

# **CS585: Big Data Management**

## **Project 3**

**Total Points: 50**

**Release Date: 06/28/2022**

**Due Date: 07/08/2022 (11:59PM)**

## **Short Description**

In this project, you will write Scala/SparkSQL code for executing jobs over Spark infrastructure.

## **Problem 1 SparkSQL (Transaction Data Processing) [25 points]**

Use the Transaction dataset **T** that you created in Project 1 and create a Spark workflow to do the following. [Use **SparkSQL** & **DataFrames** to write this workflow.]

Start with T as a dataframe.

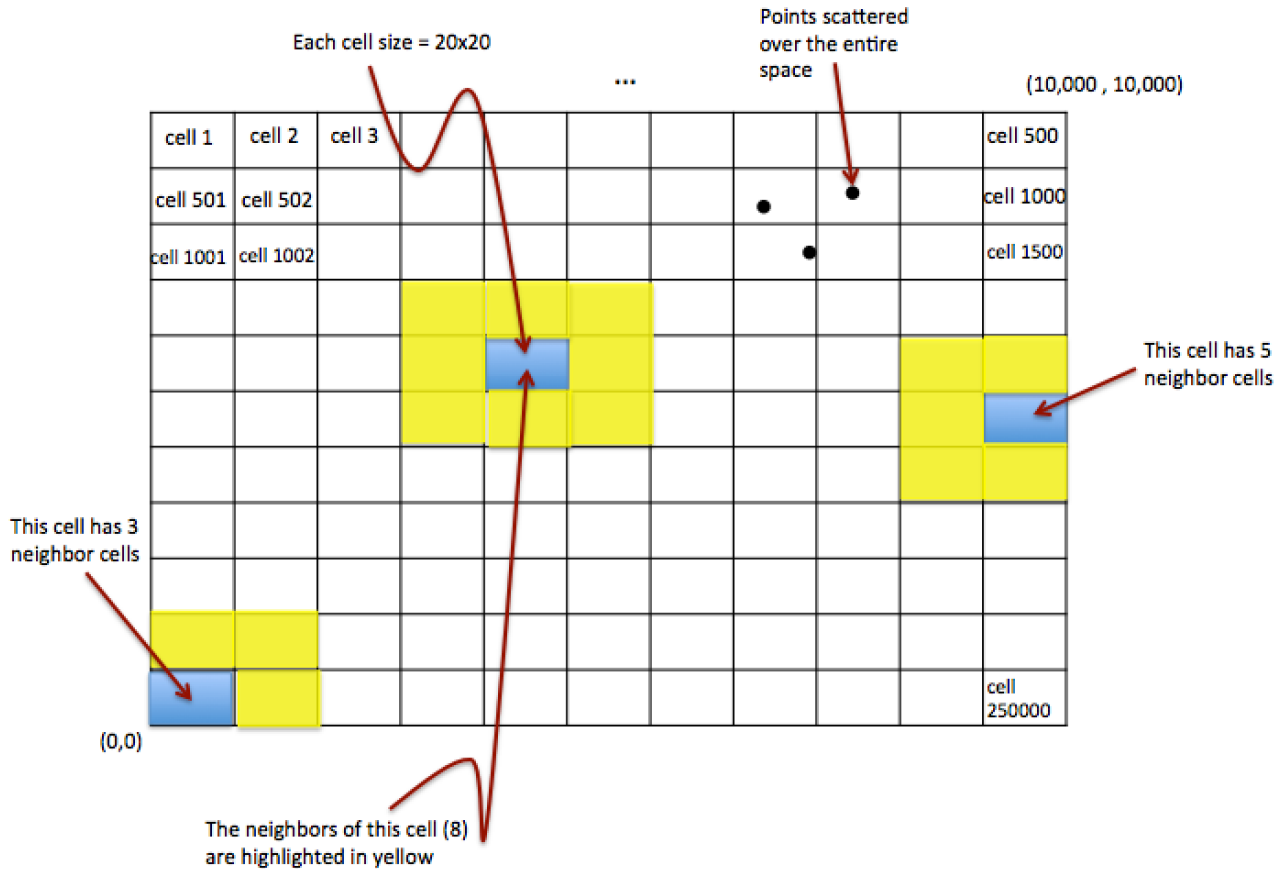
- 1) T1: Filter out (drop) the transactions from **T** whose total amount is less than \$200
- 2) T2: Over T1, group the transactions by the Number of Items it has, and for each group calculate the sum of total amounts, the average of total amounts, the min and the max of the total amounts.
- 3) Report back T2 to the client side
- 4) T3: Over T1, group the transactions by customer ID, and for each group report the customer ID, and the transactions' count.
- 5) T4: Filter out (drop) the transactions from **T** whose total amount is less than \$600
- 6) T5: Over T4, group the transactions by customer ID, and for each group report the customer ID, and the transactions' count.
- 7) T6: Select the customer IDs whose  $T5.count * 5 < T3.count$
- 8) Report back T6 to the client side

## **Problem 2 Spark-RDDs (Grid Cells of High Relative-Density Index) [25 points]**

### **Overview:**

Assume a two-dimensional space that extends from 1...10,000 in each dimension as shown in Figure 1. There are points scattered all around the space. The space is divided into pre-defined grid-cells, each of size 20x20. That is, there is  $500 \times 500 = 250,000$  grid cell in the space. Each cell has a unique ID as indicated in the Figure. Given an ID of a grid cell, you can calculate the row and the column it belongs to using a simple mathematical equation.

//It is up to you to start the Cell Id in any way you want, e.g., you may assign the cell next to (0,0) Id = 1, and so on. Just make it clear in the Readme file.



**Neighbor Definition:** For a given grid cell  $X$ ,  $N(X)$  is the set of all neighbor cells of  $X$ , which are the cells with which  $X$  has a common edge or corner. The Figure illustrates different examples of neighbors. Each non-boundary grid cell has 8 neighbors. However, boundary cells will have less number of neighbors (See the figure). Since the grid cell size is fixed, the IDs of the neighbor cells of a given cell can be computed using a formula (mathematical equations) in a short procedure.

Example:  $N(\text{Cell } 1) = \{\text{Cell } 2, \text{Cell } 501, \text{Cell } 502\}$

$N(\text{Cell } 1002) = \{\text{Cell } 501, \text{Cell } 502, \text{Cell } 503, \text{Cell } 1001, \text{Cell } 1003, \text{Cell } 1501, \text{Cell } 1502, \text{Cell } 1503\}$

**Relative-Density Index:** For a given grid cell  $X$ ,  $I(X)$  is a decimal number that indicates the relative density of cell  $X$  compared to its neighbors. It is calculated as follows.

$$I(X) = X.\text{count} / \text{Average}(Y_1.\text{count}, Y_2.\text{count}, \dots, Y_n.\text{count})$$

Where “ $X.\text{count}$ ” means the count of points inside grid cell  $X$ , and  $\{Y_1, Y_2, \dots, Y_n\}$  are the neighbors of  $X$ . That is  $N(X) = \{Y_1, Y_2, \dots, Y_n\}$

### **Step 1 (Create the Datasets)[5 Points] //You can re-use your code from Project 2**

- Your task in this step is to create one dataset  $P$  (set of 2D points). Assume the space extends from 1...10,000 in the both the  $X$  and  $Y$  axis. Each line in the file should contain one point in the format  $(a, b)$ , where  $a$  is the value in the  $X$  axis, and  $b$  is the value in the  $Y$  axis.
- Scale the dataset to be at least 100MB.

- Choose the appropriate random function (of your choice) to create the points.
- Upload and store the file into HDFS

**Step 2 (Report the TOP 50 grid cells w.r.t Relative-Density Index)[15 Points]**

In this step, you will write Scala or Java code (it is your choice) to manipulate the file and report the top 50 grid cells (the grid cell IDs not the points inside) that have the highest **I** index. Write the workflow that reports the cell IDs along with their relative-density index.

Your code should be fully parallelizable (distributed) and scalable.

**Step 3 (Report the Neighbors of the TOP 50 grid)[5 Points]**

Continue over the results from Step 2, and for each of the reported top 50 grid cells, report the IDs and the relative-density indexes of its neighbor cells.

**What to Submit (for each student)**

- You will submit a single zip file containing all problems and source code
- The zip file should also include a “Readme.pdf” file. In this file include:
  - Any assumptions that you have made
  - Create a table as shown below and fill it up

Question	Status (Select one) Fully Working/ Partially Working/ Not Working	Comment
Q1		
Q2		

- Any comments you would like to provide regarding your code.

**How to Submit**

- Use the Canvas system to submit your files.