Course: Cloud Data Management

Professor: Dr.Zhang, Xuechen

Final Project: Freeway Data of Portland Oregon Metropolitan Region

Team Member: Alseny Diallo, Soklong Lim

I. Scope of Final Project:

First, introduce students to the concepts and aspects that are specific to distributed data management systems. Second, give students a chance to acquire practical experience with a distributed data management system of their choice.

II. Introduction and Background:

A distributed database is a collection of multiple interconnected databases, which are spread physically across various locations that communicate via a computer network.

Advantages of Distributed Databases

Following are the advantages of distributed databases over centralized databases.

Modular Development − If the system needs to be expanded to new locations or new units, in centralized database systems, the action requires substantial efforts and disruption in the existing functioning. However, in distributed databases, the work simply requires adding new computers and local data to the new site and finally connecting them to the distributed system, with no interruption in current functions.

More Reliable − In case of database failures, the total system of centralized databases comes to a halt. However, in distributed systems, when a component fails, the functioning of the system continues may be at a reduced performance. Hence DDBMS is more reliable.

Better Response − If data is distributed in an efficient manner, then user requests can be met from local data itself, thus providing faster response. On the other hand, in centralized systems, all queries have to pass through the central computer for processing, which increases the response time.

Lower Communication Cost − In distributed database systems, if data is located locally where it is mostly used, then the communication costs for data manipulation can be minimized. This is not feasible in centralized systems.

III. Objectives:

The learning Objectives for the projects nails down to the understanding of what a distributed database management system (DDBMS) is and what its components are. How database implementation is affected by different levels of data and process distribution. How transactions are managed in a distributed database environment. How database design is affected by the distributed database environment

IV. System Requirements:

Operating System: macOS Sierra 10.12.x
Software/platform: MongoDB 3.2.10
Programming Tools: Sublime Text

Why we choose MongoDB as the database:
MongoDB is a document model database which corresponds to native types in many programming languages including JavaScript which we use for query executions in this project. It supports both embedded data models reduce I/O activity and automatic failover as well as avoid redundancy. Since the given data set for this project is in .csv format, importing the files into MongoDB is fairly easy because as mentioned above it is an document model database.

V. Design and Methodology:

Data

For this project, we use data collected for the section of I-205 NB for a 2-month test period: freeway_loopdata.csv, freeway_detectors.csv, freeway_stations.csv, highways.csv.

Methodology:

- Imported data (freeway_loopdata.csv, freeway_detectors.csv, freeway_stations.csv, highways.csv) in to the mongodb DDMS.
- Converted date field of the loop_data.csv collection.
- Uploaded the query implementation written in java script to the mongodb data base
- Run all queries.

VI. Implementation and Execution Results:

(see query implementation '.js' source file for the solution and execution of each query)

QUERY NUMBER	IMPLEMENTATION TIME IN (SEC)
1	8.642 sec
2	8.93 sec
3	8.856
4	14.149 sec
5	15.177 sec
6	0.003 sec

Query number 6: Route Finding: Find a route from Johnson Creek to Columbia Blvd on I-205 NB using the upstream and downstream fields.

```
//QUERY #6
var before = new Date();
//find the source station
var sourceStation = db.freeway_stations.find({locationtext : "Johnson Cr NB"}).toArray();

//find the destination station
var destinationStation = db.freeway_stations.find({locationtext : "Columbia to I-205 NB"}).toArray();


var route = sourceStation[0].locationtext + " -> ";             //variable containing the route
var nextid = sourceStation[0].downstream;   // //variable containing the current station
var cursor = "";
// loop through the table and match current upstream and downstream to the next upstream and downstream
do{
    cursor = db.freeway_stations.find({stationid : nextid});
    //cursor = db.freeway_stations.find({downstream : nextid});
    if(cursor){
        route += cursor[0].locationtext + " -> ";
    }
    nextid = cursor[0].downstream;

    if(nextid == destinationStation[0].stationid)
        break;

} while(true);

route += destinationStation[0].locationtext;
print(route);

var after = new Date();
execution_mills = after - before;
print("Query execution time " + execution_mills/1000 + " sec");
```

Results:

```
> load("queries-6.js")
Johnson Cr NB -> Foster NB -> Powell to I-205 NB -> Division NB -> Glisan to I-205 NB -> Columbia to
I-205 NB
Query execution time 0.013 sec
true
>
```

VII. Problems and Solutions

- Problem: We have issue on coding JavaScript since we have not had a lot programming project which requires JavaScript.
- Solution: We need to spend time learning about JavaScript.
- Problem: MongoDB is new to us so its shell commands and behaviors are a bit different.
- Solution: We use a lot of time figure out how to install, import data and query execution.
- Problem: The given data set is huge and contains many variables and information.
- Solution: We need to learn and understand how the data is stored in the file and how to access each collection, document and field.
- Problem: Run into some issue with the freeway_loopdata.csv ‘start-time’ field. Hadoop read the field as a string. Run into difficulty to compute run time
- Solution: For a solution we converted the string date to ISODate format to allow the computation of average time.
- Problem: Running to some issue with data formatting.
- Solution: had to use json format in order to print the result in a friendly readable format.

VIII. Contribution and Lesson Learned

The challenge throughout this project was to learn the API and implementation of the features of the mangodb distributed management system data base. From this project I learned first how to install mongodb on Mac operating system, learned different utilization of the mongodb API and features. Improved my JavaScript language knowledge. Last but not least, I also developed my team work, task distribution, time management and communication skill.
I contributed with the development and implementation of queries and code debugging.

VIII. Conclusions

Our choice to use mongodb turns out to be a very clever choice. Uploading the file to mongodb was very fast and easy. And the execution time for every query was well kept under 20 second. Even for the query that involved scanning the entire data collection and applying an aggregation ran very fast.
Future work may involve optimizing the java script implementation of the query. We believe there is always room for improvement. With that said, we were able to implement the query, however, there maybe be a better and most efficient way of implementing the queries.

References

[1] https://docs.mongodb.com/v3.2/tutorial/install-mongodb-on-os-x/
[2] https://mongodb.github.io/node-mongodb-native/api-articles/nodekoarticle1.html
[3] https://www.tutorialspoint.com/mongodb/mongodb_environment.htm
[4] https://docs.mongodb.com/v3.0/core/shell-types/
[5] https://docs.mongodb.com/v3.0/tutorial/map-reduce-examples/
[6] https://docs.mongodb.com/v3.0/reference/program/mongoimport/