Mini project

Write MapReduce/Spark Program to perform

1. Matrix Vector Multiplication

```
from pyspark import
SparkContext, SparkConf

# Initialize SparkContext
conf =
SparkConf().setAppName("MatrixV
ectorMultiplication")
sc = SparkContext(conf=conf)

# Input matrix and vector
matrix = [
    (0, [1, 2, 3, 4]),
    (1, [5, 6, 7, 8]),
    (2, [9, 10, 11, 12])
]
vector = [2, 4, 6, 8]

# Broadcast the vector to all
nodes in the cluster
broadcast_vector =
sc.broadcast(vector)

# Perform matrix-vector
multiplication using RDDs
result = sc.parallelize(matrix)
\
    .map(lambda row: (row[0],
sum(a * b for a, b in
zip(row[1],
broadcast_vector.value)))) \
    .collect()

# Print the result
for row_id, value in
sorted(result):
    print(f"Row {row_id}:
{value}")
```

```
# Stop SparkContext
sc.stop()
```

Output:

```
PS C:\Users\priya\OneDrive\Desktop\BDA Project\bda-mini-project>
Setting default log level to "WARN".
To adjust logging level use sc.setLogLevel(newLevel). For SparkR,
24/03/29 11:47:20 WARN NativeCodeLoader: Unable to load native-ha
Row 0: 60
Row 1: 140
Row 2: 220
```

2. **Aggregations - Mean, Sum, Std Deviation**

```
from pyspark.sql import SparkSession
from pyspark.sql.functions import mean, sum as sql_sum, stddev

# Initialize SparkSession
spark = SparkSession.builder \
    .appName("AggregationSpark") \
    .getOrCreate()

# Dummy input data
input_data = [
    ('key1', 10),
    ('key2', 20),
    ('key1', 30),
    ('key2', 40),
    ('key1', 50),
    ('key2', 60)
]

# Create DataFrame from input data
df = spark.createDataFrame(input_data, ["key", "value"])

# Perform aggregation using DataFrame API
result_df = df.groupBy("key") \
    .agg(
        mean("value").alias("mean"),
        sql_sum("value").alias("sum"),
        stddev("value").alias("std_dev")
```

```
    )



# Show the result

result_df.show()



# Stop SparkSession

spark.stop()
```

Output:

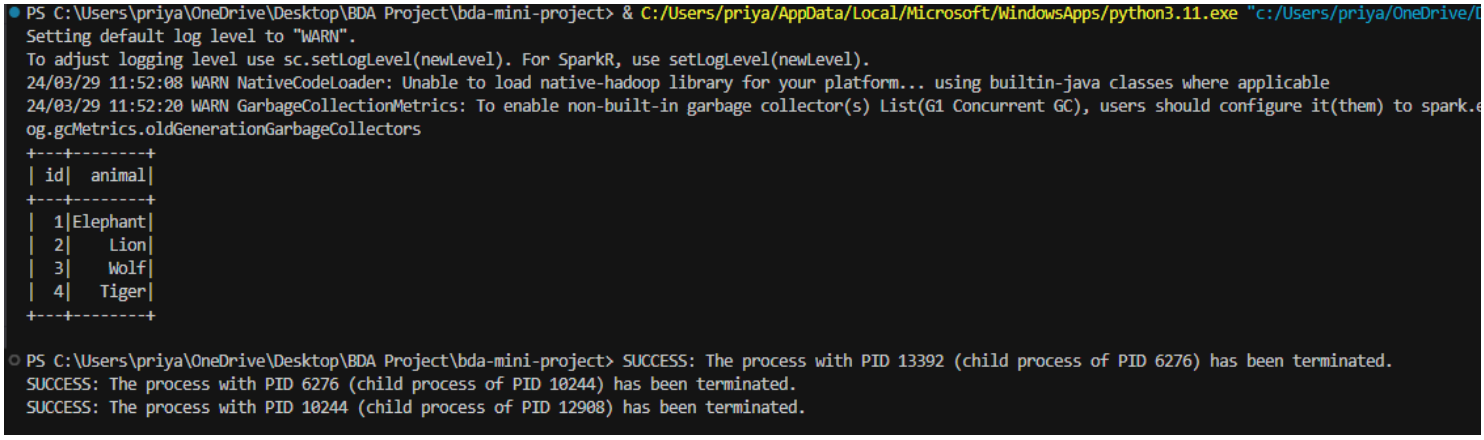
```
PS C:\Users\priya\OneDrive\Desktop\BDA Project\bda-mini-project> & C:/Users/priya/AppData/Local/Microsoft/WindowsApps/p
Setting default log level to "WARN".
To adjust logging level use sc.setLogLevel(newLevel). For SparkR, use setLogLevel(newLevel).
24/03/29 11:49:35 WARN NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java c
24/03/29 11:49:48 WARN GarbageCollectionMetrics: To enable non-built-in garbage collector(s) List(G1 Concurrent GC), us
og.gcMetrics.oldGenerationGarbageCollectors
+----+----+---+-------+
| key|mean|sum|std_dev|
+----+----+---+-------+
|key1|30.0| 90|   20.0|
|key2|40.0|120|   20.0|
+----+----+---+-------+

PS C:\Users\priya\OneDrive\Desktop\BDA Project\bda-mini-project> SUCCESS: The process with PID 13288 (child process of
SUCCESS: The process with PID 14008 (child process of PID 9712) has been terminated.
SUCCESS: The process with PID 9712 (child process of PID 7236) has been terminated.
```

3. **Sort the data**

```
from pyspark.sql import SparkSession
from pyspark.sql.functions import col

# Create a Spark session
spark = SparkSession.builder \
    .appName("SortData") \
    .getOrCreate()

# Define dummy input data
dummy_data = [
    ("3", "Wolf"),
    ("1", "Elephant"),
    ("2", "Lion"),
    ("4", "Tiger")
]

# Create DataFrame from dummy data
df = spark.createDataFrame(dummy_data, ["id", "animal"])

# Sort the DataFrame
sorted_df = df.orderBy(col("id"))

# Show the sorted DataFrame
sorted_df.show()

# Stop the Spark session
spark.stop()
```

Name: Priyanshu Hariani **Class:** D20A **Roll No:** 17

Output:



4. **Search a data element**

```
from pyspark.sql import SparkSession
from pyspark.sql.functions import col

# Create a Spark session
spark = SparkSession.builder \
    .appName("SearchElement") \
    .getOrCreate()

# Define the data to be searched
data = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]

# Create DataFrame from the data
df = spark.createDataFrame([(x,) for x in data], ["value"])

# Define the search element
search_element = 11  # Change the search element as needed

# Search for the element in the DataFrame
result_df = df.filter(col("value") == search_element)

# Check if the element is found
if result_df.count() > 0:
    print("Element found in the dataset")
else:
    print("Element not found in the dataset")

# Stop the Spark session
spark.stop()
```

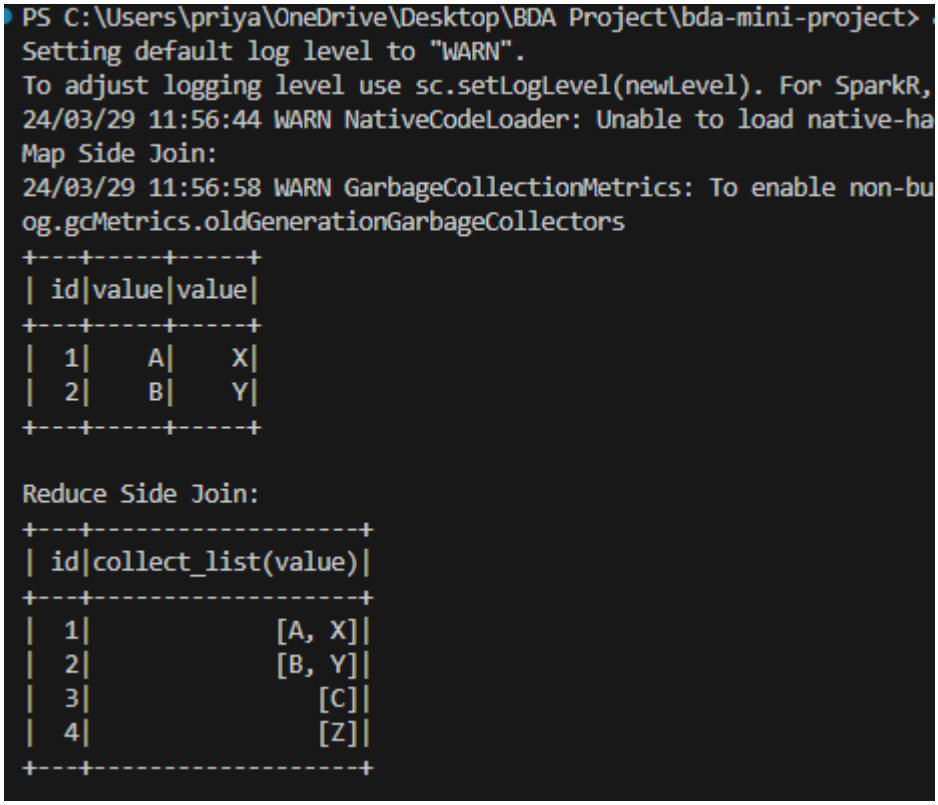
Output:

```
PS C:\Users\priya\OneDrive\Desktop\BDA Project\bda-mini-project>
Setting default log level to "WARN".
To adjust logging level use sc.setLogLevel(newLevel). For SparkR
24/03/29 11:54:29 WARN NativeCodeLoader: Unable to load native-ha
Element not found in the dataset
```

5. **Joins - Map Side and Reduce Side**

```
from pyspark.sql import SparkSession

# Create a Spark session
spark = SparkSession.builder \
    .appName("JoinExample") \
    .getOrCreate()

# Create DataFrames for left and right datasets
left_data = spark.createDataFrame([(1, "A"), (2, "B"), (3, "C")], ["id",
"value"])
right_data = spark.createDataFrame([(1, "X"), (2, "Y"), (4, "Z")], ["id",
"value"])

# Perform map-side join
map_join = left_data.join(right_data, on="id", how="inner")

# Perform reduce-side join
reduce_join = left_data.union(right_data).groupBy("id").agg({"value":
"collect_list"})

# Show the results
print("Map Side Join:")
map_join.show()

print("Reduce Side Join:")
reduce_join.show()

# Stop the Spark session
spark.stop()
```

Output:


PS C:\Users\priya\OneDrive\Desktop\BDA Project\bda-mini-project>
Setting default log level to "WARN".
To adjust logging level use sc.setLogLevel(newLevel). For SparkR,
24/03/29 11:56:44 WARN NativeCodeLoader: Unable to load native-ha
Map Side Join:
24/03/29 11:56:58 WARN GarbageCollectionMetrics: To enable non-bu
og.gcMetrics.oldGenerationGarbageCollectors
+---+-----+-----+
| id|value|value|
+---+-----+-----+
| 1| A| X|
| 2| B| Y|
+---+-----+-----+
Reduce Side Join:
+---+------------------+
| id|collect_list(value)|
+---+------------------+
1	[A, X]
2	[B, Y]
3	[C]
4	[Z]
+---+------------------+