Final Examination

Due: In 120 minutes.

Course: CS 5703 Spring 1995.

Instructions:

1. There are five questions in this paper. All questions have three parts each.
2. Please use the space provided (below the questions) to write the answers. Show the reasoning behind the answers. The process of arriving at answers carries as much weight as the correctness of answers.
3. Budget your time to answer various questions to avoid spending too much time on a particular question. ATTEMPT QUESTION 1 AFTER FINISHING OTHER QUESTIONS.
4. It is an open book examination. You may need to consult the book for some questions.

NAME:	
UMN Identification Number:	
Account:	
Machine:	
Classification:	day / extension / NTU / UNITE / Rochester

Score Table	
Question	Score
Q1a	
Q1b	
Q1c	
Q2a	
Q2b	
Q2c	
Q3a	
Q3b	
Q3c	
Q4a	
Q4b	
Q4c	
Q5a	
Q5b	
Q5c	

Q1a. Consider the management of an interest-bearing checking account at a bank. The standard transaction is to write a check debiting the account or to deposit (or credit) funds in the account. The bank also has an Automatic Teller Machine (ATM) acting as a terminal; that is, the input customer makes at ATM is a message to the transaction processor (TP) software, and the final message from the TP makes the ATM dispense cash. The debit and credit transactions can be performed via ATM.

Q1a. (4 points)

Consider the use of paper-based checkbooks. Match each of the following scenarios to an ACID property which may be violated due to the scenario.

ACID Property	Scenario
	S1: Bank pays you interest after you write a check but before it is cashed.
	S2: A check gets lost in mail and Bank database is not updated.
	S3: You lose your checkbook and balance.
	S4: You write a check for more than the current account balance.

Q1b. (4 points)

Consider the use of ATM for debit and credit transactions. Assume that ATM has a local controller (say a small computer system) distinct from the host (say another large computer system) of TP process. Consider the following system features: (f1) small durable storage at ATM controller, (f2) distributed two phase commit, (f3) cash dispense action is testable (i.e. if cash was dispensed or not), via electromagnetic counters, (f4) semi-join facility.

For each fault, list the relevant system features needed to guarantee atomicity. Note that some faults can not be handled by the features f1, ... , f4; while recovery from other faults may require more than one feature.

Feature	Faults
	(c1) crash of the system running TP
	(c2) crash of the ATM
	(c3) crash of communication link between ATM and TP
	(c4) Paper jam: A bill gets stuck in dispenser and the client tears it off.

Q1c. (4 points)

Suppose a system has perfect hardware (e.g. no power supply/CPU/disk failures), a perfect environment (e.g. no fires/floods), and perfect operations (no rollbacks), so that only failures are due to software. *Does the recovery system still need to maintain logs and distinguish between volatile and durable memories?* Justify your answer briefly.

Q2. Let Wi(j) denote a write operation by Transaction Ti on item j in the database. Let Ri(j) denote a read operation by Transaction Ti on item j in the database. Consider a database with items X, Y, Z and transactions T1, T2, and T3. Each transaction commits right after its last operation is completed.

Transaction T1: R1(X), W1(X), R1(Y), W1(Y)
Transaction T2: R2(Z), R2(X), W2(X), R2(Y), W2(Y)
Transaction T3: R3(Y), R3(Z), W3(Y), W3(Z)
S1:R3(Y),R3(Z),R1(X),W1(X),W3(Y),W3(Z),R2(Z),R2(X),W2(X),R1(Y),W1(Y),R2(Y),W2(Y)
S2:R3(Y),R3(Z),W3(Y),W3(Z),R2(Z),R2(X),W2(X),R2(Y),W2(Y),R1(X),W1(X),R1(Y),W1(Y)
S3:R2(Z),R2(X),W2(X),R3(Y),R3(Z),R1(X),W1(X),W3(Y),W3(Z),R2(Y),R1(Y),W1(Y),W2(Y)

Time-stamp for each data-item is initially set to 0. The time-stamps for the transactions are positive (> 0) and are in the order of TS(T3) < TS(T1) < TS(T2). In basic two-phase locking relevant BINARY LOCK(A) is acquired by a transaction T just before first read/write operation on data-item A and released after all lock operations as well as the last operation on A in T. In strict two-phase locking, all LOCKS are released just after the last operation of the transaction. In conservative two-phase locking, all LOCKS are acquired before the first operation in the transaction.

Q2a. (5 points)

Draw conflict graphs and state if S1, S2 and S3 are conflict-serializable.

Q2b. (5 points)

Determine if S1, S2 and S3 are consistent with TIME-STAMP ORDERING based transaction scheduling. Justify your answer. (To save time you may work with the edges of conflict schedule graph rather than history of time-stamps for data-items.)

Q2c. (5 points)

Determine if S1, S2 and S3 are consistent with BASIC 2-phase locking based transaction scheduling. Justify your answer.

Q3. Refer to the schedules S1, S2, and S3 defined in the previous question.

Q3a. (5 points)

Determine if S1, S2 and S3 satisfy STRICT 2-phase locking based transaction scheduling. Justify your answer.

Q3b. (5 points)

Determine if S1, S2 and S3 CONSERVATIVE 2-phase locking based transaction scheduling. Justify your answer.

Q3c. (5 points)

Determine if S1, S2 and S3 may suffer from cascaded rollback during recovery. Justify your answer.

Q4. Consider the log corresponding to a particular schedule at the point of system crash for the four transactions T1, T2, T3, and T4. S(Ti) denotes start_transaction operation by Ti. C(Ti) denotes commit_transaction operation by Ti. R(Ti, A) denotes read_item(A) operation by Ti. W(Ti, A, AI, BI) denotes write_item(A) operation by Ti with after image AI and before image BI. Field Sr. denotes the log-order of operation. The values for AI depends on update policy and have not been specified. If you like fill in the values for ?x1 .. ?x6 for AIs.

Sr.	1	2	3	4	5	6	7
	S(T1)	R(T1,A)	R(T1,D)	W(T1,D,20,?x1)	S(T2)	Checkpoint	C(T1)
Sr.	8	9	10 11	12	13	14	
	R(T2,B)	W(T2,B,12,?x2)	S(T4)	R(T4,B)	W(T4,B,15,?x3)	S(T3)	W(T3,A,30,?x4)
Sr.	15	16	17	18	19	20	
	R(T4,A)	W(T4,A,20,?x5)	C(T4)	R(T2,D)	W(T2,D,25,?x6)	crash	

Q4a. (5 points)

Suppose we use the IMMEDIATE update protocol with checkpointing. Describe the recovery process from the system crash by filling in the following table.

Whether cascading rollback takes place?	
How should RIU_M (pp 588) be modified for this situation?	
Which transactions are rolled back?	
Which operations in the log are redone?	
Which operations are undone?	

Q4b. (5 points)

Suppose we use the deferred update protocol with checkpointing. Describe the recovery process from the system crash by filling in the following table. (Assume optimistic concurrency control algorithm, i.e. one of T2 and T4 would be rolled back at its commit time.)

Whether cascading rollback takes place?	
How should RDU_M (pp 584) be modified for this situation?	
Which transactions are rolled back?	
Which operations in the log are redone?	
Which operations are undone?	

Q4c. (5 points)

Discuss the factors that do not appear in a centralized system that affect concurrency control and recovery in distributed databases.

Q5. Consider a complex object representation of orthogonal rectangles in a plane (say X-Y plane), with **sides parallel to coordinate axis**. There are three relations:

RECTANGLE(rectangle-name, anEdge),
EDGE(edge-name, aPoint),
POINT(point-name, x-coordinate, y-coordinate).

Example: Unit square is represented in the database as a tuple in the RECTANGLE class with name unit-square.

rectangle(unit-square, edge1), rectangle(unit-square, edge2),
rectangle(unit-square, edge3), rectangle(unit-square, edge4),
edge(edge1, point1), edge(edge1, point2), edge(edge2, point2), edge(edge2, point3),
edge(edge3, point4), edge(edge3, point3), edge(edge4, point1), edge(edge4, point4),
point(point1, 0, 0), point(point2, 1, 0), point(point3, 1, 1), point(point4, 0, 1).

Geometric computations can be represented as queries over this database using nested predicates.

Q5a. (10 points)

Give an expression in SQL (no user defined functions) or in relational algebra using nested predicates for the following queries:- Q1: Print the rectangle-names of all rectangles, which are squares (i.e. their length and width are identical).
DO NOT use any non-relational concepts such as user-defined types, functions etc.

Q5b. (5 points)

Discuss how an object-oriented query language can simplify the representation of queries Q1. Provide examples from Postgres/Illustra to support your answer.

Q5c. (5 points)

How does the concept of an object in the object oriented model differ from the concept of an tuple in an relational model?