ISAM Indexes

Topics to Learn

- Important concepts
  - Dense index vs. sparse index
  - Primary index vs. secondary index
    (= clustering index vs. non-clustering index)
  - Tree-based vs. hash-based index
- Tree-based index
  - Indexed sequential file
  - B+-tree
- Hash-based index
  - Static hashing
  - Extendible hashing

Basic Problem

-
  ```
  SELECT *
  FROM Student
  WHERE sid = 40
  ```

sid	name	GPA
20	Elaine	3.2
70	Peter	2.6
40	Susan	3.7

- How can we answer the query?

Random-Order File

- How do we find sid=40?

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=40?

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

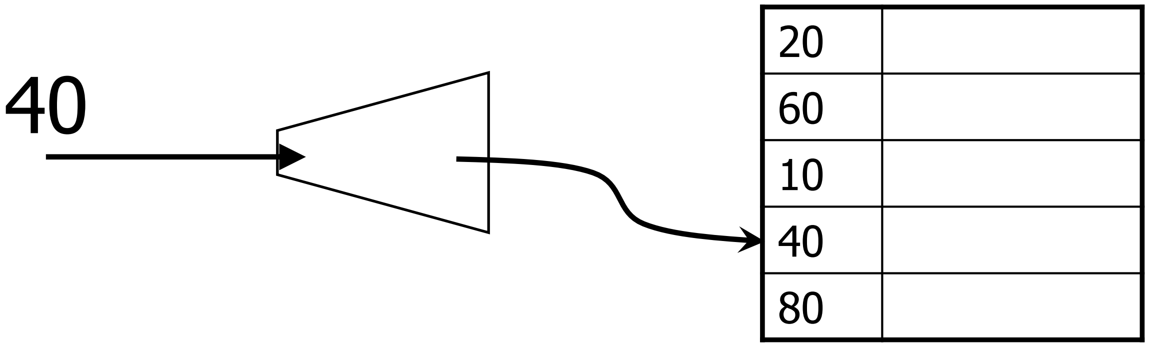
Binary Search

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

- Any better way?
  - In a library, how do we find a book?

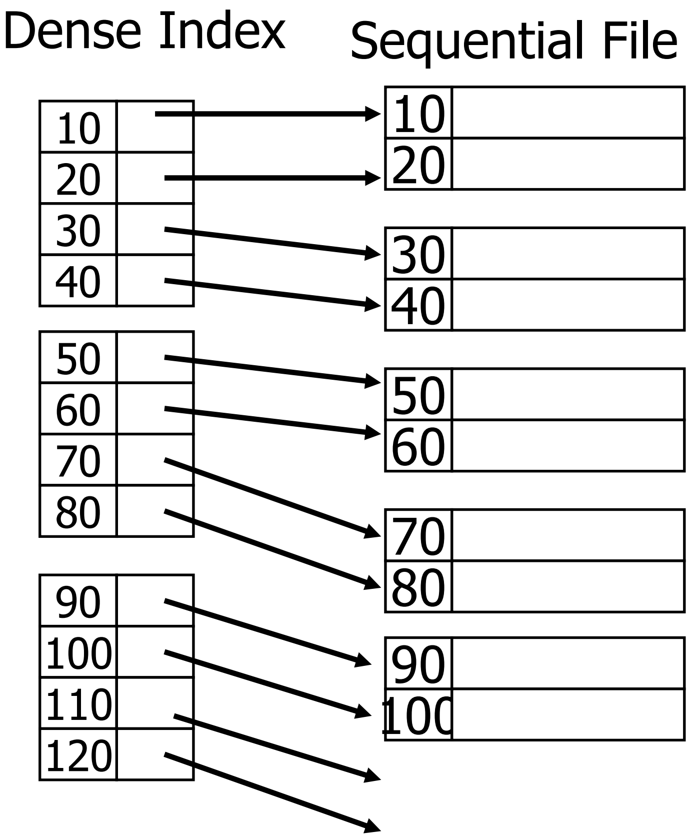
Basic Idea

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



20	
60	
10	
40	
80	

Dense, Primary Index



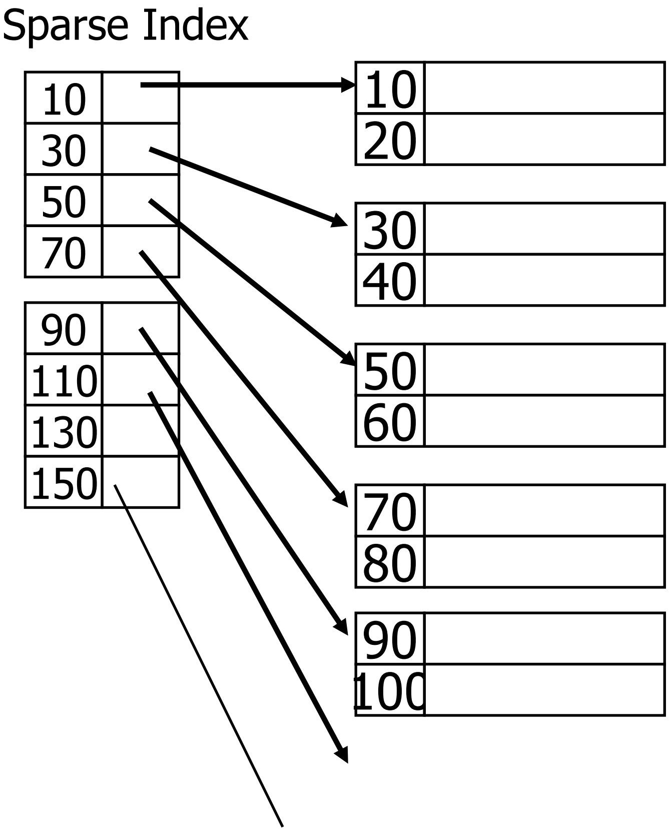
- Primary index (clustering index)
  - Index on the search key
- Dense index
  - (key, pointer) pair for every record
- Find the key from index and follow pointer
  - Maybe through binary search
- Q: Why dense index?
  - Isn’t binary search on the file the same?

Why Dense Index?

- Example
  - 10,000,000 records (900-bytes/rec)
  - 4-byte search key, 4-byte pointer
  - 4096-byte block. Unspanned tuples

- Q: How many blocks for table (how big)?
  - How many unspanned records per block: 4096 div 900= 4
  - How many blocks 10,000,00 div 4= 2,500,000.
- Q: How many blocks for index (how big)?
  - For dense index, one search key and one pointer per record requiring 4+4=8 bytes
  - 8 x 10,000,000= 80,000,000 bytes needed for dense index
  - How many keys per block: 4096 div 8= 512
  - How many blocks for indices: 80000000 div 512=156,250 Thus 156250 are needed for the index.

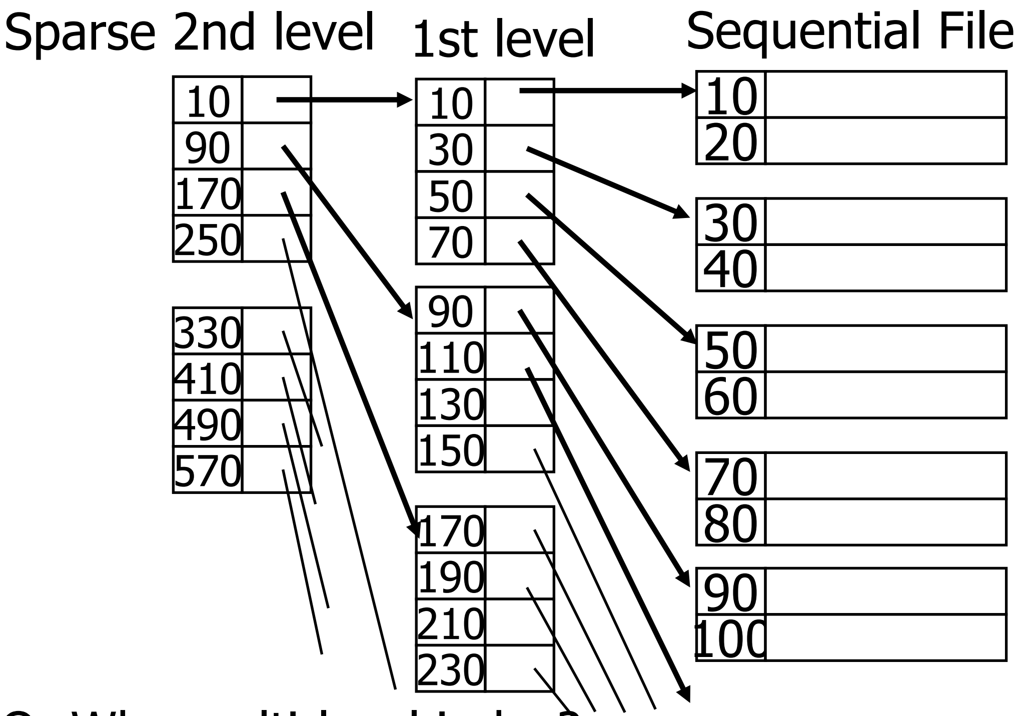
Sparse, Primary Index



- Sparse index
  - (key, pointer) pair per every "block"
  - (key, pointer) pair points to the first record in the block

- Q: How can we find 60?

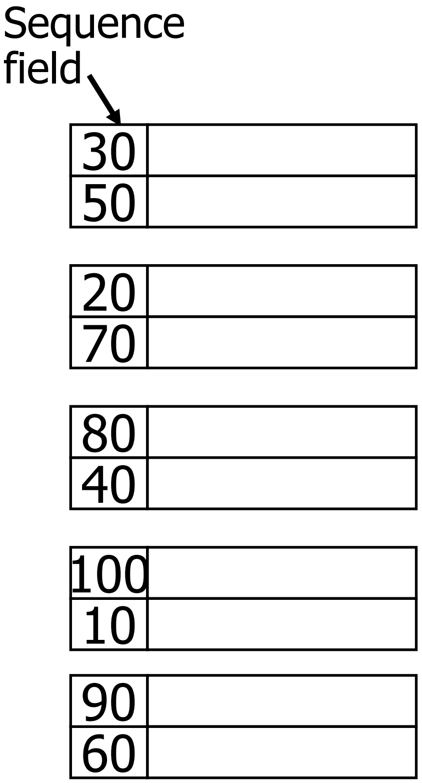
Multi-level index



Q: Why multi-level index?

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

Secondary (non-clustering) Index



Sequence field

30
50

20
70

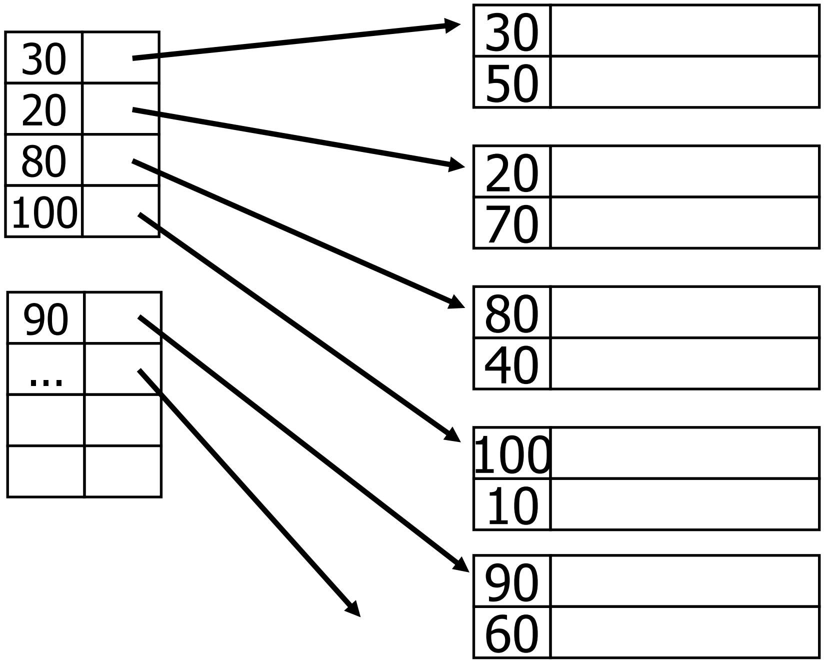
80
40

100
10

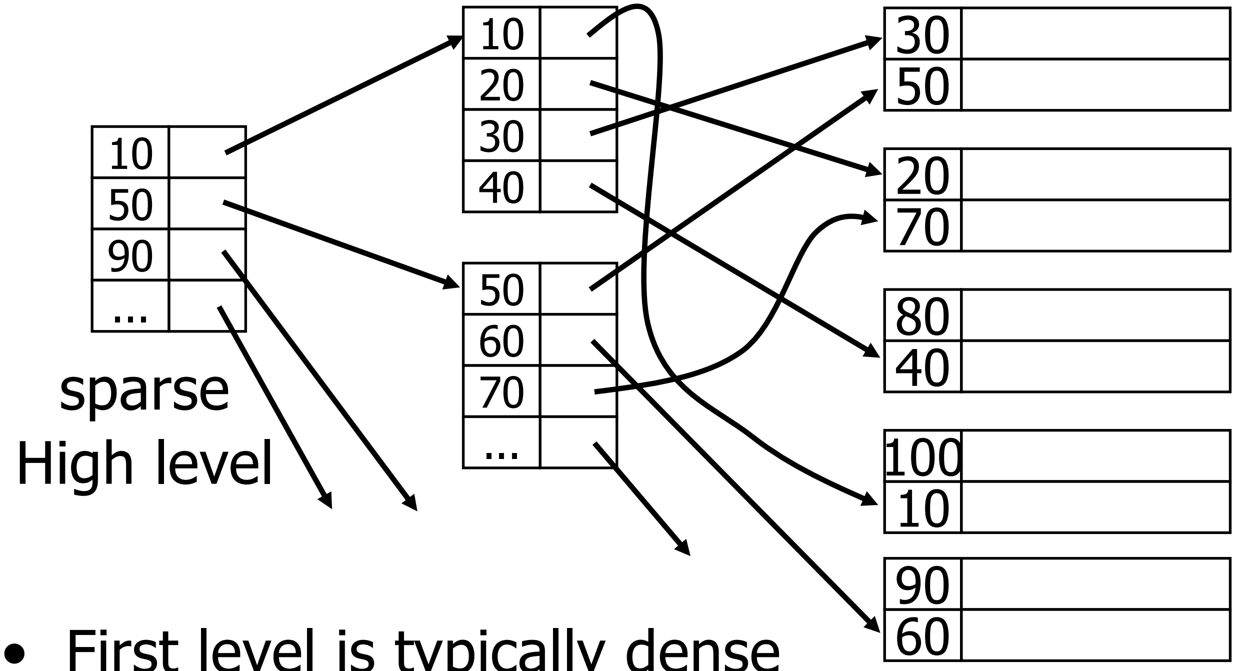
90
60

- 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?

Sparse and secondary index?



Secondary index



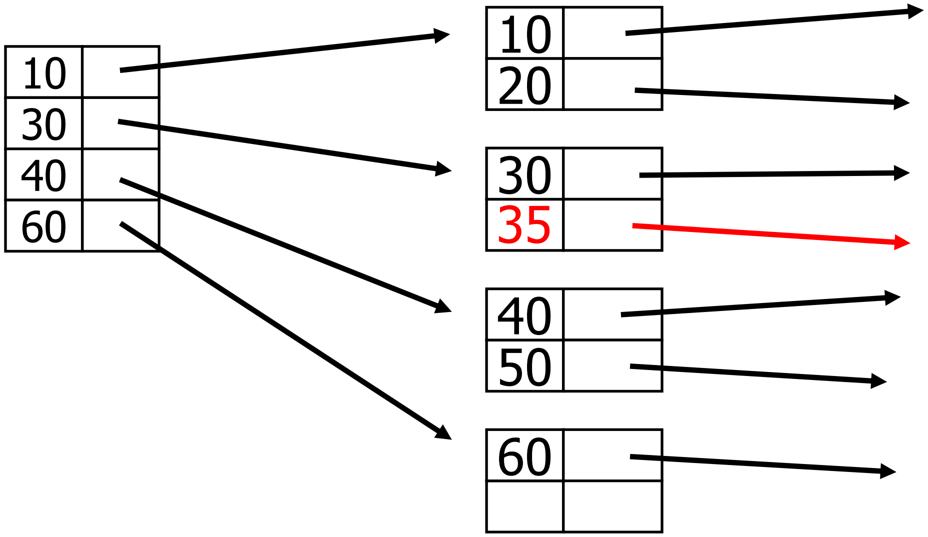
- First level is typically dense
- Sparse from the second level

Important terms

- Dense index vs. sparse index
- Primary index vs. secondary index
  - Clustering index vs. non-clustering index
- Multi-level index
- Indexed sequential file
  - Sometimes called ISAM (indexed sequential access method)
- Search key (≠ primary key)

Insertion

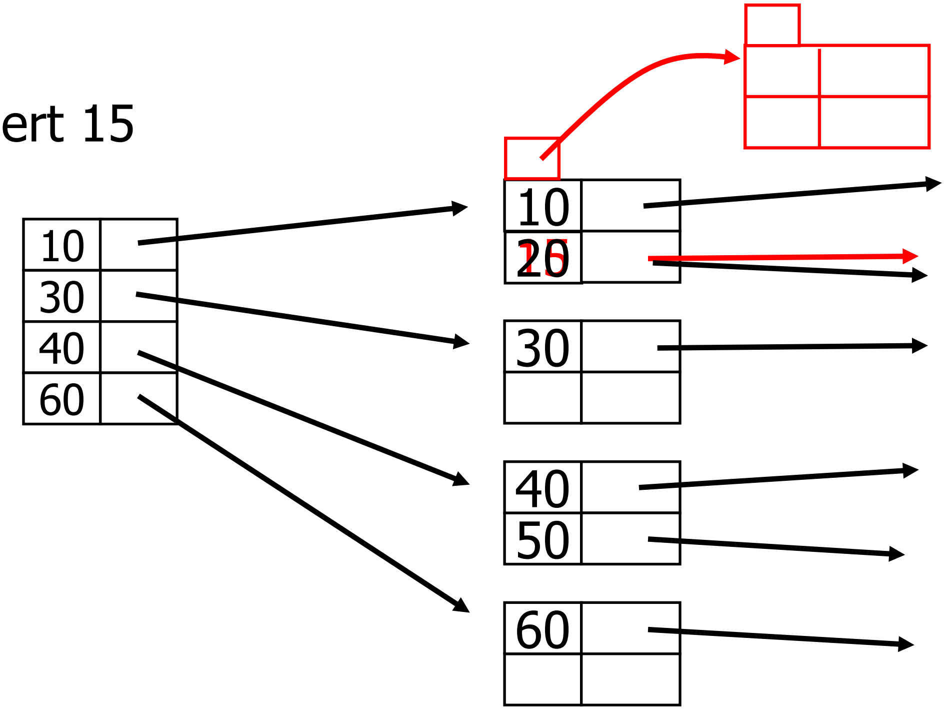
Insert 35



Q: Do we need to update higher-level index?

Insertion

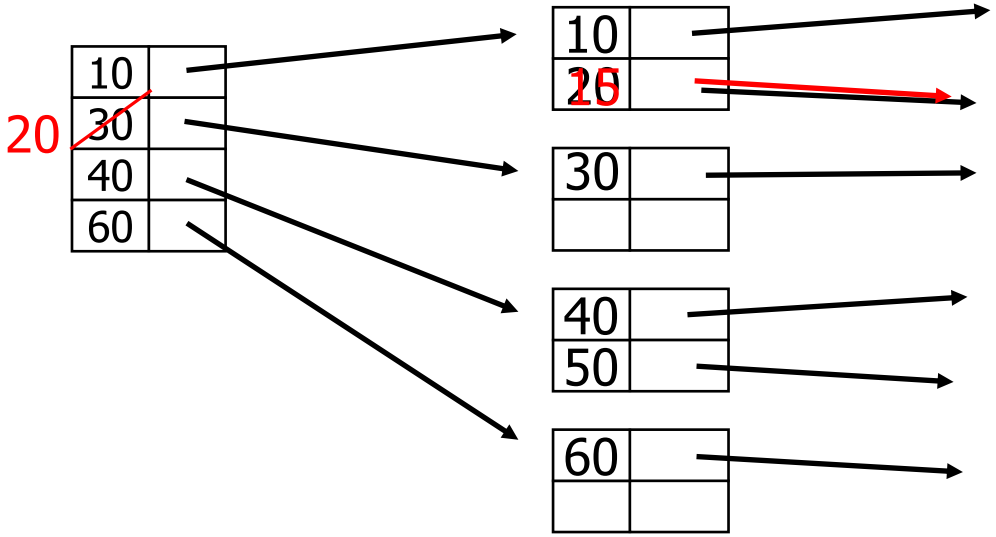
Insert 15



Q: Do we need to update higher-level index?

Insertion

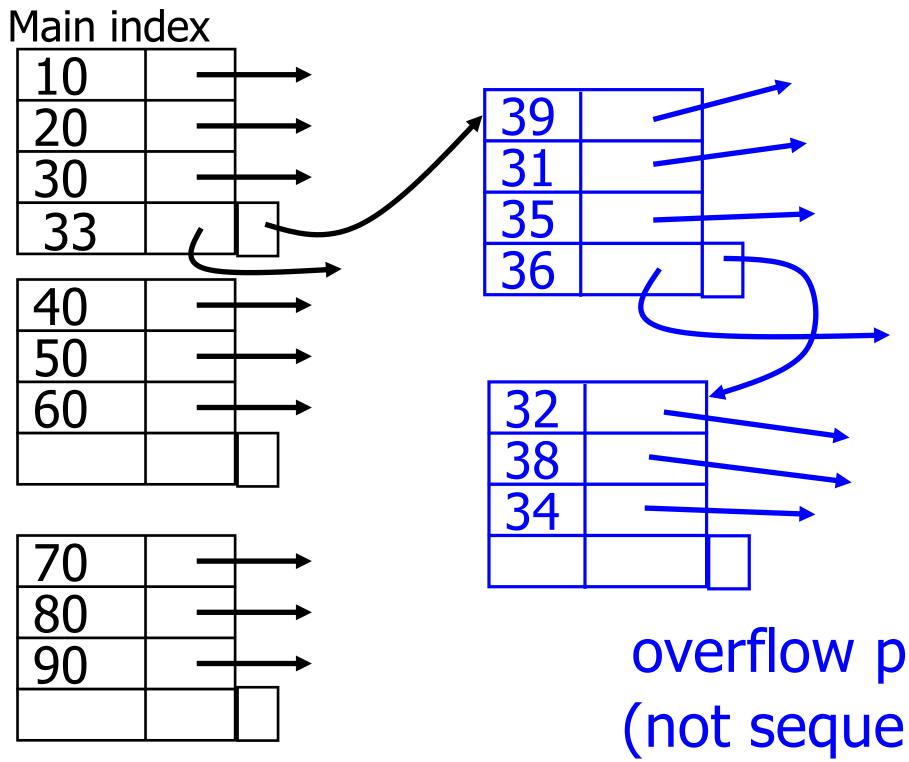
Insert 15



Q: Do we need to update higher-level index?

Potential performance problem

After many insertions…



overflow pages
(not sequential)

Traditional Index (ISAM)

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