

CS 186 - Fall 2024

Exam Prep Section 2

Disks and Files

Question 1: Files, Pages, Records

Consider the following relation:

```
CREATE TABLE Cats (  
    collar_id INTEGER PRIMARY KEY, -- cannot be NULL!  
    age INTEGER NOT NULL,  
    name VARCHAR(20) NOT NULL,  
    color VARCHAR(10) NOT NULL  
);
```

You may assume that:

- INTEGERS are 4 bytes long;
- VARCHAR(n) can be up to n bytes long.

(a) As the records are variable length, we will need a *record header* in the record. How big is the record header? You may assume pointers are 4 bytes long, and that the record header only contains pointers.

(b) Including the record header, what is the smallest possible record size (in bytes) in this schema? (Note: NULL is treated as a special value by SQL, and an empty string VARCHAR is different from NULL, just like how a 0 INTEGER value is also different from NULL. We will provide the necessary clarification should similar questions appear on an exam.)

(c) Including the record header, what is the largest possible record size (in bytes) in this schema?

(d) Now let's look at pages. Suppose we are storing these records using a slotted page layout with variable length records. The page footer contains an integer storing the record count and a pointer to free space, as well as a slot directory storing, for each record, a pointer and length. What is the maximum number of records that we can fit on a 8KB page? (Recall that one KB is 1024 bytes.)

(e) Suppose we stored the maximum number of records on a page, and then deleted one record. Now we want to insert another record. Are we guaranteed to be able to do this? Explain why or why not.

(f) Now suppose we deleted 3 records. Without reorganizing any of the records on the page, we would like to insert another record. Are we guaranteed to be able to do this? Explain why or why not.

Question 2: Files, Pages, Records

Consider the following relation:

```
CREATE TABLE Student (  
    student_id INTEGER PRIMARY KEY,  
    age INTEGER NOT NULL,  
    units_passed INTEGER NOT NULL,  
);
```

- (a) Are Student records represented as fixed or variable length?
- (b) To store these records, we will use an unpacked representation with a page header. This page header will contain nothing but a bitmap, rounded up to the nearest byte. How many records can we fit on a 4KB page? (Recall that one KB is 1024 bytes.)

- (c) Suppose there are 7 pages worth of records. We would like to execute

```
SELECT * FROM Student WHERE student_id = 3034213355; -- just some number
```

Suppose these pages are stored in a heap file implemented as a linked list. What is the minimum and maximum number of I/Os required to answer the query?

- (d) Now suppose these pages are stored in a sorted file, sorted on student id. What is the minimum and maximum number of I/Os required to answer the query? You can assume sorted files do not have header pages.

Question 3: Files, Pages, Records

- (a) Suppose we are storing variable length records in a linked list heap file. In the "pages with space" list, suppose there happens to be 5 pages. What is the maximum number of I/Os required in order to insert a record?

You may assume that *at least one of these pages contains enough space*, and additionally that *it will not become full after insertion*.

- (b) Continuing from part (a), suppose that the page does become full after insertion. Now, we need to move that page to the "full pages" list.

Assume we have already done all necessary page reads for part (a)'s worst case (and that those pages are still in memory), but have not yet done any page writes.

How many additional I/Os do we need to move the page to the "full pages" list?

- (c) Now suppose records are fixed length; what is the maximum number of I/Os to insert a record?
Assume that the page we insert into does not fill up after the insertion.

- (d) Now suppose we are using a page directory, with one directory page. What is the maximum number of I/Os required to insert a record?