School of Computer Science, McGill University

COMP-421 Database Systems, Winter 2018

Written Assignment 3: Query Evaluation

Due Date April 05, 11:59pm

This is an individual assignment. You are required to work on your own to create the solution.

In this assignment, we evaluate queries for a manufacturer database. We look at three relations `Suppliers`, `Parts`, and `Catalog`, listed below.

Suppliers (sid:CHAR(10), sname:CHAR(40), address:CHAR(160), country:CHAR(20))
's3', 'BigSupp', '1180 Rue. St. Mathieu, Montreal', 'Canada'
Parts (pid:INT, pname:CHAR(20), color:CHAR(20), memo:CHAR(200))
'p2', 'bearing', 'brown', 'Memo for bearing'
Catalog (pid:INT, sid:CHAR(10), productionYear:INT, price:FLOAT)
$\longrightarrow$ pid references Parts, sid references Suppliers.
'p2', 's3', 2003, 205

INT and FLOAT have 10 Bytes, a char has 1 Byte. The relation `Suppliers` has around 5,000 tuples on 350 data pages. The relation `Parts` has 20,000 tuples that are stored on 1500 pages. Each part has on average 10 different suppliers, i.e., 10 catalog entries. Hence, `Catalog` has around 200,000 entries on 3000 pages.

For indexes, a single data entry might be spread over more than one leaf page. Assume there are 2000 different prices. Each rid has 10 Bytes, each pointer (of internal index pages) has 6 Bytes. Leaf pages are filled on an average 75%. An index page has 4KBytes (use 4000 Bytes). The root might have any fill factor. Assume that the root and intermediate pages are in memory.

Ex. 1 — (24 Points)

One typical query checks the `Catalog` table for a given part `X` and a range of price `Y`. `X` and `Y` represent parameters that might differ for each execution.

```
SELECT *
FROM  Catalog
WHERE pid = X AND price <= Y
```

Assume the price is uniformly distributed between $1 to $2000.

(a) Indicate the access costs (in number of pages retrieved leading to I/O) for this query in each of the scenarios: (Give the access costs both in general (for `X` and `Y`), and for the concrete values: X=1123 and Y=500.)
  (i) (4 Points) you do not use any index.
  (ii) (6 Points) you use an unclustered index on price.
  (iii) (6 Points) you use an unclustered index on pid.
  (iv) (4 Points) you use both indexes.

(b) (4 Points) Would a clustered index on either `price` or `pid` increase performance? Give a short explanation.

Ex. 2 — (20 Points)

To make your computation easier, you can assume an extra one or two buffer page if you want.
Now assume there is an index on pid on Parts. Calculate the estimated I/O and give an estimate of the number of output tuples for an:

(a)(6 Points) index nested loop join between `Parts` and `Catalog`.

(b)(6 Points) block nested loop join between `Parts` and `Catalog` and `Parts` is outer relation (50 buffer pages available).

(c)(8 Points) sort merge join between `Parts` and `Catalog` (50 buffer pages available).

Ex. 3 — (36 Points)

We will now have a look at the following query:

```
SELECT S.sname
FROM Suppliers S, Parts P, Catalogs C
WHERE  P.pname= 'bearing'
  AND  S.sid = C.sid
  AND  P.pid = C.pid
  AND  S.country = 'China'
```

A non-optimized relational expression for this query is:
$\pi_{sname}(\sigma_{country='China' \wedge pname='bearing'}(Suppliers \times Parts) \bowtie Catalog)$

(a)(6 Points) Perform an algebraic optimization of this expression according to the rules discussed in class. Do not consider any data cardinalities for this sub-question. Write down the resulting relational algebra expression (no need to draw any tree).

(b)(30 Points)

Now assume the only existing index is on sid of `Catalog`. Give an execution plan for your optimal expression and indicate how you are going to execute each of the operators. Draw the execution plan tree for this. You need not include number of rows and bytes passing through the operators in the tree that you are drawing.

Give rough estimations of I/O costs for your execution plan. Assume that you have 50 buffers available.

Note that our query optimizer, if it does not know how many different values exist for a given attribute, it assumes that there are 10 different values (uniformly distributed) by default. For instance, it assumes that suppliers come from 10 different countries.

Ex. 4 — (20 Points)

Consider the following query that calculates the distinct number of parts produced in each country.

```
SELECT S.country, COUNT(DISTINCT C.pid)
FROM Suppliers S, Catalog C
WHERE S.sid = C.sid
GROUP BY S.country
```

Find the optimal execution plan and its cost (you need not draw the algebraic optimization tree in the solution, but doing it might help you find the solution faster.).
Assume that both `Suppliers` and `Catalog` have indexes on their primary keys. For multi-attribute indexes, the index order is the same as the order in their table definition.
Assume you have 50 buffers. (Assume another extra buffer or two if it makes your computation easy).