Data Cleaning

High-Level Data Pipeline Description

This data pipeline appears to be designed for cleaning and processing climate data, likely focused on a specific analysis requiring a single data point per country.

Technology Used

This code snippet utilizes the Pandas library for data manipulation in Python. Pandas DataFrames and their functionalities are used for cleaning, transforming, and analyzing the data.

Transformation and Cleaning Steps

The code performs several cleaning and transformation steps:

- **Handling Missing Values:**
- **Resetting Index**
- **Removing Duplicates**
- **Grouping and Selecting Maximum Temperature:**
- **Renaming Columns**

Load the Dataset

```
climate_data = pd.read_csv('Project/Breakdown_Region.csv')

temperature_data =
pd.read_csv('Project/GlobalLandTemperaturesByCountry.csv')

# Display the first few rows of each dataset
climate_data.head(), temperature_data.head()
```

Merge the Dataset

Data Cleaning

```
# Ensure there are no NaN values in 'Country' and 'AverageTemperature'
columns
merged_data = merged_data.dropna(subset=['Country',
'AverageTemperature','Climate change: (1/1/04 - 9/27/21)', 'Global
Warming: (1/1/04 - 9/27/21)','AverageTemperatureUncertainty'])

# Reset index to avoid alignment issues
merged_data = merged_data.reset_index(drop=True)

# Remove duplicates from the dataset
merged_data = merged_data.drop_duplicates()
```

```python
# Group by 'Country' and keep the row with the highest
'AverageTemperature'
result_data = merged_data.loc[merged_data.groupby('Country')
['AverageTemperature'].idxmax()]

# Save the result to a new CSV file (uncomment the line below if you
need to save the file)
# result_data.to_csv('filtered_data.csv', index=False)

# Display the result
print(result_data)

# Reset index to avoid alignment issues
merged_data = merged_data.reset_index(drop=True)
# Display the result
print(result_data)

# Convert the 'Date' column to datetime format
merged_data['dt'] = pd.to_datetime(merged_data['dt'])

# Extract the month from the 'Date' column
merged_data['Month'] = merged_data['dt'].dt.month

# Reset index to avoid alignment issues
merged_data = merged_data.reset_index(drop=True)
merged_data

result_data

# Convert the Index object to a list
column_names_list = list(result_data.columns)

# Print the list of column names
print("Column names as list:", column_names_list)


#Change the Columns Name
# Rename specific columns (example: 'OldName1' to 'NewName1' and
'OldName2' to 'NewName2')
columns_to_rename = {
    'Climate change: (1/1/04 - 9/27/21)': 'Climate change',
    'Global Warming: (1/1/04 - 9/27/21)': 'Global Warming',

    # Add other column renaming as needed
}
merged_data = result_data.rename(columns=columns_to_rename)
merged_data
```

```
            Country Climate change Global Warming          dt  \
147321  Afghanistan            68%            32%  1997-07-01
66911       Albania            75%            25%  1757-07-01
```

```
177845      Algeria             86%             14%  2003-07-01
332397    Argentina             72%             28%  2012-01-01
259738      Armenia             75%             25%  2006-08-01
...             ...             ...             ...         ...
203091   Uzbekistan             63%             37%  1984-07-01
322375    Venezuela             75%             25%  2010-03-01
251442      Vietnam             63%             37%  1912-06-01
32062        Zambia             74%             26%  2005-10-01
18429      Zimbabwe             72%             28%  1995-10-01

        AverageTemperature  AverageTemperatureUncertainty
147321              28.533                          0.410
66911               25.843                          5.336
177845              35.829                          0.400
332397              23.290                          0.333
259738              25.291                          0.254
...                    ...                            ...
203091              30.375                          0.305
322375              27.807                          0.418
251442              28.463                          0.358
32062               26.282                          0.325
18429               26.601                          0.201

[147 rows x 6 columns]
```

```
# Convert the Index object to a list
column_names_list = list(merged_data.columns)

# Print the list of column names
print("Column names as list:", column_names_list)
```

```
Column names as list: ['Country', 'Climate change', 'Global Warming',
'dt', 'AverageTemperature', 'AverageTemperatureUncertainty']
```