INFO20003 Database Systems

Week 8

Reduction factor

1. **Reduction factor (Selectivity)**
   a. Col = value

      RF = 1/NKeys(Col)

   b. Col > value

      RF = (High(Col) – value) / (High(Col) – Low(Col))

   c. Col < value

      RF = (val – Low(Col)) / (High(Col) – Low(Col))

   d. Col_A = Col_B (for joins)

      RF = 1/ (Max (NKeys(Col_A), NKeys(Col_B)))

   e. In no information about NKeys or interval, use a “magic number” 1/10

      RF = 1/10

Costs

3. **Indexing Cost**
    a. B+-tree index
        i. Just a single tuple (selection over a primary key)

        Cost = Height(I) + 1

        ii. Clustered index (multiple tuples)

        Cost = (NPages(I)+NPages(R)) * Π RF_i

        iii. Unclustered (multiple tuples)

        Cost = (NPages(I)+NTuples(R)) * Π RF_i

    b. Hash Index
        i. Just a single tuple (selection over a primary key)

        Cost = 1.2 + 1 = 2.2

        ii. Clustered index (multiple tuples)

        Cost = (NPages (R)) * Π RF_i * 2.2

        iii. Unclustered index (multiple tuples)

        Cost = (NTuples(R)) * Π RF_i * 2.2

4. **Sequential Scan (i.e. Heap Scan) Cost**

    Cost = NPages(R)

Q1) Single-relation plans

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

RMB: full table scan is always a possible access path

Information

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.

E:

- NP = 10,000
- NT = 200,000
- age in [20, 60] (nKeys = 40)
- Sal in [0, 50000] (nKeys = 50,000)
- Eid in [1, 200000] (nKeys = 200,000)

B+tree index: NP(I) = 500
Hash index: NP(I) = 500

Q1a)

E:
- NP = 10,000
- NT = 200,000
- age in [20, 60] (nKeys = 40)
- Sal in [0, 50000] (nKeys = 50,000)
- Eid in [1, 200000] (nKeys = 200,000)

B+tree index: NP(I) = 500
Hash index: NP(I) = 500

- ~~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*)~~

a. sal > 20,000

The reduction factor (RF) is

$$\text{RF} = \frac{\text{High}(I) - \text{value}}{\text{High}(I) - \text{Low}(I)} = \frac{50{,}000 - 20{,}000}{50{,}000 - 0} = 0.6$$

There are two possible access paths for this query:

- The unclustered B+ tree index on *sal*,
- Full table scan, with cost 10,000 I/Os.

$$\begin{aligned}\text{Cost} &= \text{product of RFs of matching selects} \times (\text{NTuples}(R) + \text{NPages}(I)) \\ &= 0.6 \times \big((20 \times 10{,}000) + 500\big) \\ &= 120{,}300 \text{ I/Os}\end{aligned}$$

Q1b)

E:
- NP = 10,000
- NT = 200,000
- age in [20, 60] (nKeys = 40)
- Sal in [0, 50000] (nKeys = 50,000)
- Eid in [1, 200000] (nKeys = 200,000)

B+tree index: NP(I) = 500
Hash index: NP(I) = 500

- ~~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*)

b. age = 25

The reduction factor is

$$RF = \frac{1}{NKeys(I)} = \frac{1}{40}$$

3 possible access paths:
- Unclustered hash index on age
- Clustered B+ tree index on (age, sal)
- Full table scan

Q1b)

- The clustered B+ tree index on (*age*, *sal*), with cost

$$\text{Cost} = \text{product of RFs of matching conditions} \times (\text{NPages}(R) + \text{NPages}(I))$$
$$= \frac{1}{40} \times (500 + 10{,}000)$$
$$= 263 \text{ I/Os approx.}$$

- The unclustered hash index on *age*, with cost

$$\text{Cost} = \text{product of RFs of matching conditions} \times \text{hash lookup cost} \times \text{NTuples}(R)$$
$$= \frac{1}{40} \times 2.2 \times (20 \times 10{,}000)$$
$$= 11{,}000 \text{ I/Os}$$

- Full table scan, with cost 10,000 I/Os.

Q1c)

E:
- NP = 10,000
- NT = 200,000
- age in [20, 60] (nKeys = 40)
- Sal in [0, 50000] (nKeys = 50,000)
- Eid in [1, 200000] (nKeys = 200,000)

B+tree index: NP(I) = 500
Hash index: NP(I) = 500

- ~~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*)

c. $age > 30$

The reduction factor is

$$RF = \frac{High(I) - value}{High(I) - Low(I)} = \frac{60 - 30}{60 - 20} = 0.75$$

There are two possible access paths for this query:

- The clustered B+ tree index on (*age*, *sal*),

Cost = product of RFs of matching conditions × (NPages(*R*) + NPages(*I*))
= 0.75 × (500 + 10,000)
= 7875 I/Os

- Full table scan, with cost 10,000 I/Os.

Q1d)

E:
- NP = 10,000
- NT = 200,000
- age in [20, 60] (nKeys = 40)
- Sal in [0, 50000] (nKeys = 50,000)
- Eid in [1, 200000] (nKeys = 200,000)

B+tree index: NP(I) = 500
Hash index: NP(I) = 500

- 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*)~~

d. eid = 1000

- RF = 1/200000

Cost = hash lookup cost + 1 data page access = 1.2 + 1 = 2.2

: full table scan (cost 10,000).

Q1e)

E:
- NP = 10,000
- NT = 200,000
- age in [20, 60] (nKeys = 40)
- Sal in [0, 50000] (nKeys = 50,000)
- Eid in [1, 200000] (nKeys = 200,000)

B+tree index: NP(I) = 500
Hash index: NP(I) = 500

- ~~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*)

e. $sal > 20{,}000 \wedge age > 30$

$$RF_{age} = \frac{High(I) - value}{High(I) - Low(I)} = \frac{60 - 30}{60 - 20} = 0.75$$

$$RF_{sal} = \frac{High(I) - value}{High(I) - Low(I)} = \frac{50{,}000 - 20{,}000}{50{,}000 - 0} = 0.6$$

3 possible access paths:
- Unclustered B+ tree index on sal
- Clustered B+ tree index on (age, sal)
- Full table scan

Q1e)

- The unclustered B+ tree index on *sal*, with cost

$$\begin{aligned}\text{Cost} &= \text{product of RFs of } \textbf{matching} \text{ conditions} \times (\text{NTuples(R)} + \text{NPages(I)}) \\ &= 0.6 \times \big((20 \times 10{,}000) + 500\big) \\ &= 120{,}300 \text{ I/Os (same as part a)}\end{aligned}$$

- The clustered B+ tree index on (*age*, *sal*), with cost

$$\begin{aligned}\text{Cost} &= \text{product of RFs of matching conditions} \times (\text{NPages(R)} + \text{NPages(I)}) \\ &= 0.75 \times 0.6 \times (10{,}000 + 500) \\ &= 4725 \text{ I/Os}\end{aligned}$$

- Full table scan, with cost 10,000 I/Os.

Joins

5. **Joins (between relations R and S, R = outer, S = inner) Cost**
   a. NLJ
      i. Tuple-oriented NLJ

         Cost = NPages(R) + NTuples(R) * NPages(S)

      ii. Page-oriented NLJ

         Cost = NPages(R) + NPages (R) * NPages(S)

      iii. Block-oriented NJL (for block_size B)

         Cost = NPages(R) + ceil(NPages (R)/(B-2)) * NPages(S)

   b. Hash Join

      Cost = 3*(NPages(R) + NPages(S))

   c. Sort-Merge Join

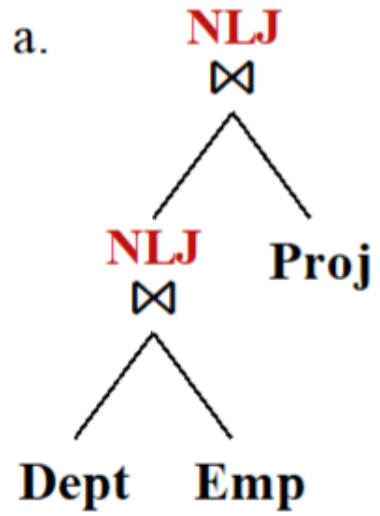
      $Cost_{SMJ}$ = NPages(R) + NPages(S) +

      2* NPages(R)* num_passes(R) +

      2* NPages(S)* num_passes(S)

Pipeline

- Pipelining = the direct “streaming” in memory of the output of one operation as the input of another operation, without writing the output to disk



i.e.
After gaining an output of the first NLJ, the output is used by the next operations before writing to the disk

This saves the cost of reading (Dept X Emp) by once during the second NLJ

Q2) Multi-relation plans

2. Multi-relation plans:

Consider the following schema:

Emp (<u>eid</u>, sal, age, did) (FK)

Dept (<u>did</u>, <u>projid</u>, budget, status) (FK)

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

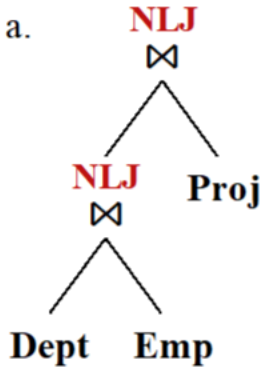
The number of tuples in Emp is 20,000 and each page can hold 20 records. The Dept relation has 5000 tuples and each page contains 40 records. There are 500 distinct *did*s in Dept. One page can fit 100 resulting tuples of Dept JOIN Emp. Similarly, Proj has 1000 tuples and each page can contain 10 tuples. 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:

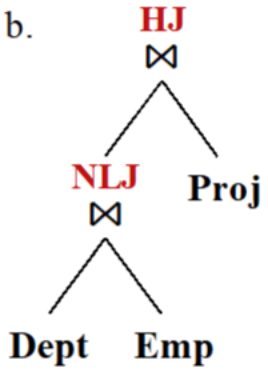
```
SELECT E.eid, D.did, P.projid
FROM Emp AS E, Dept AS D, Proj AS P
WHERE E.did = D.did
  AND D.projid = P.projid;
```

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

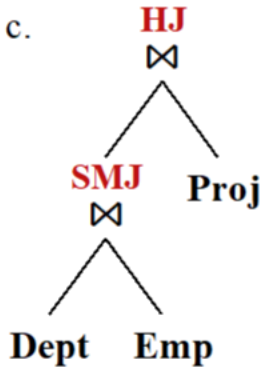
a.



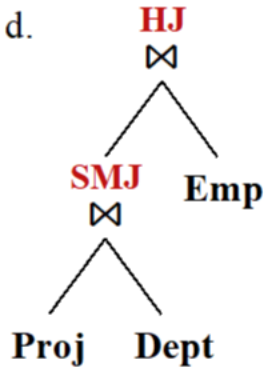
b.



c.



d.



Information

2. Multi-relation plans:

Consider the following schema:

Emp (eid, sal, age, did) (FK)

Dept (did, projid, budget, status) (FK)

Proj (projid, 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'.

Emp:
Ntuples = 20,000
Npages = 1000

Dept:
Ntuples = 5000
Npages = 125
Nkeys(did) = 500

Proj:
Ntuples = 1000
Npages = 100
Nkeys(projid) = 1000

- NT/P(DxE) = 100
- NT/P(PxD) = 50
- 2 passes

Q2a)

Emp:
Ntuples = 20,000
Npages = 1000

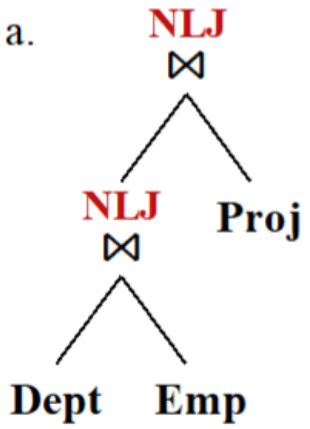
Dept:
Ntuples = 5000
Npages = 125
Nkeys(did) = 500

Proj:
Ntuples = 1000
Npages = 100
Nkeys(projid) = 1000

- NT/P(DxE) = 100
- NT/P(PxD) = 50
- 2 passes

```
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;
```

a.



$Cost(D_{NLJ}E) = Npages(D) + Npages(D) * Npages(E) = 125{,}125$ I/O

$$size(D_{NLJ}E) = RF \times NTuples(Outer) \times NTuples(Inner)$$
$$= \frac{1}{\max(NKeys(E.did), NKeys(D.did))} \times NTuples(Dept) \times NTuples(Emp)$$
$$= \frac{1}{NKeys(D.did)} \times NTuples(Dept) \times NTuples(Emp)$$
$$= \frac{1}{500} \times 5000 \times 20{,}000$$
$$= 200{,}000 \text{ tuples} \quad = 2000 \text{ pages}$$

$Cost(_{NLJ}P) = Npages(D_{NLJ}E) * Npages(P) = 200{,}000$ I/O

Total cost = 125,125 + 200,000 = **325,125** I/O

Q2b)

Emp:
Ntuples = 20,000
Npages = 1000

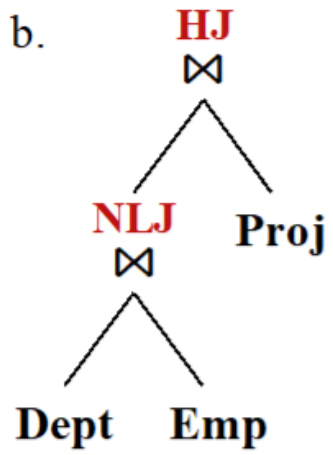
Dept:
Ntuples = 5000
Npages = 125
Nkeys(did) = 500

Proj:
Ntuples = 1000
Npages = 100
Nkeys(projid) = 1000

- NT/P(DxE) = 100
- NT/P(PxD) = 50
- 2 passes

```
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;
```

b.



$Cost(D_{NLJ}E) = Npages(D) + Npages(D) * Npages(E) = 125{,}125$ I/O

$$size(D_{NLJ}E) = \frac{1}{NKeys(I)} \times NTuples(Dept) \times NTuples(Emp)$$
$$= \frac{1}{500} \times 5000 \times 20{,}000$$
$$= 200{,}000 \text{ tuples} = 2000 \text{ pages}$$

$Cost(_{HJ}P) = 3 * (Npages(D_{NLJ}E) + Npages(P)) - 2000 = 4300$ I/O

Total cost = 125,125 + 4300 = **129,425** I/O

Q2c)

Emp:
Ntuples = 20,000
Npages = 1000

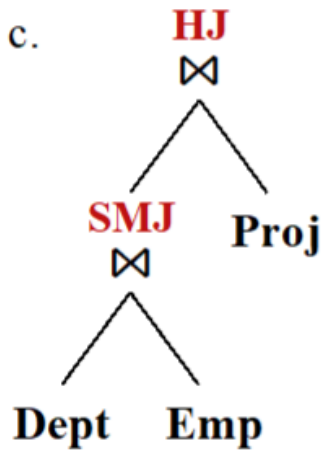
Dept:
Ntuples = 5000
Npages = 125
Nkeys(did) = 500

Proj:
Ntuples = 1000
Npages = 100
Nkeys(projid) = 1000

- NT/P(DxE) = 100
- NT/P(PxD) = 50
- 2 passes

```
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;
```

c.



$Cost(D_{SMJ}E) = 2 * NPasses * Npages(D) + 2 * Npasses * Npages(E) + Npages(D) + Npages(E)$
$= 500 + 4000 + 125 + 1000 = 5625$ I/O

$$size(D_{SMJ}E) = \frac{1}{NKeys(I)} \times NTuples(Dept) \times NTuples(Emp)$$
$$= \frac{1}{500} \times 5000 \times 20{,}000$$
$$= 200{,}000 \text{ tuples} = 2000 \text{ pages}$$

$Cost(_{HJ}P) = 3 * (Npages(D_{SMJ}E) + Npages(P)) - 2000 = 4300$ I/O

Total cost = 5625 + 4300 = **9925** I/O

Q2d)

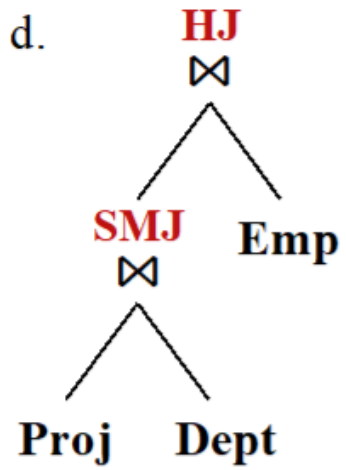
Emp:
Ntuples = 20,000
Npages = 1000

Dept:
Ntuples = 5000
Npages = 125
Nkeys(did) = 500

Proj:
Ntuples = 1000
Npages = 100
Nkeys(projid) = 1000

- NT/P(DxE) = 100
- NT/P(PxD) = 50
- 2 passes

```
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;
```



$Cost(P_{SMJ}D) = 2 * NPasses * Npages(P) + 2 * Npasses * Npages(D) + Npages(P) + Npages(D)$
$= 400 + 500 + 100 + 125 = 1125$ I/O

$$size(P_{SMJ}D) = \frac{1}{NKeys(I)} \times NTuples(Proj) \times NTuples(Dept)$$
$$= \frac{1}{1000} \times 1000 \times 5000$$
$$= 5000 \text{ tuples} \quad = 100 \text{ pages}$$

$Cost(_{HJ}E) = 3 * (Npages(P_{SMJ}D) + Npages(E)) - 100 = 3200$ I/O

Total cost = 1125 + 3200 = **4325** I/O

More practice – Take Home Questions

On tutorial sheet