Assignment Specifications

FIT3182-FIT5148 – Big Data Management & Processing

Due: ~~Friday, June 5, 2020, 11:55 PM (Local Campus Time)~~
Friday, June 12, 2020, 11:55 PM (Local Campus Time)
Worth: 20% of the overall unit assessment 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 the latter part of the previous day.

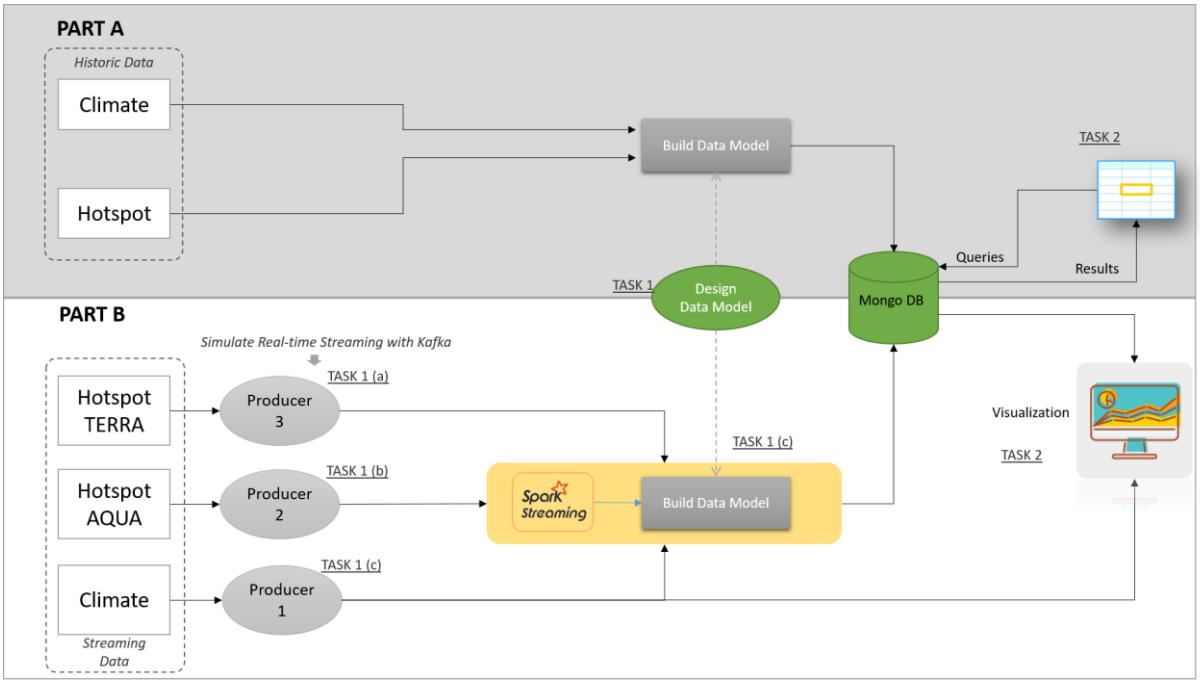
.00 indicates no measurable precipitation (includes a trace). **Missing = 99.99** (*For metric version, units = millimeters to tenths & missing = 999.9.*)

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 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.

Architecture

The overall architecture of the assignment setup is represented by the following figure.



Part A of the assignment consists of reading data from csv files, building a model, saving to MongoDB and running various queries against the database. **Part B** of the assignment consists of simulating real-time streaming of climate and hotspot data and processing it with Apache

Kafka and Spark Streaming. Both approaches should use the same data model and thus the same database.

Part A (40%)

Task 1. MongoDB Data Model

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 2. Querying MongoDB using PyMongo

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 or fit3182_assignment_db)**. The collection(s) in **fit5148_assignment_db or fit3182_assignment_db** will be based on the document model you have designed in Task A1.
   - Please use `csv` library to read the files.
   - *Please DO NOT use mongo aggregation query to do this task.*

2. Write queries to answer the following tasks on **fit5148_assignment_db or fit3182_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 2018*.
   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 2018.
   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.

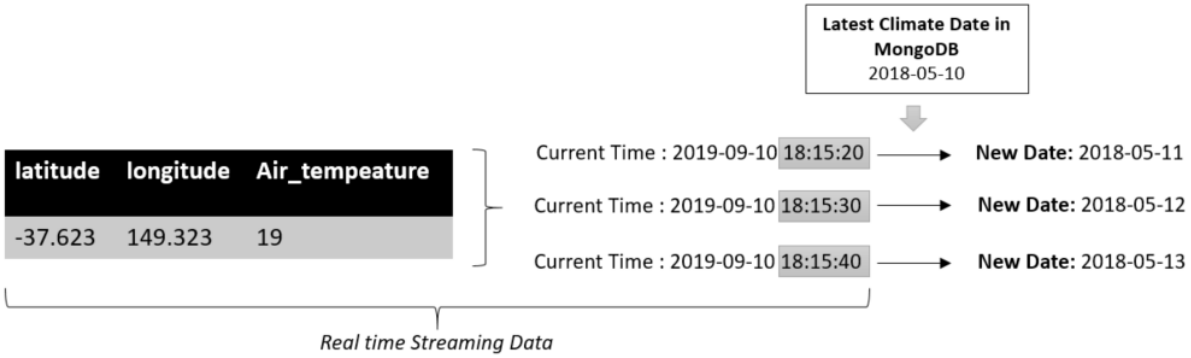
Combine all your answers for Task 1 and Task 2 into a single Jupyter Notebook file called **Assignment_PartA.ipynb**.

Part B (60%)

Task 1. Processing Data Stream

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.

Important : *In this task, you are required to use the same data model from Part A. To make the streaming data consistent for the model, you will need to make some changes to the streaming data before building the model or inserting to mongodb. The figure below gives an example of the date modification.*



First, find the **latest date** in the climate data that was inserted into the MongoDB database you created in Part A, Task 2. *Treat every 10 seconds data as 1 day*. Therefore, you will have to create a new date for each 10 seconds data (e.g. new_date = latest date in climate data + 1 day). For the next batch of 10 seconds, we will have a new_date = new_date + 1 day and so on.

Now, you are required to implement three Kafka-producers to simulate the real-time streaming of data.

a. **Event Producer 1:** Write a python program that loads all the data from *climate_streaming.csv* and randomly (with replacement) feed the data to the stream every 10 seconds. You will need to append additional information such as producer information to identify the producer and created date. Save the file as **Assignment_PartB_Producer1.ipynb.**

b. **Event Producer 2:** Write a python program that loads all the data from *hotspot_AQUA_streaming.csv* and randomly (with replacement) feed the data to the stream every 2 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 producer information to identify the producer and created date & time. Save the file as **Assignment_PartB_Producer2.ipynb.**

c. **Event Producer 3:** Write a python program that loads all the data from *hotspot_TERRA_streaming.csv* and randomly (with replacement) feeds the data to the stream every 2 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 producer information to identify the producer and created date & time. Save the file as **Assignment_PartB_Producer3.ipynb.**

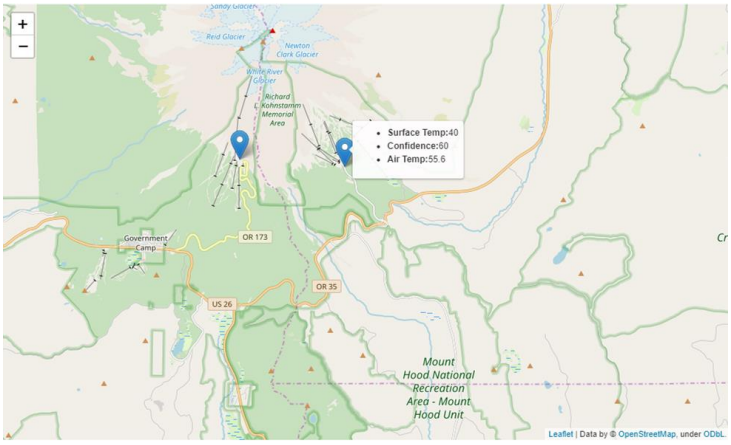
d. **Streaming Application:** Write a streaming application in Apache Spark Streaming which has a local streaming context with two execution threads and a batch interval of 10 seconds. The streaming application will receive streaming data from all three producers. If the streaming application has data from all or at least two of the producers, do the processing as follows:
   - Group the streams based on the location (i,e, latitude and longitude) and create the data model developed in Part A.
   - You can find if two locations are close to each other or not by implementing the geo-hashing algorithm or find a library that does the job for you. The precision number in the algorithm determines the number of characters in the Geohash. Please use precision 3. If the climate data and hotspot data are not close to each other we can ignore the hotspot data and just store the climate data.
   - 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.
   - If we receive the data from two different satellites AQUA and TERRA for the same location (to determine whether the two locations are the same or not please use geohash with precision 5), then average the 'surface temperature' and 'confidence'.

   Save the file as **Assignment_PartB_Streaming_Application.ipynb.**

Task 2 : Data Visualisation

1. **Streaming data visualization**
   a. 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.

2. **Static data visualization**
   Write python programs using pymongo to get the data from the MongoDB collection(s) created in Part B, Task 1 and perform the following visualizations.
   a. Plot a bar chart to visualize the total number of fire records based on each hour.
   b. In a map visualize *fire locations* as markers. Display detailed information such as *air temperature, surface temperature, relative humidity and confidence* with the marker tooltip. See the example below. You can use Folium for map visualization.



Save the file as **Assignment_PartB_Data_Visualisation.ipynb.**

Assignment Marking

Marking of this assignment is based on the quality of work that you have submitted rather than just quantity. Marking starts from zero and goes up based on the tasks you have successfully completed and it's quality, for example how well the code submitted follows *programming standards, code documentation, presentation of the assignment, readability of the code, organisation of code and so on*. Please find the PEP 8 -- Style Guide for Python Code here for your reference.

Submission

You should submit your 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). This should contain:
    - **Contribution declaration form**
    - **Assignment_PartA.ipynb**
    - **Assignment_PartB_Producer1.ipynb**
    - **Assignment_PartB_Producer2.ipynb**
    - **Assignment_PartB_Producer3.ipynb**
    - **Assignment_PartB_Streaming_Application.ipynb**
    - **Assignment_PartB_Data_Visualisation.ipynb**

  This should be a ZIP file and *NOT any other kind of compressed folder (e.g. .rar, .7zip, .tar).*
- The assignment submission should be uploaded and finalised by Friday June 12th, 2020, 11:55 PM (Local Campus Time).
- Your assignment will be assessed based on the contents of the Assignment folder you have submitted via Moodle. We will use the same FIT servers as provided to you when marking your assignments.

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

Where to get help

You can ask questions about the assignment on the 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 also book an appointment for the consultation sessions if the problems and confusions are still not solved.

Plagiarism and collusion

Plagiarism and collusion are serious academic offences at Monash University. Students must not share their work with any student. 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

Extensions and other individual alterations to the assessment regime will only be considered using the Faculty Special Consideration Policy.

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