Assignment Specification

FIT 5148 - Distributed Databases and Big Data

Due: Friday May 24, 2019, 11:55 PM (Local Campus Time)
Worth: 20% of the final marks

Background

StopFire is a campaign started by Monash University to predict and stop the fire in Victorian cities. They have employed sensors in different cities of Victoria and have collected a large amount of data. The data is so big that their techniques have failed to provide the results on time to predict fire. They have hired us as *the data analyst* to migrate their data to the NoSQL database (MongoDB). They want us to analyse their data and provide them with results. In addition, they want us to build an application, a complete setup from streaming to storing and analyzing the data for them using Apache Kafka, Apache Spark Streaming and MongoDB.

What you are provided with

- Five datasets:
    - hotspot_historic.csv
    - climate_historic.csv
    - hotspot_AQUA_streaming.csv
    - hotspot_TERRA_streaming.csv
    - climate_streaming.csv
- These files are available in Moodle in the Assessment section in Assignment Data folder.

Information on Dataset

Climate data is recorded on a daily basis whereas Fire data is recorded based on the occurrence of a fire on a particular day. Therefore, for one climate data, there can be zero or many fire data.
The data is NOT row per weather station basis. You can simply think of it as, Station 1 was reporting data for X number of days and then Station 2 started reporting data because Station 1 was shut down for instance.

Total precipitation (rain and/or melted snow) reported during the day in inches and hundredths; will usually not end with the midnight observation --i.e., may include latter part of the previous day.

.00 indicates no measurable precipitation (includes a trace). Missing = 99.99 (For metric version, units = millimeters to tenths & missing = 999.9.)

Note: Many stations do not report '0' on days with no precipitation --therefore, '99.99' will often appear on these days. Also, for example, a station may only report a 6-hour amount for the period during which rain fell. See Flag field information below the source of data.

- A = 1 report of 6-hour precipitation amount.
- B = Summation of 2 reports of 6-hour precipitation amount.
- C = Summation of 3 reports of 6-hour precipitation amount.
- D = Summation of 4 reports of 6-hour precipitation amount.
- E = 1 report of 12-hour precipitation amount.
- F = Summation of 2 reports of 12-hour precipitation amount.
- G = 1 report of 24-hour precipitation amount.
- H = Station reported '0' as the amount for the day (eg, from 6-hour reports), but also reported at least one the occurrence of precipitation in hourly observations --this could indicate a trace occurred but should be considered as incomplete data for the day.
- I = Station did not report any precipitation data for the day and did not report any occurrences of precipitation in its hourly observations --it's still possible that precipitation occurred but was not reported.

Programming Tasks

Task A. MongoDB Data Model (10 Marks)

1. Based on the two data sets provided i.e. *hotspot_historic.csv* and *climate_historic.csv*, design a suitable data model to support the efficient querying of the two data sets in MongoDB. Justify your data model design.
   The output of this task should be
   - *An example of the data model*
   - *The justification for choosing that data model*

Task B. Querying MongoDB using PyMongo (30 Marks)

1. Write a python program that will read the data from *hotspot_historic.csv* and *climate_historic.csv* and load them to the new database **(e.g. fit5148_assignment_db)**. The collection(s) in **fit5148_assignment_db** will be

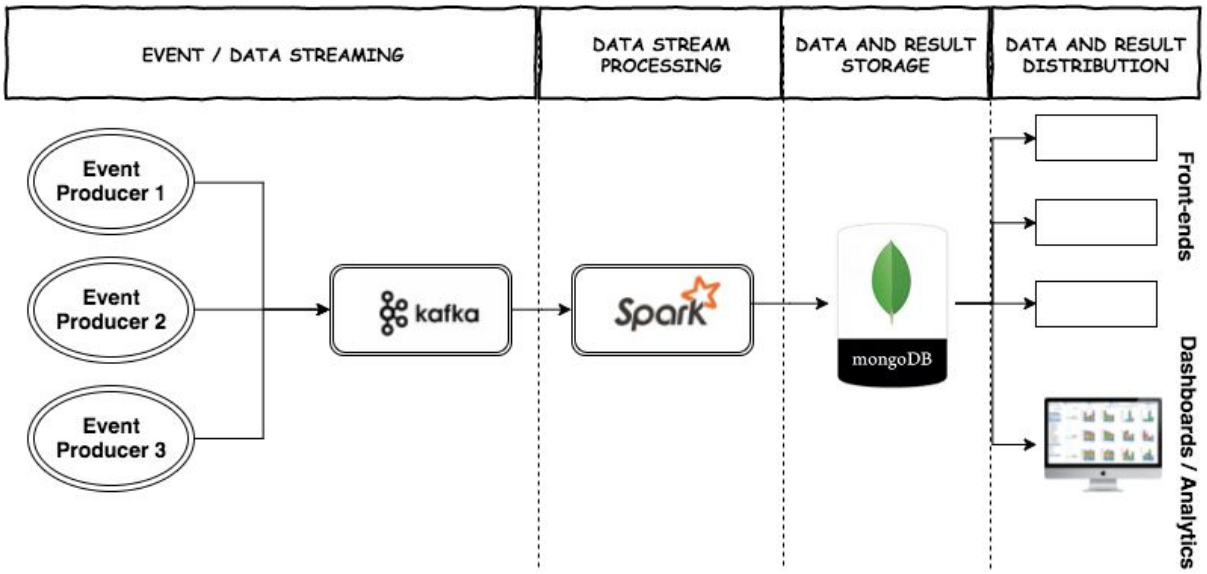
based on the document model you have designed in Task A1. *Please DO NOT use mongo aggregation query to do this task.*

2. Write queries to answer the following tasks on **fit5148_assignment_db** and corresponding collection(s). You need to write the queries as a python program using the `pymongo` library in Jupyter Notebook.
    a. Find climate data on *10th December 2017*.
    b. Find the *latitude*, *longitude, surface temperature* (°C), and *confidence* when the surface temperature (°C) was between *65 °C* and *100 °C*.
    c. Find *date*, *surface temperature (°C), air temperature (°C), relative humidity* and *max wind speed on* 15th and 16th of December 2017.
    d. Find *datetime, air temperature (°C), surface temperature (°C)* and *confidence* when the *confidence* is between 80 and 100.
    e. Find the top 10 records with the highest *surface temperature (°C).*
    f. Find the number of fire in each day. You are required to only display *the total number of fire* and *the date* in the output.
    g. Find the *average surface temperature (°C)* for each day. You are required to only display *average surface temperature (°C)* and *the date* in the output.

Combine all your answers for Task A and Task B into a single Jupyter Notebook file called **Assignment_TaskA_B.ipynb**.

Task C. Processing Data Stream (60 Marks)

In this task, we will implement multiple Apache Kafka producers to simulate the real-time streaming of the data which will be processed by Apache Spark Streaming client and then inserted into MongoDB. The overall system architecture you will be developing is shown in the figure below.



1. Simulating real-time data using Apache Kafka Producers.
    a. **Event Producer 1:** Write a python program that loads all the data from *climate_streaming.csv* and randomly feed the data to the stream every 5 seconds. You will need to append additional information such as sender_id and created_time. Save the file as **Assignment_TaskC_Producer1.ipynb.**
    b. **Event Producer 2:** Write a python program that loads all the data from *hotspot_AQUA_streaming.csv* and randomly feed the data to the stream every 10 - 30 seconds. AQUA is the satellite from NASA that reports latitude, longitude, confidence and surface temperature of a location. You will need to append additional information such as sender_id and created_time. Save the file as **Assignment_TaskC_Producer2.ipynb.**
    c. **Event Producer 3:** Write a python program that loads all the data from *hotspot_TERRA_streaming.csv* and randomly feed the data to the stream every 10 - 30 seconds. TERRA is another satellite from NASA that reports latitude, longitude, confidence and surface temperature of a location. You will need to append additional information such as sender_id and created_time. Save the file as **Assignment_TaskC_Producer3.ipynb.**
2. Stream Processing using Apache Spark Streaming.
    a. **Streaming Application:** Write a streaming application in Apache Spark Streaming which has a local streaming context with two execution threads and a batch interval of 10 seconds. The streaming application will receive streaming data from all three producers. If the streaming application has data from all or at least two of the producers, do the processing as follows:
        - Join the streams based on the location (i,e, latitude and longitude) and create the data model developed in Task A.
        - Find if two locations are close to each other or not. You can do this by implementing the geo-hashing algorithm or find the library that does the job for you. Use precision 5. The precision determines the number of characters in the Geohash.

- If we receive the data from two different satellites AQUA and TERRA for the same location, then average the 'surface temperature' and 'confidence'.
- If the streaming application has the data from only one producer (Producer 1), it implies that there was no fire at that time and we can store the climate data into MongoDB straight away.

Save the file as **Assignment_TaskC_Streaming_Application.ipynb.**

3. Data Visualisation using MatPlotLib
    a. **Streaming data visualization:**
        i. For the incoming climate data plot the line graph of air temperature against arrival time. You need to label some interesting points such as maximum and minimum values.
    b. **Static data visualization:** Write python programs using pymongo to get the data from the MongoDB collection(s) created in Task C.2 and perform the following visualizations.
        i. Records with the top 10 number of fires. Plot a bar chart with time as the x-axis and number of fires as the y-axis.
        ii. Plot fire locations in the map with air temperature, surface temperature, relative humidity and confidence. See the example below.

Save the file as **Assignment_TaskC_Data_Visualisation.ipynb.**

Google Drive

One of the team members should create a Google drive space as a collaboration space for the team (*for example the folder could be named Lab_02_David_Prajwol*). Each of the students should show us a clear progression of your project throughout the development process. Label each version clearly with the last saved date, e.g. TaskA-v.1-20-Mar.ipynb. **Please share the google drive folder with your tutor.** Your tutor will keep track of the activity on the google drive. The activity (contribution) counts

towards your assignment marks. We will not assess your Moodle submission if there is no evidence of activity in your team Google drive.

Submission

Your team should submit their final version of the assignment solution online via Moodle; You must submit the following:

- A zip file of your Assignment folder named based on your authcate name (e.g. psan002_mar005). This should contain:
    - **Assignment_TaskA_B.ipynb**
    - **Assignment_TaskC_Producer1.ipynb**
    - **Assignment_TaskC_Producer2.ipynb**
    - **Assignment_TaskC_Producer3.ipynb**
    - **Assignment_TaskC_Streaming_Application.ipynb**
    - **Assignment_TaskC_Data_Visualisation.ipynb**
    - **filled contribution declaration form**.

  This should be a ZIP file and *NOT any other kind of compressed folder (e.g. .rar, .7zip, .tar).*
- The same assignment submission should be uploaded by EACH of the team members and must be finalised by **Week 11 Friday May 24, 2019, 11:55 PM (Local Campus Time)**. *Please note: your entire team needs to accept the assignment submission statement individually on Moodle.*
- Your assignment will be accessed based on the contents of the Assignment folder you have submitted via Moodle. You should ensure that the copy of the assignment submitted to Moodle is the final version that exists in your Google Drive. We will use the same FIT5148 server as provided to you when marking your assignments.

Assignment Code Interview

During Week 12 tutorial your tutor will spend a few minutes interviewing each team member to individually gauge the student's personal understanding of your Assignment code. The purpose of this is to ensure that each member of the team has contributed to the assignment and understands the code submitted by the team in their name. Tutors will basically focus on three questions

- One question from the part you have completed
- One question from the part you teammate has completed
- One random question (if required)

Please note: Tutorial and interview sessions will be running in parallel.

The final grade of your assignment will be provided as an individual grade. Individual's grade may vary depending on the **contribution level** evidence by activities in the Google drive, **tasks distribution** (if any) and the result of **interview**.

Resources

FireData: https://sentinel.ga.gov.au/#/
WeatherData: http://www.bom.gov.au/vic/?ref=hdr
Edward, Shakuntala Gupta, and Navin Sabharwal. *Practical MongoDB: Architecting, Developing, and Administering MongoDB*. Apress, 2015.
https://docs.mongodb.com/tutorials/

Other Information

How to collaborate effectively for this assignment

To effectively work in a team for this assignment, we suggest the following process be adopted by the group.

1. Establish the first meeting to plan for the project. The plan includes:
    a. an agreed regular time for a meeting,
    b. an agreed task level to complete,
    c. an agreed breakdown of the tasks and allocation
    d. agreed deadlines of the task completion by the member.

    Have all of these in writing and place the document in the Google drive. Every member is accountable for the agreed decision made on the document.

    Place an entry on an online calendar system, e.g. Google calendar on the agreed meeting dates/time and tasks deadline.

    The meeting can be conducted face-to-face or using conferencing technology such as Zoom. Monash provides access to Zoom for all students. https://monash.zoom.us/.

2. Every member needs to be able to understand and be able to write the code of all the submitted tasks. If you decide the break the task and requires an individual to complete a certain task, have a plan on:
    a. Overview of the overall program/code design.
    b. The data model that will be used by the program,
    c. Possible reusable components of the code. If there is, agree on this and write this code first so every member can use it for different tasks.
    d. How to manage the versioning of codes or components in your project.

3. Create a separate file for each task, eg taskA.1_v1_20-Mar.ipnyb. Attempt each task in a separate workbook. Upload the individual task regularly to Google drive. Follow the agreed version naming to avoid confusion. Your tutor will check this during marking to see the progression of your project.

Complete the task each week after you complete the tutorial exercises for the topic. Don't leave it too close to the submission deadline. If you can't meet the deadline agreed in point 1 for the task allocated to you, discuss it with your team member as soon as you realise the problem. Don't leave it until the deadline to tell your partner that you can't complete the task. If you can't do the task, seek help early.

4. Use the regular meeting to:
    a. Check on the project progress.
    b. Consolidate the completed individual tasks to the project. This includes explaining your code to your team member.
    c. Resolve any possible confusion/problem.

    Once the meeting time is agreed as in point 1, don't change it unless it is really unavoidable. Be punctual in attending the meeting. It is disrespectful to your team member if you come late or inform your partner you can't make it in the last minute.

5. Once all the tasks are completed, combine all the codes into the notebook provided for submission. Place this final version in Google drive and name it clearly so all members can find it for the final submission in Moodle.

Having a plan, being organised and be committed to the agreed plan will increase your chance to work well with your partner.

Where to get help

You can ask questions about the assignment on the Assignment Discussion Forum on the unit's Moodle page. This is the preferred venue for assignment clarification-type questions. You should check this forum (and the News forum) regularly, as the responses of the teaching staff are "official" and can constitute amendments or additions to the assignment specification. Also, you can visit the consultation sessions if the problems and the confusions are still not solved.

Plagiarism and collusion

Plagiarism and collusion are serious academic offences at Monash University. Students must not share their team's work with any student outside of their team. Students should consult the policy linked below for more information.

https://www.monash.edu/students/academic/policies/academic-integrity

See also the video linked on the Moodle page under the Assignment block.

Students involved in collusion or plagiarism will be subject to disciplinary penalties, which can include:

- The work not being assessed
- A zero grade for the unit

- Suspension from the University
- Exclusion from the University

Late submissions

Submission must be made by the due date otherwise penalties will be enforced. You must negotiate any extensions formally with your campus unit lecturer via the in-semester special consideration process:

http://www.monash.edu.au/exams/special-consideration.html

There is a **5% penalty per day including weekends** for the late submission.

.