[GRADED] Vitamin 2: Disks, Buffers, Files

Q1 Ordering

1 Point

Order the following from largest to smallest in terms of layers of storing information in a relational database:

- ○ Files, records, pages
- ○ Pages, files, records
- ◉ Files, pages, records
- ○ Records, pages, files

EXPLANATION

Files contain pages, and pages contain records.

✔ **Correct**

Save Answer Last saved on **Feb 02 at 1:22 PM**

Q2 Free Space in Linked List

2 Points

Q2.1

Assume you have a heap file implemented in a linked list structure with the header page connected to a linked list of full pages and a linked list of pages with free space. There are 5 full pages and 10 pages with free space. In the worst case, how many pages would you have to read to see if there is a page with enough space to store some record with a given size? Assume at least 1 page has enough space to fit this record.

Note: *For all problems in this course that only specify whether the file is a heap or sorted file without providing the underlying implementation, ignore the I/O cost associated with reading/writing the file's header pages.*

However, for all problems that provide a specific file implementation (i.e. heap file implemented with a linked list or page directory), include the I/O cost associated with reading/writing the file's header pages.

- ○ 5
- ○ 6
- ○ 10
- ◉ 11

EXPLANATION

You have to read the header page and then you may have to sequentially read all 10 pages with free space to see if any of them have enough space for this record, for a total of 11 pages.

Q2.2

Same question as above (5 full pages and 10 pages with free space), but now what if instead you had a page directory implementation of a heap file with 8 directory entries per header page?

- ◉ 2
- ○ 3
- ○ 8
- ○ 6
- ○ 11

EXPLANATION

You would have to read at most 2 pages because each directory entry stores how much free space is in the data page it points to. So, you would have to check at-worst all 15 data page directory entries (including the full pages which are also stored in the page directory with 0 space), and there are 8 directory entries per header page, so ceiling(15 / 8) = 2.

✔ Correct

Save Answer Last saved on **Feb 04 at 11:10 PM**

Q3 Heap File vs Sorted File

2 Points

Answer the questions with the following assumptions:

1. Inserts and deletes only involve one record at a time
2. For Heap Files: Inserts always append to the end of the file
3. For Sorted Files: The files are sorted based on the sort key used during lookups
4. For Sorted Files: The page layout is packed

Q3.1

For which of these workloads should we use heap files over sorted files? Assume the operations listed in the answer choice are the only operations that we will do.

- ✔ frequent full scans, occasional inserts
- ✔ frequent deletes, occasional full scans
- ✔ frequent inserts, rare reads
- ☐ frequent selecting range of records by key, rare inserts
- ☐ frequent key look up, infrequent writes
- ✔ frequent updates, occasional reads

Q3.2

For which workloads should we use sorted files over heap files?

- [ ] frequent full scans, occasional inserts
- [ ] frequent deletes, occasional full scans
- [ ] frequent inserts, rare reads
- [x] frequent selecting range of records by key, rare inserts
- [x] frequent key look up, infrequent writes
- [ ] frequent updates, occasional reads

EXPLANATION

Let B = The number of pages in the file. Full scans cost the same number of IOs for heap files and sorted files, because you read all B pages. Heap files are better for inserts as you append to the end of the file with a cost of 2 IOs (reading and writing the last page). However, sorted files have a cost of log(B) + B IOs (finding the page to insert the record on and pushing all records to the right over). Heap files are also better for frequent deletes and updates. On average, heap file deletes cost (0.5 * B) + 1 IOs (finding the record to delete, and writing the page back). Sorted file deletes cost the same as an insert in a sorted file, because all records to the right of the deleted record need to shifted over to maintain the packed layout. Heap file updates have the same cost as a heap file delete. Sorted file updates include the cost of a delete and an insert in a sorted file, which makes them costly. On the other hand, sorted files are a lot better for frequent reads and range lookups because we can take advantage of the sorted order, while heap files will involve a full scan for any lookup.

✔ **Correct**

Save Answer Last saved on **Feb 04 at 11:29 PM**

Q4 How Many Records?

2 Points

For the next two questions, consider the following schema:

```
CREATE TABLE Inventory (
  item_id integer PRIMARY KEY,
  quantity integer,
  item_description text
);
```

Q4.1

Given the schema above, at most how many variable-length records from this table can be stored in a 64 KB page (1 KB = 1024 Bytes)? Assume each page's footer contains a 10-byte area for the record count and pointer to free space, as well as a slot directory where it takes 4 bytes/record to store the pointer to the record in the page and 4 bytes/record to store the length of that record. Assume integers are 4 bytes each. Assume records do not have record headers or pointers to the variable-length fields in the records.

- ○ 2047
- ○ 2048
- ◉ 4095
- ○ 4096

EXPLANATION

The number of bytes available in the page is given by 64*1024. The record count and free space pointer take up 10 bytes. Per record, we need 4 + 4 = 8 bytes to store the slot directory information, and since the max number of records fitting in the page is achieved when the length of the variable length field "item_description" is always 0 bytes, the total amount of bytes we need in the page per record is 8 + 8 = 16 (8 bytes for slot directory entry, 8 bytes for tuple from the two integer fields). Thus, the most number of records is floor((64*1024 – 10)/16) = 4095 records.

Q4.2

Assume we are using the same schema from the last question, but each record now has a record header that takes up 4 bytes in addition to storing a 32-bit pointer to the end of each variable-length field's value. What is the size in bits of the smallest record possible?

- ○ 32
- ○ 64
- ○ 96
- ◉ 128

EXPLANATION

8*8 + 4*8 + 32 = 128 bits. 8 bytes for the fixed-length integer fields, 4 bytes for the record header, and 32 bits for the pointer to the single variable-length field "item_description" (which would be 0 bits long since the question is asking for the size of the smallest record).

✔ Correct

Save Answer Last saved on **Feb 04 at 11:33 PM**

Q5 Heap File vs Sorted File

3 Points

For the following questions, consider this SQL table:

```
CREATE TABLE Students (
  sid INT PRIMARY KEY,
  name VARCHAR(20),
  email VARCHAR(20),
  GPA float,
  years_enrolled INT
);
```

And the following 3 queries:

A:

```
SELECT * FROM Students WHERE years_enrolled >= 2;
```

B:

```
DELETE FROM Students WHERE sid = 123456;
```

C:

```
INSERT INTO STUDENTS VALUES (9999, 'Joe', j@b.com, 2);
```

For the following questions, **assume we leave each page about 2/3 full.** Order these queries in ascending order by the total number of I/Os required to execute them; ties can be listed in any order. If there is a tie, select all possible answers. (For example, if you believe that A and B have the same cost and C's cost is greater, select the answer choices for A, B, C and B, A, C).

Q5.1

If the Students table is stored with a heap file, what is the order of the queries (in the average case)?

- [ ] A, B, C
- [ ] A, C, B
- [ ] B, A, C
- [ ] B, C, A
- [ ] C, A, B
- [x] C, B, A

EXPLANATION

The insert is the fastest, because we can tack that record onto the end of the heap file. The delete is the next fastest, as we, on average, scroll through half the pages before deleting it. The first case is equivalent to a full scan and requires us to read all N pages in the file.

Q5.2

If the Students table is stored with a sorted file, sorted on `sid`, what is the order of queries?

- [ ] A, B, C
- [ ] A, C, B
- [ ] B, A, C
- [x] B, C, A
- [ ] C, A, B
- [x] C, B, A

EXPLANATION

Note the key that the table is sorted on. Option A is a range search, but on the wrong key, so it's not actually any better than performing a full scan. Since the pages are kept 2/3 full, an insertion or deletion would not require shifting the pages. This is because the delete query is deleting based on the value of the sort key for the file.

Q5.3

What would be the size of a single tuple using fixed length records? Assume that integers and floats are 4 bytes long.

- ○ 56 Bytes
- ◉ 52 Bytes
- ○ 40 Bytes
- ○ 16 Bytes

EXPLANATION

4 + 20 + 20 + 4 + 4 = 52 (floats and ints are 4 bytes)

✔ **Correct**

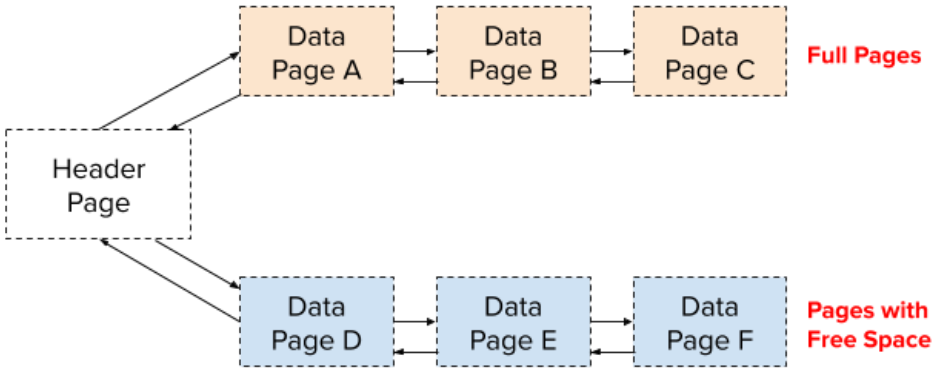
Save Answer Last saved on **Feb 05 at 3:29 AM**

Q6 What's the cost?

2 Points

Given the following Heap File with a Linked List implementation, what is the worst case I/O cost to insert a record? Provide only a number in the box.

Assume the following:
1. Once a page is read into memory, it stays there and a subsequent read incurs no additional IO cost.
2. If a Free Space page becomes full after an insert, it is added to the front of the Full Pages list.
3. At least one page has enough space to fit the record



10

EXPLANATION

The worst case occurs when a record is inserted to Page E causing the page to become full.
The IO cost involves: reading the header page (1), reading Page D and E (2), updating the next and previous pointers on Page D and F (3, because Page D was already in memory and needs to be written, and Page F has to be read and written), writing the record to Page E and updating its pointers so it becomes part of the Full Pages list (1), reading in Page A and updating its previous pointer (2), updating the next pointer of the Header Page for the Full Pages list (1). This results in a total cost of: 1 + 2 + 3 + 1 + 2 + 1 = **10 IOs**.
Note: inserting to Page F and causing it to become full would result in only 9 IOs.

✔ Correct

Save Answer Last saved on **Feb 05 at 3:30 AM**

Save All Answers Submit & View Submission >