CPSC 310: Database Systems / CSPC 603: Database Systems and Applications
Final Exam
Fall 2005

Name: _____

Instructions:

1. This is a closed book exam. Do not use any notes or books, other than your three 8.5-by-11 inch review sheets. Do not confer with any other student. Do not use any computer equipment.
2. Show your work. Partial credit will be given. Grading will be based on correctness, clarity and neatness.
3. I suggest that you read the whole exam before beginning to work any problem. Budget your time wisely—according to the point distribution.
4. There are 6 questions worth a total of 100 points, on 8 pages (including this page).

DO NOT BEGIN THE EXAM UNTIL INSTRUCTED TO DO SO. GOOD LUCK!

Please sign the academic integrity statement:

"On my honor, as an Aggie, I have neither given nor received unauthorized aid on this academic work. In particular, I certify that I have not received or given any assistance that is contrary to the letter or the spirit of the guidelines for this exam."

Signature: _____

References for these problems:

- web site for our textbook
- *Teach Yourself SQL in 10 Minutes*, Ben Forta, Sams Publishing, Indianapolis, IN, 2004.
- *Database System Concepts*, A. Silberschatz, H. Korth and S. Sudarshan, McGraw-Hill, 2006.

1. (15 pts total)

(a) (6 pts) Consider the following sequence of log records for undo logging:

```
<START T>
<T,A,10>
<START U>
<U,B,20>
<T,C,30>
<U,D,40>
<COMMIT U>
<T,E,50>
<COMMIT T>
```

Suppose the last log record that appears on disk at the time of a crash is <COMMIT U>. What will the recovery manager do to recover from this crash, in terms of updates to the disk and to the log?

(b) (9 pts) Consider the following sequence of log records for undo/redo logging:

```
<START S>
<S,A,60,61>
<COMMMIT S>
<START T>
<T,A,61,62>
<start nonquiescent checkpoint here>
<START U>
<U,B,20,21>
<T,C,30,31>
<START V>
<U,D,40,41>
<V,F,70,71>
<COMMIT U>
<T,E,50,51>
<COMMIT T>
<V,B,21,22>
<COMMIT V>
```

What is the initial state of the database (3 pts)? Which transactions must be indicated in the checkpoint start (3 pts)? Recall that there is flexibility in when the corresponding end checkpoint record occurs. Circle each log record that can be immediately followed by the corresponding end checkpoint record (3 pts).

2. (20 pts total) Conflict-serializability.

(a) (3 pts) What is the definition of a schedule being *conflict-serializable?*

(b) (3 pts) What is the definition of a *precedence graph* (also called a conflict graph) for a schedule?

(c) (4 pts) Explain how to use a precedence graph to determine if a schedule is conflict-serializable.

The rest of this problem refers to the following schedule S:

$$r_1(A); r_2(A); r_3(B); w_1(A); r_2(C); r_2(B); w_2(B); w_1(C);$$

(d) (5 pts) Draw the precedence graph for S.

(e) (5 pts) Is S conflict-serializable? If so, then give the equivalent serial schedule. If not, explain why not.

3. (20 pts total) Two-phase locking.

(a) (4 pts) What are the rules for two-phase locking?

(b) (3 pts) Circle all of the following properties that two-phase locking guarantees concerning the schedules that it generates:

- serial
- serializable
- recoverable
- conflict-serializable
- avoid cascading rollback
- strict

(c) (3 pts) What are the rules for *strict* two-phase locking?

(d) (4 pts) Circle all of the following properties that strict two-phase locking guarantees concerning the schedules that it generates:

- serial
- serializable
- recoverable
- conflict-serializable
- avoid cascading rollback
- strict

(e) (6 pts) Consider these two transactions:
T_1: $r_1(X); w_1(Y);$
T_2: $r_2(Y); w_2(X);$

For each of the following schedules of transactions T_1 and T_2, indicate whether it can be generated by both two-phase locking (2PL) and strict two-phase locking (S-2PL), by 2PL only, by S-2PL only, or by neither.

- $\ell_1(X); r_1(X); \ell_1(Y) u_1(X) \ell_2(X); w_1(Y); u_1(Y); \ell_2(Y); r_2(Y); w_2(X); u_2(Y); u_2(X);$ _____
- $\ell_2(Y); r_2(Y); u_2(Y); \ell_2(X); w_2(X); u_2(X) \ell_1(X); r_1(X); \ell_1(Y); w_1(Y); u_2(X); u_2(Y);$ _____
- $\ell_1(X) \ell_1(Y) r_1(X); w_1(Y); u_1(X); u_1(Y); \ell_2(Y); \ell_2(X); r_2(Y); w_2(X); u_2(Y); u_2(X);$ _____

4. (15 pts total) Suppose there are three transactions, T_1 with timestamp 100, T_2 with timestamp 200, and T_3 with timestamp 300. Suppose that at some point in time, the following queue of lock requests is waiting to be processed (notation $L_i(X)$ means transaction T_i requests a lock on database element X):

$$L_1(A); L_2(B); L_3(C); L_1(B); L_2(C); L_3(A);$$

For simplicity, we make the following assumptions:

- If a transaction dies or is wounded, it immediately gives up its locks and waits D time before restarting. D is chosen to be large enough so that any transaction that is active at time t has completed (committed or aborted) by time $t + D$. So all of the dead/wounded transaction's lock requests go to the end of the lock request queue.
- If a transaction ever gets all the locks it wants, then it completes its work, commits and releases its locks in a negligible amount of time (i.e., imagine it is instantaneous).
- When a lock is released, it is immediately given to a transaction waiting for it (in first-come first-served manner).

(a) (7 pts) Assume that the wait-die algorithm is used to handle lock requests and avoid deadlock. In what order do the transactions T_1, T_2 and T_3 commit?

(b) (8 pts) Assume that the wound-wait algorithm is used to handle lock requests and avoid deadlock. In what order do the transactions T_1, T_2, and T_3 commit?

5. (6 pts total) SQL.

Consider the following database schema for a simplified order entry system:

```
Customers(cust_id,cust_name,cust_address)
Vendors(vend_id,vend_name,vend_address)
Orders(order_num,order_date,cust_id)
Products(prod_id,vend_id,prod_name,prod_price,prod_desc)
OrderItems(order_num,order_item,prod_id,quantity,item_price)
```

(a) (3 pts) Write an SQL query to return the name of all customers who ordered the product with id "RGAN01". In your query, use subquery(ies), but no join(s).

(b) (3 pts) Redo part (a) but this time use join(s) instead of subquery(ies).

6. (24 pts total) Short answer.

(a) (3 pts) Suppose 10 tuples of R can fit in one block, R has 100,000 tuples, and there are among these tuples 20,000 distinct tuples. We wish to compute $\delta(R)$ (duplicate elimination) in one pass. How many main memory buffers are needed to do this? Assume each main memory buffer is the same size as a disk block.

(b) (3 pts) What is the *elevator algorithm* and what is it good for?

(c) (3 pts) *Multiple choice:* Consider a relation R that always contains exactly one tuple. Is R in Boyce-Codd Normal Form (BCNF)? Circle one:

- yes
- no
- can't tell without seeing the actual schema and FD's
- can't tell without seeing the actual data
- can't tell without seeing the actual schema, FD's, and data

(d) (4 pts) Suppose we have a relation $R(A, B, C, D, E)$ and functional dependencies $A \to D$, $B \to C$, $D \to E$, $CE \to B$. What are the key(s) of R?

(e) (3 pts) Let $R(x, y, z)$ be a relation. Which *one* of the following relational algebra formulas is an identity (true of all such relations R)? Circle it.

- $\pi_{x,y}(\delta(R)) = \delta(\pi_{x,y}(R))$
- $\pi_{x,y}(R) \bowtie \pi_{y,z}(R) = R$
- $\gamma_{x,y,z}(R) = \delta(R)$

(f) (2 pts) What does it mean to *swizzle a pointer* and why is it important to do so?

(g) (2 pts) What is the main advantage of using a B-tree index over using a sequential index?

(h) (4 pts) Suppose we have these relations in a database schema:

```
Branch(branch_name,branch_city,assets)
Account(account_number,branch_name,balance)
Customer(customer_name,account_number)
```

Consider the following initial logical query plan to find the name of all customers that bank at the Snook branch and have a balance less than 1000. Draw an optimized version of this logical query plan, in which you have tried to reduce the size of intermediate relations as much as possible.