 HB 2042 Sale Price/Value Requirement',
'Odometer Reading', 'Odometer Code'],
                  axis=1, inplace=True)
    print("Display api data with null:\n", evcp_api.isnull())
    print("Display counts of null from api data:\n", evcp_api.isnull().sum())
    evcp_api = evcp_api.fillna(" ")
    print("Display api data with replaced nulls:\n", evcp_api)
    print("Display recounts of null from api data:\n",
evcp_api.isnull().sum())


# Rename Clean Alternative Fuel Vehicle Type column from api data
def api_rename_col():
    res = requests.get('https://data.wa.gov/resource/rpr4-cgyd.json')
    evcp_api =
pd.read_csv('Electric_Vehicle_Title_and_Registration_Activity.csv')
    evcp_api.drop(['VIN (1-10)', 'Sale Price', 'DOL Transaction Date', '2015
HB 2778 Exemption Eligibility',
                   'Sale Date', '2019 HB 2042 Clean Alternative Fuel Vehicle
(CAFV) Eligibility',
                   'Meets 2019 HB 2042 Electric Range Requirement', 'Meets
2019 HB 2042 Sale Date Requirement',
                   'Meets 2019 HB 2042 Sale Price/Value Requirement',
'Odometer Reading', 'Odometer Code'],
                  axis=1, inplace=True)
    evcp_api = evcp_api.fillna(" ")
    evcp_api.rename(columns={'Clean Alternative Fuel Vehicle Type':
'Charge'}, inplace=True)
    print("Rename Clean Alternative Fuel Vehicle Type column from api
data:\n", evcp_api)


# Display count plot of New or Used Vehicle column from api data
```

```
def api_showCountplot_NU():
    res = requests.get('https://data.wa.gov/resource/rpr4-cgyd.json')
    evcp_api =
pd.read_csv('Electric_Vehicle_Title_and_Registration_Activity.csv')
    evcp_api.drop(['VIN (1-10)', 'Sale Price', 'DOL Transaction Date', '2015
HB 2778 Exemption Eligibility',
                   'Sale Date', '2019 HB 2042 Clean Alternative Fuel Vehicle
(CAFV) Eligibility',
                   'Meets 2019 HB 2042 Electric Range Requirement', 'Meets
2019 HB 2042 Sale Date Requirement',
                   'Meets 2019 HB 2042 Sale Price/Value Requirement',
'Odometer Reading', 'Odometer Code'],
                  axis=1, inplace=True)
    evcp_api = evcp_api.fillna(" ")
    evcp_api.rename(columns={'Clean Alternative Fuel Vehicle Type':
'Charge'}, inplace=True)
    sns.countplot(x='New or Used Vehicle', data=evcp_api)
    plt.title('N/U')
    plt.show()


# Display count plot of Electric Vehicle Fee Paid column from api data
def api_showCountplot_Fee():
    res = requests.get('https://data.wa.gov/resource/rpr4-cgyd.json')
    evcp_api =
pd.read_csv('Electric_Vehicle_Title_and_Registration_Activity.csv')
    evcp_api.drop(['VIN (1-10)', 'Sale Price', 'DOL Transaction Date', '2015
HB 2778 Exemption Eligibility',
                   'Sale Date', '2019 HB 2042 Clean Alternative Fuel Vehicle
(CAFV) Eligibility',
                   'Meets 2019 HB 2042 Electric Range Requirement', 'Meets
2019 HB 2042 Sale Date Requirement',
                   'Meets 2019 HB 2042 Sale Price/Value Requirement',
'Odometer Reading', 'Odometer Code'],
                  axis=1, inplace=True)
    evcp_api = evcp_api.fillna(" ")
    evcp_api.rename(columns={'Clean Alternative Fuel Vehicle Type':
'Charge'}, inplace=True)
    sns.countplot(x='Electric Vehicle Fee Paid', data=evcp_api)
    plt.title('Fee')
    plt.show()


# Display count plot of Charge column from api data
def api_showCountplot_Charge():
    res = requests.get('https://data.wa.gov/resource/rpr4-cgyd.json')
    evcp_api =
pd.read_csv('Electric_Vehicle_Title_and_Registration_Activity.csv')
    evcp_api.drop(['VIN (1-10)', 'Sale Price', 'DOL Transaction Date', '2015
HB 2778 Exemption Eligibility',
                   'Sale Date', '2019 HB 2042 Clean Alternative Fuel Vehicle
(CAFV) Eligibility',
                   'Meets 2019 HB 2042 Electric Range Requirement', 'Meets
2019 HB 2042 Sale Date Requirement',
                   'Meets 2019 HB 2042 Sale Price/Value Requirement',
'Odometer Reading', 'Odometer Code'],
                  axis=1, inplace=True)
```

```
    evcp_api = evcp_api.fillna(" ")
    evcp_api.rename(columns={'Clean Alternative Fuel Vehicle Type':
'Charge'}, inplace=True)
    sns.countplot(x='Charge', data=evcp_api)
    plt.title('charge')
    plt.show()


# Display final format of api data
def read_api_2():
    res = requests.get('https://data.wa.gov/resource/rpr4-cgyd.json')
    evcp_api =
pd.read_csv('Electric_Vehicle_Title_and_Registration_Activity.csv')
    evcp_api.drop(['VIN (1-10)', 'Sale Price', 'DOL Transaction Date', '2015
HB 2778 Exemption Eligibility',
                   'Sale Date', '2019 HB 2042 Clean Alternative Fuel Vehicle
(CAFV) Eligibility',
                   'Meets 2019 HB 2042 Electric Range Requirement', 'Meets
2019 HB 2042 Sale Date Requirement',
                   'Meets 2019 HB 2042 Sale Price/Value Requirement',
'Odometer Reading', 'Odometer Code'],
                  axis=1, inplace=True)
    evcp_api = evcp_api.fillna(" ")
    evcp_api.rename(columns={'Clean Alternative Fuel Vehicle Type':
'Charge'}, inplace=True)
    evcp_api.to_csv('Electric Vehicle Title-Registration Activity.csv')


# Read db file
def read_db_file():
    conn = sqlite3.connect("EVCP")
    cur = conn.cursor()
    cur.execute("SELECT UserID, CPID, Connector, StartDate, StartTime,
EndDate, EndTime, TotalkWh, Site, Charge FROM "
                "Q1Q2")
    table = cur.fetchall()

    for i in table:
        print(i)

    cur.execute("SELECT County, City, State, ZIPCode, ModelYear, Make, Model,
Charge, CAFV, ElectricRange, BaseMSRP, "
                "LegislativeDistrict, DOLVehicleID, VehicleLocation FROM
Pop")
    table2 = cur.fetchall()

    for i in table2:
        print(i)

    cur.execute("SELECT Charge, ModelYear, Make, Model, NewUsed,
TransactionType, TransactionYear, FeePaid, County, "
                "City, Zip, ElectricRange, BaseMSRP, PrimaryUse, State,
DOLVehicleID, LegislativeDistrict FROM "
                "TitleRegistration")
    table3 = cur.fetchall()

    for i in table3:
```

```
        print(i)


# Combine tables from db file
def db_merge():
    conn = sqlite3.connect("EVCP")
    cur = conn.cursor()
    cur.execute("SELECT UserID, CPID, Connector, StartDate, StartTime,
EndDate, EndTime, TotalkWh, Site, Charge FROM "
                "Q1Q2")
    table = cur.fetchall()
    df = pd.read_csv('2019 EVCP use Q1-Q2.csv')
    print(df)
    cur.execute("SELECT County, City, State, ZIPCode, ModelYear, Make, Model,
Charge, CAFV, ElectricRange, BaseMSRP, "
                "LegislativeDistrict, DOLVehicleID, VehicleLocation FROM
Pop")
    table2 = cur.fetchall()
    df2 = pd.read_csv('Electric Vehicle Pop Data.csv')
    print(df2)
    cur.execute("SELECT Charge, ModelYear, Make, Model, NewUsed,
TransactionType, TransactionYear, FeePaid, County, "
                "City, Zip, ElectricRange, BaseMSRP, PrimaryUse, State,
DOLVehicleID, LegislativeDistrict FROM "
                "TitleRegistration")
    table3 = cur.fetchall()
    df3 = pd.read_csv('Electric Vehicle Title-Registration Activity.csv')
    print(df3)
    df_final = df.merge(df2, how='outer') \
        .merge(df3, how='outer')
    print("Database shape: ", df_final.shape)


# Plots of db file
def db_showPlots():
    conn = sqlite3.connect("EVCP")
    cur = conn.cursor()
    cur.execute("SELECT UserID, CPID, Connector, StartDate, StartTime,
EndDate, EndTime, TotalkWh, Site, Charge FROM "
                "Q1Q2")
    table = cur.fetchall()
    df = pd.read_csv('2019 EVCP use Q1-Q2.csv')
    cur.execute("SELECT County, City, State, ZIPCode, ModelYear, Make, Model,
Charge, CAFV, ElectricRange, BaseMSRP, "
                "LegislativeDistrict, DOLVehicleID, VehicleLocation FROM
Pop")
    table2 = cur.fetchall()
    df2 = pd.read_csv('Electric Vehicle Pop Data.csv')
    cur.execute("SELECT Charge, ModelYear, Make, Model, NewUsed,
TransactionType, TransactionYear, FeePaid, County, "
                "City, Zip, ElectricRange, BaseMSRP, PrimaryUse, State,
DOLVehicleID, LegislativeDistrict FROM "
                "TitleRegistration")
    table3 = cur.fetchall()
    df3 = pd.read_csv('Electric Vehicle Title-Registration Activity.csv')
    df_final = df.merge(df2, how='outer') \
        .merge(df3, how='outer')
```

```
    db = pd.read_csv('EVCP.csv')
    print(db)
    sns.countplot(x='Charge', data=db)
    plt.title('charge')
    plt.show()
    sns.countplot(x='Electric Range', data=db)
    plt.title('ER')
    plt.show()
    sns.countplot(x='Base MSRP', data=db)
    plt.title('b MSRP')
    plt.show()
    sns.countplot(y='Make', data=db)
    plt.title('make')
    plt.show()
    sns.countplot(x='Model', data=db)
    plt.title('model')
    plt.show()


if __name__ == "__main__":
    read_csv_file()
    csv_drop()
    csv_cpr_null()
    csv_rename_col()
    csv_showCountplot_kWh()
    csv_showCountplot_Site()
    csv_showCountplot_Charge()
    read_web()
    web_drop()
    web_cpr_null()
    web_rename_col()
    web_showCountplot_ER()
    web_showCountplot_County()
    web_showCountplot_Charge()
    read_api()
    api_drop()
    api_cpr_null()
    api_rename_col()
    api_showCountplot_NU()
    api_showCountplot_Fee()
    api_showCountplot_Charge()
    read_db_file()
    db_merge()
    db_showPlots()
```

Output:

Csv data:

	Charging event	...	Model
0	8124494	...	APT 7kW Dual Outlet
1	8124522	...	APT 7kW Dual Outlet
2	8124828	...	APT 7kW Dual Outlet
3	8124987	...	APT 7kW Dual Outlet
4	8125100	...	APT 7kW Dual Outlet
...	...	...	...
3401	8702065	...	APT Triple Rapid Charger
3402	8702103	...	APT Triple Rapid Charger
3403	8702255	...	APT Triple Rapid Charger
3404	8702426	...	APT 7kW Dual Outlet
3405	8702978	...	APT 7kW Dual Outlet

[3406 rows x 11 columns]

Remove 1st column from csv data:

	User ID	CP ID	...	Site	Model
0	User 406	70204	...	Woodhouse Lane Car Park	APT 7kW Dual Outlet
1	User 546	70204	...	Woodhouse Lane Car Park	APT 7kW Dual Outlet
2	User 279	80085	...	Temple Green Park and Ride	APT 7kW Dual Outlet
3	User 399	70202	...	Woodhouse Lane Car Park	APT 7kW Dual Outlet
4	User 771	70202	...	Woodhouse Lane Car Park	APT 7kW Dual Outlet
...	...	...	...	...	...
3401	User 131	80164	...	Elland Road Park and Ride	APT Triple Rapid Charger
3402	User 573	80164	...	Elland Road Park and Ride	APT Triple Rapid Charger
3403	User 418	80164	...	Elland Road Park and Ride	APT Triple Rapid Charger
3404	User 306	70204	...	Woodhouse Lane Car Park	APT 7kW Dual Outlet

3405 User 308 70204 ... Woodhouse Lane Car Park APT 7kW Dual Outlet

[3406 rows x 10 columns]

Display csv data with null:

	User ID	CP ID	Connector	Start Date	...	End Time	Total kWh	Site	Model
0	False	False	False	False	...	False	False	False	False
1	False	False	False	False	...	False	False	False	False
2	False	False	False	False	...	False	False	False	False
3	False	False	False	False	...	False	False	False	False
4	False	False	False	False	...	False	False	False	False
...	...	...	...	...	...	...	...	...	...
3401	False	False	False	False	...	True	True	False	False
3402	False	False	False	False	...	False	False	False	False
3403	False	False	False	False	...	False	False	False	False
3404	False	False	False	False	...	False	False	False	False
3405	False	False	False	False	...	False	False	False	False

[3406 rows x 10 columns]

Display counts of null from csv data:

User ID	0
CP ID	0
Connector	0
Start Date	0
Start Time	0
End Date	52
End Time	52
Total kWh	52
Site	0

Model 0

dtype: int64

Display csv data with replaced nulls:

	User ID	CP ID	...	Site	Model
0	User 406	70204	...	Woodhouse Lane Car Park	APT 7kW Dual Outlet
1	User 546	70204	...	Woodhouse Lane Car Park	APT 7kW Dual Outlet
2	User 279	80085	...	Temple Green Park and Ride	APT 7kW Dual Outlet
3	User 399	70202	...	Woodhouse Lane Car Park	APT 7kW Dual Outlet
4	User 771	70202	...	Woodhouse Lane Car Park	APT 7kW Dual Outlet
...	...	...	...	...	...
3401	User 131	80164	...	Elland Road Park and Ride	APT Triple Rapid Charger
3402	User 573	80164	...	Elland Road Park and Ride	APT Triple Rapid Charger
3403	User 418	80164	...	Elland Road Park and Ride	APT Triple Rapid Charger
3404	User 306	70204	...	Woodhouse Lane Car Park	APT 7kW Dual Outlet
3405	User 308	70204	...	Woodhouse Lane Car Park	APT 7kW Dual Outlet

[3406 rows x 10 columns]

Display recounts of null from csv data:

User ID 0

CP ID 0

Connector 0

Start Date 0

Start Time 0

End Date 0

End Time 0

Total kWh 0

Site 0

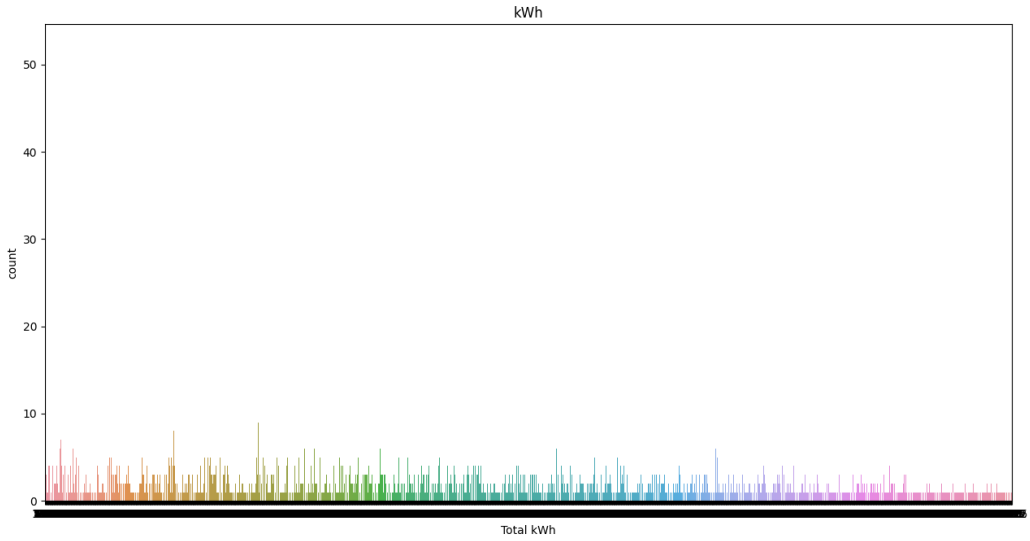
Model 0

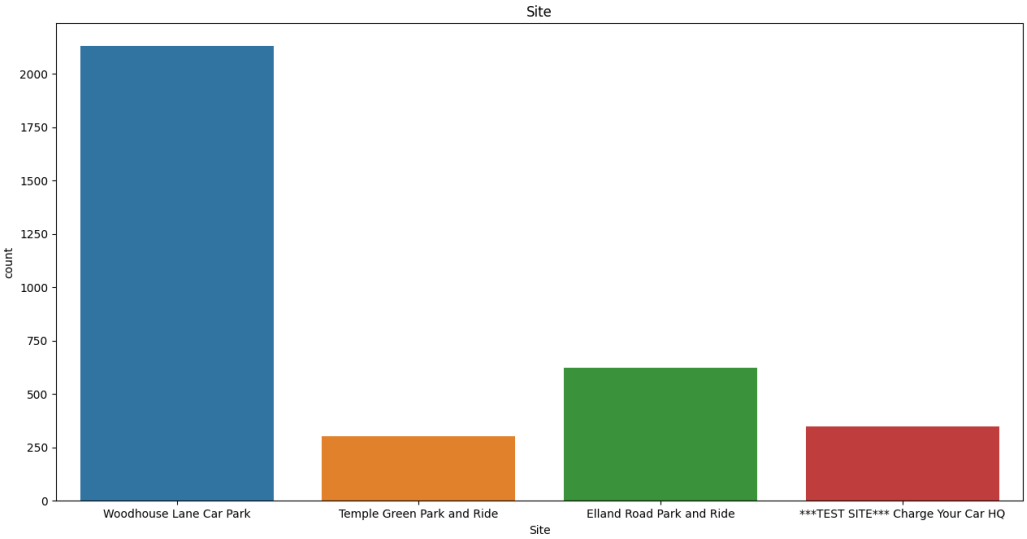
dtype: int64

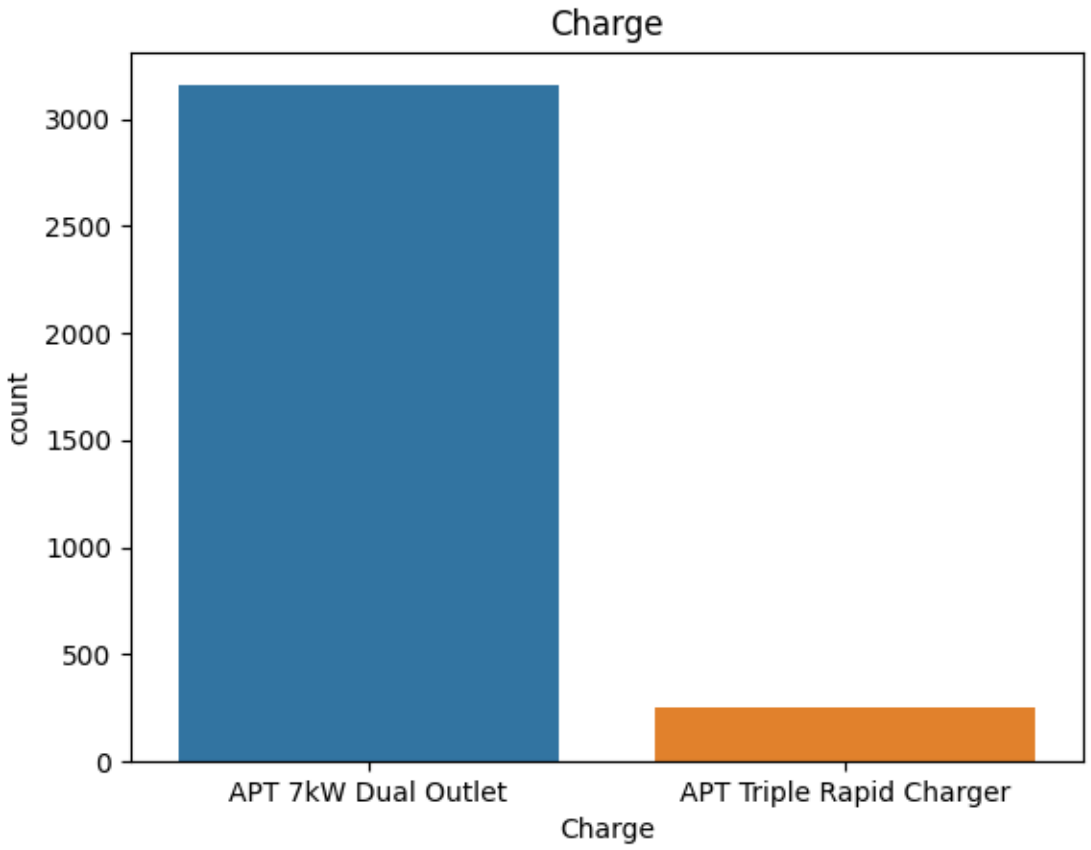
Rename Model column from csv data:

	User ID	CP ID	...	Site	Charge
0	User 406	70204	...	Woodhouse Lane Car Park	APT 7kW Dual Outlet
1	User 546	70204	...	Woodhouse Lane Car Park	APT 7kW Dual Outlet
2	User 279	80085	...	Temple Green Park and Ride	APT 7kW Dual Outlet
3	User 399	70202	...	Woodhouse Lane Car Park	APT 7kW Dual Outlet
4	User 771	70202	...	Woodhouse Lane Car Park	APT 7kW Dual Outlet
...	...	...	...	...	...
3401	User 131	80164	...	Elland Road Park and Ride	APT Triple Rapid Charger
3402	User 573	80164	...	Elland Road Park and Ride	APT Triple Rapid Charger
3403	User 418	80164	...	Elland Road Park and Ride	APT Triple Rapid Charger
3404	User 306	70204	...	Woodhouse Lane Car Park	APT 7kW Dual Outlet
3405	User 308	70204	...	Woodhouse Lane Car Park	APT 7kW Dual Outlet

[3406 rows x 10 columns]







Web data:

VIN (1-10) ... Vehicle Location

0 3FA6P0SU3L ... POINT (-122.11667400000002 47.363112)

1 5YJYGDEE6L ... POINT (-122.137386 47.444808)

2 KNDCC3LG6L ... POINT (-122.215501 47.476576)

3 1N4AZ0CP5D ... POINT (-122.31336800000001 47.54411)

4 5YJSA1H22E ... POINT (-122.297534 47.685291)

... ...

63850 YV4BC0ZX1H ... POINT (-117.50543600000002 47.633834)

63851 5YJ3E1EC5L ... POINT (-122.30033 47.585339)

63852 KNDCE3LG3K ... POINT (-122.97996899999998 47.078241)

63853 5YJ3E1EBXJ ... POINT (-122.227947 47.565443)

63854 5YJ3E1EA1L ... POINT (-122.132064 47.494834)

[63855 rows x 15 columns]

Remove 1st column from web data:

County ... Vehicle Location

0 King ... POINT (-122.11667400000002 47.363112)

1 King ... POINT (-122.137386 47.444808)

2 King ... POINT (-122.215501 47.476576)

3 King ... POINT (-122.31336800000001 47.54411)

4 King ... POINT (-122.297534 47.685291)

... ...

63850 Spokane ... POINT (-117.50543600000002 47.633834)

63851 King ... POINT (-122.30033 47.585339)

63852 Thurston ... POINT (-122.97996899999998 47.078241)

63853 King ... POINT (-122.227947 47.565443)

63854 King ... POINT (-122.132064 47.494834)

[63855 rows x 14 columns]

Display web data with null:

	County	City	...	DOL Vehicle ID	Vehicle Location
0	False	False	...	False	False
1	False	False	...	False	False
2	False	False	...	False	False
3	False	False	...	False	False
4	False	False	...	False	False
...	...	...	...	...	...
63850	False	False	...	False	False
63851	False	False	...	False	False
63852	False	False	...	False	False
63853	False	False	...	False	False
63854	False	False	...	False	False

[63855 rows x 14 columns]

Display counts of null from web data:

County	2
City	0
State	0
ZIP 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	43

DOL Vehicle ID 0

Vehicle Location 2

dtype: int64

Display web data with replaced nulls:

	County	...	Vehicle Location
0	King	...	POINT (-122.11667400000002 47.363112)
1	King	...	POINT (-122.137386 47.444808)
2	King	...	POINT (-122.215501 47.476576)
3	King	...	POINT (-122.31336800000001 47.54411)
4	King	...	POINT (-122.297534 47.685291)
...	...	...	...
63850	Spokane	...	POINT (-117.50543600000002 47.633834)
63851	King	...	POINT (-122.30033 47.585339)
63852	Thurston	...	POINT (-122.97996899999998 47.078241)
63853	King	...	POINT (-122.227947 47.565443)
63854	King	...	POINT (-122.132064 47.494834)

[63855 rows x 14 columns]

Display recounts of null from web data:

County 0

City 0

State 0

ZIP 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

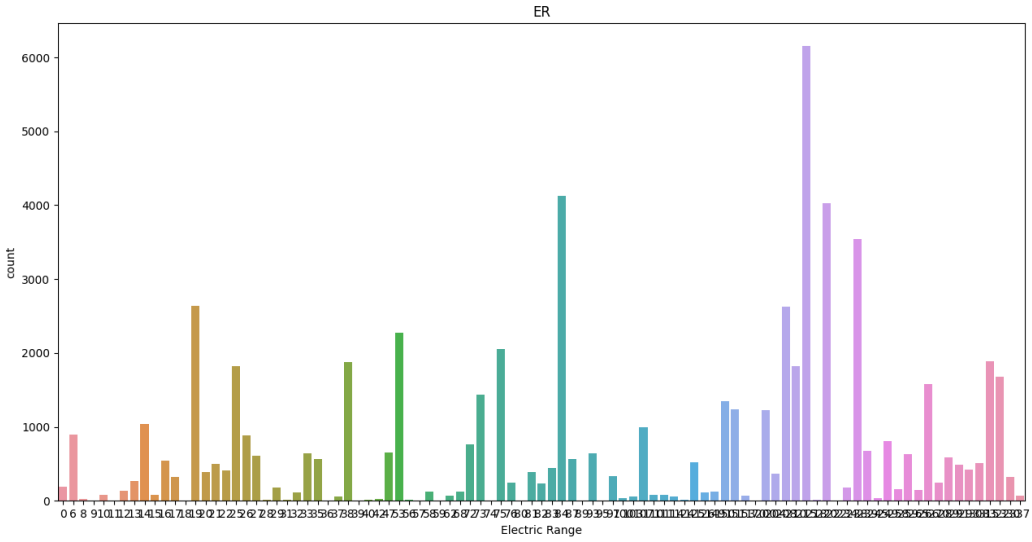
Vehicle Location 0

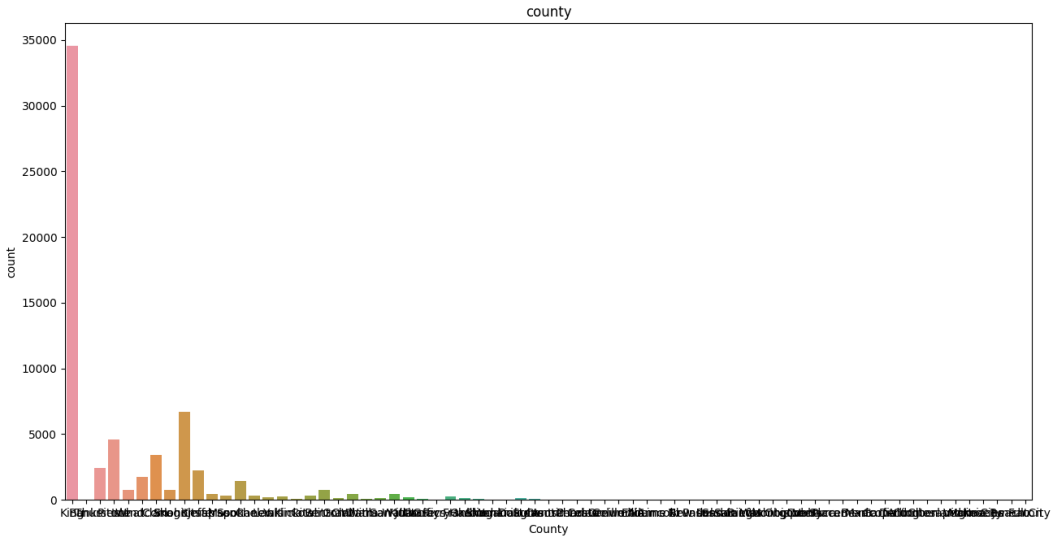
dtype: int64

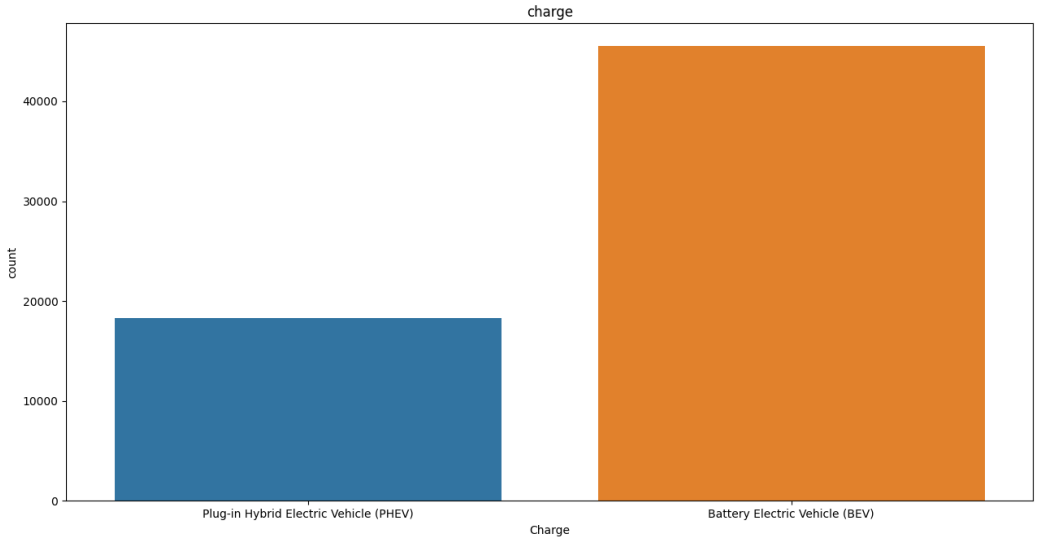
Rename Electric Vehicle Type column from web data:

	County	...	Vehicle Location
0	King	...	POINT (-122.11667400000002 47.363112)
1	King	...	POINT (-122.137386 47.444808)
2	King	...	POINT (-122.215501 47.476576)
3	King	...	POINT (-122.31336800000001 47.54411)
4	King	...	POINT (-122.297534 47.685291)
...	...	...	...
63850	Spokane	...	POINT (-117.50543600000002 47.633834)
63851	King	...	POINT (-122.30033 47.585339)
63852	Thurston	...	POINT (-122.97996899999998 47.078241)
63853	King	...	POINT (-122.227947 47.565443)
63854	King	...	POINT (-122.132064 47.494834)

[63855 rows x 14 columns]


ER
count
Electric Range


county
count
County



<Response [200]>

API data:

	Clean Alternative Fuel Vehicle Type	...	Odometer Code
0	Battery Electric Vehicle (BEV)	...	Odometer reading is not collected at time of r...
1	Battery Electric Vehicle (BEV)	...	Odometer reading is not collected at time of r...
2	Plug-in Hybrid Electric Vehicle	...	Odometer reading is not collected at time of r...
3	Plug-in Hybrid Electric Vehicle	...	Odometer reading is not collected at time of r...
4	Battery Electric Vehicle (BEV)	...	Odometer reading is not collected at time of r...
...	...	...	...
366734	Plug-in Hybrid Electric Vehicle	...	Odometer reading is not collected at time of r...
366735	Plug-in Hybrid Electric Vehicle	...	Actual Mileage
366736	Plug-in Hybrid Electric Vehicle	...	Odometer reading is not collected at time of r...
366737	Plug-in Hybrid Electric Vehicle	...	Actual Mileage
366738	Battery Electric Vehicle (BEV)	...	Odometer reading is not collected at time of r...

[366739 rows x 28 columns]

Removal of multiple columns from api data:

	Clean Alternative Fuel Vehicle Type	...	Legislative District

0	Battery Electric Vehicle (BEV)	...	41.0
1	Battery Electric Vehicle (BEV)	...	24.0
2	Plug-in Hybrid Electric Vehicle	...	37.0
3	Plug-in Hybrid Electric Vehicle	...	24.0
4	Battery Electric Vehicle (BEV)	...	39.0
...	...	...	...
366734	Plug-in Hybrid Electric Vehicle	...	40.0
366735	Plug-in Hybrid Electric Vehicle	...	40.0
366736	Plug-in Hybrid Electric Vehicle	...	24.0
366737	Plug-in Hybrid Electric Vehicle	...	24.0
366738	Battery Electric Vehicle (BEV)	...	21.0

[366739 rows x 17 columns]

Display api data with null:

	Clean Alternative Fuel Vehicle Type	...	Legislative District
0	False	...	False
1	False	...	False
2	False	...	False
3	False	...	False
4	False	...	False
...	...	...	...
366734	False	...	False
366735	False	...	False
366736	False	...	False
366737	False	...	False
366738	False	...	False

[366739 rows x 17 columns]

Display counts of null from api data:

```
Clean Alternative Fuel Vehicle Type      0
Model Year                               0
Make                                     0
Model                                    0
New or Used Vehicle                      0
Transaction Type                         0
Transaction Year                         0
Electric Vehicle Fee Paid                0
County                                  12
City                                    91
Zip                                      4
Electric Range                           0
Base MSRP                                0
Vehicle Primary Use                      0
State of Residence                       4
DOL Vehicle ID                           0
Legislative District                   465
dtype: int64
```

Display api data with replaced nulls:

```
       Clean Alternative Fuel Vehicle Type  ...  Legislative District
0          Battery Electric Vehicle (BEV)  ...                  41.0
1          Battery Electric Vehicle (BEV)  ...                  24.0
2         Plug-in Hybrid Electric Vehicle  ...                  37.0
3         Plug-in Hybrid Electric Vehicle  ...                  24.0
4          Battery Electric Vehicle (BEV)  ...                  39.0
...                                   ...  ...                   ...
366734    Plug-in Hybrid Electric Vehicle  ...                  40.0
```

366735 Plug-in Hybrid Electric Vehicle ... 40.0

366736 Plug-in Hybrid Electric Vehicle ... 24.0

366737 Plug-in Hybrid Electric Vehicle ... 24.0

366738 Battery Electric Vehicle (BEV) ... 21.0

[366739 rows x 17 columns]

Display recounts of null from api data:

Clean Alternative Fuel Vehicle Type 0

Model Year 0

Make 0

Model 0

New or Used Vehicle 0

Transaction Type 0

Transaction Year 0

Electric Vehicle Fee Paid 0

County 0

City 0

Zip 0

Electric Range 0

Base MSRP 0

Vehicle Primary Use 0

State of Residence 0

DOL Vehicle ID 0

Legislative District 0

dtype: int64

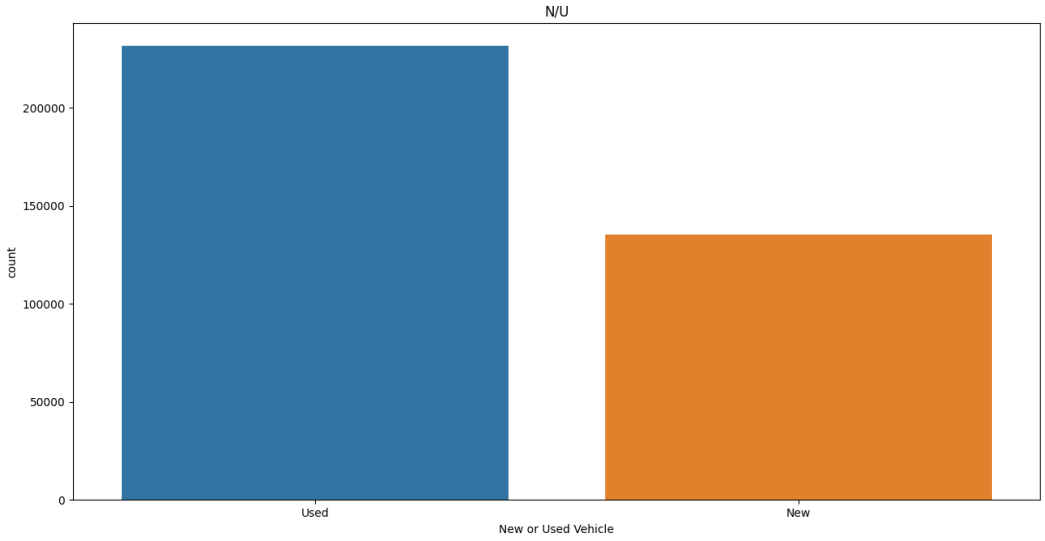
Rename Clean Alternative Fuel Vehicle Type column from api data:

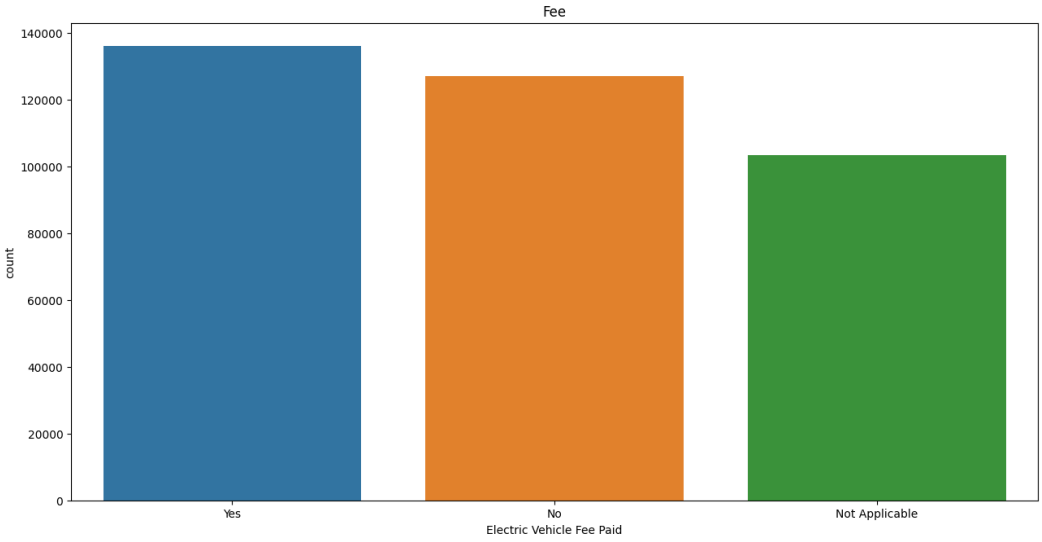
Charge ... Legislative District

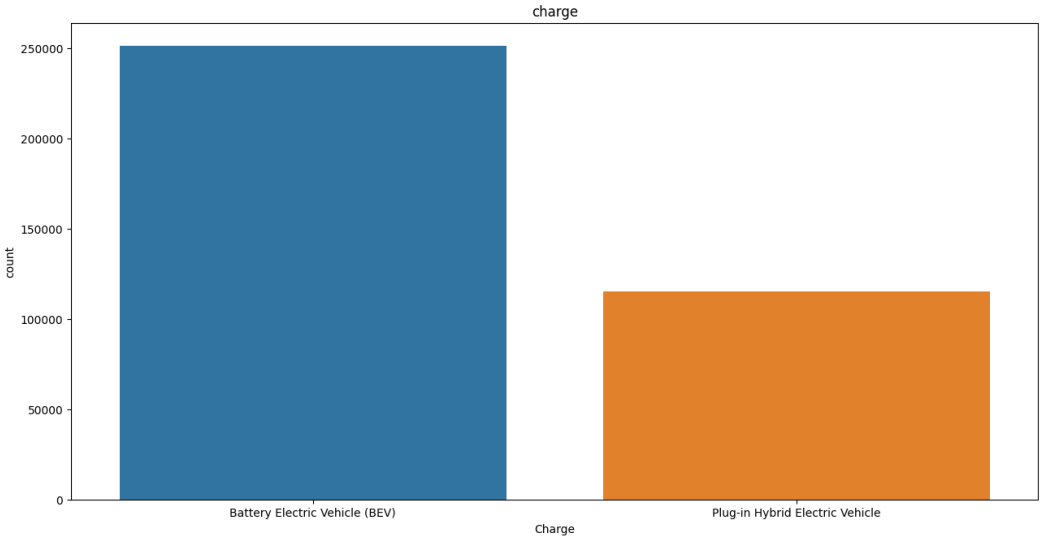
0 Battery Electric Vehicle (BEV) ... 41.0

1	Battery Electric Vehicle (BEV)	...	24.0
2	Plug-in Hybrid Electric Vehicle	...	37.0
3	Plug-in Hybrid Electric Vehicle	...	24.0
4	Battery Electric Vehicle (BEV)	...	39.0
...	...	...	...
366734	Plug-in Hybrid Electric Vehicle	...	40.0
366735	Plug-in Hybrid Electric Vehicle	...	40.0
366736	Plug-in Hybrid Electric Vehicle	...	24.0
366737	Plug-in Hybrid Electric Vehicle	...	24.0
366738	Battery Electric Vehicle (BEV)	...	21.0

[366739 rows x 17 columns]







	Unnamed: 0	...	Charge
0	0	...	APT 7kW Dual Outlet
1	1	...	APT 7kW Dual Outlet
2	2	...	APT 7kW Dual Outlet
3	3	...	APT 7kW Dual Outlet
4	4	...	APT 7kW Dual Outlet
...	...	...	...

3401 3401 ... APT Triple Rapid Charger

3402 3402 ... APT Triple Rapid Charger

3403 3403 ... APT Triple Rapid Charger

3404 3404 ... APT 7kW Dual Outlet

3405 3405 ... APT 7kW Dual Outlet

[3406 rows x 11 columns]

	Unnamed: 0	...	Vehicle Location
0	0	...	POINT (-122.11667400000002 47.363112)
1	1	...	POINT (-122.137386 47.444808)
2	2	...	POINT (-122.215501 47.476576)
3	3	...	POINT (-122.31336800000001 47.54411)
4	4	...	POINT (-122.297534 47.685291)
...	...	...	...
63850	63850	...	POINT (-117.50543600000002 47.633834)
63851	63851	...	POINT (-122.30033 47.585339)
63852	63852	...	POINT (-122.97996899999998 47.078241)
63853	63853	...	POINT (-122.227947 47.565443)
63854	63854	...	POINT (-122.132064 47.494834)

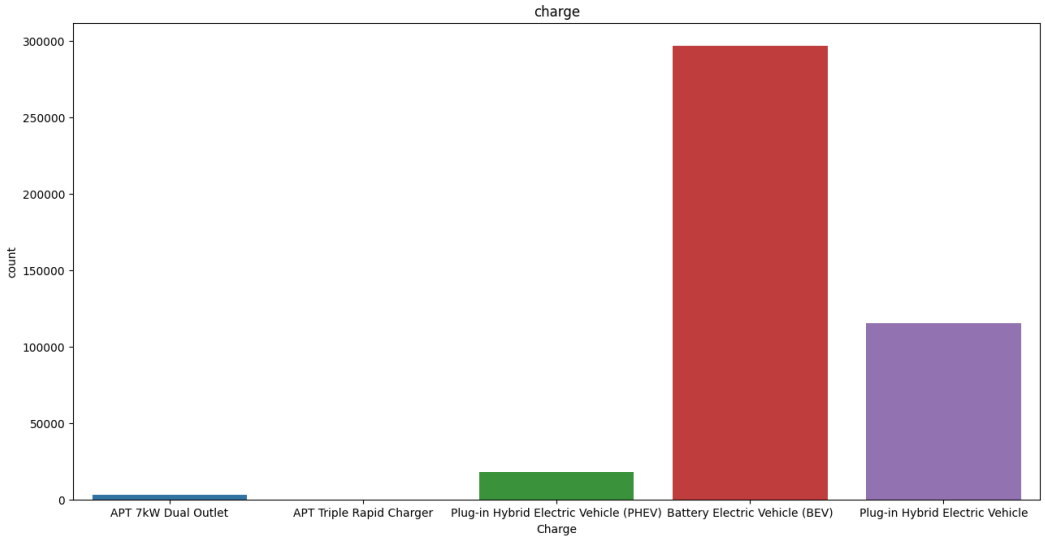
[63855 rows x 15 columns]

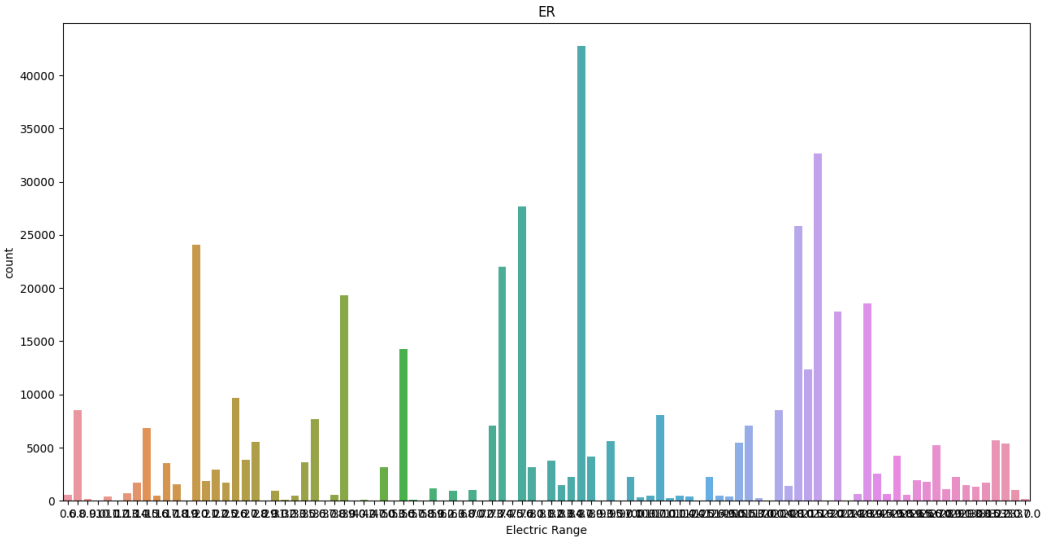
	Unnamed: 0	...	Legislative District
0	0	...	41.0
1	1	...	24.0
2	2	...	37.0
3	3	...	24.0
4	4	...	39.0
...	...	...	...

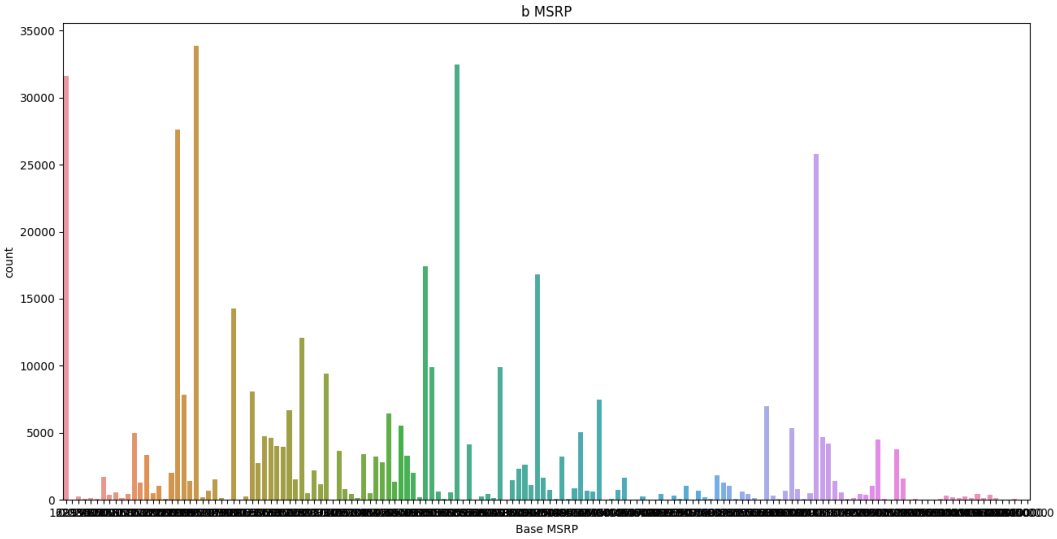
```
366734  366734  ...          40.0
366735  366735  ...          40.0
366736  366736  ...          24.0
366737  366737  ...          24.0
366738  366738  ...          21.0

[366739 rows x 18 columns]
Database shape:  (434000, 31)
Database:
        Unnamed: 0   User ID  ...  Vehicle Primary Use  State of Residence
0                0  User 406  ...                  NaN                 NaN
1                1  User 546  ...                  NaN                 NaN
2                2  User 279  ...                  NaN                 NaN
3                3  User 399  ...                  NaN                 NaN
4                4  User 771  ...                  NaN                 NaN
...            ...       ...  ...                  ...                 ...
433995      366734       NaN  ...            Passenger                  WA
433996      366735       NaN  ...            Passenger                  WA
433997      366736       NaN  ...            Passenger                  WA
433998      366737       NaN  ...            Passenger                  WA
433999      366738       NaN  ...            Passenger                  WA

[434000 rows x 31 columns]
        Unnamed: 0   User ID  ...    Zip  Vehicle Primary Use
0              0.0  User 406  ...  98056            Passenger
1              1.0  User 546  ...  98382            Passenger
2              2.0  User 279  ...  98144            Passenger
3              3.0  User 399  ...  98368            Passenger
```

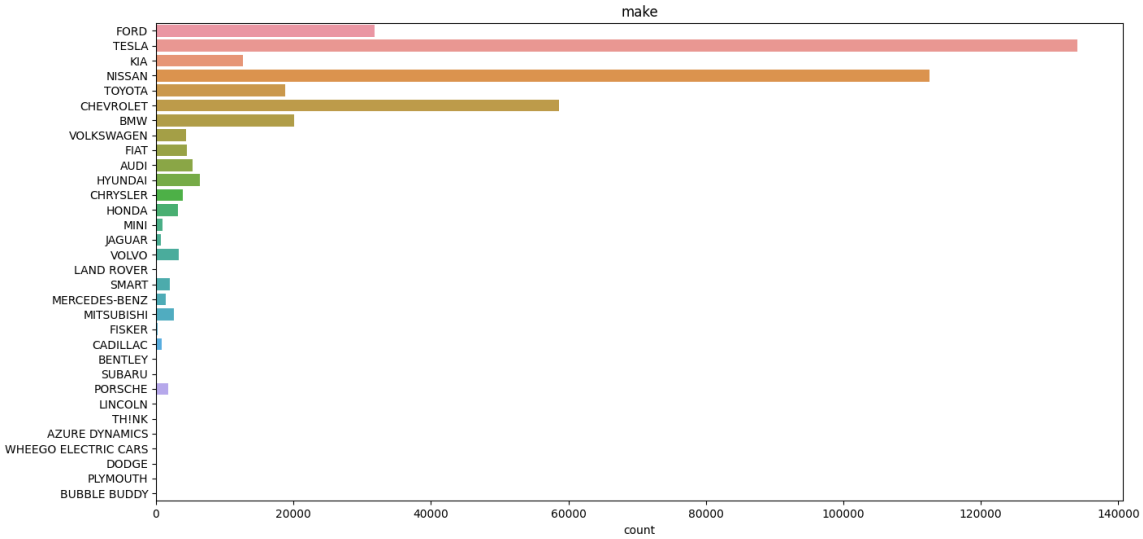
4	4.0	User	771	...	98290	Passenger
...	...	...	...	...	...	...
433995	NaN	NaN	...	NaN		NaN
433996	NaN	NaN	...	NaN		NaN
433997	NaN	NaN	...	NaN		NaN
433998	NaN	NaN	...	NaN		NaN
433999	NaN	NaN	...	NaN		NaN

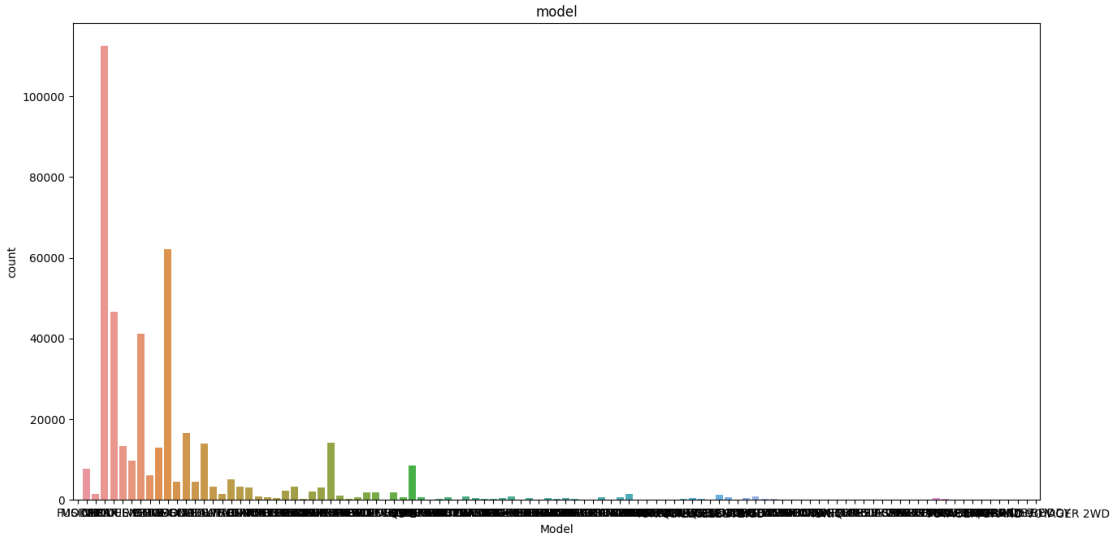
[434000 rows x 30 columns]




ER
count
Electric Range


b MSRP
count
Base MSRP


make
FORD
TESLA
KIA
NISSAN
TOYOTA
CHEVROLET
BMW
VOLKSWAGEN
FIAT
AUDI
HYUNDAI
CHRYSLER
HONDA
MINI
JAGUAR
VOLVO
LAND ROVER
SMART
MERCEDES-BENZ
MITSUBISHI
FISKER
CADILLAC
BENTLEY
SUBARU
PORSCHE
LINCOLN
THINK
AZURE DYNAMICS
WHEEGO ELECTRIC CARS
DODGE
PLYMOUTH
BUBBLE BUDDY
0
20000
40000
60000
80000
100000
120000
140000
count


model
100000
80000
60000
40000
20000
0
count
Model