Tugas Besar IF2220 Probabilitas dan Statistika

Penarikan Kesimpulan dan Pengujian Hipotesis

Suryanto - 13520059
Steven - 13520131

In [1]:

```
# mengimport library pandas
import pandas as pd
import matplotlib.pyplot as plt
import scipy.stats as s
import math
import seaborn as sns
import warnings
warnings.simplefilter(action="ignore", category=FutureWarning)

# membaca file csv
df = pd.read_csv("water_potability.csv",
        names = ["id", "pH", "Hardness", "Solids", "Chloramines", "Sulfate", "Conductivi
ty", "OrganicCarbon", "Trihalomethanes",
                "Turbidity", "Potability"])
```

1. Deskripsi Statistika (Descriptive Statistics)

1.1 Kolom pH

In [2]:

```
print("============================")
print("Deskripsi Statistika Kolom pH")
print("============================")
col_ph = df["pH"]

# mean
print("Mean :", col_ph.mean())
print("----------------------------")

# median
print("Median :", col_ph.median())
print("----------------------------")

# modus
if (len(col_ph.mode()) != 2010):
    print("Terdapat "+ str(len(col_ph.mode())) + " buah modus, yaitu:")
    for i in range (len(col_ph.mode()) - 1):
        print(str(col_ph.mode()[i]) + ", ", end="")
    print(str(col_ph.mode()[len(col_ph.mode()) - 1]))
else:
    print("Terdapat 2010 buah modus")
    print("Salah satu modusnya adalah:")
    print(col_ph.mode()[0])
    print("Hal ini mengimplementasikan")
    print("bahwa semua data pada")
    print("kolom ini bersifat unik")
print("----------------------------")

# standar deviasi
print("Stander deviasi :", col_ph.std())
print("----------------------------")

# variansi
```

```
print("Variansi :", col_ph.var())
print("-----------------------------")

# range
range_data = col_ph.max() - col_ph.min()
print("Range :", range_data)
print("-----------------------------")

# nilai minimum
print("Minimum :", col_ph.min())
print("-----------------------------")

# maksimum
print("Maksimum :", col_ph.max())
print("-----------------------------")

# kuartil
print("Kuartil bawah  :", col_ph.quantile(.25) )
print("Kuartil tengah :", col_ph.quantile(.50) )
print("Kuartil atas   :" , col_ph.quantile(.75) )
print("-----------------------------")

# IQR
iqr_data = col_ph.quantile(.75) - col_ph.quantile(.25)
print("IQR :", iqr_data)
print("-----------------------------")

# skewness
print("Skewness :", col_ph.skew())
print("-----------------------------")

# kurtosis
print("Kurtosis: ", col_ph.kurtosis())
print("-----------------------------")
```

```
==============================
Deskripsi Statistika Kolom pH
==============================
Mean : 7.0871927687138285
-----------------------------
Median : 7.029490455474185
-----------------------------
Terdapat 2010 buah modus
Salah satu modusnya adalah:
0.2274990502021987
Hal ini mengimplementasikan
bahwa semua data pada
kolom ini bersifat unik
-----------------------------
Stander deviasi : 1.5728029470456655
-----------------------------
Variansi : 2.4737091102355304
-----------------------------
Range : 13.7725009497978
-----------------------------
Minimum : 0.2274990502021987
-----------------------------
Maksimum : 13.999999999999998
-----------------------------
Kuartil bawah  : 6.09078502142353
Kuartil tengah : 7.029490455474185
Kuartil atas   : 8.053006240791538
-----------------------------
IQR : 1.9622212193680078
-----------------------------
Skewness : 0.04853451405270669
-----------------------------
Kurtosis:  0.6269041256617065
-----------------------------
```

1.2 Kolom Hardness

In [3]:

```
print("==================================")
print("Deskripsi Statistika Kolom Hardness")
print("==================================")
col_hardness = df["Hardness"]

# mean
print("Mean :", col_hardness.mean())
print("----------------------------------")

# median
print("Median :", col_hardness.median())
print("----------------------------------")

# modus
if (len(col_hardness.mode()) != 2010):
    print("Terdapat "+ str(len(col_hardness.mode())) + " buah modus, yaitu:")
    for i in range (len(col_hardness.mode()) - 1):
        print(str(col_hardness.mode()[i]) + ", ", end="")
    print(str(col_hardness.mode()[len(col_hardness.mode()) - 1]))
else:
    print("Terdapat 2010 buah modus")
    print("Salah satu modusnya adalah:")
    print(col_ph.mode()[0])
    print("Hal ini mengimplementasikan")
    print("bahwa semua data pada")
    print("kolom ini bersifat unik")
print("----------------------------------")

# standar deviasi
print("Stander deviasi :", col_hardness.std())
print("----------------------------------")

# variansi
print("Variansi :", col_hardness.var())
print("----------------------------------")

# range
range_data = col_hardness.max() - col_hardness.min()
print("Range :", range_data)
print("----------------------------------")

# nilai minimum
print("Minimum :", col_hardness.min())
print("----------------------------------")

# maksimum
print("Maksimum :", col_hardness.max())
print("----------------------------------")

# kuartil
print("Kuartil bawah  :", col_hardness.quantile(.25) )
print("Kuartil tengah :", col_hardness.quantile(.50) )
print("Kuartil atas   :" , col_hardness.quantile(.75) )
print("----------------------------------")

# IQR
iqr_data = col_hardness.quantile(.75) - col_hardness.quantile(.25)
print("IQR :", iqr_data)
print("----------------------------------")

# skewness
print("Skewness :", col_hardness.skew())
print("----------------------------------")

# kurtosis
print("Kurtosis: ", col_hardness.kurtosis())
print("----------------------------------")
```

==========

```
Deskripsi Statistika Kolom Hardness
==================================
Mean : 195.96920903783524
----------------------------------
Median : 197.20352491941043
----------------------------------
Terdapat 2010 buah modus
Salah satu modusnya adalah:
0.2274990502021987
Hal ini mengimplementasikan
bahwa semua data pada
kolom ini bersifat unik
----------------------------------
Stander deviasi : 32.643165859429864
----------------------------------
Variansi : 1065.5762773262472
----------------------------------
Range : 243.84589036652147
----------------------------------
Minimum : 73.4922336890611
----------------------------------
Maksimum : 317.33812405558257
----------------------------------
Kuartil bawah  : 176.74065667669896
Kuartil tengah : 197.20352491941043
Kuartil atas   : 216.44758866727156
----------------------------------
IQR : 39.7069319905726
----------------------------------
Skewness : -0.08532104172868622
----------------------------------
Kurtosis:  0.5254804942991402
----------------------------------
```

1.3 Kolom Solids

In [4]:

```
print("==================================")
print("Deskripsi Statistika Kolom Solids")
print("==================================")
col_solids = df["Solids"]

# mean
print("Mean :", col_solids.mean())
print("----------------------------------")

# median
print("Median :", col_solids.median())
print("----------------------------------")

# modus
if (len(col_solids.mode()) != 2010):
    print("Terdapat "+ str(len(col_solids.mode())) + " buah modus, yaitu:")
    for i in range (len(col_solids.mode()) - 1):
        print(str(col_solids.mode()[i]) + ", ", end="")
    print(str(col_solids.mode()[len(col_solids.mode()) - 1]))
else:
    print("Terdapat 2010 buah modus")
    print("Salah satu modusnya adalah:")
    print(col_ph.mode()[0])
    print("Hal ini mengimplementasikan")
    print("bahwa semua data pada")
    print("kolom ini bersifat unik")
print("----------------------------------")

# standar deviasi
print("Stander deviasi :", col_solids.std())
print("----------------------------------")
```

```
# variansi
print("Variansi :", col_solids.var())
print("---------------------------------")

# range
range_data = col_solids.max() - col_solids.min()
print("Range :", range_data)
print("---------------------------------")

# nilai minimum
print("Minimum :", col_solids.min())
print("---------------------------------")

# maksimum
print("Maksimum :", col_solids.max())
print("---------------------------------")

# kuartil
print("Kuartil bawah  :", col_solids.quantile(.25) )
print("Kuartil tengah :", col_solids.quantile(.50) )
print("Kuartil atas   :" , col_solids.quantile(.75) )
print("---------------------------------")

# IQR
iqr_data = col_solids.quantile(.75) - col_solids.quantile(.25)
print("IQR :", iqr_data)
print("---------------------------------")

# skewness
print("Skewness :", col_solids.skew())
print("---------------------------------")

# kurtosis
print("Kurtosis: ", col_solids.kurtosis())
print("---------------------------------")
```

```
================================
Deskripsi Statistika Kolom Solids
================================
Mean : 21904.673439053095
--------------------------------
Median : 20926.88215534375
--------------------------------
Terdapat 2010 buah modus
Salah satu modusnya adalah:
0.2274990502021987
Hal ini mengimplementasikan
bahwa semua data pada
kolom ini bersifat unik
--------------------------------
Stander deviasi : 8625.397911190576
--------------------------------
Variansi : 74397489.12637076
--------------------------------
Range : 56167.72980146483
--------------------------------
Minimum : 320.942611274359
--------------------------------
Maksimum : 56488.67241273919
--------------------------------
Kuartil bawah  : 15614.412961614333
Kuartil tengah : 20926.88215534375
Kuartil atas   : 27170.534648603603
--------------------------------
IQR : 11556.12168698927
--------------------------------
Skewness : 0.5910113724580447
--------------------------------
Kurtosis:  0.33732026745944976
--------------------------------
```

1.4 Kolom Chloramines

In [5]:

```
print("=====================================")
print("Deskripsi Statistika Kolom Chloramines")
print("=====================================")
col_chloramines = df["Chloramines"]

# mean
print("Mean :", col_chloramines.mean())
print("-------------------------------------")

# median
print("Median :", col_chloramines.median())
print("-------------------------------------")

# modus
if (len(col_chloramines.mode()) != 2010):
    print("Terdapat "+ str(len(col_chloramines.mode())) + " buah modus, yaitu:")
    for i in range (len(col_chloramines.mode()) - 1):
        print(str(col_chloramines.mode()[i]) + ", ", end="")
    print(str(col_chloramines.mode()[len(col_chloramines.mode()) - 1]))
else:
    print("Terdapat 2010 buah modus")
    print("Salah satu modusnya adalah:")
    print(col_ph.mode()[0])
    print("Hal ini mengimplementasikan")
    print("bahwa semua data pada")
    print("kolom ini bersifat unik")
print("-------------------------------------")

# standar deviasi
print("Stander deviasi :", col_chloramines.std())
print("-------------------------------------")

# variansi
print("Variansi :", col_chloramines.var())
print("-------------------------------------")

# range
range_data = col_chloramines.max() - col_chloramines.min()
print("Range :", range_data)
print("-------------------------------------")

# nilai minimum
print("Minimum :", col_chloramines.min())
print("-------------------------------------")

# maksimum
print("Maksimum :", col_chloramines.max())
print("-------------------------------------")

# kuartil
print("Kuartil bawah  :", col_chloramines.quantile(.25) )
print("Kuartil tengah :", col_chloramines.quantile(.50) )
print("Kuartil atas   :" , col_chloramines.quantile(.75) )
print("-------------------------------------")

# IQR
iqr_data = col_chloramines.quantile(.75) - col_chloramines.quantile(.25)
print("IQR :", iqr_data)
print("-------------------------------------")

# skewness
print("Skewness :", col_chloramines.skew())
print("-------------------------------------")

# kurtosis
print("Kurtosis: ", col_chloramines.kurtosis())
print("-------------------------------------")
```

```
====================================
Deskripsi Statistika Kolom Chloramines
====================================
Mean : 7.134322344600104
------------------------------------
Median : 7.1420143046226645
------------------------------------
Terdapat 2010 buah modus
Salah satu modusnya adalah:
0.2274990502021987
Hal ini mengimplementasikan
bahwa semua data pada
kolom ini bersifat unik
------------------------------------
Stander deviasi : 1.5852140982642102
------------------------------------
Variansi : 2.512903737335613
------------------------------------
Range : 11.736129095114823
------------------------------------
Minimum : 1.3908709048851806
------------------------------------
Maksimum : 13.127000000000002
------------------------------------
Kuartil bawah  : 6.138326387572855
Kuartil tengah : 7.1420143046226645
Kuartil atas   : 8.109933216133502
------------------------------------
IQR : 1.9716068285606472
------------------------------------
Skewness : 0.013003497779569528
------------------------------------
Kurtosis:  0.5497821097667472
------------------------------------
```

1.5 Kolom Sulfate

In [6]:

```
print("====================================")
print("Deskripsi Statistika Kolom Sulfate")
print("====================================")
col_sulfate = df["Sulfate"]

# mean
print("Mean :", col_sulfate.mean())
print("------------------------------------")

# median
print("Median :", col_sulfate.median())
print("------------------------------------")

# modus
if (len(col_sulfate.mode()) != 2010):
    print("Terdapat "+ str(len(col_sulfate.mode())) + " buah modus, yaitu:")
    for i in range (len(col_sulfate.mode()) - 1):
        print(str(col_sulfate.mode()[i]) + ", ", end="")
    print(str(col_sulfate.mode()[len(col_sulfate.mode()) - 1]))
else:
    print("Terdapat 2010 buah modus")
    print("Salah satu modusnya adalah:")
    print(col_ph.mode()[0])
    print("Hal ini mengimplementasikan")
    print("bahwa semua data pada")
    print("kolom ini bersifat unik")
print("------------------------------------")

# standar deviasi
print("Stander deviasi :", col_sulfate.std())
print("------------------------------------")
```

```
# variansi
print("Variansi :", col_sulfate.var())
print("---------------------------------")

# range
range_data = col_sulfate.max() - col_sulfate.min()
print("Range :", range_data)
print("---------------------------------")

# nilai minimum
print("Minimum :", col_sulfate.min())
print("---------------------------------")

# maksimum
print("Maksimum :", col_sulfate.max())
print("---------------------------------")

# kuartil
print("Kuartil bawah  :", col_sulfate.quantile(.25) )
print("Kuartil tengah :", col_sulfate.quantile(.50) )
print("Kuartil atas   :" , col_sulfate.quantile(.75) )
print("---------------------------------")

# IQR
iqr_data = col_sulfate.quantile(.75) - col_sulfate.quantile(.25)
print("IQR :", iqr_data)
print("---------------------------------")

# skewness
print("Skewness :", col_sulfate.skew())
print("---------------------------------")

# kurtosis
print("Kurtosis: ", col_sulfate.kurtosis())
print("---------------------------------")
```

```
=================================
Deskripsi Statistika Kolom Sulfate
=================================
Mean : 333.211376415189
---------------------------------
Median : 332.2141128069568
---------------------------------
Terdapat 2010 buah modus
Salah satu modusnya adalah:
0.2274990502021987
Hal ini mengimplementasikan
bahwa semua data pada
kolom ini bersifat unik
---------------------------------
Stander deviasi : 41.21111102560979
---------------------------------
Variansi : 1698.355671965137
---------------------------------
Range : 352.03064230599716
---------------------------------
Minimum : 129.00000000000003
---------------------------------
Maksimum : 481.0306423059972
---------------------------------
Kuartil bawah  : 307.6269864860709
Kuartil tengah : 332.2141128069568
Kuartil atas   : 359.26814739141554
---------------------------------
IQR : 51.641160905344634
---------------------------------
Skewness : -0.04572780443653543
---------------------------------
Kurtosis:  0.7868544988131605
---------------------------------
```

1.6 Kolom Conductivity

In [7]:

```
print("=======================================")
print("Deskripsi Statistika Kolom Conductivity")
print("=======================================")
col_conductivity = df["Conductivity"]

# mean
print("Mean :", col_conductivity.mean())
print("---------------------------------------")

# median
print("Median :", col_conductivity.median())
print("---------------------------------------")

# modus
if (len(col_conductivity.mode()) != 2010):
    print("Terdapat "+ str(len(col_conductivity.mode())) + " buah modus, yaitu:")
    for i in range (len(col_conductivity.mode()) - 1):
        print(str(col_conductivity.mode()[i]) + ", ", end="")
    print(str(col_conductivity.mode()[len(col_conductivity.mode()) - 1]))
else:
    print("Terdapat 2010 buah modus")
    print("Salah satu modusnya adalah:")
    print(col_ph.mode()[0])
    print("Hal ini mengimplementasikan")
    print("bahwa semua data pada")
    print("kolom ini bersifat unik")
print("---------------------------------------")

# standar deviasi
print("Stander deviasi :", col_conductivity.std())
print("---------------------------------------")

# variansi
print("Variansi :", col_conductivity.var())
print("---------------------------------------")

# range
range_data = col_conductivity.max() - col_conductivity.min()
print("Range :", range_data)
print("---------------------------------------")

# nilai minimum
print("Minimum :", col_conductivity.min())
print("---------------------------------------")

# maksimum
print("Maksimum :", col_conductivity.max())
print("---------------------------------------")

# kuartil
print("Kuartil bawah  :", col_conductivity.quantile(.25) )
print("Kuartil tengah :", col_conductivity.quantile(.50) )
print("Kuartil atas   :" , col_conductivity.quantile(.75) )
print("---------------------------------------")

# IQR
iqr_data = col_conductivity.quantile(.75) - col_conductivity.quantile(.25)
print("IQR :", iqr_data)
print("---------------------------------------")

# skewness
print("Skewness :", col_conductivity.skew())
print("---------------------------------------")

# kurtosis
print("Kurtosis: ", col_conductivity.kurtosis())
```

```
print("----------------------------------------")
```

```
========================================
Deskripsi Statistika Kolom Conductivity
========================================
Mean : 426.47670835257907
----------------------------------------
Median : 423.43837202443706
----------------------------------------
Terdapat 2010 buah modus
Salah satu modusnya adalah:
0.2274990502021987
Hal ini mengimplementasikan
bahwa semua data pada
kolom ini bersifat unik
----------------------------------------
Stander deviasi : 80.70187180729437
----------------------------------------
Variansi : 6512.792113200974
----------------------------------------
Range : 551.7228828031471
----------------------------------------
Minimum : 201.6197367551575
----------------------------------------
Maksimum : 753.3426195583046
----------------------------------------
Kuartil bawah  : 366.61921929632433
Kuartil tengah : 423.43837202443706
Kuartil atas   : 482.2097724598859
----------------------------------------
IQR : 115.5905531635616
----------------------------------------
Skewness : 0.26801233302645316
----------------------------------------
Kurtosis:  -0.23720600574806516
----------------------------------------
```

1.7 Kolom Organic Carbon

In [8]:

```
print("========================================")
print("Deskripsi Statistika Kolom Organic Carbon")
print("========================================")
col_organic_carbon = df["OrganicCarbon"]

# mean
print("Mean :", col_organic_carbon.mean())
print("----------------------------------------")

# median
print("Median :", col_organic_carbon.median())
print("----------------------------------------")

# modus
if (len(col_organic_carbon.mode()) != 2010):
    print("Terdapat "+ str(len(col_organic_carbon.mode())) + " buah modus, yaitu:")
    for i in range (len(col_organic_carbon.mode()) - 1):
        print(str(col_organic_carbon.mode()[i]) + ", ", end="")
    print(str(col_organic_carbon.mode()[len(col_organic_carbon.mode()) - 1]))
else:
    print("Terdapat 2010 buah modus")
    print("Salah satu modusnya adalah:")
    print(col_ph.mode()[0])
    print("Hal ini mengimplementasikan")
    print("bahwa semua data pada")
    print("kolom ini bersifat unik")
print("----------------------------------------")

# standar deviasi
```

```
print("Stander deviasi :", col_organic_carbon.std())
print("----------------------------------------")

# variansi
print("Variansi :", col_organic_carbon.var())
print("----------------------------------------")

# range
range_data = col_organic_carbon.max() - col_organic_carbon.min()
print("Range :", range_data)
print("----------------------------------------")

# nilai minimum
print("Minimum :", col_organic_carbon.min())
print("----------------------------------------")

# maksimum
print("Maksimum :", col_organic_carbon.max())
print("----------------------------------------")

# kuartil
print("Kuartil bawah  :", col_organic_carbon.quantile(.25) )
print("Kuartil tengah :", col_organic_carbon.quantile(.50) )
print("Kuartil atas   :" , col_organic_carbon.quantile(.75) )
print("----------------------------------------")

# IQR
iqr_data = col_organic_carbon.quantile(.75) - col_organic_carbon.quantile(.25)
print("IQR :", iqr_data)
print("----------------------------------------")

# skewness
print("Skewness :", col_organic_carbon.skew())
print("----------------------------------------")

# kurtosis
print("Kurtosis: ", col_organic_carbon.kurtosis())
print("----------------------------------------")
```

```
========================================
Deskripsi Statistika Kolom Organic Carbon
========================================
Mean : 14.357939902048074
----------------------------------------
Median : 14.323285610653329
----------------------------------------
Terdapat 2010 buah modus
Salah satu modusnya adalah:
0.2274990502021987
Hal ini mengimplementasikan
bahwa semua data pada
kolom ini bersifat unik
----------------------------------------
Stander deviasi : 3.3257700016987197
----------------------------------------
Variansi : 11.0607461041991
----------------------------------------
Range : 24.80670661116602
----------------------------------------
Minimum : 2.1999999999999886
----------------------------------------
Maksimum : 27.00670661116601
----------------------------------------
Kuartil bawah  : 12.122530374047727
Kuartil tengah : 14.323285610653329
Kuartil atas   : 16.683561746173808
----------------------------------------
IQR : 4.561031372126081
----------------------------------------
Skewness : -0.02021975629181238
----------------------------------------
Kurtosis:  0.031018388192253
```

1.8 Kolom Trihalomethanes

In [9]:

```
print("=========================================")
print("Deskripsi Statistika Kolom Trihalomethanes")
print("=========================================")
col_trihalomethanes = df["Trihalomethanes"]

# mean
print("Mean :", col_trihalomethanes.mean())
print("-----------------------------------------")

# median
print("Median :", col_trihalomethanes.median())
print("-----------------------------------------")

# modus
if (len(col_trihalomethanes.mode()) != 2010):
    print("Terdapat "+ str(len(col_trihalomethanes.mode())) + " buah modus, yaitu:")
    for i in range (len(col_trihalomethanes.mode()) - 1):
        print(str(col_trihalomethanes.mode()[i]) + ", ", end="")
    print(str(col_trihalomethanes.mode()[len(col_trihalomethanes.mode()) - 1]))
else:
    print("Terdapat 2010 buah modus")
    print("Salah satu modusnya adalah:")
    print(col_ph.mode()[0])
    print("Hal ini mengimplementasikan")
    print("bahwa semua data pada")
    print("kolom ini bersifat unik")
print("-----------------------------------------")

# standar deviasi
print("Stander deviasi :", col_trihalomethanes.std())
print("-----------------------------------------")

# variansi
print("Variansi :", col_trihalomethanes.var())
print("-----------------------------------------")

# range
range_data = col_trihalomethanes.max() - col_trihalomethanes.min()
print("Range :", range_data)
print("-----------------------------------------")

# nilai minimum
print("Minimum :", col_trihalomethanes.min())
print("-----------------------------------------")

# maksimum
print("Maksimum :", col_trihalomethanes.max())
print("-----------------------------------------")

# kuartil
print("Kuartil bawah  :", col_trihalomethanes.quantile(.25) )
print("Kuartil tengah :", col_trihalomethanes.quantile(.50) )
print("Kuartil atas   :" , col_trihalomethanes.quantile(.75) )
print("-----------------------------------------")

# IQR
iqr_data = col_trihalomethanes.quantile(.75) - col_trihalomethanes.quantile(.25)
print("IQR :", iqr_data)
print("-----------------------------------------")

# skewness
print("Skewness :", col_trihalomethanes.skew())
print("-----------------------------------------")
```

```
# kurtosis
print("Kurtosis: ", col_trihalomethanes.kurtosis())
print("----------------------------------------")
```

```
========================================
Deskripsi Statistika Kolom Trihalomethanes
========================================
Mean : 66.40071666307466
----------------------------------------
Median : 66.48204080309809
----------------------------------------
Terdapat 2010 buah modus
Salah satu modusnya adalah:
0.2274990502021987
Hal ini mengimplementasikan
bahwa semua data pada
kolom ini bersifat unik
----------------------------------------
Stander deviasi : 16.08110898232513
----------------------------------------
Variansi : 258.60206610141796
----------------------------------------
Range : 115.4229870670162
----------------------------------------
Minimum : 8.577012932983806
----------------------------------------
Maksimum : 124.0
----------------------------------------
Kuartil bawah  : 55.94999302803186
Kuartil tengah : 66.48204080309809
Kuartil atas   : 77.2946128060674
----------------------------------------
IQR : 21.344619778035543
----------------------------------------
Skewness : -0.05138268451619478
----------------------------------------
Kurtosis:  0.2230167810639787
----------------------------------------
```

1.9 Kolom Turbidity

In [10]:

```
print("====================================")
print("Deskripsi Statistika Kolom Turbidity")
print("====================================")
col_turbidity = df["Turbidity"]

# mean
print("Mean :", col_turbidity.mean())
print("------------------------------------")

# median
print("Median :", col_turbidity.median())
print("------------------------------------")

# modus
if (len(col_turbidity.mode()) != 2010):
    print("Terdapat "+ str(len(col_turbidity.mode())) + " buah modus, yaitu:")
    for i in range (len(col_turbidity.mode()) - 1):
        print(str(col_turbidity.mode()[i]) + ", ", end="")
    print(str(col_turbidity.mode()[len(col_turbidity.mode()) - 1]))
else:
    print("Terdapat 2010 buah modus")
    print("Salah satu modusnya adalah:")
    print(col_ph.mode()[0])
    print("Hal ini mengimplementasikan")
    print("bahwa semua data pada")
    print("kolom ini bersifat unik")
print("------------------------------------")
```

```
# standar deviasi
print("Stander deviasi :", col_turbidity.std())
print("-----------------------------------")

# variansi
print("Variansi :", col_turbidity.var())
print("-----------------------------------")

# range
range_data = col_turbidity.max() - col_turbidity.min()
print("Range :", range_data)
print("-----------------------------------")

# nilai minimum
print("Minimum :", col_turbidity.min())
print("-----------------------------------")

# maksimum
print("Maksimum :", col_turbidity.max())
print("-----------------------------------")

# kuartil
print("Kuartil bawah  :", col_turbidity.quantile(.25) )
print("Kuartil tengah :", col_turbidity.quantile(.50) )
print("Kuartil atas   :" , col_turbidity.quantile(.75) )
print("-----------------------------------")

# IQR
iqr_data = col_turbidity.quantile(.75) - col_turbidity.quantile(.25)
print("IQR :", iqr_data)
print("-----------------------------------")

# skewness
print("Skewness :", col_turbidity.skew())
print("-----------------------------------")

# kurtosis
print("Kurtosis: ", col_turbidity.kurtosis())
print("-----------------------------------")
```

```
===================================
Deskripsi Statistika Kolom Turbidity
===================================
Mean : 3.9694969126303676
-----------------------------------
Median : 3.967373963531836
-----------------------------------
Terdapat 2010 buah modus
Salah satu modusnya adalah:
0.2274990502021987
Hal ini mengimplementasikan
bahwa semua data pada
kolom ini bersifat unik
-----------------------------------
Stander deviasi : 0.7804710407083957
-----------------------------------
Variansi : 0.6091350453844462
-----------------------------------
Range : 5.044748555990993
-----------------------------------
Minimum : 1.45
-----------------------------------
Maksimum : 6.494748555990993
-----------------------------------
Kuartil bawah  : 3.442881623557439
Kuartil tengah : 3.967373963531836
Kuartil atas   : 4.5146627202018825
-----------------------------------
IQR : 1.0717810966444437
-----------------------------------
Skewness : -0.03226597968019271
```

```
-----------------------------------
Kurtosis:  -0.049830796949249745
-----------------------------------
```

2. Visualisasi Plot Distribusi

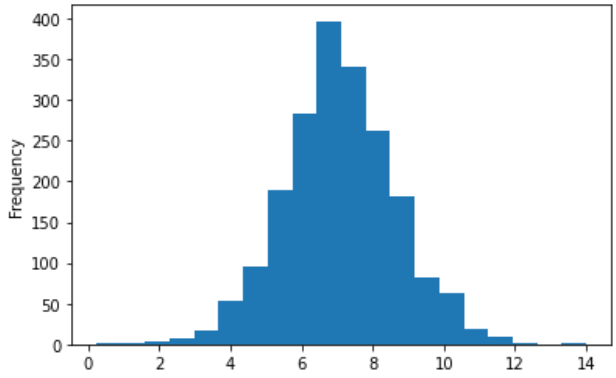
2.1 Kolom pH

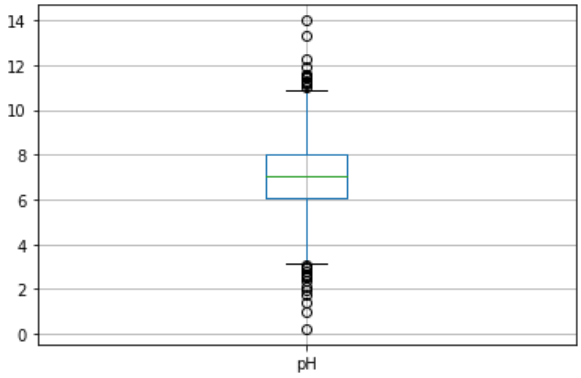
In [11]:

```
col_ph = df["pH"]
plt.subplots_adjust(left=2, right=4, wspace=0.5)
plt.subplot(121)
col_ph.plot(kind="hist", bins=20)
plt.subplot(122)
df.boxplot(column="pH")
```

Out[11]:

<AxesSubplot:>





Dari kedua buah grafik di atas, dapat disimpulkan bahwa:

1. **Pada kolom pH, *skewnessnya* bersifat *normal***
2. **Terdapat data pencilan pada kolom pH, baik itu merupakan pencilan atas maupun pencilan bawah**

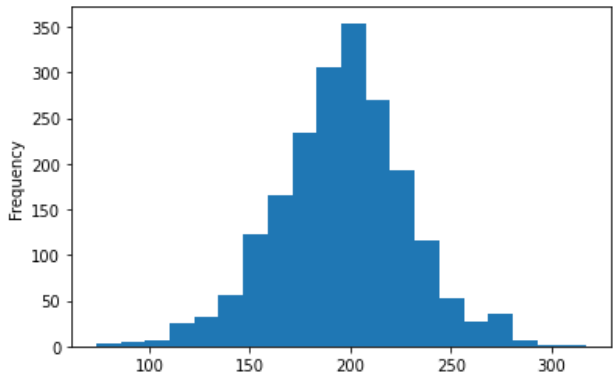
2.2 Kolom Hardness

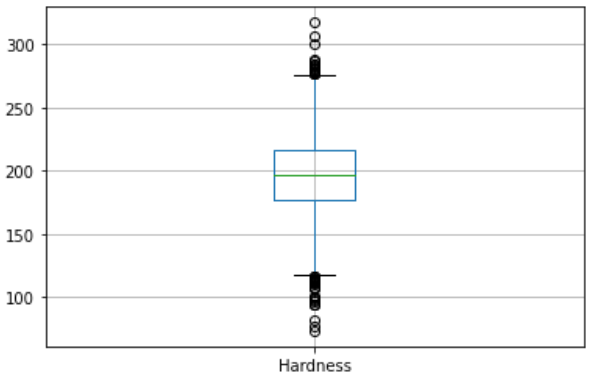
In [12]:

```
col_hardness = df["Hardness"]
plt.subplots_adjust(left=2, right=4, wspace=0.5)
plt.subplot(121)
col_hardness.plot(kind="hist", bins=20)
plt.subplot(122)
df.boxplot(column="Hardness")
```

Out[12]:

<AxesSubplot:>





Dari kedua buah grafik di atas, dapat disimpulkan bahwa:

1. **Pada kolom Hardness, *skewnessnya* bersifat *normal***
2. **Terdapat data pencilan pada kolom Hardness, baik itu merupakan pencilan atas maupun pencilan bawah**

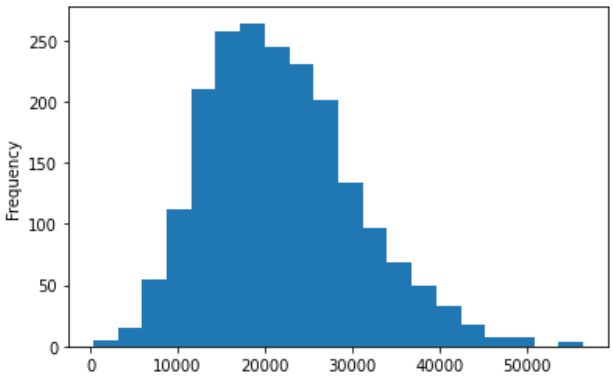
2.3 Kolom Solids

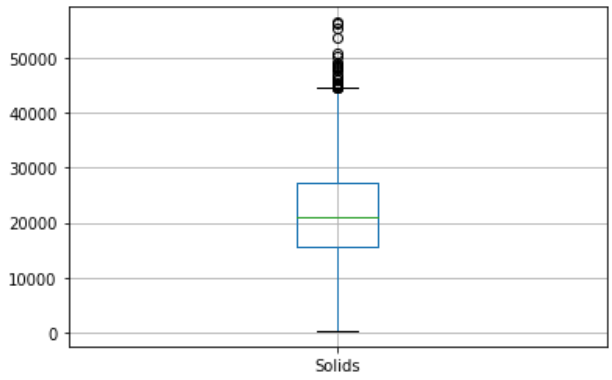
In [13]:

```
col_solids = df["Solids"]
plt.subplots_adjust(left=2, right=4, wspace=0.5)
plt.subplot(121)
col_solids.plot(kind="hist", bins=20)
plt.subplot(122)
df.boxplot(column="Solids")
```

Out[13]:

<AxesSubplot:>





Dari kedua buah grafik di atas, dapat disimpulkan bahwa:

1. **Pada kolom Solids, *skewnessnya* bersifat *skewed right***
2. **Terdapat data pencilan pada kolom Solids, yaitu data pencilan atas**

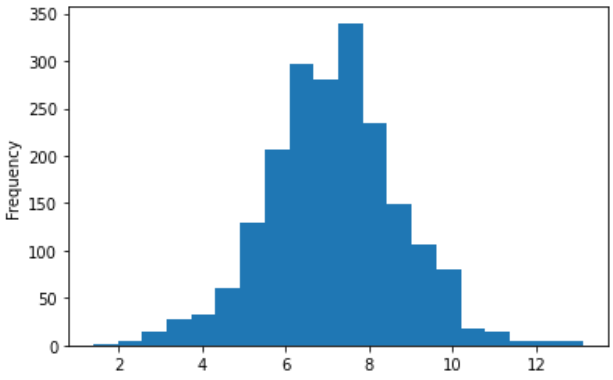
2.4 Kolom Chloramines

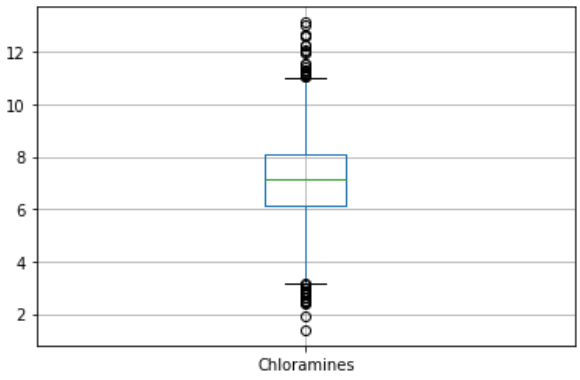
In [14]:

```
col_chloramines = df["Chloramines"]
plt.subplots_adjust(left=2, right=4, wspace=0.5)
plt.subplot(121)
col_chloramines.plot(kind="hist", bins=20)
plt.subplot(122)
df.boxplot(column="Chloramines")
```

Out[14]:

<AxesSubplot:>





Dari kedua buah grafik di atas, dapat disimpulkan bahwa:

1. **Pada kolom Chloramines, *skewnessnya* bersifat *normal***
2. **Terdapat data pencilan pada kolom Chloramines, baik itu merupakan pencilan atas maupun pencilan bawah**

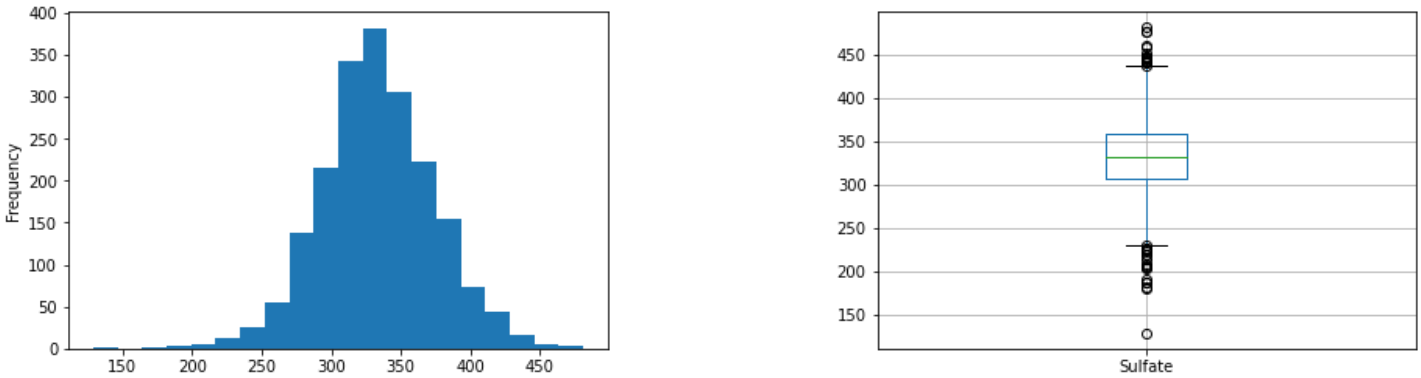
2.5 Kolom Sulfate

In [15]:

```
col_sulfate = df["Sulfate"]
plt.subplots_adjust(left=2, right=4, wspace=0.5)
plt.subplot(121)
col_sulfate.plot(kind="hist", bins=20)
plt.subplot(122)
df.boxplot(column="Sulfate")
```

Out[15]:

<AxesSubplot:>



Pada kedua buah grafik di atas, dapat disimpulkan bahwa:

1. **Pada kolom Sulfate, *skewnessnya* bersifat *normal***
2. **Terdapat data pencilan pada kolom Sulfate, baik itu merupakan pencilan atas maupun pencilan bawah**

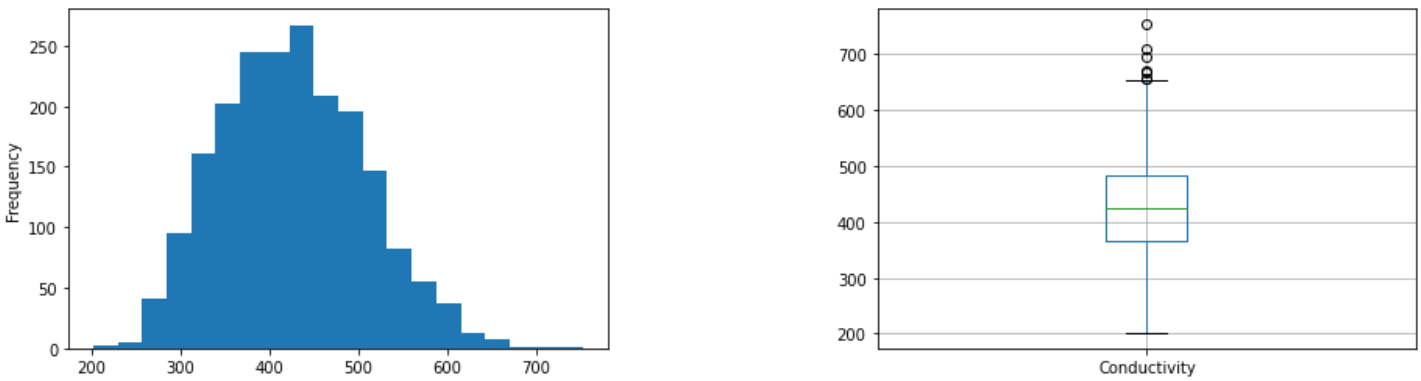
2.6 Kolom Conductivity

In [16]:

```
col_conductivity = df["Conductivity"]
plt.subplots_adjust(left=2, right=4, wspace=0.5)
plt.subplot(121)
col_conductivity.plot(kind="hist", bins=20)
plt.subplot(122)
df.boxplot(column="Conductivity")
```

Out[16]:

<AxesSubplot:>



Pada kedua buah grafik di atas, dapat disimpulkan bahwa:

Pada kedua buah grafik di atas, dapat disimpulkan bahwa:

1. **Pada kolom Conductivity, *skewnessnya* bersifat *skewed right***
2. **Terdapat data pencilan pada kolom Conductivity, yaitu data pencilan atas**

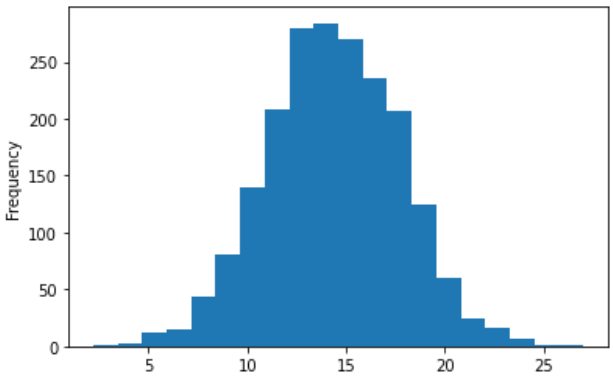
2.7 Organic Carbon

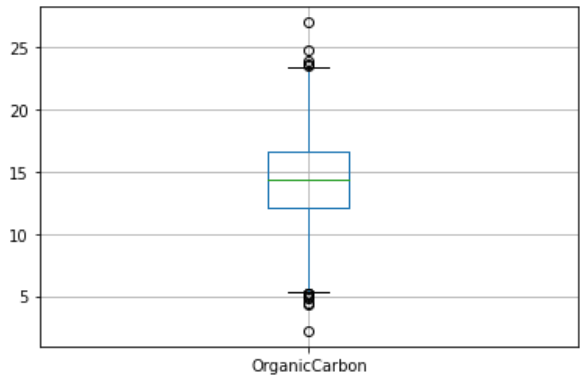
In [17]:

```
col_organic_carbon= df["OrganicCarbon"]
plt.subplots_adjust(left=2, right=4, wspace=0.5)
plt.subplot(121)
col_organic_carbon.plot(kind="hist", bins=20)
plt.subplot(122)
df.boxplot(column="OrganicCarbon")
```

Out[17]:

<AxesSubplot:>





Pada kedua buah grafik di atas, dapat disimpulkan bahwa:

1. **Pada kolom Organic Carbon, *skewnessnya* bersifat *normal***
2. **Terdapat data pencilan pada kolom Organic Carbon, baik itu merupakan pencilan atas maupun pencilan bawah**

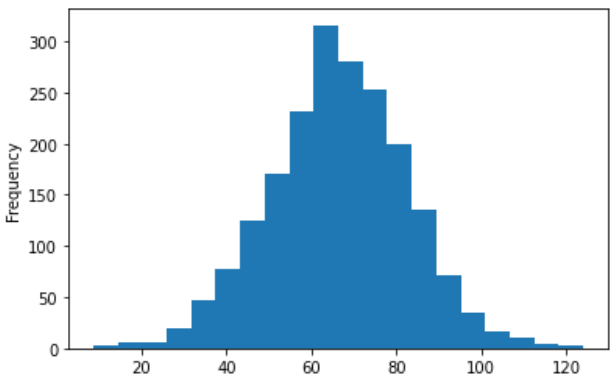
2.8 Kolom Trihalomethanes

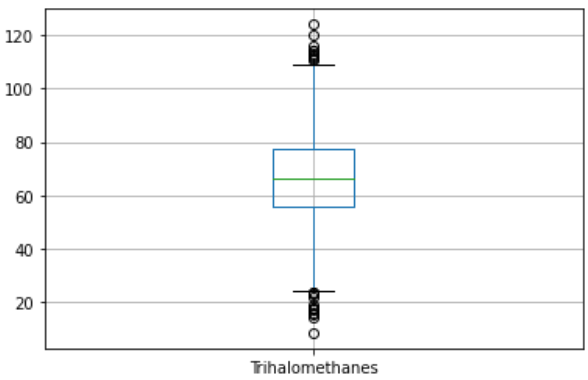
In [18]:

```
col_trihalomethanes= df["Trihalomethanes"]
plt.subplots_adjust(left=2, right=4, wspace=0.5)
plt.subplot(121)
col_trihalomethanes.plot(kind="hist", bins=20)
plt.subplot(122)
df.boxplot(column="Trihalomethanes")
```

Out[18]:

<AxesSubplot:>





Pada kedua buah grafik di atas, dapat disimpulkan bahwa:

1. **Pada kolom Trihalomethanes, *skewnessnya* bersifat *normal***
2. **Terdapat data pencilan pada kolom Trihalomethanes, baik itu merupakan pencilan atas maupun pencilan bawah**

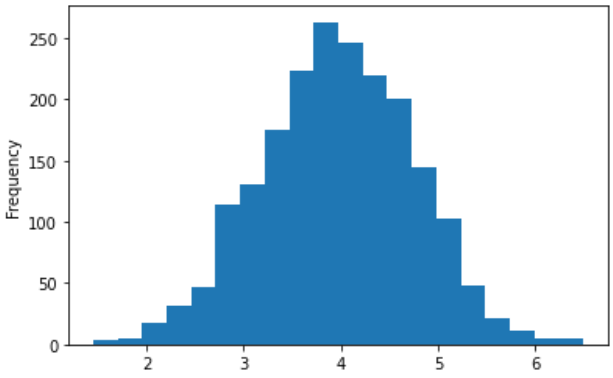
2.9 Kolom Turbidity

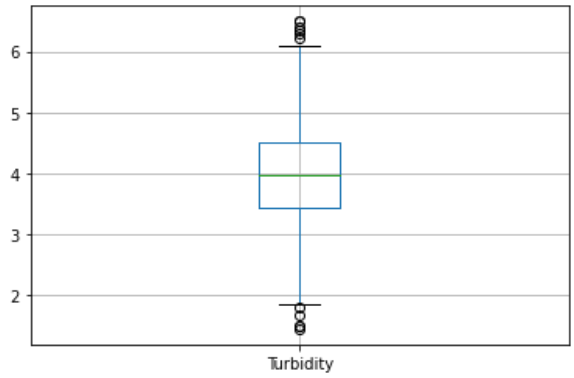
In [19]:

```
col_turbidity= df["Turbidity"]
plt.subplots_adjust(left=2, right=4, wspace=0.5)
plt.subplot(121)
col_turbidity.plot(kind="hist", bins=20)
plt.subplot(122)
df.boxplot(column="Turbidity")
```

Out[19]:

<AxesSubplot:>





Pada kedua buah grafik di atas, dapat disimpulkan bahwa:

1. **Pada kolom Turbility, *skewnessnya* bersifat *normal***
2. **Terdapat data pencilan pada kolom Turbility, baik itu merupakan pencilan atas maupun pencilan bawah**

3. Tes Distribusi Normal (Normality Test)

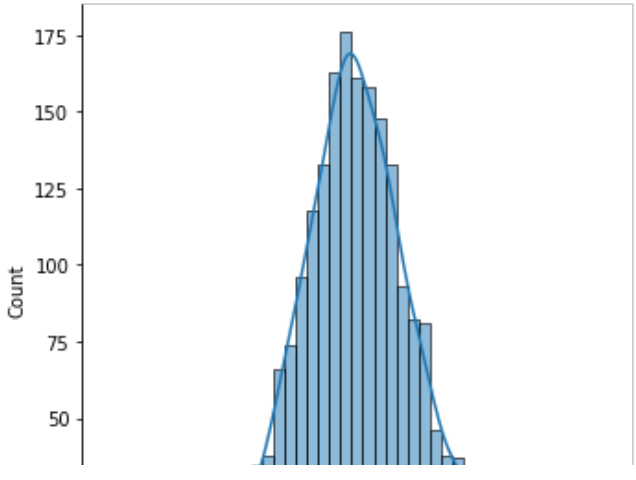
3.1 Kolom pH

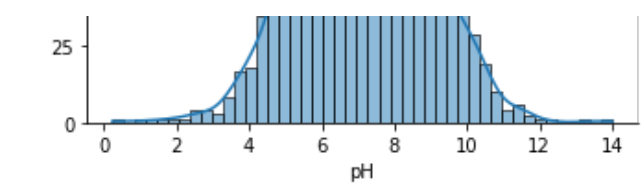
In [20]:

```
sns.displot(data = df, x = "pH", kde = True)
```

Out[20]:

<seaborn.axisgrid.FacetGrid at 0x1348c021be0>





Histogram di atas menunjukkan bahwa data pada kolom pH tidak terdistribusi secara normal. Hal ini dapat dilihat dari bentuk histogram yang tidak menyerupai bell curve.

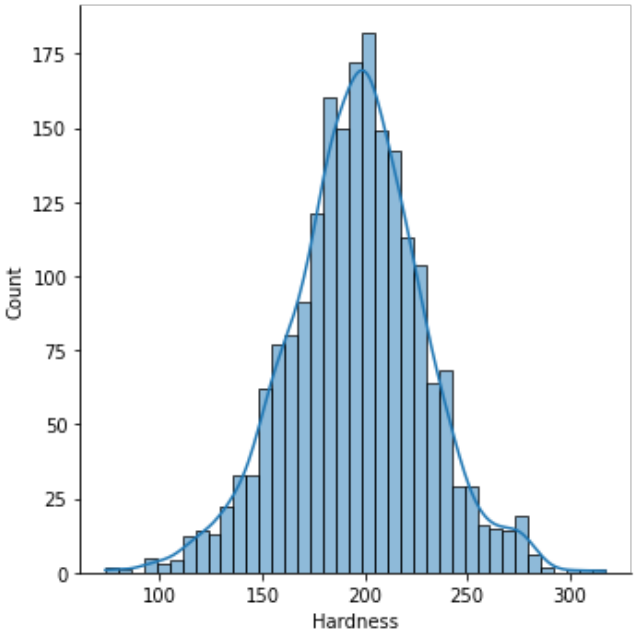
3.2 Kolom Hardness

In [21]:

```
sns.displot(data = df, x = "Hardness", kde = True)
```

Out[21]:

```
<seaborn.axisgrid.FacetGrid at 0x1348c3cac40>
```



Histogram di atas menunjukkan bahwa data pada kolom Hardness tidak terdistribusi secara normal. Hal ini dapat dilihat dari bentuk histogram yang tidak menyerupai bell curve.

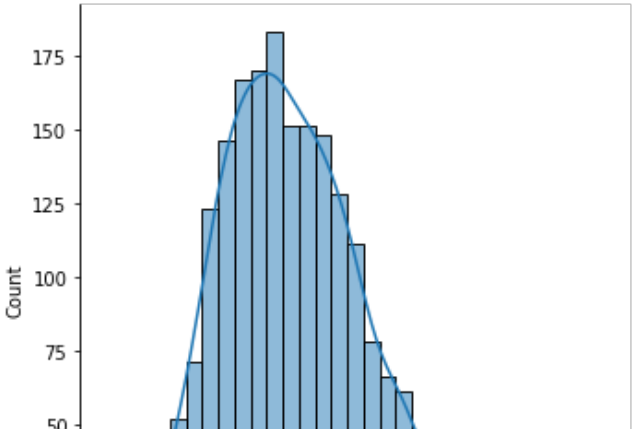
3.3 Kolom Solids

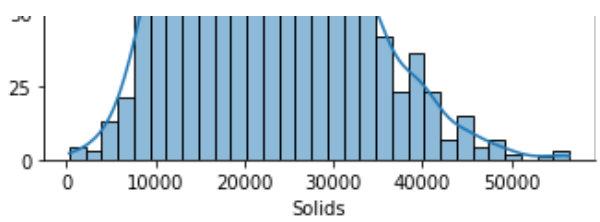
In [22]:

```
sns.displot(data = df, x = "Solids", kde = True)
```

Out[22]:

```
<seaborn.axisgrid.FacetGrid at 0x1348da00e50>
```





Histogram di atas menunjukkan bahwa data pada kolom Solids tidak terdistribusi secara normal. Hal ini dapat dilihat dari bentuk histogram yang tidak menyerupai bell curve.

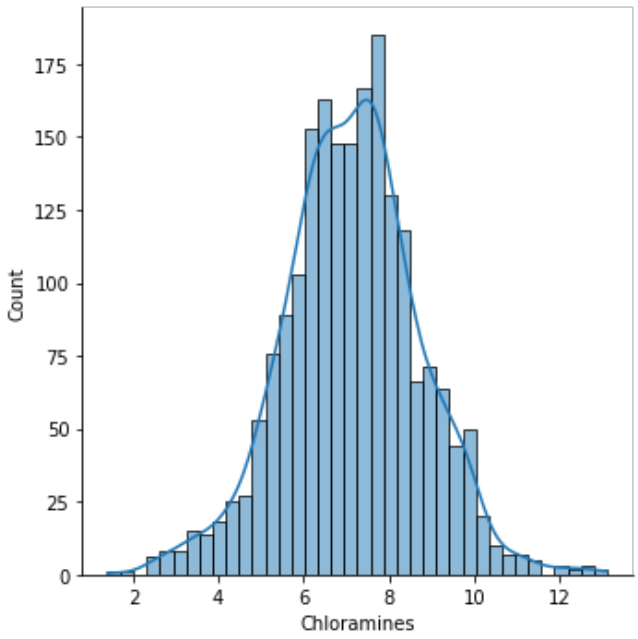
3.4 Kolom Chloramines

In [23]:

```
sns.displot(data = df, x = "Chloramines", kde = True)
```

Out[23]:

```
<seaborn.axisgrid.FacetGrid at 0x1348da163d0>
```



Histogram di atas menunjukkan bahwa data pada kolom Chloramines tidak terdistribusi secara normal. Hal ini dapat dilihat dari bentuk histogram yang tidak menyerupai bell curve.

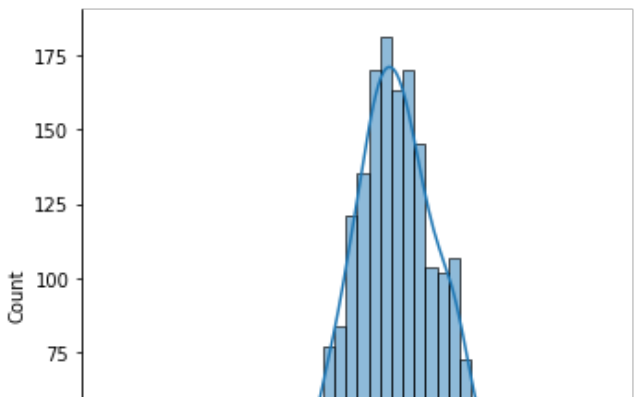
3.5 Kolom Sulfate

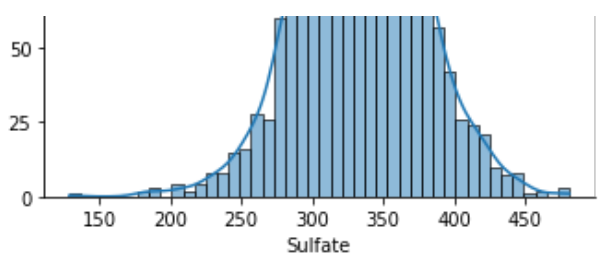
In [24]:

```
sns.displot(data = df, x = "Sulfate", kde = True)
```

Out[24]:

```
<seaborn.axisgrid.FacetGrid at 0x1348da16250>
```





Histogram di atas menunjukkan bahwa data pada kolom Sulfate tidak terdistribusi secara normal. Hal ini dapat dilihat dari bentuk histogram yang tidak menyerupai bell curve.

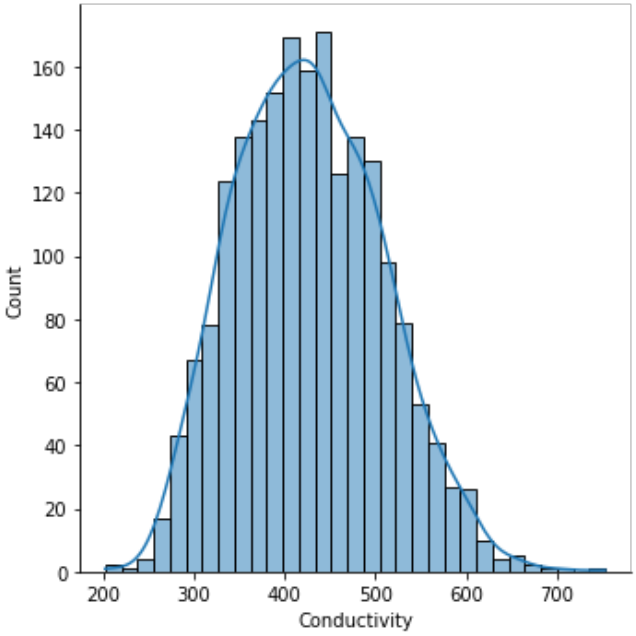
3.6 Kolom Conductivity

In [25]:

```
sns.displot(data = df, x = "Conductivity", kde = True)
```

Out[25]:

```
<seaborn.axisgrid.FacetGrid at 0x1348dc61be0>
```



Histogram di atas menunjukkan bahwa data pada kolom Conductivity tidak terdistribusi secara normal. Hal ini dapat dilihat dari bentuk histogram yang tidak menyerupai bell curve.

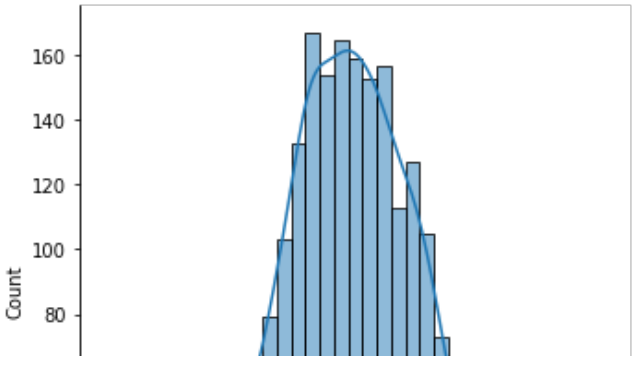
3.7 Kolom Organic Carbon

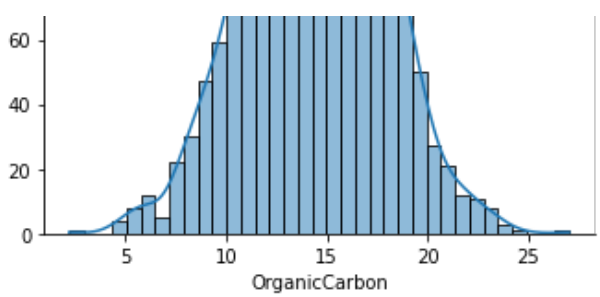
In [26]:

```
sns.displot(data = df, x = "OrganicCarbon", kde = True)
```

Out[26]:

```
<seaborn.axisgrid.FacetGrid at 0x1348dd88520>
```





Histogram di atas menunjukkan bahwa data pada kolom OrganicCarbon terdistribusi secara normal. Hal ini dapat dilihat dari bentuk histogram yang menyerupai bell curve.

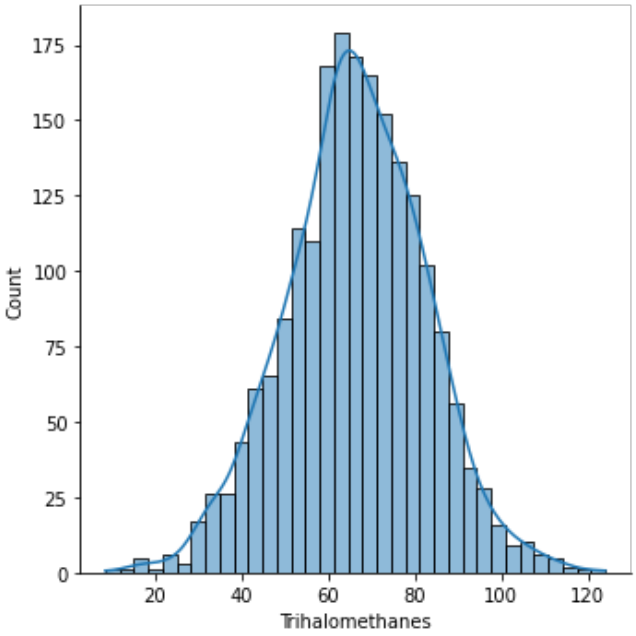
3.8 Kolom Trihalomethanes

In [27]:

```
sns.displot(data = df, x = "Trihalomethanes", kde = True)
```

Out[27]:

```
<seaborn.axisgrid.FacetGrid at 0x1348d4e3e50>
```



Histogram di atas menunjukkan bahwa data pada kolom Trihalomethanes terdistribusi secara normal. Hal ini dapat dilihat dari bentuk histogram yang menyerupai bell curve.

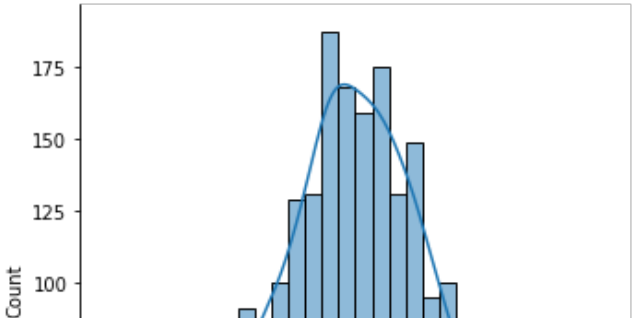
3.9 Kolom Turbidity

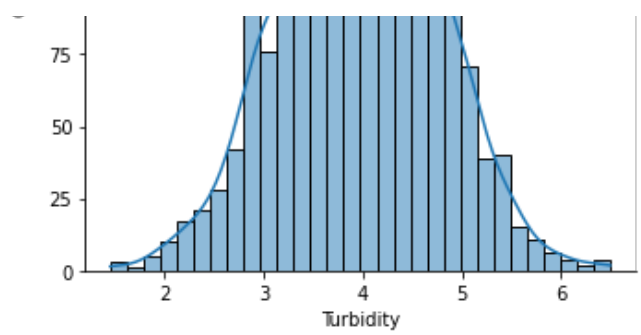
In [28]:

```
sns.displot(data = df, x = "Turbidity", kde = True)
```

Out[28]:

```
<seaborn.axisgrid.FacetGrid at 0x1348db070d0>
```





Histogram di atas menunjukkan bahwa data pada kolom Turbidity terdistribusi secara normal. Hal ini dapat dilihat dari bentuk histogram yang menyerupai bell curve.

4. Tes Hipotesis Satu Sampel

In [29]:

```
def tvalue(x, mean, std, n):
        return (x-mean)*math.sqrt(n)/std

def binomial(p1,p0, n):
        q0 = 1 - p0
        return(p1-p0) / math.sqrt(p0*q0/n)
```

4.1 Nilai rata-rata pH di atas 7

1. **Tentukan hipotesis nol**

   H_0 : μ = 7
2. **Tentukan hipotesis alternatif**

   H_1 : μ > 7
3. **Tentukan tingkat signifikan**

   α = 0.05
4. **Tentukan uji statistik yang sesuai dan tentukan daerah kritis.**

   Uji statistik : One Tail Mean Test

   Daerah kritis : t > t_α **: t > 1.645**
5. **Hitung nilai uji statistik**
6. **t >** t_α**, maka tolak hipotesis null. Jadi rata-rata pH di atas 7.**

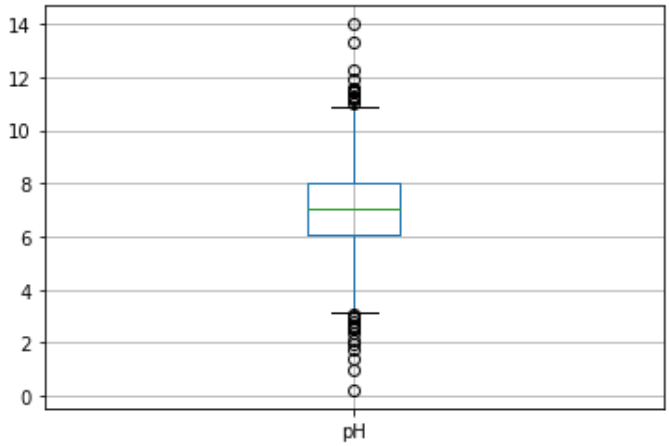
In [30]:

```
mean = 7
rataan = df['pH'].mean()
std = df['pH'].std()
n = df['pH'].count()

t =  tvalue(rataan, mean, std, n)
talpha = s.t.ppf(0.95, n - 1)
print("Nilai t : " +str(t))
print("Nilai t-alpha : " +str(talpha))

pValue = s.norm.sf(abs(t))
print("\nNilai P-value :" , pValue)
if (pValue > 0.05) :
    print("H0 gagal ditolak")
else :
    print("H0 ditolak")

df.boxplot(column="pH")
```

```
Nilai t : 2.485445147379887
Nilai t-alpha : 1.6456124504017113

Nilai P-value : 0.006469476288896462
H0 ditolak
```

Out[30]:

```
<AxesSubplot:>
```



4.2 Nilai rata-rata Hardness tidak sama dengan 205

1. **Tentukan hipotesis nol**

   H_0 : μ = 205
2. **Tentukan hipotesis alternatif**

   H_1 : $\mu \neq$ 205
3. **Tentukan tingkat signifikan**

   α = 0.05
4. **Tentukan uji statistik yang sesuai dan tentukan daerah kritis.**

   Uji statistik : Two-Tailed Mean Test

   Daerah kritis : t > $t_{\alpha/2}$ **or t <** $-t_{\alpha/2}$

   : t > 1.96 or t < -1.96
5. **Hitung nilai uji statistik**
6. **t <** $-t_{\alpha/2}$**, maka tolak hipotesis null. Jadi rata-rata Hardness tidak sama dengan 205 .**

In [31]:

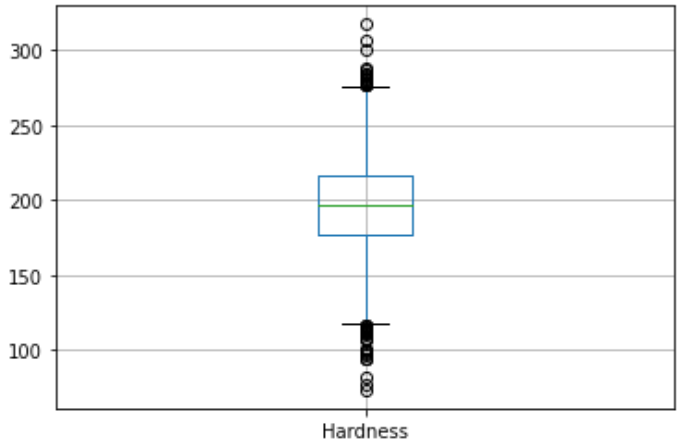
```
mean = 205
rataan = df['Hardness'].mean()
std = df['Hardness'].std()
n = df['Hardness'].count()

print("Nilai t : " + str(tvalue(rataan, mean, std, n)))
print("Nilai t(alpha/2) : " + str(s.t.ppf(0.975, n - 1)))

pValue = s.norm.sf(abs(tvalue(rataan, mean, std, n)))
print("\nNilai P-value :" , pValue)
if (pValue > 0.05) :
    print("H0 gagal ditolak")
else :
    print("H0 ditolak")

df.boxplot(column="Hardness")
```

```
Nilai t : -12.403137170010732
Nilai t(alpha/2) : 1.9611455060885261
```

```
Nilai P-value : 1.2564452447572327e-35
H0 ditolak
```

Out[31]:

```
<AxesSubplot:>
```



4.3 Nilai rata-rata 100 baris pertama kolom Solids bukan 21900

1. **Tentukan hipotesis nol**

   H_0 : μ = 21900
2. **Tentukan hipotesis alternatif**

   H_1 : $\mu \neq$ 21900
3. **Tentukan tingkat signifikan**

   α = 0.05
4. **Tentukan uji statistik yang sesuai dan tentukan daerah kritis.**

   Uji statistik : Two Tailed Mean Test

   Daerah kritis : t > $t_{\alpha/2}$ **or t < -** $t_{\alpha/2}$

   : t > 1.98 or t < -1.98
5. **Hitung nilai uji statistik**
6. **t tidak berada pada daerah kritis. Hipotesis nol gagal ditolak sehingga tidak cukup bukti untuk menyimpulkan bahwa rata-rata 100 baris pertama kolom Solids bukan 21900.**

In [50]:

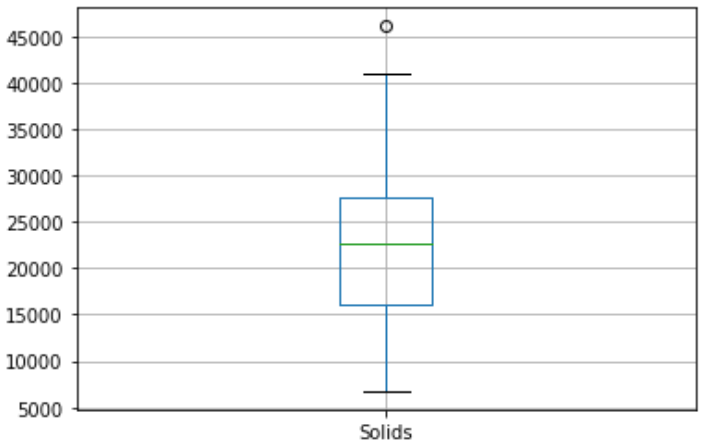
```
sampel = df['Solids'].head(100)
mean = 21900
rataan = sampel.mean()
std = sampel.std()
n = sampel.size

print("Nilai t : " + str(tvalue(rataan, mean, std, n)))
print("Nilai t(alpha/2) : " + str(s.t.ppf(0.975, n - 1)))

pValue = 2*s.norm.sf(abs(tvalue(rataan, mean, std, n)))
print("\nNilai P-value :" , pValue)
if (pValue > 0.05) :
    print("H0 gagal ditolak")
else :
    print("H0 ditolak")

df.head(100).boxplot(column="Solids")
```

```
Nilai t : 0.5636797715721551
Nilai t(alpha/2) : 1.9842169515086827
```

```
Nilai P-value : 0.5729720864655174
H0 gagal ditolak
```

Out[50]:

```
<AxesSubplot:>
```



4.4 Proporsi nilai Conductivity yang lebih dari 450, adalah tidak sama dengan 10%

1. **Tentukan hipotesis nol**

   H_0 : p = 0.1
2. **Tentukan hipotesis alternatif**

   H_1 : $p \neq$ 0.1
3. **Tentukan tingkat signifikan**

   α = 0.05
4. **Tentukan uji statistik yang sesuai dan tentukan daerah kritis.**

   Uji statistik : Uji Variabel Binomial X dengan p = p0

   Daerah kritis : z > $z_{\alpha/2}$ **or z < -** $z_{\alpha/2}$

   ```
   : z > 1.96 or z < -1.96
   ```
5. **z >** $z_{\alpha/2}$**. Hipotesis nol ditolak sehingga Proporsi nilai Conductivity yang lebih dari 450, adalah tidak sama dengan 10%.**

In [33]:

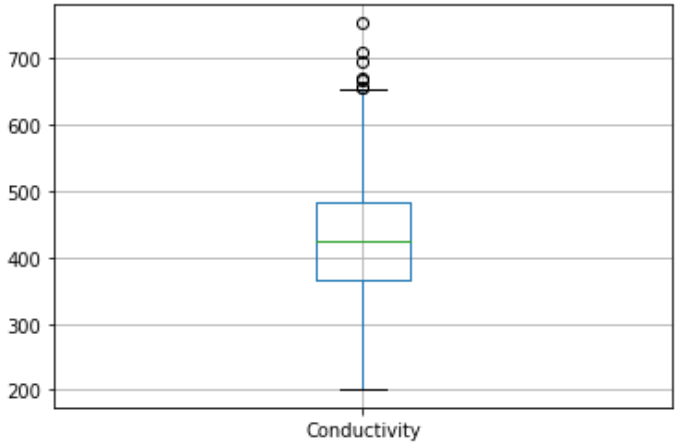
```
sampel = df['Conductivity'].loc[df['Conductivity'] > 450]
p0 = 0.1
q0 = 1 - p0
n_sampel = sampel.size
n = df['Conductivity'].size
p1 = n_sampel / n

z = (p1-p0) / math.sqrt(p0*q0/n)
zalpha = s.norm.ppf(0.975)
print("Nilai z : " + str(z))
print("Nilai z(alpha/2) : " + str(zalpha))

pValue = s.norm.sf(abs(z))
print("\nNilai P-value :" , pValue)
if (pValue > 0.05) :
    print("H0 gagal ditolak")
else :
    print("H0 ditolak")

df.boxplot(column="Conductivity")
```

```
Nilai z : 40.44637613158932
Nilai z(alpha/2) : 1.959963984540054
```

```
Nilai z(alpha/2) : 1.959963984540054

Nilai P-value : 0.0
H0 ditolak
```

Out[33]:

```
<AxesSubplot:>
```



4.5 Proporsi nilai Trihalomethanes yang kurang dari 40, adalah kurang dari 5%

1. **Tentukan hipotesis nol**

   H_0 : p = 0.05
2. **Tentukan hipotesis alternatif**

   H_1 : p < 0.05 (one-tailed test)
3. **Tentukan tingkat signifikan**

   α = 0.05
4. **Tentukan uji statistik yang sesuai dan tentukan daerah kritis.**

   Uji statistik : Uji Variabel Binomial X dengan p = p0

   Daerah kritis : z < - z_α : z < -1.645
5. **Hitung nilai uji statistik**
6. **z tidak berada pada daerah kritis. Hipotesis nol gagal ditolak sehingga tidak cukup bukti untuk menyimpulkan bahwa Proporsi nilai Trihalomethanes yang kurang dari 40, adalah kurang dari 5%**

In [51]:

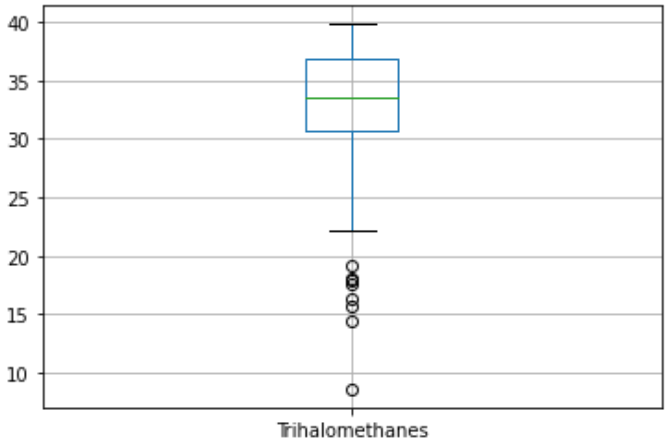
```
sampel = df['Trihalomethanes'].loc[df['Trihalomethanes'] < 40]
p0 = 0.05
q0 = 1 - p0
n_sampel = sampel.size
n = df['Trihalomethanes'].size
zalpha = s.norm.ppf(0.95)
p1 = n_sampel / n

z = (p1-p0) / math.sqrt(p0*q0/n)

print("Nilai z : " + str(z))
print("Nilai zalpha : " + str(zalpha))

pValue = s.norm.sf(abs(z))
print("\nNilai P-value :" , pValue)
if (pValue > 0.05) :
    print("H0 gagal ditolak")
else :
    print("H0 ditolak")

df.loc[df['Trihalomethanes'] < 40].boxplot(column="Trihalomethanes")
```

```
Nilai z : 0.5628826416670951
```

```
Nilai zalpha : 1.6448536269514722

Nilai P-value : 0.286757400490763
H0 gagal ditolak
```

Out[51]:

```
<AxesSubplot:>
```



5. Test Hipotesis 2 Sampel

In [35]:

```
def tvalue_twomean(d0, x1, x2, s1, s2, n1, n2):
    sp =  math.sqrt( ( (n1-1)*s1*s1  + (n2-1)*s2*s2) / (n1+n2-2))
    return((x1 - x2) - d0)/(sp*math.sqrt((1/n1) + (1/n2)))
```

5.1 Data kolom Sulfate dibagi 2 sama rata: bagian awal dan akhir kolom. Benarkah rata-rata kedua bagian tersebut sama?

1. **Tentukan hipotesis nol**

   H_0 : $\mu_1 = \mu_2$
2. **Tentukan hipotesis alternatif**

   H_1 : $\mu_1 \neq \mu_2$
3. **Tentukan tingkat signifikan**

   α = 0.05
4. **Tentukan uji statistik yang sesuai dan tentukan daerah kritis.**

   Uji statistik : Two Sample Two-Tailed Mean Test

   Daerah kritis : t < - $t_{\alpha/2}$ or t > $t_{\alpha/2}$: t < -1.96 or t > 1.96
5. **Hitung nilai uji statistik**
6. **Nilai t < -1.96, maka tolak hipotesis null. Jadi rata-rata kedua bagian tidak sama**

In [36]:

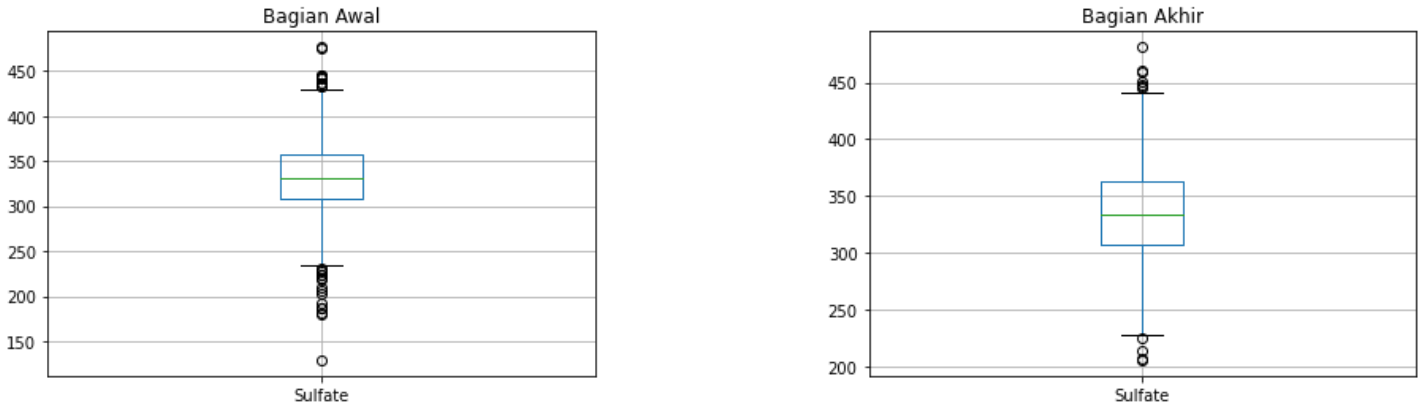
```
sampel = df['Sulfate']

d0 = 0

talpha = s.t.ppf(0.975, sampel.size - 1)
bagian1 = sampel.head(sampel.size // 2)
bagian2 = sampel.tail(sampel.size // 2)

mean1 = bagian1.mean()
mean2 = bagian2.mean()

std1 = bagian1.std()
```

```
std2 = bagian2.std()
t = tvalue_twomean(d0, mean1, mean2, std1, std2, bagian1.size, bagian2.size)

print("Nilai t : " + str(t))
print("Nilai t(alpha/2) : " + str(talpha))
pValue = 2*s.norm.sf(abs(t))
print("Nilai P-value :" , pValue)
if (pValue > 0.05) :
    print("H0 gagal ditolak")
else :
    print("H0 ditolak")

plt.subplots_adjust(left=2, right=4, wspace=0.5)
plt.subplot(121)
df.head(1005).boxplot("Sulfate").set_title("Bagian Awal")
plt.subplot(122)
df.tail(1005).boxplot("Sulfate").set_title("Bagian Akhir")
```

```
Nilai t : -2.0752690696871983
Nilai t(alpha/2) : 1.9611455060885261
Nilai P-value : 0.0379616043851286
H0 ditolak
```

Out[36]:

```
Text(0.5, 1.0, 'Bagian Akhir')
```



5.2 Data kolom Organic Carbon dibagi 2 sama rata: bagian awal dan bagian akhir kolom. Benarkah rata-rata bagian awal lebih besar dari bagian akhir sebesar 0.15?

1. **Tentukan hipotesis nol**

   H_0 : μ_1 - μ_2 = 0.15
2. **Tentukan hipotesis alternatif**

   H_1 : μ_1 - $\mu_2 \neq$ 0.15
3. **Tentukan tingkat signifikan**

   α = 0.05
4. **Tentukan uji statistik yang sesuai dan tentukan daerah kritis.**

   Uji statistik : Two Sample Two-Tailed Mean Test

   Daerah kritis : t < - $t_{\alpha/2}$ or t > $t_{\alpha/2}$: t < -1.96 or t > 1.96
5. **Hitung nilai uji statistik**
6. **Nilai t < -1.96, maka tolak hipotesis null. Jadi rata-rata bagian awal tidak lebih besar dari bagian akhir sebesar 0.15**

In [37]:

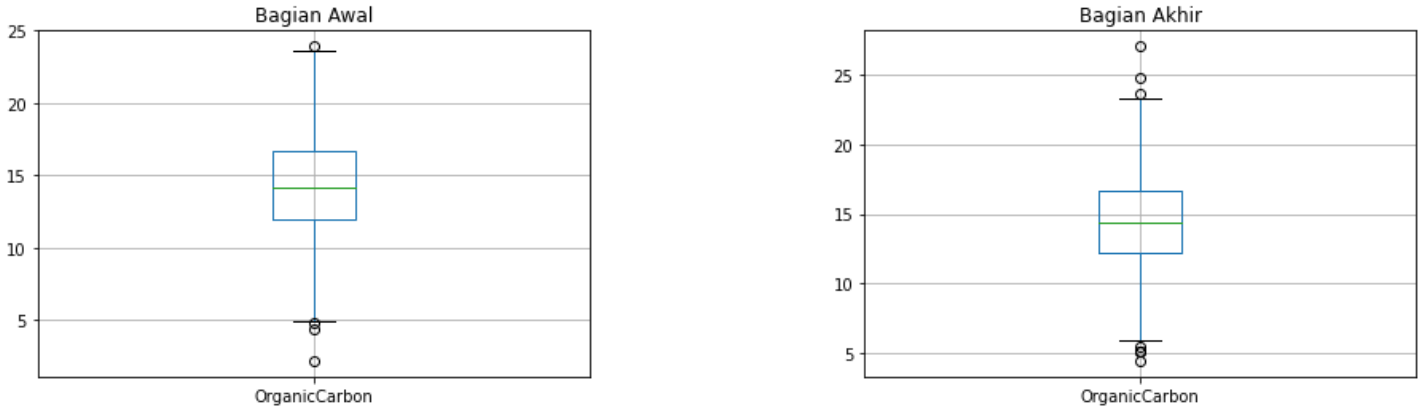
```
sampel = df['OrganicCarbon']

d0 = 0.15
```

```
talpha = s.t.ppf(0.975, sampel.size - 1)
bagian1 = sampel.head(sampel.size // 2)
bagian2 = sampel.tail(sampel.size // 2)

mean1 = bagian1.mean()
mean2 = bagian2.mean()

std1 = bagian1.std()
std2 = bagian2.std()
t = tvalue_twomean(d0, mean1, mean2, std1, std2, bagian1.size, bagian2.size)

print("Nilai t : " + str(t))
print("Nilai t(alpha/2) : " + str(talpha))
pValue = 2*s.norm.sf(abs(t))
print("Nilai P-value :" , pValue)
if (pValue > 0.05) :
    print("H0 gagal ditolak")
else :
    print("H0 ditolak")

plt.subplots_adjust(left=2, right=4, wspace=0.5)
plt.subplot(121)
df.head(1005).boxplot("OrganicCarbon").set_title("Bagian Awal")
plt.subplot(122)
df.tail(1005).boxplot("OrganicCarbon").set_title("Bagian Akhir")
```

```
Nilai t : -2.413145517798807
Nilai t(alpha/2) : 1.9611455060885261
Nilai P-value : 0.01581550381760006
H0 ditolak
```

Out[37]:

```
Text(0.5, 1.0, 'Bagian Akhir')
```



5.3 Rata-rata 100 baris pertama kolom Chloramines sama dengan 100 baris terakhirnya?

1. **Tentukan hipotesis nol**

   H_0 : μ_1 - μ_2 = 0
2. **Tentukan hipotesis alternatif**

   H_1 : $\mu_1 \neq \mu_2$
3. **Tentukan tingkat signifikan**

   α = 0.05
4. **Tentukan uji statistik yang sesuai dan tentukan daerah kritis.**

   Uji statistik : Two Sample Two-Tailed Mean Test

   Daerah kritis : t < - $t_{\alpha/2}$ or t > $t_{\alpha/2}$: t < -1.96 or t > 1.96
5. **Hitung nilai uji statistik**
6. **Nilai t tidak berada pada daerah kritis. Hipotesis nol gagal ditolak sehingga tidak cukup bukti untuk menyimpulkan bahwa Rata-rata 100 baris pertama kolom Chloramines sama dengan 100 baris terakhirnya**

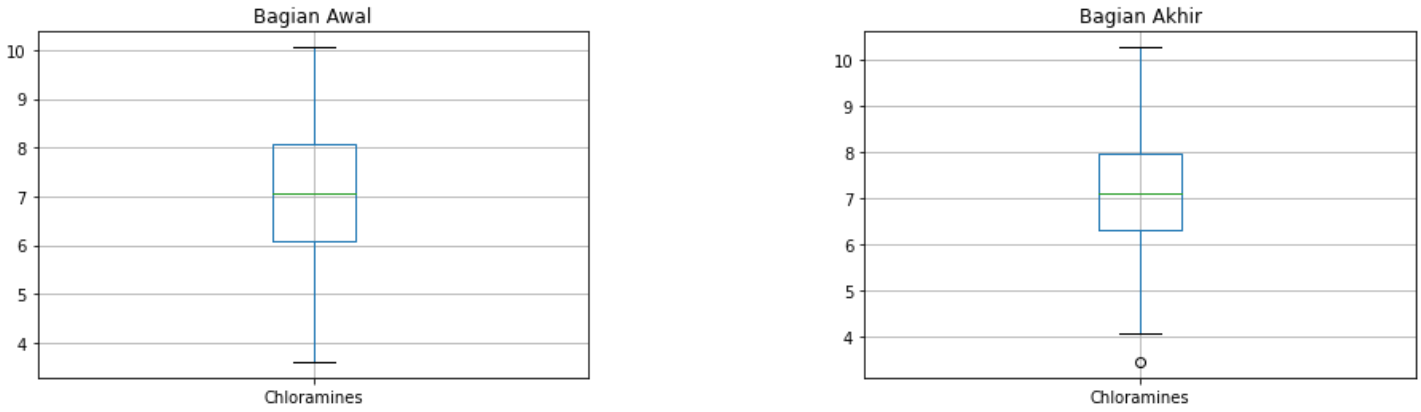
In [38]:

```
sampel = df['Chloramines']

d0 = 0

talpha = s.t.ppf(0.975, sampel.size - 1)
bagian1 = sampel.head(100)
bagian2 = sampel.tail(100)

mean1 = bagian1.mean()
mean2 = bagian2.mean()

std1 = bagian1.std()
std2 = bagian2.std()
t = tvalue_twomean(d0, mean1, mean2, std1, std2, bagian1.size, bagian2.size)

print("Nilai t : " + str(t))
print("Nilai t(alpha/2) : " + str(talpha))
pValue = 2*s.norm.sf(abs(t))
print("Nilai P-value :" , pValue)
if (pValue > 0.05) :
    print("H0 gagal ditolak")
else :
    print("H0 ditolak")

plt.subplots_adjust(left=2, right=4, wspace=0.5)
plt.subplot(121)
df.head(100).boxplot("Chloramines").set_title("Bagian Awal")
plt.subplot(122)
df.tail(100).boxplot("Chloramines").set_title("Bagian Akhir")
```

```
Nilai t : -0.7059424842236872
Nilai t(alpha/2) : 1.9611455060885261
Nilai P-value : 0.48022390604502796
H0 gagal ditolak
```

Out[38]:

```
Text(0.5, 1.0, 'Bagian Akhir')
```



5.4 Proporsi nilai bagian awal Turbility yang lebih dari 4, adalah lebih besar daripada, proporsi nilai yang sama di bagian akhir Turbility

1. **Tentukan hipotesis nol**

   H_0 : p_1 - p_2 = 0
2. **Tentukan hipotesis alternatif**

   H_1 : p_1 - p_2 > 0
3. **Tentukan tingkat signifikan**

   α = 0.05
4. **Tentukan uji statistik yang sesuai dan tentukan daerah kritis.**

Uji statistik : Uji statistik satu parameter populasi pengujian proporsi dua sampel one-tailed tes normal

Daerah kritis :z > z_α : z > 1.645

5. **Hitung nilai uji statistik**
6. **Nilai z tidak berada pada daerah kritis. Hipotesis nol gagal ditolak sehingga tidak cukup bukti untuk menyimpulkan bahwa Proporsi nilai bagian awal Turbidity yang lebih dari 4, adalah lebih besar daripada, proporsi nilai yang sama di bagian akhir Turbidity**

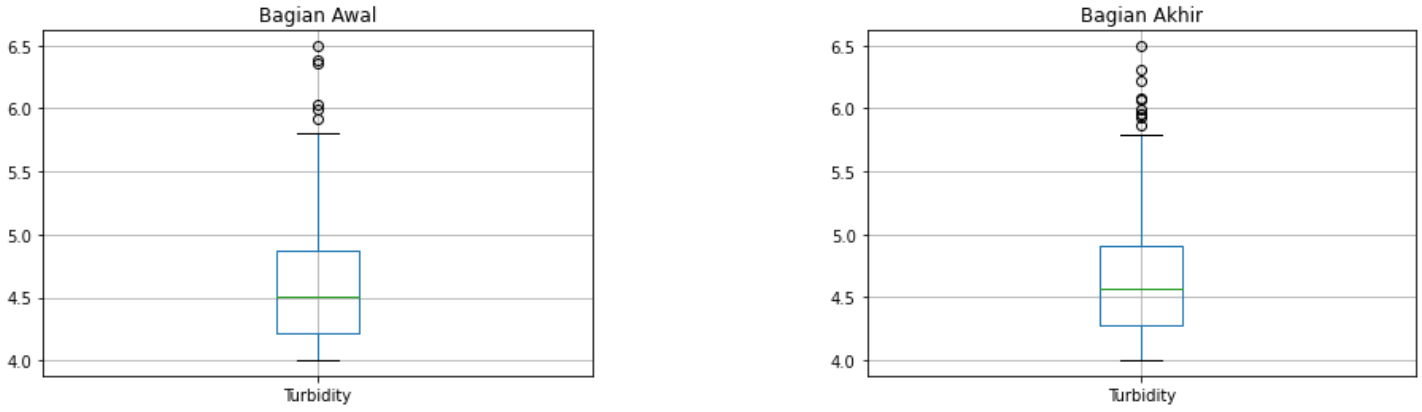
In [52]:

```
sampel = df['Turbidity']

zalpha = s.norm.ppf(0.95)
bagian1 = sampel.head(sampel.size // 2).loc[sampel > 4]
bagian2 = sampel.tail(sampel.size // 2).loc[sampel > 4]

p1 = bagian1.size/ (sampel.size //2)
p2 = bagian2.size/ (sampel.size //2)

p = (bagian1.size + bagian2.size) / sampel.size
q = 1-p

z = (p1 - p2) / (math.sqrt(p*q *( (1/(sampel.size//2)) + (1/(sampel.size//2))  )))

print("Nilai z : " + str(z))
print("Nilai z(alpha) : " + str(zalpha))

pValue = s.norm.sf(abs(z))
print("Nilai P-value :" , pValue)
if (pValue > 0.05) :
    print("H0 gagal ditolak")
else :
    print("H0 ditolak")


plt.subplots_adjust(left=2, right=4, wspace=0.5)
plt.subplot(121)
df.head(1005).loc[sampel > 4].boxplot("Turbidity").set_title("Bagian Awal")
plt.subplot(122)
df.tail(1005).loc[sampel > 4].boxplot("Turbidity").set_title("Bagian Akhir")
```

```
Nilai z : -0.13388958661778735
Nilai z(alpha) : 1.6448536269514722
Nilai P-value : 0.4467449424088169
H0 gagal ditolak
```

Out[52]:

```
Text(0.5, 1.0, 'Bagian Akhir')
```



5.5 Bagian awal kolom Sulfate memiliki variansi yang sama dengan bagian akhirnya

1. **Tentukan hipotesis nol**

$H_0 : \sigma_1^2 = \sigma_2^2$

2. **Tentukan hipotesis alternatif**

   $H_1 : \sigma_1^2 \neq \sigma_2^2$
3. **Tentukan tingkat signifikan**

   α **= 0.05**
4. **Tentukan uji statistik yang sesuai dan tentukan daerah kritis.**

   Uji statistik : Distribusi F

   Daerah kritis : f < $f_{1-\alpha/2(v1,v2)}$ **or f >** $f_{\alpha/2(v1,v2)}$ **: f < 0.883 or f > 1.131**
5. **Hitung nilai uji statistik**
6. **Nilai f tidak berada pada daerah kritis, maka terima hipotesis null. Hipotesis nol gagal ditolak sehingga tidak cukup bukti untuk menyimpulkan bahwa Bagian awal kolom Sulfate memiliki variansi yang sama dengan bagian akhirnya**

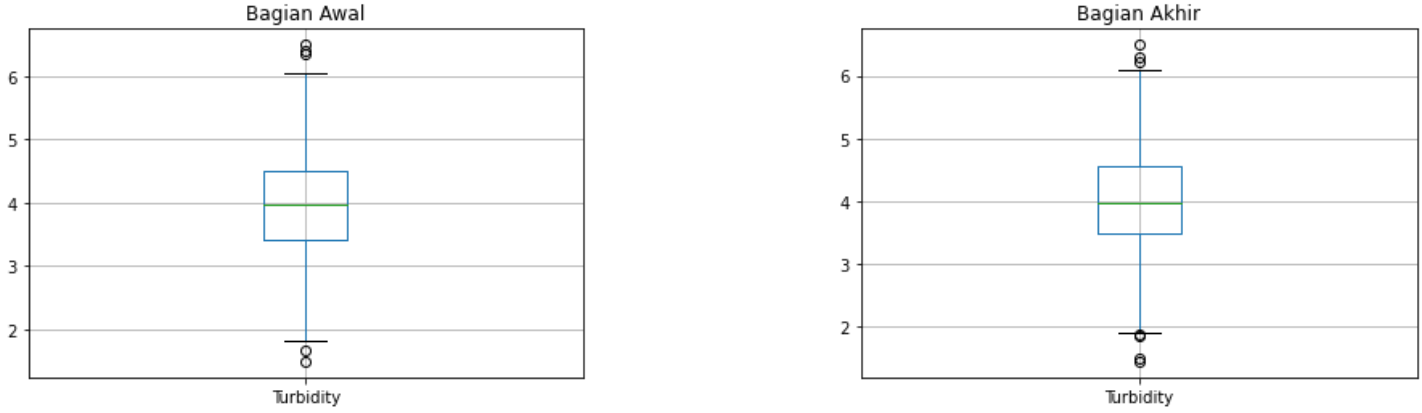
In [40]:

```
sampel = df['Sulfate']

zalpha = s.norm.ppf(0.95)
bagian1 = sampel.head(sampel.size // 2)
bagian2 = sampel.tail(sampel.size // 2)

std1 = bagian1.std()
std2 = bagian2.std()

if std1 > std2 :
    f = std1 / std2
else:
    f = std2 / std1

f1 = s.f.ppf(1 - 0.025, bagian1.size - 1, bagian2.size - 1)
f2 = s.f.ppf(0.025, bagian1.size - 1, bagian2.size - 1)
print("nilai f : " + str(f) )

pValue = s.f.cdf(f, bagian1.size - 1, bagian2.size - 1)
print("Nilai P-value :" , pValue)
if (pValue > 0.05) :
    print("H0 gagal ditolak")
else :
    print("H0 ditolak")

plt.subplots_adjust(left=2, right=4, wspace=0.5)
plt.subplot(121)
df.head(1005).boxplot("Turbidity").set_title("Bagian Awal")
plt.subplot(122)
df.tail(1005).boxplot("Turbidity").set_title("Bagian Akhir")
```

```
nilai f : 1.0075966972926254
Nilai P-value : 0.5477067289575377
H0 gagal ditolak
```

Out[40]:

```
Text(0.5, 1.0, 'Bagian Akhir')
```



6. Test Korelasi

Rule Korelasi:

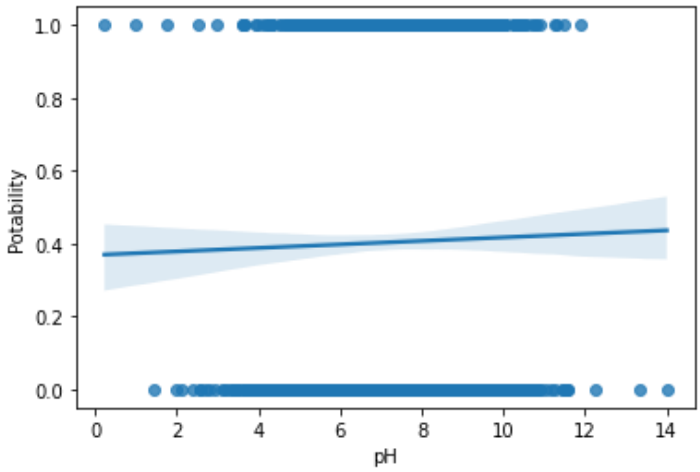
1. **Semakin mendekati 0, semakin dua kolom tidak berkorelasi**
2. **Semakin mendekati 1, semakin dua kolom berbanding lurus**
3. **Semakin mendekati -1, semakin dua kolom berbanding terbalik**

6.1 Korelasi Kolom pH dengan Kolom Target (Potability)

In [41]:

```
col_ph = df["pH"]
col_target = df["Potability"]
corr_ph_target = col_ph.corr(col_target)

sns.regplot(col_ph, col_target)
print("Kolerasi kolom pH dengan kolom Target adalah:", corr_ph_target)
```

```
Kolerasi kolom pH dengan kolom Target adalah: 0.01547509440843348
```



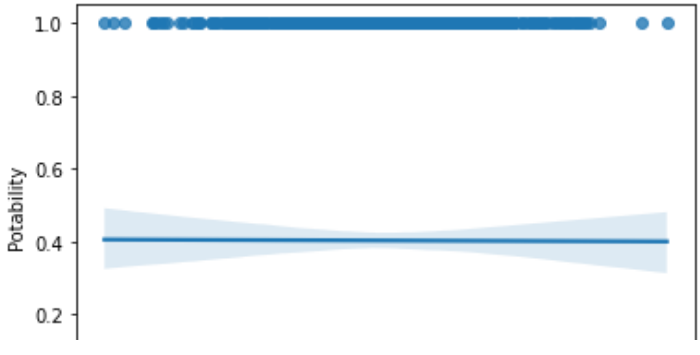
Kesimpulan: Dikarenakan nilai korelasi antara kolom pH dengan kolom Target mendekati 0, maka dapat dikatakan bahwa kedua kolom tersebut tidak berkolerasi

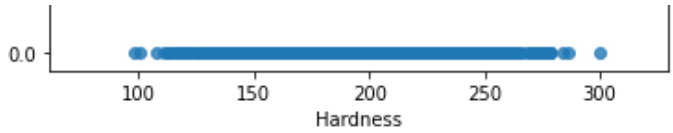
6.2 Korelasi Kolom Hardness dengan Kolom Target (Potability)

In [42]:

```
col_hardness = df["Hardness"]
col_target = df["Potability"]
corr_hardness_target = col_hardness.corr(col_target)

sns.regplot(col_hardness, col_target)
print("Kolerasi kolom Hardness dengan kolom Target adalah:", corr_hardness_target)
```

```
Kolerasi kolom Hardness dengan kolom Target adalah: -0.0014631528959479344
```





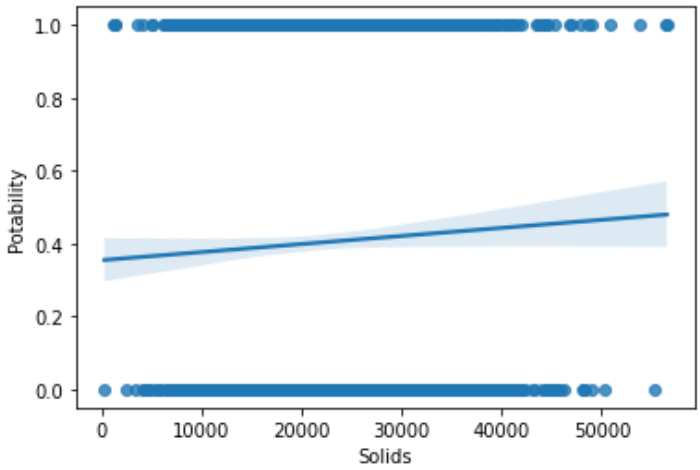
Kesimpulan: Dikarenakan nilai korelasi antara kolom Hardness dengan kolom Target mendekati 0, maka dapat dikatakan bahwa kedua kolom tersebut tidak berkolerasi

6.3 Korelasi Kolom Solids dengan Kolom Target (Potability)

In [43]:

```
col_solids = df["Solids"]
col_target = df["Potability"]
corr_solids_target = col_solids.corr(col_target)

sns.regplot(col_solids, col_target)
print("Kolerasi kolom Solids dengan kolom Target adalah:", corr_solids_target)
```

```
Kolerasi kolom Solids dengan kolom Target adalah: 0.03897657818173466
```



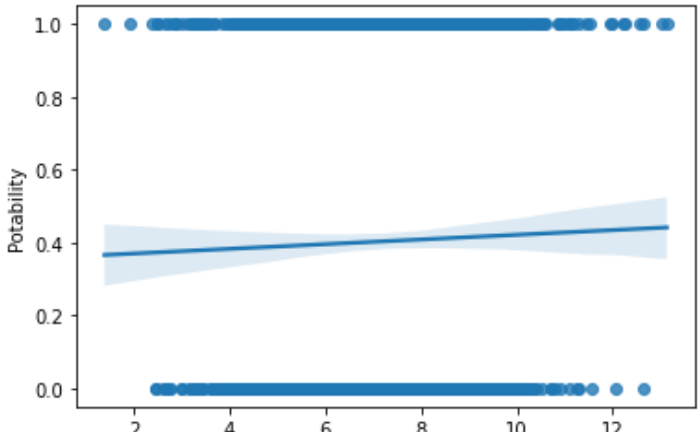
Kesimpulan: Dikarenakan nilai korelasi antara kolom Solids dengan kolom Target mendekati 0, maka dapat dikatakan bahwa kedua kolom tersebut tidak berkolerasi

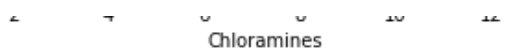
6.4 Korelasi Kolom Chloramines dengan Kolom Target (Potability)

In [44]:

```
col_chloramines = df["Chloramines"]
col_target = df["Potability"]
corr_chloramines_target = col_chloramines.corr(col_target)

sns.regplot(col_chloramines, col_target)
print("Kolerasi kolom Chloramines dengan kolom Target adalah:", corr_chloramines_target)
```

```
Kolerasi kolom Chloramines dengan kolom Target adalah: 0.02077892184052409
```





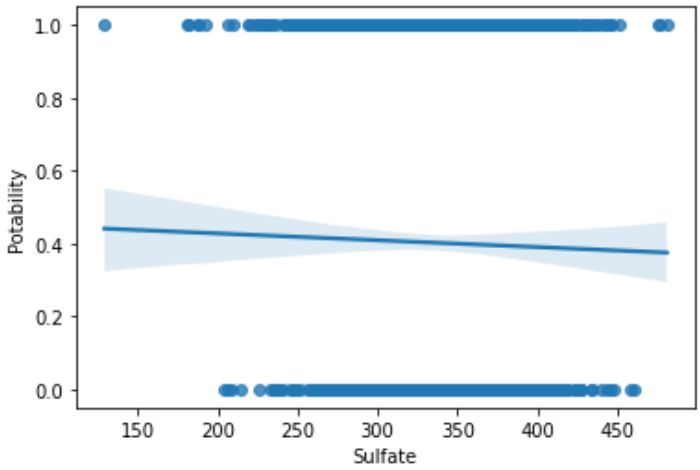
Kesimpulan: Dikarenakan nilai korelasi antara kolom Chloramines dengan kolom Target mendekati 0, maka dapat dikatakan bahwa kedua kolom tersebut tidak berkolerasi

6.5 Korelasi Kolom Sulfate dengan Kolom Target (Potability)

In [45]:

```
col_sulfate = df["Sulfate"]
col_target = df["Potability"]
corr_sulfate_target = col_sulfate.corr(col_target)

sns.regplot(col_sulfate, col_target)
print("Kolerasi kolom Sulfate dengan kolom Target adalah:", corr_sulfate_target)
```

```
Kolerasi kolom Sulfate dengan kolom Target adalah: -0.015703164419273778
```



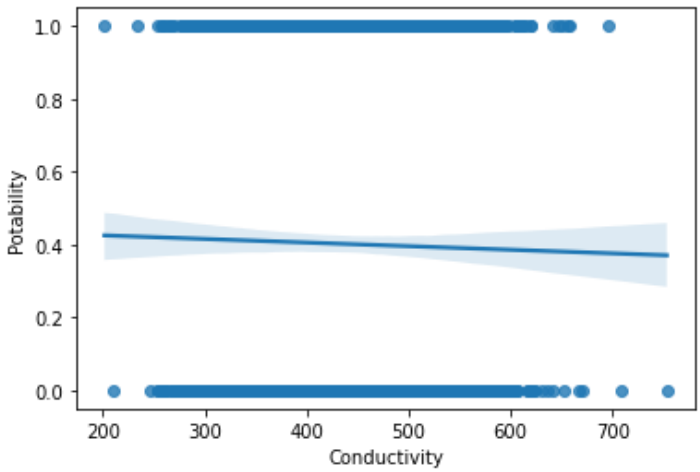
Kesimpulan: Dikarenakan nilai korelasi antara kolom Sulfate dengan kolom Target mendekati 0, maka dapat dikatakan bahwa kedua kolom tersebut tidak berkolerasi

6.6 Korelasi Kolom Conductivity dengan Kolom Target (Potability)

In [46]:

```
col_conductivity = df["Conductivity"]
col_target = df["Potability"]
corr_conductivity_target = col_conductivity.corr(col_target)

sns.regplot(col_conductivity, col_target)
print("Kolerasi kolom Conductivity dengan kolom Target adalah:", corr_conductivity_target
)
```

```
Kolerasi kolom Conductivity dengan kolom Target adalah: -0.016257120111377067
```



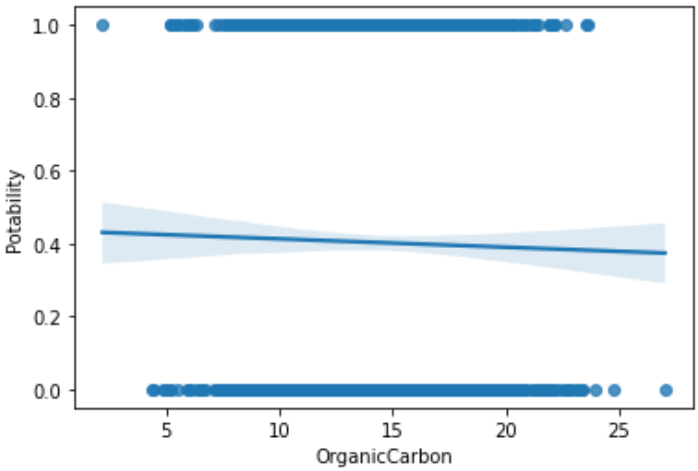
Kesimpulan: Dikarenakan nilai korelasi antara kolom Conductivity dengan kolom Target mendekati 0, maka dapat dikatakan bahwa kedua kolom tersebut tidak berkolerasi

6.7 Korelasi Kolom Organic Carbon dengan Kolom Target (Potability)

In [47]:

```
col_organic_carbon = df["OrganicCarbon"]
col_target = df["Potability"]
corr_organic_carbon_target = col_organic_carbon.corr(col_target)

sns.regplot(col_organic_carbon, col_target)
print("Kolerasi kolom Organic Carbon dengan kolom Target adalah:", corr_organic_carbon_ta
rget)
```

```
Kolerasi kolom Organic Carbon dengan kolom Target adalah: -0.015488461910747259
```



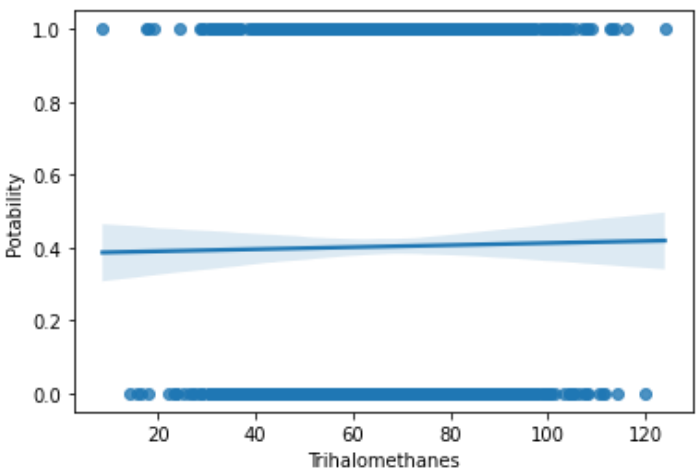
Kesimpulan: Dikarenakan nilai korelasi antara kolom Organic Carbon dengan kolom Target mendekati 0, maka dapat dikatakan bahwa kedua kolom tersebut tidak berkolerasi

6.8 Korelasi Kolom Trihalomethanes dengan Kolom Target (Potability)

In [48]:

```
col_trihalomethanes = df["Trihalomethanes"]
col_target = df["Potability"]
corr_trihalomethanes_target = col_trihalomethanes.corr(col_target)

sns.regplot(col_trihalomethanes, col_target)
print("Kolerasi kolom Trihalomethanes dengan kolom Target adalah:", corr_trihalomethanes_
target)
```

```
Kolerasi kolom Trihalomethanes dengan kolom Target adalah: 0.009236711064712997
```



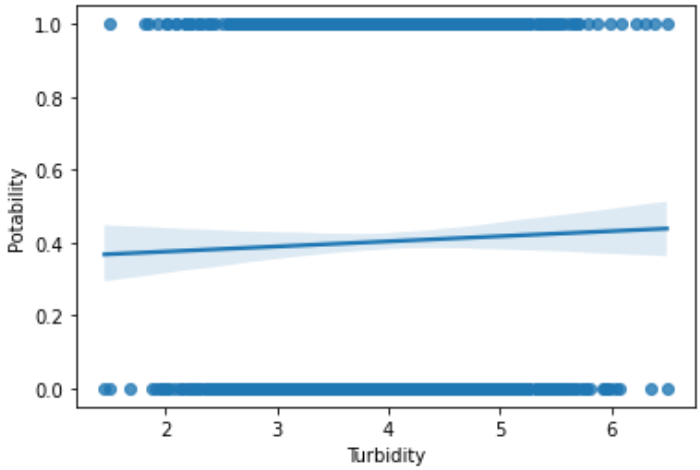
Kesimpulan: Dikarenakan nilai korelasi antara kolom Trihalomethanes dengan kolom Target mendekati 0, maka dapat dikatakan bahwa kedua kolom tersebut tidak berkolerasi

6.9 Korelasi Kolom Turbidity dengan Kolom Target (Potability)

In [49]:

```
col_turbidity = df["Turbidity"]
col_target = df["Potability"]
corr_turbidity_target = col_turbidity.corr(col_target)

sns.regplot(col_turbidity, col_target)
print("Kolerasi kolom Turbidity dengan kolom Target adalah:", corr_turbidity_target)
```

```
Kolerasi kolom Turbidity dengan kolom Target adalah: 0.022331042640622665
```



Kesimpulan: Dikarenakan nilai korelasi antara kolom Turbility dengan kolom Target mendekati 0, maka dapat dikatakan bahwa kedua kolom tersebut tidak berkolerasi