RDBMS and SQL

Madhavan Mukund

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

Lecture 11, 13 November 2025

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

Selection

- Linear search
- Clustering index — index height h_i
  - Equality on key
  - Equality on nonkey
- Secondary index (key, non-key)
- Clustering index, comparison on A
  - Sorted on A
  - Not sorted on A
- Boolean combinations
  - Conjunctive selection using one index
  - Conjunctive selection using composite index
  - Disjunction, negation, ...

External merge sort

- N records, b_r blocks, M blocks in memory
- Compute sorted runs of size M
- Merge sorted runs
- Complexity
  - b_r/M sorted runs, $\lceil \log_{M-1}(b_r/M) \rceil$ merge passes
  - Block transfers — $b_r\ (2\lceil \log_{M-1}(b_r/M) \rceil + 1)$
  - Block seeks — $2\lceil b_r/M \rceil + b_r\ (2(\lceil \log_{M-1}(b_r/M) \rceil - 1)$

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

Student $b_r = 100$, $n_r = 5000$
Takes $b_s = 400$

$n_r \cdot b_s = 2 \times 10^6$
$+ b_r = 2{,}000{,}100$

Take $b_r = 400$, $n_r = 10000$
Student $b_s = 100$

$n_r \cdot b_s = 1 \times 10^6$
$+ 400 = 1{,}000{,}400$

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: $b_r + b_r \cdot b_s$ vs $n_r \cdot b_s$
  - Block seeks: $b_r + b_r = 2b_r$

Student-Takes $100 + 100 \times 400$

200

Takes-Student $400 + 100 \times 4 n$

> 800

What is the output size ?

Indexed nested-loop join

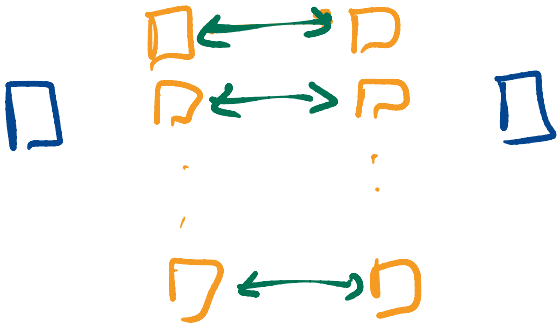
- (5000 rows, 100 blocks) *Student* $\bowtie$ *Takes* (10000 rows, 400 blocks)
- Complexity
  - $r \bowtie_\theta s$ — r is outer relation, s is inner relation
  - Total cost: $b_r(t_T + t_S) + n_r \cdot c$
    - c is cost of single selection on s

Merge join

- (5000 rows, 100 blocks) *Student* $\bowtie$ *Takes* (10000 rows, 400 blocks)
- Complexity
  - $r \bowtie_\theta s$ — r is outer relation, s is inner relation
    - Assume relations are sorted (add cost of sorting)
  - Block transfers: $b_r + b_s$
  - Block seeks: $\lceil b_r/b_b \rceil + \lceil b_s/b_b \rceil$
    - Read a chunk of blocks b_b at a time

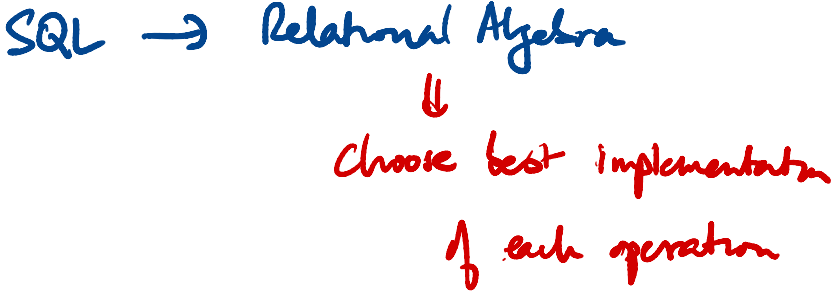
Hash join

- (5000 rows, 100 blocks) *Student* $\bowtie$ *Takes* (10000 rows, 400 blocks)
- Hash function on join attribute A, n_h output values
- Join each pair of hash buckets — build index and probe



Query optimization

- Choose plan with lowest cost

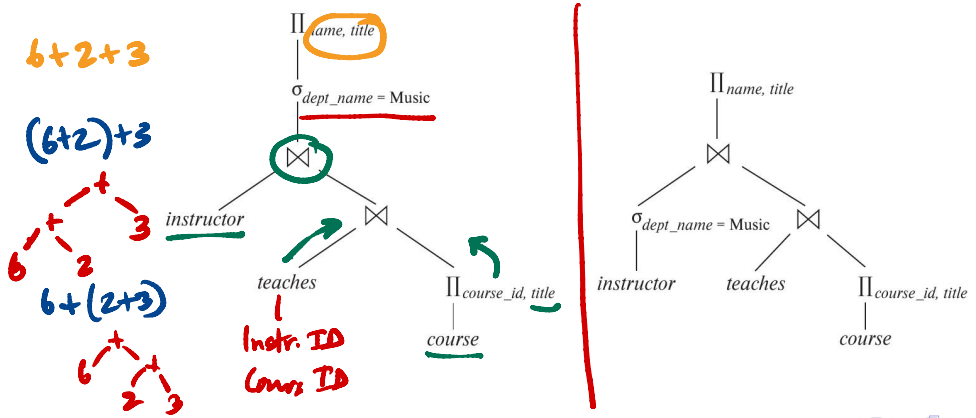


Query optimization

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

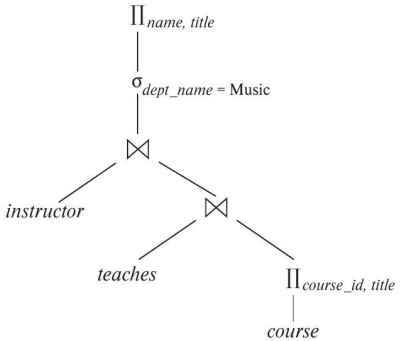
Query optimization

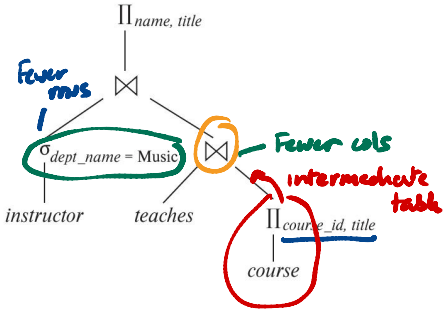
- Choose plan with lowest cost
- *Find names and course titles of courses taught by instructors from Music Dept*



Query optimization

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

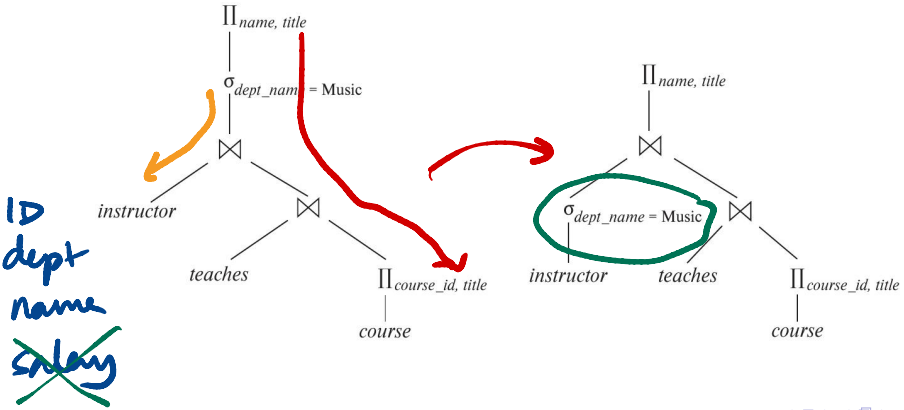




$\Pi_{n.}\left(\sigma_{dy}\left(\text{inst.} \bowtie (\text{teaches} \bowtie \text{course})\right)\right)$

Query optimization

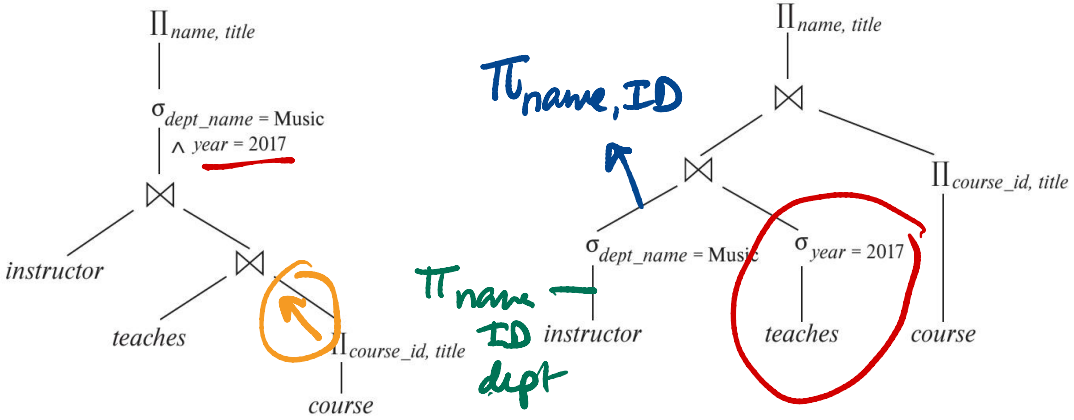
- Choose plan with lowest cost
- *Find names and course titles of courses taught by instructors from Music Dept*



Transforming expressions

Systematic set of rules to transform expressions, preserve & output

Transforming expressions



Maintaining a database catalogue

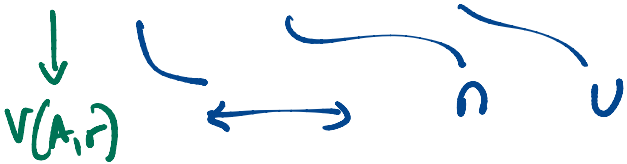
- n_r — number of tuples in r
- b_r — number of blocks used by r
- ℓ_r — size of a tuple in r
- f_r — blocking factor of r, how many tuples fit in a block
- $V(A, r)$ — number of distinct values of attribute A in r
  - Store distribution of values as histogram

Background update

Pandas → data summary

Estimating output of an operation

- Selection
  - Simple, range, conjunction, disjunction



Estimating output of an operation

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

Estimating output of an operation

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

Estimating output of an operation

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

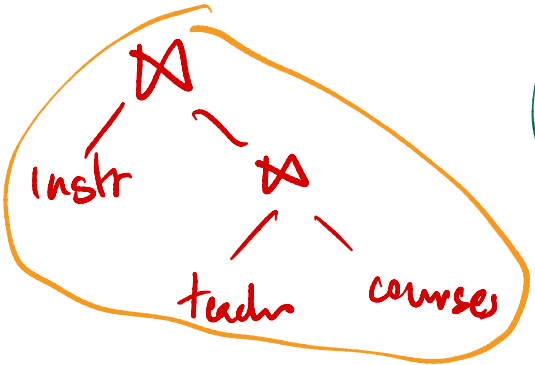
Estimating output of an operation

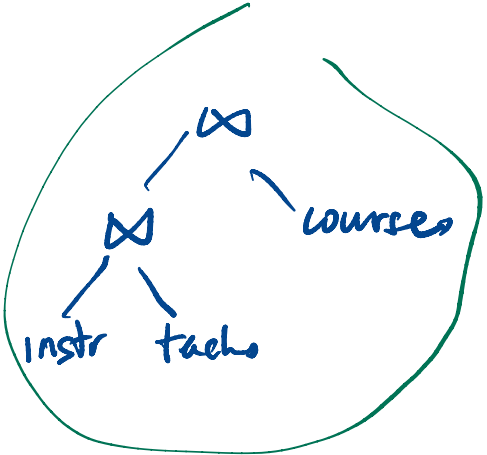
- Selection
  - Simple, range, conjunction, disjunction
- Join
  - Keys and non-keys
- Projection
- Aggregation
- Set operations

Estimating output of an operation

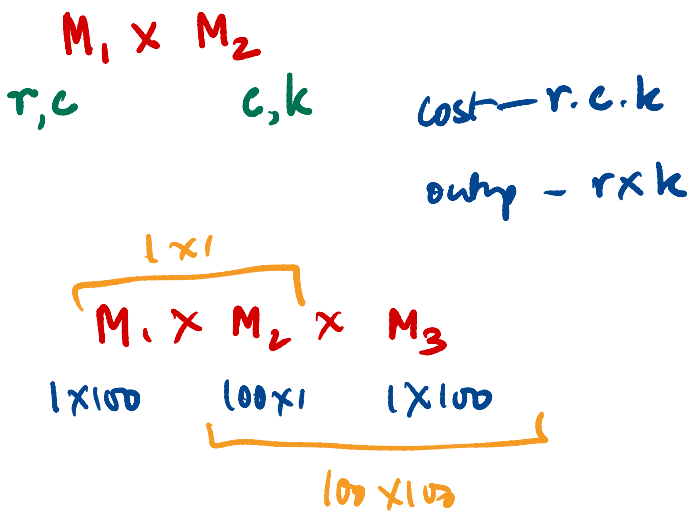
- Selection
  - Simple, range, conjunction, disjunction
- Join
  - Keys and non-keys
- Projection
- Aggregation
- Set operations
- Outer joins

Join ordering





Join ordering



Heuristics

- Perform selection early

$\sigma_{Dept = Music}$

$\sigma_{year = 2017}$

Heuristics

- Perform selection early
- Perform projection early

$\pi_{\text{courseid, title}}$

Heuristics

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

Query planny

Transactions

Multiple low level operations ⇒ One high level op

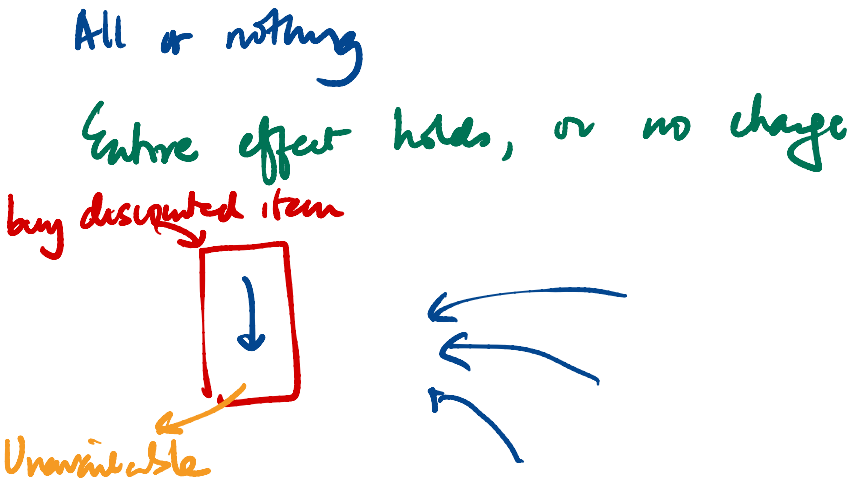
Buy ticket

1. Check avail.
2. Choose flight
3. Pay

What guarantees should transactions provide

Desirable properties

- Atomicity



Desirable properties

- Atomicity
- Consistency

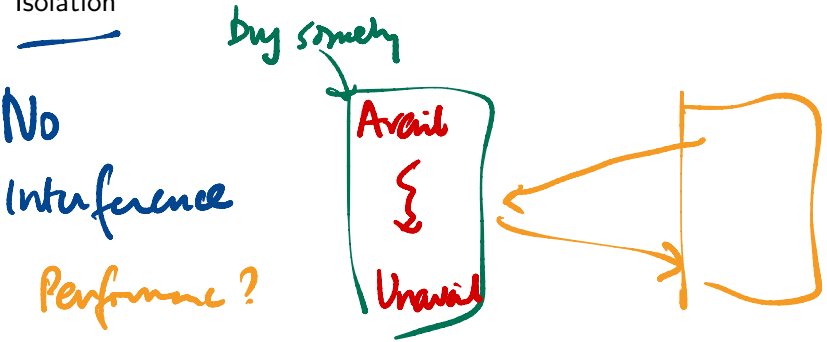
dept_name in instructor is a foreign key in dept

Bank transfer with a bad insert a new dept, add instructor

Desirable properties

- Atomicity
- Consistency
- Isolation

Atomicity in the presence of concurrency



Desirable properties

- Atomicity
- Consistency
- Isolation
- Durability

Desirable properties

- Atomicity
- Consistency
- Isolation
- Durability
- ACID properties

Transaction logs

- Log each update before it happens
- Rollback updates in case of failure