Unit V - Indexes

Chapter 10 -

Indexing Structures for Files Storage of Databases

- Databases are stored physically as files of records.
- Each record is collections of data values that can be interpreted as facts about entities, their attributes and their relationships
- Records should be stored on disk in a manner that make it possible to locate them efficiently
  whenever they are needed
- Records in a file can be organized in sorted or unsorted

Indexes

- An index is an auxiliary file which is usually defined on a single field of a data file for efficient access to records.
- Any field of the data file can be used to create an index
- Multiple indexes on different fields can be constructed on the same data file
- A field used for creating an index is called indexing field
- Index typically stores
  - each value of the indexing field along with
  - a list of the pointers that reference to the record or block of records in the data file

- Index file structure is generally of the form: **<field value, record/block pointer>** which is ordered by field value
- To search a record or records in a data file based on a certain selection condition on a indexing field, one has to access the index which points to the required records

- Thus, an index provides an alternative ways of accessing the records without affecting the physical storage of records on disk
- The index file usually occupies considerably less disk blocks than the data file because its entries are much smaller

Types of Indexes

- Different types of indexes can be created based on properties of indexing field
- Indexing field can be ordering key field, ordering non-key field or non-ordering field of the data file

Primary Index

- defined on the key field of an ordered data file
- includes one index entry *for each block* in the data file; the index entry has the key field value for the *first record* in the block, which is called the *block anchor*

Clustering index

- defined on the non key field of an ordered data file
- includes one index entry ***for each distinct value*** of the field; the index entry points to the first data block that contains records with that field value.

FIGURE 14.1: Primary index on the ordering key field of the file shown in Figure 13.7

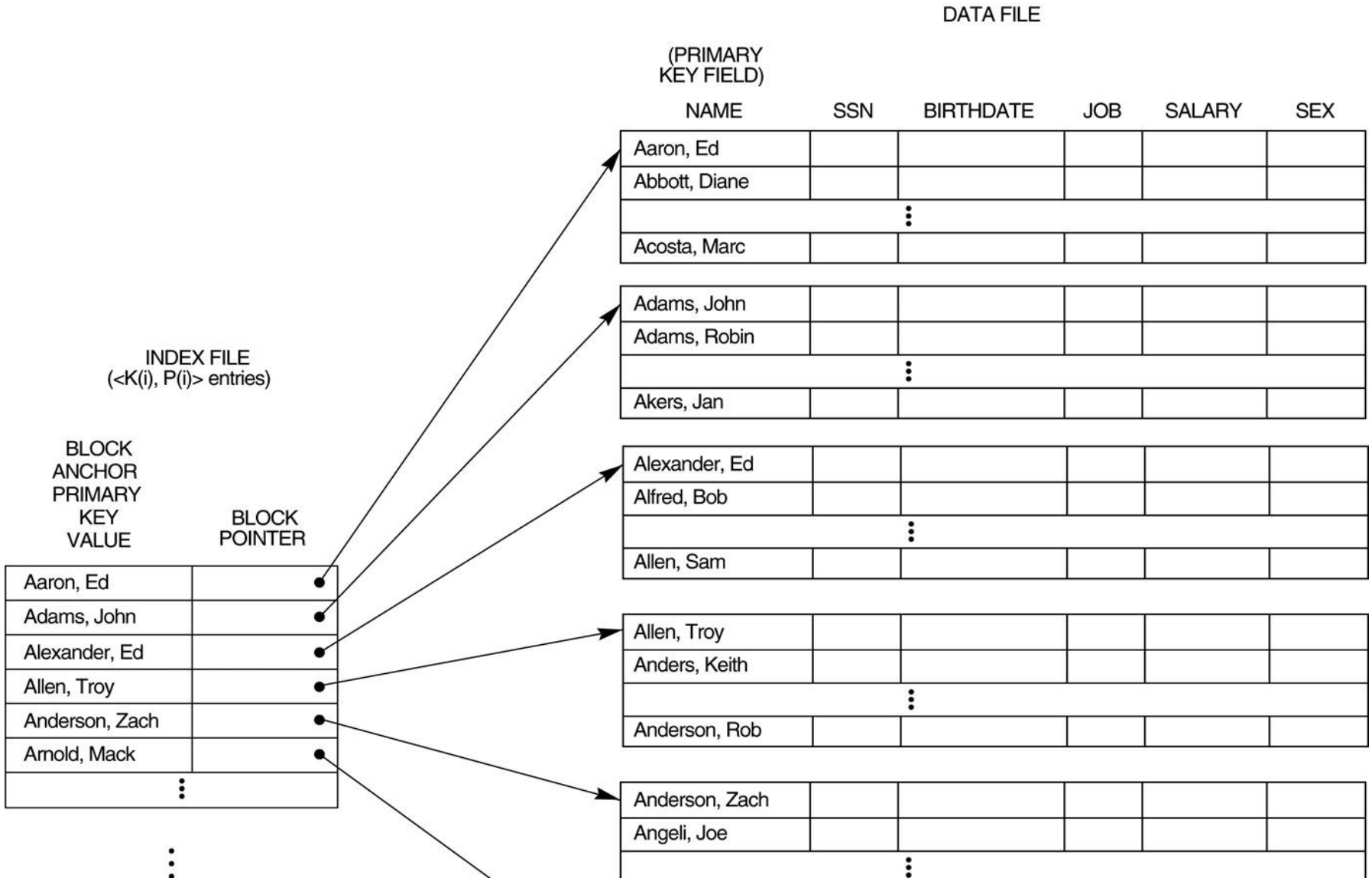
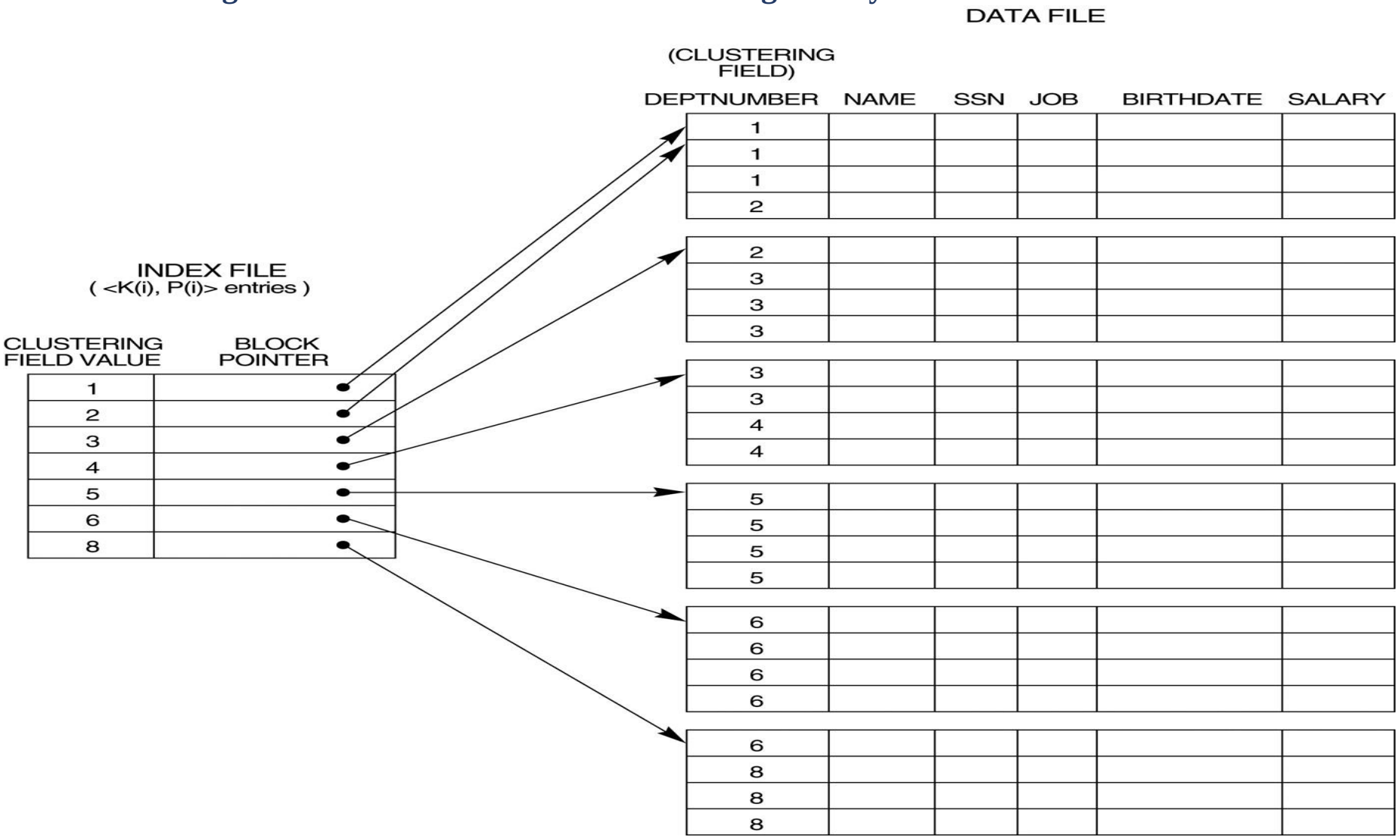


FIGURE 14.2

A clustering index on the DEPTNUMBER ordering nonkey field of an EMPLOYEE file.



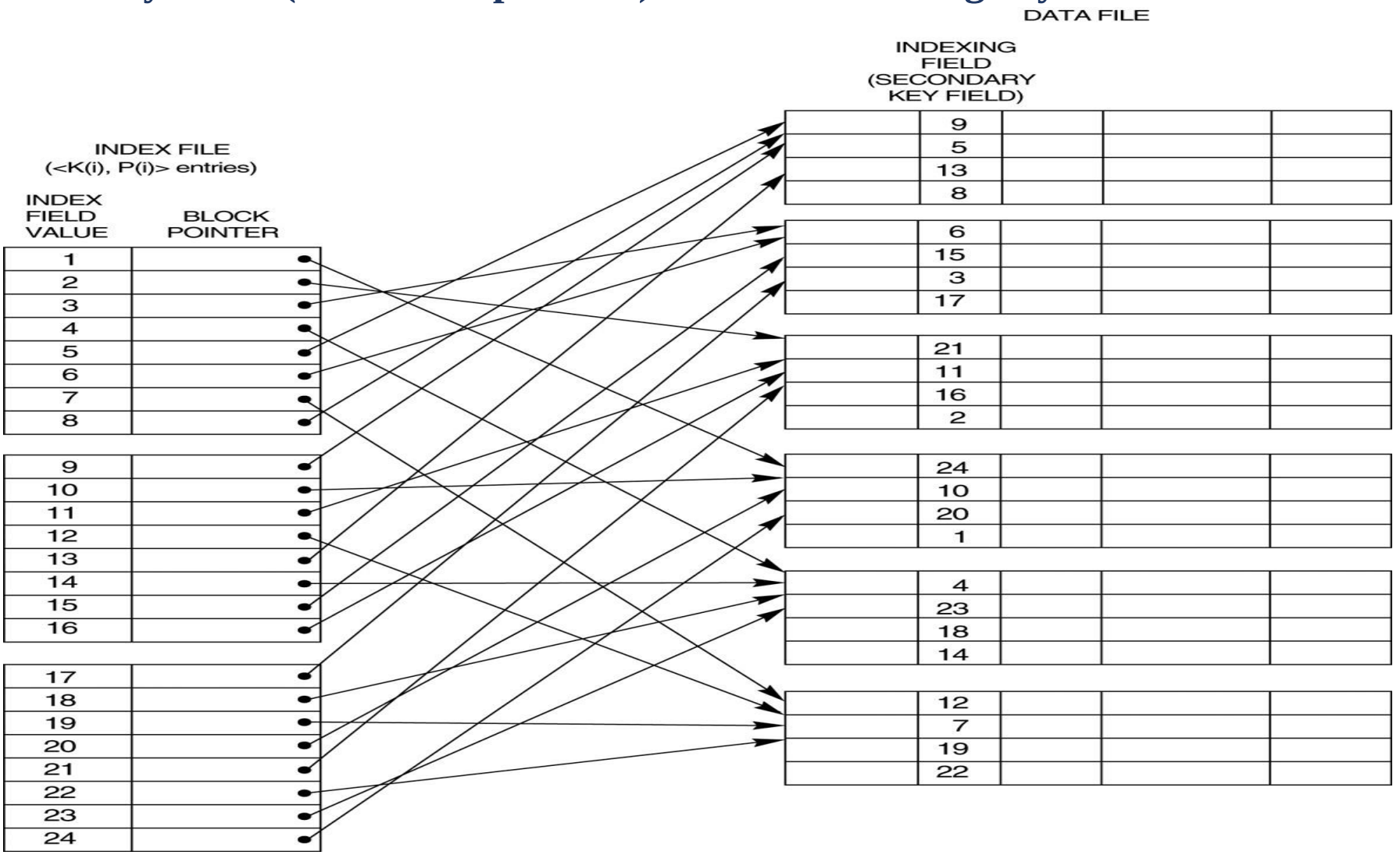
Types of Indexes Cont..

Secondary Index:

- can be created to provide a secondary means of accessing a file for which some primary access already exists.
- defined on a non-ordering field of the data field which is a
  - candidate key and has a unique value in every record, OR
  - non-key with duplicate values.
- Includes one index entry *for each record* in the data file
- Usually needs more storage space & longer search time than primary index because of its larger number of entries

FIGURE 14.4

A secondary index (with block pointers) on a non-ordering key field of a file.



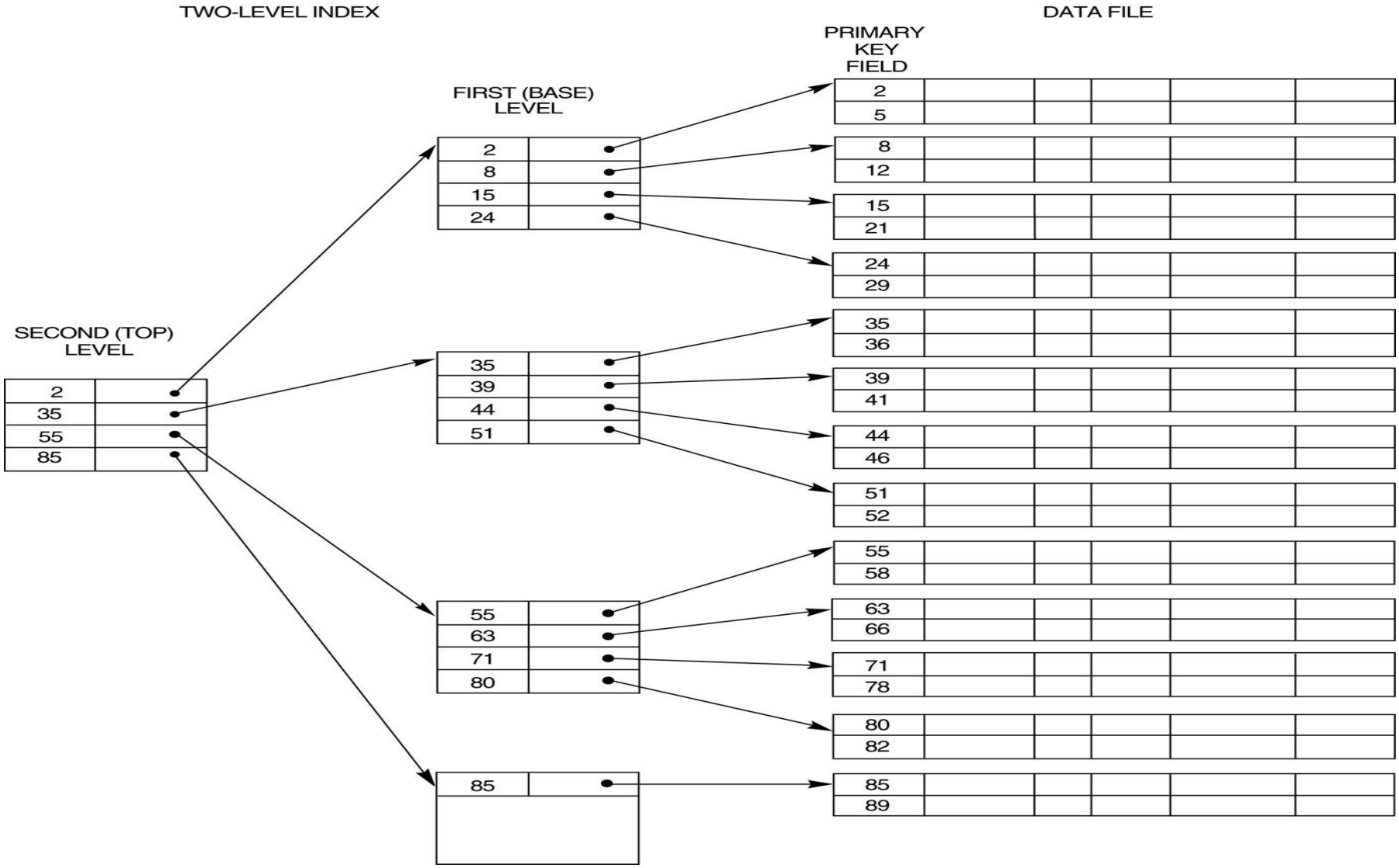
Multi-Level Indexes

- Because a single-level index is an ordered file, we can create a primary index *to the index itself* ; in this case, the original index file is called the *first-level index* and the index to the index is called the *second-level index.*

- We can repeat the process, creating a third, fourth, ..., top level until all entries of the *top level* fit in one disk block

- A multi-level index can be created for any type of first level index (primary, secondary, clustering) as long as the first-level index consists of *more than one* disk block

FIGURE 14.6: A two-level primary index resembling ISAM (Indexed Sequential Access Method) organization.



Multi-Level Indexes Cont…

- Such a multi-level index is a form of *search tree* and reduces the number of blocks accessed when searching for a record

- However, insertion and deletion of new index entries is a severe problem because every level of the index is an *ordered file*.

- To reduce these problems, dynamic multilevel index can be created by using **B-trees** data structure.