

Chapter 18

Indexing Structures for Files

References/Resources:

R. Elmasri, S.B. Navathe, “Fundamentals of Database Systems”, 6th Edition,
Pearson Education

In this chapter we will study how company database more specifically how tables(file- Employee,Department...) is stored in disk (Disk storage and file organization you have already studied in operating system).

Basic terminology used in this chapter

- ❖ The file (Employee, Department...) is referred as data file.
- ❖ Each tuple of data file is referred as data record/record.
- ❖ Three or four (depend upon type of index) data records together called as disk block.
- ❖ There is another file referred as Index file with two columns; index field and pointer.
- ❖ Each index file tuple is referred as record.

Eg: Book index at the end(alphabetical order), to find a particular word you search in the last pages first, then see the page no. associated with it and then go to the specified page.

In this case last pages are index file,

particular word is index field,

associated page no. is pointer

specified page that contain the word is data file.

Index access structure

- It is defined on a single field of a file, called an indexing field (or indexing attribute).
- The index stores each value of the index field along with a list of pointers to all disk blocks that contain records with that field value.
- The values in the index are ordered so that we can do a binary search on the index.

Index file

First col
Index
field

Second
col
pointers

Actual File eg. Employee file on disk blocks

NOTE: In file organization, field is same as attribute.

Index-Characterized

```
graph TD; A[Index-Characterized] --> B[Dense Index]; A --> C[Sparse Index/  
Nondense Index];
```

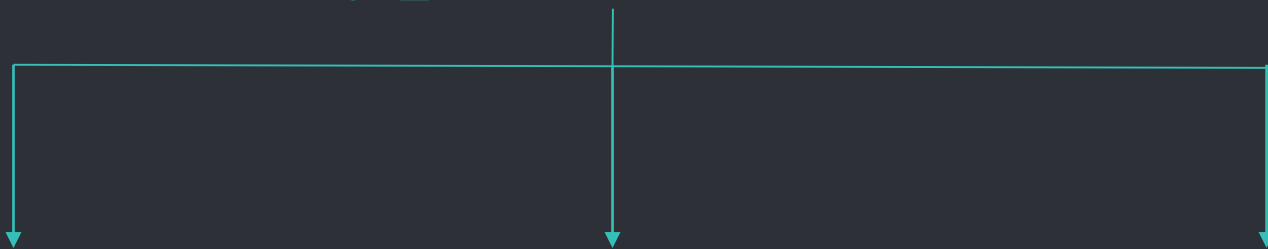
Dense Index

It has an index entry for every search key value (and hence every record) in the data file.

Sparse Index/ Nondense Index

It has index entries for only some of the search values. A sparse index has fewer entries than the number of records in the file.

Types of Indexes



Primary Index

NOTE: Please see the eg on slide no.7, then read for defination.

Clustering Index

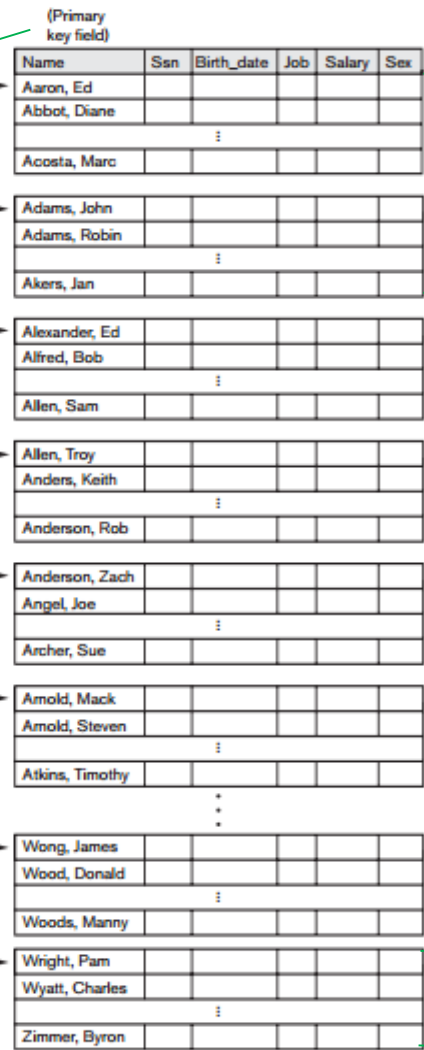
NOTE: Please see the eg on slide no.13, then read the defination.

Secondary Index

• Primary Indexes

- It is an ordered file whose records are of fixed length with two fields.
- The first field is of the same data type as the ordering key(alphabetical order) field of the data file(PK of data file) and the second field is a pointer to a disk block (a block address).
- Each index entry has the value of the primary key field for the *first* record in a block and a pointer to that block, The two field values of index entry i as $\langle K(i), P(i) \rangle$.
- The total number of entries in the index = number of disk blocks.
- The first record in each block of the data file is called the anchor record of the block or the block anchor.
- A primary index is a nondense (sparse) index, since it includes an entry for each disk block of the data file and the keys of its anchor record rather than for every record.

Figure 18.1
 Primary index on the ordering key field of
 the file shown in Figure 17.7.

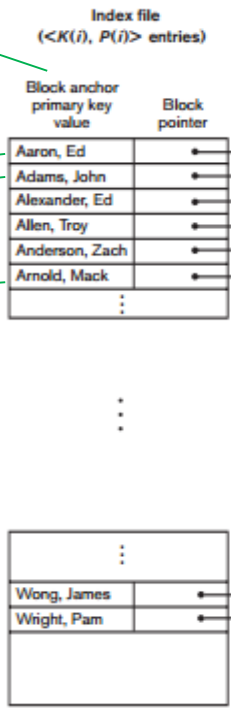


Block 1

Block 2

Block n

First record in each
 block of data file is
 called anchor
 record/block
 anchor. Eg: (Aaron,
 Ed), (Adams, John),
 (Alexander, Ed),
 (Allen, Troy).....



Same data type

Entry in
 index
 field is
 the first
 record in
 block

NOTE:
 Based on
 key value
 (hence
 unique) &
 data file is
 ordered.

Advantages of Primary Indexes

- The index file for a primary index occupies less space as compare to data file because of two reasons:
 - there are fewer index entries than there are records in the data file.
 - each index entry is smaller in size than a data record.
- A binary search on the index file requires fewer block accesses than a binary search on data file.

To access the block containing the record will require $\log_2 b_i + 1$ access

Where b_i is the no. of blocks, +1 is the access to the block containing the record.

Blocking factor (bfr): The no. of records in a block.

$$\text{bfr} = \lfloor B/R \rfloor = \left\lfloor \frac{\text{Block length}}{\text{length of each record contained in the block}} \right\rfloor$$

Disadvantage of Primary Index

Insertion and deletion of records are problematic, if we attempt to insert a record in its correct position in the data file, we must not only move records to make space for the new record but also change some index entries, since moving records will change the anchor records of some blocks.

• Numerical of Primary Index

Q. An ordered file with $r = 30,000$ records stored on a disk block size $B = 1024$ bytes. File records are of fixed size and with record length $R = 100$ bytes.

(i). Find the blocking factor

(ii). The no. of blocks needed for the file.

(iii). How many block access are needed by a binary search on this data file.

Suppose that a primary index is created on the data file with key field of size 9 bytes and block pointer's size 6 bytes.

(iv). Find the blocking factor.

(v). How many block access are needed by a binary search on this index file.

(vi). How many block access are needed by a binary search to access the record of data file.

Solution: $r = 30,000$, $B = 1024$ bytes, $R = 100$ bytes

(i). $bfr = \lfloor B/R \rfloor = \lfloor 1024/100 \rfloor = 10$

(ii). $b = \left\lceil \frac{r}{bfr} \right\rceil = \left\lceil \frac{30000}{10} \right\rceil = 3000$ blocks

(iii). No. of block access $\lceil \log_2 b \rceil = \lceil (\log_2 3000) \rceil = 12$ block accesses

(iv). $V = 9$ bytes, $P = 6$ bytes, the size of each index entry $R_i = (9+6) = 15$ bytes

$bfr_i = \lfloor B/R_i \rfloor = \lfloor 1024/15 \rfloor = 68$ entries per block

(v). $b_i = \left\lceil \frac{r_i}{bfr_i} \right\rceil = \left\lceil \frac{30000}{68} \right\rceil = 45$ blocks.

No. of block access $\lceil \log_2 b_i \rceil = \lceil (\log_2 45) \rceil = 6$ block accesses

(vi). One additional block access to the data file

$\therefore 6+1 = 7$ block accesses

• Clustering Indexes

- The data file often called clustered file in this case, are physically ordered on a nonkey field.
- It is used to speed up retrieval of all the records that have the same value for the clustering field.
- It has two fields; the first field is of the same data type as the clustering field and the second field is a disk block pointer.
- There is one entry in the clustering index for each distinct value of the clustering field, and it contains the value and a pointer to the first block in the data file that has a record with that value for its clustering field.
- Duplicate records in data files are allowed.
- A clustering index is a nondense index because it has an entry for every distinct value of the indexing field, which is a nonkey by definition and hence has duplicate values.

Pointers are at first occurrence of record

Same data type

Non key field, ∴ duplicate

Block 1

Block n

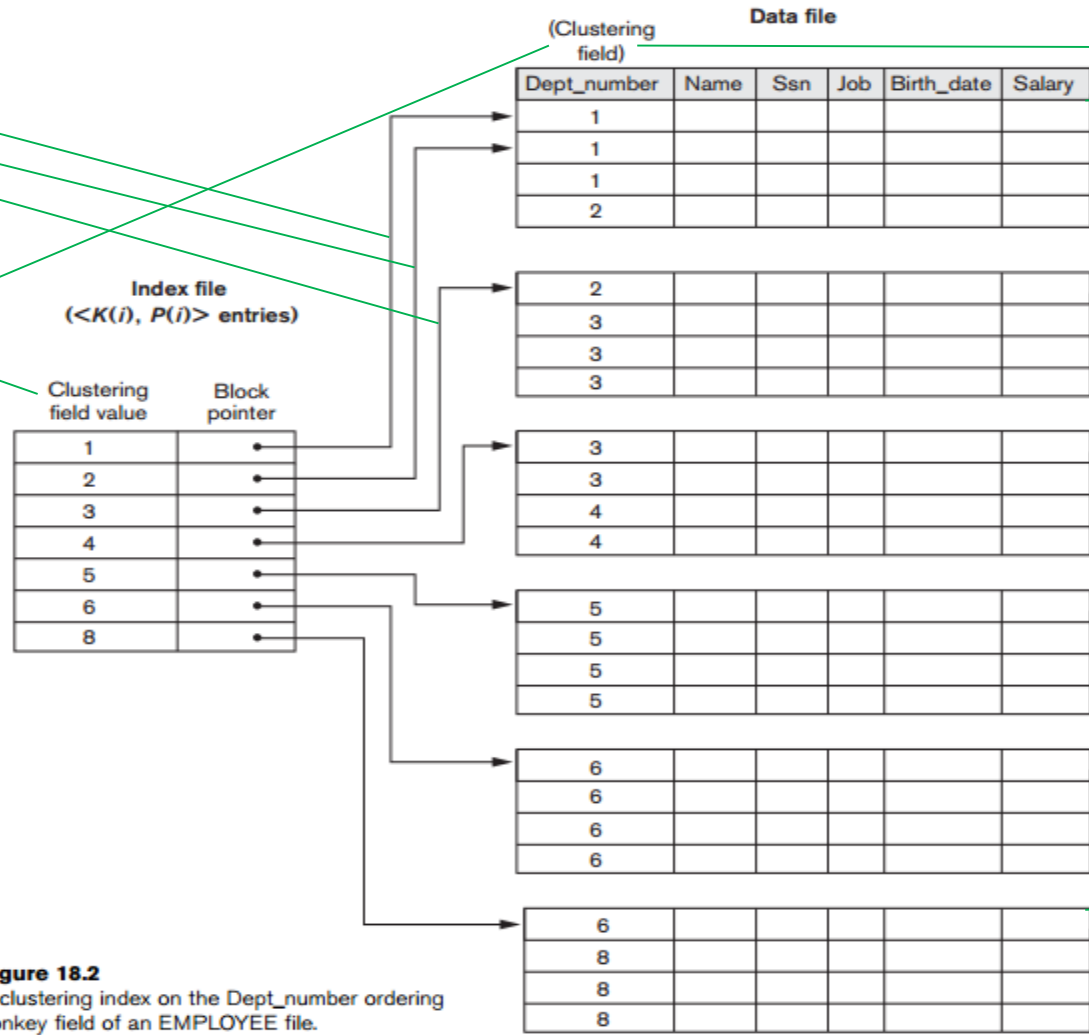


Figure 18.2
A clustering index on the Dept_number ordering nonkey field of an EMPLOYEE file.

NOTE:
Based on non key value (hence duplicate) but data file is ordered.

• Disadvantage of Clustering Index

Record insertion and deletion cause problems because the data records are physically ordered.

Solution: Reserve a whole block for each value of the clustering field; all records with that value are placed in the block. This makes insertion and deletion relatively straightforward. (see next slide for diagrammatic representation)

Figure 18.3
Clustering index with a separate block cluster for each group of records that share the same value for the clustering field.

