PROJECT 2

INDIVIDUAL REPORT

Pavan Kumar Mupparam

ASU ID: 1223455688

SUMMARY:

The project serves as an example of how to analyze spatiotemporal big data using Spark Scala, a big data tool. By avoiding the need for subqueries, user-created functions are crucial in lowering the time complexity of data processing. For this research, three files in.csv format including monthly taxi trips from 2009 to 2012 are needed. The spark is loaded with these datasets. The creation and organization of DataFrame objects take the form of a table, which is comparable to a table in a relational database but offers faster runtimes.

HOTZONE TASKS:

Two hotspot analysis tasks are carried out in this study. Hotzone analysis is one of those responsibilities. this evaluation. This spatial query combines a point dataset with a rectangular dataset using a range join function. The total number of points contained within each rectangle is calculated.

```
  def ST_Contains(queryRectangle: String, pointString: String ): Boolean = {

    val rectangle_coordinates = queryRectangle.split(",")
    val target_point_coordinates = pointString.split(",")

    val point_x: Double = target_point_coordinates(0).trim.toDouble
    val point_y: Double = target_point_coordinates(1).trim.toDouble
    val rect_x1: Double = math.min(rectangle_coordinates(0).trim.toDouble, rectangle_coordinates(2).trim.toDouble)
    val rect_y1: Double = math.min(rectangle_coordinates(1).trim.toDouble, rectangle_coordinates(3).trim.toDouble)
    val rect_x2: Double = math.max(rectangle_coordinates(0).trim.toDouble, rectangle_coordinates(2).trim.toDouble)
    val rect_y2: Double = math.max(rectangle_coordinates(1).trim.toDouble, rectangle_coordinates(3).trim.toDouble)

    if ((point_x >= rect_x1) && (point_x <= rect_x2) && (point_y >= rect_y1) && (point_y <= rect_y2)) {
        return true
    }
    return false
  }
}
```

Description of ST Contains Function:

def ST_Contains (queryRectangle: String, pointString: String) takes two String data-type parameters. query Hotzone provides the coordinates for the rectangular boundary's geographic

location to a rectangle. The coordinates of the pickup locations are specified in csv and pointString. More points and an analysis of the hotzone result from a hotter rectangle.

HOTCELL TASK:

The goal of this work is to use spatial statistics to analyze large amounts of data (spark program). The hotness of a cell is calculated using the Getis-Ord statistic of the NYC Taxi Trip dataset (from yellow tripdata 2009-01 point.csv). The following modifications were made to lower the computing power:

1) In terms of latitude and longitude, the cell unit size of 0.01 * 0.01.

2) Every day is divided into time steps, with the first day of the month being step 1.

3) so, the Z-score is calculated.

Z-score:

The hotcell function only accepts one string data type input. The parameter uses the yellow tripdata 2009-01 point.csv dataset's pickup locations.

$$G_i^* = \frac{\sum_{j=1}^{n} w_{i,j}x_j - \bar{X}\sum_{j=1}^{n} w_{i,j}}{S\sqrt{\frac{\left[n\sum_{j=1}^{n} w_{i,j}^2 - \left(\sum_{j=1}^{n} w_{i,j}\right)^2\right]}{n-1}}}$$

where x_j is the attribute value for cell j, $w_{i,j}$ is the spatial weight between cell i and j, n is equal to the total number of cells, and:

$$\bar{X} = \frac{\sum_{j=1}^{n} x_j}{n}$$

$$S = \sqrt{\frac{\sum_{j=1}^{n} x_j^2}{n} - (\bar{X})^2}$$

```
//To calculate zscore
def CalculateZScore(mean: Double, stddev: Double, numOfNb: Int, sigma: Int, numCells: Int): Double =
{
  val numerator = sigma-(mean*numOfNb)
  val denominator = stddev*Math.sqrt((numCells*numOfNb - numOfNb*numOfNb)/(numCells-1))

  return numerator/denominator
}
```

Evaluating and Results:

Create a jar executable file to test the project's code using the datasets in the resources folder.

Building a tool for Scala and JAVA projects requires running the sbt-assembly and the CSE511-Hotspot-Analysis-Template-assembly-0.1.0.

Run a jar executable file to submit a spark program using spark-submit. To receive outputs from hotzone and hotcell, respectively, use the command below.

The output obtained and the results from the project's test cases exactly match each other.