Assignment 3- INFO20003 Semester 1 2018

Date Due: Friday 20th October 2017 11:59pm AEST

Late submissions will not be assessed unless you have an Academic Adjustment plan or written approval from the INFO20003 Subject Coordinator.

Submission Attempts: 3 – (Only the last submitted assignment will be assessed)

Weighting: 10% of your total assessment

Question 1 (5 marks)

Consider two relations A and B. A has 10,000 tuples, and B has 200,000 tuples. Both relations store 100 tuples per a page.

Consider the following SQL statement:

*SELECT **

FROM A, B

WHERE A.a = B.a;

We wish to evaluate an equijoin between A and B, with an equality condition A.a = B.a. There are 52 buffer pages available for this operation. Both relations are stored as (unsorted) heap files. Neither relation has any indexes built on it. Assume that Sort-Merge Join could be done in 2 passes.

Consider alternative join strategies described below and calculate the cost of each alternative. Evaluate the algorithms using the number of disk I/O's (i.e. pages) as the cost. For each strategy, provide the formulae you use to calculate your cost estimates.

NPages(A) = 10,000/100 = 100, NPages(B)= 200,000/100 = 2000

a) Page-oriented nested loops join. Consider A as the outer relation. **(1 mark)**

Total Cost= (# of pages in outer) + (# of pages in outer) *(# of pages in inner)

Total Cost = 100 + 100*2000 = 200,100

b) Block-oriented nested loops join. Consider A as the outer relation. **(1 mark)**

# of blocks= ceil (# of pages in outer/ (Number of Buffers-2))= ceil (100/50)=2

Total Cost= (# of pages in outer) + (# of blocks)*(# of pages in inner)

Total Cost = 100 + 2*2,000 = 4,100

c) Sort-Merge join **(1 mark)**

Cost of sorting A = 2*2*100 =400

Cost of sorting B = 2*2*2000 = 8000

Total cost = 400+8,000+100+2,000 = 10,500

d) Hash Join (1 mark)

Total cost = 3*(100 +2000) = 6,300

e) What would the lowest possible I/O cost be for joining A and B using any join algorithm and how much buffer space would be needed to achieve this cost? Explain briefly. **(1 mark)**

Min cost = 100+ 2000 = 2,100 for Buffer size = 100 + 2 = 102 pages

Question 2 (5 marks)

Consider a relation with the following schema:

Managers (id: integer, name:string, title:string, level: integer)

The Managers relation consists of 500,000 tuples stored in disk pages. The relation is stored as simple heap file and each page stores 100 tuples. Possible titles in the Managers hierarchy are “CFO”, “CEO”, “CTO”, “Architect”, and “Team Lead”. Manager levels are ranging from 0-20.

Suppose that the following SQL query is executed frequently using the given relation:

SELECT name
FROM Managers
WHERE title = “Architect” and level > 18;

Your job is to analyse the access paths given below and estimate the cost of the best access path utilizing the information given about different indexes in each part.

a) Compute the estimated result size and the reduction factors (selectivity) of this query. **(1 mark)**

RF(title) = 1/5=0.2

RF(level) = (20-18)/(20-0)= 0.1

Result size = num tuples * RF (title) * RF (level)

Result size = 500,000 * 0.2 * 0.1 =10,000 tuples

b) Compute the estimated cost of the best access path assuming that a *clustered B+ tree* index on *(title, level)* is (the only index) available. Suppose there are 200 index pages. Discuss and calculate alternative access paths. **(1 mark)**

Cost of full scan = 500,000/100 = 5000 I/O
Cost of index scan = (200 + 5000)*0.1*0.2 = 104 I/O
The cost of the best access path here is 104 I/O.

c) Compute the estimated cost of the best access path assuming that an *unclustered B+ tree* index on *(level)* is (the only index) available. Suppose there are 200 index pages. Discuss and calculate alternative access paths. **(1 mark)**

Cost of full scan = 500,000/100 = 5000 I/O
Cost of index scan = (200 + 500,000)*0.1 = 50,020 I/O
The cost of the best access path here is 5000 I/O which is the cost of full table scan.

d) Compute the estimated cost of the best access path assuming that an *unclustered Hash* index on *(title)* is (the only index) available. Discuss and calculate alternative access paths. **(1 mark)**

Cost of full scan = 500,000/100 = 5000 I/O
Cost of hash index = 500,000*0.2*2.2 = 220,000 I/O
The cost of the best access path here is 5000 I/O which is the cost of full table scan.

e) Compute the estimated cost of the best access path assuming that an *unclustered Hash* index on *(level)* is (the only index) available. Discuss and calculate alternative access paths. **(1 mark)**

Cost of full scan = 500,000/100 = 5000 I/O
The hash index is not applicable here, because we have a range query on level!
The cost of the best plan here is 5000 I/O which is the cost of full table scan.

Question 3 (10 marks)

Consider the following relational schema and SQL query. The schema captures information about employees, departments, and company finances. The finance sector is

organized on a per department basis, i.e. each department has its own financial budget (and thus the corresponding record in the Finance relation).

Emp(eid: integer, did: integer, sal: integer, hobby: char(20))
Dept(did: integer, dname: char(20), floor: integer, phone: char(10))
Finance(did: integer, budget: real, sales: real, expenses: real)

Consider the following query:
SELECT D.dname, F.budget
FROM Emp E, Dept D, Finance F
WHERE E.did=D.did AND D.did=F.did
AND E.sal ≥ 90000 AND E.hobby = 'bungee jumping;

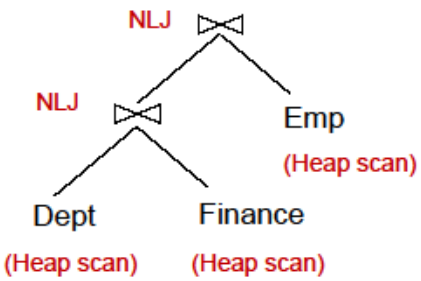
The system's statistics indicate that employee salaries range from 50,000 to 100,000, and employees enjoy 200 different hobbies. There are a total of 10,000 employees and 5,000 departments (each with corresponding financial record in the Finance relation) in the database. Each relation fits 100 tuples in a page. Suppose there exists a *clustered B+ tree* index on *(Emp.did)* of size 50 pages. Since selection over filtering predicates is not marked in the plans, assume it will happen later on-the-fly after all joins are performed.

a) Compute the estimated result size and the reduction factors (selectivity) of this query **(2 marks)**

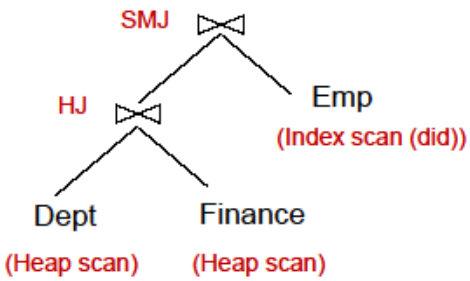
Reduction factor (sal) = 10000/50000 = 0.2
Reduction factor (hobby) = 1/200 = 0.005
Reduction factor (E.did = D.did) = 1/5000
Reduction factor (D.did = F.did) = 1/5000
Result size = 10,000*5000*5000*0.2*0.005*1/5000 * 1/5000= 10

b) Compute the cost of the plans shown below. Assume that sorting of any relation (if required) can be done in *2 passes.* NLJ is a *Page-oriented* Nested Loops Join. Assume that *did* is the candidate key, and that 100 tuples of a resulting join between Emp and Dept fit in a page. Similarly, 100 tuples of a resulting join between Finance and Dept fit in a page.

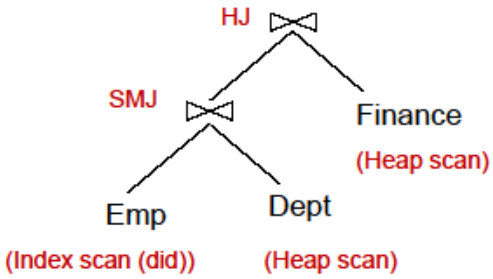
1)



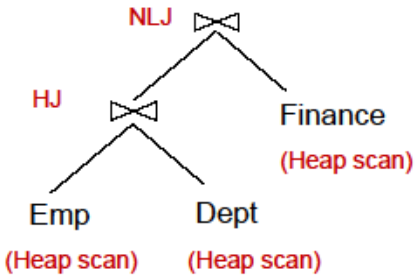
2)



3)



4)



B1)

Number of pages of Emp = 10,000/100 = 100 pages

Number of pages of Dept = 5000/100 = 50 pages

Number of pages of Finance = 5000/100 = 50 pages

Number of resulting tuples for Dept JOIN Finance = (1/5000)*(5000*5000) = 5000 tuples

Number of pages for Dept JOIN Finance = 5000 / 100 = 50 pages

Cost of scanning Dept = 50 pages

Cost to join with Finance = 50* 50 = 2500

Cost to Join with Emp = 50 * 100 = 5,000

Total Cost = 50 + 2500 + 5,000 = 7,550

B2)

Number of pages of Emp = 50,000/100 = 100 pages

Number of pages of Dept = 5000/100 = 50 pages

Number of pages of Finance = 5000/100 = 50 pages

Number of resulting tuples for Dept JOIN Finance = (1/5000)*(5000*5000) = 5000 tuples

Number of pages for Dept JOIN Finance = 5000 / 100 = 50 pages

Cost of scanning Dept = 5000/100 = 50 pages

Cost to join with Finance = 2*50 + 3*50 = 250 pages

Cost to sort Dept JOIN Finance = 3*50 = 150 pages **(note first read is already done)**

Emp already sorted on DID no sorting cost

Cost of Emp = 50 + 100 pages

Cost to join with Emp = 50 + 150 = 200 pages

Total Cost = 50 + 250 + 150+ 50 + 150= 650 pages

B3)

Number of pages of Emp = 10,000/100 = 100 pages

Number of pages of Dept = 5000/100 = 50 pages

Number of pages of Finance = 5000/100 = 50 pages

Number of resulting tuples for Emp JOIN Dept = (1/5000)*(10,000*5000) = 10,000 tuples

Number of pages for Emp JOIN Dept = 10,000 / 100 = 100 pages

Cost to scan Emp with index scan = 50+100 = 150 pages

Cost to sort Dept = 2*2* 50 = 200 pages

Cost to join with Dept = 50 pages

Cost to join with Finance = 2*100 + 3* 50 = 350 pages

Total Cost = 150 + 250 + 350 = 750 pages

B4)

Number of pages of Emp = 10,000/100 = 100 pages

Number of pages of Dept = 5000/100 = 50 pages

Number of pages of Finance = 5000/100 = 50 pages

Number of resulting tuples for Emp JOIN Dept = (1/5000)*(10,000*5000) = 10,000 tuples

Number of pages for Emp JOIN Dept = 50,000 / 100 = 100 pages

Cost of scanning Emp = 100 pages

Cost to join with Dept = 2*100 + 3* 50 = 350 pages

Cost to join with Finance = 100*50 = 5,000 pages

Total cost = 100 + 350 +5,000 =5,450 pages