Name: **Matamorosa, Janine Ishe M.** Lab No. 1

Git Repo/ Colab Link: https://github.com/gUaanineJIM/Matamorosa_LabAct.git Date: Jan 25, 2025

Objective

This activity focuses on learning how to manipulate and analyze large datasets using Spark RDDs in Python. My objective is to learn how to effectively use these transformations, including map, filter, flatMap, distinct, and sortBy, to manipulate and analyze data efficiently.

Introduction

This lab introduces Apache Spark, a tool for handling huge datasets. It concentrates on RDDs, which are unique data structures that may be shared between many computers. Using PySpark, Spark with Python, I will learn how to deal with and analyze data effectively using RDD operations such as map, filter, flatMap, distinct, and sortBy which I will be showing in this laboratory.

Methodology

This lab will show how to use different operations in PySpark such as map, filter, flatMap, distinct, and sortBy which I will be showing in this laboratory.

1. Initialize Spark Session
2. Create RDD from a List

```
data = ["Josefina", "Jasmin", "Jorjeana", "Jorge", "Pastisa",
"Rosefta", "Kumaqr", "Katherine", "Keizny", "Kiwran"]
```

3. Apply the five transformations.
    a. map()
    b. filter()
    c. flatMap()
    d. distinct()
    e. sortBy()
4. Data Flow:
    - The list of names goes through a series of steps, with each step creating a new version of the data. Each step builds upon the previous one. This process starts with a map() operation and ends with sort().
5. Show Final Results.

Results and Analysis

RESULTS

1. **map()**

```
rdd_map = rdd.map(lambda s: s + "_Moral")
result_map = rdd_map.collect()
print("Appending '_Moral' to each name using map()")
print(result_map)
```

✓ 0.8s Python

```
Appending '_Moral' to each name using map()
['Josefina_Moral', 'Jasmin_Moral', 'Jorjeana_Moral', 'Jorge_Moral', 'Pastisa_Moral', 'Rosefta_Moral', 'Kumaqr_Moral', '
```

Before transformation:

data = ["Josefina", "Jasmin", "Jorjeana", "Jorge", "Pastisa", "Rosefta", "Kumaqr", "Katherine", "Keizny", "Kiwran"]

After Transformation:

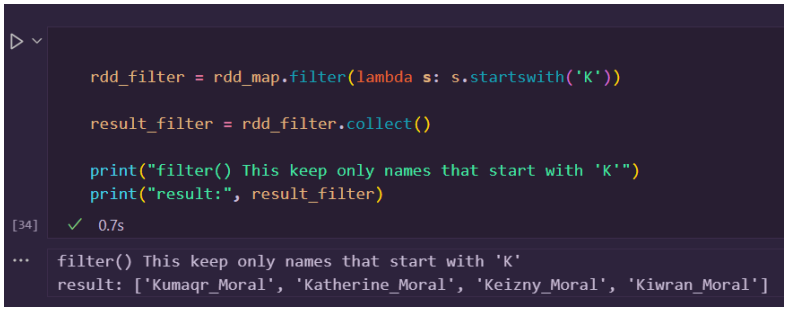
Appending '_Moral' to each name using map()

['Josefina_Moral', 'Jasmin_Moral', 'Jorjeana_Moral', 'Jorge_Moral', 'Pastisa_Moral', 'Rosefta_Moral', 'Kumaqr_Moral', 'Katherine_Moral', 'Keizny_Moral', 'Kiwran_Moral']

OUTPUT Analysis:

The original list of names was transformed by adding "_Moral" to each name using the map() function, resulting in a new list with modified names.

2. **filter()**



Before transformation:

Appending '_Moral' to each name using map()

['Josefina_Moral', 'Jasmin_Moral', 'Jorjeana_Moral', 'Jorge_Moral', 'Pastisa_Moral', 'Rosefta_Moral', 'Kumaqr_Moral', 'Katherine_Moral', 'Keizny_Moral', 'Kiwran_Moral']

After Transformation:

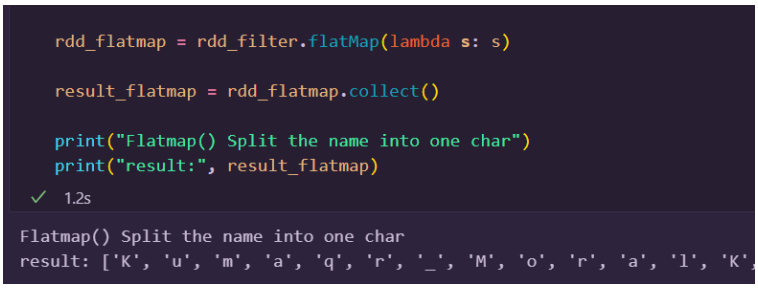
filter() This keep only names that start with 'K'

result: ['Kumaqr_Moral', 'Katherine_Moral', 'Keizny_Moral', 'Kiwran_Moral']

OUTPUT Analysis:

The list of names was filtered to keep only those that start with the letter 'K'.

3. **flatMap()**



Before transformation:

filter() This keep only names that start with 'K'

result: ['Kumaqr_Moral', 'Katherine_Moral', 'Keizny_Moral', 'Kiwran_Moral']

After Transformation:

Flatmap() Split the name into one char

result: ['K', 'u', 'm', 'a', 'q', 'r', '_', 'M', 'o', 'r', 'a', 'l', 'K', 'a', 't', 'h', 'e', 'r', 'i', 'n', 'e', '_', 'M', 'o', 'r', 'a', 'l', 'K', 'e', 'i', 'z', 'n', 'y', '_', 'M', 'o', 'r', 'a', 'l', 'K', 'i', 'w', 'r', 'a', 'n', '_', 'M', 'o', 'r', 'a', 'l']

OUTPUT Analysis:

The list of names was split into individual characters using the flatMap() transformation.

4. **distinct()**

```
rdd_distinct = rdd_flatmap.distinct()

result_distinct = rdd_distinct.collect()

print("distinct() transformation: Remove duplicate characters")
print("result:", result_distinct)
```

```
[36]  ✓ 1.8s
...  distinct() transformation: Remove duplicate characters
     result: ['K', 'u', 'm', 'a', 'q', 'r', '_', 'M', 'o', 'l', 't', 'h', 'e',
```

Before transformation:

Flatmap() Split the name into one char

result: ['K', 'u', 'm', 'a', 'q', 'r', '_', 'M', 'o', 'r', 'a', 'l', 'K', 'a', 't', 'h', 'e', 'r', 'i', 'n', 'e', '_', 'M', 'o', 'r', 'a', 'l', 'K', 'e', 'i', 'z', 'n', 'y', '_', 'M', 'o', 'r', 'a', 'l', 'K', 'i', 'w', 'r', 'a', 'n', '_', 'M', 'o', 'r', 'a', 'l']

After Transformation:

distinct() transformation: Remove duplicate characters

result: ['K', 'u', 'm', 'a', 'q', 'r', '_', 'M', 'o', 'l', 't', 'h', 'e', 'i', 'n', 'z', 'y', 'w']

OUTPUT Analysis:

Duplicate characters were removed from the list.

5. **sortBy()**

```
rdd_sorted = rdd_distinct.sortBy(lambda s: s)

result_sorted = rdd_sorted.collect()

print("sortBy() transformation: Sort the distinct characters alphabetically")
print("result:", result_sorted)
```

```
✓ 0.7s
sortBy() transformation: Sort the distinct characters alphabetically
result: ['K', 'M', '_', 'a', 'e', 'h', 'i', 'l', 'm', 'n', 'o', 'q', 'r', 't', 'u', 'w', 'y', 'z']
```

Before transformation:

distinct() transformation: Remove duplicate characters

result: ['K', 'u', 'm', 'a', 'q', 'r', '_', 'M', 'o', 'l', 't', 'h', 'e', 'i', 'n', 'z', 'y', 'w']

After Transformation:

sortBy() transformation: Sort the distinct characters alphabetically

result: ['K', 'M', '_', 'a', 'e', 'h', 'i', 'l', 'm', 'n', 'o', 'q', 'r', 't', 'u', 'w', 'y', 'z']

OUTPUT Analysis:

The list of unique characters was arranged in alphabetical order using the sortBy() transformation.

Challenges and Solutions

I was using SparkContext to set up Spark, but it caused errors because Spark doesn't allow more than one SparkContext at a time.

```
from pyspark import SparkContext

sc = SparkContext("local", "UseMap")
```

I switched to SparkSession, and it worked because it automatically takes care of everything, making it easier to use and adding more features.

```
from pyspark.sql import SparkSession
spark = SparkSession.builder.master("local").appName("UseMap").getOrCreate()

# List of names
data = ["Josefina", "Jasmin", "Jorjeana", "Jorge", "Pastisa", "Rosefta", "Kuma
```

Conclusion

Some key takeaways from the lab include hands-on experience with adding suffixes, filtering depending on specifications, splitting data, deleting duplicates, and sorting data, which are all important processes in data analysis. Yes, I completed the objectives. I learnt how to handle and analyze data with PySpark's RDD operations. I became more familiar with Spark's RDD operations and how they might be used at various phases of data processing.