Heap File Assignment, PSU CS 487/587 (Database Implementation)

Introduction

In this assignment, you will implement the HeapFile class in the heap package. This assignment is to be done individually or in pairs. Learning objectives for this assignment are to familiarize yourself with the concepts of data vs. index/directory pages and efficiency in managing database files.

Details

Begin by reviewing Sections 9.5.1, 9.6.2 and 12.4.3 of the text. Then:

- Retrieve the code from the web site and unpack it.
- You are not given source code for the bufmgr layer – only the classes in a jar file. This is to avoid distributing a solution. You may wish to substitute your own source code for the bufmgr layer so that during debugging you can trace inside that layer, if you are confident that your source works.
- Remind yourself about methods available in the bufmgr and diskmgr package/layers, you will be calling them.
- Your task is to implement the HeapFile class, following the algorithm described on page 326 of the text. You should deal with free space intelligently, using the text's directory (not linked list) method to identify pages with room for records. When a record is deleted, you must update your information about available free space. When a record is inserted, you must try to use free space on existing pages before allocating a new page. Run the HFTest.java to ensure that your code works.
- In the heap package, you will find the class HFPage, which is an implementation of the slotted page structure on page 329 of the text. You will also find classes DirPage, DataPage and HeapScan. Review these thoroughly. Note that HeapScan is implemented as an interator.
- Optional Advice: The most common problem students have with HeapFile is to confuse directory with data records and pointers.
- You should catch, and clean up from, any exception thrown by a method in the heapfile layer. For simplicity you may assume that any methods called from other layers do not throw exceptions. Keep track of what pages are pinned upon exit from each method you write.

Where to Find Code

The code (in .tar and .zip files) is available on the course web site. The contents are:

- *bm.jar* You will need to put this in the search path of your compiler/IDE (in Ecliplse, click on Project/Properties/Java Build Path/Libraries/Add Jar), or use your own code for bufmgr.
- *bufmgr:* This is empty so that you can insert your own code if you wish to use it instead of bm.jar.
- *diskmgr, global:* Same as in your previous assignment
- *tests:* The public tests are in tests.HFTest.

- *heap:* This contains the heapfile skeleton that you should change to heapfile.java as in the previous assignment. It also contains code for other classes you will need. Feel free to change these if you wish, but turn in the changed files if you do.
- *output.txt*: as before

Grading & What to Turn In

Project will be graded by demonstration as with the Buffer Manager assignment. You are also required to turn in your source code. Email source code with all changed files to tufte@pdx.edu.