Name: Barquilla, Meshe Mae N. Lab No. 1

Git Repo/ Colab Link: https://github.com/meshemae25/Barquilla.git Date: 26/01/2025

Objective
The objective of this lab was to learn how to create and execute a Spark RDD pipeline. The primary focus was on applying five transformations to a sample dataset, specifically filtering out specific words, sorting, and removing duplicates.

Introduction
In this lab, I worked with Spark RDDs (Resilient Distributed Datasets), which are essential for efficiently handling large datasets in parallel using Apache Spark. I applied five commonly used transformations—*flatMap, map, filter, distinct,* and *sortBy*—on a small dataset to understand how each transformation operates and contributes to data manipulation.

Methodology

1. **Start Spark Session**: The first step was to initialize the Spark session and obtain the SparkContext to interact with RDDs.
2. **Sample Data**: A small list of fruit names was used as the sample dataset for the experiment.
3. **Create RDD**: I created an RDD from the sample dataset using the *sc.parallelize*() function.
4. **Transformation 1 - FlatMap**: I applied the *flatMap* transformation to split each string in the dataset into individual words.
5. **Transformation 2 - Map**: The next transformation, map, was used to convert all the words to lowercase.
6. **Transformation 3 - Filter**: I then used the *filter* transformation to remove the word 'apple' from the dataset.
7. **Transformation 4 - Distinct**: To eliminate any duplicate words, I applied the distinct transformation.
8. **Transformation 5 - SortBy**: The final transformation was *sortBy*, which sorted the words in alphabetical order.
9. **Action - Collect**: Once the transformations were completed, I used *collect()* to retrieve and display the final results.
10. **Stop Spark Session**: After completing all tasks, I stopped the Spark session to ensure proper cleanup.

Results and Analysis

- **Original RDD**: ["apple banana", "orange apple", "banana orange", "apple mango", "grape apple"]
- **After FlatMap**: ["apple", "banana", "orange", "apple", "banana", "orange", "apple", "mango", "grape", "apple"]
- **After Map (Lowercase)**: ["apple", "banana", "orange", "apple", "banana", "orange", "apple", "mango", "grape", "apple"]
- **After Filter (Remove 'apple')**: ["banana", "orange", "banana", "orange", "mango", "grape"]
- **After Distinct**: ["banana", "orange", "mango", "grape"]
- **After SortBy (Alphabetical Order)**: ["banana", "grape", "mango", "orange"]

Challenges and Solutions

- **Challenge**: Initially, I was uncertain about how the distinct transformation worked, particularly after filtering out the word 'apple'.

- **Solution**: Upon reviewing the Spark documentation, I realized that the *distinct* transformation removes any duplicate values from the RDD, which was exactly what I needed after filtering out 'apple'.

Conclusion:
This lab was an insightful exercise in understanding how to use Spark RDDs and apply various transformations to manipulate data. By utilizing transformations such as *flatMap, map, filter, distinct,* and *sortBy,* I gained hands-on experience in processing and cleaning data. In general, this exercise enhanced my understanding of how Spark can be leveraged for data processing tasks.