PRACTICAL 3 - WRITE A SPARK CODE TO HANDLE THE STREAMING OF DATA USING RDD AND DATA FRAME.

ENTER IN THE SPARK SHELL USING SPARK-SHELL , START THE HADOOP SERVICES (HDFS AND YARN , SPARK) BEFORE LAUNCHING THE SPARK SHELL.

STEP 1: IMPORT THE NECESSARY LIBRARIES

```
scala> import org.apache.spark.graphx.Edge
import org.apache.spark.graphx.Edge

scala> import org.apache.spark.graphx.Graph
import org.apache.spark.graphx.Graph

scala> import org.apache.spark.graphx.lib._
import org.apache.spark.graphx.lib._
```

STEP 2: DEFINE THE VERTEX AND EDGE DATA

```
scala> val verArray = Array(
     |   (1L, ("Philadelphia", 1580863)),
     |   (2L, ("Baltimore", 620961)),
     |   (3L, ("Harrisburg", 49528)),
     |   (4L, ("Wilmington", 70851)),
     |   (5L, ("New York", 8175133)),
     |   (6L, ("Scranton", 76089))
     | )
verArray: Array[(Long, (String, Int))] = Array((1,(Philadelphia,1580863)), (2,(Baltimore,620961)), (3,(Harrisburg,49528)), (4,(Wilmington,70851)), (5,(New Yo
rk,8175133)), (6,(Scranton,76089)))
```

```
scala> val edgeArray = Array(
     |   Edge(2L, 3L, 113),
     |   Edge(2L, 4L, 106),
     |   Edge(3L, 4L, 128),
     |   Edge(3L, 5L, 248),
     |   Edge(3L, 6L, 162),
     |   Edge(4L, 1L, 39),
     |   Edge(1L, 6L, 168),
     |   Edge(1L, 5L, 130),
     |   Edge(5L, 6L, 159)
     | )
edgeArray: Array[org.apache.spark.graphx.Edge[Int]] = Array(Edge(2,3,113), Edge(2,4,106), Edge(3,4,128), Edge(3,5,248), Edge(3,6,162), Edge(4,1,39), Edge(1,6
,168), Edge(1,5,130), Edge(5,6,159))
```

STEP 3: CREATE THE SPARK RDDS AND THE GRAPH

```
scala> val verRDD = sc.parallelize(verArray)
verRDD: org.apache.spark.rdd.RDD[(Long, (String, Int))] = ParallelCollectionRDD[0] at parallelize at <console>:34

scala> val edgeRDD = sc.parallelize(edgeArray)
edgeRDD: org.apache.spark.rdd.RDD[org.apache.spark.graphx.Edge[Int]] = ParallelCollectionRDD[1] at parallelize at <console>:34

scala> val graph = Graph(verRDD, edgeRDD)
graph: org.apache.spark.graphx.Graph[(String, Int),Int] = org.apache.spark.graphx.impl.GraphImpl@5a4fc46f
```

STEP 4: FILTER VERTICES BY POPULATION

```
scala> graph.vertices.filter {
     |   case (id, (city, population)) => population > 50000
     | }.collect.foreach {
     |   case (id, (city, population)) =>
     |     println(s"The population of $city is $population")
     | }
[Stage 0:>                 (0 + 0) / 2][Stage 1:>                 (0 + 0) / 2]24/04/12 22:45:50 WARN cluster.YarnScheduler: Initial job has not accepted an
y resources; check your cluster UI to ensure that workers are registered and have sufficient resources
The population of Wilmington is 70851
The population of Scranton is 76089
The population of Baltimore is 620961
The population of Philadelphia is 1580863
The population of New York is 8175133
```

STEP 5: TRAVERSE THE GRAPH EDGES

```
scala> for (triplet <- graph.triplets.collect) {
     |   println(s"The distance between ${triplet.srcAttr._1} and ${triplet.dstAttr._1} is ${triplet.attr} kilometers")
     | }
The distance between Baltimore and Harrisburg is 113 kilometers
The distance between Baltimore and Wilmington is 106 kilometers
The distance between Harrisburg and Wilmington is 128 kilometers
The distance between Harrisburg and New York is 248 kilometers
The distance between Philadelphia and New York is 130 kilometers
The distance between Philadelphia and Scranton is 168 kilometers
The distance between Harrisburg and Scranton is 162 kilometers
The distance between Wilmington and Philadelphia is 39 kilometers
The distance between New York and Scranton is 159 kilometers
```

STEP 6: FILTER EDGES BY DISTANCE

```
scala> graph.edges.filter {
     |   case Edge(city1, city2, distance) => distance < 150
     | }.collect.foreach {
     |   case Edge(city1, city2, distance) =>
     |     println(s"The distance between $city1 and $city2 is $distance")
     | }
The distance between 2 and 3 is 113
The distance between 2 and 4 is 106
The distance between 3 and 4 is 128
The distance between 1 and 5 is 130
The distance between 4 and 1 is 39
```