Hands On: Joining Graph Datasets Reading I Coursera

Hands On: Joining Graph Datasets

1. Create a new dataset
2. Join two datasets with JoinVertices
3. Join two datasets with outerJoinVertices
4. Create a new return type for the joined vertices

Run the Spark Shell

Open a terminal in the Cloudera Quick Start virtual machine by clicking **Applications**, **System Tools** then **Terminal**.

Once the terminal is open, start the Spark Shell.

```
spark-shell
```

It may take several seconds for the Spark Shell to start. Be patient and wait for the **scala>** prompt.

Create a new dataset

Set log level to error in order to suppress the info and warn messages so the output is easier to read.

```
import org.apache.log4j.Logger
import org.apache.log4j.Level

Logger.getLogger("org").setLevel(Level.ERROR)
Logger.getLogger("akka").setLevel(Level.ERROR)
```

Import the GraphX and RDD libraries.

```
import org.apache.spark.graphx._
import org.apache.spark.rdd._
```

Define a simple list of vertices containing five international airports.

input:

```
val airports: RDD[(VertexId, String)] = sc.parallelize(
    List((1L, "Los Angeles International Airport"),
      (2L, "Narita International Airport"),
      (3L, "Singapore Changi Airport"),
      (4L, "Charles de Gaulle Airport"),
      (5L, "Toronto Pearson International Airport")))
```

output:

```
airports: org.apache.spark.rdd.RDD[(org.apache.spark.graphx.VertexId,
String)] = ParallelCollectionRDD[0] at parallelize at <console>:29
```

Define a list of edges that will make up the flights.

Two airports are connected in this graph if there is a flight between them. We will assign a made up flight number to each flight.

input:

```
val flights: RDD[Edge[String]] = sc.parallelize(
  List(Edge(1L,4L,"AA1123"),
    Edge(2L, 4L, "JL5427"),
    Edge(3L, 5L, "SQ9338"),
    Edge(1L, 5L, "AA6653"),
    Edge(3L, 4L, "SQ4521")))
```

output:

```
flights: org.apache.spark.rdd.RDD[org.apache.spark.graphx.Edge[String]] =
ParallelCollectionRDD[1] at parallelize at <console>:29
```

Define the flightGraph graph from the airports vertices and the flights edges.

input:

```
val flightGraph = Graph(airports, flights)
```

output:

```
flightGraph: org.apache.spark.graphx.Graph[String,String] =
org.apache.spark.graphx.impl.GraphImpl@5051d760
```

Print the departing and arrival airport and the flight number for each triplet in the flightGraph graph.

Each triplet in the flightGraph graph represents a flight between two airports.

input:

```
flightGraph.triplets.foreach(t => println("Departs from: " + t.srcAttr + " -
Arrives at: " + t.dstAttr + " - Flight Number: " + t.attr))
```

output:

```
Departs from: Los Angeles International Airport - Arrives at: Charles de
Gaulle Airport - Flight Number: AA1123
Departs from: Los Angeles International Airport - Arrives at: Toronto Pearson
International Airport - Flight Number: AA6653
Departs from: Narita International Airport - Arrives at: Charles de Gaulle
Airport - Flight Number: JL5427
Departs from: Singapore Changi Airport - Arrives at: Charles de Gaulle
Airport - Flight Number: SQ4521
Departs from: Singapore Changi Airport - Arrives at: Toronto Pearson
International Airport - Flight Number: SQ9338
```

Define an AirportInformation class to store the airport city and code.

Lets define a dataset with airport information so we can practice joining the airport information dataset with the datasets that we have already defined.

input:

```
case class AirportInformation(city: String, code: String)
```

output:

```
defined class AirportInformation
```

Define the list of airport information vertices.

Note: We do not have airport information defined for each airport in **flightGraph** graph and we have airport information for airports not in **flightGraph** graph.

input:

```
val airportInformation: RDD[(VertexId, AirportInformation)] = sc.parallelize(
  List((2L, AirportInformation("Tokyo", "NRT")),
    (3L, AirportInformation("Singapore", "SIN")),
    (4L, AirportInformation("Paris", "CDG")),
    (5L, AirportInformation("Toronto", "YYZ")),
    (6L, AirportInformation("London", "LHR")),
    (7L, AirportInformation("Hong Kong", "HKG"))))
```

output:

```
airportInformation:
org.apache.spark.rdd.RDD[(org.apache.spark.graphx.VertexId,
AirportInformation)] = ParallelCollectionRDD[19] at parallelize at
<console>:31
```

Join two datasets with JoinVertices

In this first example we are going to use joinVertices to join the airport information **flightGraph** graph.

Create a mapping function that appends the city name to the name of the airport. The mapping function should return a string since that is the vertex attribute type of the flightsGraph graph.

input:

```
def appendAirportInformation(id: VertexId, name: String, airportInformation:
AirportInformation): String = name + ":"+ airportInformation.city
```

output:

```
appendAirportInformation: (id: org.apache.spark.graphx.VertexId, name:
String, airportInformation: AirportInformation)String
```

Use joinVertices on flightGraph to join the airportInformation vertices to a new graph called flightJoinedGraph using the appendAirportInformation mapping function.

input:

```
val flightJoinedGraph =  flightGraph.joinVertices(airportInformation)
```

```
(appendAirportInformation)
flightJoinedGraph.vertices.foreach(println)
```

output:

```
(4,Charles de Gaulle Airport:Paris)
(1,Los Angeles International Airport)
(3,Singapore Changi Airport:Singapore)
(5,Toronto Pearson International Airport:Toronto)
(2,Narita International Airport:Tokyo)
```

Join two datasets with outerJoinVertices

Use outerJoinVertices on flightGraph to join the airportInformation vertices with additional airportInformation such as city and code, to a new graph called flightOuterJoinedGraph usingthe => operator which is just syntactic sugar for creating instances of functions.

input:

```
val flightOuterJoinedGraph =
flightGraph.outerJoinVertices(airportInformation)((_,name,
airportInformation) => (name, airportInformation))
flightOuterJoinedGraph.vertices.foreach(println)
```

output:

```
(4,(Charles de Gaulle Airport,Some(AirportInformation(Paris,CDG))))
(1,(Los Angeles International Airport,None))
(3,(Singapore Changi Airport,Some(AirportInformation(Singapore,SIN))))
(5,(Toronto Pearson International
Airport,Some(AirportInformation(Toronto,YYZ))))
(2,(Narita International Airport,Some(AirportInformation(Tokyo,NRT))))
```

Use outerJoinVertices on flightGraph to join the airportInformation vertices with additional airportInformation such as city and code, to a new graph called flightOuterJoinedGraphTwobut this time printing 'NA' if there is no additional information.

input:

```
val flightOuterJoinedGraphTwo =
flightGraph.outerJoinVertices(airportInformation)((_, name,
airportInformation) => (name,
airportInformation.getOrElse(AirportInformation("NA","NA"))))
flightOuterJoinedGraphTwo.vertices.foreach(println)
```

output:

```
(4,(Charles de Gaulle Airport,AirportInformation(Paris,CDG)))
(1,(Los Angeles International Airport,AirportInformation(NA,NA)))
(3,(Singapore Changi Airport,AirportInformation(Singapore,SIN)))
(5,(Toronto Pearson International Airport,AirportInformation(Toronto,YYZ)))
(2,(Narita International Airport,AirportInformation(Tokyo,NRT)))
```

Create a new return type for the joined vertices

Create a case class called Airport to store the information for the name, city, and code of the airport.

input:

```
case class Airport(name: String, city: String, code: String)
```

output:

```
defined class Airport
```

Print the airportInformation with the name, city, and code within each other.

input:

```
  val flightOuterJoinedGraphThree =
flightGraph.outerJoinVertices(airportInformation)((_, name, b) => b match {
  case Some(airportInformation) => Airport(name, airportInformation.city,
airportInformation.code)
  case None => Airport(name, "", "")
})
flightOuterJoinedGraphThree.vertices.foreach(println)
```

output:

```
(4,Airport(Charles de Gaulle Airport,Paris,CDG))
```

```
(1,Airport(Los Angeles International Airport,,))
(3,Airport(Singapore Changi Airport,Singapore,SIN))
(5,Airport(Toronto Pearson International Airport,Toronto,YYZ))
(2,Airport(Narita International Airport,Tokyo,NRT))
```