CS157b Final Spring 2020

Name:
StudID:

Instructions:

1. This final is due midnight, May 20.
2. To complete the final, print it out, fill in your answers on the final, and scan it back into a file Final.pdf where the total size is less than 10MB.
3. If you don't have a printer, copy and paste each problem into a word processor document. Then after each problem write your solution. Make a less than 10MB Final.pdf file of the result and submit that.
4. Use the same submit mechanism as for the homeworks to submit your completed final.
5. Each problem on this final is worth the same amount (3pts).
6. If you have a question on the interpretation of a problem on the final, you can email me at chris@pollett.org.
7. Due to the coronavirus this is an open book, open internet final.
    a. What that means is that you can consult any static (on the order of static for weeks) source of information related to the final material.
    b. You cannot directly or indirectly ask another person how to do any problem off the final.
    c. To receive credit on problems that make use of your personal information, you need to have correctly filled in that personal information.
    d. When you submit your completed final, you are asserting all of the work in the final is your own.

Problem	Grade	Problem	Grade
1		6	
2		7	
3		8	
4		9	
5		10	

1. Write a simple transaction using SQL and Java (make sure you have the SQL to start and end your transaction as worth 0.5pts each) connecting to a BANK database of the DBMS of your choice (0.5pts), which sets the isolation level to SERIALIZABLE (0.5), reads from the ACCOUNT table the BALANCE of the user with ID your student ID (1pt), and decreases this value by 40 (1pt).

2. Let *X* be the leftmost non-zero digit of your student ID +10. For example, if your ID was 000056921, this would be 5+10 = 15. Write a small program to generate *X* many 4 digit binary numbers (0.5pts to show code). Insert these numbers in the order they were generated, showing work for each insertion, into an extensible hash table with a bucket size of 2 (2.5pts).

3. Let X be the left-most four non-zero digits of your student ID. Let Y be the right-most four non-zero digits of your student ID. For example, if your ID was 04547300, then X would be 4547 and Y would be 5473. Suppose we have two tables R with X rows, and S with Y rows that share a single attribute Z. S has an index on Z and V(S,Z)=10. (a) Estimate the cost of performing an index join using the technique from class (1pt). (b) Give an example with explanation of a query optimization heuristic discussed in class involving projections (1pt). (c) Give an example with explanation of a query optimization heuristic discussed in class involving selections (1pt).

4. Let X be the maximum of the rightmost non-zero digit of your student ID and 5. For example, if your ID was 74547300, then X would be max(3,5) = 5. Give sufficient conditions for recover from a system failure for each of the following kinds of logging: (a) undo logging (0.5pts), (b) redo logging (0.5pts). Suppose we are doing undo/redo logging with checkpointing and archiving. Give an example log file with explanation (0.5pts for explanation and overall log file correction) showing how the non-quiescent archive process works. This log file should have $X - 1$ active transactions at the time the checkpoint occurs (0.5pts). Exactly one of these transactions aborts during the checkpoint, all others commit (0.5pts). One new transaction starts during the checkpoint (0.5pts).

5. Let X and Y be the rightmost two nonzero digits of your student ID. So if your ID was 04547300 then $X = 7$ and $Y = 3$. Give examples using transactions T_X and T_Y (argue why your examples work to receive credit) of each of the following (1pt each): (a) a schedule achievable by 2PL that is not possible with timestamping, (b) a recoverable schedule that is not strict 2PL, and (c) a serializable schedule which is not conflict serializable.

6. Explain the SQL CUBE and ROLLUP operations (0.5pts). Give examples to illustrate how they differ (0.5pts). Let Y be 4 + mod 2 of the rightmost nonzero digit of your student ID. So if your ID was 74547300 then $Y = 4 + 1 = 5$ as 3 mod 2 is 1. Come up with a list of $Y + 2$ household items to hoard. For example, you might hoard toilet paper. Make a table consisting of Y hoarders by these $Y + 2$ items. Roll a six-sided dice for each pair (user i, item j), if the dice comes up 1 or 2, say that item is hoarded by that user. Submit the table your got (0.5pts). Show how the A Priori algorithm would operate on this table with support threshold 3 (1.5pts).

7. Suppose you are trying to develop a travel website capable of booking flights either of two airlines. If *x* is your first name (I.e., for your professor, x=Chris) and *y* is your last name (I.e., for your professor y=Pollett), then one is called *x* Air and the other is called *y* Airlines. On *x* Air tickets are associated with a seat on a plane, for example, seat 7C, but on *y* Airlines, seating is first come first serve -- a ticket has a number associated with it which is less than the plane capacity. Suggest DB schemas (in SQL) the two airlines might each be using (1pt). Your website should support queries by users for flights with available seats at a given time (airline unspecified). Explain how this might look in terms of a mediator (0.5pts), give appropriate wrapper queries (in SQL) for each airline (0.5pts), and explain how query results might be integrated into a single returned result set (1pt).

8. Give an example of a kd-tree with more than 4 leaves that queries at least two distinct attributes and which queries at least one of these twice. Explain how a look up could be performed on this tree.

9. Define the following distributed locking concepts: (a) centralized locking, (b) Lock Coordinator, (c) primary copy locking. Suggest an algorithm to avoid deadlocks in the distributed setting.

10. Give the parallel algorithm from class for computing the join of two relations $R \bowtie S$. Give its cost on N machines. Give the distributed algorithm from class for computing $R \bowtie S$ on two machines.