Name : Tanishka Vishnu Tapkir

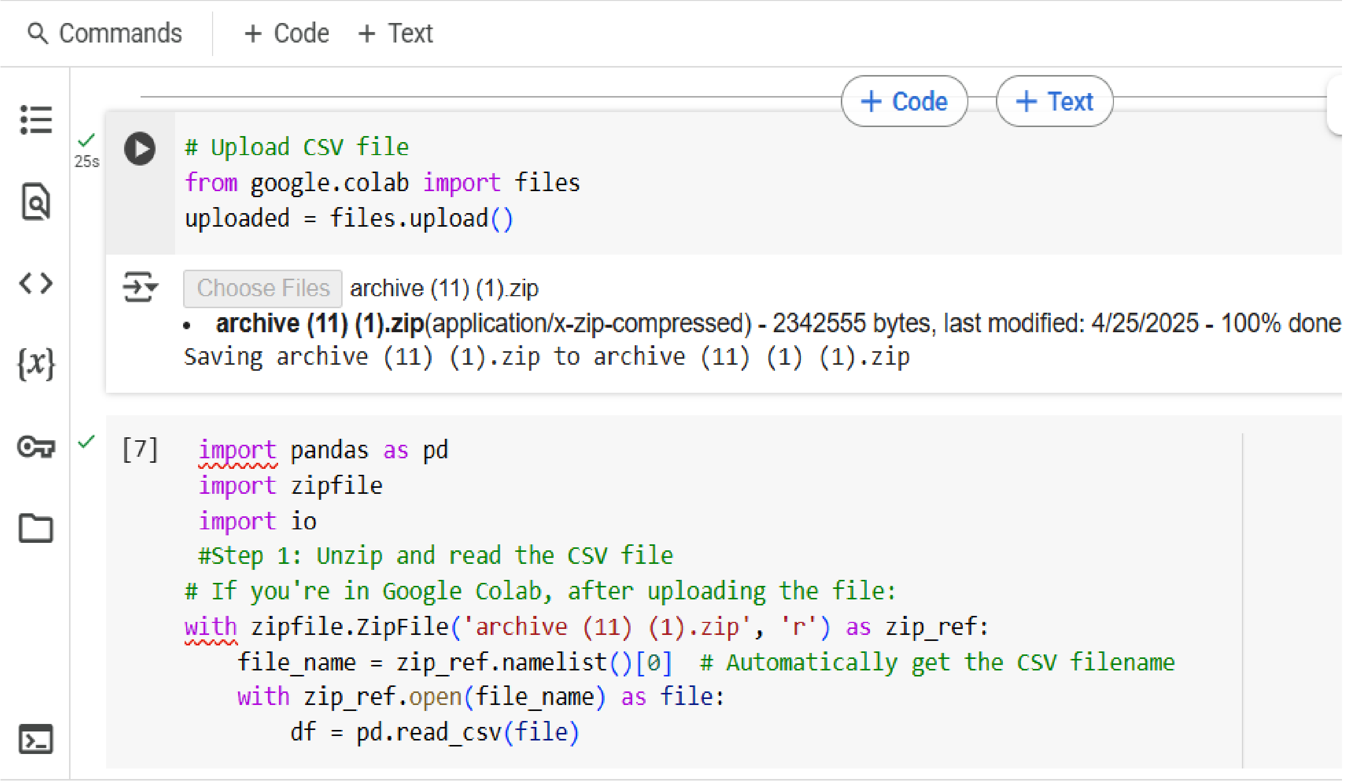
Div : CS7

Roll no : 86

PRN : 202401110071

Batch : CS74

URL Link: https://www.kaggle.com/datasets/muthuj7/weather-dataset



```
# Upload CSV file
from google.colab import files
uploaded = files.upload()
```

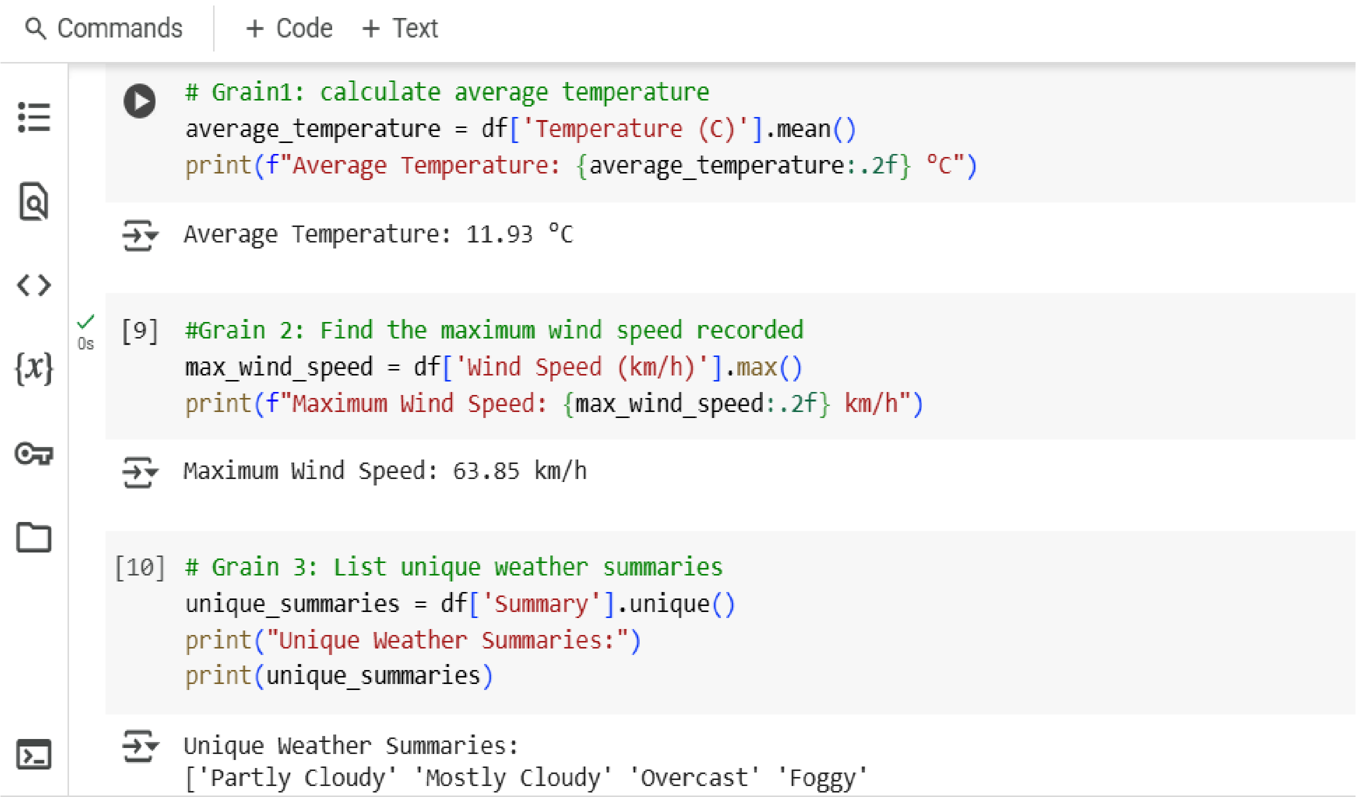
archive (11) (1).zip

- **archive (11) (1).zip**(application/x-zip-compressed) - 2342555 bytes, last modified: 4/25/2025 - 100% done

```
Saving archive (11) (1).zip to archive (11) (1) (1).zip
```

[7]

```
 import pandas as pd
 import zipfile
 import io
 #Step 1: Unzip and read the CSV file
# If you're in Google Colab, after uploading the file:
with zipfile.ZipFile('archive (11) (1).zip', 'r') as zip_ref:
    file_name = zip_ref.namelist()[0]  # Automatically get the CSV filename
    with zip_ref.open(file_name) as file:
        df = pd.read_csv(file)
```



```
# Grain1: calculate average temperature
average_temperature = df['Temperature (C)'].mean()
print(f"Average Temperature: {average_temperature:.2f} °C")
```

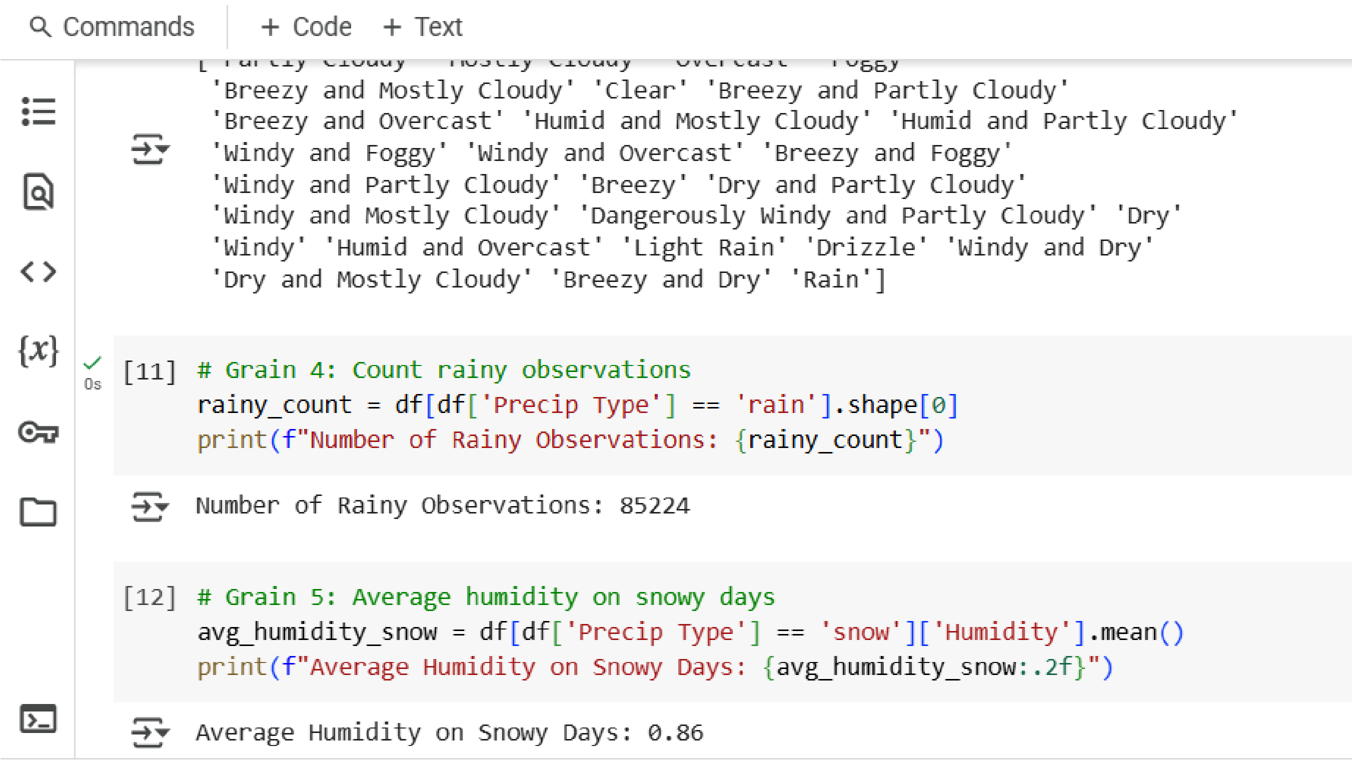
```
Average Temperature: 11.93 °C
```

```
#Grain 2: Find the maximum wind speed recorded
max_wind_speed = df['Wind Speed (km/h)'].max()
print(f"Maximum Wind Speed: {max_wind_speed:.2f} km/h")
```

```
Maximum Wind Speed: 63.85 km/h
```

```
# Grain 3: List unique weather summaries
unique_summaries = df['Summary'].unique()
print("Unique Weather Summaries:")
print(unique_summaries)
```

```
Unique Weather Summaries:
['Partly Cloudy' 'Mostly Cloudy' 'Overcast' 'Foggy'
```



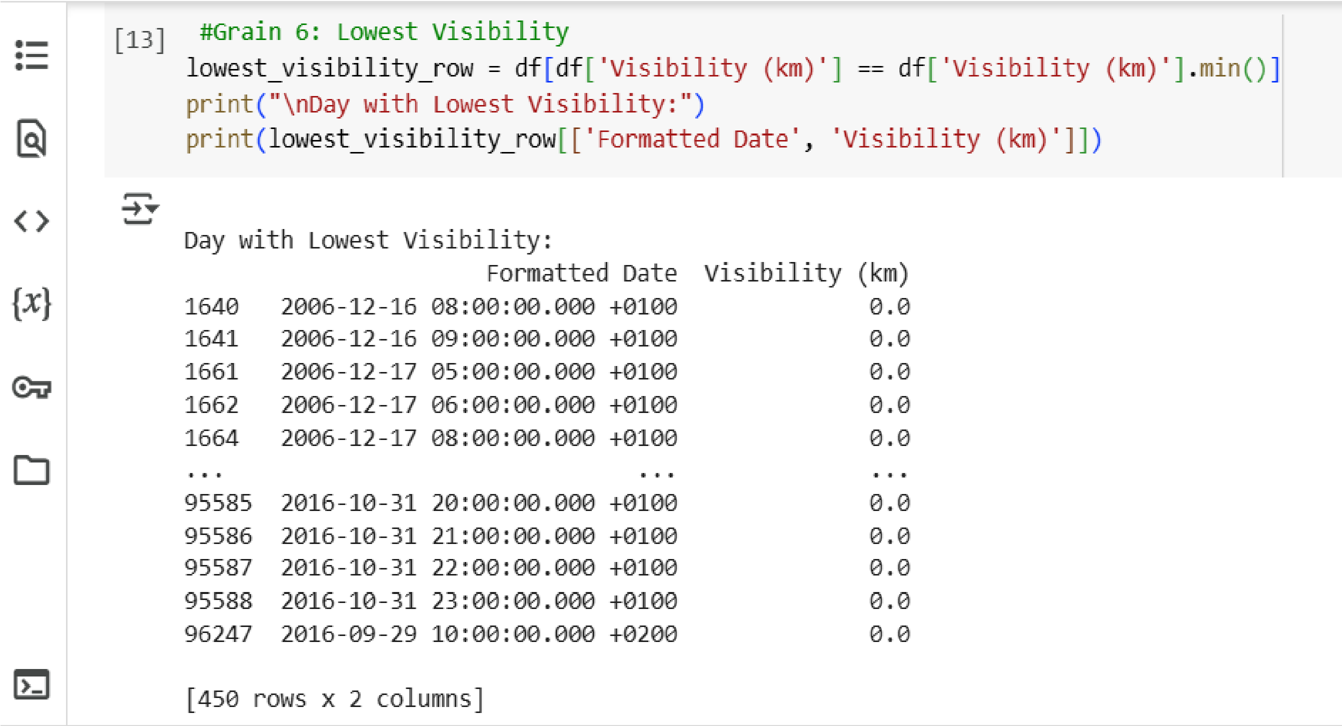
```
['Partly Cloudy' 'Mostly Cloudy' 'Overcast' 'Foggy'
 'Breezy and Mostly Cloudy' 'Clear' 'Breezy and Partly Cloudy'
 'Breezy and Overcast' 'Humid and Mostly Cloudy' 'Humid and Partly Cloudy'
 'Windy and Foggy' 'Windy and Overcast' 'Breezy and Foggy'
 'Windy and Partly Cloudy' 'Breezy' 'Dry and Partly Cloudy'
 'Windy and Mostly Cloudy' 'Dangerously Windy and Partly Cloudy' 'Dry'
 'Windy' 'Humid and Overcast' 'Light Rain' 'Drizzle' 'Windy and Dry'
 'Dry and Mostly Cloudy' 'Breezy and Dry' 'Rain']
```

```
# Grain 4: Count rainy observations
rainy_count = df[df['Precip Type'] == 'rain'].shape[0]
print(f"Number of Rainy Observations: {rainy_count}")
```

```
Number of Rainy Observations: 85224
```

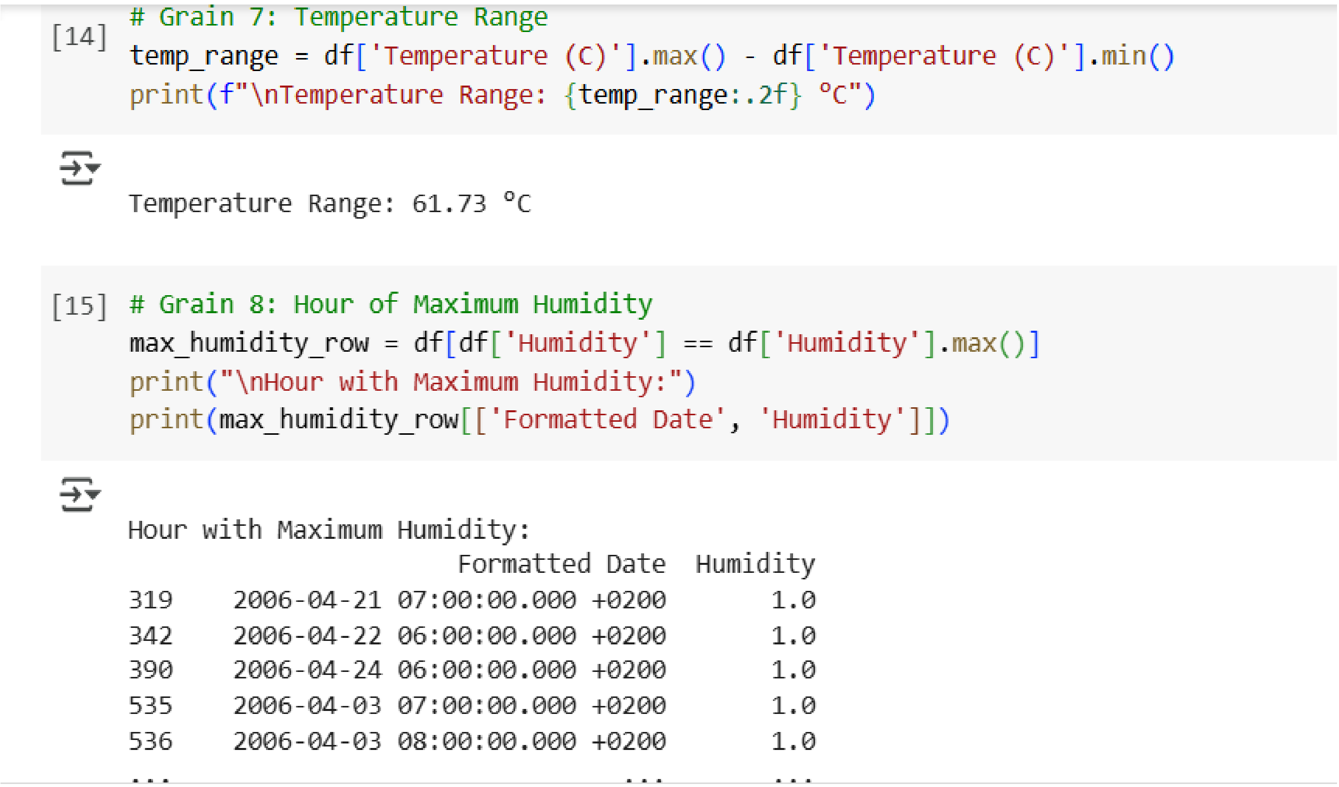
```
# Grain 5: Average humidity on snowy days
avg_humidity_snow = df[df['Precip Type'] == 'snow']['Humidity'].mean()
print(f"Average Humidity on Snowy Days: {avg_humidity_snow:.2f}")
```

```
Average Humidity on Snowy Days: 0.86
```



```
[13]  #Grain 6: Lowest Visibility
     lowest_visibility_row = df[df['Visibility (km)'] == df['Visibility (km)'].min()]
     print("\nDay with Lowest Visibility:")
     print(lowest_visibility_row[['Formatted Date', 'Visibility (km)']])
```

```
Day with Lowest Visibility:
                      Formatted Date  Visibility (km)
1640   2006-12-16 08:00:00.000 +0100              0.0
1641   2006-12-16 09:00:00.000 +0100              0.0
1661   2006-12-17 05:00:00.000 +0100              0.0
1662   2006-12-17 06:00:00.000 +0100              0.0
1664   2006-12-17 08:00:00.000 +0100              0.0
...                              ...              ...
95585  2016-10-31 20:00:00.000 +0100              0.0
95586  2016-10-31 21:00:00.000 +0100              0.0
95587  2016-10-31 22:00:00.000 +0100              0.0
95588  2016-10-31 23:00:00.000 +0100              0.0
96247  2016-09-29 10:00:00.000 +0200              0.0

[450 rows x 2 columns]
```

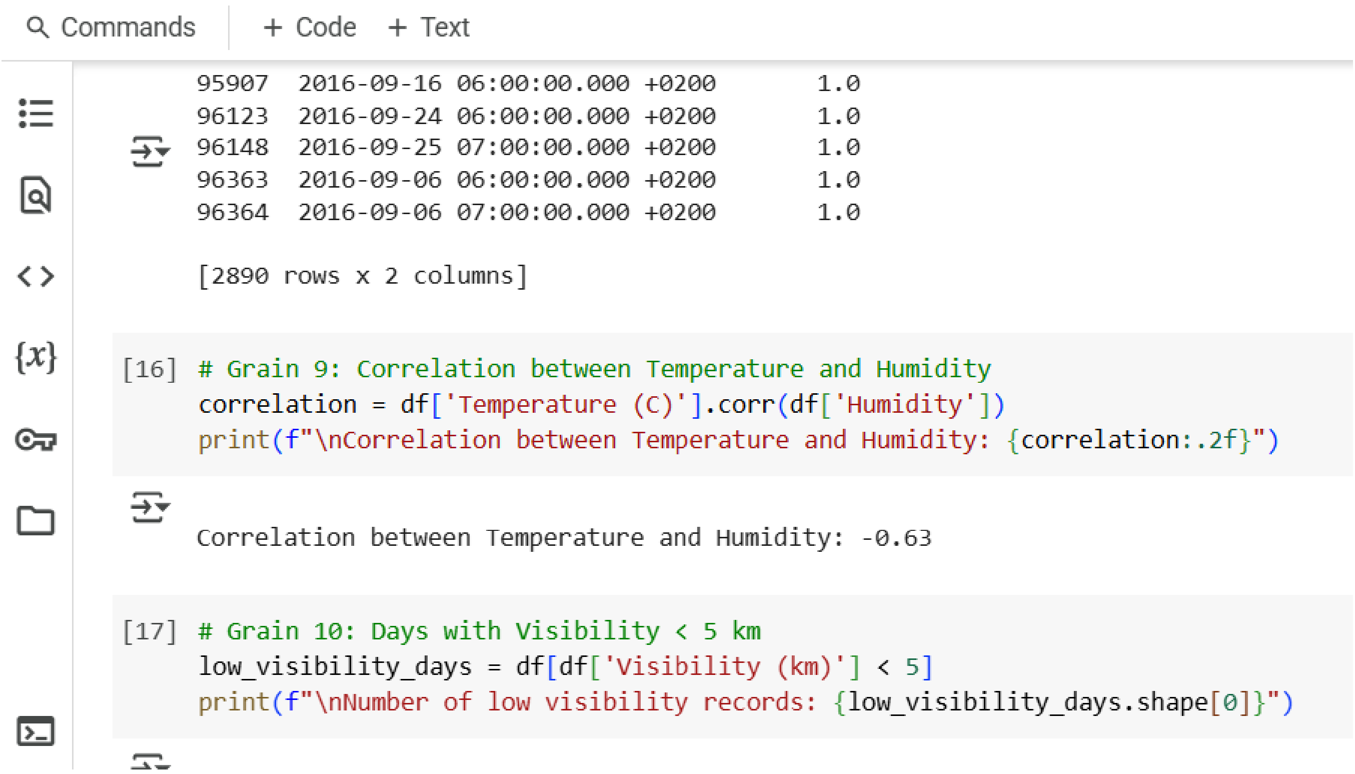


```
[14]  # Grain 7: Temperature Range
      temp_range = df['Temperature (C)'].max() - df['Temperature (C)'].min()
      print(f"\nTemperature Range: {temp_range:.2f} °C")
```

```
Temperature Range: 61.73 °C
```

```
[15]  # Grain 8: Hour of Maximum Humidity
      max_humidity_row = df[df['Humidity'] == df['Humidity'].max()]
      print("\nHour with Maximum Humidity:")
      print(max_humidity_row[['Formatted Date', 'Humidity']])
```

```
Hour with Maximum Humidity:
                      Formatted Date  Humidity
319    2006-04-21 07:00:00.000 +0200       1.0
342    2006-04-22 06:00:00.000 +0200       1.0
390    2006-04-24 06:00:00.000 +0200       1.0
535    2006-04-03 07:00:00.000 +0200       1.0
536    2006-04-03 08:00:00.000 +0200       1.0
...                              ...       ...
```

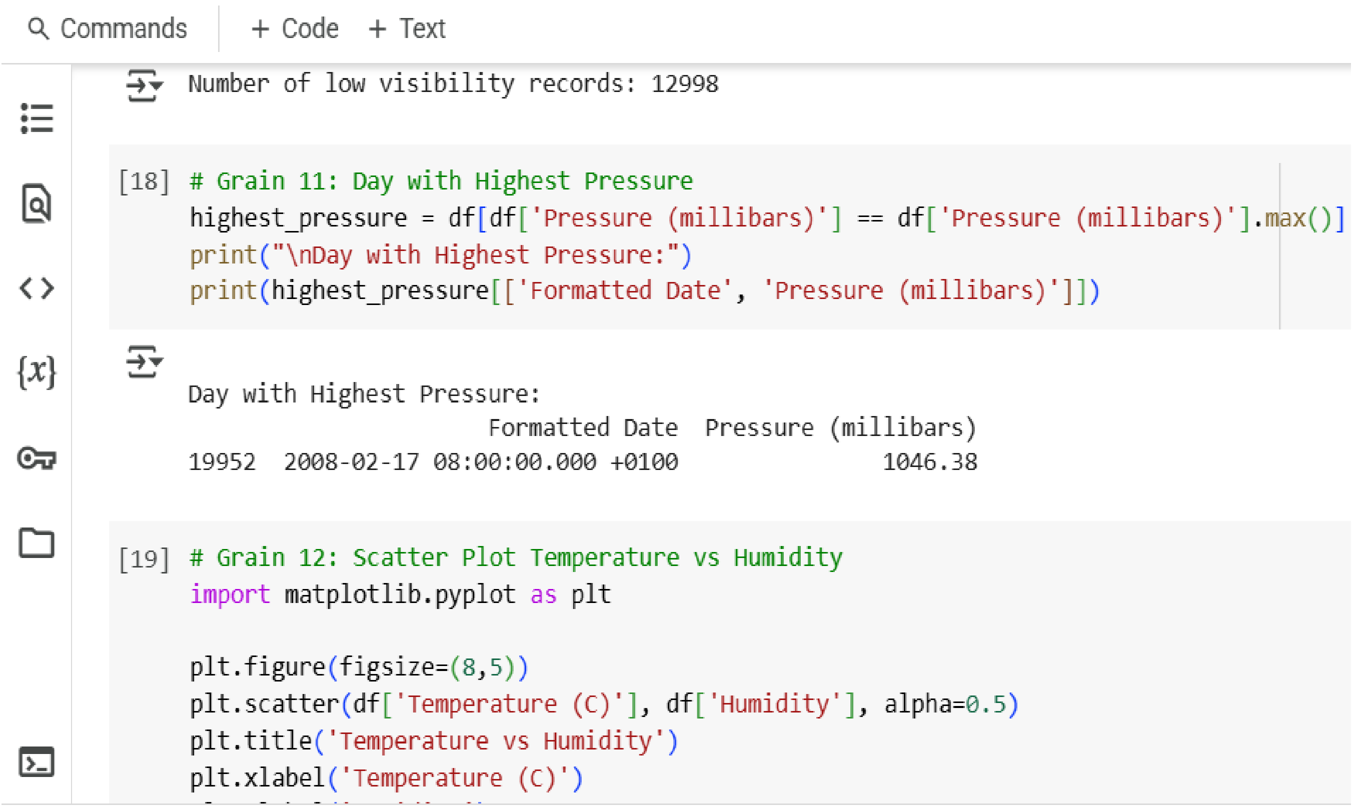


```
95907  2016-09-16 06:00:00.000 +0200       1.0
96123  2016-09-24 06:00:00.000 +0200       1.0
96148  2016-09-25 07:00:00.000 +0200       1.0
96363  2016-09-06 06:00:00.000 +0200       1.0
96364  2016-09-06 07:00:00.000 +0200       1.0

[2890 rows x 2 columns]
```

```
[16] # Grain 9: Correlation between Temperature and Humidity
     correlation = df['Temperature (C)'].corr(df['Humidity'])
     print(f"\nCorrelation between Temperature and Humidity: {correlation:.2f}")
```

```

Correlation between Temperature and Humidity: -0.63
```

```
[17] # Grain 10: Days with Visibility < 5 km
     low_visibility_days = df[df['Visibility (km)'] < 5]
     print(f"\nNumber of low visibility records: {low_visibility_days.shape[0]}")
```

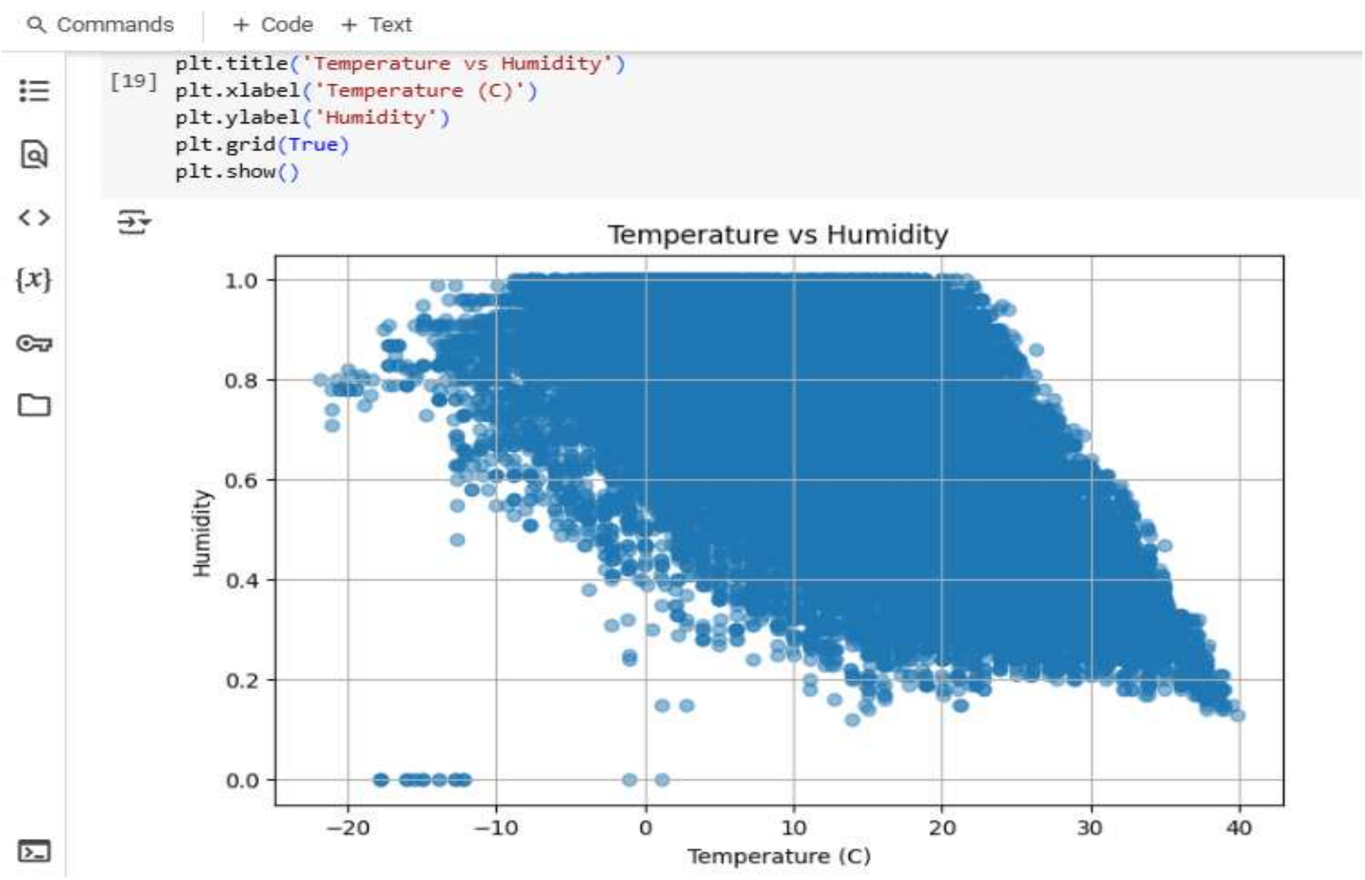


```
Number of low visibility records: 12998
```

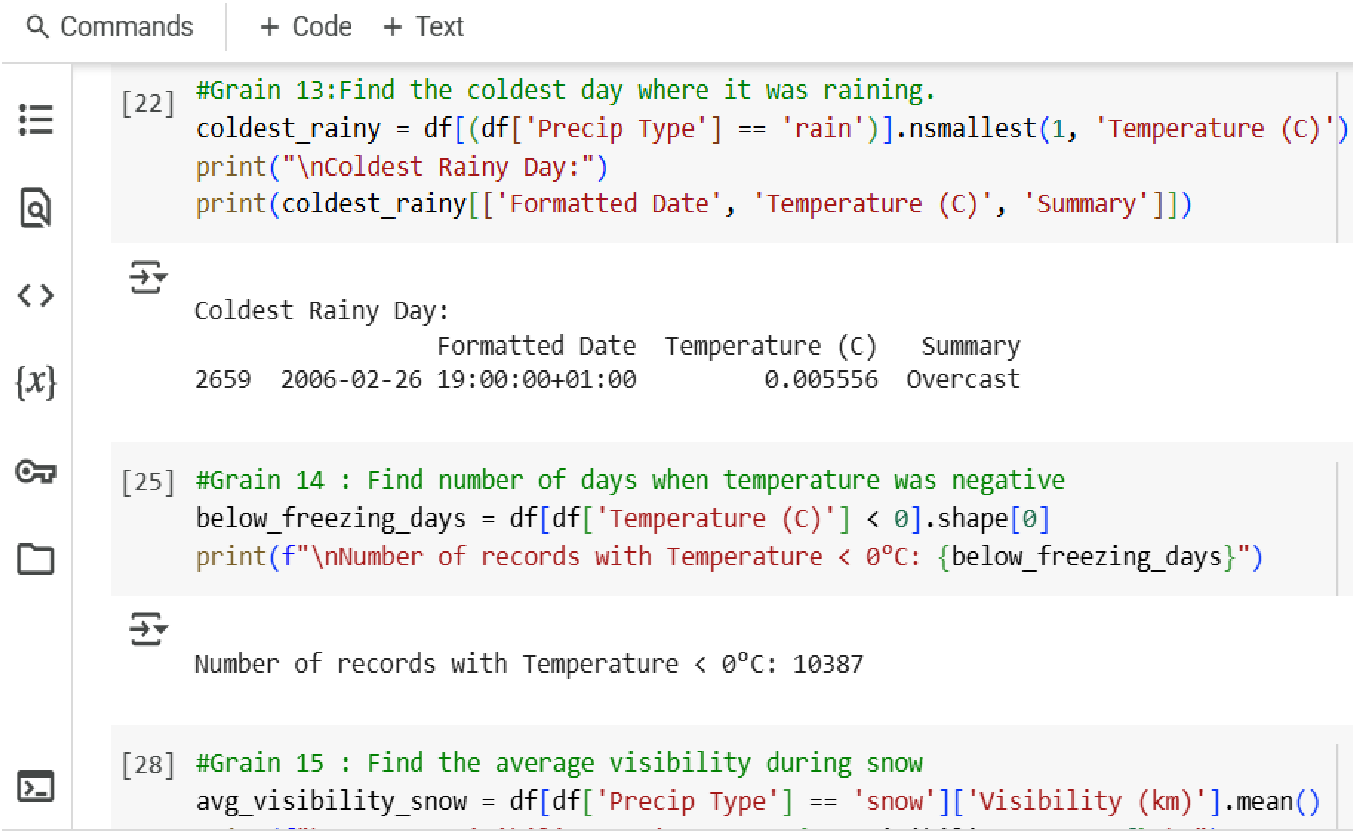
```
[18] # Grain 11: Day with Highest Pressure
     highest_pressure = df[df['Pressure (millibars)'] == df['Pressure (millibars)'].max()]
     print("\nDay with Highest Pressure:")
     print(highest_pressure[['Formatted Date', 'Pressure (millibars)']])
```

```

Day with Highest Pressure:
                      Formatted Date  Pressure (millibars)
19952  2008-02-17 08:00:00.000 +0100               1046.38
```

```
[19] # Grain 12: Scatter Plot Temperature vs Humidity
     import matplotlib.pyplot as plt

     plt.figure(figsize=(8,5))
     plt.scatter(df['Temperature (C)'], df['Humidity'], alpha=0.5)
     plt.title('Temperature vs Humidity')
     plt.xlabel('Temperature (C)')
```



```
[19] plt.title('Temperature vs Humidity')
     plt.xlabel('Temperature (C)')
     plt.ylabel('Humidity')
     plt.grid(True)
     plt.show()
```



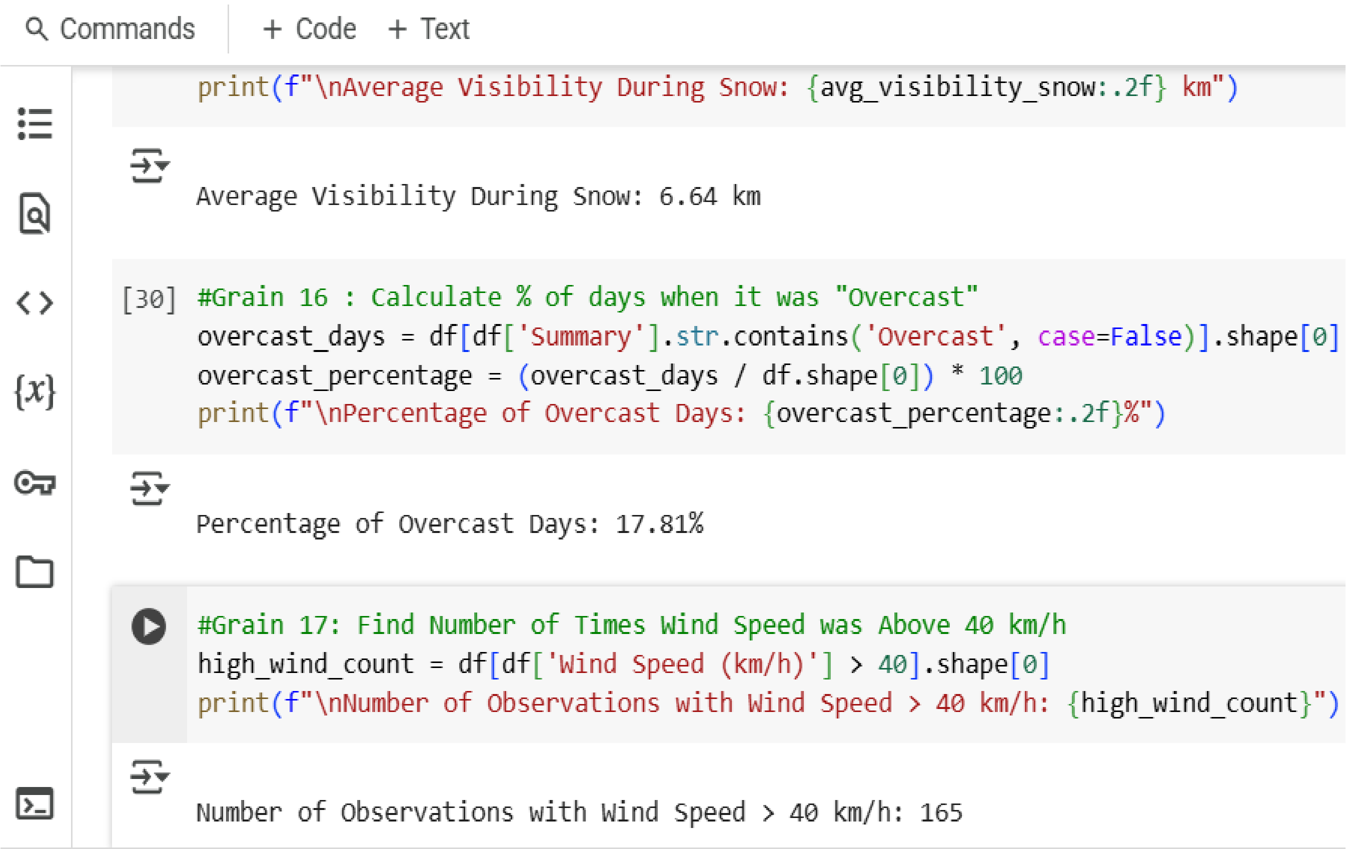
```
[22] #Grain 13:Find the coldest day where it was raining.
     coldest_rainy = df[(df['Precip Type'] == 'rain')].nsmallest(1, 'Temperature (C)')
     print("\nColdest Rainy Day:")
     print(coldest_rainy[['Formatted Date', 'Temperature (C)', 'Summary']])
```

```
Coldest Rainy Day:
                     Formatted Date  Temperature (C)   Summary
2659  2006-02-26 19:00:00+01:00         0.005556  Overcast
```

```
[25] #Grain 14 : Find number of days when temperature was negative
     below_freezing_days = df[df['Temperature (C)'] < 0].shape[0]
     print(f"\nNumber of records with Temperature < 0°C: {below_freezing_days}")
```

```
Number of records with Temperature < 0°C: 10387
```

```
[28] #Grain 15 : Find the average visibility during snow
     avg_visibility_snow = df[df['Precip Type'] == 'snow']['Visibility (km)'].mean()
```



```
print(f"\nAverage Visibility During Snow: {avg_visibility_snow:.2f} km")
```

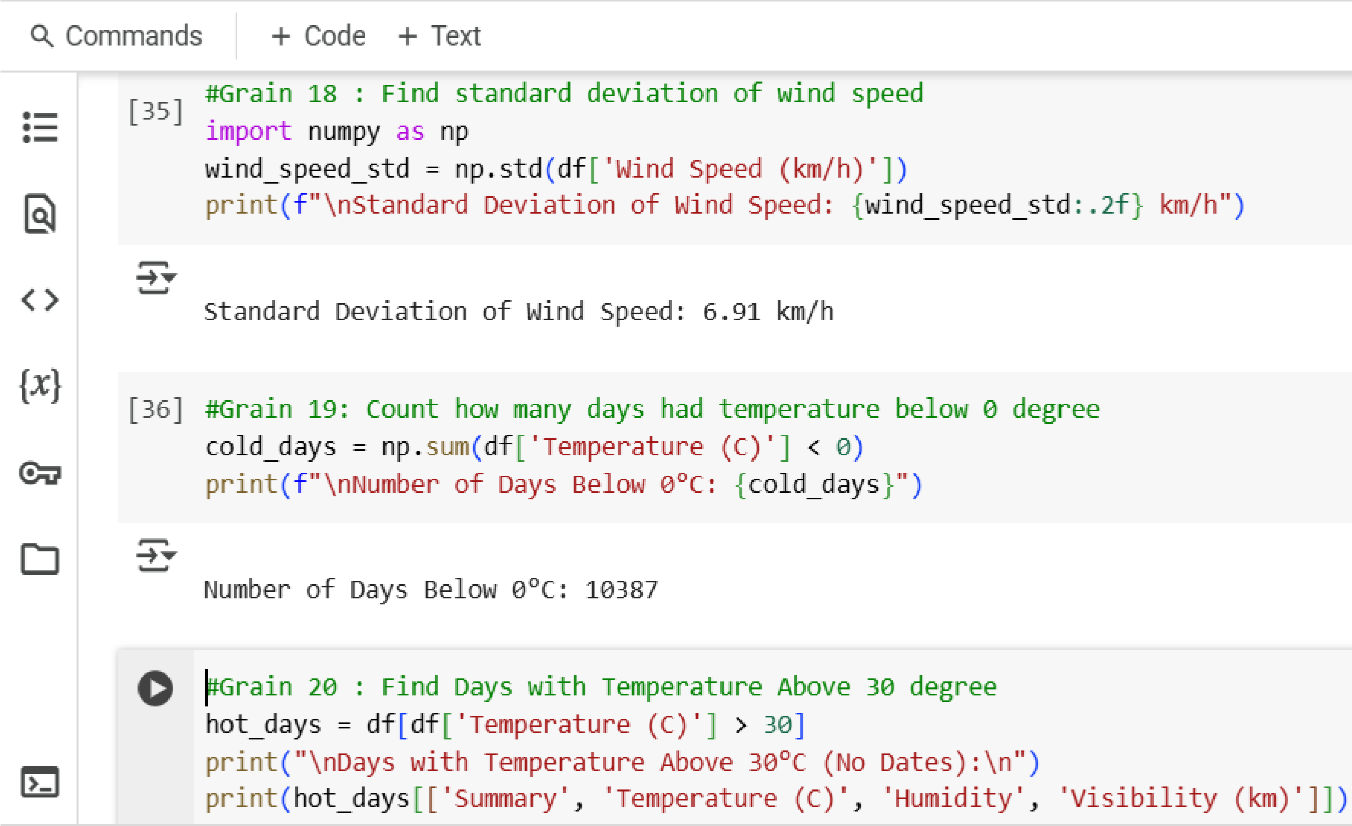
```
Average Visibility During Snow: 6.64 km
```

```
[30] #Grain 16 : Calculate % of days when it was "Overcast"
     overcast_days = df[df['Summary'].str.contains('Overcast', case=False)].shape[0]
     overcast_percentage = (overcast_days / df.shape[0]) * 100
     print(f"\nPercentage of Overcast Days: {overcast_percentage:.2f}%")
```

```
Percentage of Overcast Days: 17.81%
```

```
#Grain 17: Find Number of Times Wind Speed was Above 40 km/h
high_wind_count = df[df['Wind Speed (km/h)'] > 40].shape[0]
print(f"\nNumber of Observations with Wind Speed > 40 km/h: {high_wind_count}")
```

```
Number of Observations with Wind Speed > 40 km/h: 165
```



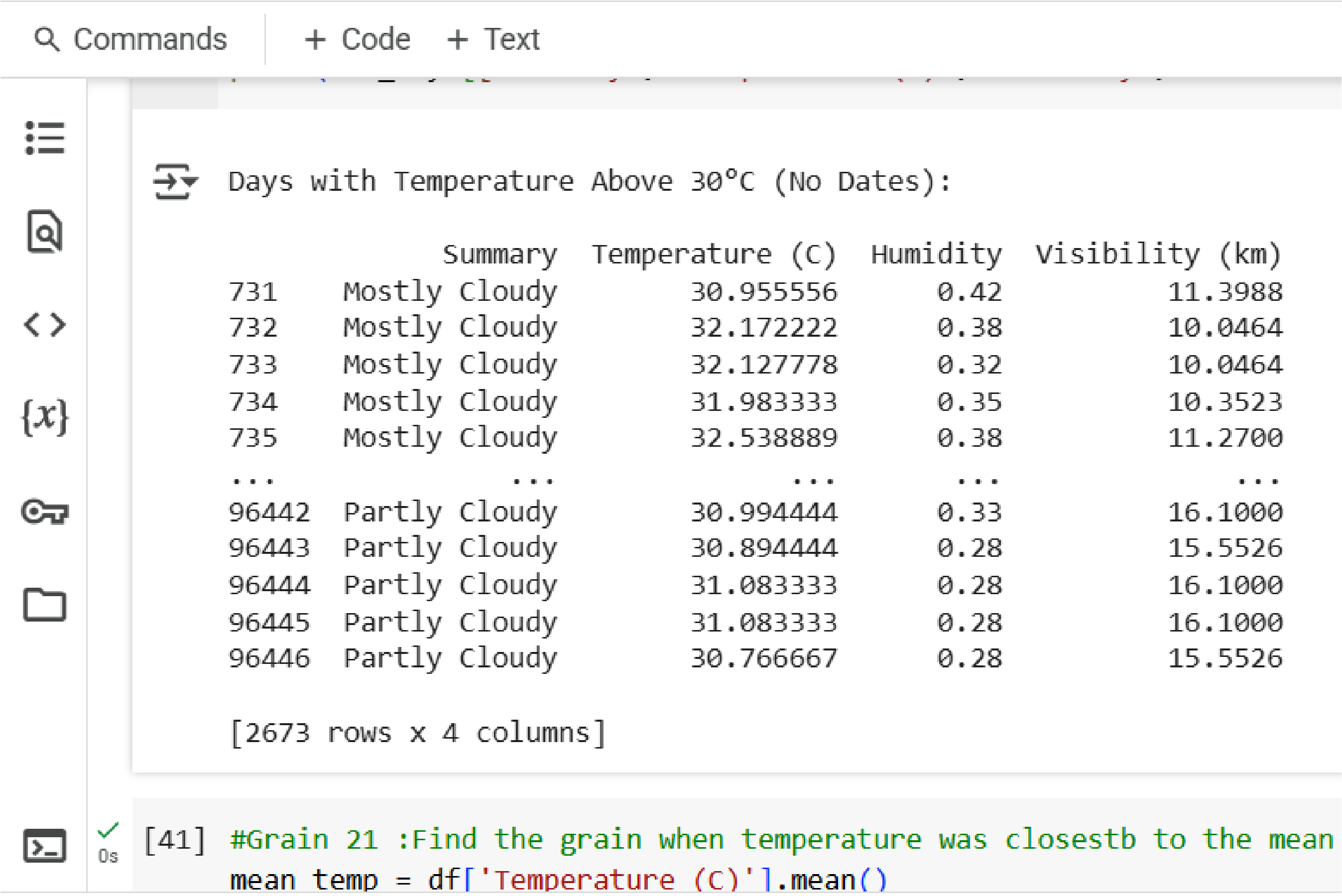
```
[35] #Grain 18 : Find standard deviation of wind speed
     import numpy as np
     wind_speed_std = np.std(df['Wind Speed (km/h)'])
     print(f"\nStandard Deviation of Wind Speed: {wind_speed_std:.2f} km/h")
```

```
Standard Deviation of Wind Speed: 6.91 km/h
```

```
[36] #Grain 19: Count how many days had temperature below 0 degree
     cold_days = np.sum(df['Temperature (C)'] < 0)
     print(f"\nNumber of Days Below 0°C: {cold_days}")
```

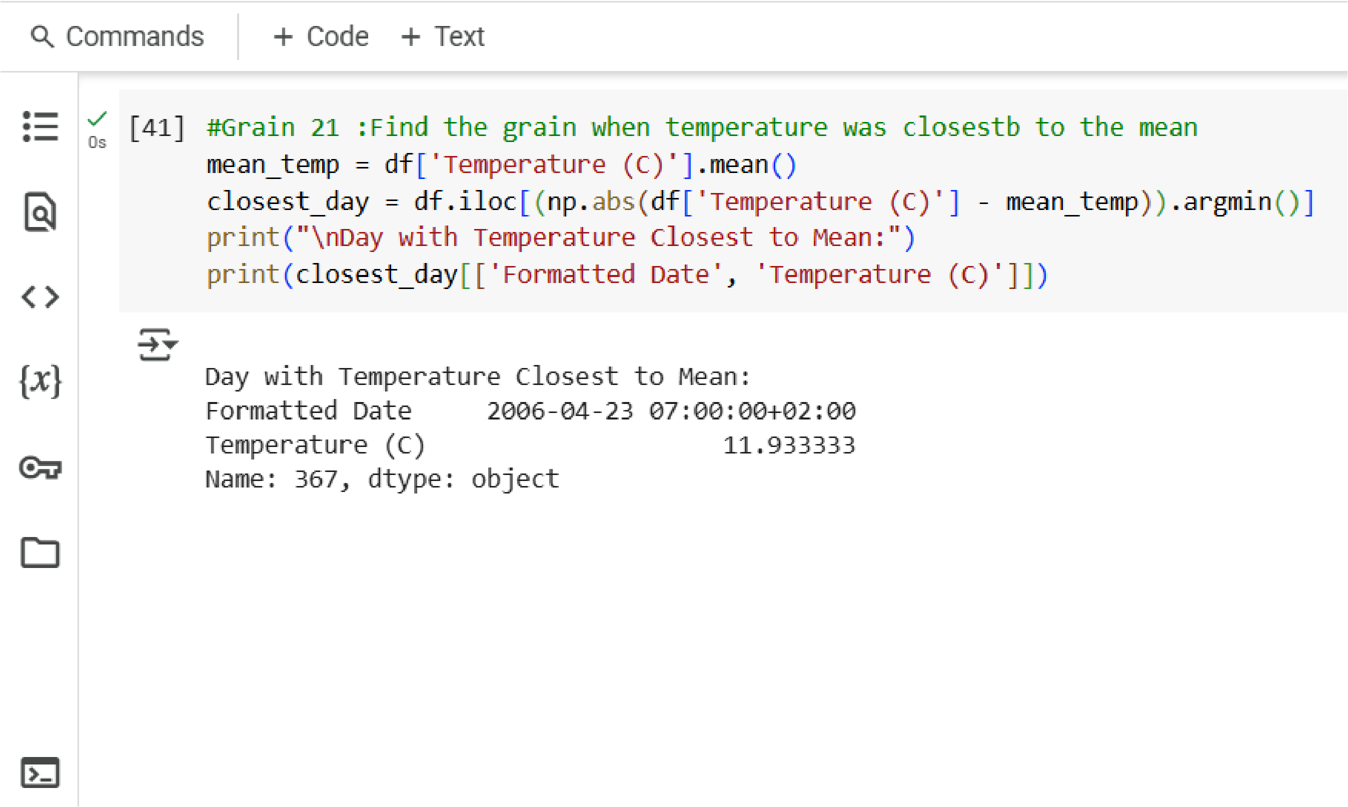
```
Number of Days Below 0°C: 10387
```

```
#Grain 20 : Find Days with Temperature Above 30 degree
hot_days = df[df['Temperature (C)'] > 30]
print("\nDays with Temperature Above 30°C (No Dates):\n")
print(hot_days[['Summary', 'Temperature (C)', 'Humidity', 'Visibility (km)']])
```



```
Days with Temperature Above 30°C (No Dates):

             Summary  Temperature (C)  Humidity  Visibility (km)
731    Mostly Cloudy        30.955556      0.42          11.3988
732    Mostly Cloudy        32.172222      0.38          10.0464
733    Mostly Cloudy        32.127778      0.32          10.0464
734    Mostly Cloudy        31.983333      0.35          10.3523
735    Mostly Cloudy        32.538889      0.38          11.2700
...              ...              ...       ...              ...
96442  Partly Cloudy        30.994444      0.33          16.1000
96443  Partly Cloudy        30.894444      0.28          15.5526
96444  Partly Cloudy        31.083333      0.28          16.1000
96445  Partly Cloudy        31.083333      0.28          16.1000
96446  Partly Cloudy        30.766667      0.28          15.5526

[2673 rows x 4 columns]
```

```
[41] #Grain 21 :Find the grain when temperature was closestb to the mean
     mean_temp = df['Temperature (C)'].mean()
```



```
[41] #Grain 21 :Find the grain when temperature was closestb to the mean
     mean_temp = df['Temperature (C)'].mean()
     closest_day = df.iloc[(np.abs(df['Temperature (C)'] - mean_temp)).argmin()]
     print("\nDay with Temperature Closest to Mean:")
     print(closest_day[['Formatted Date', 'Temperature (C)']])
```

```

Day with Temperature Closest to Mean:
Formatted Date     2006-04-23 07:00:00+02:00
Temperature (C)                    11.933333
Name: 367, dtype: object
```