SIDDHI S. KOTRE
ROLL NO: 50

Assignment 7
Big Data Analytics and Visualization
Lab 7 : Apache Spark Assignment

Aim: To perform data processing tasks in Apache Spark using Databricks, including reading files into RDDs, applying transformations like map and filter, and using actions like reduce for data manipulation and aggregation.

Theory:

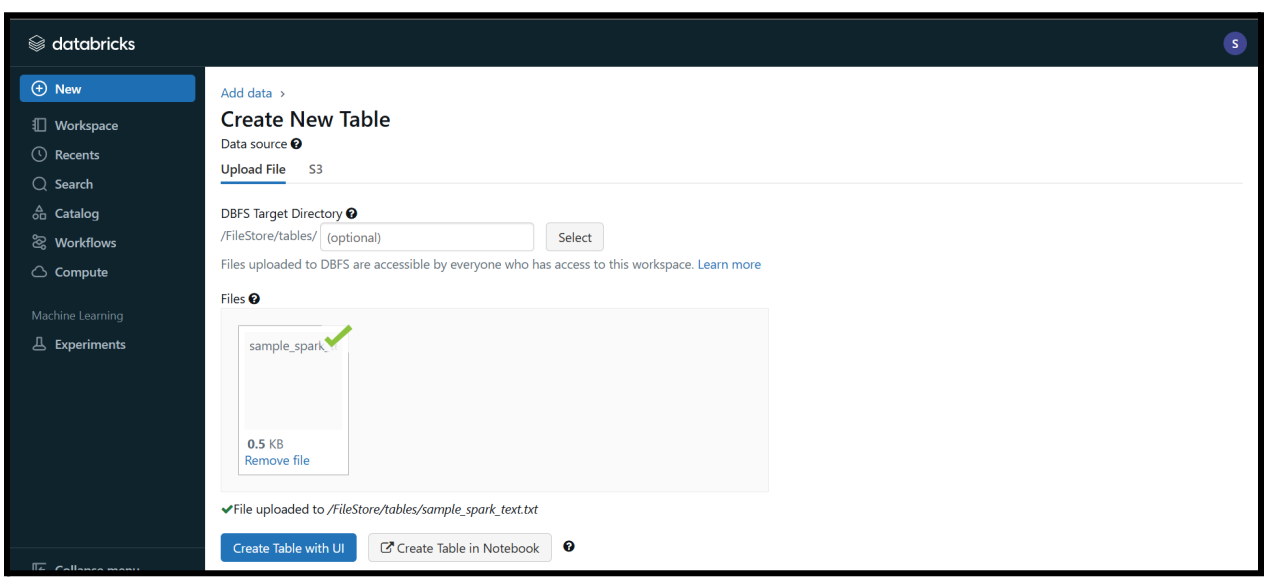
Apache Spark is a powerful distributed computing system that allows for processing large datasets across clusters. It supports a range of transformations and actions on Resilient Distributed Datasets (RDDs), enabling efficient data processing. Databricks provides a collaborative environment with cloud infrastructure, making it an ideal platform for performing big data operations with Spark.

Resilient Distributed Datasets (RDDs) are Spark's core abstraction, providing fault-tolerant, parallel data structures that allow operations on large data sets. RDDs support two main types of operations:

1. Transformations (e.g., map, filter): These create new RDDs from existing ones, enabling data transformation without modifying the original dataset.
2. Actions (e.g., reduce, collect): These trigger computation and return results.

Steps:

Add the .txt and .csv files to FileStore in DataBricks.

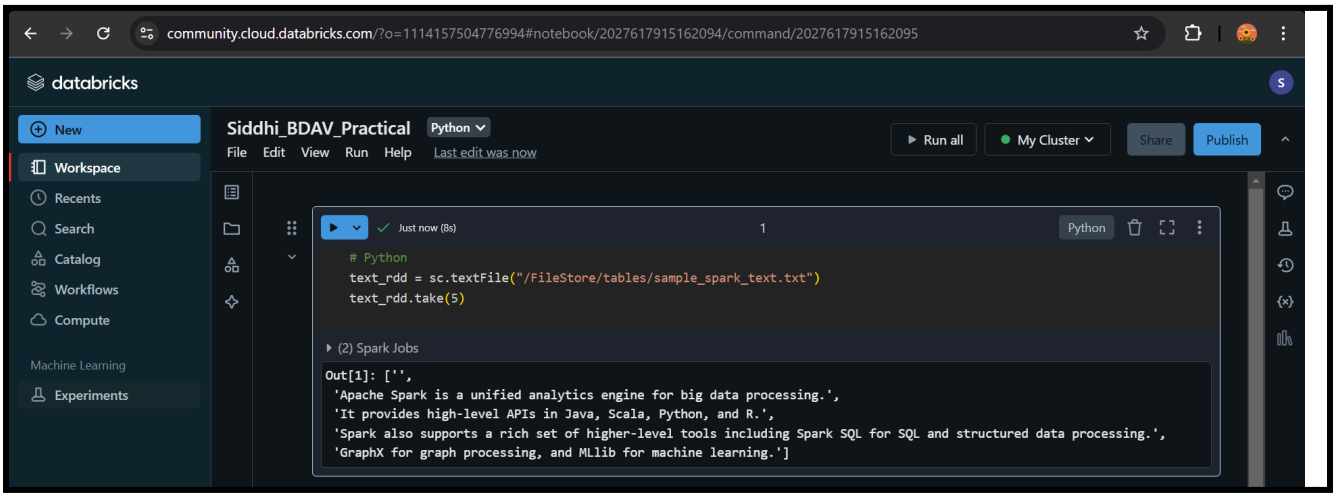


1. Read the .txt file into RDD

Code:

text_rdd = sc.textFile("/FileStore/tables/sample_spark_text.txt")
text_rdd.take(5)

Output:

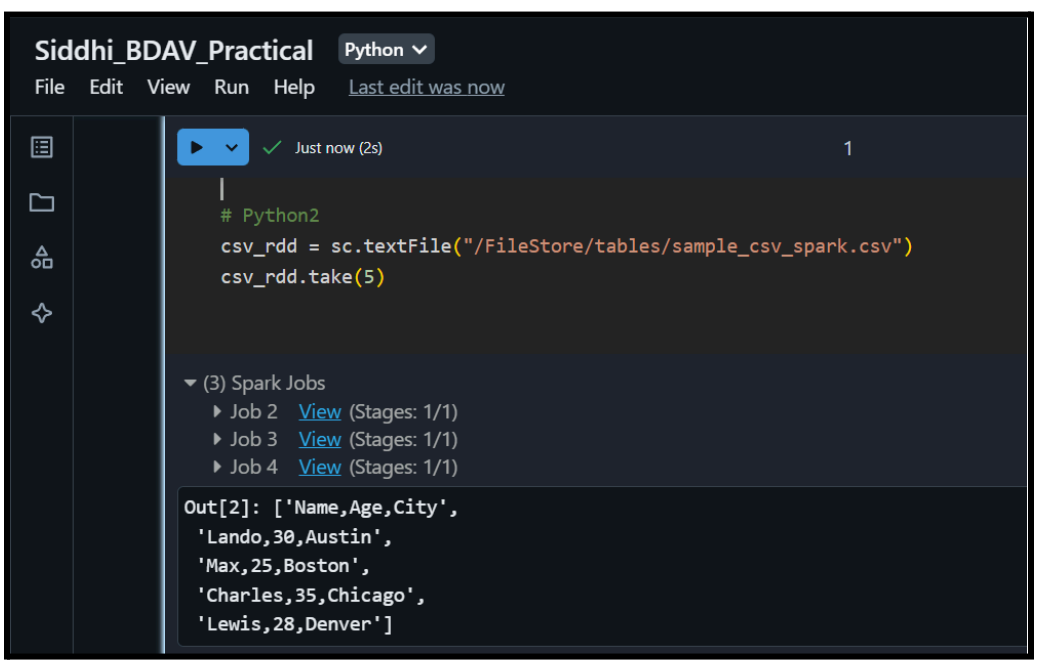


2. Read CSV file into RDD

Code:

csv_rdd = sc.textFile("/FileStore/tables/sample_csv_spark.csv")
csv_rdd.take(5)

Output:

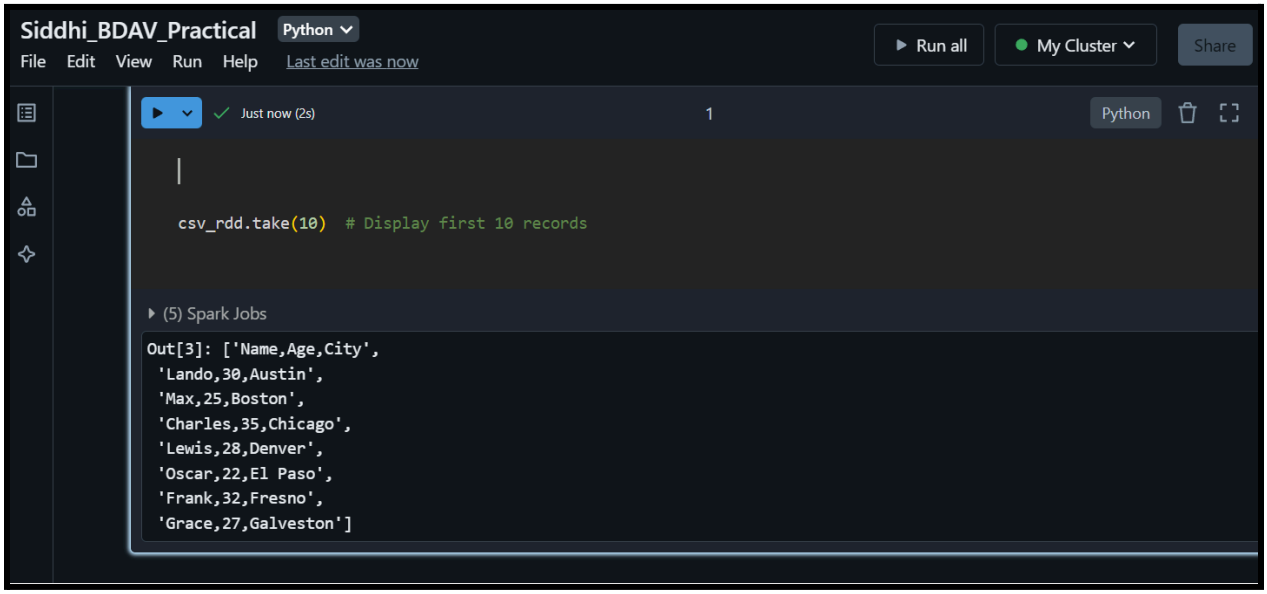


3. Display a limited number of record

Code:

csv_rdd.take(10) # Display first 10 records

Output:



4. Convert data into uppercase using the map function

Code:

uppercase_rdd = csv_rdd.map(lambda line: line.upper())

uppercase_rdd.take(5)

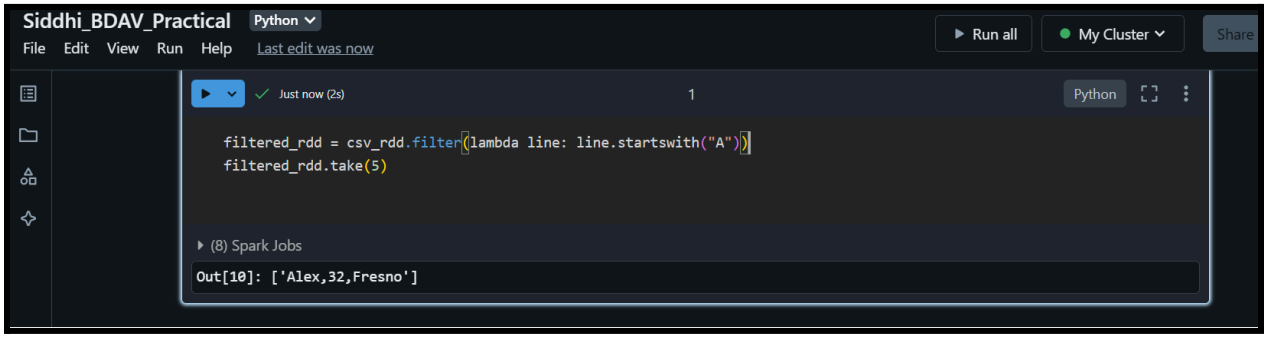
Output:

5. Display data Start with A using the filter function

Code:

```
filtered_rdd = csv_rdd.filter(lambda line: line.startswith("A"))
filtered_rdd.take(5)
```

Output:



6. Skip header from CSV file

Code:

```
header = csv_rdd.first()
data_rdd = csv_rdd.filter(lambda line: line != header)
data_rdd.take(5)
```

Output:

7. Read multiple CSV files into RDD

Code:

```
multi_csv_rdd = sc.textFile("/FileStore/tables/*.csv")
multi_csv_rdd.take(5)
```

Output:

8. Reduce a list – Calculate min, max, and total of elements

Code:

```
num_rdd = sc.parallelize([4, 5, 8, 6, 10, 2])
min_val = num_rdd.reduce(lambda a, b: a if a < b else b)
max_val = num_rdd.reduce(lambda a, b: a if a > b else b)
sum_val = num_rdd.reduce(lambda a, b: a + b)

print("Min:", min_val)
print("Max:", max_val)
print("Total:", sum_val)
```

Output:

9. Reduce function on Tuple RDD(String,Int)

Code:

```
tuple_rdd = sc.parallelize([("apple", 10), ("banana", 20), ("apple", 30)])
reduced_rdd = tuple_rdd.reduceByKey(lambda a, b: a + b)
reduced_rdd.collect()
```

Output:

10. implement word count in Apache spark using a map and reduce

Code:

```
word_rdd = text_rdd.flatMap(lambda line: line.split(" "))
word_count = word_rdd.map(lambda word: (word, 1)).reduceByKey(lambda a, b: a + b)
word_count.collect()
```

Output: