PySpark RDD HandsOn 3rd September

SaiPrabath Chowdary S

RDD Exercise

```
[54] # Initialize SparkSession
     spark = SparkSession.builder.appName("SalesDataAnalysis").getOrCreate()
```

```
[55] sales_data = [
         ("ProductA", 100),
         ("ProductB", 150),
         ("ProductA", 200),
         ("ProductC", 300),
         ("ProductB", 250),
         ("ProductC", 100)
     ]

     # step 1 -> spark context
     sc = spark.sparkContext

     # step 2
     # task 1 -> creating RDD of sales_data and Printing the first few elements of the RDD
     sales_rdd = sc.parallelize(sales_data)
     print(sales_rdd.take(3))
```

```
[('ProductA', 100), ('ProductB', 150), ('ProductA', 200)]
```

```
[79] # step 3 -> Grouping and Aggregating Data

     # taks 2 -> Group data by product name
     grouped_sales_rdd = sales_rdd.groupByKey()
     print("Grouped data:")
     for k,v in grouped_sales_rdd.collect():
       print(k,list(v))

     # taks 3 -> Calculate total sales by product
     total_sales_by_product = sales_rdd.reduceByKey(lambda x, y: x + y)
     print("Total sales by product:")
     print(total_sales_by_product.collect())

     # taks 4 -> Sort products by total sales
     sorted_products = total_sales_by_product.sortBy(lambda x: x[1], ascending=False)
     print("Sorted products by total sales:")
     print(sorted_products.collect())
```

```
Grouped data:
ProductA [100, 200]
ProductB [150, 250]
ProductC [300, 100]
Total sales by product:
[('ProductA', 300), ('ProductB', 400), ('ProductC', 400)]
Sorted products by total sales:
[('ProductB', 400), ('ProductC', 400), ('ProductA', 300)]
```

```
[58] #step 4 -> Additional transformations

     # taks 5 -> Filter products with high sales
     high_sales_products = total_sales_by_product.filter(lambda x: x[1] > 300)
     print("Products with high sales:")
     print(high_sales_products.collect())

     # task 6 -> Combine Regional Sales Data

     # regional sales data RDD
     regional_sales_data = [
         ("ProductA", 50),
         ("ProductC", 150)
     ]
     regional_sales_rdd = sc.parallelize(regional_sales_data)

     # Combining the two RDDs
     combined_sales_rdd = sales_rdd.union(regional_sales_rdd)

     # Calculating new total sales
     new_total_sales_by_product = combined_sales_rdd.reduceByKey(lambda x, y: x + y)
     print("Combined sales data:")
     print(new_total_sales_by_product.collect())
```

```
Products with high sales:
[('ProductB', 400), ('ProductC', 400)]
Combined sales data:
[('ProductA', 350), ('ProductC', 550), ('ProductB', 400)]
```

```
[69] # step 5 ->  Perform Actions on the RDD

     # task 7 -> Count the number of distinct products
     distinct_products_count = sales_rdd.map(lambda x: x[0]).distinct().count()
     print("Count of distinct products:", distinct_products_count)

     # task 8 -> Identify the product with maximum sales
     max_sales = total_sales_by_product.max()[1]
     max_sales_products = total_sales_by_product.filter(lambda x: x[1] == max_sales)
     print("Products with maximum sales:", max_sales_products.map(lambda x: x[0]).collect())
```

```
Count of distinct products: 3
Products with maximum sales: ['ProductB', 'ProductC']
```

```
[84] # challenge -> Calculate the Average Sales per Product
     for k,v in grouped_sales_rdd.collect():
       print(k,sum(list(v))/len(list(v)))
```

```
ProductA 150.0
ProductB 200.0
ProductC 200.0
```