Database Presentation and Final Exam for CS493E Concurrency.

Database presentation will be done on Monday and Tuesday of DeadWeek in my office. As we have discussed, your program will need to perform in accordance with the specs on eCampus. At that time, you will give me a demo of your database program and be prepared to discuss your code and answer question about your implementation. In particular, I am interested in a rational mechanism using the AKKA actor library to deal with any concurrency issues that you feel may occur. This is a core issue in the course and so you will definitely want to have this well thought out. You may use either Actors, Futures or a combination.

You will also need to submit (by email or you may give me a printed copy at the time of your exam) a copy of all your source code. Please only send me your Scala files – I do not need the complete project. I prefer that you submit your code one day before your exam, so that I might look at it before your presentation which will expedite the process. At a minimum, your code is due at the time of your exam.

A couple of notes:

1. The commands Begin, End, and Pause will not be used.

2. Instead, all communication with your database will be via telnet on the demo machine (you may bring your laptop to demo if you wish and this will give you an environment that you can test ahead of time. If you wish to use my desktop, you may bring your code on a memory stick or you may email me your code ahead of time with an indication that you will be using the attached code for your project.

3. Telnet. I have included two files with a working implementation of an Actor based telnet communication system. You can run this program using the main method in Database.scala. You are free to adapt any of this code for your project.

   You will notice that, once running, you can communicate with the program through the command line using telnet and any of the 3 ports I listed in the program (ie: 31733, 29001, 43061). Please use these exact port numbers. This will be the only way to communicate with the database and allows for 3 concurrent users to simultaneous make commands. So, once your program is running, the user will be able to open a command line and type “telnet localhost 31733” (substituting either of the other ports if desired). Once connected, the user can then pass commands via the command line. During your test, we will likely open 3 command line interfaces and by doing so we can definitely communicate concurrently.

   The telnet code I have provided is fairly primitive and hopefully will be self-explanatory. I have added a number of comments which are intended to make its function clearer, but which you will not necessarily want to be part of your final program, so please modify it as you wish. In particular, Database.scala is intended to be merely a plumbed out skeleton and should be replaced by your own code.

I have included a list of possible times for your presentation. Please email me back with several times that will work for you. Times will be awarded via a FIFO algorithm ha ha. At that time we will run your program and then discuss your implementation. Please be punctual as the schedule is quite tight. Obviously, there will be no class on Monday of Dead Week.

A successful presentation will function properly and will have a well thought out and functioning approach to any concurrency issues which demonstrates mastery of the material covered in this course. If either of these is lacking, you will receive appropriate feedback, and you will have a SECOND chance to demo on Saturday morning of Dead Week. If you miss your first appointment, there will be no second chance.

I have also included a sampling of example commands so there is no confusion about syntax. All commands I provide should work, so that (unlike CS210) the emphasis is not on syntax and error handling. That being said, this is a senior level programming course for CS majors, so I fully expect your program to function properly, to be written in Scala, and to have a logical, implemented mechanism for dealing with concurrency using the Actor model.

Some syntactically correct commands:

```
 print dictionary;
print emp;
exit;
define table emp having fields ( a integer, b real, c date, e boolean, d varchar);
delete emp where sal > 2100.0;
update emp set age = 24 where sal > 2100.0;
select emp where sal > 2100.0;
insert (22, 333.0, '12/01/2012', false, 'hello world') into emp;
join a and b;
union a and b;
minus a and b;
intersect a and b;
minus intersect a and b and join d and e;
order minus a and b by sal;
```

Test times: Please email me back with several times that work for you. These will be assigned on a first come basis:

Mon, Dec 3	12:30:00 PM	09:30:00 AM	01:30:00 PM
09:00:00 AM	02:00:00 PM	10:00:00 AM	02:00:00 PM
09:30:00 AM	02:30:00 PM	10:30:00 AM	02:30:00 PM
10:00:00 AM	04:00:00 PM	11:00:00 AM	03:00:00 PM
10:30:00 AM	04:30:00 PM	11:30:00 AM	03:30:00 PM
11:00:00 AM		12:00:00 PM	04:00:00 PM
11:30:00 AM	Tues, Dec 4	12:30:00 PM	04:30:00 PM
12:00:00 PM	09:00:00 AM	01:00:00 PM	05:00:00 PM