Fall, 2003 CIS 550

Database and Information Systems

Homework 7

November 20, 2003; Due December 2 at 1:30 PM

Problem 1: Consider a relation with the following schema:

Employees(*eid*:integer,*ename*:string,*sal*:integer,*title*:string,*age*:integer)

Suppose that the following indexes, all using Alternative(2) (see page 276 or your slides) for data entries, exist: an unclustered B+ tree index on *sal*, a hash index on *age*, and a clustered B+ tree index on <*age,sal*>. Each Employees record is 100 bytes long, and you can assume that that each index data entry is 20 bytes long. The Employees relation contains 10,000 pages.

Suppose that, for each of the following selection conditions, you want to retrieve the average salary of qualifying tuples. For each selection condition describe the least expensive evaluation method and state its cost.

1. $sal > 100$
2. $sal > 200 \wedge age = 20$

Problem 2: Consider a relational algebra expression of the form $\sigma_c(\pi_l(R \times S))$. Suppose that the equivalent expression with selections and projections pushed as much as possible, taking into account only relational algebra equivalences, is in one of the following forms. In each case give an example relational expression using the form, filling in selection conditions and projection lists (instead of $c, l, c1, l1$,etc.).

1. *Equivalent maximally pushed form*: $\pi_{l1}(\sigma_{c1}(R) \times \sigma_{c2}(S))$.
2. *Equivalent maximally pushed form*: $\sigma_{c1}(\pi_{l1}(\sigma_{c2}(\pi_{l2}(R)) \times S))$.

Problem 3: Using Oracle on eniac (the `sql` command), we will have you specify query plans using the TPC-H benchmark data set of Homework 6. To do this assignment, you will need to repeat each query 5 times, time it with a stopwatch or the PC clock, and average the running times. (Averaging is necessary in case others are using eniac at the same time and affecting your running times.)

Start with the query

```
select count(*)
from zives.lineitem, zives.orders
where l_orderkey = o_orderkey
and o_totalprice < 1500.75
```

1. What index or indices would be most useful on the lineitem and orders tables?

2. As with before, we are going to use Oracle's *hints* to tell the optimizer how to run the query. Time the query to the nearest second:

```
select /*+ USE_HASH(l o) */ count(*)
from zives.lineitem l, zives.orders o
where l_orderkey = o_orderkey
and o_totalprice < 1500.75
```

3. Replace "USE_HASH" with "USE_MERGE" and repeat.

4. Replace with "USE_NL" and repeat.

5. Was there any substantive difference? Which plan or plans seemed to perform best? Explain why.