PRINCIPLES OF DATA ORGANISATION

File Organization

MOTIVATION

- How to organize data?
- Implementation

HEAP FILE

- Note: not heap!!
- Variable-length records ~ file is not homogeneous
  - E.g., a log file
- Data not sorted in any way, a record placed always at the end of the file
- Usually used along with another supporting structure
- Insert : O(1)
  - Fetch the last block (keep in memory) in the file and append the new record
- Find : O(N / b)
  - N – number of records
  - b – average blocking factor
    - $b = \lfloor B/R \rfloor$
      - B = block size
      - R = record size
  - Whole file needs to be scanned

UNSORTED SEQUENTIAL FILE

- Fixed-size record
- Data not sorted in any way
  - Heap file
- Suitable when data are collected without any relationship to other data
  - We can query for a record using its index

- Insert : O(1)
- Find : O(N / b)

Block	Name	Department	…
0	Galvin Janice	Purchasing	
	Walters Rob	Marketing	
	Brown Kevin	Marketing	
1	Walters Rob	Developlment	
	Duffy Terri	Research	
	Brown Kevin	PR	
2	Duffy Terri	Developlment	
	Walters David	Production	
	Brown Kevin	Purchasing	
3	Matthew Gigi	Purchasing	
	Walters Rob	PR	
	…	…	

SORTED SEQUENTIAL FILE

- Fixed size record
- Records sorted in the file according to the primary search key
  - According to only one – the most often searched

Fetch

- Sequential scan
  - O (N / b)
- Binary search
  - Direct-access medium
  - O (log (N / b))
- Range query
  - Find start and read k records

Block	Name	Department	…
0	Brown Kevin	PR	
	Brown Kevin	Purchasing	
1	Brown Kevin	Marketing	
	Duffy Terri	Developlment	
2	Duffy Terri	Research	
	Galvin Janice	Purchasing	
3	Matthew Gigi	Purchasing	
	Walters David	Production	
4	Walters Rob	Marketing	
	Walters Rob	Developlment	
5	Walters Rob	PR	
	…	…	

SORTED SEQUENTIAL FILE – MODIFICATION

Insert

- Inserting a new record is costly
  - All the following records would have to be shifted
- Auxiliary file/blocks called overflow file/bucket need to be established where the new records are inserted
  - Outside the primary file
- The (main) file is periodically reorganized

Update

- Simple if the update does not include the primary search key
  - If so, it is delete and insert

Delete

- Deleted records are not directly removed
  - Reorganization would have to take place
- A bit designating deleted records is set
- Deleted records are removed during periodical reorganization

What if we want to access the data using various attributes?

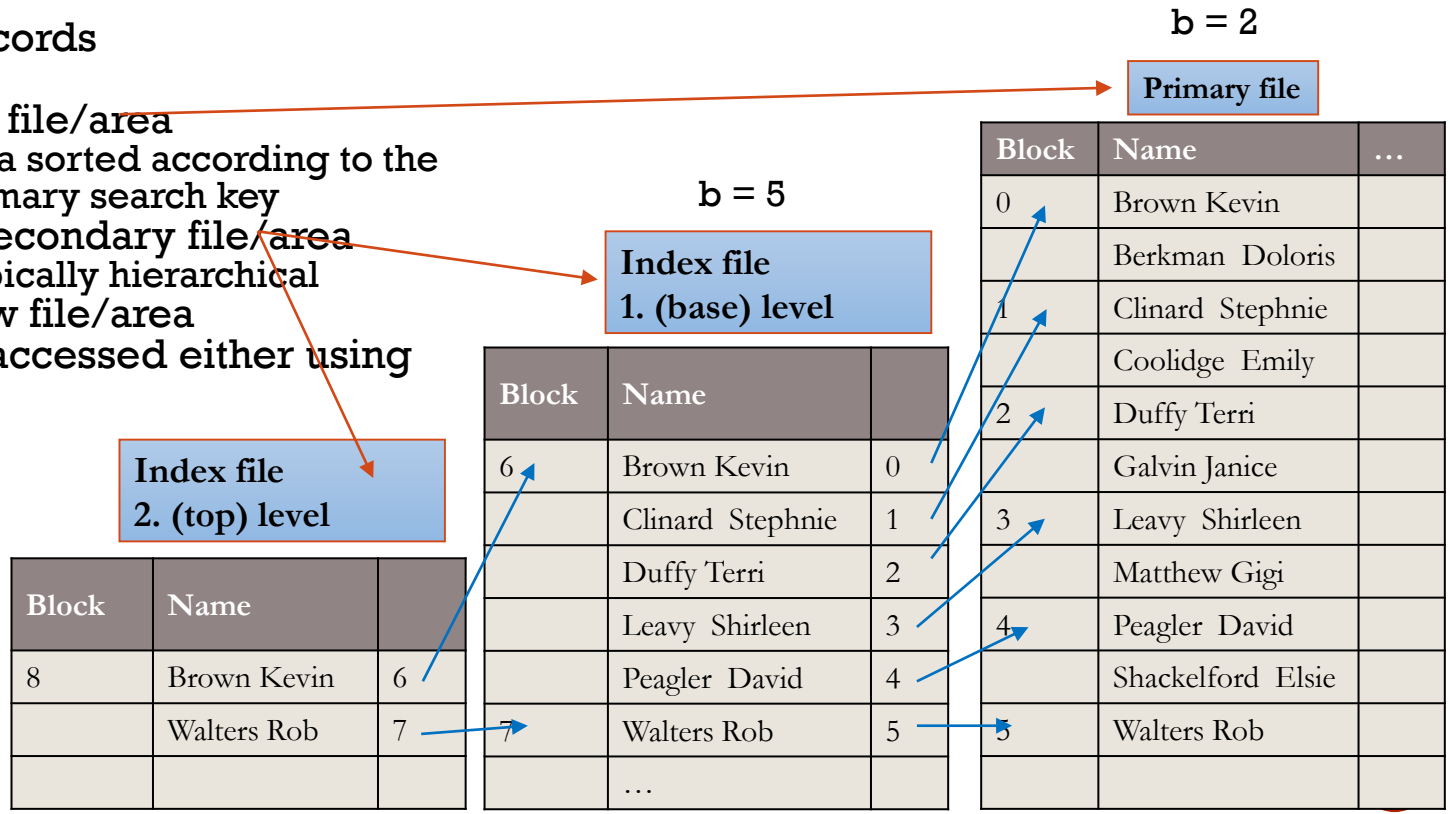
INDEX

- Motivation:
  - Can we do it better than sorting the data?
  - Yes
    - Recall binary tree, a-b tree, …

- An index is an auxiliary structure for a data file that consists of a specifically arranged structure containing key-pointer pairs
  - E.g., name-pointer to the block with the record
- Storage of the index
  - Main memory
    - Cashed
  - Secondary memory
    - Accessing index must also be taken into account when computing the find/fetch time
    - In real use: blocking factor of the index >> blocking factor of the primary file

INDEXED SORTED SEQUENTIAL FILE

- Fixed size records
- Structure
  - Primary file/area
    - Data sorted according to the primary search key
  - Index/secondary file/area
    - Typically hierarchical
  - Overflow file/area
- Data can be accessed either using the index or sequentially



INDEXED SEQUENTIAL FILE – FETCH

Searching for a specific value (query key)

- Check the top level of the index and identify a key-value pair with the highest value lower than the query key
- Fetch the block referenced by the value
- Repeat the previous steps with lower index levels until a primary file block is reached
  - Fetch time depends on the height of the tree
  - Each level = disc access
- Search the primary file block for the specified key

Searching for a range of values

- Search for the lower bound key of the interval
- Sequentially scan the blocks of the primary file until the record corresponding to the upper bound key is found

INDEXED SEQUENTIAL FILE – FETCH

Search for Galvin Janice (G)
Search for Galvin Janice – Walters Rob (G – W)

b = 2

Primary file

Block	Name	…
0	Brown Kevin	
	Berkman Doloris	
1	Clinard Stephnie	
	Coolidge Emily	
2	Duffy Terri	
	Galvin Janice	
3	Leavy Shirleen	
	Matthew Gigi	
4	Peagler David	
	Shackelford Elsie	
5	Walters Rob	

b = 5

Index file
1. (base) level

Block	Name	
6	Brown Kevin	0
	Clinard Stephnie	1
	Duffy Terri	2
	Leavy Shirleen	3
	Peagler David	4
7	Walters Rob	5
	…	

Index file
2. (top) level

Block	Name	
8	Brown Kevin	6
	Walters Rob	7

INDEXED SEQUENTIAL FILE – INSERT

- When an index is created, index nodes are fixed and do not change during modifications of the primary file
  - Index structure is static
    - Later we will see that it does not have to be
- New records need to be stored in reserved areas (pockets) within the primary file
  - Long pockets decrease efficiency
- Overflown data are inserted into a new block (created dynamically) – overflown block
  - Outside the primary file
  - Buckets can be chained and therefore theoretically the ISF does not need to be rebuilt
    - But decrease performance
- Pointer to the overflow area
  - for each record in a block
    - More space
    - Shorter sequences in the overflow area
  - for each block

INDEXED SEQUENTIAL FILE

Pros

- Fast access using primary search key
- Shares pros of the sequential file

Cons

- Fast access only when using primary search key
  - Otherwise sequential scan
- Problems with primary file when updating
  - Pockets slow down data access
  - Occasional reorganisation (also slow)

INDEXED FILE ORGANIZATION

- Allows to search the file according to different attributes without the need to scan the whole file sequentially
- The primary file stays unsorted or is sorted according to one key only (primary index)
  - Sorted = we need to keep the ordering
    - If sorted using an artificial key, range queries are not common
  - Unsorted – e.g., heap with additional smart structure
- For each query key an index file can be built
  → one primary data file, multiple index files
- Basically corresponds to a standard database table
  - One table
  - Multiple indexes built over it (possibly of different types)

INDEX

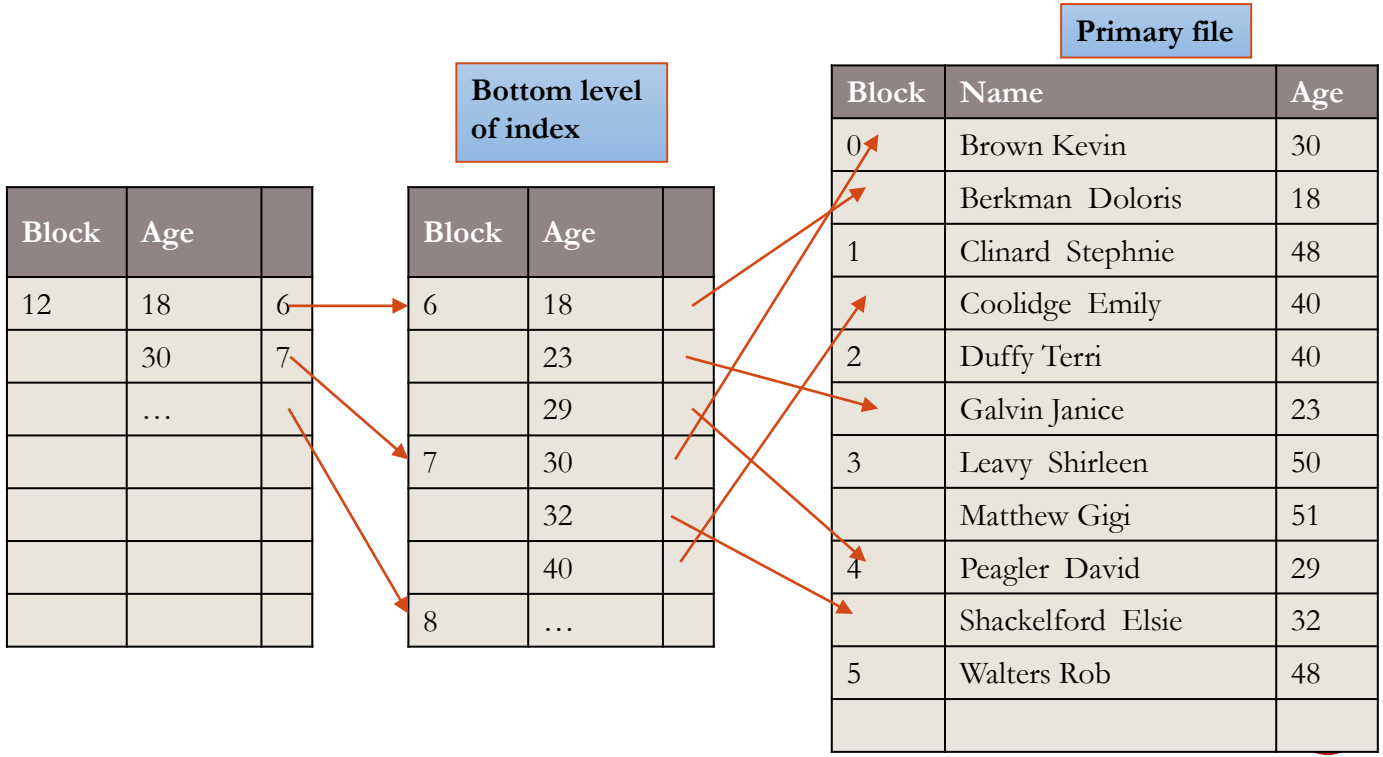
Primary index

- Index over the attribute based on which the records in the primary file are sorted
  - Only one
- If the value of the primary attribute is modified, the file needs to be reorganised → should be relatively invariable
- Well-suited for range queries
- There does not have to be a primary index
- It is desirable to keep it in memory
  - Small keys (integer, not string)

Secondary index

- There can be multiple secondary indexes
- We do not index blocks of the primary file, but a sorted list of indexed values (with pointers to the blocks with the data)
  - The bottom level of the index = we index records, not blocks
  - Next levels = we index blocks (with sorted records)
- Range queries for long ranges can be very expensive

SECONDARY INDEX



INDEX

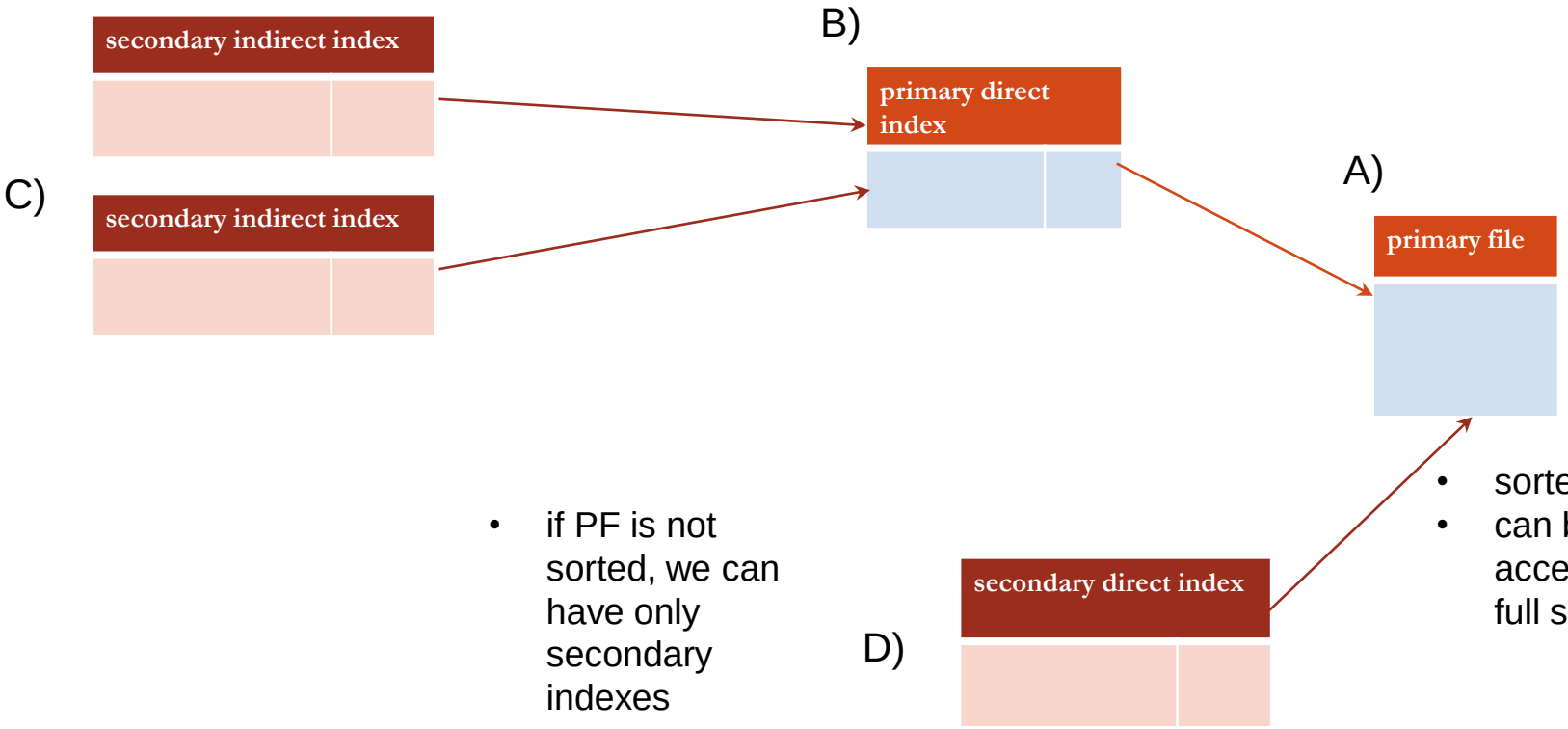
Direct index

- Index is bound directly to records
  - Pointers to the primary data file
- Primary file reorganization → modification of indexing structures

Indirect index

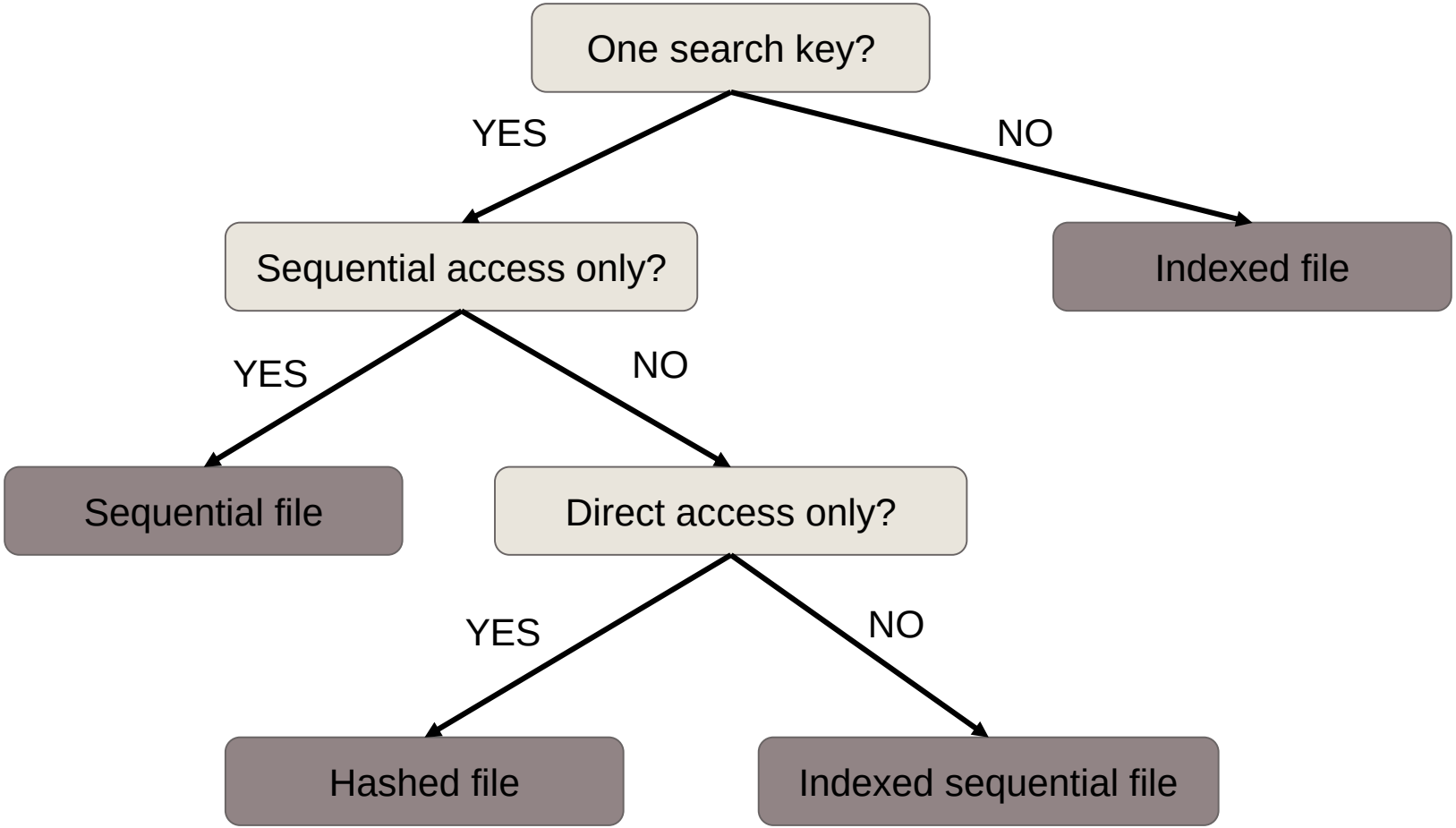
- Contains keys of the data (which are in the primary index)
  - Not pointers to the primary file
- Accessing a record needs one more access to the primary index
- If the primary file is reorganised, the secondary indexes stay intact

INDEX


secondary indirect index
B)
primary direct index
C)
secondary indirect index
A)
primary file
• if PF is not sorted, we can have only secondary indexes
secondary direct index
D)
• sorted/unsorted
• can be accessed using full scan

HASHED FILE ORGANIZATION

- Direct access with one unique key
- Use hash function to map records to pages/blocks addresses
- If the data can not fit into a page/block when inserting, an overflow strategy is employed
- Placement within the page is not specified
- When file is being reorganized, the pages are filled only to, e.g., 80%
  - To avoid overflow with next insert
  - The value depends on expected insert count


One search key?
YES
NO
Sequential access only?
Indexed file
YES
NO
Sequential file
Direct access only?
YES
NO
Hashed file
Indexed sequential file