In [ ]:
```
## Task 1
```

In [1]:
```python
import pandas as pd
```

In [5]:
```python
# load the Excel File.
df = pd.read_excel('FEV-data-Excel.xlsx')

# display the few rows of data.
df.head()
```

Out[5]:

	Car full name	Make	Model	Minimal price (gross) [PLN]	Engine power [KM]	Maximum torque [Nm]	Type of brakes	Drive type	Battery capacity [kWh]	Range (WLTP) [km]	...	Permissable gross weight [kg]	Maximum load capacity [kg]	Number of seats	Number of doors	Tire size [in]	Maximum speed [kph]	Boot capacity (VDA) [l]	Acceleration 0-100 kph [s]	Maximum DC charging power [kW]	mean - Energy consumption [kWh/100 km]
0	Audi e-tron 55 quattro	Audi	e-tron 55 quattro	345700	360	664	disc (front + rear)	4WD	95.0	438	...	3130.0	640.0	5	5	19	200	660.0	5.7	150	24.45
1	Audi e-tron 50 quattro	Audi	e-tron 50 quattro	308400	313	540	disc (front + rear)	4WD	71.0	340	...	3040.0	670.0	5	5	19	190	660.0	6.8	150	23.80
2	Audi e-tron S quattro	Audi	e-tron S quattro	414900	503	973	disc (front + rear)	4WD	95.0	364	...	3130.0	565.0	5	5	20	210	660.0	4.5	150	27.55
3	Audi e-tron Sportback 50 quattro	Audi	e-tron Sportback 50 quattro	319700	313	540	disc (front + rear)	4WD	71.0	346	...	3040.0	640.0	5	5	19	190	615.0	6.8	150	23.30
4	Audi e-tron Sportback 55 quattro	Audi	e-tron Sportback 55 quattro	357000	360	664	disc (front + rear)	4WD	95.0	447	...	3130.0	670.0	5	5	19	200	615.0	5.7	150	23.85

5 rows × 25 columns

In [ ]:
```
## Filter out EVs that meet the criteria of budget of 350,000 PLN and wants an EV with a minimum range of 400 km.
```

In [6]:
```python
# Apply the filter
filtered_df = df[(df['Minimal price (gross) [PLN]'] <= 350000) & (df['Range (WLTP) [km]'] >= 400)]

# Show the filtered results
filtered_df.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]	Range (WLTP) [km]	...	Permissable gross weight [kg]	Maximum load capacity [kg]	Number of seats	Number of doors	Tire size [in]	Maximum speed [kph]	Boot capacity (VDA) [l]	Acceleration 0-100 kph [s]	Maximum DC charging power [kW]	mean - Energy consumption [kWh/100 km]
0	Audi e-tron 55 quattro	Audi	e-tron 55 quattro	345700	360	664	disc (front + rear)	4WD	95.0	438	...	3130.0	640.0	5	5	19	200	660.0	5.7	150	24.45
8	BMW iX3	BMW	iX3	282900	286	400	disc (front + rear)	2WD (rear)	80.0	460	...	2725.0	540.0	5	5	19	180	510.0	6.8	150	18.80
15	Hyundai Kona electric 64kWh	Hyundai	Kona electric 64kWh	178400	204	395	disc (front + rear)	2WD (front)	64.0	449	...	2170.0	485.0	5	5	17	167	332.0	7.6	100	15.40
18	Kia e-Niro 64kWh	Kia	e-Niro 64kWh	167990	204	395	disc (front + rear)	2WD (front)	64.0	455	...	2230.0	493.0	5	5	17	167	451.0	7.8	100	15.90
20	Kia e-Soul 64kWh	Kia	e-Soul 64kWh	160990	204	395	disc (front + rear)	2WD (front)	64.0	452	...	1682.0	498.0	5	5	17	167	315.0	7.9	100	15.70

5 rows × 25 columns

In [ ]:
```
## Group them by the manufacturer
```

In [8]:
```python
filtered_df.groupby('Make')['Model'].count().reset_index(name='Car Count')
```

Out[8]:

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

In [ ]:
```
## Calculate the average battery capacity for each manufacturer.
```

In [14]:
```python
avg_battery_by_make = filtered_df.groupby('Make')['Battery capacity [kWh]'].mean().reset_index()
avg_battery_by_make.rename(columns={'Battery capacity [kWh]': 'Average Battery Capacity (kWh)'}, inplace=True)
avg_battery_by_make
```

Out[14]:

	Make	Average Battery Capacity (kWh)
0	Audi	95.000000
1	BMW	80.000000
2	Hyundai	64.000000
3	Kia	64.000000
4	Mercedes-Benz	80.000000
5	Tesla	68.000000
6	Volkswagen	70.666667

Task 1 Analysis

We filtered electric vehicles with a **budget below 350,000 PLN** and **minimum range of 400 km**. The resulting dataset includes models from multiple manufacturers.

- Grouping by **Make** helped identify which brands offer options within this range.
- We then calculated the **average battery capacity** per brand to evaluate which manufacturers provide better battery performance under the given budget and range constraints.

These insights can guide the customer towards value-for-money options while ensuring sufficient driving range.

In [ ]:
```
## Task 2
```

In [25]:
```python
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

In [ ]:
```
## Find the  outliers in the mean- Energy consumption [kWh/100 km] column.
```

In [32]:
```python
from scipy.stats import zscore

col = 'mean - Energy consumption [kWh/100 km]'
df[col] = pd.to_numeric(df[col], errors='coerce')
df_clean = df.dropna(subset=[col])

# Calculate Z-scores
df_clean['z_score'] = zscore(df_clean[col])

# Use a lower threshold (like 2)
threshold = 2
outliers = df_clean[(df_clean['z_score'] > threshold) | (df_clean['z_score'] < -threshold)]

print("Number of outliers found:", len(outliers))
outliers.head()
```

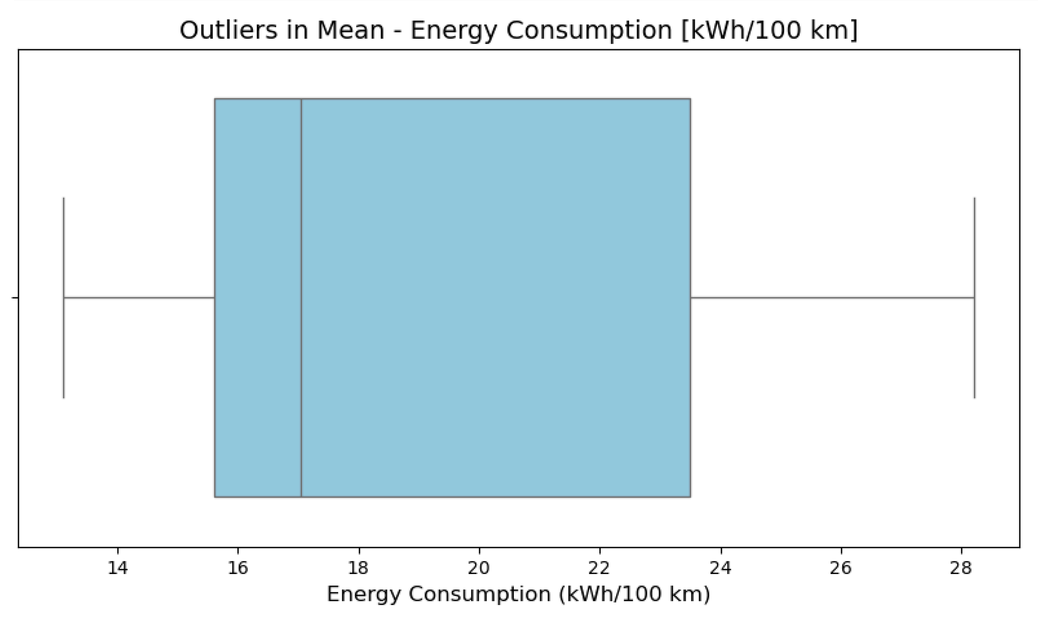
```
Number of outliers found: 1
C:\Users\Shruti\AppData\Local\Temp\ipykernel_12512\209100722.py:8: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  df_clean['z_score'] = zscore(df_clean[col])
```

Out[32]:

	Car full name	Make	Model	Minimal price (gross) [PLN]	Engine power [KM]	Maximum torque [Nm]	Type of brakes	Drive type	Battery capacity [kWh]	Range (WLTP) [km]	...	Maximum load capacity [kg]	Number of seats	Number of doors	Tire size [in]	Maximum speed [kph]	Boot capacity (VDA) [l]	Acceleration 0-100 kph [s]	Maximum DC charging power [kW]	mean - Energy consumption [kWh/100 km]	z_score
51	Mercedes-Benz EQV (long)	Mercedes-Benz	EQV (long)	339480	204	362	NaN	2WD (front)	90.0	356	...	865.0	6	5	17	160	NaN	NaN	110	28.2	2.107645

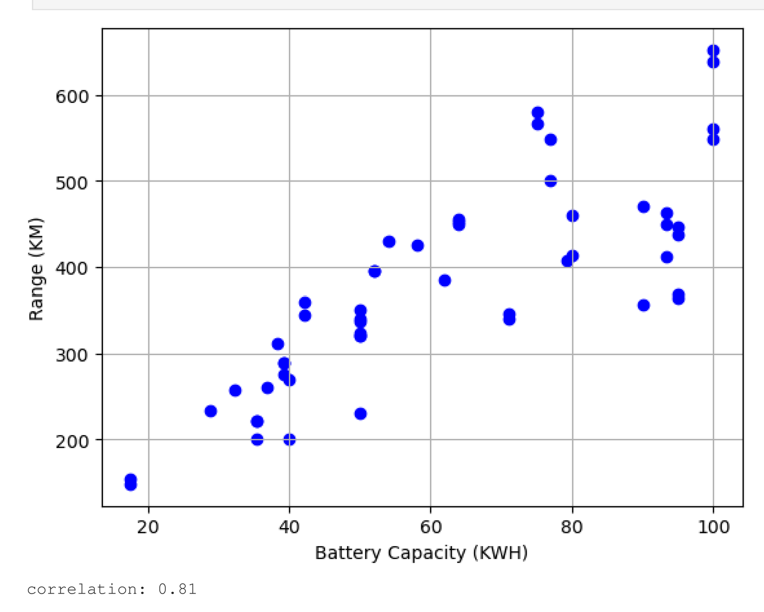
1 rows × 26 columns

In [35]:
```python
# Set figure size
plt.figure(figsize=(10, 5))

# Draw the box plot
sns.boxplot(x=df_clean[col], color='skyblue')

# Add title and labels
plt.title('Outliers in Mean - Energy Consumption [kWh/100 km]', fontsize=14)
plt.xlabel('Energy Consumption (kWh/100 km)', fontsize=12)

# Show the plot
plt.show()
```



In [ ]:
```
## Task 3 to know if there's a strong relationship between battery  capacity and range.
```

In [4]:
```python
data = df [['Battery capacity [kWh]','Range (WLTP) [km]']]
data.columns = ['Battery','Range']
plt.scatter(data['Battery'],data['Range'],color='blue')
plt.xlabel("Battery Capacity (KWH)")
plt.ylabel("Range (KM)")
plt.grid(True)
plt.show()

#correlation
corr= data['Battery'].corr(data['Range'])
print ("correlation:",round(corr,2))
```




```
correlation: 0.81
```

Important Insights

As the **battery capacity increases**, the **driving range of electric vehicles also increases**. This is supported by the **correlation value of 0.81**, which shows a strong **positive relationship** between battery size and range. It means **vehicles with larger batteries tend to go farther on a full charge**.

From the scatter plot, we can also observe that most EVs with a **battery capacity above 80 kWh** achieve a range of over **500 km**, which could be important for long-distance drivers or premium vehicle models."

In [ ]:
```
## Task 4  Build an EV recommendation class.
```

In [9]:
```python
import pandas as pd

data = pd.read_excel('FEV-data-Excel.xlsx')

# Create a class
class EV_Recommendation:
    def __init__(self, data):
        self.data = data

    def recommend(self, budget, min_range, min_battery):

        # Filter the data using the user’s inputs
        result = self.data[
            (self.data['Minimal price (gross) [PLN]'] <= budget) &
            (self.data['Range (WLTP) [km]'] >= min_range) &
            (self.data['Battery capacity [kWh]'] >= min_battery)
        ]
        # Return top 3 matching EVs
        return result[['Car full name', 'Make', 'Range (WLTP) [km]',
                       'Battery capacity [kWh]', 'Minimal price (gross) [PLN]']].head(3)

# Take inputs from the user
budget = int(input("Enter your budget (in PLN): "))
min_range = int(input("Enter minimum range you want (in km): "))
min_battery = float(input("Enter minimum battery capacity (in kWh): "))

# Create object and call the function
recommender = EV_Recommendation(data)
top_evs = recommender.recommend(budget, min_range, min_battery)

# Show the result
print("\nTop 3 EVs for you:\n")
print(top_evs)
```

```
Top 3 EVs for you:

Empty DataFrame
Columns: [Car full name, Make, Range (WLTP) [km], Battery capacity [kWh], Minimal price (gross) [PLN]]
Index: []
```

In [ ]:
```
## Task 5 Hypothesis Testing whether there is a significant  difference in the average Engine power [KM] of vehicles Tesla & Audi
```

In [5]:
```python
from scipy.stats import ttest_ind

df = df.dropna(subset=["Engine power [KM]", "Make"])
tesla_power = df[df["Make"] == "Tesla"]["Engine power [KM]"]
audi_power = df[df["Make"] == "Audi"]["Engine power [KM]"]

# Perform independent t-test
t_stat, p_value = ttest_ind(tesla_power, audi_power, equal_var=False)  # Welch's t-test

print("T-statistic:", t_stat)
print("P-value:", p_value)
```

```
T-statistic: 1.7939951827297178
P-value: 0.10684105068839565
```

Task 5: Hypothesis Testing – Tesla vs Audi Engine Power

In this task, we aimed to determine whether there is a statistically significant difference in the average **engine power (KM)** between electric vehicles from **Tesla** and **Audi**.

Hypotheses:

- **Null Hypothesis (H_0):** There is **no significant difference** in the average engine power between Tesla and Audi.
- **Alternative Hypothesis (H_1):** There **is a significant difference** in the average engine power between Tesla and Audi.

Test Used:

We used a **two-sample independent t-test (Welch’s t-test)**, which is appropriate when comparing two groups that may have unequal variances.

Interpretation:

The p-value is approximately **0.107**, which is greater than the significance level of 0.05.

Therefore, we fail to reject the null **hypothesis**.

Conclusion:

There is no statistically significant difference in the average engine power between Tesla and Audi vehicles based on the data provided.

Actionable Insights:

1. **No Major Difference in Power**
   The test indicates that Tesla and Audi vehicles have **similar engine power**, on average. This suggests that **engine power is not a major differentiator** between the two brands in this dataset.

2. **Focus on Other Differentiating Features**
   Since engine power doesn’t show a significant difference, companies or analysts should focus on **other aspects** like **battery range, price, design, or technology** to better understand the strengths and market appeal of each brand.

3. **Data-Driven Brand Positioning**
   Both brands can highlight performance equivalency in terms of engine power, and instead **emphasize unique features** that set them apart for marketing or customer targeting.

In [ ]:

VIDEO LINK: https://drive.google.com/file/d/1ZgxRbKfPrOqlt_SSoQJkW5jehLuUYJ-7/view?usp=sharing

In [ ]: