

Database and Information Systems

Homework 6

November 22, 2005; Due November 29 at 1:30 PM

Problem 1 [50 points]:

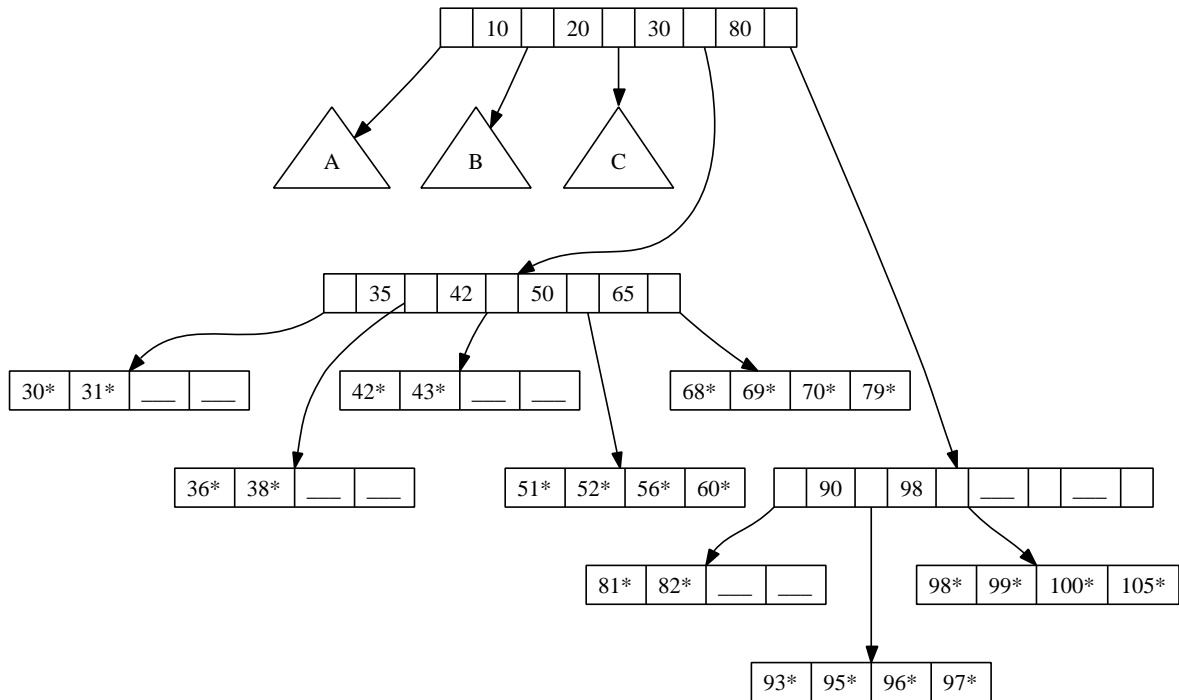


Figure 1: B+ Tree for Problem 1

Consider the B+ tree index shown in Figure 1, which uses alternative(1) for data entries. Each intermediate node can hold up to five pointers and four key values. Each leaf node can hold up to four records (indicated as a key value plus a “*”), and leaf nodes are doubly linked to their predecessor and successor nodes in the index (although this is not shown in the figure). The minimum number of keys allowed in an intermediate node is two, and the minimum number of records allowed in a leaf node is also two. In other words, the *order* of the tree is two. Underscores (“___”) indicate unallocated entries.

1. Insert a record with search key 94 into the tree.

2. Delete the record with search key 38 from the tree.

Problem 2 [50 points]: For this part of the assignment, you will need to run Oracle on *eniac*, using the `sql` command, as in previous assignments. The goal is to see the benefits of indexing in action.

The Oracle setup on *eniac* has a series of sample tables created for running database benchmarks — the so-called TPC-H benchmark. Additionally, each of these tables may have several alternative index structures. We will be comparing performance depending on which indices are used. To do this assignment, you will need to first enter the Oracle command `set timing on`, which will cause Oracle to report an “elapsed” time in seconds. You should run each query

The `lineitem` table has nearly 300,000 rows and takes approximately 37MB of space. We have two indices in `lineitem`, each with a name: `sys_c0013539` is an index over the attribute pair (`l_ordinal`, `l_linenum`) and `lineno` is on the singleton attribute (`l_linenum`).

1. Start by running the query:

```
select avg(l_linenum)
  from zives.lineitem
 where l_linenum >= 3;
```

which uses whatever indices Oracle prefers. Time the run and repeat it 5 times, averaging the total. What is that value?

2. Next, repeat the query with a special *optimizer hint* that says you want to scan the full table and avoid indices:

```
select /*+ full(l) */ avg(l_linenum)
  from zives.lineitem l
 where l_linenum >= 3;
```

Average the running time for 5 runs and report the result. What can you say about the difference between the two runs? Do these results suggest that Oracle uses an index by default?

3. Try another hint, which makes use of the first index:

```
select /*+ index(l sys_c0013539) */ avg(l_linenum)
  from zives.lineitem l
 where l_linenum >= 3;
```

Run 5 times and report the result. How did this compare to the first two cases? Did that particular index offer any benefits?

4. Which index, or indices, might have been used in the default, un-hinted Oracle query? Generalizing from the Oracle SQL hints shown above, what is the query hint that is equivalent to that default query?
5. For each of the two indices *and* for the non-indexed case, explain how what data and/or index structures will be examined in answering the query.
6. Run the two queries:

```
select /*+ index(l sys_c0013539) */ count(distinct l_suppkey)
  from zives.lineitem l
 where l_linenumber < 400;
```

```
select /*+ full(l) */ count(distinct l_suppkey)
  from zives.lineitem l
 where l_linenumber < 400;
```

Report the averaged running times over 5 runs. Explain the relative differences (or lack thereof) in performance, based on your knowledge of index and data layout characteristics, as well as your knowledge of “covering” indices.