CSE 511-DATA PROCESSING AT SCALE
PROJECT REPORT
HOT SPOT ANALYSIS

REFLECTION:

This project focusses on extracting crucial data from the provided database by performing many spatial queries using Scala and Apache Spark. The technology requirements are met as specified in the installation document. The database has the information for hot-zones and hot-cells of New-York taxi trip datasets. Hot-cells corresponds to the pickup points and only pickup locations are considered. We are asked to complete two hotspot analysis tasks.

Task 1: Hot zone Analysis Reflection and Implementation

We are given with the rectangle dataset where each rectangle corresponds to a zone and a point dataset where each point corresponds to a pickup location. The range join operation on these datasets are already done as part of the template. We need to find how many points(pickup locations) are located in each rectangle(zone). This specifies the hotness of each zone. If we have more pickup locations, then the zone becomes the hottest zone. We need to find the hotness of all zones and the results must be sorted according to the rectangle string.

To achieve that, as a first step edit on HotzoneUtils is done. The function ST_Contains takes the co-ordinates of the rectangle (Top-left,Bottom-right) and the point as input and expect a Boolean output. This function helps in calculating the hotness of zones. We should take individual latitudinal co-ordinates of the rectangle and find the minimum and maximum (Let us take it as rect_lat_min and rect_lat_max). Same has to be done for longitudinal co-ordinates(Let us take it as rect_long_min and rect_long_max). We need to check whether the latitude of point is lesser than or equal to rect_lat_max and greater than or equal to rect_lat_min. And also, check whether the longitude of the point is lesser than or equal to rect_long_max and greater than or equal to rect_long_min. If all these conditions satisfy, only then this function returns true or else it returns false.

Now this function ST_Contains becomes the child function and it has to be called in HotzoneAnalysis file. I wrote a SQL query for counting the points, grouping by rectangle and arranging in ascending order alone because, while joining the data, we take only the values which satisfy this 'ST_Contains' function. The answer from this query is returned as result.

Task 2: Hot cell Analysis Reflection and Implementation

We need to calculate the Getis-Ord Statistics from the input taxi trip dataset. This determines how significant a particular location is. The topic of this task is from ACM SIGSPATIAL GISCUP 2016. Problem Definition page This page specifies the formula for Getis-Ord and explains the variables involved in it. The result of this task gives the hotness of the cell. A cell is represented in time and space. Each cell has a step size of 0.01 times the latitude and also has a step size of 0.01 times the longitude. The z axis would be the date. So co-ordinates x,y and z are represented as (latitude/0.01, longitude/0.01, date) The x,y,z co-ordinates are calculated as part of the template. yellow_trip_sample_100000.csv has 100000 data to reduce computation power.

For this task, I changed HotcellUtils and HotcellAnalysis in parallel. In HotcellAnalysis file, we are only considering values which fall in between the given X, Y and Z minimum and maximum range, and leave out data which falls apart or below this range. A view is created with these data and it is named as xyz_details. This becomes our primary data from now on.

To start with the formula, I proceeded on Mean and Standard Deviation calculation. For that, I started counting the number of pickups on a particular day and at a particular location(attribute value for the cell). I made grouping and sorting of data by z,y,x. As we need to finally return the values in descending order, I maintained the same ordering everywhere right from the start. This dataframe is stored with the name 'xyz_count_details'. As a next step we need to calculate the summation of attribute values for all cells. Along with that, I calculated the summation of square of attribute values for all cells. These values are extracted as a single query and stored in a view named 'xyz_sum_details'. The mentioned view names is just to differentiate one view from the other. We then take the first value from the xyz_sum_details and divide it by the given numCells value and cast it to double. This becomes our mean value. For standard deviation, we take the square of the mean and subtract it from the second value of the xyz_sum_details divided by the given numCells and take square root of the whole value.

Next step is to count the neighbors for each cell. Function for that is written in HotcellUtils file. It takes minimum and maximum values of x,y and z co-ordinates and the co-ordinates of the current cell to return the number of neighbors. It has four conditions as follows.

- If the cell is the centre most cell of a 3*3 spatial time cube, then it has 26 neighbors.
- If the cell has minimum or maximum value for all of the x,y,z coordinates, then it has 7 neighbors.
- If the cell has minimum or maximum value for any two of the x,y,z coordinates, then it has 11 neighbors.
- If the cell has minimum or maximum value for any one of the x,y,z coordinates, then it has 17 neighbors.

In the HotcellAnalysis file, after calling the neighbor_count function, the values are stored in neighbor_details view.

Final step is to calculate the z-score. For that we input the co-ordinates of the cell, neighbor cell count, hotness of the cell, mean and standard deviation of the cell and number of cells.
To calculate the numerator, we subtract the product of mean and neighbor cell count from the hotness of the cell. For denominator, we take the square root of temp_value and multiply it with standard deviation. temp_value can be calculated by squaring the neighbor cell count and subtracting it from the product of number of cells and the neighbor cell count. We divide the numerator and the denominator for final z-score value.

In HotcellAnalysis, we call the HotcellUtils to get the z-score value along with the co-ords, and sort it by descending order of Z-score value and save it as z-score_df. From z-score_df we take only the co-ordinates and return it as result.

Entrance.scala file has the function calls for both hot zone analysis and hot cell analysis.

Once the code logic is completed, in terminal, we need to give 'sbt assembly' command after moving to our project root folder. I installed it using brew. I tried installing intellij IDEA for MAC, but due to the version mismatch, I was not able to proceed further on that. So, after we give 'sbt assembly' command, we can find the compilation errors. Once those errors are sorted, I gave the following command, for output production. "spark-submit target/scala-2.11/CSE512-Hotspot-Analysis-Template-assembly-0.1.0.jar test/output hotzoneanalysis src/resources/point_hotzone.csv src/resources/zone-hotzone.csv hotcellanalysis src/resources/yellow_trip_sample_100000.csv". Then we can find the output generated in the newly created test folder.

LESSONS LEARNED:

- Went through the basics of the scala programming, syntax for fetching values and performing basic operations.

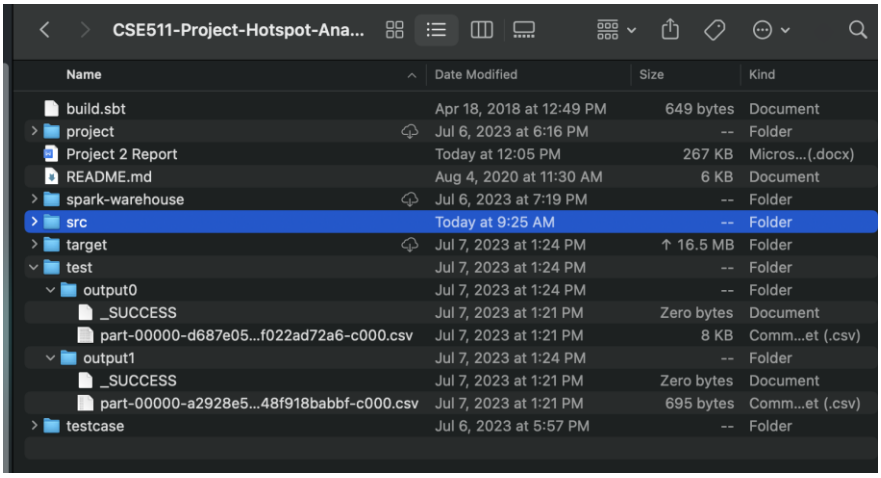
- Understood how to fetch the computed values by a function call from another file containing the function.
- Learnt the sparkSQL syntax for fetching values from the views.
- Learnt the installation of Apache spark and got hands on training on executing spark programs.
- Understood how real time spatial data are processed and used.
- Used .toDouble to cast the values for appropriate results and avoid type casting errors.

RESULTS:

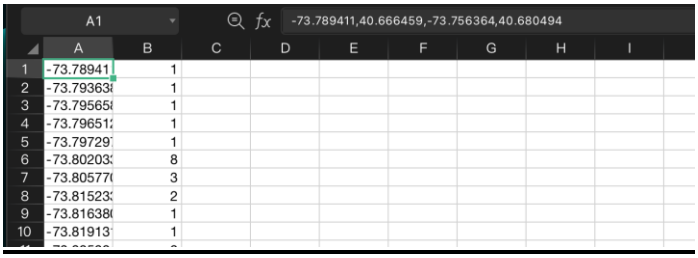
```
(base) sailendrigr@Sailendris-Air CSE511-Project-Hotspot-Analysis % sbt clean assembly
[info] welcome to sbt 1.9.1 (Homebrew Java 20.0.1)
[info] loading project definition from /Users/sailendrigr/Desktop/College/CSE-511-DP/Projects/CSE511-Project-Hotspot-Analysis/project/project
[info] loading settings for project cse511-project-hotspot-analysis-build from plugins.sbt ...
[info] loading project definition from /Users/sailendrigr/Desktop/College/CSE-511-DP/Projects/CSE511-Project-Hotspot-Analysis/project
[info] loading settings for project root from build.sbt ...
[info] set current project to CSE512-Hotspot-Analysis-Template (in build file:/Users/sailendrigr/Desktop/College/CSE-511-DP/Projects/CSE511-Project-Ho
lysis/)
[success] Total time: 0 s, completed Jul 7, 2023, 1:14:29 PM
[info] compiling 5 Scala sources to /Users/sailendrigr/Desktop/College/CSE-511-DP/Projects/CSE511-Project-Hotspot-Analysis/target/scala-2.11/classes .
[warn] there was one deprecation warning; re-run with -deprecation for details
[warn] one warning found
[info] Including: scala-library-2.11.11.jar
[info] Checking every *.class/*.jar file's SHA-1.
[info] Merging files...
[warn] Merging 'META-INF/MANIFEST.MF' with strategy 'discard'
[warn] Strategy 'discard' was applied to a file
[info] SHA-1: 9151a47a9b0be47328253c10303360f15eaeb1dd
[warn] Ignored unknown package option FixedTimestamp(Some(1262304000000))
[success] Total time: 5 s, completed Jul 7, 2023, 1:14:35 PM
```

```
(base) sailendrigr@Sailendris-Air CSE511-Project-Hotspot-Analysis % spark-submit target/scala-2.11/CSE512-Hotspot-Analysis-Template-assembly-0.1.0.jar test/out
put hotzoneanalysis src/resources/point_hotzone.csv src/resources/zone-hotzone.csv hotcellanalysis src/resources/yellow_trip_sample_100000.csv
23/07/07 13:14:37 WARN NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
23/07/07 13:14:38 INFO SparkContext: Running Spark version 3.4.1
23/07/07 13:14:38 INFO ResourceUtils: ==============================================================
23/07/07 13:14:38 INFO ResourceUtils: No custom resources configured for spark.driver.
23/07/07 13:14:38 INFO ResourceUtils: ==============================================================
23/07/07 13:14:38 INFO SparkContext: Submitted application: CSE512-HotspotAnalysis-Sailendri
23/07/07 13:14:38 INFO ResourceProfile: Default ResourceProfile created, executor resources: Map(cores -> name: cores, amount: 1, script: , vendor: , memory ->
 name: memory, amount: 1024, script: , vendor: , offHeap -> name: offHeap, amount: 0, script: , vendor: ), task resources: Map(cpus -> name: cpus, amount: 1.0)
23/07/07 13:14:38 INFO ResourceProfile: Limiting resource is cpu
23/07/07 13:14:38 INFO ResourceProfileManager: Added ResourceProfile id: 0
23/07/07 13:14:38 INFO SecurityManager: Changing view acls to: sailendrigr
23/07/07 13:14:38 INFO SecurityManager: Changing modify acls to: sailendrigr
23/07/07 13:14:38 INFO SecurityManager: Changing view acls groups to:
23/07/07 13:14:38 INFO SecurityManager: Changing modify acls groups to:
23/07/07 13:14:38 INFO SecurityManager: SecurityManager: authentication disabled; ui acls disabled; users with view permissions: sailendrigr; groups with view
permissions: EMPTY; users with modify permissions: sailendrigr; groups with modify permissions: EMPTY
```

…..



Hotzone Results:



Hotcell Results:

	A	B	C	D	E
1	-7399	4075	15		
2	-7399	4075	22		
3	-7399	4075	14		
4	-7399	4075	29		
5	-7398	4075	15		
6	-7399	4075	16		
7	-7399	4075	21		
8	-7399	4075	28		