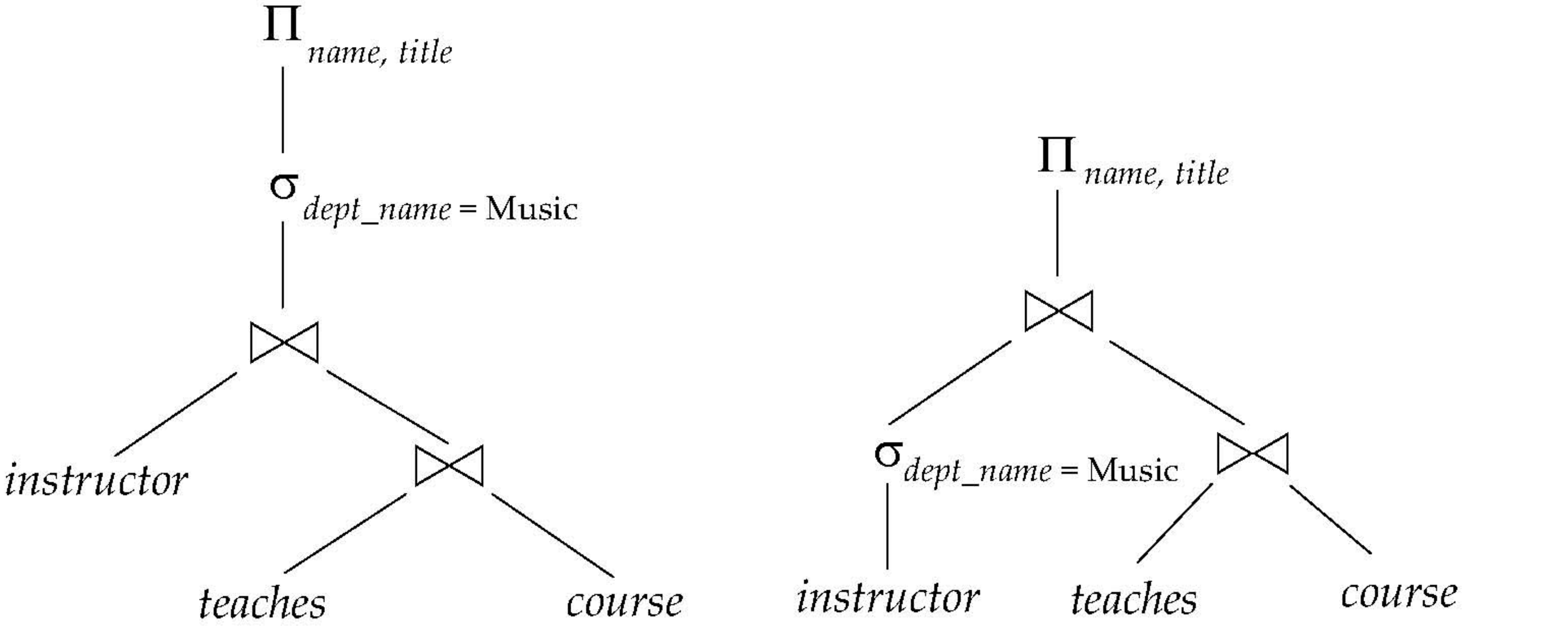
Query Evaluation and Optimization

Main Steps

1. Translate into RA: select/project/join
2. Greedy optimization of RA: by pushing selection and projection into RA expression
3. Cost-based estimation and selection of **best join order** (for N relations N! possible orders)
4. Select best implementation for each operator

Greedy optimization of RA: pushing selection and projection into RA expression

instructor(name, dep_name, salary), teaches(name, cid), course(cid, title)



Selection and Indexing

- If we have a S.C condition supported by an existing index we use the index
- If we have a conjunction, such as S.A>5 and S.B <10 with indexes on both, then we can select the better of the two (optimization)
- If there is no index, do we create one?

3. Select best implementation for each operator

- **Selection**: use the appropriate index if one is available. Otherwise visit all the blocks
- **Projection**: trivial, unless we need to eliminate duplicates—duplicates eliminated by sorting or hashing.
- **Joins**
  - Nested loop, reasonable only with indexes
  - Nested Block Join
  - Sort-Merge Join
  - Hash Join

Query Evaluation and Optimization

Main Steps

1. Translate into RA: select/project/join
2. Greedy optimization of RA: by pushing selection and projection into RA expression
3. Select best implementation for each operator
4. **Cost-based estimation and selection of best join order**.

Join Ordering Example $\bowtie$

- For all relations $r_1, r_2,$ and r_3,

  $$(r_1 \bowtie r_2) \quad r_3 = r_1 \bowtie (r_2 \bowtie r_3)$$

  (Join Associativity: any order will do: which one is best?)

- If $r_2 \bowtie r_3$ is quite large (Say N blocks) and $r_1 \bowtie r_2$ is small (M blocks) use

  $$(r_1 \bowtie r_2) \bowtie r_3$$

  so that we compute and store a smaller temporary relation.

Cost of block joins B1+ B2 + M + B3 versus B3 + B2 + N + B1 (when there is plenty of memory and both M and N fit. If not , the larger N will cause worse slowdown than M)

Estimating the size of joins:

Special case*: *foreign keys: R(A, B) $\bowtie$ S(<u>B</u>, C) *where* B *in* R is *declared as the foreign key referencing* S. *Then the size of the join is exactly the size of* R.

Estimation of the Size of Joins (Cont.)

- If $R \cap S = \{A\}$ is not a key for R or S.
  If we assume that every tuple t in R produces tuples in $R \bowtie S$, the number of tuples in $R \bowtie S$ is estimated to be:

  $$\frac{n_r * n_s}{V(A,s)}$$

  If the reverse is true, the estimate obtained will be:

  $$\frac{n_r * n_s}{V(A,r)}$$

  The lower of these two estimates is probably the more accurate one.
- Can improve on above if histograms are available
  - Use formula similar to above, for each cell of histograms on the two relations
  - V(A,s), V(A,r): the average number of A-values in s and r (resp)

Just a rough estimate

Example:

Students(Name, Major)$\bowtie$ Courses(Title, Major)

1000 students, 100 Courses, 10 majors

What is the size of the result?

lstudent x course l= 100,000

Number of Courses tuples fpr a given Major: 100/10=10
join size estimate: 1000 x 10 = 10000

Number of Student tuples fpr a given Major: 1000/10=100
join size estimate: 100x100 = 10000

Same size: in reality they tend to be quite different.

Statistics collection commands

Statistics collection commands

- **DBMS has to collect statistics on tables/indexes**
  - For optimal performance
  - Without stats, DBMS does stupid things…
- **DB2**
  - RUNSTATS ON TABLE <userid>.<table> AND INDEXES ALL
- **Oracle**
  - ANALYZE TABLE <table> COMPUTE STATISTICS
  - ANALYZE TABLE <table> ESTIMATE STATISTICS (cheaper than COMPUTE)
- Run the command after major update/index construction

Simple Questions

Multiple choice questions. A. Relations R and S have attributes a and b. Then, which of the properties below is true for the following RA queries:

$$\text{Q1: } \pi_a(R) \cap \pi_a(S) \qquad\qquad \text{Q2: } \pi_a(R \cap S)$$

a) Q1 and Q2 produce the same answer

b) is always contained in the answer to Q2.

c) The answer to Q2 is always contained in the answer to Q1.

d) Q1 and Q2 produce different answers.

Next Question

When null values are allowed in column R.a or column R.b, the value of the following logic expression in SQL

R.a > R.b AND R.a < 0 AND R.b > 0 can be, depending on the tuple considered:

(a) only TRUE or FALSE
(b) only FALSE or UNKNOWN
(c) only TRUE or UNKNOWN
(d) Any of TRUE, FALSE, or UNKNOWN

Last Question

Explain the way in which overflow buckets are used in

(a) Closed Hashing, versus the way in which they are used in

(b) Extensible Hashing.