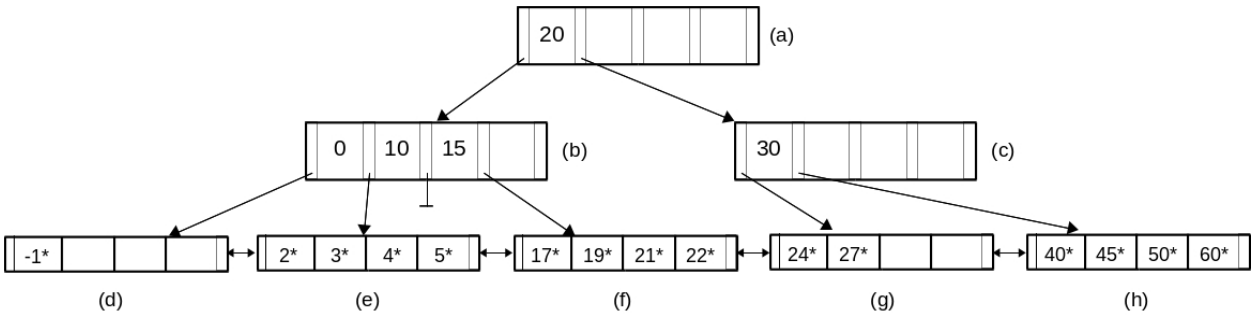
Lab 6: Indexing, Sorting, and Advanced SQL

You may work with one other person on this lab. To submit your assignment, place a PDF in your `~/cs44/labs/6/` *directory and use* `handin44` *to electronically submit the lab. Be sure both names are on the document. Your assignment should be submitted by 11:59pm on* **Saturday, April 26, 2014**.

B+ Tree Indexes

1. Consider the B+ tree in the following figure of order $d = 2$. List all violations of a property of B+ trees. For each violation, simply list the violation and the offending node. For example, if we had a hypothetical node (x) that had too many values, you would list "(x) exceeds $2d$ keys". Note that not all nodes are in violation, and some nodes may have multiple violations.



2. **Exercise 10.1**, Ramakrishnan and Gehrke, subproblem 2 only. You only need to redraw relevant portions of the tree (i.e., modified nodes).

3. **Exercise 10.2, subproblems 1,3-5**, Ramakrishnan and Gehrke. For 3, only draw the relevant portions of the tree. Use the algorithms covered in class (splitting for inserts, redistribution and merge for deletes). The labels for the nodes begin with an "I" for all non-leaf nodes and an "L" for all leaf nodes.

4. **Exercise 10.7**, Ramakrishnan and Gehrke,

5. Recall that we discussed options for storing key-value pairs: alternative 1 used the index as the file organization itself (i.e., key-tuple pairs) while alternative 2 utilizes an auxiliary index of key-rid pairs on top of an existing file structure. If space utilization is of primary concern, why might one choose alternative 2 (index + heap file) over alternative 1 (index file)? Assume that the node utilization is the same in both cases.

Hash Indexes

6. **Exercise 11.1, subproblems 4-7**, Ramakrishnan and Gehrke.

7. **Exercise 11.7**, Ramakrishnan and Gehrke.

8. **Exercise 11.8**, Ramakrishnan and Gehrke.

9. For the following question, assume one block can fit either a) K key-value pairs plus 1 pointer or b) P pointers to other pages (i.e., `PageId`s):

   (a) For static hashing, what is the minimum number of pages needed to represent a hash table of size M? How about for extendible hashing?

   (b) For a global depth of G, what is the range of possible values for the *local depth* of any page containing entries? Is there a limitation (minimum or maximum) on the number of pages where the local depth equals the global depth?

   (c) For a global depth of G and local depth of L for a given page, how many directory entries share the given page (i.e., point to the same page)?

10. An administrator is most interested in avoiding worst-case behaviors. Why might this administrator use a histogram of key values when deciding whether to use a hash table index vs a B+ tree for equality searches?

Sorting

11. **Exercise 13.1**, Ramakrishnan and Gehrke. Note that the equations given in class were approximations; you should read Section 13.3 for the exact formulas for the number of passes and the number of sets in a run. **Please specify if you are using the book's algorithm or the class algorithm.** Please see the Piazza post: "Lab 6: Sorting question" for clarifications about this question.

SQL

12. **Exercise 5.2, subproblems 1-4**, Ramakrishnan and Gehrke.

13. **Exercise 5.4, subproblems 1-2**, Ramakrishnan and Gehrke. Note that there is a typo in the book: the Dept relation should appear as follows:

    Dept(*did*: integer, *dname*: string, *budget*: real, *managerid*: integer)