Purpose: This lab is the first of two that address the larger theme of NoSQL databases. It focuses on installing and becoming somewhat familiar with the MongoDB document database.

MongoDB is very appropriate for use in situations where the data in a collection consists of documents that have different fields within them. The individual “records” that would normally be stored as separate rows in a relational database table are stored as separate documents in the collection. Because MongoDB allows each document in a collection to have different rows, it is as if the rows are written with slightly different schemas; almost an “is-a” relationship in a type hierarchy. This would be equivalent to trying to merge multiple tables consisting of rows that have different fields in each of the source tables. Such data does not fit the normalized relational database model.

MongoDB supports the idea that even if you know your data schema today, you may not know the data in the future, and are likely to want to execute searches and/or queries against the disparate data in your collections.

Deliverable: To earn 2 marks towards your lab score, submit the requirements listed below in a single document **and demonstrate the results to your lab professor.**

## Requirements:

1. Navigate to the following sites:

<https://www.mongodb.com/try/download/community>

and

<https://www.mongodb.com/try/download/enterprise>

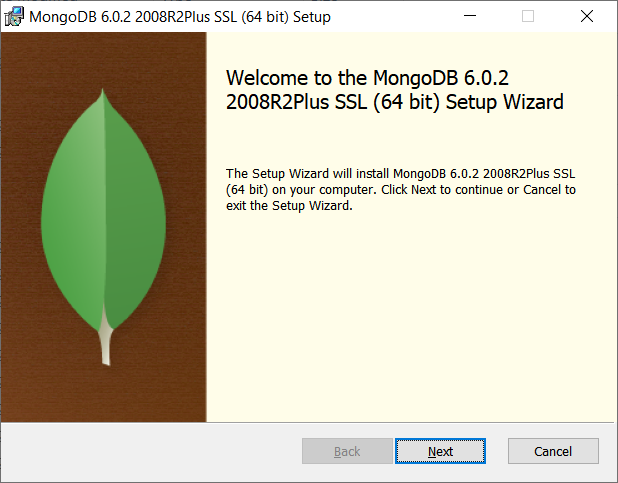
(Fill in the blank). The main differences between the community server version and the enterprise server version of MongoDB are:

**MongoDB Community Server is free and open-source, offering basic functionality suitable for individual developers and smaller projects. It includes fundamental security features, but lacks advanced security options and operational tools. In contrast, MongoDB Enterprise Server is a subscription-based, commercial product designed for larger enterprises. It provides enhanced security features, including advanced encryption and auditing capabilities, as well as additional performance optimizations, professional support, and management tools for large-scale deployments.**

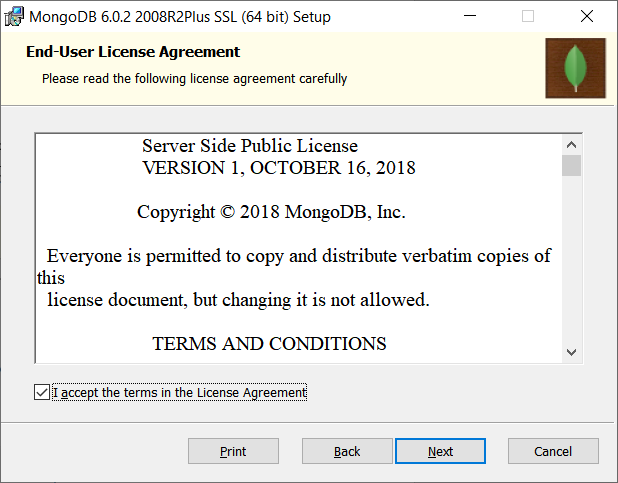
1. You will be downloading and installing the latest version of the MongoDB community server (Version 6 or later) and the Mongo shell. (1.6 or later). The best place (because it has explicit instructions) to download each of these (two downloads!) is from:

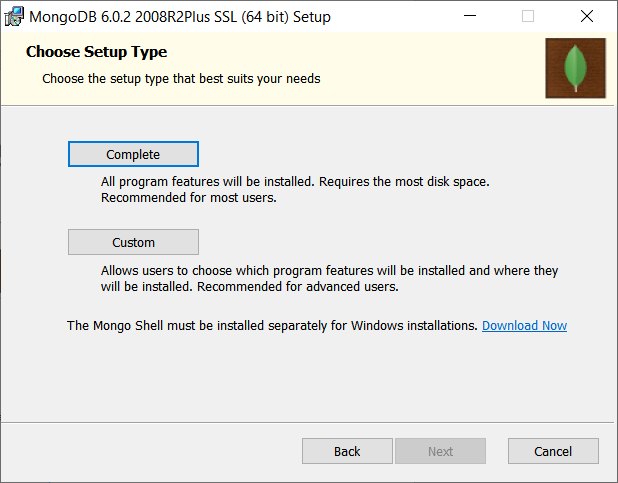
<https://www.mongodb.com/docs/manual/tutorial/install-mongodb-on-windows/>

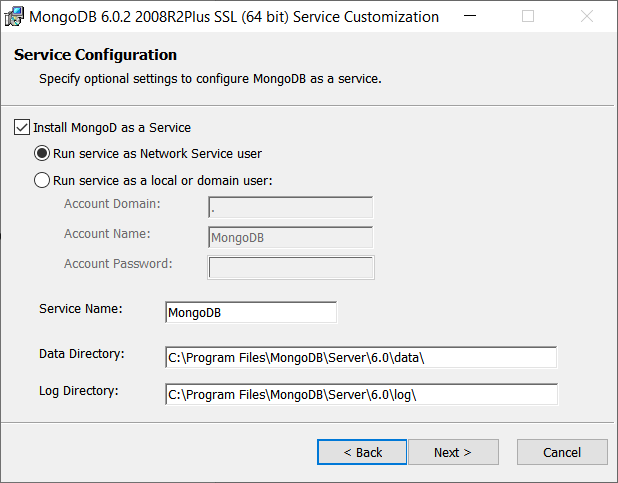
1. Download the latest version of the MongoDB community server version 6.0 or later to your laptop.
2. Download the latest version of mongoshell (mongosh). Unzip the mongoshell archive and you will have a new bin directory with mongosh inside it.
3. Get prepared to reboot during the MongoDB server install. (Close all unnecessary windows).
4. If you have antivirus software or a firewall enabled, this would be a good time to temporarily disable them. You will be installing a server software package and may otherwise experience some installation problems.)
5. Follow the installation instructions below, being careful to select that you want to do a local installation and that you want to run the mongoDB service as a network service on startup.
6. Complete the installation as follows:
   * 1. On the Welcome screen in the install wizard, click “Next”.



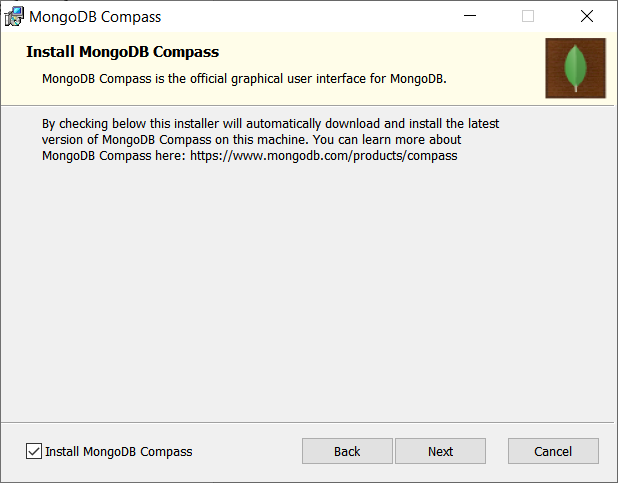
* + 1. On the End-User License Agreement, click the “I accept the terms in the License Agreement”, and then click “Next”.



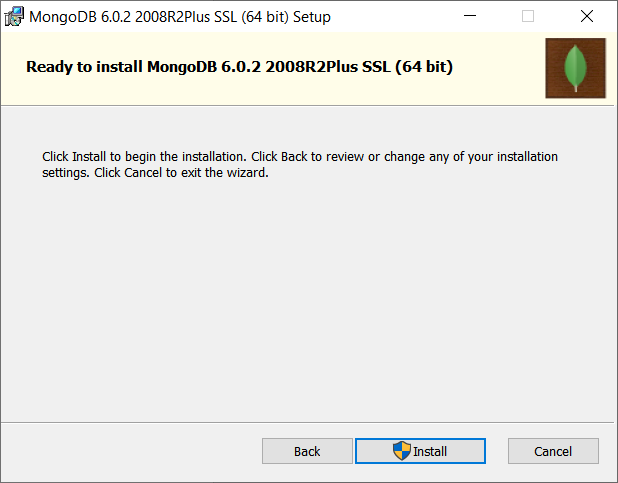
* + 1. On the Choose Setup, click Complete. 
    2. Under Service Configuration, keep the defaults: Install MongoDB as a Service, Run service as Network Service user, Service Name “MongoDB”, etc. Click Next.



* + 1. On the next screen, make sure “Install MongoDB Compass” is selected and then click “Next”.

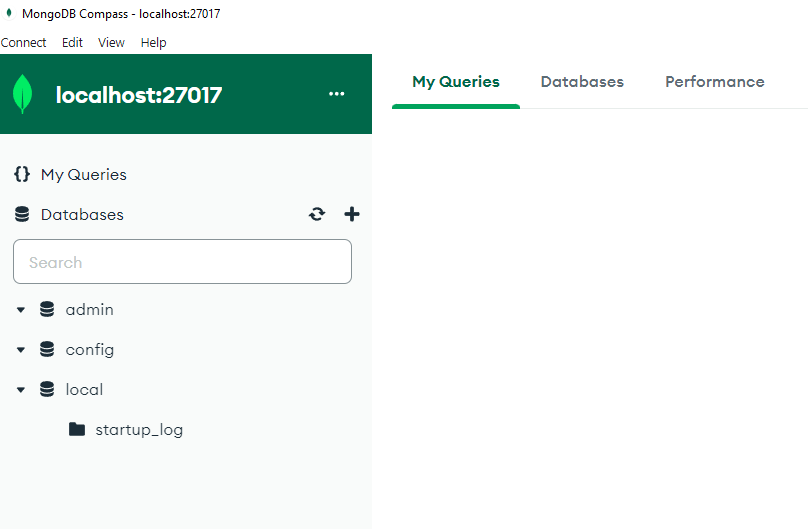


* + 1. On the Ready screen, click “Install”.



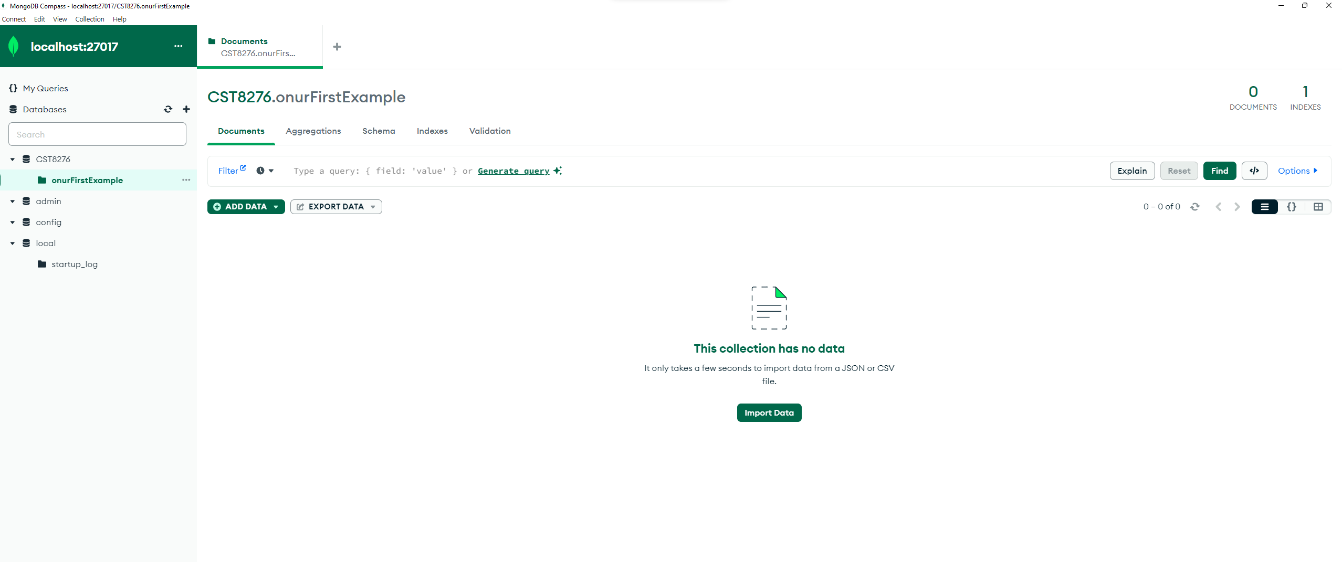
* + 1. If you are prompted to reboot, click “OK”
    2. On the MongoDB Compass Community Application that is launched, read the license agreement and click “Agree”.
    3. On the “Completed the MongoDB Setup” screen, click “Finish”.
    4. If you are prompted to restart your system, click “Yes”. Your system will reboot.

1. The MongoDB community server will be running as a service after the reboot.)
2. If the MongoDB Compass tool (the ‘Workbench’ tool) does not start automatically, start the MongoDB Compass Community application using the Windows Start Menu.
3. On the “New Connection” screen keep the defaults and click “Connect”
4. Take a screen dump of the Compass Community screen showing your list of default databases.



1. Create a database with database name “CST8276” and collection name “*myFirstname*FirstExample” (e.g., dougFirstExample).

Take a screen shot and paste it below.



Deselect the view.

Create a file “tiny.json” using Notepad containing the following

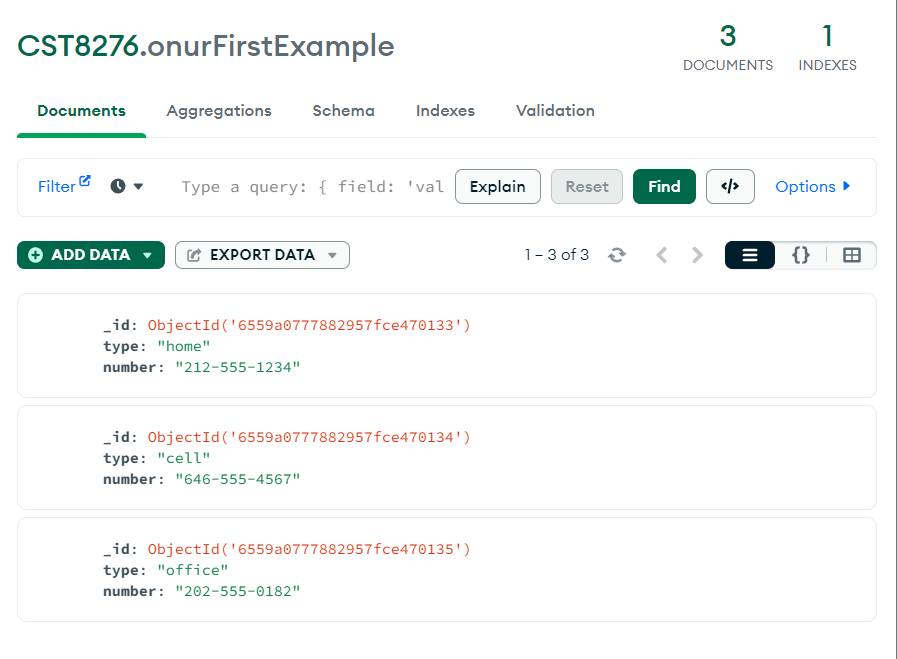
{ "type": "home", "number": "212-555-1234" }

{ "type": "cell", "number": "646-555-4567" }

{ "type": "office", "number": "202-555-0182"}

You will now import these three documents into your MongoDB collection.

1. In the MonoDB Compass Community tool, select the Collection > Import from the menu bar. Import your tiny.json file. Take a screen shot showing your CST876.*yourFirstName*FirstExample collection containing the three new documents.



1. Read the blog post:

<https://www.mongodb.com/blog/post/why-xml-in-rdbms-is-not-mongodb>

1. Try to use the xml conversion tool at

<https://convert.town/xml-to-json>

to convert the sampleXML.xml file supplied in BrightSpace to a file containing json. [Note: you may need to edit the XML and/or the output from the XML converter to get it to work correctly. Look at the output of the tool and make sure the array of expense items has been handled correctly – alternatively, you could use a different converter – see part i) below…] Save the output of the tool to “sample3.json”

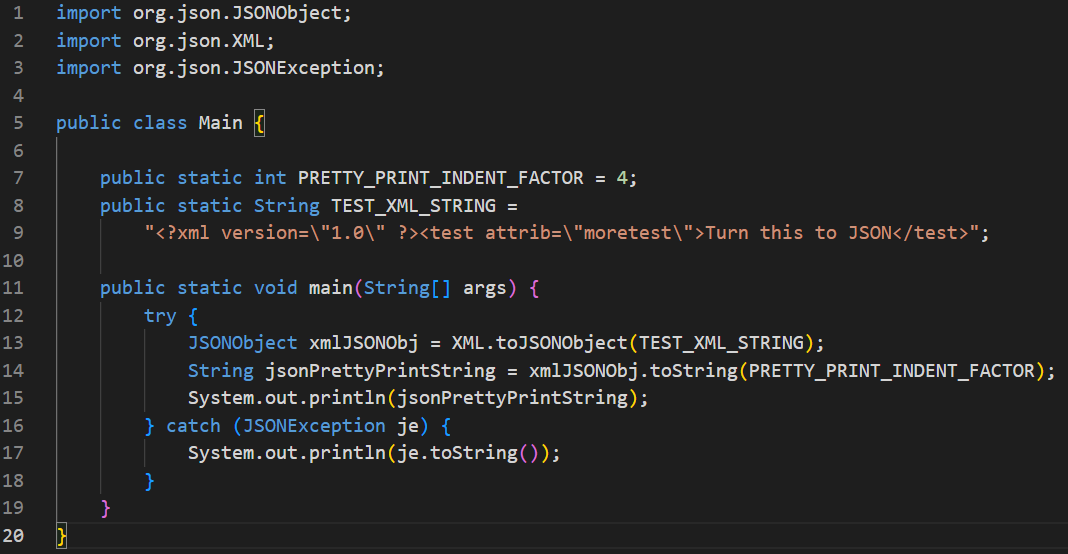
* + 1. Find 2 other online xml to json converters (e.g., using Google) and list the urls below:

<https://codebeautify.org/xmltojson>

https://jsonformatter.org/xml-to-json

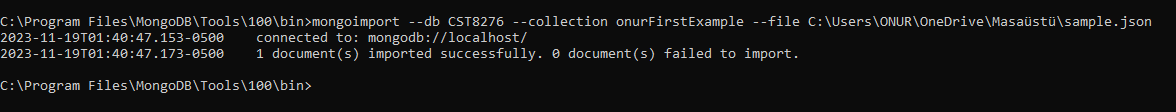
* + 1. Find an online example showing how you would convert a file containing xml to json within a Java or JavaScript program (either one.) Paste the example code below. (Just the portion that does the conversion).



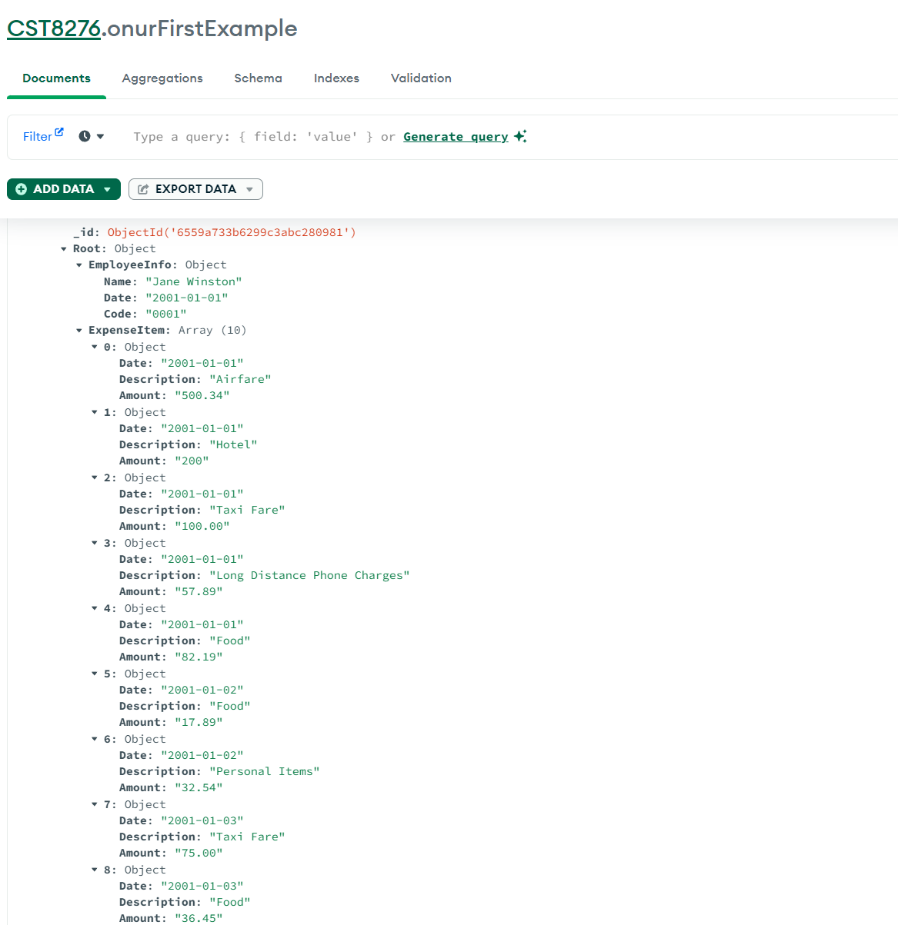


1. You will now use the “mongoimport” utility to import a json file.
   1. You need to download the database tools from:  
      <https://www.mongodb.com/try/download/database-ools>
      1. Scroll down to the “MongoDB Command Line Database Tools” section.
      2. Copy the appropriate .zip file to your PC.
      3. Then, extract **all** of the tools into your “bin” directory for MongoDB (e.g., C:\Program Files\MongoDB\Server\6.0\bin). **Hint**: You can do this by copying the files from the bin subdirectory in the .zip file into the C:\Program Files\MongoDB\Server\6.0\bin folder.
   2. Using the example screenshot below as a model, navigate to YOUR MongoDB bin directory and use “mongoimport” to import your “sample3.json” file. The example below imports “sample3.json”. Take a screen shot showing your results.

Cut and paste your screen shot into the area below



1. Navigate in MongoDB Compass to your collection. Click the refresh icon to ensure your list of objects in the collection is correct. Select the LIST option (Not TABLE) and then Click on the new document “Root”. Expand the content by clicking on each item to show the EmployeeInfo and several of the ExpenseItem array objects. Take a screen shot and paste it below.



Is the structure what you expected? Why? Explain the structure you see….

It consists of a single document with a unique identifier \_id. The document contains two main nested objects:

**EmployeeInfo:** Holds information about an employee named Jane Winston, with an associated date and code

**ExpenseItem:** An array of objects, each detailing a specific expense incurred by the employee. There are 10 entries each with a date, a description, and an amount, indicating individual costs associated with travel, accommodation, food, and other work-related expenses over a period spanning from January to March 2001.

1. Continue with the installation for mongosh (the mongo shell command-line tool) that is available at:

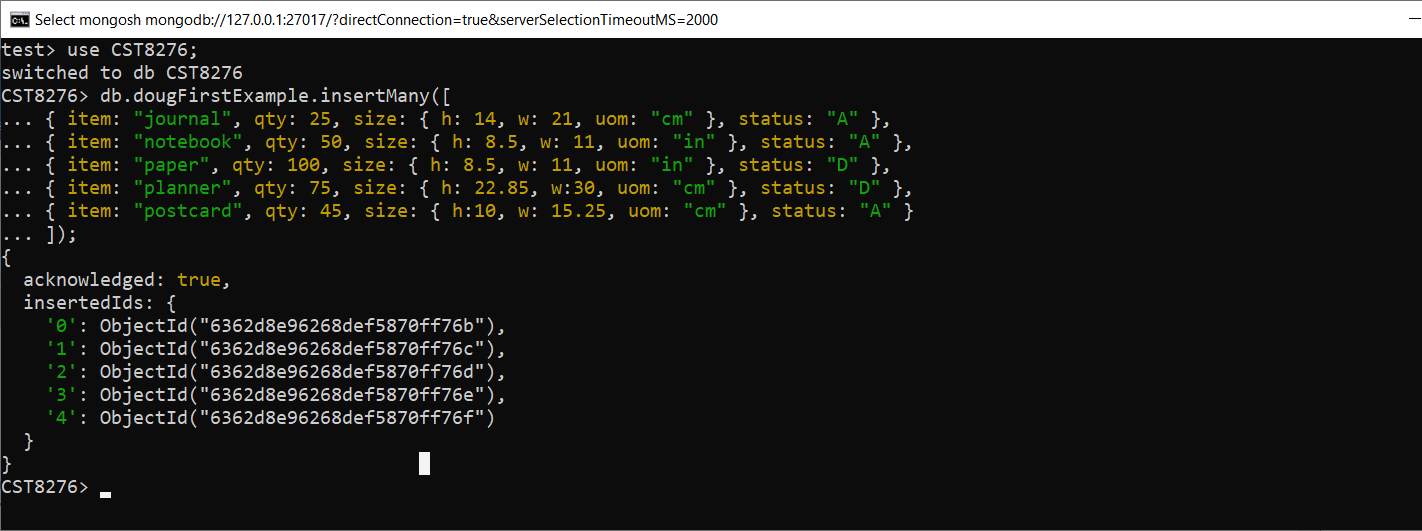
<https://www.mongodb.com/try/download/shell>

* 1. Copy the .zip file to your PC.
  2. Then, extract the bin folder files into your “bin” directory for MongoDB (e.g., C:\Program Files\MongoDB\Server\6.0\bin). **Hint**: You can do this by copying the files from the bin subdirectory in the .zip file into the C:\Program Files\MongoDB\Server\6.0\bin folder
  3. Once the copying is complete, you will have mongosh available for use later in this lab.

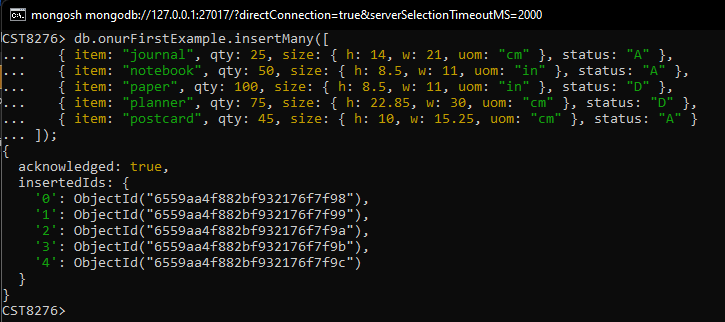
1. Open the mongo shell by opening a cmd prompt, navigating to the mongo bin directory, and running the command “mongosh”. Modify the following commands to use your collection (not the “dougFirstExample” collection) and run them in the mongo shell.

The command switches to your course database.

Here is copy of the output on my server:



1. Take a screen shot of the output in your cmd window and paste it below.

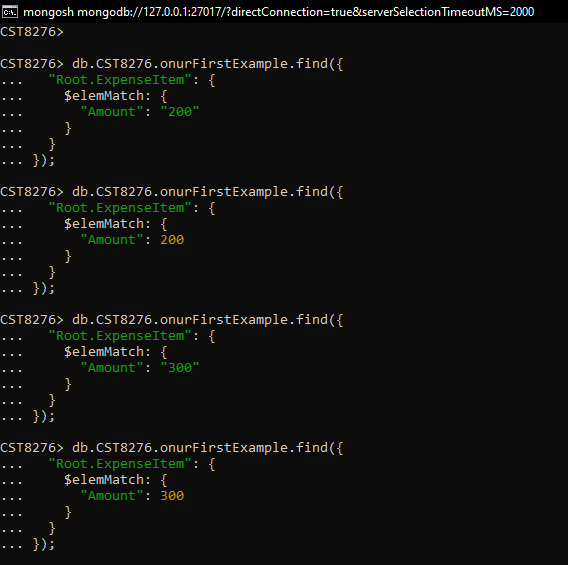


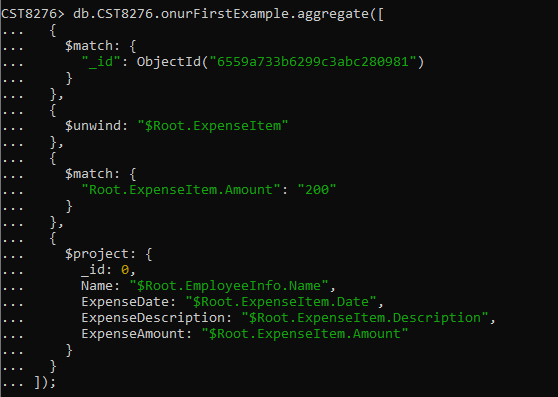
1. Look at the mongo documentation at:

<https://docs.mongodb.com/manual/tutorial/query-documents/>

Focus on queries to select from nested objects. Create a query to find all objects where the Root.ExpenseItem.Amount is equal to “200”, and again for equal to 200. Do the same for objects where the Amount is “300”, and 300. Run the queries in the mongo shell and paste the results below. [You may be surprised by the output.]

1. Rewrite your query above to make use of a combination of the “aggregate”, “match”, “find”, “unwind”, and “project” features of the query language to produce a query that returns just the name of the person and the particular expense item details for the item that actually match the value “200”. (Make sure you project the data so that other expense items are not included. Hint: an “aggregate” pipeline with unwind, match, and project will work…. You will need to look at the mongo documentation.) Provide a Screenshot showing your new query and the output.





**You're done.**