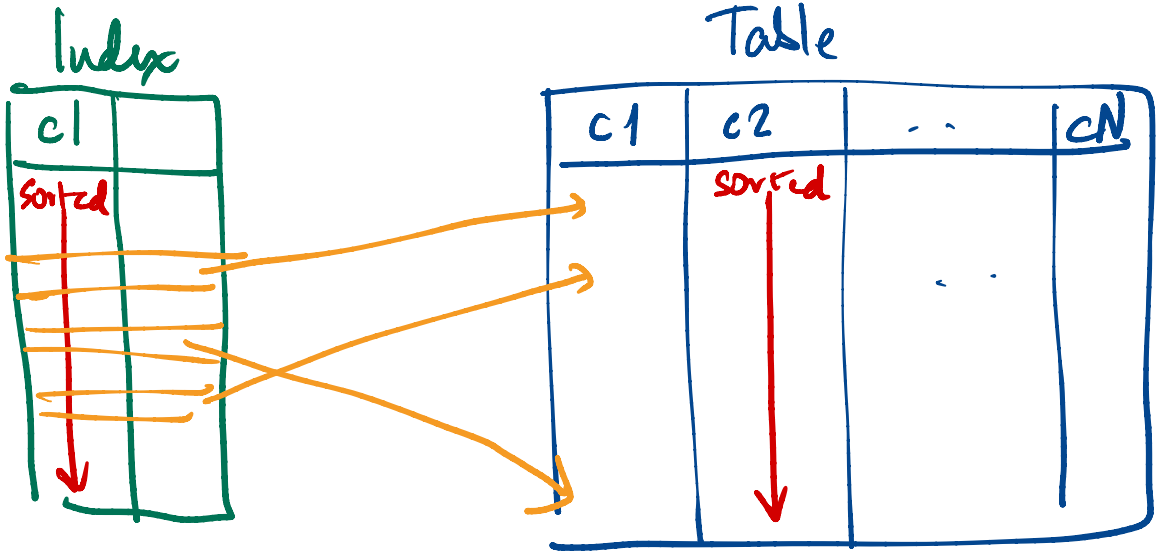
Building an index

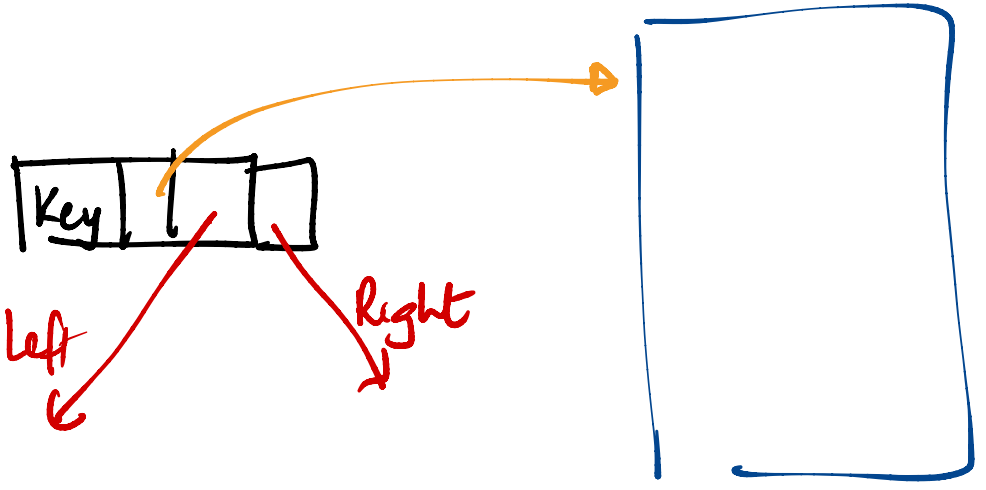


Sequential index

Updates are expensive

Instead

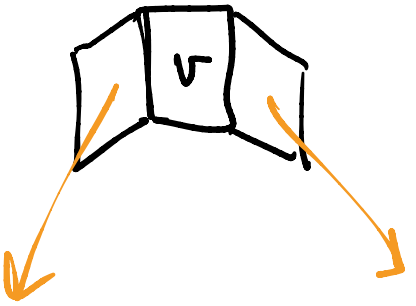
Maintain a search tree



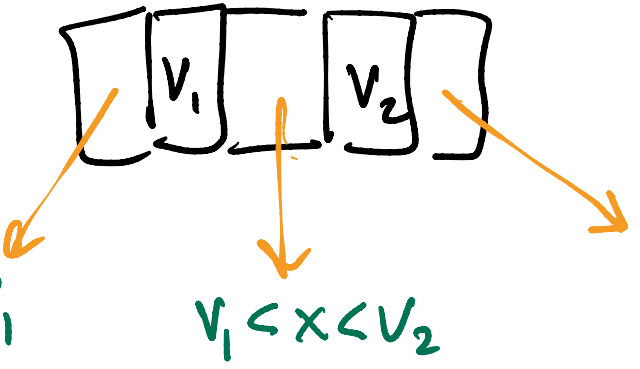
Challenge: Index tree is on disk — each node access reads a block from disk

Have a more elaborate type of node

Binary tree



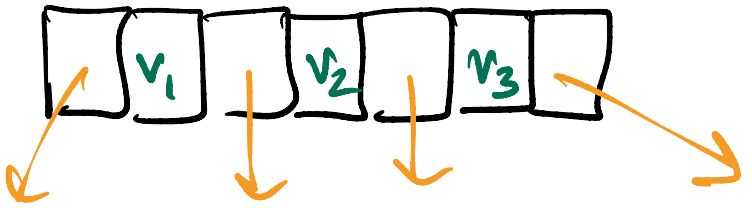
Ternary tree



$v_1 < v_2$

height is $\log_3(\text{size})$

4 way tree



k-way tree

Choose k so that one node fill up a block

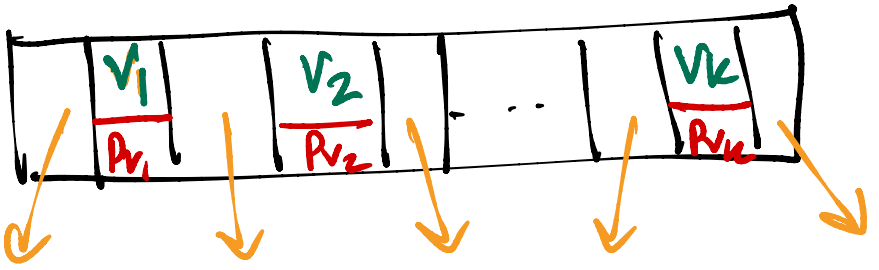
$$\text{height} = \log_k(\text{size})$$

Each value v should be paired with a pointer to the appropriate row in the original table

(v, P_v)

Option 1

B-Tree



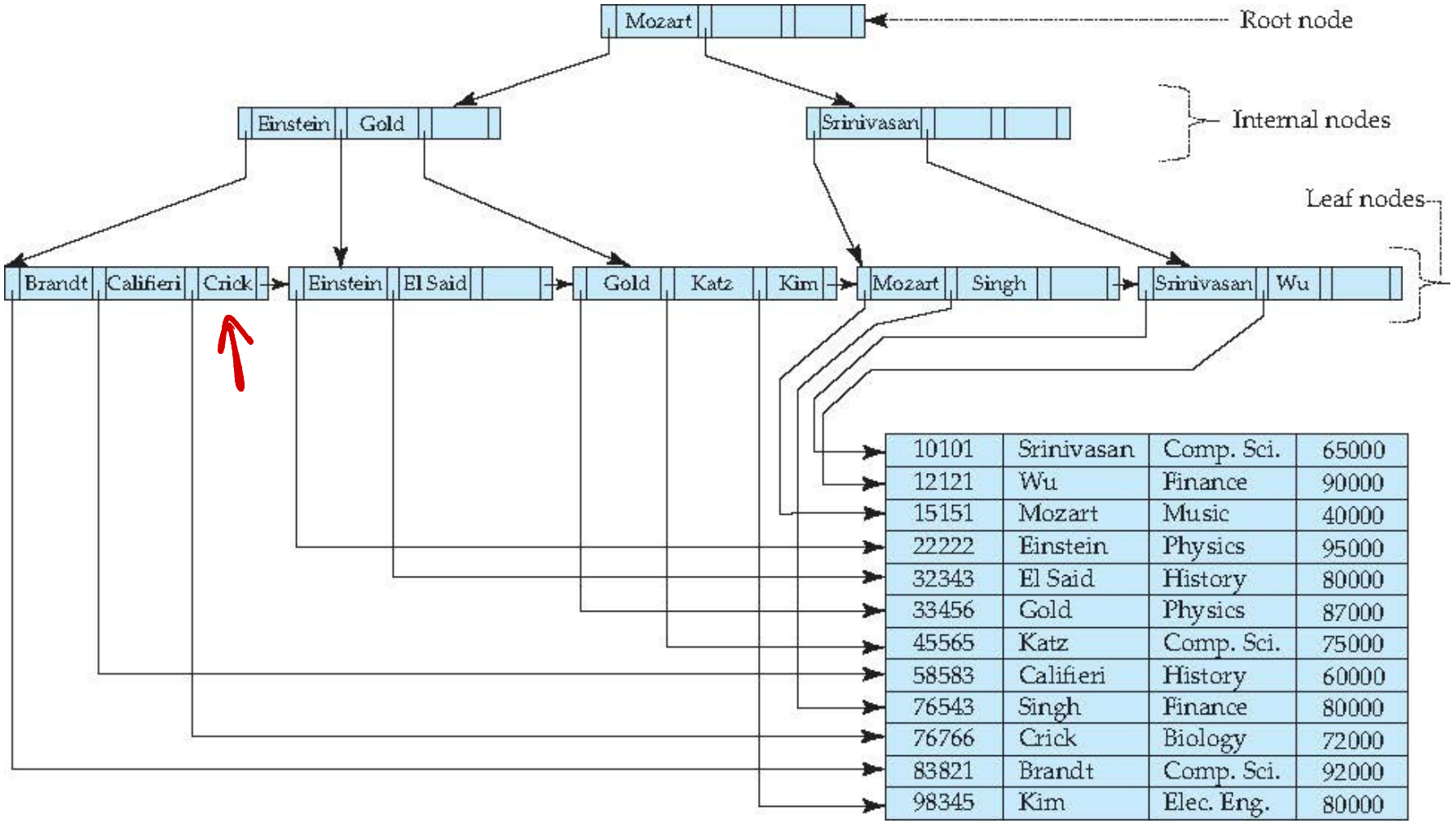
Option 2

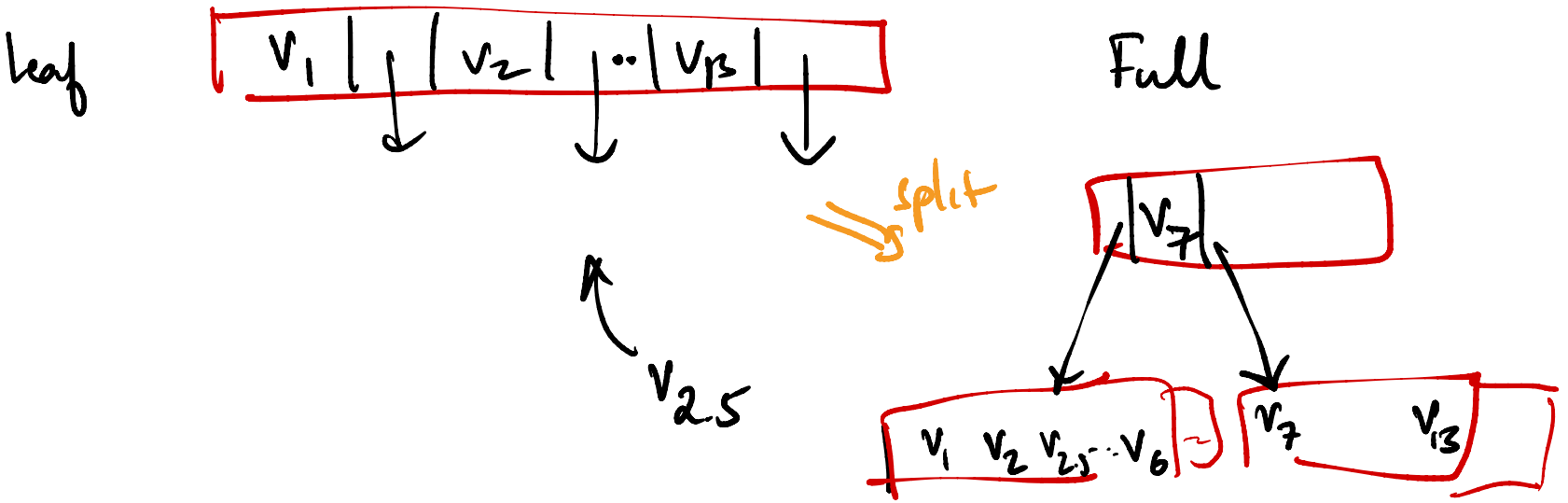
Store all actual (V, PV) at leaves

Leaf node

B^+-Tree

Example of B+-Tree




leaf
v_1 | | v_2 | .. | v_B |
Full
split
$v_{2.5}$
v_7
v_1 v_2 $v_{2.5}$ .. v_6
v_7
v_{13}

B+ tree

Every node other than root is at least half full

See the book (or any other reference) for details

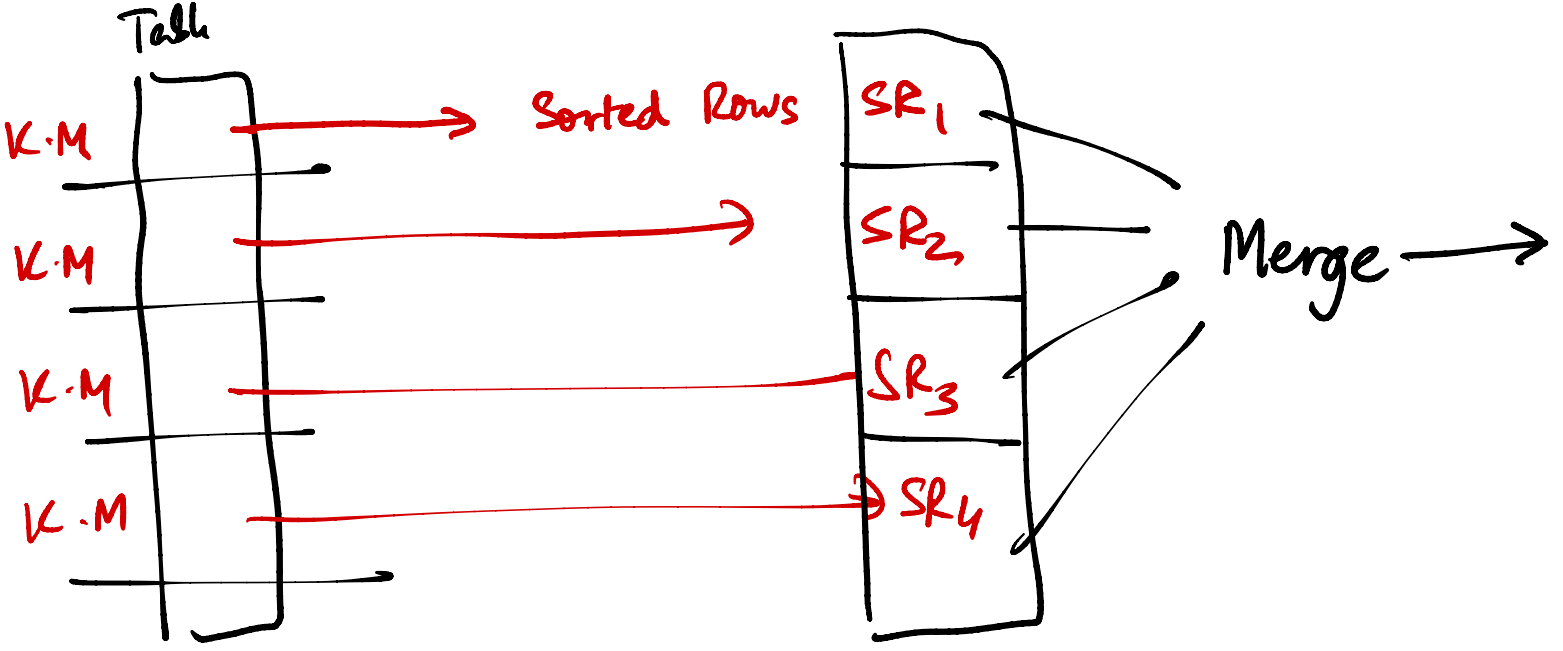
Sorting on disk

Table has N rows

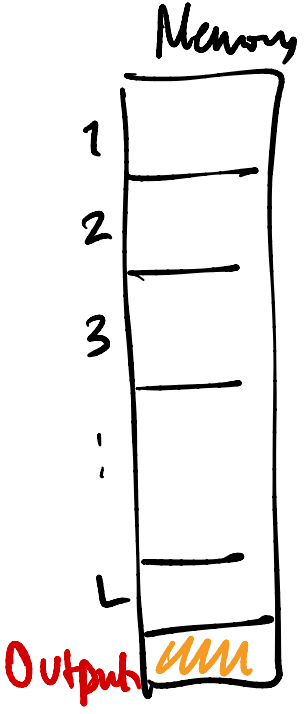
- K rows occupy 1 block
- M blocks fit in memory at a time

1. K · M rows fit in memory at one time
2. Any in memory computation is "free"

Can sort K·M rows at a time



Merge L subrelations

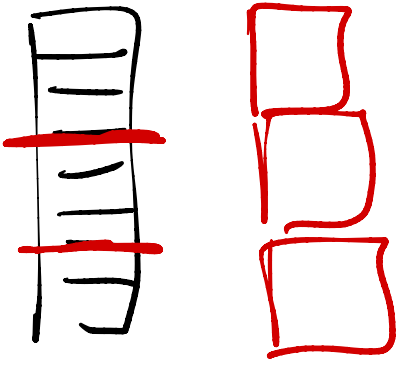


If $L > M-1$

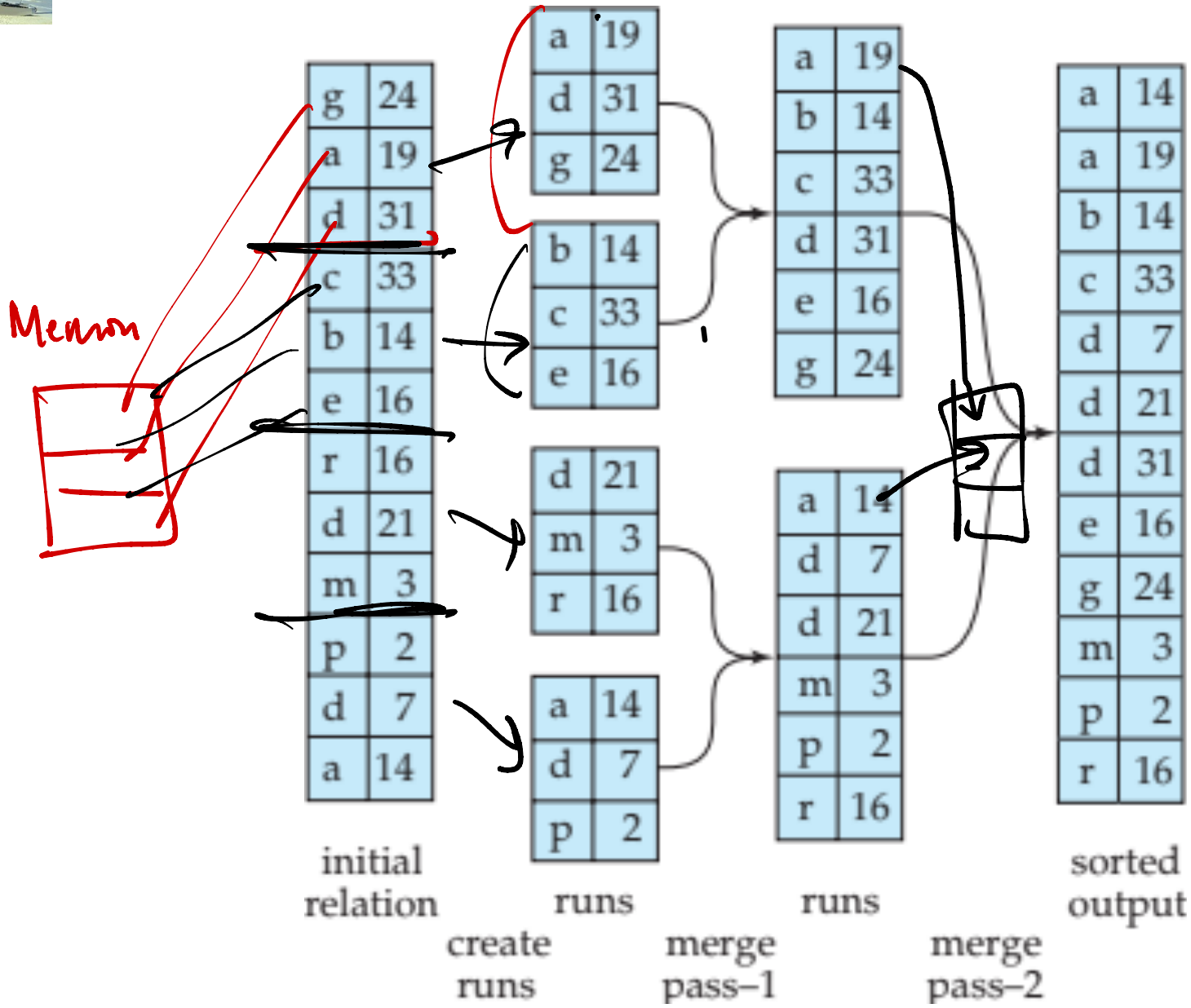
Merge $M-1 \Longrightarrow SR_1'$

Next $M-1 \Longrightarrow SR_2'$

Recursive external merge sort

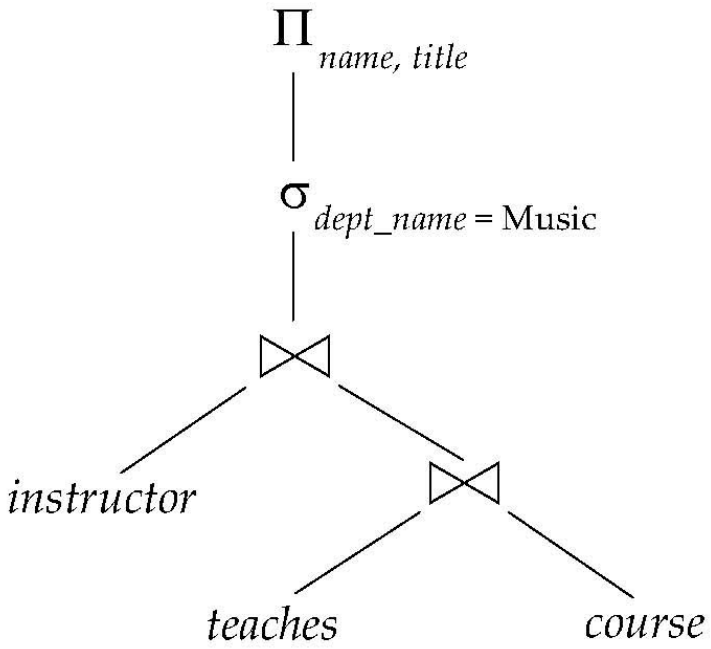


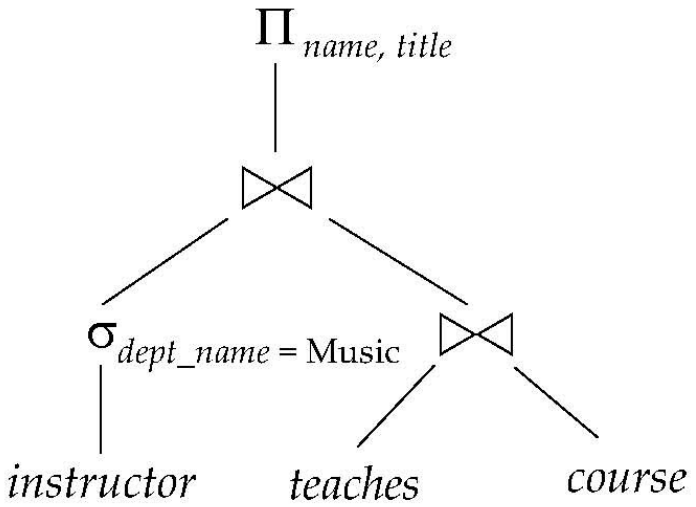
Example: External Sorting Using Sort-Merge



Introduction

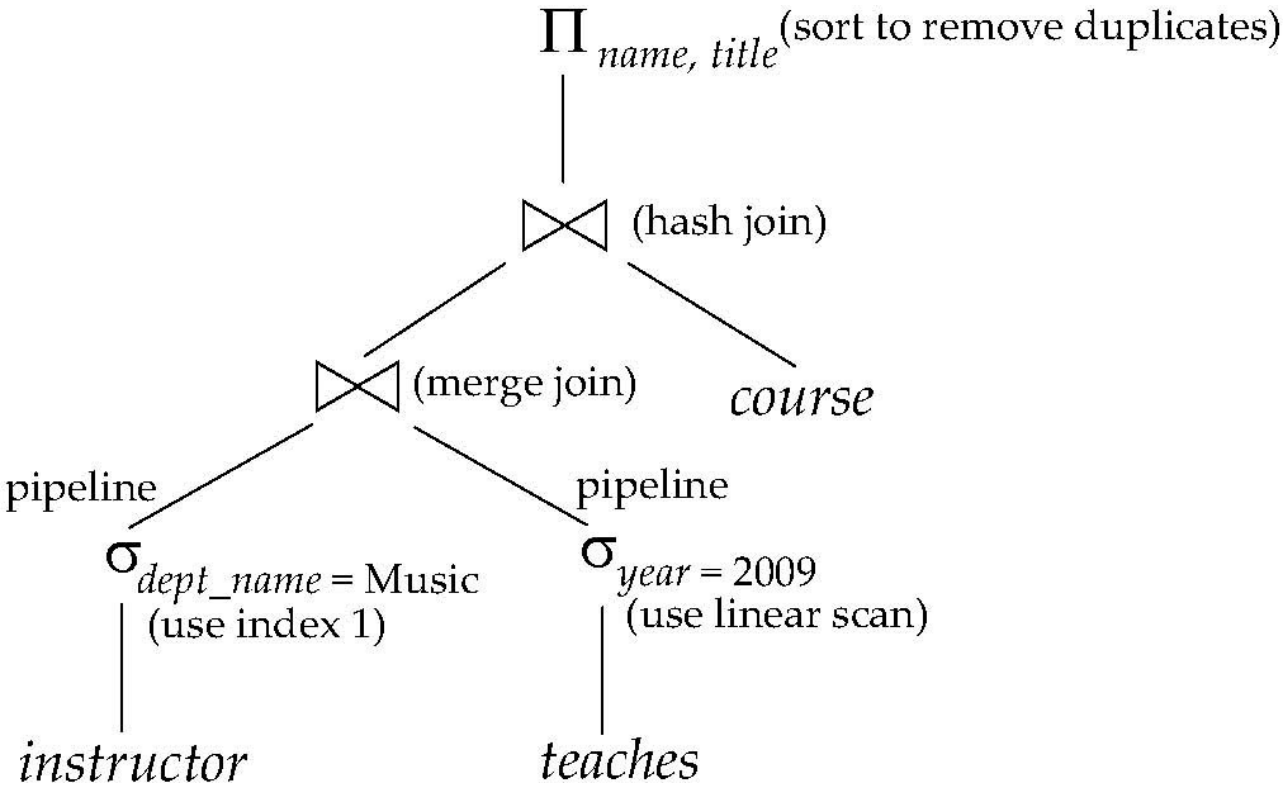
- Alternative ways of evaluating a given query
  - Equivalent expressions
  - Different algorithms for each operation





Introduction (Cont.)

- An **evaluation plan** defines exactly what algorithm is used for each operation, and how the execution of the operations is coordinated.



- Find out how to view query execution plans on your favorite database

Introduction (Cont.)

- Cost difference between evaluation plans for a query can be enormous
  - E.g. seconds vs. days in some cases
- Steps in **cost-based query optimization**
  1. Generate logically equivalent expressions using **equivalence rules**
  2. Annotate resultant expressions to get alternative query plans
  3. Choose the cheapest plan based on **estimated cost**
- Estimation of plan cost based on:
  - Statistical information about relations. Examples:
    - number of tuples, number of distinct values for an attribute
  - Statistics estimation for intermediate results
    - to compute cost of complex expressions
  - Cost formulae for algorithms, computed using statistics