Query Evaluation (20%)

Problem 1. Consider a relation R(a,b,c,d,e) containing 5,000,000 records, where each data page of the relation holds 10 records. R is organized as a sorted file with secondary indexes. Assume that $R.a$ is a candidate key for R, with values lying in the range 0 to 4,999,999, and that R is stored in $R.a$ order. For each of the following relational algebra queries, state which of the following three approaches is most likely to be the cheapest:

- Access the sorted file for R directly.
- Use a (clustered) B+ tree index on attribute $R.a$.
- Use a linear hashed index on attribute $R.a$.

1. $\sigma_{a<50,000}(R)$
2. $\sigma_{a=50,000}(R)$
3. $\sigma_{a>50,000 \wedge a<50,010}(R)$
4. $\sigma_{a\neq 50,000}(R)$

External Sorting (20%)

Problem 2. Consider a disk with an average seek time of 10ms, average rotational delay of 5ms, and a transfer time of 1ms for a 4K page. Assume that the cost of reading/writing a page is the sum of these values (i.e., 16ms) unless a *sequence* of pages is read/written. In this case, the cost is the average seek time plus the average rotational delay (to find the first page in the sequence) plus 1ms per page (to transfer data). You are given 320 buffer pages and asked to sort a file with 10,000,000 pages.

1. Why is it a bad idea to use the 320 pages to support virtual memory, that is, to 'new' $10{,}000{,}000 \cdot 4K$ bytes of memory, and to use an in-memory sorting algorithm such as Quicksort?
2. Assume that you begin by creating sorted runs of 320 pages each in the first pass. Evaluate the cost of the following approaches for the subsequent merging passes:

   (a) Do 319-way merges.

(b) Create 256 ‘input’ buffers of 1 page each, create an ‘output’ buffer of 64 pages, and do 256-way merges.

(c) Create 16 ‘input’ buffers of 16 pages each, create an ‘output’ buffer of 64 pages, and do 16-way merges.

(d) Create eight ‘input’ buffers of 32 pages each, create an ‘output’ buffer of 64 pages, and do eight-way merges.

(e) Create four ‘input’ buffers of 64 pages each, create an ‘output’ buffer of 64 pages, and do four-way merges.

Concurrency Control (20%)

Problem 3. Consider the following classes of schedules: *serializable, conflict-serializable, view-serializable, recoverable, avoids-cascading-aborts,* and *strict.* For each of the following schedules, state which of the preceding classes it belongs to. If you cannot decide whether a schedule belongs in a certain class based on the listed actions, explain briefly.

The actions are listed in the order they are scheduled and prefixed with the transaction name. If a commit or abort is not shown, the schedule is incomplete; assume that abort or commit must follow all the listed actions.

1. T1:R(X), T2:R(X), T1:W(X), T2:W(X)
2. T1:W(X), T2:R(Y), T1:R(Y), T2:R(X)
3. T1:R(X), T2:R(Y), T3:W(X), T2:R(X), T1:R(Y)
4. T1:R(X), T1:R(Y), T1:W(X), T2:R(Y), T3:W(Y), T1:W(X), T2:R(Y)
5. T1:R(X), T2:W(X), T1:W(X), T2:Abort, T1:Commit
6. T1:R(X), T2:W(X), T1:W(X), T2:Commit, T1:Commit
7. T1:W(X), T2:R(X), T1:W(X), T2:Abort, T1:Commit
8. T1:W(X), T2:R(X), T1:W(X), T2:Commit, T1:Commit
9. T1:W(X), T2:R(X), T1:W(X), T2:Commit, T1:Abort
10. T2: R(X), T3:W(X), T3:Commit, T1:W(Y), T1:Commit, T2:R(Y), T2:W(Z), T2:Commit

Crash Recovery (20%)

Problem 4. Consider the execution shown in Figure

1. Extend the figure to show prevLSN and undonextLSN values.
2. Describe the actions taken to rollback transaction $T2$.
3. Show the log after $T2$ is rolled back, including all prevLSN and undonextLSN values in log records.

LSN	LOG
00	update: T1 writes P2
10	update: T1 writes P1
20	update: T2 writes P5
30	update: T3 writes P3
40	T3 commit
50	update: T2 writes P5
60	update: T2 writes P3
70	T2 abort

Schema Refinement (20%)

Problem 5. Consider a relation R with five attributes $ABCDE$. You are given the following dependencies: $A \rightarrow B$, $BC \rightarrow E$, and $ED \rightarrow A$.

1. List all keys for R.
2. Is R in 3NF?
3. Is R in BCNF?