CS 222 – Fall 2010
Department of Computer Science, UC Irvine
Prof. Chen Li
Final Exam
(Max. Points: 100)

Instructions:

- This exam has **six (6)** questions.
- This exam is closed book and closed notes.
- The total time for the exam is **120** minutes, so budget your time accordingly.
- Be sure to answer each part of each question after reading the whole question carefully.
- If you don't understand something, ask for clarification.
- If you still find ambiguities in a question, write down the interpretation you are taking and then answer the question based on that interpretation.

STUDENT NAME: **STUDENT ID:**

QUESTION	POINTS	SCORE
1	15	
2	10	
3	15	
4	20	
5	20	
6	20	
TOTAL	100	

Problem 1 (15 pts): Histograms

Consider a relation **Review(rid, movieid, rating, comments, …)** with information about reviews on movies including their ratings. The following table shows the number of reviews for each rating range (between 0 and 5). For example, there are 8 reviews with a rating between 3.0 and 3.5.

Rating	(0,0.5]	(0.5,1.0]	(1.0,1.5]	(1.5,2.0]	(2.0,2.5]	(2.5,3.0]	(3.0,3.5]	(3.5,4.0]	(4.0,4.5]	(4.5,5.0]
# of Reviews	2	4	10	9	7	16	8	8	11	5

(1) **(3 pts)** We want to build an **equi-width** histogram with **5** buckets. Show the structure of the histogram.

(2) **(4 pts)** Show how to use the histogram to estimate the number of records in the answer to the following query:

SELECT *
FROM Review
WHERE rating > 1.3 AND rating <= 3.5;

(3) **(4 pts)** We want to build an **equi-height** histogram with **5** buckets. Show the structure of the histogram.

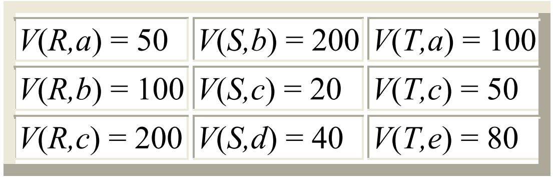
(4) **(4 pts)** Show how to use the histogram to estimate the number of records in the answer to the same query above.

Problem 2 (10 pts): Intermediate-Size Estimation

In this question, we use the notation $T(R)$ for the number of tuples in relation R, and $V(R,A)$ for the number of distinct values in attribute A of relation R. We make the following assumptions:

1. All possible values of an attribute are equally likely to appear.
2. Values of different attributes are independent.
3. The **Containment of Value Sets Assumption**: if $V(R,A) <= V(S,B)$, then every value appearing in $R.a$ also appears in $S.b$.
4. The **Preservation of Value Sets Assumption**: if an attribute is not one of the join attributes, then all values of that attribute that appeared in one of the operand relations of a join also appear in the result.

Consider the join of three relations $R(a,b,c)$, $S(b,c,d)$, and $T(a,c,e)$. Let these relations have r, s, and t tuples, respectively. Let the numbers of values of the attributes be:

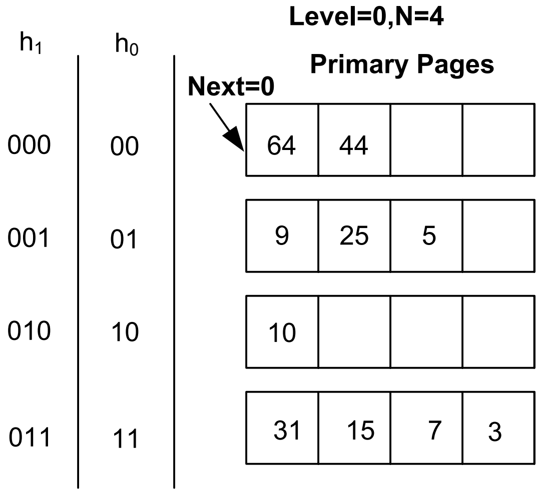


$V(R,a) = 50$	$V(S,b) = 200$	$V(T,a) = 100$
$V(R,b) = 100$	$V(S,c) = 20$	$V(T,c) = 50$
$V(R,c) = 200$	$V(S,d) = 40$	$V(T,e) = 80$

We want to compute the natural join of R, S, and T. Show the join sequence that can minimize the number of tuples in the intermediate results, and explain how to you compute this sequence.

Problem 3 (15 pts): Linear Hashing

Consider the snapshot of the linear hashing index shown in the following figure.



Assume: h_0(key) = last two digits of "key", and h_1 (key) = last three digits of "key". Also assume that a bucket split occurs whenever an overflow page is created.

a) **(4 pts)** What is the ***maximum*** number of data entries that can be inserted (given the best possible distribution of keys) before you have to split a bucket? Explain your answer briefly.

b) **(4 pts)** Show the file after inserting a ***single*** record whose insertion causes a bucket split.

c) **(7 pts)** What is the ***minimum*** number of record insertions that will cause a split of ***all*** four buckets? What is the value of “Next” after making these insertions? Explain your answer briefly.

Problem 4 (20 pts): Merge Sort

Consider a disk with an average seek time of 10ms, an average rotational delay of 5 ms, and a transfer time of 1ms for a 4KB 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 11 buffer pages and asked to sort a file with 110 pages.

a) **(4 pts)** Why is it a bad idea to use the 11 pages to support virtual memory (that is, to "new" 110 * 4K bytes of memory), and to use the in-memory sorting algorithm such as Quicksort?

b) **(7 pts)** Assume that you begin by creating sorted runs of 11 pages each in the first pass. Then create 10 input buffers of 1 page each, create an output buffer of 1 page, and do a 10-way merge. Calculate the total time cost of this approach. Make sure to provide the details of your analysis, and specify the number of sequential IOs and random IOs.

c) **(9 pts)** Again, assume that you begin by creating sorted runs of 11 pages each in the first pass. Then create 5 input buffers of 2 pages each, create an output buffer of 1 page, and do 5-way merges. Calculate the total time cost of this approach. Make sure to provide the details of your analysis, and specify the number of sequential IOs and random IOs.

Problem 5 (20 pts): Join

Consider two relations R(A,B) and S(B,C), and their natural join on their B attributes. We have the following information:

- Relation R contains 1,000 tuples and has 10 tuples per page.
- Relation S contains 2,000 tuples and also has 10 tuples per page.
- Attribute B of relation S is the primary key for S.
- Both relations are stored as simple heap files.
- Neither relation has any indexes built on it.
- 52 buffer pages are available.

In the following sub-questions, "cost" is the number of disk IOs. Also ignore the IOs to write the final results.

a) **(6 pts)** What is the cost of doing the join using a **page-oriented simple nested loops join**? What is the minimum number of buffer pages required for this cost to remain unchanged?

b) **(6 pts)** What is the cost of doing the join using a **block nested loops join**? What is the minimum number of buffer pages required for this cost to remain unchanged?

c) **(8 pts)** What is the cost of doing the join using a **grace hash join**? What is the minimum number of buffer pages required for this cost to remain unchanged?

Problem 6 (20 pts): System-R Optimizer

Suppose we have the following relations:

- Emp(**eid**, did, name, sal, …)
- Dept(**did, projid,** budget, …)
- Project(**projid**, projname, …)

Each relation has its primary key attribute(s) underlined. Assume employee salaries are uniformly distributed in the range between 10,009 and 110,009, and the project budgets are uniformly distributed in the range between 10,000 and 30,000. There is a B+ index on **Emp.eid** and a B+ tree on **Emp.sal**.

Consider the following query:

```
SELECT E.eid, D.did, P.projid
FROM Emp E, Dept D, Proj P
WHERE E.sal = 50,000 AND D.budget > 20,000 AND
          E.did = D.did AND D.projid = P.projid
ORDER BY E.eid;
```

We want to find an efficient plan for the query using the System-R optimizer. The cost is the number of disk IOs. Ignore the cost of writing out the final result.

a) **(6 pts)** For each of the following sets of joined relations, specify the attributes whose orders are interesting.

- {E}: ____

- {D}: ____

- {P}: ____

- {E JOIN D}: ____

- {D JOIN P}: ____

- {E JOIN D JOIN P}:____

b) **(6 pts)** Describe at least two different plans to access the relation Emp. For each of the attributes whose order is interesting, specify whether each described plan can produce an interesting order on that attribute.

c) **(8 pts)** Briefly explain how the optimizer can find the optimal plan. Specifically describe how the information of interesting orders in early subplans is used to generate subplans with more relations.

You can use this page as scratch paper.