CSE 414 Final Exam

June 10, 2013

Name _____

Question 1	/ 28
Question 2	/ 14
Question 3	/ 15
Question 4	/ 12
Question 5	/ 12
Question 6	/ 12
Question 7	/ 12
Question 8	/ 15
Question 9	/ 8
Question 10	/ 12
Total	/ 140

The exam is open textbook but otherwise you may not use any materials other than one sheet of letter-size (or smaller) paper with hand-written notes on one or both sides plus your hand-written notes from the midterm. No computers, electronics devices, phones of the smart or not-so-smart variety, telegraphs, telepathy, smoke signals, magic spells, or other contraptions permitted.

The exam lasts 110 min. Please budget your time so you get to all questions.

Please wait to turn the page until everyone has their exam and you are told to begin.

Relax. You are here to learn.

Question 1. (28 points) SQL. The following database contains information about actors, plays, and roles performed.

Actor(actor_id, name, year_born)
Play(play_id, title, author, year_written)
Role(actor_id, character_name, play_id)

Where:

- Actor is a table of actors, their names, and the year they were born. Each actor has a unique actor_id, which is a key.
- Play is a table of plays, giving the title, author, and year written for each play. Each play has a unique play_id, which is a key.
- Role records which actors have performed which roles (characters) in which plays. Attributes actor_id and play_id are foreign keys to Actor and Play respectively. All three attributes make up the key since it is possible for a single actor to play more than one character in the same play.

(a) (8 points) Write the SQL statements that define the relational schema (tables) for this database. Assume that actor_id, play_id, year_born, and year_written are all integers, and that name, title, author, and character_name are strings. Be sure to define appropriate keys and foreign key constraints.

Actor(actor_id, name, year_born)
Play(play_id, title, author, year_written)
Role(actor_id, character_name, play_id)

Question 1. (cont) (b) (6 points) Write a SQL query that returns the number of actors who have performed in three or more different plays written by the author "August Wilson"

(c) (6 points) Write a SQL query that returns the names of all actors who have performed some play by the author "Chekhov" and have never performed in any play written by author "Shakespeare". The list should not contain duplicates but does not need to be ordered.

Actor(actor_id, name, year_born)
Play(play_id, title, author, year_written)
Role(actor_id, character_name, play_id)

Question 1. (cont.) (d) (8 points) The following query returns information about all persons who have acted in a play that they have written:

```
SELECT a.name, p.title, r.character_name
FROM Actor a, Play p, Role r
Where a.name = p.author AND a.actor_id = r.actor_id AND p.play_id = r.play_id
```

Give a relational algebra query plan drawn as a tree that correctly computes this query.

Question 2. (14 points) Bert the Payroll Guy is about to retire after 40 years and it's time to replace his manual time card system with some sort of computerized database. You have been asked to come up with the database design. As best we can tell, the time card system has the following properties:

- A *timecard* contains hours worked and date submitted
- Each *timecard* is associated with exactly one *employee*
- Each *timecard* has a unique id
- Each *timecard* has a status: approved, not approved, or pending (not examined yet)
- Each *employee* has a name, address, and a unique id
- Each *employee* submits a time card every pay period. i.e. In 1 year, they will submit multiple time cards
- Each *employee* is associated with exactly one *manager*
- Each *manager* is also an *employee*
- Each *manager* is in charge of one or more employees
- Each *manager* approves time cards for one or more employees

Draw an ER diagram that captures this information.

Question 3. (15 points) Consider the relation with schema R(A,B,C,D,E,F) and the following functional dependencies (FDs):

A → BC D → AF

(a) (7 points) What are the keys and superkeys of this relation? (Recall that a key is a minimal superkey.) Justify your answer(s) by showing the closures that are involved, and be sure to clearly label which superkeys are also keys.

(question continued on next page)

Question 3. (cont.) Relation R(A,B,C,D,E,F) and FDs A → BC and D → AF

(b) (8 points) Is relation R in BCNF? If it is, explain why it is. If it is not, explain why not and give a decomposition of R into a collection of relations that are in BCNF.

Question 4. (12 points) Cost estimation. Suppose B(R) = 5000, T(R) = 250000, B(S) = 1000, T(S) = 4000, and M = 1200.

(a) (6 points) What is the expected cost of performing a nested loop join between R and S? If you have any choices in how to carry out the join operation, pick the strategy with the lowest cost – but be sure it is a nested loop join. Briefly explain your strategy (pseudo-code would be helpful) and show enough work so we can understand how you arrived at your answer.

(b) (6 points) Is it possible to perform a hash join between R and S, given the above values for B(R), B(S), and M? If so, describe the high-level algorithm steps and compute the cost of the join. If it is not possible to perform a hash join under these circumstances, explain why not.

Question 5. (12 points) Consider an XML document containing information about job postings and job applications. The postings are grouped by company, and applications are listed under postings. An application contains only the applicant's name. For example:

```
<jobs>
  <company>
    <name> MicroScience Corp. </name>
    <posting>
      <jobtitle> sales rep </jobtitle>
      <salary> 30000 </salary>
      <application> Mark </application>
      <application> Lucy </application>
      <application> Frank </application>
    </posting>
    <posting>
      <jobtitle> technology consultant </jobtitle>
      <salary> 80000 </salary>
      <application> Lucy </application>
      <application> Fred </application>
      <application> Mark </application>
      <application> Betty </application>
    </posting>
  </company>
  <company>
    <name> MacroConsulting Inc. </name>
    <posting>
      <jobtitle> technology consultant </jobtitle>
      <salary> 40000 </salary> <application> Frank </application>
    </posting>
    <posting>
      <jobtitle> debugger </jobtitle>
      <salary> 20000 </salary>
    </posting>
    <posting>
      <jobtitle> programmer analyst </jobtitle>
      <salary> 35000 </salary>
      <application> Lucy </application>
      <application> Mark </application>
    </posting>
  </company>
</jobs>
```

Answer questions about this document on the next page. You can remove this page from the exam if you wish.

Question 5. (cont) (a) (6 points) For each of the XPath expressions below, indicate how many answers it will return on the example XML document on the previous page. For example, if the XPath expression is

```
/jobs/company[name/text()='MacroConsulting Inc.']/posting
```

then your answer should be 3. Your answer to each part should be only a number.

(i)
```
//salary
```

(ii)
```
/jobs/company/posting[salary/text()>50000]//application
```

(iii)
```
/jobs/company[posting/salary/text()>50000]//application
```

(b) (6 points) Write an XQuery expression that returns the names of all applicants who have submitted applications to at least two companies. Your query should return a list of <name> elements inside a root element <applicants>, and each element should be included only once. For example, if your query were run on the XML document shown above, it should return

```
<applicants>
<name>Mark</name>
<name>Lucy </name>
<name>Frank </name>
</applicants>
```

Question 6. (12 points, 6 each) Serializibility. For each of the following transaction schedules, draw the precedence (conflict) graph and decide if the schedule is conflict-serializable. If the schedule is conflict-serializable, give an equivalent serial schedule (you just need to list the order of transactions, not all the individual read-write operations, although you can give the full schedule if it is helpful). If the schedule is not conflict-serializable, explain why not.

(a) r1(A); r2(A); w2(A); r3(B); r2(B); w3(B); w2(B); w1(A)

(b) r1(A); r2(A); r3(B); w1(A); r2(C); r2(B); w2(B); w1(C)

Question 7. (12 points) Locking. In an attempt to guarantee conflict-serializable execution of transactions we introduced the notion of locks. The idea is that transaction *i* should perform a lock operation Li(A) on element A before reading or writing that element, then perform an unlock operation Ui(A) when it is done. But we discovered that this rule was not sufficient to guarantee a serializable schedule and we needed to use the more complex two-phase locking algorithm (2PL) instead.

(a) (6 points) Give an example showing how the use of simple locks without 2PL is not enough to guarantee conflict serializibility.

(b) (6 points) What is two phase locking (2PL) and how does it solve the problem(s) you identified in part (a) above?

Question 8. (15 points) Map-Reduce. We have some very large log files storing information about the traffic on a computer messaging service. The entries in the `Listens(receiver, sender)` log record each pair of users where one receives messages sent by the other. For example, the entry (Alice, Bob) means that Alice receives all messages sent by Bob. Each receiver may listen to messages sent by many senders, and each sender may have many receivers. Users may receive their own messages, i.e., the entry (Pat, Pat) means that all messages sent by Pat are also received by Pat.

The other log, `Message(sender, contents)`, contains each message sent on the system and the name of the sender.

To simplify the problem you can assume that the Listens log is a complete list of the (receiver, sender) pairs during the entire time that all the messages were sent, and that its contents did not change. You may also assume there are no duplicate entries in either log.

Describe a sequence of one or more map-reduce jobs (not Pig programs) to count the number of messages received by each user. The output of the final job should contain one entry for each user and the number of messages received by that user, i.e., if (Chris, 4217) appears in the final output, it means that 4217 messages were received by Chris.

You need to clearly describe the (key, value) pairs that are input to and output from each map and reduce phase of the map-reduce job(s) needed. But the exact format is up to you – it can be pseudo-code or pseudo-java. But you need to clearly describe the input and output (key, value) pairs from each map and reduce stage, and explain how the output of each map or reduce stage is computed from its input. If it takes more than one map-reduce job to compute the final result you should show how the output of each job is used as input to subsequent ones.

(more space is available on the next page for your answer if needed)

Question 8. (cont.) Additional space for your map-reduce algorithm if needed.

Question 9. (8 points) The promised "what does this mean" question. ACID is a collection of four properties provided by database transactions. For each of the four, give a one- or two-sentence definition of what that term means. Please be very brief, but precise.

Atomic:

Consistent:

Isolation:

Durable:

Question 10. (12 points) Many of the no-SQL file systems that support large-scale parallel execution guarantee "eventual consistency" rather than the ACID properties.

(a) (6 points) What does this mean? Explain briefly how this can produce different results compared to a parallel SQL system that guarantees the ACID properties. A very short example might be helpful.

(b) (6 points) Why is this done? Give a technical reason why no-SQL systems find it useful to provide eventual consistency rather than ACID guarantees.