Discussion 3: Files

吕炜杰 lvwj@shanghaitech.edu.cn

Reminder:

- Homework 1 is due Thursday(10.1) at 23:59!
- TA’s office hour is on Monday 19:00 to 20:00 @ 1A-106

Files, Pages and Records

- File (corresponds to a table)
    - Page (many per file)
        - Record (many per page)

File Organization

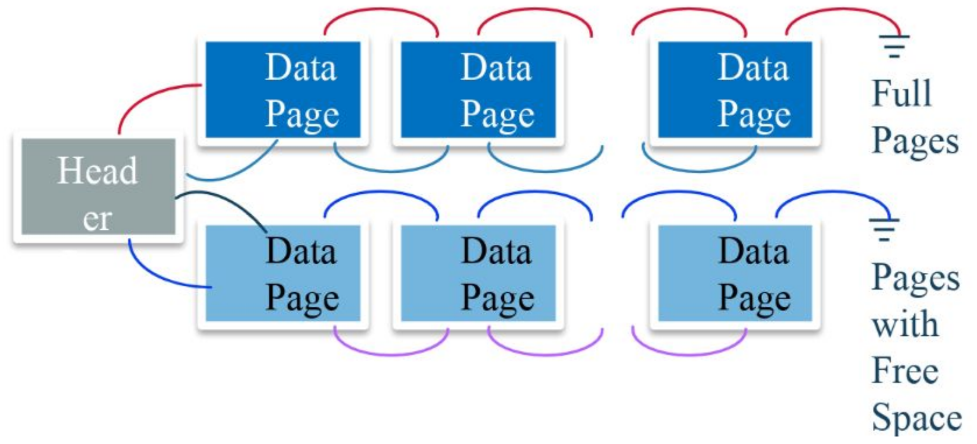
- Different DB Files Structures
  - Heap Files: a file with no order enforced.
  - Sorted Files: is similar to a heap file, except we require it be sorted on a key (a subset of the fields).
  - Index Files: B+ Trees, Linear Hashing, ...

File Organization

- Two Way to Implement Heap File
    - As a List
    - Use a Page Directory (Better)

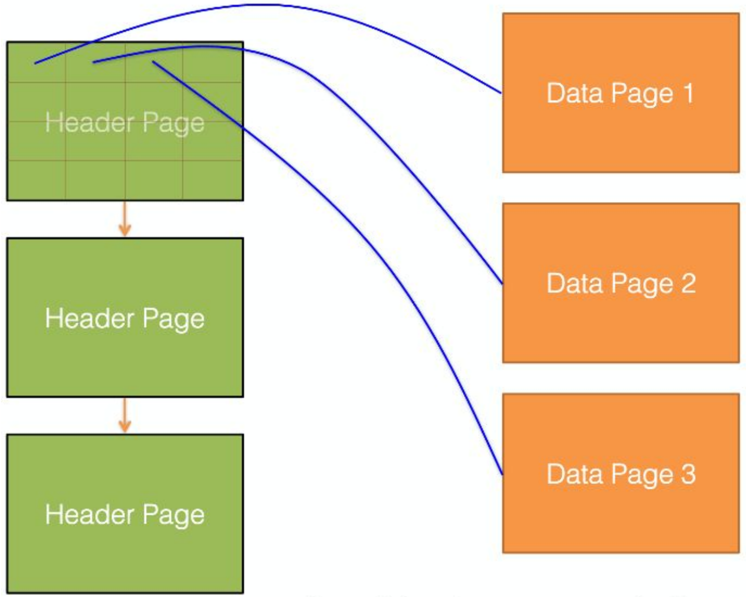
Implementing Heap File As A List

- Each page has two pointers, free space, and data.
- We have two linked lists of pages, both connected to a header page
  - List of full pages
  - List of pages with some empty space



Implementing Heap File With Page Directory

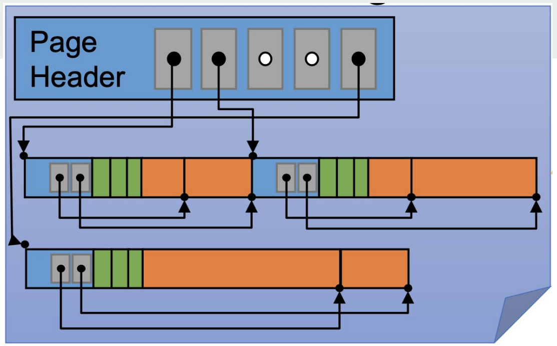
- We have a linked list of **header pages**, which are each responsible for a set of data pages
    - Stores the amount of free space per data page



Pages and I/Os

- Pages are managed
  - in memory by the buffer manager: higher levels of database only operate in memory
  - on disk by the disk space manager: reads and writes pages to physical disk/files
- Unit of accesses to physical disk is the page
  - **Cannot** fetch fractions of a page
  - I/O: unit of transferring a page of data between memory and disk (read OR write 1 page = 1 I/O)

Page basics: the header



- The **page header** is a portion of each page reserved to keep track of the records in the page.
- The page header may contain fields such as:
  - Number of records in the page
  - Pointer to segment of free space in the page
  - Bitmap indicating which parts of the page are in use

Records

- Fixed Length Records
  - Paced: no gaps between records, record ID is location in page
  - Unpaced: allow gaps between records, use a bitmap to keep track of where the gaps are
- Variable Length Records
  - Contains a record header to store field offset indicating where each variable length field ends

Variable Length Records: Fragmentation

- Deleting records causes fragmentation if we use an unpacked layout:
    - [3 byte record] [2 byte record] [3 byte record] [2 bytes free space]
    - If we delete the 2 byte record, we have 4 bytes free on the page but cannot insert a 4 byte record

Exercise 1: True or False

- When querying for an 16 byte record, exactly 16 bytes of data is read from disk.

Exercise 1: True or False

- When querying for an 16 byte record, exactly 16 bytes of data is read from disk.
- False. an entire page of data is read from disk.

Exercise 2: True or False

- In a heap file, all pages must be filled to capacity except the last page.

Exercise 2: True or False

- In a heap file, all pages must be filled to capacity except the last page.
- False, there is no such requirement.

Exercise 3: True or False

- Assuming integers take 4 bytes and pointers take 4 bytes, a slot directory that is 512 bytes can address 64 records in a page.

Exercise 3: True or False

- Assuming integers take 4 bytes and pointers take 4 bytes, a slot directory that is 512 bytes can address 64 records in a page.
- False, we have the free space pointer, which doesn't fit after 64 * (4 + 4) = 512 bytes of per-record data in the slot directory.

Exercise 4: True or False

- In a page containing fixed-length records with no nullable fields, the size of the bitmap never changes.

Exercise 4: True or False

- In a page containing fixed-length records with no nullable fields, the size of the bitmap never changes.
- True, the size of the records is fixed, so the number we can fit on a page is fixed.

Exercise 5: True or False

- Variable length records always match or beat space cost when compared to fixed-length record format

Exercise 5: True or False

- Variable length records always match or beat space cost when compared to fixed-length record format
- False, extra space is required for the record header.

Exercise 6: True or False

- Is fragmentation an issue with packed fixed length record page format? What about a slotted page?

Exercise 6: True or False

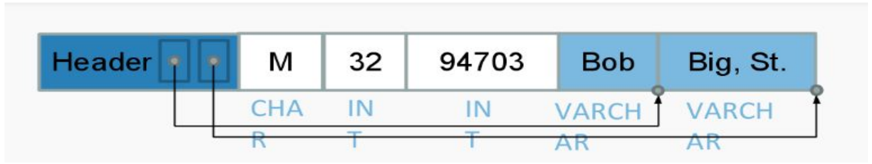
- Is fragmentation an issue with packed fixed length record page format? What about a slotted page?
- No, records are compacted upon deletion.
- Yes

Exercise: Record Format

Assume we have a table that looks like this:

```
CREATE TABLE Questions (
    qid integer PRIMARY KEY,
    answer integer,
    qtext text,
);
```

(a) Recall that integers and pointers are 4 bytes long. Assume for this question that the record header stores pointers to all of the variable length fields (but that is all that is in the record header). How many bytes will the smallest possible record be?



Exercise: Record Format

12 bytes. The record header will only contain one pointer to the end of the qtext field so it will only be 4 bytes long. The qid and answer fields are both integers so they are 4 bytes long, and in the smallest case qtext will be 0 bytes. This gives us a total of 12 bytes.

Assume we have a table that looks like this:

```
CREATE TABLE Questions (
    qid integer PRIMARY KEY,
    answer integer,
    qtext text,
);
```

(a) Recall that integers and pointers are 4 bytes long. Assume for this question that the record header stores pointers to all of the variable length fields (but that is all that is in the record header). How many bytes will the smallest possible record be?

Exercise: Record Format

(b) Now assume each field is nullable so we add a bitmap to the beginning of our record header indicating whether or not each field is null. Assume this bitmap is padded so that it takes up a whole number of bytes (i.e. if the bitmap is 10 bits it will take up 2 full bytes). How big is the largest possible record assuming that the qtext is null?

Exercise: Record Format

(b) Now assume each field is nullable so we add a bitmap to the beginning of our record header indicating whether or not each field is null. Assume this bitmap is padded so that it takes up a whole number of bytes (i.e. if the bitmap is 10 bits it will take up 2 full bytes). How big is the largest possible record assuming that the qtext is null?

13 bytes. We have 3 fields so there will be 3 slots in our bitmap (and thus 3 bits), meaning our record header will only be 1 byte longer than it was in part a. In the max case, both the qid and answer field will be present and still be 4 bytes each. Therefore, the fields haven't changed at all, and the total record length is now 13 bytes.

Exercise: Calculate the I/Os

Assume we have a heap file A implemented with a linked list and heap file A has 5 full pages and 2 pages with free space, at least one of which has enough space to fit a record.

In the worst case, how many IOs are required to find a page with enough free space?

Exercise: Calculate the I/Os

Assume we have a heap file A implemented with a linked list and heap file A has 5 full pages and 2 pages with free space, at least one of which has enough space to fit a record.

In the worst case, how many IOs are required to find a page with enough free space?

3 I/Os, you read in header and then the 2 pages in the free pages linked list before finding a page with enough free space in the final page.