02 PySpark Weather Data Analysis Assignment

PySpark Weather Data Analysis Assignment

Assignment Overview

In this assignment, you will work with historical weather data to analyze temperature patterns using Apache Spark's distributed computing capabilities. You'll implement data processing pipelines to find minimum and maximum temperatures recorded by various weather stations.

Learning Objectives

By completing this assignment, you will:

- Understand the fundamentals of Apache Spark and PySpark
- Learn to process large datasets using Spark RDDs (Resilient Distributed Datasets)
- Practice data transformation operations like `map`, `filter`, and `reduceByKey`
- Work with real-world CSV weather data
- Apply functional programming concepts in distributed computing

Dataset Description

You will be working with historical weather data from the year 1800 in CSV format. The dataset contains weather measurements from multiple weather stations worldwide.

Dataset URL:
https://raw.githubusercontent.com/KirkYagami/learn_databricks/refs/heads/main/04_Spark/03_Datasets/1800.csv

Data Format

Each line in the CSV file contains the following comma-separated fields:

1. **Station ID** (String): Unique identifier for the weather station
2. **Date** (String): Date of the measurement
3. **Measurement Type** (String): Type of measurement (TMIN, TMAX, PRCP, etc.)
4. **Temperature** (Integer): Temperature value in tenths of degrees Celsius
5. **Additional fields** (varies): Other metadata fields

Key Measurement Types

- **TMIN**: Minimum temperature for the day
- **TMAX**: Maximum temperature for the day
- **PRCP**: Precipitation amount
- Other types may be present but are not relevant for this assignment

Assignment Tasks

Task 1: Minimum Temperature Analysis

Write a PySpark program that analyzes the dataset to find the **lowest minimum temperature** recorded by each weather station.

Requirements:

1. Set up a Spark configuration with application name "MinTemperatures"
2. Read the weather data from the CSV file
3. Parse each line to extract station ID, measurement type, and temperature
4. Convert temperature from tenths of degrees Celsius to Fahrenheit using the formula:

```
Fahrenheit = (Celsius * 0.1) * (9/5) + 32
```

5. Filter the data to include only minimum temperature readings (TMIN)
6. Find the lowest minimum temperature for each station
7. Display results in the format: `StationID Temperature°F` (rounded to 2 decimal places)

Task 2: Maximum Temperature Analysis

Write a PySpark program that analyzes the dataset to find the **highest maximum temperature** recorded by each weather station.

Requirements:

1. Set up a Spark configuration with application name "MaxTemperatures"
2. Follow similar steps as Task 1, but focus on maximum temperature readings (TMAX)
3. Find the highest maximum temperature for each station
4. Apply the same temperature conversion formula
5. Display results in the same format as Task 1

Task 3: Code Analysis and Optimization

Answer the following questions in a separate document:

1. **Explain the Spark Operations**: Describe what each of the following operations does in the context of your programs:
   - `map()`
   - `filter()`
   - `reduceByKey()`
   - `collect()`

2. **Performance Considerations**:
   - Why is `reduceByKey()` preferred over `groupByKey()` for this type of operation?
   - What happens when you call `collect()`? When should you be cautious about using it?
3. **Data Processing Pipeline**: Draw or describe the data flow from reading the CSV file to displaying the final results.
4. **Potential Improvements**: Suggest two ways you could optimize or extend these programs.

Technical Requirements

Environment Setup

- Google Colab environment
- PySpark will be installed directly in the notebook
- Access to the provided weather dataset (hosted on GitHub)

Installation Instructions for Google Colab

Add these commands at the beginning of your notebook:

```python
# Install PySpark
!pip install pyspark

# Import required libraries
from pyspark import SparkConf, SparkContext
import os
```

Data Access

Since you're using Google Colab, you'll need to:

1. Download the dataset from the GitHub URL directly in your notebook
2. Use the appropriate file path for Colab's file system

Example for loading data in Colab:

```python
# Download dataset (replace with actual GitHub URL)
!wget "YOUR_GITHUB_DATASET_URL" -O weather_data.csv

# Read the file in your Spark program
lines = sc.textFile("weather_data.csv")
```

File Structure

Your submission should include:

```
assignment_submission/
├── Weather_Analysis.ipynb (Jupyter notebook with both tasks)
├── analysis_answers.md (or include answers in notebook markdown cells)
└── README.md (optional)
```

Google Colab Guidelines

- Use separate code cells for Task 1 and Task 2
- Include markdown cells to explain your approach
- Make sure to properly initialize Spark in each section
- Remember to stop the SparkContext when switching between tasks
- Use clear section headers in your notebook

Code Guidelines

- Use meaningful variable names
- Include comments explaining complex operations
- Use the appropriate file path for Google Colab environment
- Follow Python PEP 8 style guidelines
- Include markdown cells to document your process and findings
- Make sure to properly initialize and stop SparkContext between tasks

Expected Output Format

Sample Output for Minimum Temperatures:

```
ITE00100554    5.36°F
GM000010962    -11.56°F
EZE00100082    7.70°F
...
```

Sample Output for Maximum Temperatures:

```
ITE00100554    90.14°F
GM000010962    89.42°F
EZE00100082    91.58°F
...
```

Submission Instructions

1. Complete both temperature analysis tasks in Google Colab
2. Test your code with the provided dataset

3. Answer all questions in Task 3 (can be in markdown cells or separate document)
4. Download your completed notebook (.ipynb file)
5. Submit the notebook file and any additional documents

Evaluation Criteria

- **Correctness**: Programs produce accurate results and handle data properly
- **Code Quality**: Clean, readable, well-commented code following best practices
- **Analysis**: Thoughtful answers to the questions in Task 3
- **Documentation**: Clear explanations in markdown cells

Hints and Tips

1. Pay careful attention to the temperature conversion formula
2. Remember that the temperature in the dataset is in tenths of degrees Celsius
3. Test your parsing function with a few sample lines before processing the entire dataset
4. Use Spark's lazy evaluation to your advantage - transformations are not executed until an action is called
5. Consider the data types when performing operations (string vs. numeric fields)
6. In Google Colab, you may need to restart and clear output between major code changes
7. Use `sc.stop()` before creating a new SparkContext if you need to restart

Common Pitfalls to Avoid

- Forgetting to convert temperature from tenths of degrees
- Mixing up TMIN and TMAX in the filtering logic
- Not properly setting up the file path for Google Colab environment
- Using `groupByKey()` instead of `reduceByKey()` for aggregation operations
- Creating multiple SparkContexts without properly stopping the previous one

Resources

- PySpark Documentation
- Spark Programming Guide
- Weather Data Format Documentation