INFO20003 Tutorial – Week 8

(Tutorial: Query optimisation)

Objectives:

This tutorial will cover:

I. Estimate cost of single-relation plans – 20 mins
II. Estimate cost of multi-relation plans – 35 mins

Exercises:

1. Single-relation plans:

Consider a relation with this schema:

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

Suppose that the following indexes exist:

- An unclustered hash index on *eid*
- An unclustered B+ tree index on *sal*
- An unclustered hash index on *age*
- A clustered B+ tree index on (*age*, *sal*)

The Employees relation contains 10,000 pages and each page contains 20 tuples. Suppose there are 500 index pages for B+ tree indexes and 500 index pages for hash indexes. There are 40 distinct values of *age*, ranging from 20 to 60, in the relation. Similarly, *sal* ranges from 0 to 50,000 and there are up to 50,000 distinct values. *eid* is a candidate key; its value ranges from 1 to 200,000 and there are 200,000 distinct values.

For each of the following selection conditions, compute the Reduction Factor (selectivity) and the cost of the *cheapest* access path for retrieving all tuples from Employees that satisfy the condition:

a. sal > 20,000
b. age = 25
c. age > 30
d. eid = 1000
e. sal > 20,000 ∧ age > 30

2. Multi-relation plans:

Consider the following schema:

Emp (<u>eid</u>, sal, age, $\overset{FK}{\text{did}}$)

Dept (<u>did</u>, $\overset{FK}{\text{projid}}$, budget, status)

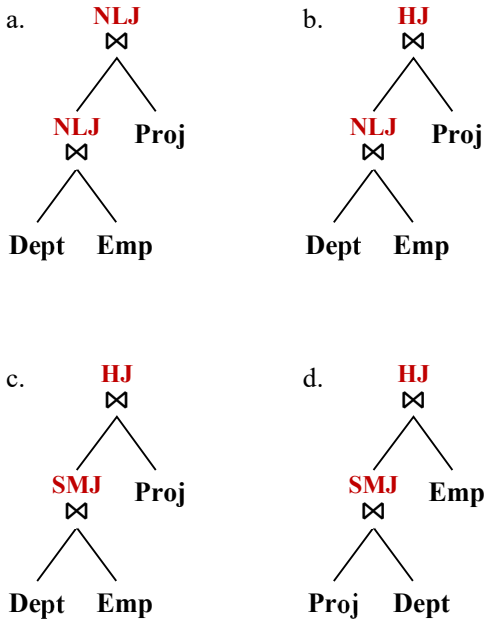
Proj (<u>projid</u>, code, report)

The number of tuples in Emp is 20,000 and each page can hold 20 records. The Dept relation has 5000 tuples and each page contains 40 records. There are 500 distinct *did*s in Dept. One page can fit 100 resulting tuples of Dept JOIN Emp. Similarly, Proj has 1000 tuples and each page can contain 10 tuples. Assuming that *projid* is the candidate key of Proj, there can be 1000 unique values for *projid*. Sort-Merge Join can be done in 2 passes. Let's assume that, if we join Proj with Dept, 50 resulting tuples will fit on a page. NLJ in this question means 'Page oriented NLJ'.

Consider the following query:

```
SELECT E.eid, D.did, P.projid
FROM Emp AS E, Dept AS D, Proj AS P
WHERE E.did = D.did
  AND D.projid = P.projid;
```

For this query, estimate the cost of the following plans, focusing on the join order and join types:



Take Home Questions:

1. Multi Relation Plans with Access Methods:

Consider the following schema:

Student (<u>studentid</u>, name, dob, degreename)

StudentSubject (<u>studentid</u>, subjectid, grade)

Subject (subjectid, name, level, coordinatorname, budget)

The number of tuples in Student is 20,000 and each page can hold 20 records. The StudentSubject relation has 50,000 tuples and each page contains 50 records. Subject has 1,000 tuples and each page can contain 10 records. One page can fit 100 resulting tuples of Student JOIN StudentSubject. 100 tuples resulting from the join of StudentSubject and Subject also fit onto a page. Assume that Subject.subjectid and Student.studentid are candidate keys. Sorting can be done in 2 passes.

There are 3 available indexes: an unclustered hash index on Student(degreename), an unclustered B+ tree index on Subject(level), and a clustered B+ index on StudentSubject(studentid). All indexes have 50 pages.

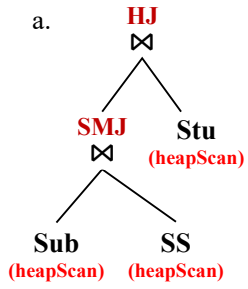
There are 10 distinct values for Subject.level, ranging from 1-10. There are known to be 40 distinct values for Student.degreename.

Consider the following query:

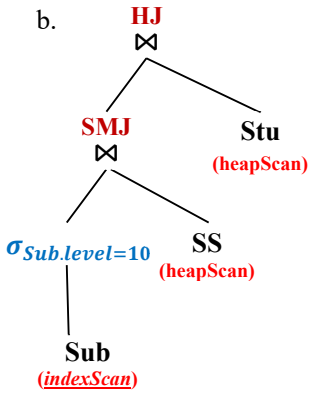
```
SELECT Stu.studentid, Sub.subjectid
FROM Student AS Stu, Subject AS Sub, StudentSubject AS SS
WHERE Stu.studentid = SS.studentid
  AND SS.subjectid = Sub.subjectid
  AND Stu.degreename = 'CompSci'
  AND Sub.level = 10
```

For this query, estimate the cost of the following plans. If selections are not marked on the tree, assume that they are done on the fly (in memory) **after** having completed all joins.

a.



b.



c.