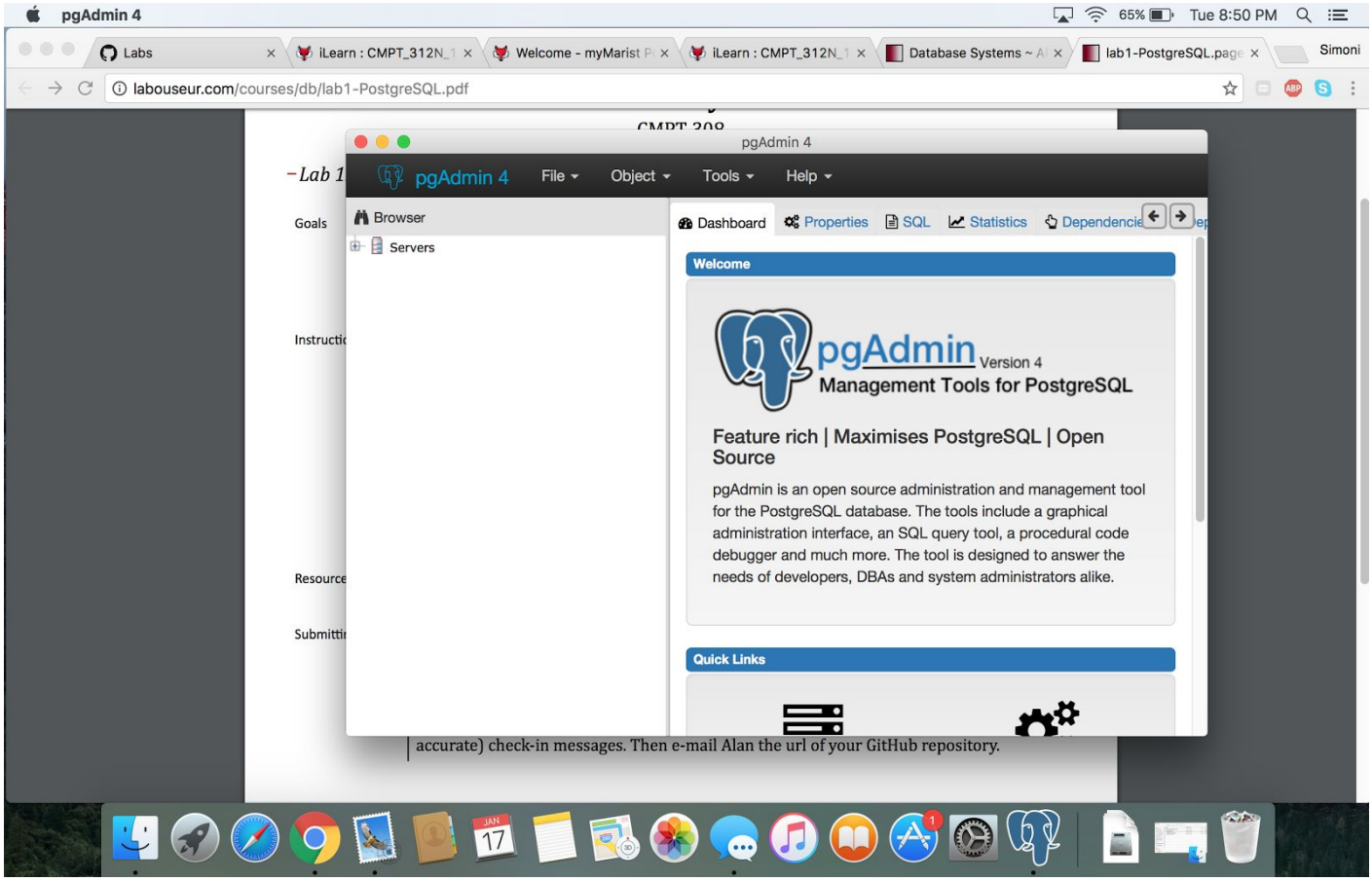
Simoni Handoo
01/20/2017
LAB 1

1) Downloading PostGres: On the top right corner it says “Simoni”- proves that it is a screenshot taken from my laptop.



2) Short Essay:

“Data” is very ambiguous- most people interchangeably use “data” and “information”, when in fact they are both two different terms. Data is raw, context-less, and unorganized facts that need to be processed. Data can be completely useless and seemingly random until it is organized and given some context. “Information” on the other hand is useful and usually formatted in a manner that allows it to be read by humans.

Data + Context = Information

The formula above processes Data into Information. The Data is now processed, organized, and structured. It is presented in a given context so as to make it useful. Data without context would be useless. An example of a database in use could be Marist College’s and each of its student’s test score. Marist College has a database that monitors each of the students’ finals and midterm grades. Data, in this case, would simply be a bunch of numbers (grades) associated with a student. To turn this data into information, an average score of the class could be derived from the given data. This would turn the data into information.

Example of Data:

Joe, Smith, 3399 North, NY, 12601, 8137196378

Simoni Handoo
01/20/2017
LAB 1

Example of Information:
Joe Smith
3399 North,
NY 12601
813-719-6378
As you can see from above, the data is meaningless and not very useful to us. The same data, when broken down into readable text by adding preciseness and context, becomes much more useful and allows you to identify that the contact information is in fact of Joe Smith. Information is therefore more valuable and much more reliable than data.

3) Short Essay:

a) Hierarchical Data Model: The hierarchical data model organized data into a tree-like structure. The data is then stored as records which are connected to one another via links. Hierarchical databases were popular in early database design, in the era of mainframe computing. It has a one-to-many relationship. Hierarchical models make the most sense where the primary focus of information gathering is on a concrete hierarchy. An example would be a list of business departments. There are, however, problems/shortcomings attached with this data model. First of all, you need to know the structure of the model in order to navigate properly. Navigation can be tricky because of its strict ownership to member connection. Secondly, it is difficult to extend the data model since everything is put into a hierarchy. It would be difficult/impossible to find a way to add an item that does not have a parent.
b) Network Data Model: The network data model is a database model that allows multiple records to be linked to the same owner profile. It has a many-to-many relationship. A network data model has good flexibility among the information files because of the multiple relationships amongst the files. Each table in a network data model is called an “entity”. The “Inventory” table for each model has a special name, it is called an “Associative Entity.” There are, however, shortcomings in this model with relation to the relational data model. All the records in a network model are maintained using pointers and hence the whole database structure becomes quite complex. Also, the insertion, deletion and updating of each record requires large number of pointer adjustments.
c) XML as a model for Data storage: I think that XML would not be a good fit for data storage. First of all, XML is a programming language, so it should be avoided in database use. Second of all, to make any sorts of changes, programming would be required. Programming is the number one source of bugs, and debugging would further take much longer. Also, due to XML’s nested tag structure, it is impossible to determine where in the file a certain value is stored without walking the entire document tree, which slows down the time complexity. A relational database has indexes, and looking up a value in an index, even with a primitive binary-search implementation, is a single look-up.