UCLA

CS143: Index

Professor Junghoo “John” Cho

Topics to Learn

- Index
- Dense index vs. sparse index
- Primary index vs. secondary index
  (= clustering index vs. non-clustering index)
- Multi-level index
- Indexed Sequential Access Method (ISAM)

Basic Problem

- SELECT *
  FROM Student
  WHERE sid = 30

sid	name	GPA
20	Susan	3.5
60	James	1.7
70	Peter	2.6
40	Elaine	3.9
30	Christy	2.9

- How can we answer the query?

Random-Order File

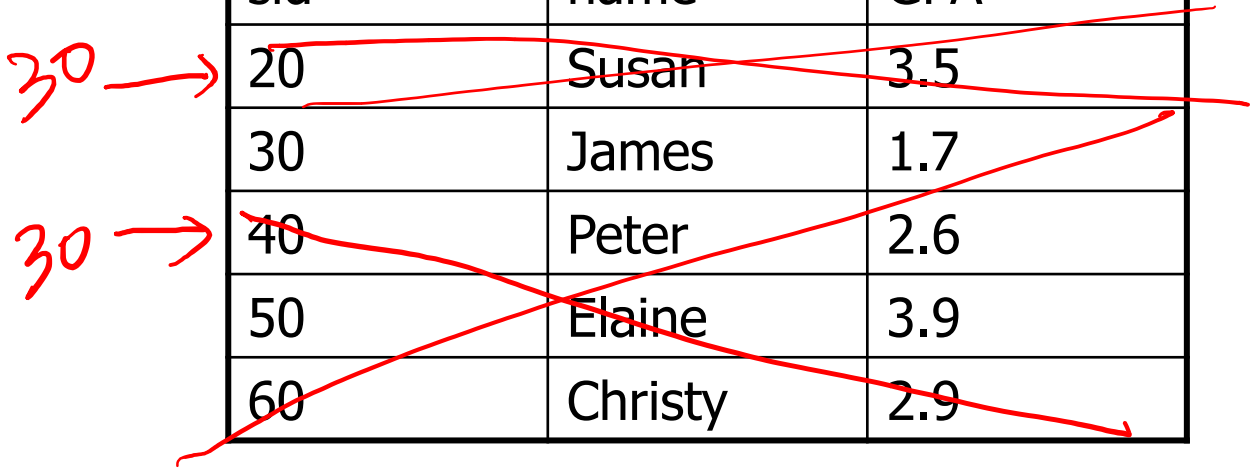
- How do we find sid=30?

sid	name	GPA
20	Susan	3.5
60	James	1.7
70	Peter	2.6
40	Elaine	3.9
30	Christy	2.9

Sequential File

- Table sequenced by sid. Find sid=30?

sid	name	GPA
20	Susan	3.5
30	James	1.7
40	Peter	2.6
50	Elaine	3.9
60	Christy	2.9



Binary Search

- 100,000 tuples
- Q: How many blocks to read?

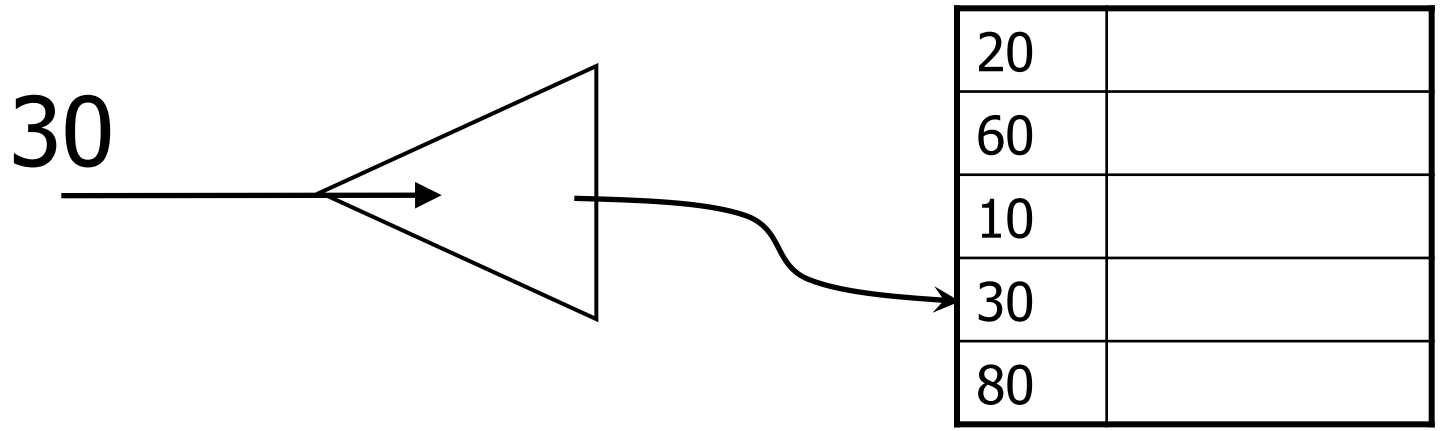
$\frac{100,000}{2^2}$ $\frac{100,000}{2^n} \leq 1$ $100,000 \leq 2^n$

$16.61 = \log_2 100,000 \leq n$ $17 \times 10ms = 170ms$

- Any better way?

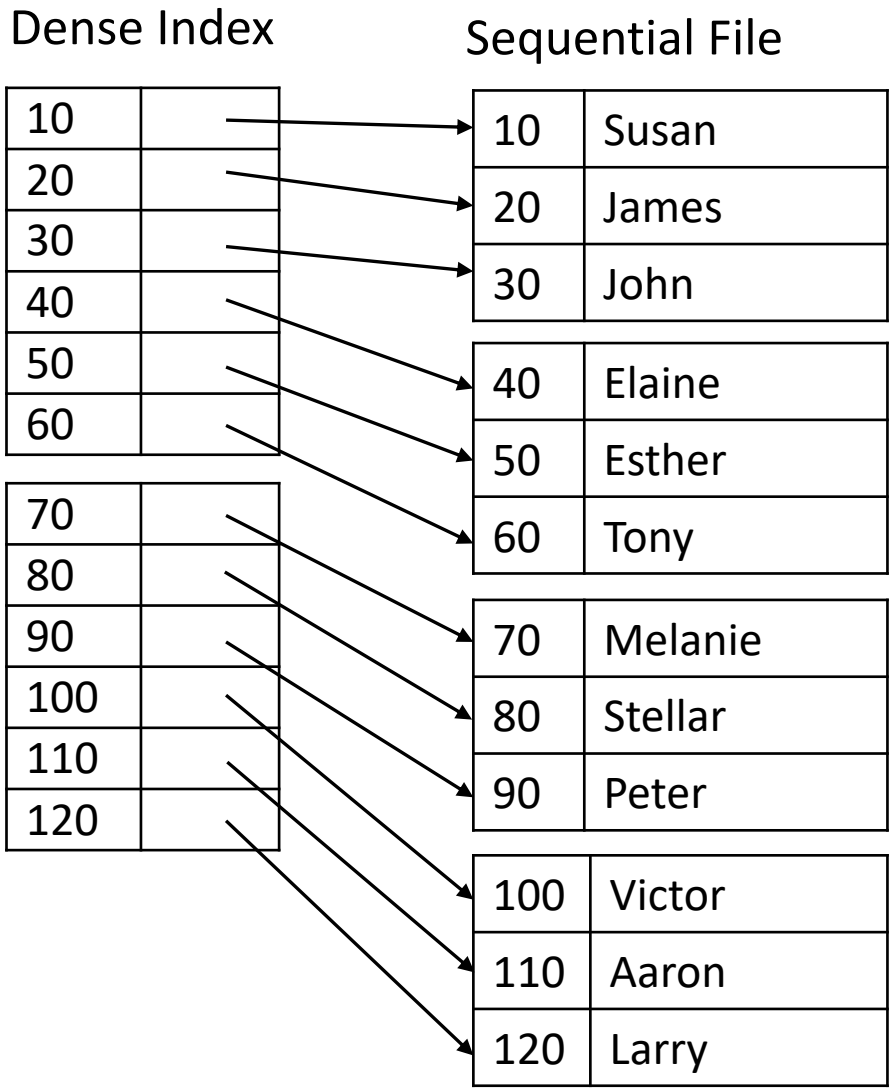
Index: Basic Idea

- Build an “index” on the table
  - An auxiliary structure to help us quickly locate a tuple given a “search key”



Dense, Primary Index

- Primary index (=clustering index)
  - Underlying table is sequenced by a key
  - Index is built on on the same key (= search key)
- Dense index
  - One (key, pointer) index entry per every tuple
- Search algorithm
  - Find the key from index and follow pointer
  - Maybe through binary search
- Q: Why dense index?
  - Isn't binary search on the file the same?



Why Dense Index?

- Example
  - 100,000,000 tuples (900-bytes/tuple)
  - 4-byte search key, 4-byte pointer
  - 4096-byte block. Unspanned tuples
- Q: How many blocks for table (how big)?

$\lfloor \frac{4096}{900} \rfloor = \lfloor 4.55 \rfloor = 4 \quad \lceil \frac{100,000,000}{4} \rceil = 25,000,000$

4K x 25M = 100GB

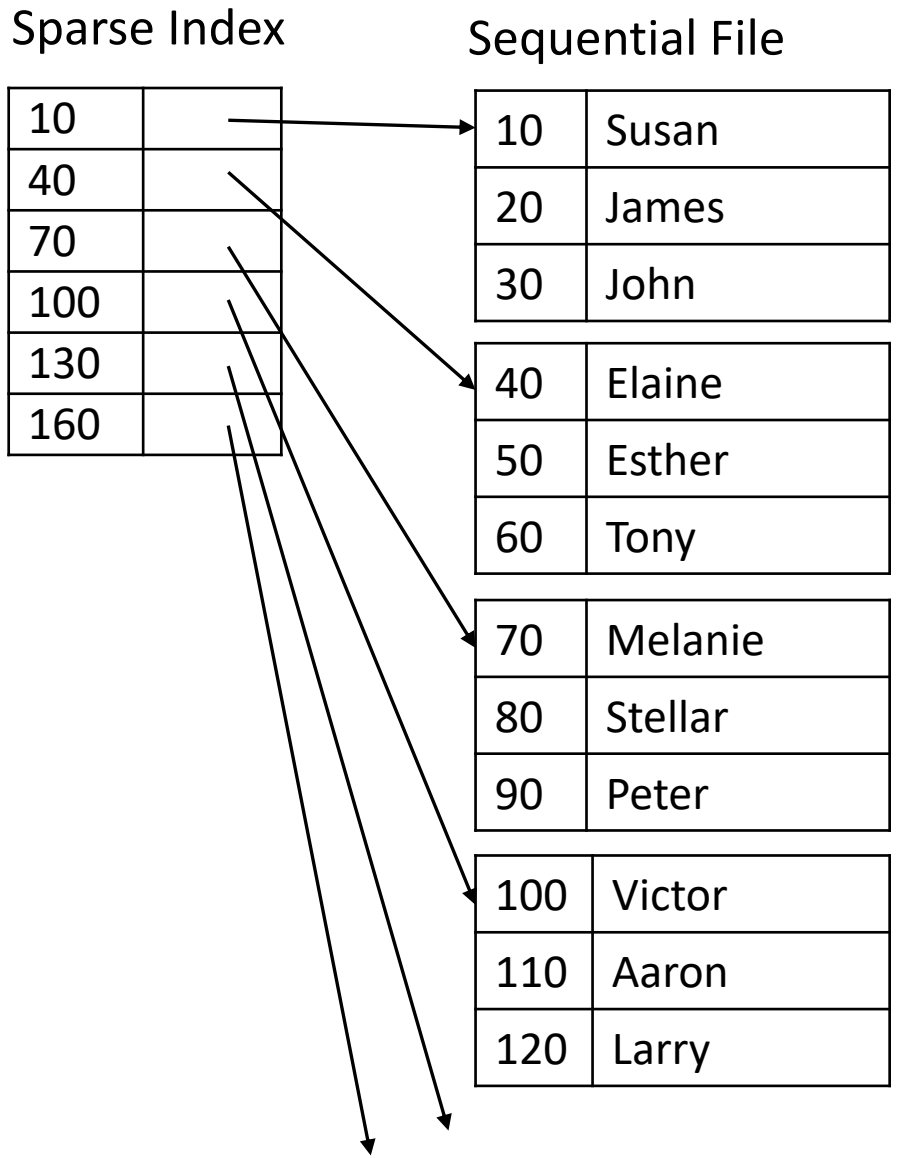
- Q: How many blocks for index (how big)?

$\lfloor \frac{4096}{8} \rfloor = 512 \quad \lceil \frac{100,000,000}{512} \rceil = 195,313$

4K x 195,313 ≈ 4K x 200,000 = 800MB

Sparse, Primary Index

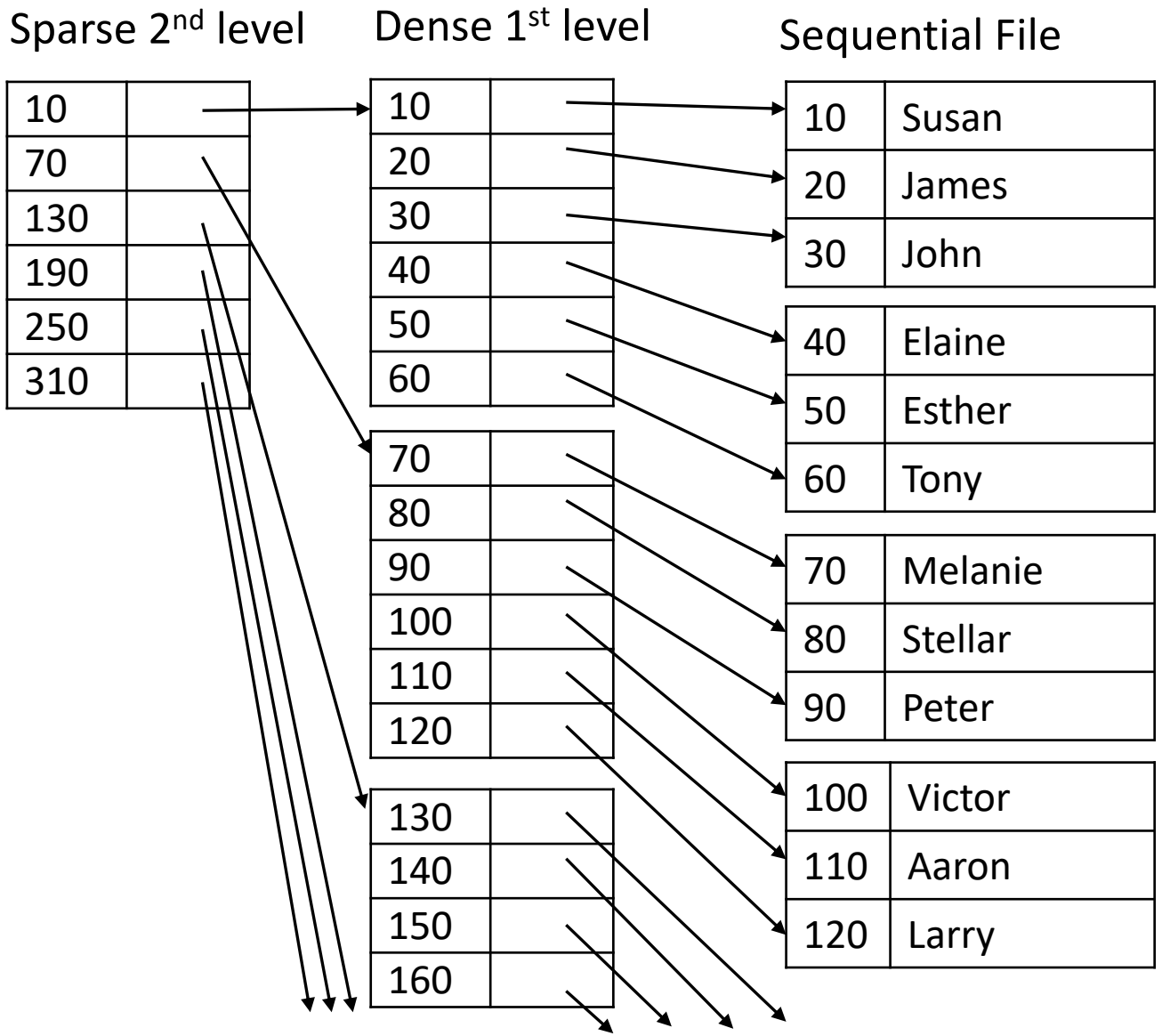
- Primary Index
  - Index is built on on the same search key as the underlying sequential file
- Sparse index
  - (key, pointer) pair per every “block”
  - (key, pointer) pair points to the first tuple in the block
- Q: How can we find 80?



Multi-level index

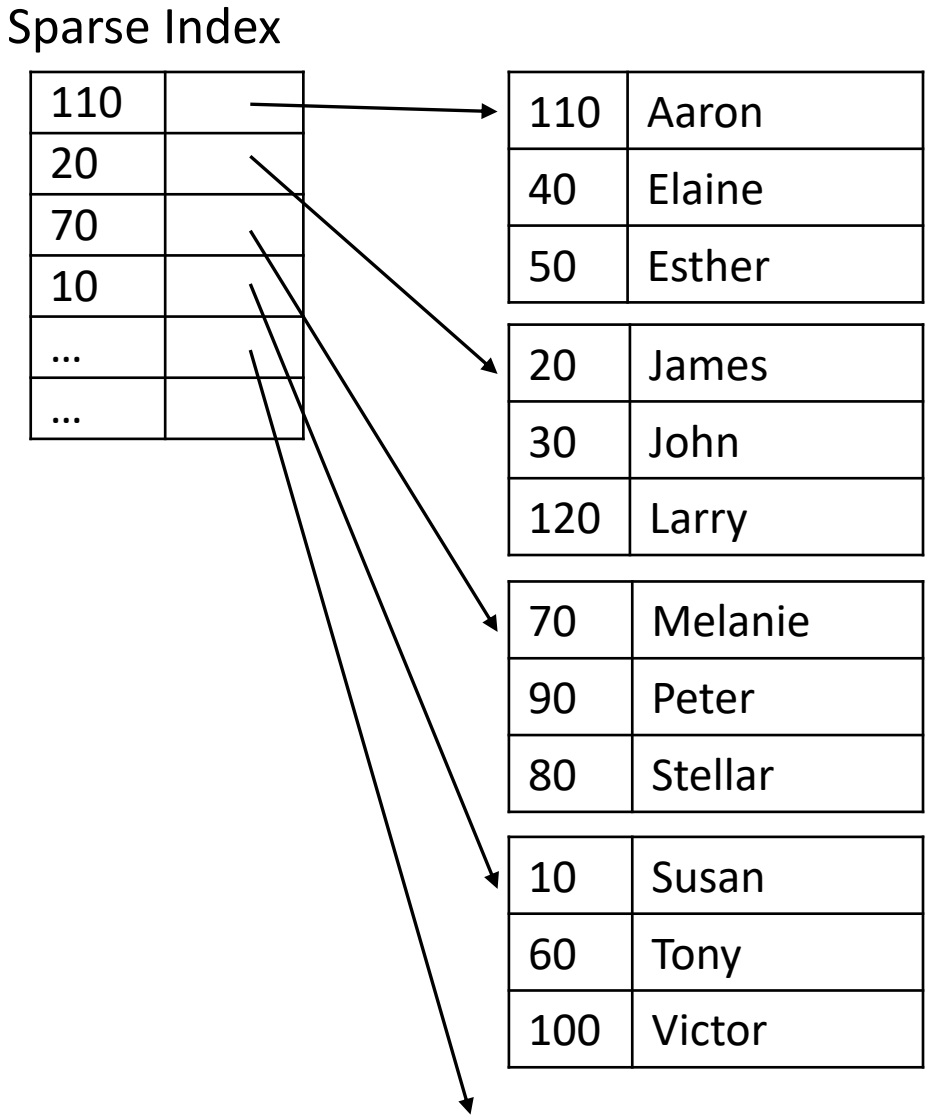
Q: Why multi-level index?

Q: Does dense, 2nd level index make sense?



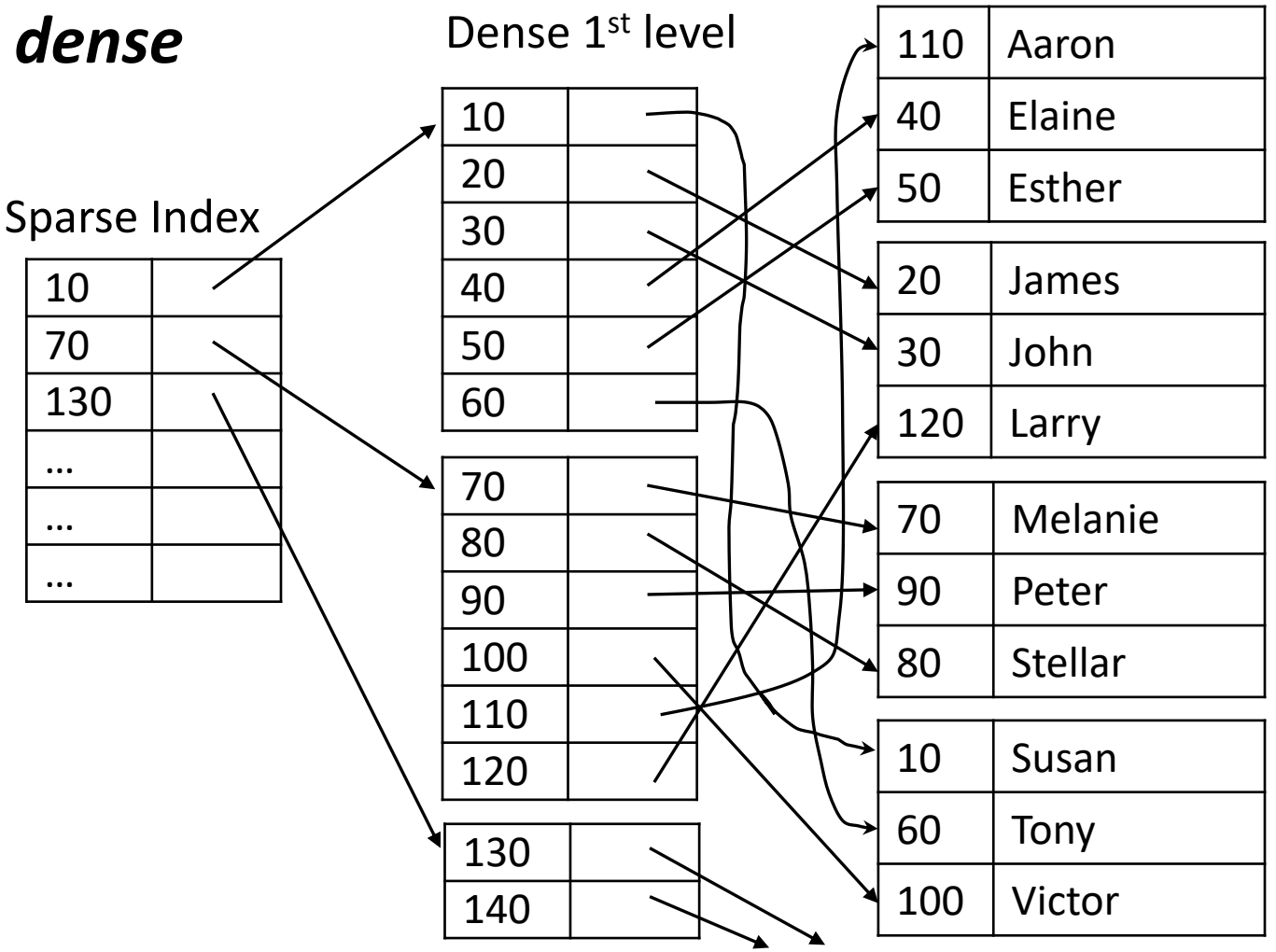
Secondary (non-clustering) Index

- Secondary (non-clustering) index
  - When tuples in the table are not ordered by the index search key
    - Index on a non-search-key for sequential file
    - Unordered file
- Q: What index?
  - Does sparse index make sense?



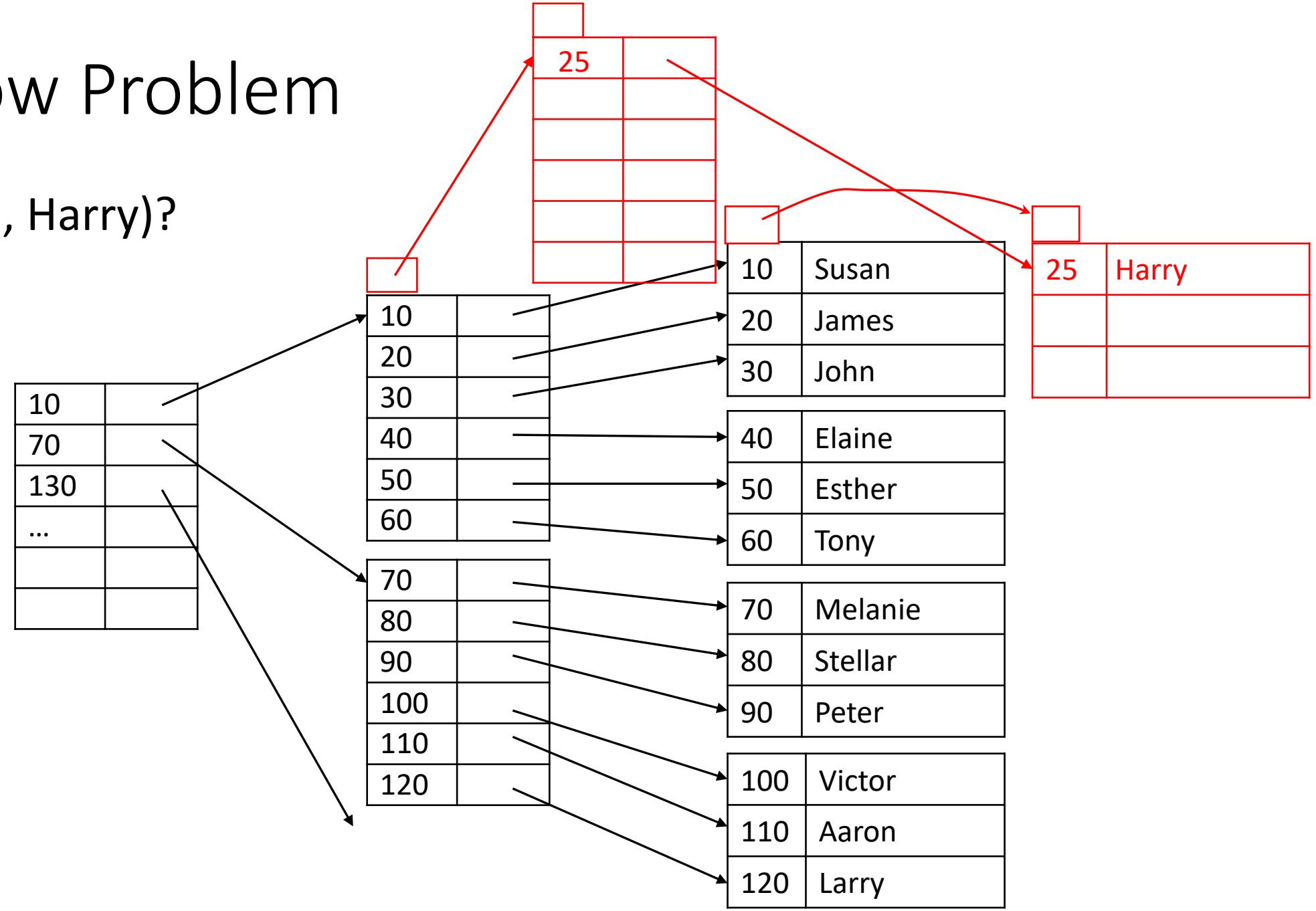
Secondary index

- First level must be ***always dense***
- Sparse from second level



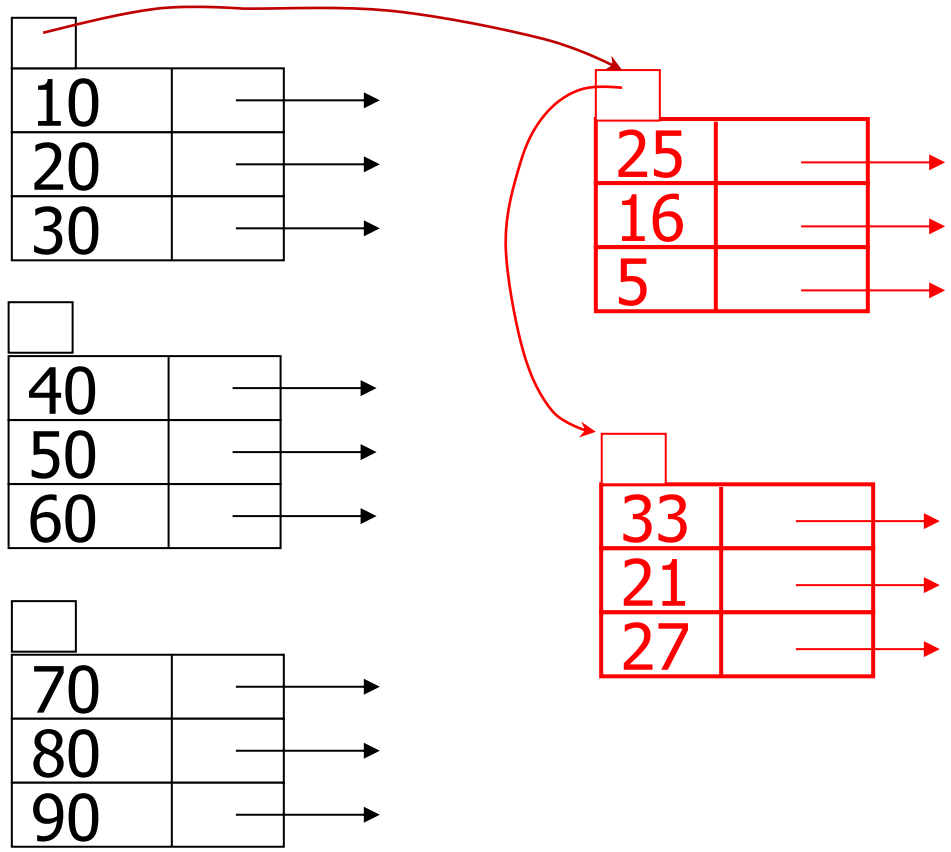
Overflow Problem

Q: Insert (25, Harry)?



Performance Problem after many insertions

- After many insertions, long chain of overflow pages



Indexed Sequential Access Method (ISAM)

- Advantage
  - Simple
  - Sequential blocks
- Disadvantage
  - Not suitable for updates
  - Becomes ugly (loses sequentiality and balance) over time

Index Creation in SQL

- CREATE INDEX <indexname> ON <table>(<attr>,<attr>,…)
- Example
  - CREATE INDEX sid_idx ON Student(sid)
    - Creates a B+tree on the attributes
    - Speeds up lookup on sid

Primary (Clustering) Index

- MySQL:
  - Primary key becomes the clustering index
- DB2:
  - CREATE INDEX idx ON Student(sid) CLUSTER
  - Tuples in the table are sequenced by sid
- Oracle: Index-Organized Table (IOT)
  - CREATE TABLE T (
    ...
) ORGANIZATION INDEX
  - B+tree on primary key
  - Tuples are stored at the leaf nodes of B+tree
- Periodic reorganization may still be necessary to improve range scan performance

Important terms

- Search key (≠ primary key)
- Primary index vs. secondary index
  - Clustering index vs. non-clustering index
- Dense index vs. sparse index
- Multi-level index
- Indexed Sequential Access Method (ISAM)