**HW 4. NoSQL - Document Database**

**IS 5315**

**What to submit:** ***Upload one Word document*** to the Canvas Course Web.

**Grading:**

You should provide at least **3 or more advanced-level queries**, using JOIN and other features of SQL.

**Database Comparison:**

|  |  |  |
| --- | --- | --- |
| **Terminology** | **MySQL** | **MongoDB** |
| **Database** | Database | Database |
| **Entity** | Table | Collection |
| **Data** | Record | Document |

**What to Write in HW Word Document:**

1. Discuss the three topics:
   1. what is Document Database
   2. what is MongoDB and identify the benefits of MongoDB
   3. why SQL queries and queries in MongoDB are different
2. Your task is using "enron" email database to achieve either customer satisfaction or employees’ performance
3. Build **Five Queries** to achieve your goas above. Each query should have:
   1. A question in plain English
   2. A query in MongodDB
   3. A screenshot of Robo 3T query results

**What to achieve:**

1. **SERVER:**

mkdir ~/mongodb

# mongodb container

docker run --name mongo -d -v ~/mongodb:/data/db -p 27017:27017 mongo

# id: bigdata pw: bigdataftw

1. **LOCAL:** In your file manager, download ‘enronemail.json’ file from the Canvas and save it into the git folder, 5315

# OS X, open a terminal

# Windows, open git bash

cd ~/web/5315

git add .

git commit -m "adding enronemail json file"

git push origin master

1. **SERVER:**

# Update your folder

cd ~/web

git pull origin master

# Copy the json file to mongo folder

cp enronemail.json ~/mongodb/

# connect to mongodb container

docker exec -it mongo bash

# import enronemail.json

mongoimport -d bigdata -c enron /data/db/enronemail.json

# mongo shell

mongo

> show databases

> use bigdata

> show collections

> db.enron.count()

> db.enron.count({"sender": "steven.kean@enron.com"})

> db.enron.find({"sender": "steven.kean@enron.com"})

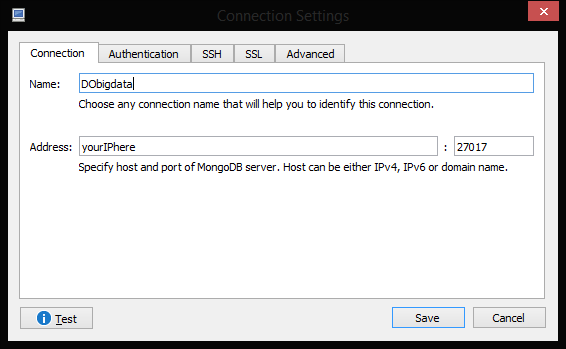
# to exit mongo shell

> ctrl+c

# to exit mongo container

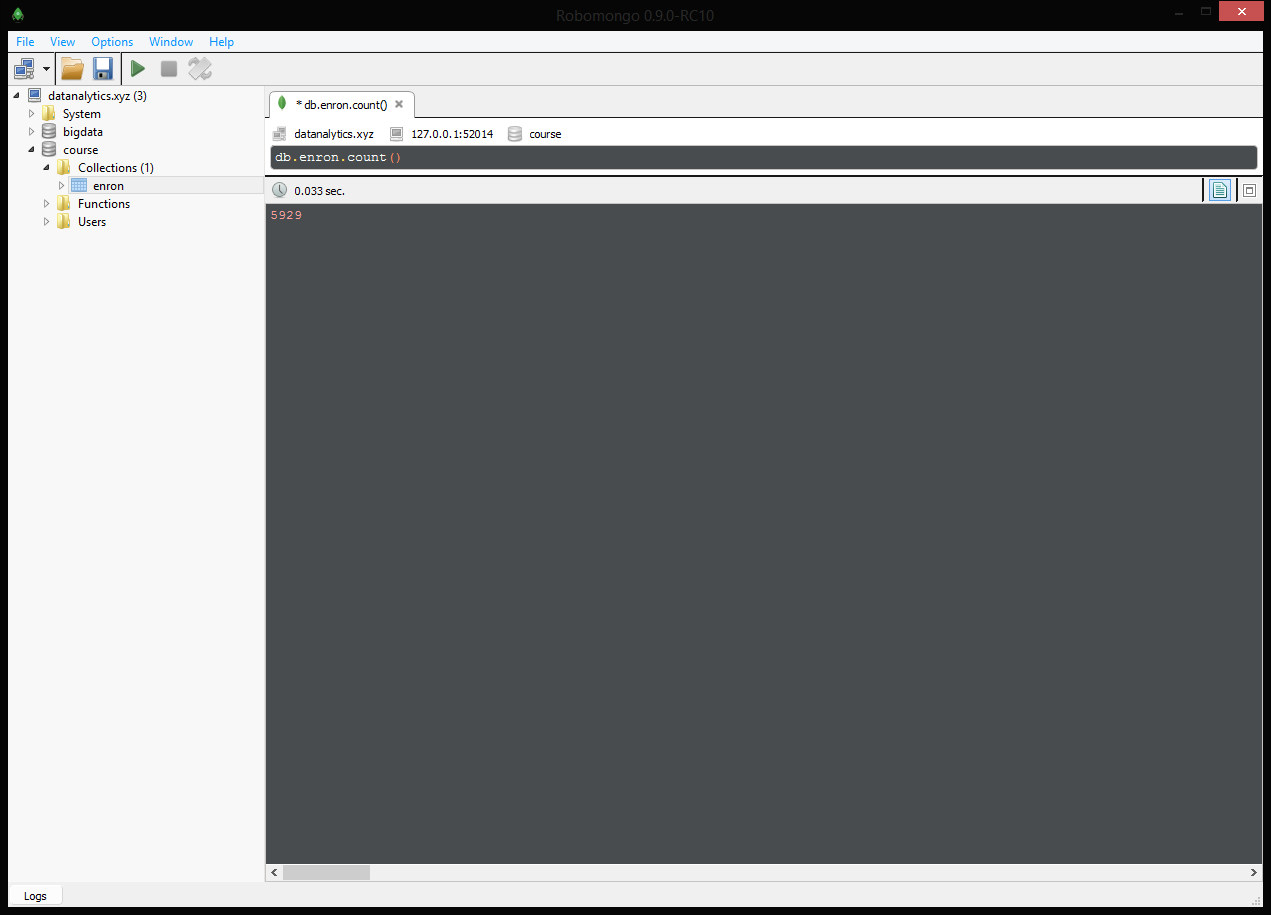
> ctrl+d

1. **Robo 3T**: Let’s intall a MongoDB admin UI ***in your local computer***. Download and install Robo 3T from the following site. <https://robomongo.org/download>
2. Run the program, then “Connect”, “Create”, finally enter your IP address with a proper name:



1. Run a query. Enter the below query to the top box then click the play button.

db.enron.count()



**Query samples:**

db.enron.find()

db.enron.find({"sender": "rosalee.fleming@enron.com"})

db.enron.find({"sender": "rosalee.fleming@enron.com"},{"date":1,"subject":1,"to":1})

db.enron.count()

db.enron.count({"sender": "steven.kean@enron.com"})

db.enron.find({"sender": "steven.kean@enron.com"})

db.enron.find({"sender": "steven.kean@enron.com"},{"date":1,"subject":1,"to":1})

db.enron.find({"sender": "steven.kean@enron.com"},{"date":1,"subject":1,"to":1,"\_id":0})

db.enron.find({"text":/breakfast/},{"sender":1,"date":1,"subject":1,"to":1,"text":1,"\_id":0})

db.enron.find({"date":{"$gt": ISODate("2001-02-01")}},{"sender":1,"date":1,"subject":1,"\_id":0})

db.enron.find( { $where: function() { return ( this.to.indexOf(this.sender) > -1) }})

db.enron.find( { $where: function() { return ( (this.cc.indexOf(this.sender) > -1) ) }})

db.enron.find( { $where: function() { return ( (this.to.indexOf(this.sender) > -1) || (this.cc.indexOf(this.sender) > -1) || (this.bcc.indexOf(this.sender) > -1) ) }}).count()

**SQL vs NoSQL (MongoDB):**

Return all the documents in an enron collection (all rows in the order table):

**SELECT \* FROM enron**

**db.enron.find()**

Select the all email for a sender of rosalee.fleming@enron.com:

**SELECT \* FROM enron WHERE sender = "rosalee.fleming@enron.com"**

**db.enron.find({"sender": "rosalee.fleming@enron.com"})**

Select date and subject for a sender of rosalee.fleming@enron.com:

**SELECT date,subject FROM enron WHERE sender = "rosalee.fleming@enron.com"**

**db.enron.find({"sender": "rosalee.fleming@enron.com"},{"date":1,"subject":1,"to":1})**

**References:**

<https://docs.mongodb.com/manual/>

<https://hub.docker.com/_/mongo/>

<http://jsonstudio.com/resources/>

**Assignment Sample** - do NOT use below queries in your HW

**Your task is using "enron" eamil database to achieve either customer satisfaction or employees’ performance**

**Question 1:** Find out all the emails with dates between 2001-01-01 and 2001-03-15.

**Query:**

db.enron.find( {"date":{"$gt": ISODate("2001-01-01"),"$lt": ISODate("2001-03-15")}}, {"sender":1,"date":1,"subject":1,"\_id":0})

**Screenshot:**

**Question 2:** Jeffrey Skilling are found guilty of conspiracy and fraud. Find emails that are sent to him, including cc, bcc, and to.

**Query:**

db.enron.find( { $where: function() { return ( (this.to.indexOf("jeffrey.skilling@enron.com") > -1) || (this.cc.indexOf("jeffrey.skilling@enron.com") > -1) || (this.bcc.indexOf("jeffrey.skilling @enron.com") > -1) ) }})

**Screenshot:**

**Question 3:** What emails include the word, board meeting, in the text and sent by rosalee.fleming@enron.com, showing with sending, date in descending order, subject, to and text?

**Query:**

db.enron.find({"text":/board meeting/, "sender":"rosalee.fleming@enron.com" },{"sender":1,"date":1,"subject":1,"to":1,"text":1}).sort({"date":-1})

**Screenshot:**

**Question 4:** Internal audit department is looking for email communication supporting documents with Shea Dugger after 2001-03-15. What’s the email with subject including “meeting”, sent to [shea\_dugger@i2.com](mailto:shea_dugger@i2.com) after 2001-03-15?

**Query:**

db.enron.find({

$and : [

{"subject":/meeting/},

{"date":{"$gt": ISODate("2001-03-15")}},

{$where: function() { return ( this.to.indexOf("shea\_dugger@i2.com") > -1)}}

]

})

**Screenshot:**

**Question 5:** There are different kinds of institutions. Assume that we identify these institutions by their email suffix (eg: edu, company, net, org, us, mil…). How many emails had been sent out from each kind of institution (excluding Enron themselves) to Enron and for each kind of institution, when did they first and last send out the email?

**Query:**

db.enron.aggregate([

{ $project :

{ senderType: { $arrayElemAt: [ { $split: ["$sender", "."] }, -1 ] },

sender:1,

recipients:1,

date : 1,

"\_id" : 0

}

},

{ $match : { $and: [{ sender : {$not:/@enron.com/}},{recipients :/@enron.com/}]}},

{ $group :

{

\_id : "$senderType" ,

maxDate : { $max: "$date"},

minDate : { $min: "$date"},

count : { $sum : 1 }

}

},

{ $sort : {count : -1}}

]);

**Screenshot:**

## **Assignment Rubric**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Capstone | Milestone Plus | Milestone | Benchmark |
| 10 | 8 | 6 | 4 |
| Explanation of Issues | Issue/problem to be considered critically is stated clearly and described comprehensively, delivering all relevant information necessary for full unerstanding. | Issue/problem to be considered critically is stated, described, and clarified so that understanding is not seriously impeded by omissions. | Issue/problem to be considered critically is stated but description leaves some terms undefined, ambiguities unexplored, boundaries undetermined, and/or backgrounds unknown. | Issue/problem to be considered critically is stated without clarification or description. |
| External Sources | Accesses information using effective, well-designed search strategies and most appropriate information sources. | Accesses information using variety of search strategies and some relevant information sources. Demonstrates ability to refine search. | Accesses information using simple search strategies, retrieves information from limited and similar sources. | Accesses information randomly, retrieves information that lacks relevance and quality. |
| IT Applicability | Implements the solution in a manner that addresses thoroughly and deeply multiple contextual factors of the problem. | Implements the solution in a manner that addresses multiple contextual factors of the problem in a surface manner. | Implements the solution in a manner that addresses the problem but ignores relevant contextual factors. | Implements the solution in a manner that does not directly addresses the problem statement. |
| Organization | Organizational pattern is clearly and consistently observable and is skillful and makes the content of the presentation cohesive. | Organizational pattern is clearly and consistently observable within the presentation. | Organizational pattern is intermittently observable within the presentation. | Organizational pattern is not observable within the presentation. |