In [1]:

```python
# importing libraries
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

In [2]:

```python
# importing file
df=pd.read_csv('electric.csv')          # pd.read_csv for importing csv file
df.head(1)      # head function for seeing selected top rows
```

Out[2]:

	Car full name	Make	Model	Minimal price (gross) [PLN]	Engine power [KM]	Maximum torque [Nm]	Type of brakes	Drive type	Battery capacity [kWh]	Rang (WLTP [km
0	Audi e-tron 55 quattro	Audi	e-tron 55 quattro	345700	360	664	disc (front + rear)	4WD	95.0	43

1 rows × 25 columns

In [3]:

```python
# calculation for task 2
x=df['mean - Energy consumption [kWh/100 km]'].max()
y=df['mean - Energy consumption [kWh/100 km]'].mean()
z=df['mean - Energy consumption [kWh/100 km]'].min()
c=df['mean - Energy consumption [kWh/100 km]'].median()
print("max:",x,"** mean:",y,"** min:",z,"** median:",c)
```

```
max: 28.2 ** mean: 18.994318181818183 ** min: 13.1 ** median: 17.05
```

In [4]:

```python
'''Task 2: You suspect some EVs have unusually high or low energy consumption. F
 outliers in the mean- Energy consumption [kWh/100 km] column'''


col = 'mean - Energy consumption [kWh/100 km]'

Q1 = df[col].quantile(.25)        # for values <=25%
Q3 = df[col].quantile(0.75)       # for values <=75%
IQR = Q3 - Q1                     # interquartile range

lower_bound = Q1 - 1.5 * IQR        # lower range
upper_bound = Q3 + 1.5 * IQR        # upper range

outliers = df[(df[col] < lower_bound) |     # finding outliers
(df[col] > upper_bound)]

print("Lower bound:", lower_bound)
print("Upper bound:", upper_bound)
print("Number of outliers:", outliers.shape[0])

outliers[[col, 'Make', 'Model']].head()
```

```
Lower bound: 3.7499999999999982
Upper bound: 35.35
Number of outliers: 0
```

Out[4]:

mean - Energy consumption [kWh/100 km]	Make	Model

So in our data the value in the column ranges from around 13 to 28 and after doing calculations we don't have any outlier the data is consistent.

In [6]:

```
'''Task 1a: A customer has a budget of 350,000 PLN and wants an EV with a minimu
 of 400 km.
  Your task is to filter out EVs that meet these criteria'''


a=df[(df['Minimal price (gross) [PLN]']<=350000) &
(df['Range (WLTP) [km]']>=400)]
# print(a.head())
a.head()
```

Out[6]:

	Car full name	Make	Model	Minimal price (gross) [PLN]	Engine power [KM]	Maximum torque [Nm]	Type of brakes	Drive type	Battery capacity [kWh]
0	Audi e-tron 55 quattro	Audi	e-tron 55 quattro	345700	360	664	disc (front + rear)	4WD	95.0
8	BMW iX3	BMW	iX3	282900	286	400	disc (front + rear)	2WD (rear)	80.0
15	Hyundai Kona electric 64kWh	Hyundai	Kona electric 64kWh	178400	204	395	disc (front + rear)	2WD (front)	64.0
18	Kia e-Niro 64kWh	Kia	e-Niro 64kWh	167990	204	395	disc (front + rear)	2WD (front)	64.0
20	Kia e-Soul 64kWh	Kia	e-Soul 64kWh	160990	204	395	disc (front + rear)	2WD (front)	64.0

5 rows × 25 columns

This filter identifies cars within a affordable price and with a long range.

In [8]:

```
''' Task 1b
Group them by the manufacturer (Make)'''


d=a.groupby('Make').size().reset_index(name='count')
print(d)
```

```
            Make  count
0           Audi      1
1            BMW      1
2        Hyundai      1
3            Kia      2
4  Mercedes-Benz      1
5          Tesla      3
6     Volkswagen      3
```

This shows that which manufacturer is leading the race within the given criteria.

In [10]:

```
''' Task 1c
Calculate the average battery capacity for each manufacturer'''


d=df.groupby("Make")['Battery capacity [kWh]'].mean()
print(d)
```

```
Make
Audi             87.000000
BMW              54.800000
Citroën          50.000000
DS               50.000000
Honda            35.500000
Hyundai          47.166667
Jaguar           90.000000
Kia              51.600000
Mazda            35.500000
Mercedes-Benz    85.000000
Mini             28.900000
Nissan           47.333333
Opel             50.000000
Peugeot          50.000000
Porsche          89.850000
Renault          52.000000
Skoda            36.800000
Smart            17.600000
Tesla            86.285714
Volkswagen       61.075000
Name: Battery capacity [kWh], dtype: float64
```

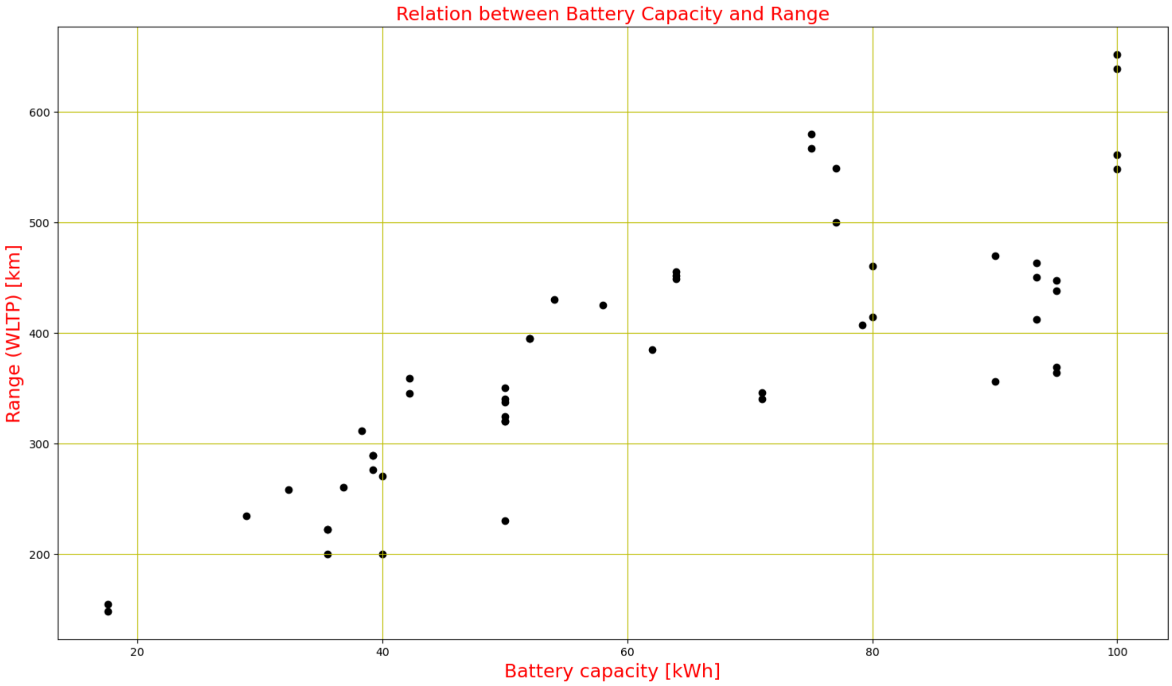
Highlights which manufacturer focus on larger or smaller battery capacity.

In [12]:

```
'''Task 3a: Your manager wants to know if there's a strong relationship between
 capacity and range.
  Create a suitable plot to visualize'''



# scatter plot to visualize the relationship between two columns
plt.figure(figsize=(18,10))
plt.scatter(x=df['Battery capacity [kWh]'],y=df['Range (WLTP) [km]'],
            color='black',alpha=1)
plt.title('Relation between Battery Capacity and Range',color='r',
          fontsize=17)
plt.xlabel('Battery capacity [kWh]',color='r',fontsize=17)
plt.ylabel('Range (WLTP) [km]',color='r',fontsize=17)
plt.grid(True,color='y')
plt.show()
```



Scatter plot showing the relationship between range and battery capacity.

```
In [14]: ''' Task 3b
         ighlight any insights'''



         corr = df['Battery capacity [kWh]'].corr(df['Range (WLTP) [km]'])
         print('Correlation:', corr)
```

```
Correlation: 0.8104385771936846
```

Correlation of positive .81 shows that higher the battery capacity more the range of the ev.

```
In [17]: '''Task 4: Build an EV recommendation class. The class should allow users to inp
          budget, desired range, and battery capacity. The class should then return the t
          matching their criteria'''



         budget=int(input("enter budget"))              # taking budget
         range_req=int(input("enter range"))            # taking range
         capacity_req=int(input("enter battery capacity"))     # taking battery capacity
         result = df[(df['Minimal price (gross) [PLN]'] <= budget) &
                     (df['Range (WLTP) [km]'] >= range_req) &
                     (df['Battery capacity [kWh]'] >= capacity_req)]
         result=result.sort_values('Range (WLTP) [km]',                    # sorting the
                                   ascending=False).head(3)
         result
```

Out[17]:

	Car full name	Make	Model	Minimal price (gross) [PLN]	Engine power [KM]	Maximum torque [Nm]	Type of brakes	Drive type	Batte capaci [kW
40	Tesla Model 3 Long Range	Tesla	Model 3 Long Range	235490	372	510	disc (front + rear)	4WD	75
48	Volkswagen ID.3 Pro S	Volkswagen	ID.3 Pro S	179990	204	310	disc (front) + drum (rear)	2WD (rear)	77
49	Volkswagen ID.4 1st	Volkswagen	ID.4 1st	202390	204	310	disc (front) + drum (rear)	2WD (rear)	77

3 rows × 25 columns

Recommending best evs based on customer requirenment on budget, range, battery capacity.

In [19]:

```
'''Task 5: Inferential Statistics– Hypothesis Testing: Test whether there is a
difference in the average Engine power [KM] of vehicles manufactured by two lea
manufacturers i.e. Tesla and Audi. What insights can you draw from the test res
Recommendations and Conclusion: Provide actionable insights based on your analy
(Conduct a two sample t-test using ttest_ind from scipy.stats module)'''



from scipy.stats import ttest_ind

# Filter Tesla and Audi data
tesla_power = df[df['Make'] == 'Tesla']['Engine power [KM]']
audi_power = df[df['Make'] == 'Audi']['Engine power [KM]']


t_stat, p_value = ttest_ind(tesla_power, audi_power,
                            equal_var=False)  # Welch's t-test (safer)
print("t-statistic:", t_stat)
print("p-value:", p_value)
```

```
t-statistic: 1.7939951827297178
p-value: 0.10684105068839565
```

The p-value is greater than 0.05, so we fail to reject the null hypothesis.This means there is no statistically significant difference in the average engine power between Tesla and Audi EVs.

video link https://drive.google.com/file/d/1fnj2LPnEUjdOf4FxgI9Pwj3584g-qfVu/view?usp=drivesdk