CS 245
Final Exam – Winter 2000

This exam is open book and notes. You have 120 minutes to complete it.

Print your name:____

The Honor Code is an undertaking of the students, individually and collectively:

1. that they will not give or receive aid in examinations; that they will not give or receive unpermitted aid in class work, in the preparation of reports, or in any other work that is to be used by the instructor as the basis of grading;
2. that they will do their share and take an active part in seeing to it that others as well as themselves uphold the spirit and letter of the Honor Code.

The faculty on its part manifests its confidence in the honor of its students by refraining from proctoring examinations and from taking unusual and unreasonable precautions to prevent the forms of dishonesty mentioned above. The faculty will also avoid, as far as practicable, academic procedures that create temptations to violate the Honor Code.

While the faculty alone has the right and obligation to set academic requirements, the students and faculty will work together to establish optimal conditions for honorable academic work.

I acknowledge and accept the Honor Code.

Signed:____

Problem	Points	Maximum
1		10
2		10
3		10
4		10
5		10
6		10
7		10
Total		70

Problem 1 (10 points)

Please answer the following questions:

1. Consider a $B+$ tree of order 100. What is the minimum number of keys a non-root leaf node can contain?

   ANSWER: 50

2. Consider the same $B+$ tree of order 100. What is the maximum number of pointers a non-root non-leaf may contain?

   ANSWER: 101

3. Consider an extensible hash table where 100 buckets are actually allocated at this point. What is the size (in number of entries) of the smallest possible directory in this case?

   ANSWER: 128

4. Again consider an extensible hash table where 100 buckets are actually allocated. What is the size (in number of entries) of the largest possible directory in this case?

   ANSWER: Infinity

5. Consider a linear hashing structure with $i = 5$ and $m = 20$, and no overflow buckets. How many disk buckets are actually allocated at this point?

   ANSWER: 21

6. A file contains 1000 blocks, each with 100 records. The file is sorted and its blocks are stored contiguously on disk. The sort key is 4 bytes long, a block pointer is 4 bytes, and a record pointer is 5 bytes. You build a dense one-level index on this file. How small can this index be? Give the size in number of bytes.

   ANSWER: 400,004

7. For the same 1000 block file, you build a one-level sparse index. How small (number of bytes) can this index be?

   ANSWER: 4004

8. Consider a disk with a 10 inch diameter, 7200 RPM speed, 8 surfaces, 1024 cylinders, average seek time of 25ms, 4KB blocks, 2GB usable capacity, and 10% overhead for gaps. How many tracks (total over the entire disk) does this disk have?

   ANSWER: 8192

9. Consider the same disk (10 inch diameter, 7200 RPM speed, 8 surfaces, 1024 cylinders, average seek time of 25ms, 4KB blocks, 2GB usable capacity, and 10% overhead for gaps). What is the average rotational delay to read a block, assuming we do not need to find the start of the track?

   ANSWER: 41.7 ms

10. What structure is better for nearest neighbor queries, a partitioned hash table or a grid file?

    ANSWER: grid file

Problem 2 (10 points)

Fill in the blanks in the following statements.

(a) In a federated database system there is a one to one connection between every pair of databases that need to talk to one another.

(b) A mediator supports a virtual view, or a collection of views, that integrates several sources in much the same way that materialized relation(s) in a warehouse integrate sources.

(c) In the wait-die deadlock prevention scheme, a transaction is allowed to wait for a younger transaction, but not for an older one.

(d) With group commit, every commit record is not immediately flushed to disk, and a transactions is allowed to release locks as soon as its commit record is in the log buffer.

(e) In the SQL standarad, isolation level read committed does not require serializability but does forbid a transaction to read dirty data.

(f) With two phase locking, a transaction cannot request locks after it has released a lock.

(g) A schedule of transactions that obey the strict locking rule is called recoverable.

(h) An action is idempotent if executing it many times has the same effect as executing it once.

(i) In a zipfian distribution the frequency if the i^{th} most common value is in proportion to $1/\sqrt{i}$.

(j) The preservation of value sets assumption states that if we join a relation R with another relation, then an attribute A that is not a join attribute (i.e., not present in both relations) does not lose values from its set of possible values.

Problem 3 (10 points)

Suppose that we run the following transactions using the validation protocol. The following table lists the transactions involved, together with their read and write sets:

Transaction	Read Set	Write Set
T_1	$\{a, b\}$	{b, c}
T_2	$\{a, b, c\}$	{h}
T_3	$\{b\}$	{d, e}
T_4	$\{c\}$	{f, g}
T_5	$\{a\}$	{d, f}
T_6	$\{g\}$	{e, g}

The following sequence of events takes place. (No other transactions run before or concurrently with T_1 ... T_6.)

1. T_1, T_2, T_3, T_4 start (in this order)
2. T_3 initiates validation
3. T_5, T_6 start (in this order)
4. T_1 initiates validation
5. T_5 initiates validation
6. T_4 initiates validation
7. T_2 initiates validation
8. T_1, T_2, T_3 finish (if they were not aborted earlier)
9. T_6 initiates validation
10. T_4, T_5, T_6 finish (if they were not aborted earlier)

In the table on the next page, please write down the outcome of each transaction, either COMMITTED if it was successful at validation, or ABORTED if it was not.

Transaction	Outcome
T_1	committed
T_2	aborted
T_3	committed
T_4	aborted
T_5	aborted
T_6	committed

Problem 4 (10 points)

A multi-granularity hierarchical locking scheme is used in an object oriented database. In particular, the objects for a class C_1 are stored in two pages P_1 and P_2. Objects o_1, o_2 and o_3 are stored in page P_1, while objects o_4 and o_5 are stored in page P_2.

The following table shows the state of the system at a particular time when *four* transactions are active. Each entry identifies the transaction holding a particular lock at this time. For example, Transactions 1 and 2 hold IS locks on class C_1, while transaction 3 holds an IX lock on C_1. Transaction 4 does not hold any locks at this time.

	IS	IX	S	SIX	X
C_1	1,2	3			
P_1	2		1		
o_1			2		
o_2					
o_3					
P_2		3			
o_4					3
o_5					

Using this same table, indicate what are *all* the possible *next* lock actions in this scenario. For example, Transaction 3 could next lock object o_5 in X mode, so the cell o_5, X should have a "3" in it. This same cell could have another number n if a Transaction n could also get this lock. Note that Transactions 3 and n could not both hold the X lock on o_5; an entry with two transactions simply means that one or the other could get the lock next.

Do not forget Transaction 4 which does not currently hold locks, but could acquire one next.

Do *not* show entries that are not useful even though they do not create a conflict. For example, it does not make sense for Transaction 1 to request an S or IS lock on o_2, so there should be no "1" in those entries.

	IS	IX	S	SIX	X
C_1	1,2,4	3,4			
P_1	2,3		1,3		
o_1			2		
o_2			2		
o_3			2		
P_2	1,2	3			
o_4					3
o_5			3		3

Problem 5 (10 points)

The following table describes the undo/redo log found on a system after a failure.

Action id	Action type	Transaction id	Other info
1	trans start	T_1	
2	db write	T_1	
3	trans start	T_2	
4	db write	T_2	
5	start dump	System	T_1 active
6	trans start	T_3	
7	db write	T_3	
8	end dump	System	
9	db write	T_3	
10	trans start	T_4	
11	db write	T_4	
12	db write	T_3	
13	commit	T_4	
14	commit	T_2	
15	start checkpoint	System	T_1, T_3 active
16	db write	T_1	
17	db write	T_3	
18	trans start	T_5	
19	db write	T_5	
20	trans start	T_6	
21	db write	T_6	
22	end checkpoint	System	
23	db write	T_5	
24	db write	T_3	
25	db write	T_1	
26	commit	T_5	
27	db write	T_6	
28	commit	T_3	
29	end of log		

The database dump that occurs between actions 5 and 8 writes a complete copy of the database to tape, to be used in case of media failure.

(a) Assume that at action 29 the system crashes but the database is *not* lost. Assume that a two pass algorithm is run for recovery, where actions are first undone, and then actions are redone. In the two lines below indicate what actions are undone in the first pass, and what action are redone. Write the actions *in the order* they would be undone or redone (left to right). Identify each action by its id given in the first column in the table.

ACTIONS TO BE UNDONE: 27, 25, 21, 16, 2

ACTIONS TO BE REDONE: 17, 19, 23, 24

Explanation : Transactions T_1 and T_6 did not commit whereas transactions T_2, T_3, T_4, and T_5 committed. Therefore, we need to undo the actions of T_1 and T_6 and redo those actions of the other transactions that took place after action 15 (the "start checkpoint" action).

(b) Now assume that the database is lost at action 29. In this case we load the saved database from tape and bring it up to date. In the two lines below again indicate what actions are undone and redone to bring the dumped database up to date.

ACTIONS TO BE UNDONE: 2

ACTIONS TO BE REDONE: (4), 7, 9, 11, 12, 17, 19, 23, 24

Explanation : Once again, we need to undo the appropriate actions of transactions T_1 and T_6 and redo the actions of the other transactions. Action 2 is the only action by T_1 or T_6 that took place before or during the dump. So, that is the only action that needs to be undone. The actions that need to be redone are those performed by transactions T_2, T_3, T_4, or T_5. If we assume that a checkpoint is taken before the dump begins, then action 4 need not be redone as it would have been reflected on disk (and hence in the dumped image). If we assume that a checkpoint is not taken, then action 4 may or may not be reflected in the dumped image - so we must redo that during recovery.

Problem 6 (10 points)

Consider two relations $R(a, b, c, d)$ and $S(d, e)$ with the following statistics:

- $T(R) = 100$; $V(R, a) = 100$; $V(R, b) = 10$; $V(R, c) = 1$; $V(R, d) = 50$.
- $T(S) = 500$; $V(S, d) = 30$, $V(S, e) = 100$.

(a) Estimate the number of tuples in $\sigma_{b=25}(R)$.

ANSWER: T(R)/V(R,b) = 10

(b) Estimate the number of tuples in $\sigma_{c=30}(R)$.

ANSWER: T(R)/V(R,c) = 100

(c) Estimate the number of tuples in $\sigma_{(b=25)\wedge(c=30)}(R)$.

ANSWER: T(R)/(V(R,b) * V(R,c)) = 10

(d) Estimate the number of tuples in $\sigma_{(b>25)}(R)$.

ANSWER: T(R)/3 = 100/3 = 33 (or 34)

(e) Estimate the number of tuples in $\sigma_{(b>25)\wedge(b=11)}(R)$.

ANSWER: 0 (as conditions contradict)

(f) Estimate the number of tuples in $\sigma_{(b=25)\vee(d=13)}(R)$.

ANSWER: T(R)/V(R,b) + T(R)/V(R,d) - T(R)/(V(R,b)*V(R,d)) = 12

(g) Estimate the number of tuples in $R \bowtie S$.

ANSWER: T(R)T(S)/maxV(R,d), V(S,d) = 1000

Problem 7 (10 points)

You have implemented an algorithm for association rule data mining, to find 2-sets. To find the high support 2-sets, you use a hash table with 100 counters. Each occurring pair is hashed, and the counter is incremented. As discussed in class, this scheme may produce false positives, so a second pass is required to discard the false positives.

On a particular run of your algorithm, you have a table with 900 transactions, each of the form (t, p_1, p_2). (Here t is the transaction id, and p_1 and p_2 are the two products bought together.) In the table there are 150 unique pairs occuring. In particular:

- There are 50 pairs that occur in 10 different transactions.
- There are 100 pairs that occur in 4 transactions.

The threshold is use is 9. Thus, the algorithm should eventually identify the 50 pairs that occur 10 times.

The hash function you use is such that:

- The probability that only one unique pair hashes to a given bucket (counter) is 0.5.
- The probability that 2 different pairs hash to the same bucket is 0.5.
- All other cases occur with negligible probability.

How many false positives will be generated by the algorithm in the first phase? Give your answer as the expected number of unique pairs that are false positives. Please explain your answer.

We expect to have $100 \times 0.5 = 50$ buckets with 2 pairs. One of these buckets will have a false positive if a high-frequency and a low frequency pair hash to it. This event occurs with probability $2 \times (100/150) \times (50/150)$ or $4/9$. Thus, the expected number of false positive pairs is $50 \times (4/9)$ or 22.22.

NUMBER OF FALSE POSITIVES: 22.22 pairs