You are editing your previous response.

Be careful when sharing the URL of this page, because it will allow others to also edit your response.

Use this link to share a blank version of the form.

https://docs.google.com/a/umich.edu/forms/d/e/1FAIpQLScI7yzD2DC7Ct101MiMQNS

Due: Tuesday, Dec. 13th, 11:59PM

Instructions:
For this homework, you may submit answers any number of times, but only the last submission counts.

If you are working in a group (at most two), that is OK. But, we ask that both you and your partner submit separately. Your group's submission must include the uniqname of your partner as the answer to the first question below. The points are going to be assigned to you based on your submission, not your partner's submission.

Make sure to save the link from your submission so that you can go back and edit your answers later, if necessary.

Your email address (**yuke@umich.edu**) will be recorded when you submit this form. Not you? Sign out

If you are working in a group, please give the uniqname of your partner:

Leave blank if working alone. Note that you may NOT work in groups of three or more. (Note: your partner should submit as well separately, indicating you as the partner. Each will get their respective scores.)

czunyu

Question 1 (10 points)

Consider an extensible hash index that uses the following hash function:
hashvalue = last d bits of the binary version of the number given, where d is the global depth.
Assume that for this index the bucket (data page) capacity, which consists of a single page, is 2 entries, the initial global depth is 1, and the index is initially empty.

The following data entries with key values 16, 23, 60, 41, 26, 37, 28 are inserted into the index in the given order. Answer the following questions based on the information given.

a) (3 points) What is the global depth of the final index?

3

b) (3 points) How many data pages contain two data entries in the final index?

2

c) (3 points) How many data pages have a local depth equal to global depth in the final index?

2

d) (1 points) When splitting a data page, must we always also double the directory? (1 point)

True [✓ False]

Question 2 (10 points)

For this question, you will consider sorting a large dataset on a modern hardware setup. Suppose that your dataset consists of an 8TB file and that you have reserved 1GB of memory (RAM) to assist in the sorting. Also suppose that your system uses a page size of 16 KB. (Note: 1KB = 1024 Bytes, 1MB = 1024KB, 1GB=1024MB, 1TB=1024GB).

a) (5 points) How many passes, on average, will be required to sort this dataset using external merge sort using the external merge sort scheme described in Section 13.3 of the book.

2

b) (5 points) Based on your analysis above, how many total I/Os (page reads and writes) will be required on average to perform the sort?

2147483648

Question 3 (10 points)

Consider the following relational schema and SQL query:

Students(sid, sname, gpa)
Takes(sid, cid)
Class(cid, cname, ctype)

```
SELECT S.sname, C.cname
FROM Students S, Takes T, Class C
WHERE S.sid = T.sid AND T.cid = C.cid AND S.GPA < 1.0 AND C.ctype = ‘Math’
```

a) (3 points) Assuming joins which are simply cross-products are not allowed, how many different join orders would a System R-style query optimizer consider? Consider Join(S,T) to be different from Join(T,S), since the outer relation is different.

4

4

b) (7 points) Which of the following indexes match this query? Select all that apply. Only consider selection predicates, not join predicates.

- ☐ <S.sid> (hash)
- ☐ <S.GPA> (hash)
- ✓ <S.GPA> (B+-tree)
- ✓ <C.ctype> (hash)
- ☐ <C.cname, C.ctype> (hash)
- ☐ <C.cname, C.ctype> (B+-tree)
- ☐ <C.ctype, C.cname> (hash)

Question 4 (25 points)

Using the schema of S, T, and C and the query from question 2, consider a query plan that works as follows:

Join(Join(Selection(S), T), Selection(C)), with the added provision that any additional projections and selections not listed in this query plan are cascaded and pushed in as far as possible. We and selections not listed in this query plan are cascaded and pushed in as far as possible. We use Grace Hash Join for any joins.

We have a clustered B+Tree index with search key of GPA to access S, an unclustered hash index with search key of ctype to access C, and a scan to access T.

Make the following assumptions:

System:
- Our system has a page size of 4096 bytes and a memory buffer size of 5 pages.
- Each attribute has a size of 8 bytes.
- Each RID (pointer to the data record) has a size of 16 bytes.
- Backward and forward pointers for the double-linked list at the leaves take up 8 bytes each.

Table Students:
- Table Students has a B+ tree index on GPA
- This index is of height 2 with 1000 leaf pages.
- Only the leaf pages of this index are on disk (i.e., all of the internal nodes are cached in RAM).
- The average fill factor for leaf nodes for this index is 2/3 (0.666).
- Assume that S.GPA < 1.0 predicate has 1% selectivity.

Table Class:

- The ctype attribute takes 100 distinct values, all equally likely.
- There are 16K (i.e., 16 * 1024) tuples in the Class relation

Table Takes:
-- There are 256K (i.e., 256 * 1024) tuples in the Takes relation.

We will assume that data records are not split across pages and the hash function used in hash join algorithms partitions the relations uniformly. All indexes use Alternative 2. Also, for simplicity, you can assume that data pages are fully packed with data (i.e., no space is required for bitmaps, back pointers, forward pointers, etc.). Thus, if a data record is 16 bytes and page size is 4KB, a page will contain 256 records (of course, the last page in the data file may be partially filled).

For each join, you may choose either operand as the inner one (whichever results in a lower cost!), and you should pipeline the execution plan as far as possible. Keep in mind that if one pass of hashing a relation results in partitions that are too large for available memory, you may need additional passes to hash each partition into smaller partitions.

a) (5 points) How many tuples are in S?

113000

b) (5 points) How many pages of data does S have?

b) (5 points) How many pages of data does S have?

663

c) (5 points) How many pages of data does T have?

1024

d) (5 points) Based on the query above, what is the total number of pages required to perform selection(S) Your answer should include the cost to read the necessary pages and materialize (write) the results. Use the most efficient strategy

24

e) (5 points) What would be the number of I/O operations for performing join(selection(S), T) based on the query in question 2? Ignore the cost of writing the final result

3093

Question 5 (15 points)

In class, we developed a cost model for various relational database operations, such as joins. We counted the total number of I/Os (i.e., writing or reading a page) required to perform each operation. From there, we could estimate the relative performance of two query plans. If we were to consider a nested-loop join vs. a hash join, for example, we might guess that using the hash join would be better because it would require fewer I/Os.

In modern magnetic disks, the cost of performing a sequential read is approximately the same as that of performing a sequential write; however, you might imagine another kind of secondary storage device where this is not true. For example, it is widely accepted that when using flash memory devices, it is cheaper to read a block than it is to write a block. Therefore, you might also imagine that replacing our magnetic disk with another device would affect our cost estimates, which could in turn affect the optimizer's plan (i.e., choice of algorithms).

For this question, suppose that you are given two relations, R and S, and that you want to perform a natural join between the two. Suppose the following is also true of the two relations:

Number of records in R = 20,000
Number of pages in R = 2000
Number of records in S = 100,000
Number of pages in S = 5,000

Number of Buffer Pages = 501

Given the above, answer the following questions. In all cases, please ignore the cost of writing the final result back to secondary storage.

a) (3 points) How many reads are required for Block Nested Loop Join?

27000

b) (3 points) How many writes are required for Block Nested Loop Join? Ignore the cost of writing the final result

0

c) (3 points) Assuming a uniform distribution, what is the average size of each partition (in pages) of R for Grace Hash Join after the initial partitioning phase?

4

d) (3 points) How many reads are required for Grace Hash Join?

14000

e) (3 points) How many writes are required for Grace Hash Join? Ignore the cost of writing the final result

7000

f) Suppose to read a page from disk will take 1 μs, and to write a page to disk will take X μs. Given your analysis above, for what value of X do Grace Hash Join and Block Nested Loop Join take take same amount of time? Round your answer to the nearest hundredth

1.86

Before You Submit

Know that once you click submit, you will receive a link that will allow you to return and edit your submission afterwards. You may continue to edit your submission until the homework deadline.

While you should also receive an email with the link to edit your submission, we recommend saving or bookmarking the link after submitting, just as a precaution.

Send me a copy of my responses.

Page 1 of 1 SUBMIT