## Week 4 Assignment: Introduction to Apache Spark using Scala and PySpark

### Objective: Dive into Apache Spark’s capabilities using both PySpark and Scala in the respective shells, experimenting with data generation and transformations.

#### **1. Environment Initialization**

* As you did in the Week 2 and Week 3 assignments, begin by navigating to the appropriate directory:
* cd bellevue-bigdata  
  cd hadoop-hive-spark-hbase
* Start your Docker containers:
* docker-compose up -d
* Access the master container:
* docker-compose exec master bash

#### **2. Running a Built-in Spark Example with PySpark**

To get hands-on experience with PySpark, we will execute the provided SparkPi example using the following command in the master container:

spark-submit --class org.apache.spark.examples.SparkPi \  
 --master yarn \  
 --deploy-mode client \  
 --driver-memory 2g \  
 --executor-memory 1g \  
 --executor-cores 1 \  
 $SPARK\_HOME/examples/jars/spark-examples\*.jar \  
 10

**Deliverable:** Screenshot of the SparkPi output.

#### **3. Starting the Spark Scala Shell**

In the master container’s terminal, initiate the Spark Scala shell:

$SPARK\_HOME/bin/spark-shell --master yarn --driver-memory 2g --executor-memory 1g --executor-cores 1

#### **4. Generating and Printing Random Numbers in the Scala Shell**

Execute the following commands in the Spark Scala shell:

val numNumbers = 10000  
val numbers = (1 to numNumbers).map(\_ => scala.util.Random.nextInt(1000))  
val numbersRDD = sc.parallelize(numbers)  
numbersRDD.take(100).foreach(println)

**Deliverable:** Screenshot of the first 100 generated random numbers.

#### **5. Generating and Transforming Random Sentences in the Scala Shell**

Generate random sentences and apply a transformation of your choice:

val numberOfSentences = 1000  
val words = List("apple", "banana", "cherry", "date", "elderberry", "fig", "grape", "honeydew")  
val sentences = (1 to numberOfSentences).map(\_ => scala.util.Random.shuffle(words).take(scala.util.Random.nextInt(6) + 1).mkString(" ") + ".")  
val sentencesRDD = sc.parallelize(sentences)  
  
// Apply your custom transformation here  
val transformedSentences = sentencesRDD.map(sentence => /\* Your transformation code here \*/)  
transformedSentences.take(100).foreach(println)

**Instructions:** Modify the placeholder /\* Your transformation code here \*/ to apply a unique transformation to each sentence.

**Deliverable:** Screenshot of a segment of the generated transformed sentences and an explanation of your unique transformation.

## Shutting Down

Ensure all Docker containers are turned off with docker-compose down for each directory. If you’re using google cloud, please shut down your virtual machine to preserve cloud costs.