Database Management Systems

Madhavan Mukund

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

Lecture 9, 17 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 optimization

- Choose plan with lowest cost
- Maintain database catalogue — number of tuples in each relationn, size of tuples, ...
- Assess cost in terms of disk access and transfer, CPU time, ...
- For simplicity, ignore in-memory costs (CPU time), restrict to disk access
- Disk accesses
  - Relation r occupies b_r blocks
  - Disk seeks — time t_S per seek
  - Block transfers — time t_T per transfer
- Other factors — buffer management etc

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

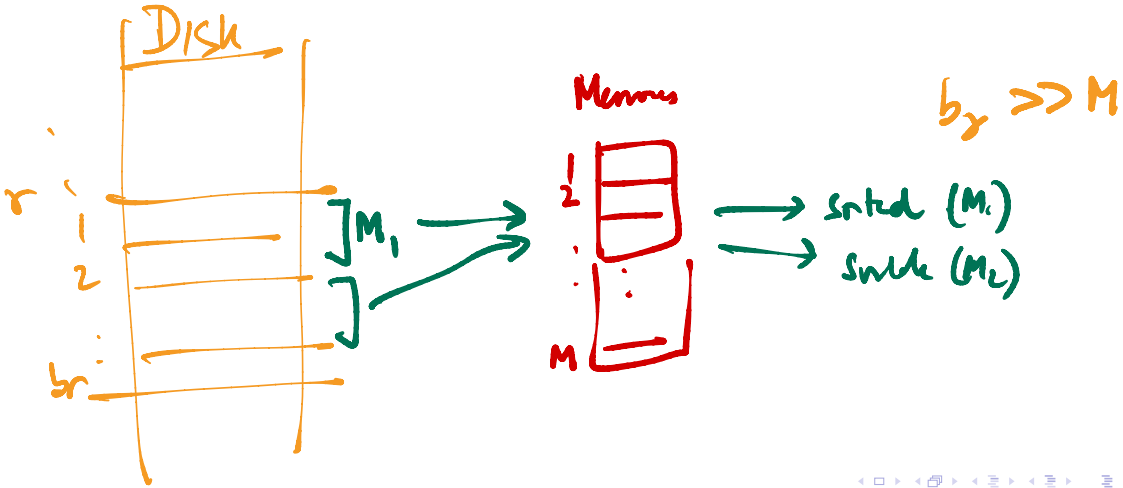
(Neg) Negation

Sorting

- In-memory sorting vs sorting on disk
- Merging sorted lists — varieties
- Traditional merge sort

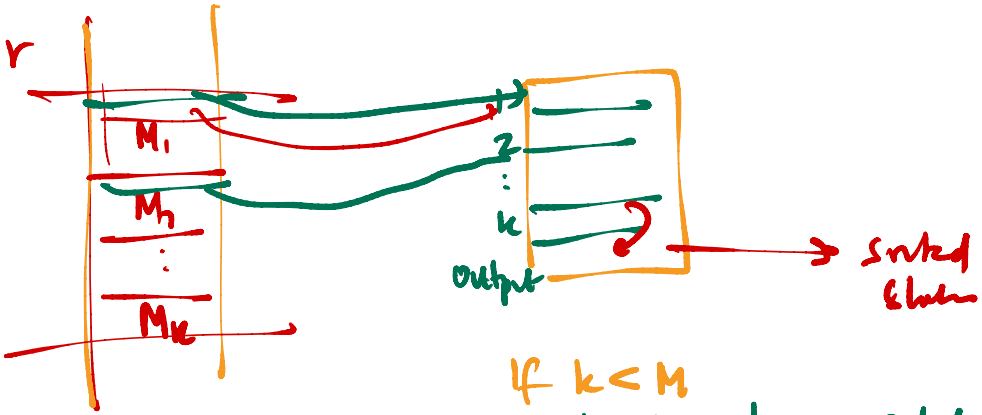
External merge sort

- N records, b_r blocks, M blocks in memory



External merge sort

- N records, b_r blocks, M blocks in memory

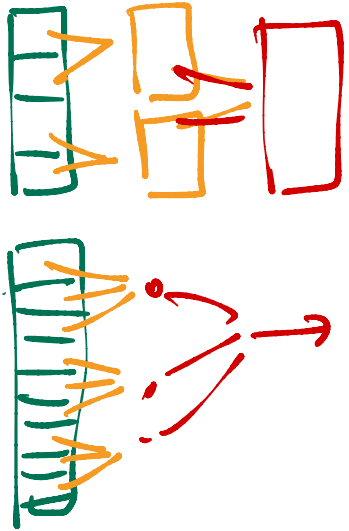


External merge sort

- 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
- Complexity
  - b_r/M sorted runs, $\lceil \log_{\lfloor M/b_b \rfloor - 1}(b_r/M) \rceil$ merge passes

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
- Complexity
  - b_r/M sorted runs, $\lceil \log_{\lfloor M/b_b \rfloor - 1}(b_r/M) \rceil$ merge passes
  - Block transfers — $b_r\ (2\lceil \log_{\lfloor M/b_b \rfloor - 1}(br/M) \rceil + 1)$
    - Why not $b_r\ (2\lceil \log_{\lfloor M/b_b \rfloor - 1}(br/M) \rceil + 2)$?



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
- Complexity
  - b_r/M sorted runs, $\lceil \log_{\lfloor M/b_b \rfloor - 1}(b_r/M) \rceil$ merge passes
  - Block transfers — $b_r \ (2\lceil \log_{\lfloor M/b_b \rfloor - 1}(br/M) \rceil + 1)$
    - Why not $b_r \ (2\lceil \log_{\lfloor M/b_b \rfloor - 1}(br/M) \rceil + 2)$?
  - Block seeks — $2\lceil b_r/M \rceil + \lceil b_r/b_b) \rceil \ (2(\lceil \log_{\lfloor M/b_b \rfloor - 1}(br/M) \rceil - 1)$

Computing joins

- Running example
  - *Student* $\bowtie$ *Takes*
  - *Student* — 5000 rows, 100 blocks
  - *Takes* — 10000 rows, 400 blocks

50 rows/block
25 rows/block

Nested-loop join

- (5000 rows, 100 blocks) *Student* ⋈ *Takes* (10000 rows, 400 blocks)

r s

n_r b_r

n_s b_s

for each row r in Student

for each row s in Takes

r as = star

$\times n_r$

$= b_r$ seeks $\times$ 1 seek + b_r transfers for r

$+ n_r$ (1 seek + b_s transfers for s)

$\downarrow$ 1 seek + b_s

$b_r + b_r + n_r(1 + b_s)$

$b_s + b_s + n_s(1 + b_r)$

$= 401 + 10000(101)$

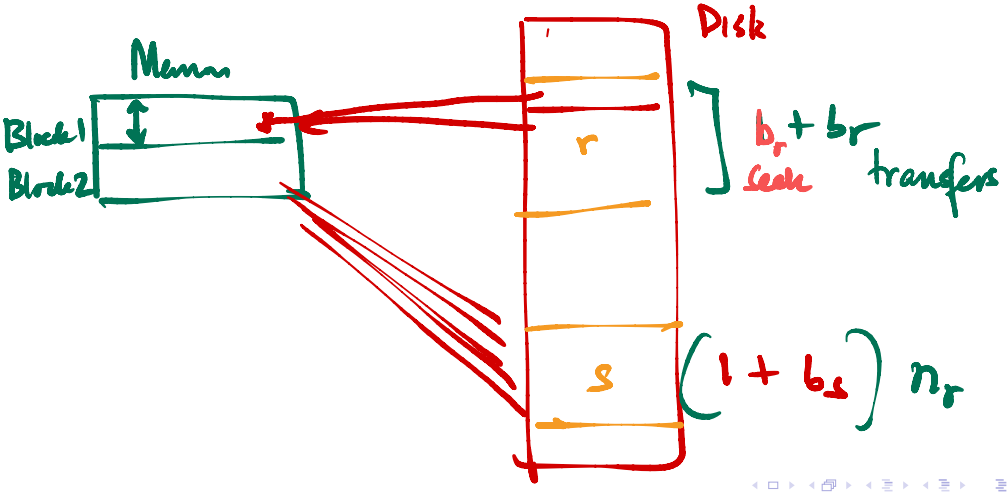
1×10^6

$101 + 5000(401)$

2×10^6

Nested-loop join

- (5000 rows, 100 blocks) *Student* ⋈ *Takes* (10000 rows, 400 blocks)



Nested-loop join

- (5000 rows, 100 blocks) *Student* $\bowtie$ *Takes* (10000 rows, 400 blocks)
- Complexity
  - $r \bowtie_\theta s$ — r is outer relation, s is inner relation
  - Block transfers: $b_r + n_r \cdot b_s$
  - Block seeks: $b_r + n_r$ — inner relation read sequentially
  - Special case: smaller relation fits in memory

As asked in class, you do re-seek r for each fresh block

Block nested-loop join

- (5000 rows, 100 blocks) *Student* $\bowtie$ *Takes* (10000 rows, 400 blocks)

for each block r in Student

for each block s in Takes

Compare all rows in r vs all rows in s

$b_r(1+b_s)$

vs

$n_r(1+b_s)$

$(1+b_r)+b_r(1+b_s)$

$(1+b_s)+b_s(1+b_r)$

Block nested-loop join

- (5000 rows, 100 blocks) *Student* $\bowtie$ *Takes* (10000 rows, 400 blocks)
- Complexity
  - $r \bowtie_\theta s$ — r is outer relation, s is inner relation
  - Block transfers: $b_r + b_r \cdot b_s$
  - Block seeks: $b_r + b_r = 2b_r$

Indexed nested-loop join

- (5000 rows, 100 blocks) *Student* $\bowtie$ *Takes* (10000 rows, 400 blocks)

for each r in Student

Use index to fetch matching rows in Takes

Indexed nested-loop join

- (5000 rows, 100 blocks) *Student* $\bowtie$ *Takes* (10000 rows, 400 blocks)
- Complexity
  - $r \bowtie_\theta s$ — r is outer relation, s is inner relation
  - Total cost: $b_r(t_T + t_S) + n_r \cdot c$
    - c is cost of single selection on s

Merge join

- (5000 rows, 100 blocks) *Student* $\bowtie$ *Takes* (10000 rows, 400 blocks)

Sort r
Sort s
Merge

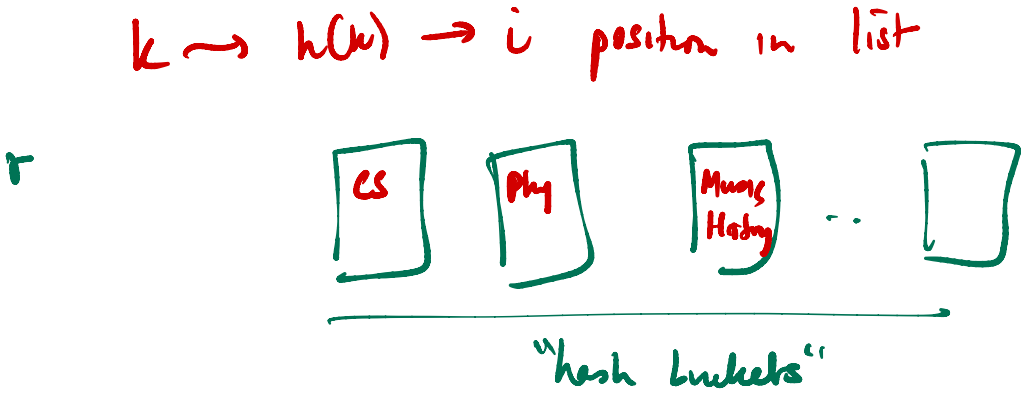
Cost of sorting r & s
+ $O(b_r + b_s)$

Merge join

- (5000 rows, 100 blocks) *Student* $\bowtie$ *Takes* (10000 rows, 400 blocks)
- Complexity
  - $r \bowtie_\theta s$ — r is outer relation, s is inner relation
  - Block transfers: $b_r + b_s$
  - Block seeks: $\lceil b_r/b_b \rceil + \lceil b_s/b_b \rceil$

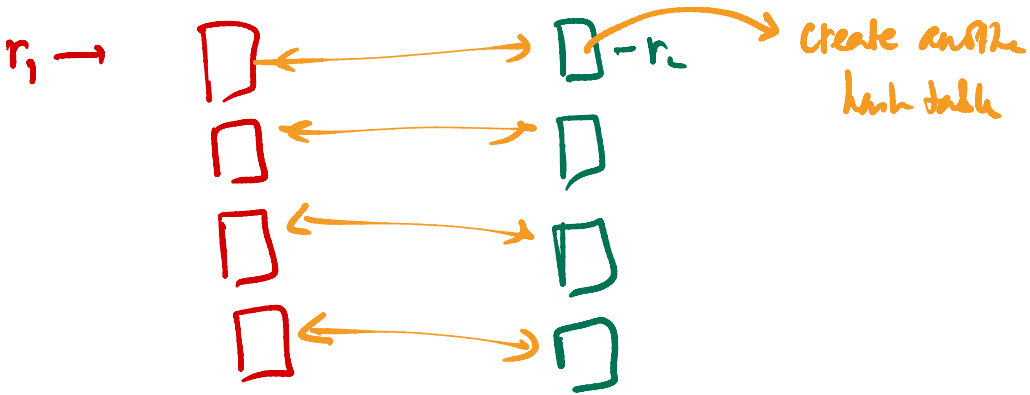
Hash join

- (5000 rows, 100 blocks) *Student* $\bowtie$ *Takes* (10000 rows, 400 blocks)



Hash join

- (5000 rows, 100 blocks) *Student* $\bowtie$ *Takes* (10000 rows, 400 blocks)

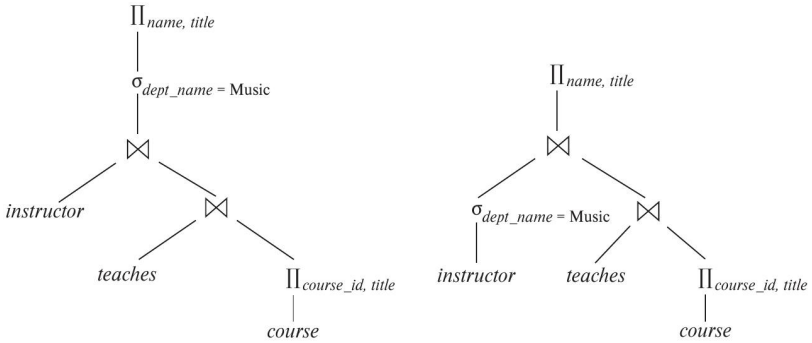


Other operations

- Duplicate removal
- Aggregrate queries with grouping
  - Aggregate while sorting/hashing
- Set theoretic operations

Query optimization

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



Transforming expressions

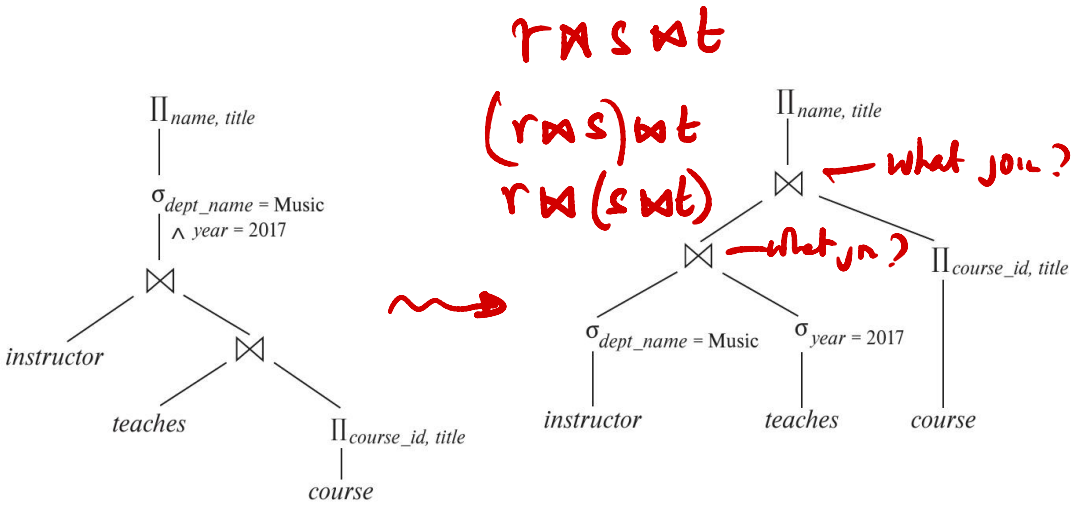
$r \bowtie s = s \bowtie r$

$\sigma_\theta(r \bowtie s) \Longrightarrow \sigma_\theta(r) \bowtie s$ if θ only refers to r

⋮

Transformation

Transforming expressions



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

Estimating output of an operation

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

Join ordering

$(M_1 \times M_2 \times \cdots \times M_n)$

Similar to matrix mult. ordering

$r_1 \bowtie r_2 \bowtie \cdots \bowtie r_n$

Heuristics

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