Midway Report

Project title

Build a weather data warehouse with MongoDB

Team members (a team of two (or one) students)

Li-Chi Chang(chanl01@pfw.edu)

Detailed description of your project (goal/objectivities and expected tasks)

- ✓ Done
- ✓ Use Google Apps Script and Google Drive to build a clawer with timer.
    - Apps script is a software-embedded programming language based on JavaScript.
    - I embedded a script function in a timer. While the timer is triggered, the function crawls data through web APIs.
- ✓ Crawling the yesterday weather data of several cities every 12 hours.
    - OpenWeatherMap supports historical data API. Get requests with my key can return a 24-hour hourly weather data in json format.
    - In a loop, crawls all cities I want.
    - The cities list:
        - Fort Wayne
        - Taipei
- ✓ Translate and save json data into MongoDB using Python package pymongo.
    - To avoid the duplication, use UTC timestamp as document ID. And before inserting, check the ID is exist or not.
    - The hourly data contains:
        - UTC time
        - Temperature
        - Feels like
        - Pressure
        - Humidity
        - Dew point
        - Clouds
        - Visibility
        - Wind speed
        - Wind degree
        - Weather description

- In Progress
- Use Python to program, summarize and visualize statistic data.
    - Build a server using Django. Show the whole information on it.

Data requirement description for your system

Json format is native supported by node JS and JavaScript. And it should be the format like below:


```
{
  "dt": 1615939200,
  "temp": 294.08,
  "feels_like": 295,
  "pressure": 1012,
  "humidity": 78,
  "dew_point": 290.1,
  "clouds": 20,
  "visibility": 10000,
  "wind_speed": 2.06,
  "wind_deg": 290,
  "weather": [
    {
      "id": 801,
      "main": "Clouds",
      "description": "few clouds",
      "icon": "02d"
    }
  ]
}
```


Conceptual database schema

Weather Data

Dt	Number PK
Lat	Float PK/FK
Lon	Float PK/FK
Temp	Float
Feels_like	Float
Pressure	Int
Dew_point	Float
Clouds	Int
Visibility	Int
Wind_speed	Float
Wind_deg	Int
Description ID	Number FK

City Data

City Name	Varchar
Lat	Float PK
Lon	Float PK

Description

Description ID	Number PK
Main	Varchar
Description	Varchar
Icon	Varchar

Expected functions for your system

- Current weather situation of several cities
- Historical highest temperatures of several cities.
- Line chart of temperatures.

Database technology and development environment expected

- Programming language
  - Python
  - Google Apps Script
- Data Warehousing
  - MongoDB

Data - real data source or synthetic generation description if available

- Openweathermap api document
  - 5 days historical data [One Call API - OpenWeatherMap]
  - Weather condition id code/ icon [Weather Conditions - OpenWeatherMap]

Work schedule

Week	Date	Description
week4	2021/2/2	project proposal. (Done)
week5	2021/2/9	Build a google script crawler. Register API key to get data. (Done)
week6	2021/2/16	Build conceptual database. (Done)
week7	2021/2/23	
week8	2021/3/2	midterm
week9	2021/3/9	Install MongoDB. (Done)
week10	2021/3/16	midway report. (Done)
week11	2021/3/23	Import all data into MongoDB. (Done)
week12	2021/3/30	Build warehouse functions. (In progress)
week13	2021/4/6	
week14	2021/4/13	Build server to visualize all information.
week15	2021/4/20	
week16	2021/4/27	Project Presentation and Demo
week17	2021/5/4	Final report/final term

Appendix

- Openweathermap api document
    - Free account quotas Pricing - OpenWeatherMap
    - 5 days historical data One Call API - OpenWeatherMap
    - Weather condition id code/ icon Weather Conditions - OpenWeatherMap
- Google App Script document
    - Free account quotas Quotas | Apps Script | Google Developers
    - Code document Overview of Google Apps Script | Google Developers
- MongoDB document
    - Installation MongoDB Community Download | MongoDB
    - APIs for Python and Node.js The MongoDB 4.4 Manual — MongoDB Manual