Database Management Systems

A Hand's-On Example for Query Plan Cost Estimation

CMPUT 391: Query Processing & Optimization

Dr. Osmar R. Zaïane

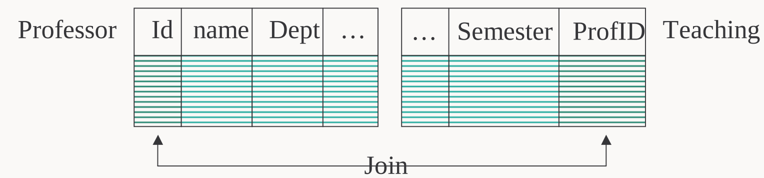
University of Alberta

The Query

Consider:

```
SELECT P.name
FROM Professor P, Teaching T
WHERE P.Id = T.ProfID
    AND T.Semester = 'F2000'
    AND P.Dept = 'CS'
```

Find the names of professors from Computing Science who taught a course in the Fall of 2000



The Query in RA

SQL

```
SELECT P.name
FROM Professor P, Teaching T
WHERE P.Id = T.ProfID
    AND T.Semester = 'F2000'
    AND P.Dept = 'CS'
```

Possible Relational Algebra Expressions

A $\pi_{name}(\sigma_{Dept=\text{'CS'} \wedge Semester=\text{'F2000'}}(Professor \bowtie_{Id=ProfID} Teaching))$

B $\pi_{name}(\sigma_{Dept=\text{'CS'}}(Professor) \bowtie_{Id=ProfID} \sigma_{Semester=\text{'F2000'}}(Teaching))$

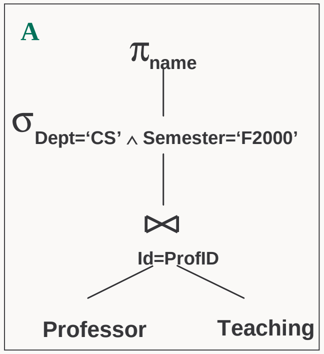
C $\pi_{name}(\sigma_{Semester=\text{'F2000'}}(\sigma_{Dept=\text{'CS'}}(Professor) \bowtie_{Id=ProfID} Teaching))$

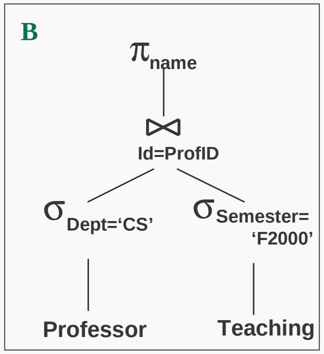
D $\pi_{name}(\sigma_{Dept=\text{'CS'}}(Professor \bowtie_{Id=ProfID} \sigma_{Semester=\text{'F2000'}}(Teaching)))$

Query Trees

A $\pi_{name}(\sigma_{Dept=\text{'CS'} \wedge Semester=\text{'F2000'}}(Professor \bowtie_{Id=ProfID} Teaching))$

B $\pi_{name}(\sigma_{Dept=\text{'CS'}}(Professor) \bowtie_{Id=ProfID} \sigma_{Semester=\text{'F2000'}}(Teaching))$

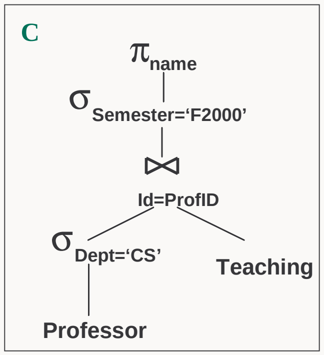


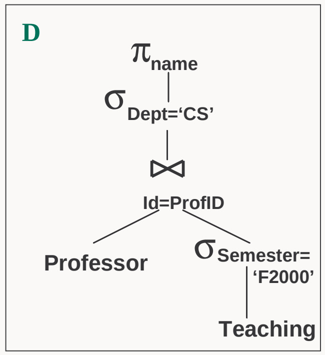


Query Trees

C $\pi_{name}(\sigma_{Semester='F2000'}(\sigma_{Dept='CS'}(Professor) \bowtie_{Id=ProfID} Teaching))$

D $\pi_{name}(\sigma_{Dept='CS'}(Professor \bowtie_{Id=ProfID} \sigma_{Semester='F2000'}(Teaching)))$



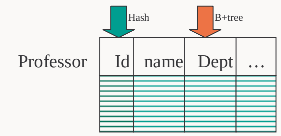


Data Dictionary and Indexes

Professor

200 pages
1000 records on professors in 50 Departments
That is 5 tuples per page

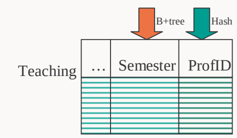
Clustered 2-level B$^+$ tree index on Dept
Hash index on Id



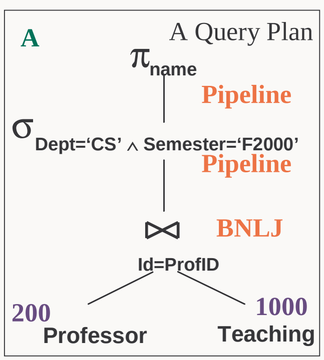
Teaching

1000 pages
10000 teaching records for the period of 4 semesters
That is 10 tuples per page

Clustered 2-level B$^+$ tree index on Semester
Hash index on ProfID

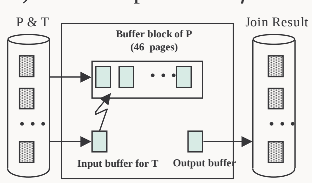


Cost Estimation for A



We have **48** buffer blocks in main memory

We will use 1 block for the output; 1 block for the input of *Teaching*; and the rest (46 blocks) for the input of *Professor*.



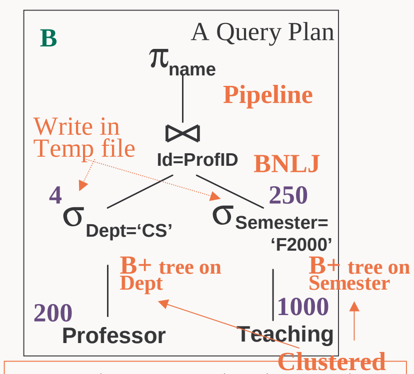
- We need to read *Professor* once ➔ 200 I/O
- We need to read *Teaching* 5 times

This is because with 46 blocks, we need to fill the buffers ⌈200/46⌉=5 times to read the whole *Professor* table. Each time we fill the buffer, we scan *Teaching*.

- There is no cost for selecting and projecting.

Cost = 200 + 5000 = **5200 I/O**
200 for reading Professor
5000 for reading Teaching 5 times

Cost Estimation for B



Cost = (4 + 250 + 4) + (4 + 250) + (4 + 250) = **766 I/O**
258 for the select; 254 for writing
The temporary files; 250 for the join

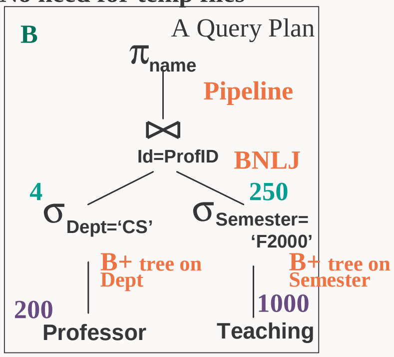
Accessing the indices:
Both indices are 2-level B+ trees. This means we need 2 I/Os for each ➔ 4 I/Os

Estimating the sizes of the selections:
- There are 1000 professors in 50 departments. Assuming a uniform distribution, CS would have 1000/50=20 professors. (4 pages) + Writing temporary file (4 pages)
- There are 10000 teachings in 4 semester. Assuming a uniform distributions, there would be 10000/4=2500 teachings in the Fall of 2000. (250 pages) + writing temp file (250 pages)

Block nested-loop Join:
All records of professors in CS fit in the buffer. We would scan the teaching of F2000 only once. ➔ 4 + 250 I/Os

Cost Estimation for B (cont')

If DBMS is smart:
No need for temp files

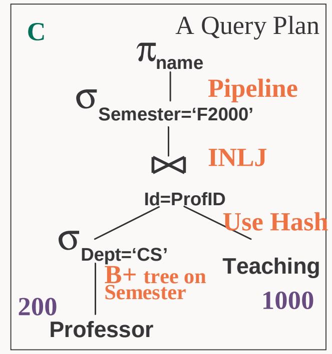


Cost = (4 + 250 + 4) = **258 I/O**
258 for the select.
the join is on the fly.

<u>Accessing the indices:</u>
Both indices are 2-level B+ trees. This means we need 2 I/Os for each ➔ 4 I/Os

<u>Estimating the sizes of the selections and BNLJ:</u>
•There are 1000 professors in 50 departments. Assuming a uniform distribution, CS would have 1000/50=20 professors. (4 pages)
• Since the result can fit in main memory, the result of the second select can play the role of the scan. As the clustered Teachings of F2000 are read, they are joined to the professor records in the buffer.
•There are 10000 teachings in 4 semester. Assuming a uniform distributions, there would be 2500 teachings in the Fall of 2000. (250 pages)

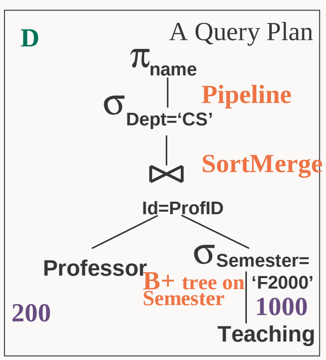
Cost Estimation for C



Cost = 2 + 4 + 224 = **230 I/O**
6 for selecting Professor
224 for the join

•The index is a 2-level B+ trees. This means we need 2 I/Os to access the CS professors.
•There are 1000 professors in 50 departments. Assuming a uniform distribution, CS would have 1000/50=20 professors. Since the records are clustered, we would need only 4 I/Os.
• The selection result is piped as input to the join.
•To match the 20 professor, we need to search the index 20 times. Thus, accessing the index costs 20 * 1.2 = 24 I/O
•Again, assuming uniform distribution, each professor teaches 10 teachings (10000/1000). Since the index is not clustered, we need 10 I/Os per professor to get all teachings. That is 200 I/Os in total (20*10) for the teachings of all professors in CS.

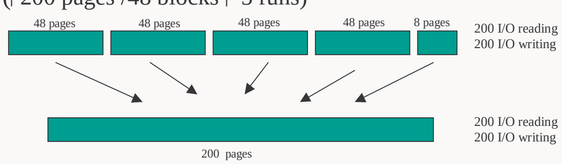
Cost Estimation for D



Cost of sort is 800 I/O so far.

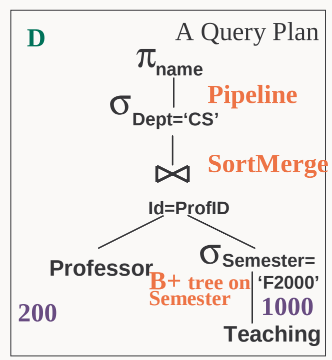
We have **48** buffer blocks in main memory

• We need to sort Professors and the teachings in the Fall of 2000. The teachings can be sorted as they are selected. While in main memory, the first runs can be produced. ➔ No temporary files for the selection.
• Sorting *Professor* with a k-way sort:
since we have 48 blocks, this allows us to get 5 runs (⌈200 pages /48 blocks⌉=5 runs)



➔ Sorting *Professor* costs us 2 * 200 + 2 * 200 = 800 I/Os.

Cost Estimation for D (cont')



Cost =800+1002+450=**2252 I/O**
800 for sorting P
1002 for selecting and sorting T
450 for the sort merge join

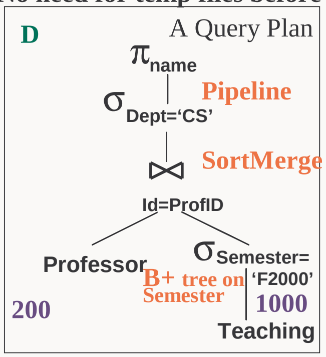
We have **48** buffer blocks in main memory

•There are 10000 teachings in 4 semester. Assuming a uniform distributions, there would be 2500 teachings in the Fall of 2000. (250 pages since 10 tuples per page) 250 I/O + 2 I/Os for B+tree.
• While in main memory after selection, the first runs can be produced by sorting the buffers in MM. The first runs are obtained after 252 I/O for selection and 250 I/Os for writing the 1st runs. We obtain ⌈250 pages /48 blocks⌉ = 6 runs
• We have enough buffers to merge them in one pass. This adds 250 + 250 I/Os to sort them all.
• Selecting and sorting the teachings costs 252 + 250 + 250 + 250 = 1002 I/Os.
•The sort merge requires an additional scan of both sorted files: 200 + 250

Cost Estimation for D (cont')

If DBMS is smart:
No need for temp files before Join

We have **48** buffer blocks in main memory



Cost =400+502+450=**1352 I/O**
400 for 1st runs of P
502 for selecting & 1st runs of T
450 for the sort merge join

- Sorting *Professor* costs us 2 * 200 I/O to generate the 1st runs. We need only one pass to merge the 5 runs but we can hold on and do this merge while doing the merge of the sort-merge join.
- As before, selecting the teachings costs 2 I/Os for the B+-tree and 250 I/Os to read the clustered 250 pages of teachings in the Fall 2000. That is 252 I/Os.
- While reading the 250 pages, they are sorted in MM and written to disk as the first runs.
- Since we have enough buffers, we can use 5 input buffers for the runs of Professor, 6 buffers for the input of the runs of the selected teaching and 1 buffer for output.
- The sort merge requires an additional scan of these runs: 200 + 250

Query Plans