**Name:** Thien Nguyen & Shen Yang

**Assignment 4 MongoDB Due:** Wednesday, April 23 (9AM)

This assignment will give you experience in using NoSQL databases for projects. NoSQL databases are very popular and represent a paradigm shift in data storage approach. A NoSQL database provides a mechanism for storage and retrieval of data that is modeled in means other than the tabular relations used in relational databases. Motivations for this approach include simplicity of design, horizontal scaling and finer control over availability. The data structure (e.g., tree, graph, key-value) differs from the RDBMS, and therefore some operations are faster in NoSQL and some in RDBMS. There are differences though and the particular suitability of a given NoSQL DB depends on the problem to be solved.

As these databases are focused on scalability and easy of storage, they behave quite differently and go without various popular relational concepts like joins etc. which are not implemented directly. The broad idea is to move all the complex operations, for example joins, to the application layer. In this was the database strictly behave as a highly optimized storage and retrieval resource. For the purpose of this assignment, we will be using MongoDB which is one of the most popular NoSQL databases.

Link to access a front end to the database: <http://gravity.cc.gatech.edu>

Note: You need to be on the campus network (or VPN) to access the site.

Credentials:

Username: CS4400X\_student

Password: spring2014

Database: flights

**For this assignment, you will turn in the MongoDB queries and their results for the following 4 queries using the given flights database described on the next page. You will also demo your queries in class on either Wednesday, April 23 or Friday, April 25 to one of the TAs or the professor.**

**Query 1:** Find all cancellation descriptions and their codes.

function () {

var cur = db.cancellation.find();

var out = [];

while (cur.hasNext())

{

obj = cur.next();

out .push(obj);

}

return (out);

}

**Query 2:** Compute the total number of cancellations for each cancellation code.

function () {

var out = [];

out.push("Cancellation code: A");

var cur = db.flight\_info.aggregate([

{ $match: {$or: [{"CANCELLATION\_CODE" :"A" }]}},

{ $group: {

\_id: null,

count: {$sum: 1} }

},

])

out.push(cur);

out.push("Cancellation code: B");

var cur1 = db.flight\_info.aggregate([

{ $match: {$or: [{"CANCELLATION\_CODE" :"B" }]}},

{ $group: {

\_id: null,

count: {$sum: 1} }

},

])

out.push(cur1);

out.push("Cancellation code: C");

var cur2 = db.flight\_info.aggregate([

{ $match: {$or: [{"CANCELLATION\_CODE" :"C" }]}},

{ $group: {

\_id: null,

count: {$sum: 1} }

},

])

out.push(cur2);

out.push("Cancellation code: D");

var cur3 = db.flight\_info.aggregate([

{ $match: {$or: [{"CANCELLATION\_CODE" :"D" }]}},

{ $group: {

\_id: null,

count: {$sum: 1} }

},

])

out.push(cur3);

return(out);

}

**Query 3:** Find flight information for instances of flight number 1778 that been cancelled because of weather.

function () {

var out = [];

var cur2 = db.flight\_info.find({"CANCELLATION\_CODE":"B", "FL\_NUM" : 1778});

while (cur2.hasNext()) {

obj2 = cur2.next();

out.push(obj2);

}

return (out);

}

**Query 4:** Find the airport ID and its city with the largest number of cancelled flights.

function () {

var cur = db.flight\_info.aggregate([

{ $match: {$or: [{"CANCELLED": 1 }]}},

{

$group: {

\_id: {

ORIGIN\_AIRPORT\_ID: "$ORIGIN\_AIRPORT\_ID",

ORIGIN\_CITY\_NAME: "$ORIGIN\_CITY\_NAME"

}, count: {$sum: 1}

}

},{$sort: {count: -1}},{$limit: 1}

])

return(cur);

}

Look at the “Tools” section on the Rockmongo UI menu bar. You will find command and execute options useful.

**NOTE:** The database is shared, use it only for read operations and do not modify the existing collections.

If you have any questions regarding MongoDB queries or the Flights database, you can take a look at some of the resources or contact Prof. O. or the TAs: Siddharth and Zero.

Rockmongo UI (the one we are using) is quite popular and many resources available online for reference.

Data Source: <http://www.transtats.bts.gov/DL_SelectFields.asp?Table_ID=236>

Mongo Query cheat sheet: <http://docs.mongodb.org/manual/reference/sql-comparison>

Javascript reference: <http://www.w3schools.com/js/>

There are also other MongoDB resources on our class T-Square site.

Schema of the Flight database (flight\_info and various lookup collections)

Lookup Tables

|  |
| --- |
| unique\_carriers |
| Code |
| Description |

|  |
| --- |
| airline\_id |
| Code |
| Description |

|  |
| --- |
| months |
| Code |
| Description |

|  |
| --- |
| yesno\_resp |
| Code |
| Description |

|  |
| --- |
| airport |
| Code |
| Description |

|  |
| --- |
| world\_area\_codes |
| Code |
| Description |

|  |
| --- |
| cancellation |
| Code |
| Description |

|  |
| --- |
| city\_market\_id |
| Code |
| Description |

|  |
| --- |
| state\_abr\_aviation |
| Code |
| Description |

|  |
| --- |
| airport\_seq\_id |
| Code |
| Description |

|  |
| --- |
| airport\_id |
| Code |
| Description |

Data Table

|  |
| --- |
| flight\_info |
| MONTH |
| UNIQUE\_CARRIER |
| AIRLINE\_ID |
| FL\_NUM |
| ORIGIN\_AIRPORT\_ID |
| ORIGIN\_AIRPORT\_SEQ\_ID |
| ORIGIN |
| ORIGIN\_CITY\_NAME |
| ORIGIN\_STATE\_ABR |
| ORIGIN\_STATE\_NM |
| ORIGIN\_WAC |
| DEST\_AIRPORT\_ID |
| DEST\_AIRPORT\_SEQ\_ID |
| DEST\_CITY\_MARKET\_ID |
| DEST |
| DEST\_CITY\_NAME |
| DEST\_STATE\_ABR |
| DEST\_STATE\_NM |
| DEST\_WAC |
| CRS\_DEP\_TIME |
| DEP\_TIME |
| DEP\_DELAY |
| DEP\_DELAY\_NEW |
| WHEELS\_OFF |
| WHEELS\_ON |
| CRS\_ARR\_TIME |
| ARR\_TIME |
| ARR\_DELAY |
| ARR\_DELAY\_NEW |
| CANCELLED |
| CANCELLATION\_CODE |
| DIVERTED |
| CRS\_ELAPSED\_TIME |
| ACTUAL\_ELAPSED\_TIME |
| AIR\_TIME |
| FLIGHTS |
| CARRIER\_DELAY |
| WEATHER\_DELAY |
| NAS\_DELAY |
| SECURITY\_DELAY |
| LATE\_AIRCRAFT\_DELAY |