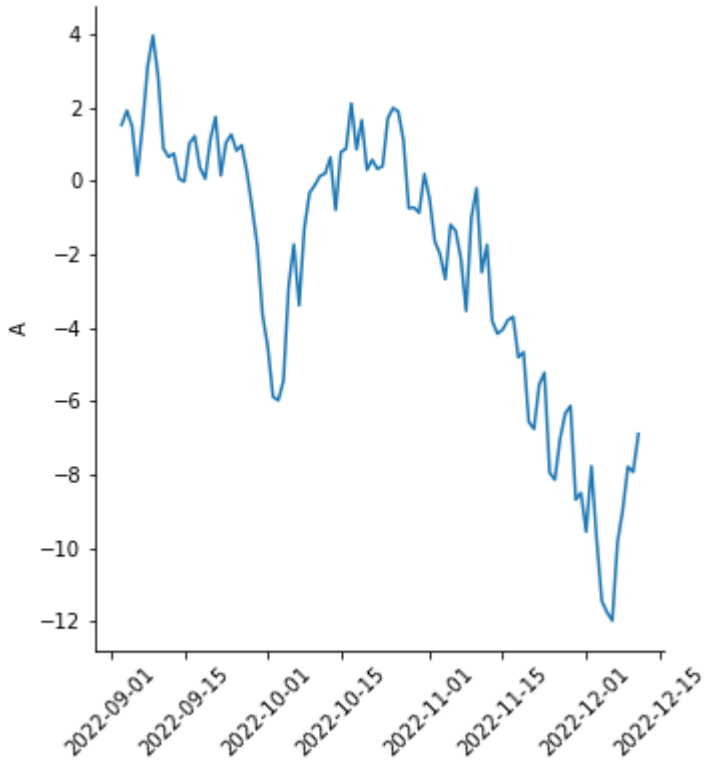
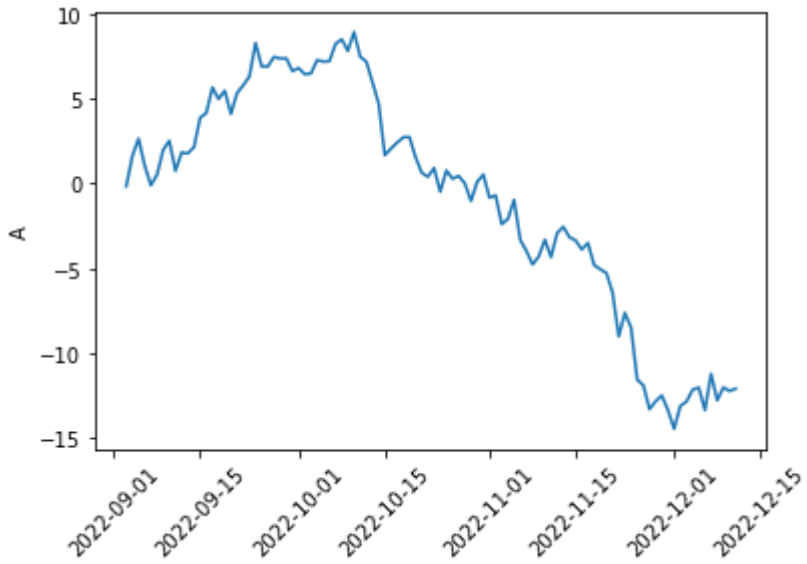
In [3]:

```
# Plotting Lines
# You can also show continuous data such as time-series data
# along a line. Time-series data has timestamp data in at least
# one column or has an index. A great example of atime series is a
# table of daily temperature records.

%matplotlib inline
import matplotlib.pyplot as plt
import pandas as pd
import numpy as np
import seaborn as sns
df = pd.DataFrame(np.random.randn(100, 4),
                  index=pd.date_range("9/3/2022",
                                      periods=100),
columns=list("ABCD"))
df = df.cumsum()
# You can use the function relplot() to draw the line as follows:
sns.relplot(x=df.index, y='A', kind="line", data=df)
plt.xticks(rotation=45)
plt.show()
```



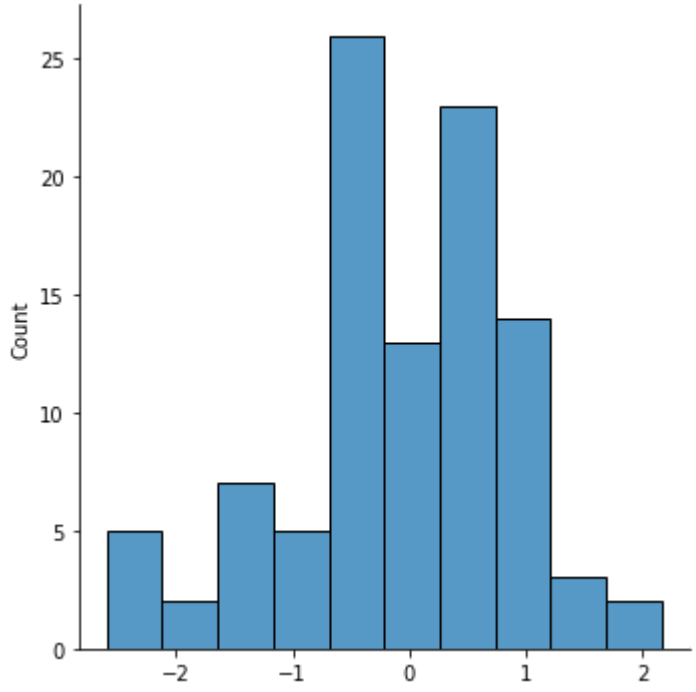
In [4]:

```
# You can also produce the output using the following code:

%matplotlib inline
import matplotlib.pyplot as plt
import pandas as pd
import numpy as np
import seaborn as sns
df = pd.DataFrame(np.random.randn(100, 4),
                  index=pd.date_range("9/3/2022",
                                      periods=100),
columns=list("ABCD"))
df = df.cumsum()
sns.lineplot(x=df.index,
             y='A', data=df)
plt.xticks(rotation=45)
plt.show()
```



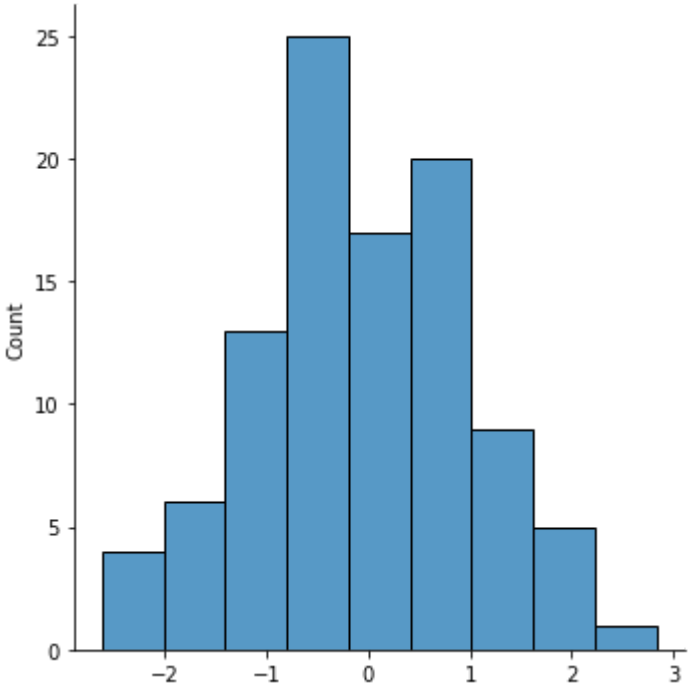
In [5]:

```
# Visualizing the Distribution of Data
# One of the most prominent examples of visualizing the
# distribution of data is a frequency table or a frequency distribution
# table. You can create buckets of value ranges that the
# data can have (the domain), and then you can list the number of items
# that satisfy the criteria for the bucket. You can also vary the bucket
# size, with the smallest size being 1.
# You can visually show the information of a frequency distribution using
# bars and lines. If you use bars, then it is known as a histogram.
# You can use the function displot() to visualize the frequency data.
# Let's start with dummy univariate data.

%matplotlib inline
import matplotlib.pyplot as plt
import numpy as np
import seaborn as sns
x = np.random.randn(100)
sns.displot(x)
plt.show()
```



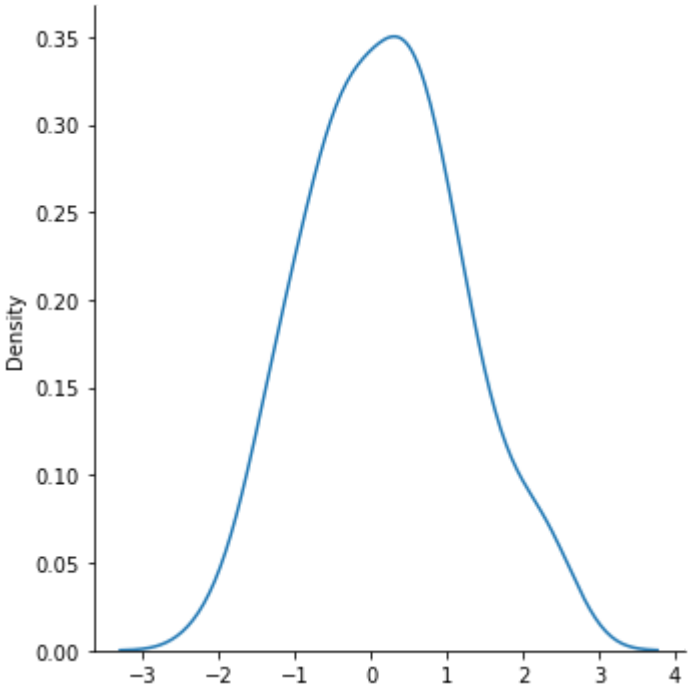
In [6]:

```
# You can also make it explicit that you need a histogram in the output
# as follows:

%matplotlib inline
import matplotlib.pyplot as plt
import numpy as np
import seaborn as sns
x = np.random.randn(100)
sns.displot(x, kind='hist')
plt.show()
```



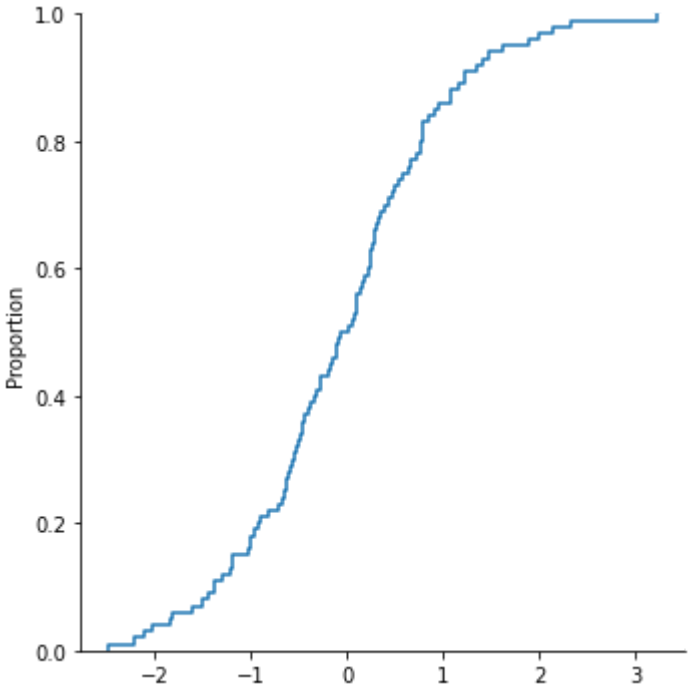
In [7]:

```
# A histogram is the default kind of graph. You can also show a Gaussian
# kernel density estimation (KDE) as follows:

%matplotlib inline
import matplotlib.pyplot as plt
import numpy as np
import seaborn as sns
x = np.random.randn(100)
sns.displot(x, kind='kde')
plt.show()
```



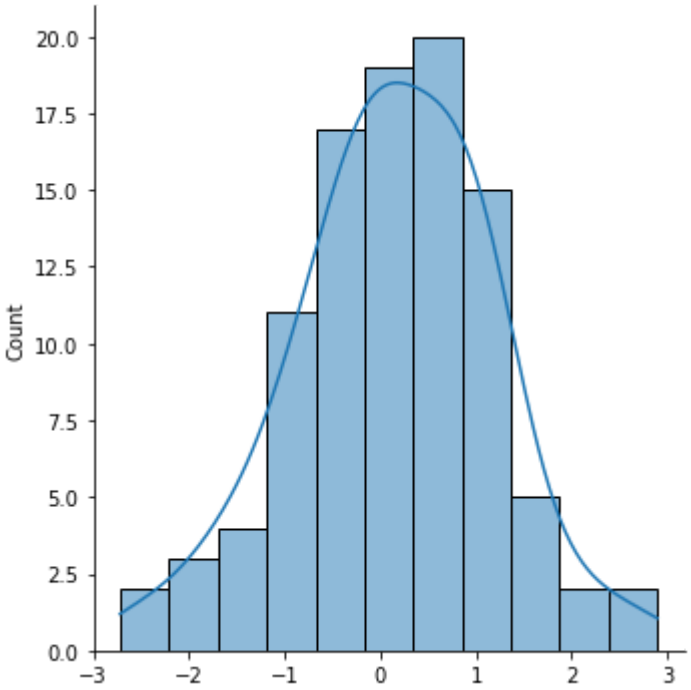
In [8]:

```
# You can visualize an empirical cumulative distribution function (eCDF)
# as follows:

%matplotlib inline
import matplotlib.pyplot as plt
import numpy as np
import seaborn as sns
x = np.random.randn(100)
sns.displot(x, kind='ecdf')
plt.show()
```



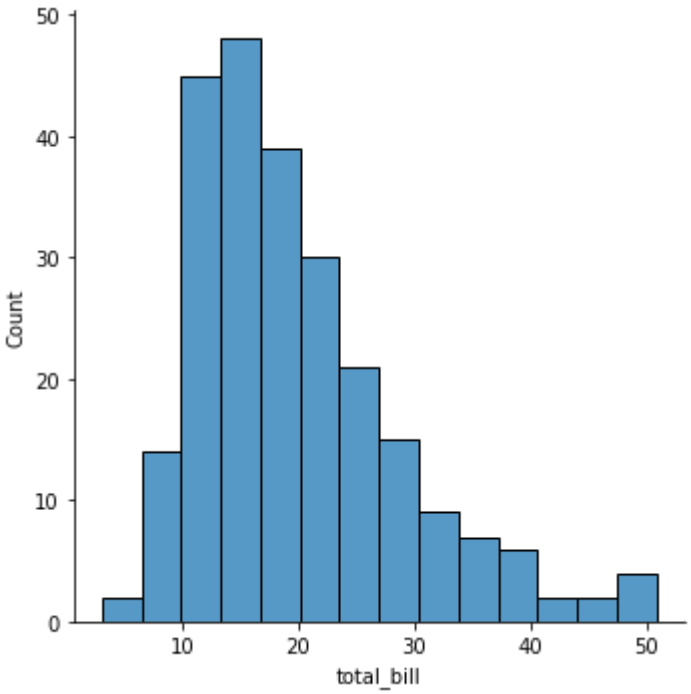
In [9]:

```
# You can combine a histogram and a KDE as follows:

%matplotlib inline
import matplotlib.pyplot as plt
import numpy as np
import seaborn as sns
x = np.random.randn(100)
sns.displot(x, kind='hist', kde=True)
plt.show()
```



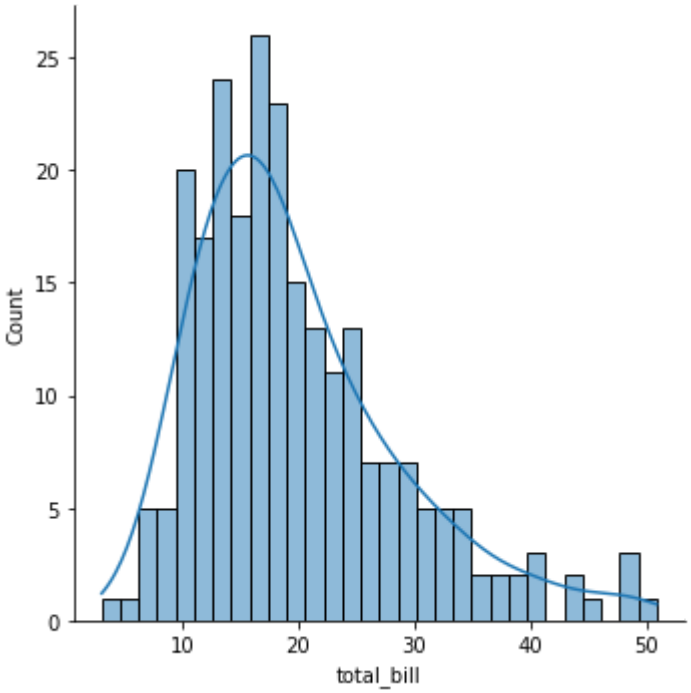
In [10]:

```
# Now let's use some real-life data, as follows:

%matplotlib inline
import matplotlib.pyplot as plt
import seaborn as sns
tips = sns.load_dataset("tips")
sns.displot(x='total_bill', data=tips, kind='hist')
plt.show()
```



In [11]:

```
# You can customize the size of bins (or buckets) in the visualization
# as follows:

%matplotlib inline
import matplotlib.pyplot as plt
import seaborn as sns
tips = sns.load_dataset("tips")
sns.displot(x='total_bill', data=tips,
            kind='hist', bins=30, kde=True)
plt.show()
```



In [12]:

```
# You can adjust the hue of the plots based on a criterion of your choice
# as follows:

%matplotlib inline
import matplotlib.pyplot as plt
import seaborn as sns
tips = sns.load_dataset("tips")
sns.displot(x='total_bill', data=tips,
            kind='kde', hue='size')
plt.show()
```