Visualization Plan Document

Introduction

1. Purpose of the visualizations:

The purpose of visualizations is to analyze and depict temperature trends in Europe over recent decades, highlighting significant changes and differences between various regions.

2. Overview of the data being visualized:

The data consists of temperature records (minimum, mean, maximum) from the European Climate Assessment & Dataset (ECA&D). The records span from 1980 to the present and include temperature measurements from 23759 various stations across Europe.

3. Plot Types and Descriptions

Line Plot:

- **Purpose:** To show temperature trends over time for selected areas.
- **X-Axis:** Time
- **Y-Axis:** Temperature (°C)
- **Details:** Each line represents a different area, showing how temperatures have changed over time.
- **Implementation Proposal:** User can manually select the region of interest clicking on the map of Europe. The program will read which station is located the closest basing on longitude and latitude of the place clicked. Then will appear 3 line plots of temperature trends (max,mean,min) based on sensor readings of a station.

Bar Chart:

- **Purpose:** To compare temperature changes between different areas.
- **X-Axis:** Selected Areas
- **Y-Axis:** Temperature Change (°C)
- **Details:** Bars represent the magnitude of mean temperature change for each area, allowing easy comparison between regions.
- **Implementation Proposal:** The user will be able to manually select 5 regions on the map. Upon clicking, the program will locate the nearest station, and its collected results will be displayed on a chart. Clicking on the station again will cancel its selection. This chart will allow for a more detailed analysis of temperature differences across various regions.

Heatmap:

- **Purpose:** To visualize temperature changes across a geographic grid.
- **X-Axis:** Longitude
- **Y-Axis:** Latitude
- **Details:** Colors represent the degree of temperature change, providing a spatial understanding of temperature variations.
- **Implementation Proposal:** Every station will create a radius of ,,x" kilometers. Then a difference between first and last mean temperature value from sensor will be calculated and depending on a result, the proper color will cover the circle.

4. Axis Configurations

Line Plot:

- **X-Axis:**
  - **Data:** Time (Years)
  - **Range:** Variable based on data from sensor
  - **Scale:** Linear
  - **Label:** "Year"
- **Y-Axis:**
  - **Data:** Temperature (°C)
  - **Range:** Variable based on data from sensor
  - **Scale:** Linear
  - **Label:** "Temperature (°C)

Bar Chart:

- **X-Axis:**
  - **Data:** Areas
  - **Type:** Categorical
  - **Label:** "Regions"
- **Y-Axis:**
  - **Data:** Temperature Change (°C)
  - **Range:** Variable based on data
  - **Scale:** Linear
  - **Label:** "Temperature Change (°C)"

Heatmap:

- **X-Axis:**
  - **Data:** Longitude
  - **Range:** Based on dataset
  - **Scale:** Linear
  - **Label:** "Longitude"
- **Y-Axis:**

- **Data:** Latitude
- **Range:** Based on dataset
- **Scale:** Linear
- **Label:** "Latitude”

5. **Data Queries**

Line Plot Data Query:

```python
# PySpark example
from pyspark.sql import SparkSession
from pyspark.sql.functions import col, avg

spark = SparkSession.builder.appName("TemperatureTrends").getOrCreate()

# Load data
data = spark.read.csv("path/to/cleaned_data", header=True, inferSchema=True)

# Filter data for the required regions and time period
regions = ["Region1", "Region2", "Region3", "Region4", "Region5"]
filtered_data = data.filter((col("region").isin(regions)) & (col("year") >= 1980))

# Calculate average temperature for each year
avg_temp_data = filtered_data.groupBy("year", "region").agg(avg("temperature").alias("avg_temp"))
```

Bar Chart Data Query:

```python
# PySpark example
# Calculate temperature change for each region
temp_change_data = filtered_data.groupBy("region").agg((max("temperature") - min("temperature")).alias("temp_change"))
```

Heatmap Data Query:

```python
# PySpark example
# Aggregate temperature change by geographic coordinates
heatmap_data = filtered_data.groupBy("longitude", "latitude").agg(avg("temperature_change").alias("avg_temp_change"))
```

Other Helpful Querries:
Receiving Station ID closest to clicked on map

```python
stations_df.createOrReplaceTempView("stations")
query = f"""
SELECT STAID, longitude, latitude,
       sqrt(pow(longitude - {mouse_x}, 2) + pow(latitude - {mouse_y}, 2)) as distance
FROM stations
ORDER BY distance
LIMIT 1
"""
nearest_station = spark.sql(query).collect()[0]
```

Receiving date and temperature (depending on which one is needed - max,min,mean):

```
data_df.createOrReplaceTempView("data")
temp_query = f"""
        SELECT date, Tx
        FROM data
        WHERE station_id = {nearest_station.station_id}
        """
temperature_data = spark.sql(temp_query).collect()
for row in temperature_data:
    print(f"Date: {row.date}, Temperature: {row.temperature}")
```

The example code for handling mouse press and creating a simple Europe map will be included.

6. Conclusion

Summary of the visualization plan: This document outlines the plan for visualizing temperature trends in Europe, including the types of plots to be used, axis configurations, and data queries. The visualizations aim to provide clear insights into temperature changes over time and across different regions.

Next steps and considerations for implementation:

- Implement the data extraction and transformation queries.
- Develop the visualizations using a suitable plotting library (e.g., Matplotlib, Seaborn).
- Ensure the visualizations are integrated into the project and available on the GitHub repository.
- Update the documentation as needed based on feedback and additional requirements.