Database Management Systems

Madhavan Mukund

https://www.cmi.ac.in/~madhavan

Sai University

Lecture 18, 1 November 2023

Query processing

- Translate the query from SQL into relational algebra
- Evaluate the relational algebra expression
- Challenges
  - Many equivalent relational algebra expressions

    $\sigma_{salary<75000}(\pi_{salary}(instructor))$ vs $\pi_{salary}(\sigma_{salary<75000}(instructor))$
  - Many ways to evaluate a given expression
- Query plan
  - Annotate the expression with a detailed evaluation strategy key values
    - Use index on *salary* to find instructors with $salary < 75000$
    - Or, scan entire relation, discard rows with $salary \geq 75000$

Query processing — Selection

(A1) Linear search

(A2) Clustering index, equality on key — index height h_i

(A3) Clustering index, equality on nonkey

(A4) Secondary index (key, non-key)

(A5) Clustering index, comparison — sorted on A

(A6) Clustering index, comparison — not sorted on A

(A7) Conjunctive selection using one index

(A8) Conjunctive selection using composite index

(A9) Conjunctive selection using intersection of pointers

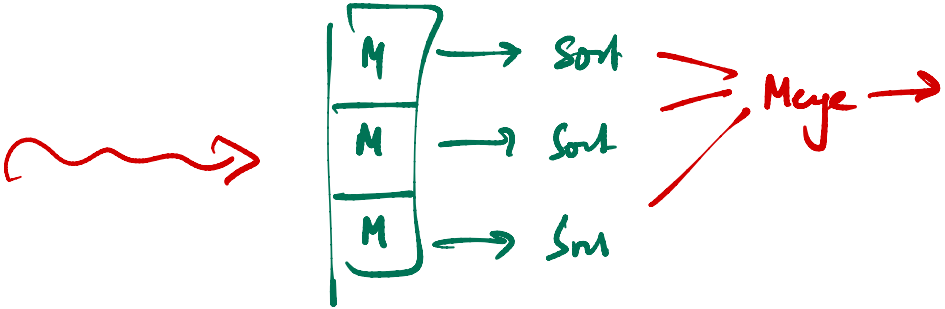
(A10) Disjunctive selection by union of pointers

(Neg) Negation

$\sigma_\theta(r)$

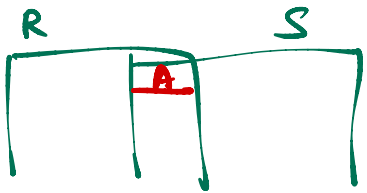
External merge sort

- N records, b_r blocks, M blocks in memory
- Compute sorted runs of size M
- Merge sorted runs, 1 block per run vs b_b blocks per run



Computing joins

- Nested-loop join
- Block nested-loop join
- Indexed nested-loop join
- Merge join — after sorted
- Hash join



Materialization vs pipeling

Cost of operations – reading & writing blocks

- Ignored cost of final write

Hash join

Read R → compute blocks B_R

Read S → compute blocks B_S

$2 \times B_R$

$2 \times B_S$

Read B_R & B_S & compute join

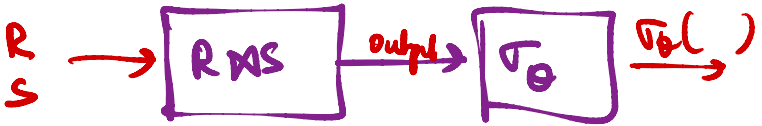
$1 \times B_R + 1 \times B_S$

Materialization vs pipeling

Store output of a stage to disk

$\sigma_\theta (R \bowtie S)$

Is output of inner operation stored?



Materialization vs pipeling

Pipelining

Next operation should be able to work incrementally

- Linear selection ✓
- Join ?

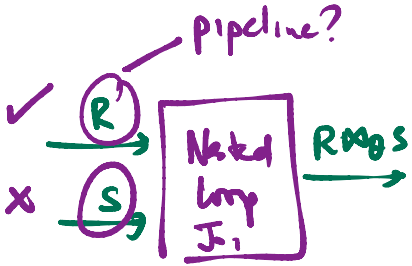
Materialization vs pipeling

Nested loop join

for each r in R

for each s in S

check if r matches s



Materialization vs pipeling

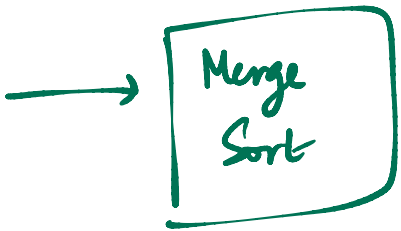
Indexed nested loop

for each r in R — pipeline ✓

probe index for common attribute in S — no pipeline

Materialization vs pipeling

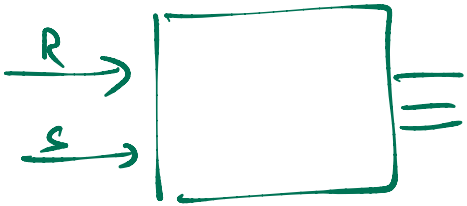
Sorting



1. Create Sorted runs
   - Can be done in batches
   - Pipelined input ✓
2. Merge Sorted runs
   - Writing to disk
   - Accounted

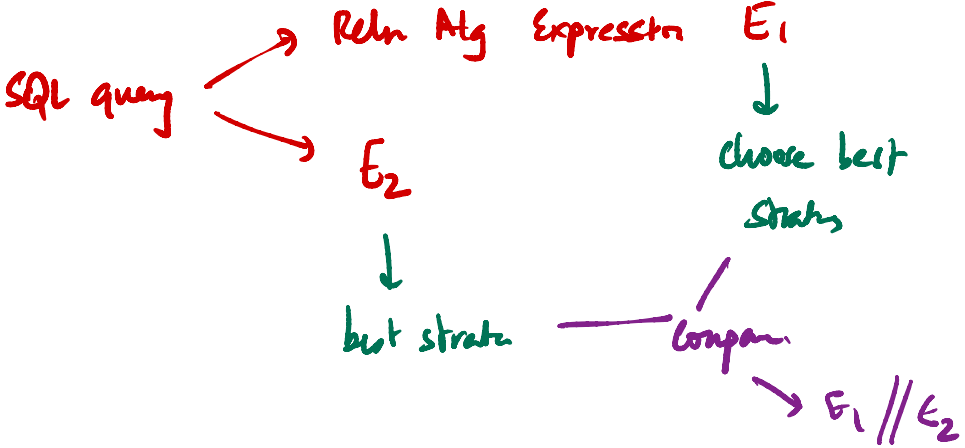
Materialization vs pipeling

Hash join – "blocks" the pipeline



Query optimization

- Choose plan with lowest cost



Query optimization

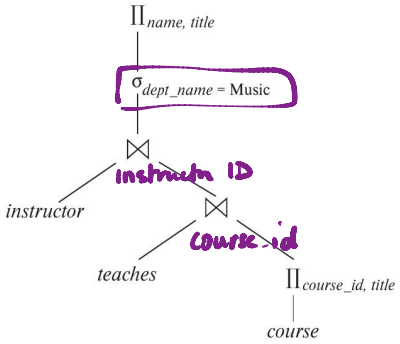
- Choose plan with lowest cost
- *Find names and course titles of courses taught by instructors from Music Dept*

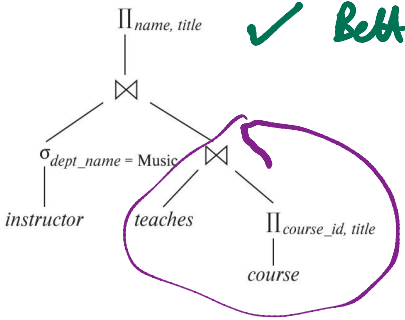
Query optimization

- Choose plan with lowest cost

SQL → RAlg → Manipulate RA exp

- *Find names and course titles of courses taught by instructors from Music Dept*





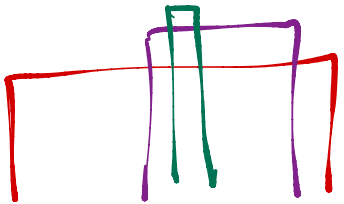
Transforming expressions

Selection

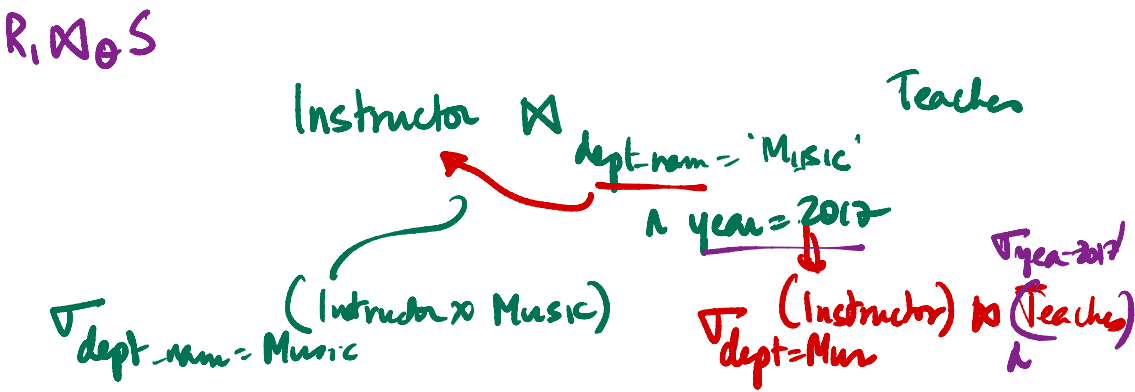
$$\sigma_{\theta_1 \wedge \theta_2}(r) \equiv \sigma_{\theta_1}(\sigma_{\theta_2}(r)) \approx \sigma_{\theta_2 \wedge \theta_1}(r)$$

Projections

Collapse a segm



Transforming expressions



Transforming expressions

Join

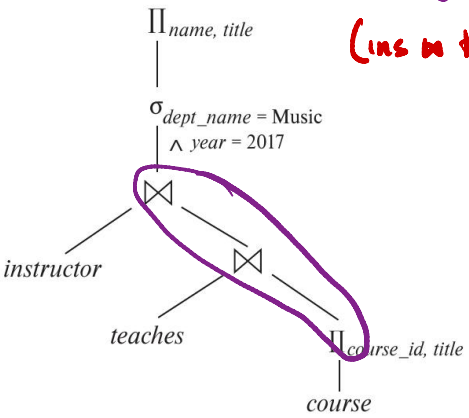
$$R_1 \bowtie R_2 = R_2 \bowtie R_1$$

"Which is the Outer relation?"

$$(R_1 \bowtie R_2) \bowtie R_3 = R_1 \bowtie (R_2 \bowtie R_3)$$

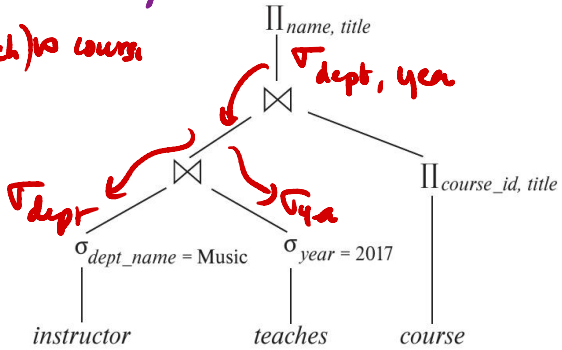
Associativity

Transforming expressions



ins ⋈ (teach ⋈ course)

(ins ⋈ teach) ⋈ course



Maintaining a database catalogue

- n_r — number of tuples in r
- b_r — number of blocks used by r
- ℓ_r — size of a tuple in r
- f_r — blocking factor of r, how many tuples fit in a block
- $V(A, r)$ — number of distinct values of attribute A in r
  - Store distribution of values as histogram

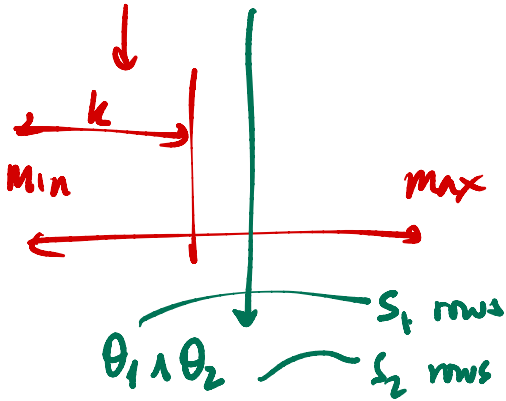
- if numeric, range

Maintaining catalogue?

Periodically recompute (when load is low)

Estimating output of an operation

- Selection
  - Simple, range, conjunction, disjunction



$\sigma_{A=v}(r)$ Histogram

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

$\frac{k}{\max - \min} \times n_r$

$\frac{s_1}{n_r} \times \frac{s_2}{n_r}$

Estimating output of an operation

- Selection
  - Simple, range, conjunction, disjunction

$a \vee b = \neg(\neg a \wedge \neg b)$

Disjunction

$\sigma_{\theta_1 \vee \theta_2}(r)$

$\downarrow s_1 / n_r$ $\downarrow s_2 / n_r$

— not (not θ_1 $\wedge$ not θ_2)

$1 - \left(1 - \frac{s_1}{n_r} \times 1 - \frac{s_2}{n_r}\right)$

Estimating output of an operation

- Selection
  - Simple, range, conjunction, disjunction
- Join
  - Keys and non-keys

$r \times s$ $\quad n_r \times n_s$

Join on A

Each row in r matches $V(A,s)$ rows in s

If A is a key in s, $V(A,s)=1$ $\quad n_r \cdot V(A,s)$

$n_s \; V(A,r)$

Estimating output of an operation

- Selection
  - Simple, range, conjunction, disjunction
- Join
  - Keys and non-keys
- Projection

All rows, but fewer column – use schem

Estimating output of an operation

- Selection
  - Simple, range, conjunction, disjunction
- Join
  - Keys and non-keys
- Projection
- Aggregation – # of values for aggregate attribs

Estimating output of an operation

- Selection
  - Simple, range, conjunction, disjunction
- Join
  - Keys and non-keys
- Projection
- Aggregation
- Set operations

Estimating output of an operation

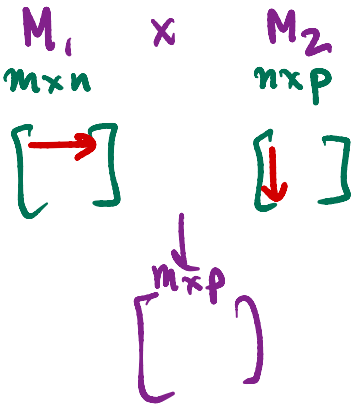
- Selection
  - Simple, range, conjunction, disjunction
- Join
  - Keys and non-keys
- Projection
- Aggregation
- Set operations
- Outer joins

– Extra cost from non matching rows

– one per row

Join ordering

Matrix multiplication

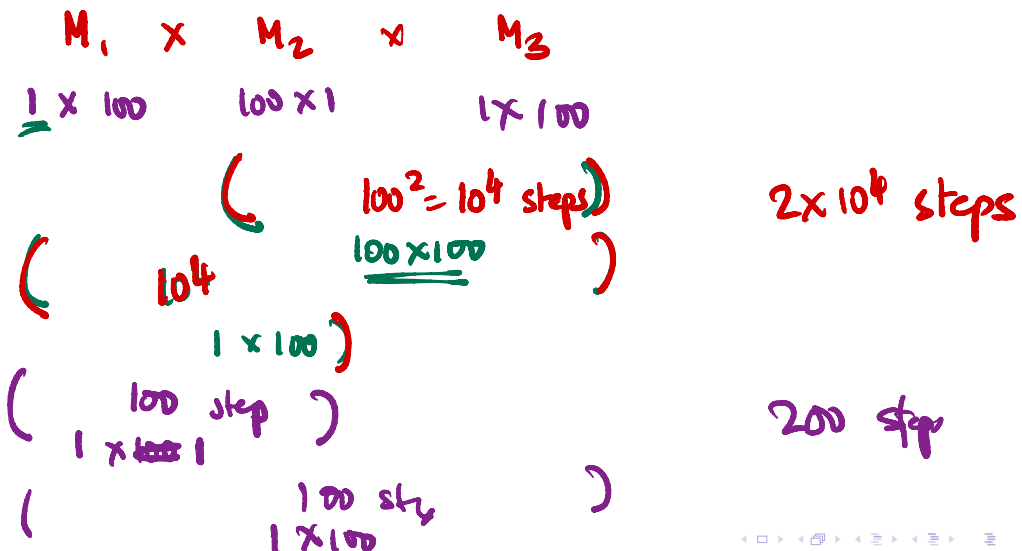


Cost is $m \cdot n \cdot p$

Output is $m \times p$

Each output entry requires n ops

Join ordering



Join ordering

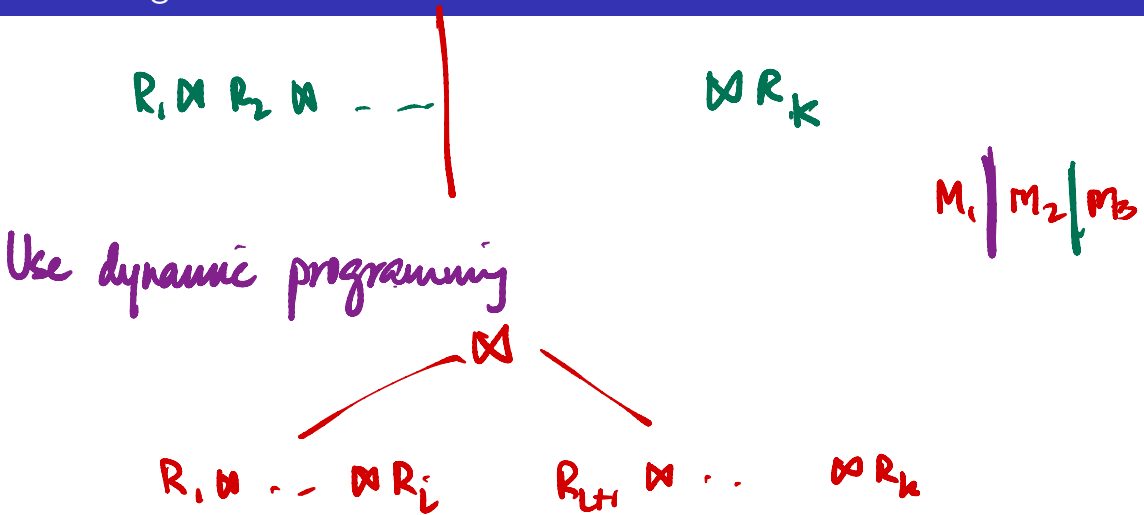
Similarly join

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

c_{12} c_{23}

$c_{(12)3}$ $c_{1(23)}$

Join ordering



Heuristics

- Perform selection early

Heuristics

- Perform selection early
- Perform projection early

Heuristics

- Perform selection early
- Perform projection early
- Perform most restrictive selection/join first