THE UNIVERSITY OF SYDNEY

School of Computer Science
Dr. Ying Zhou

COMP5338: Advanced Data Models

Sem. 2/2024

MongoDB Basic Queries

07.08.2024

1 Introduction

This assignment tests your skills in writing simple MongoDB read and write queries to explore real world data and their relations. The data set you will work with consists of climate data objects. You are asked to implement a set of target workloads to explore different features of the data. Some workloads use commands and operators not covered in the lecture or labs. The relevant MongoDB manuals are listed in the reference section.

2 Data set

The data set consists of one JSON file: `climates.json`. This file is an adaptation of the JSON document that is stored in the GitHub repository that can be found here. This file contains many JSON objects where each object stores climate information about a particular city.

An example of such an object is as follows:

```
{
  "_id": 100,
  "city": "Zurich",
  "region": "Switzerland",
  "monthlyAvg": [
    {
      "high": 4,
      "low": -2,
      "dryDays": 15,
      "snowDays": 12,
      "rainfall": 59.1
    },
    {
      "high": 6,
      "low": -2,
```

```
        "dryDays": 14,
        "snowDays": 11,
        "rainfall": 64.1
      },
      {
        "high": 11,
        "low": 1,
        "dryDays": 13,
        "snowDays": 7,
        "rainfall": 70.5
      },
      {
        "high": 15,
        "low": 3,
        "dryDays": 13,
        "snowDays": 3,
        "rainfall": 84.8
      },
      {
        "high": 20,
        "low": 8,
        "dryDays": 14,
        "snowDays": 0,
        "rainfall": 101.7
      },
      {
        "high": 23,
        "low": 11,
        "dryDays": 12,
        "snowDays": 0,
        "rainfall": 101.2
      },
      {
        "high": 25,
        "low": 13,
        "dryDays": 14,
        "snowDays": 0,
        "rainfall": 121
      },
      {
```

```
        "high": 25,
        "low": 13,
        "dryDays": 15,
        "snowDays": 0,
        "rainfall": 117
      },
      {
        "high": 20,
        "low": 9,
        "dryDays": 12,
        "snowDays": 0,
        "rainfall": 85.9
      },
      {
        "high": 15,
        "low": 6,
        "dryDays": 11,
        "snowDays": 2,
        "rainfall": 90.2
      },
      {
        "high": 8,
        "low": 1,
        "dryDays": 11,
        "snowDays": 8,
        "rainfall": 80.4
      },
      {
        "high": 4,
        "low": -1,
        "dryDays": 14,
        "snowDays": 12,
        "rainfall": 71.8
      }
    ]
}
```

Each JSON object in climates.json contains the following fields:

- `_id`: A unique integer identifier;
- `city`: The name of the city in string format;

- `region`: The name of the country or dependent territory in string format; and
- `monthlyAvg`: An array containing data on the city's climate for each calendar month whose structure is described in more detail below.

The `monthlyAvg` field contains a 12-element array, where each element represents a calendar month in ascending order (January to December). Each element consists of a document containing the following fields:

- `high`: The monthly average maximum temperature in degrees Celsius (°C);
- `low`: The monthly average minimum temperature in degrees Celsius (°C);
- `dryDays`: The monthly average number of days where there is no precipitation;
- `snowDays`: The monthly average number of days where snow falls; and
- `rainfall`: The monthly average amount of rainfall that occurs in millimetres (mm).

For example, the fifth element in the `monthlyAvg` array given in the example above gives the following climatic information on Zurich, Switzerland:

- The average maximum temperature in May is 20 °C;
- The average minimum temperature in May is 8 °C;
- The average number of days in May that experience no precipitation is 14;
- The average number of days in May that experience snowfall is 0; and
- The monthly average precipitation for May is 101.7 mm.

Some JSON objects contain some fields whose value is null. You may ignore null values in your query design.

3 MongoDB setup

You are asked to import the data into a MongoDB database called `city_data`. Data modelling is NOT required for this assignment. You should import the data into a single collection called `climates` with each row imported as a document in the collection. In a plain text file called `import.txt`, include either the `mongoimport` command that you have used for importing the data from `climates.json` into your MongoDB database or an explanation on the procedures you have followed to import the data from `climates.json` if you have used another tool.

4 Query workload

Q1 Write a **find** query to determine the number of cities in the data set that are in Germany.

Q2 Write a **find** query to determine the number of cities in the data set that have a monthly average of at least one snow day in November. You may ignore null values.

Q3 Write a **distinct** query to determine the names of all regions that have at least one city that has a monthly average minimum temperature of 2 °C or less for March and a monthly average maximum temperature of 25 °C or more for July. Your query must return an array of strings. Each string must represent the name of a region. Each relevant region must be displayed exactly once.

Q4 Write a **find** query to determine the top five cities that have the highest average monthly rainfall for June. Each city in your result must be a document in the format {city: city_name, region: region_name} where city_name is the name of a city and region_name is the region in which this city is located. Sort the results by average monthly rainfall in June in descending order.

Q5 Write a **find** query to determine the names of all cities whose names contain at least one character that appears twice consecutively (for example, “Brussels” meets this requirement since it contains “ss” but “Sydney“ does not). Each city in your result must be a document in the format {city: city_name} where city_name is the name of a relevant city. Sort the results by city_name in ascending order.

Q6 Write an **updateMany** query to determine to insert a new field called rainy_mid_year and set its value to “yes” in every document representing a city where the monthly average number of dry days in each of June, July, and August is 15 or less.

5 Implementation Requirements

Implement queries one to five using a single MongoDB find or distinct command as indicated in the query description; implement query six using a single updateMany command. You may also use the count, limit, and sort methods for queries one to five. However, you must not use other commands such as aggregate. Each workload implementation must be placed in a separate JavaScript file using the following filenames:

- q1.js - Q1 implementation;
- q2.js - Q2 implementation;
- q3.js - Q3 implementation;

- q4.js - Q4 implementation;
- q5.js - Q5 implementation; and
- q6.js - Q6 implementation

No other files are permitted (other than `a1_driver.js`). You must not modify `a1_driver.js`.

Your JavaScript files must be able to execute correctly when they are called from the supplied `a1_driver.js` file. Running this driver will display the results of all queries in order.

For queries one to five, you must store your result in a variable called `res`.

Dummy scripts (w1.js - w6.js) are provided that show the required format.

Do not include configuration statements in your JavaScript files; include only the query statements in your JavaScript scripts. The driver file contains all the required configuration statements.

6 Generative Artificial Intelligence and Writing Assistance Tools Use Policy

You must not use generative artificial intelligence or writing assistance tools. Zero marks may be awarded for your assignment submission if evidence of use of such tools is found in your submission.

7 Deliverables and Submission Rules

The deliverable for this assignment is a single zip file. This file must contain `import.txt` as described in the "MongoDB setup" section and the JavaScript files of your solutions as described in the previous section.

Submit your file to the appropriate Assignment submission inbox in Canvas. The submission deadline is 16 August 2024, 23:59 Sydney time. Late penalties apply for late submissions. Note that this is a short release assignment. Simple extensions are not applicable.

Do not include `a1_driver.js` in your submission.

8 References

- MongoDB regular expression for string matching
- MongoDB distinct command manual