# PILOT STUDENTS – Oct/Nov 2015 (this applies to you)

# (1) When you see an instruction to upload something, upload it to your desktop. If faculty needs it, they will ask for it. Contact info is in your FAQ.

# Intro to NoSQL Lab 3: The Document Database / MongoDB Hands On

## What Am I Supposed to Do?

1. Follow the instructions of the Lab.
2. Periodically, you’ll be instruction to **STOP** and do what is requested. Typically, we ask you to copy a screen shot. **YOU MUST DO THIS IF YOU WANT TO PASS THE COURSE!**

## BEGIN LAB:

1. Make sure you have your node.js Setup from Lab 1 up and running. You’ll need the following packages unpacked and **ready** (i.E. npm install executed within their directory)
   1. nosql-lab-dataimport
   2. nosql-lab-mongo
2. Install MongoDB
   1. You can get it from <https://www.mongodb.org/downloads#production>. Download and Install the Windows edition (Release 3.0.7 / Windows 64 bit 2008 R2+ legacy).
   2. Select custom install and installl mongoDB somewhere in the c: (like c:\mongoDB)
   3. Install ☺
3. Start the MongoDB database
   1. Create a data folder for the database
      1. In Command Prompt, navigate to the directory where mongoDB is installed
      2. Run “mkdir .\data”
   2. Run “.\bin\mongod.exe --dbpath .\data”
4. Import the data into MongoDB
   1. In the Dataimport directory, execute:
      1. node mongo.js and then
      2. node mongo\_comments.js

(Windows might throw firewall access request, should be allowed. If you get strange errors re-check point 1 of this lab…there could be dependencies missing)

1. Check in mongo client (mongo.exe) if the data is there.
   * 1. Run “.\bin\mongo.exe” to start the mongo shell
     2. In the mongo shell
        + Execute command ‘show dbs’ to list the databases
        + Execute command ‘use nosqllab’ to switch to the database used in the lab exercises
        + Execute command ‘show collections’ and see if the following collections are available in the **nosqllab** database
          - **posts**
          - **counters**
        + Execute the following commands to check the number of documents in the **posts** collection and the structure of the **counters** collection
          - > db.posts.count()

Should return count of 6206

* + - * + > db.posts.find( {comments: {"$exists": true}} ).count()

Should return count of 4467

* + - * + > db.counters.findOne()

Should return 1 record – with sequence = 27515

1. Please make sure you have Installed plugin JSONView to your browser (to view the posts in JSON format) as part of the lab1.
   1. Works with Google Chrome or Firefox

## Perform the Lab

Launch the mongo sample app

1. In the nosql-lab-mongo folder execute:
   1. SET DEBUG=nosql-lab-mongo:\* & npm start
2. You can now check out the app at <http://localhost:3000>
3. Check out what you get, get a feeling for it. You should be able to
   * 1. List the existing posts
     2. Create a new post
     3. Respond to the post
     4. Comment on the post (\*\* **try this after step 8 \*\***)
4. **STOP:** How exactly are the **responses/comments** saved in comparison to the **posts**? In a relational database, how will you design the data model for **posts** and **responses**
   1. **Hint:** normalized VS embedded ..
5. **STOP:** How would you describe the schema of this database?
   1. **Hint:** Compare the posts that you created in step 3 with the existing posts
6. Update the **‘All Posts’** query to list the first 100 posts in ascending order (from oldest to newest)
   1. **Hint**: Right now the last 50 posts are listed in descending order (newest to oldest) so you can see the posts that you created in step 3.
      1. **Program** 🡪 app.js
      2. **Route 🡪**app.get('/posts', function(req, res) {
      3. **Statement 🡪** Post.find( query, fields, options).sort({\_id: -1}).limit(50).exec(function(err, posts) {

(Restart the application after saving the code changes)

1. In Command prompt, Ctrl-C (to stop the application)
2. SET DEBUG=nosql-lab-mongo:\* & npm start (to start the app)

(\*\* *After this step, go back to the old setting, so you can see your posts \*\*. Don’t forget to restart the application after the code change.*)

1. Update the **‘New Post’** to save your name as the **owner** instead of the **default ‘student’**
   1. **Hint:** Hard coding is fine
      1. **Program🡪** app.js
      2. **Route 🡪** (app.post('/new', function(req, res, next) { )
      3. **Statement** 🡪 var newPost = new Post({

(Restart the application after saving the code changes)

* In the command prompt, Ctrl-C (to stop the application)
* SET DEBUG=nosql-lab-mongo:\* & npm start (to start the app)

1. Update the following to save the **comments** to the **posts** (Similar to how the **responses**  were saved – as an array of embedded documents)
   1. Hint (Code in this module)
      1. **Program** 🡪 app.js
      2. **Route 🡪** app.post('/comment/:postId', function(req, res, next) {

(Restart the application after saving the code changes)

1. In the command prompt, Ctrl-C (to stop the application)
2. SET DEBUG=nosql-lab-mongo:\* & npm start (to start the app)
3. On the mongo Console: Find the post with \_id = 3

db.posts.find( {\_id: 3} ).pretty()

Display with and without the .pretty() and compare against

db.posts.findOne( {\_id: 3} )

1. **STOP:** On the mongo Console: Find all the posts created on or after 2014-10-01 and create a screenshot of this with the query and result well visible. Display only the \_id, creationDate and Title.
2. **STOP:** Create an index on the creationDate field.

Run the query used in step 10 before and after creating the index and find the time taken to process the query. Again, create a screenshot of this with the query and result well visible.

Hint: Use the explain method in the ‘executionStatus’ mode.

1. **STOP:** On the mongo Console: Find all the unanswered posts (0 responses), again upload screenshot with result and query well visible.
2. **STOP:** On the mongo Console: Find the most popular posts (Top 5 posts – in terms of the number of responses to the post), again upload screenshot with result and query well visible
3. **STOP:** On the mongo console: List the first 5 posts with at least 3 responses), again upload screenshot with result and query well visible