


File Organization

1

File Organization


- The database is stored as a collection of *files*. Each file is a sequence of *records*. A record is a sequence of *fields*.
- One approach:
 - assume record size is fixed
 - each file has records of one particular type only
 - different files are used for different relations



2

Fixed-Length Records

- Simple approach:
 - Store record i starting from byte $n * (i - 1)$, where n is the size of each record.
 - Record access is simple but records may cross blocks
 - Modification: do not allow records to cross block boundaries



3

Fixed-Length Records


- Deletion of record i : alternatives:
 - move records $i + 1, \dots, n$ to $i, \dots, n - 1$
 - move record n to i
 - do not move records, but link all free records on a *free list*



4

Fixed-Length Records


record 0	10101	Srinivasan	Comp. Sci.	65000
record 1	12121	Wu	Finance	90000
record 2	15151	Mozart	Music	40000
record 3	22222	Einstein	Physics	95000
record 4	32343	El Said	History	60000
record 5	33456	Gold	Physics	87000
record 6	45565	Katz	Comp. Sci.	75000
record 7	58583	Califieri	History	62000
record 8	76543	Singh	Finance	80000
record 9	76766	Crick	Biology	72000
record 10	83821	Brandt	Comp. Sci.	92000
record 11	98345	Kim	Elec. Eng.	80000



5

Deleting record 3 and compacting

record 0	10101	Srinivasan	Comp. Sci.	65000
record 1	12121	Wu	Finance	90000
record 2	15151	Mozart	Music	40000
record 4	32343	El Said	History	60000
record 5	33456	Gold	Physics	87000
record 6	45565	Katz	Comp. Sci.	75000
record 7	58583	Califieri	History	62000
record 8	76543	Singh	Finance	80000
record 9	76766	Crick	Biology	72000
record 10	83821	Brandt	Comp. Sci.	92000
record 11	98345	Kim	Elec. Eng.	80000



6

Deleting record 3 and moving last record

record 0	10101	Srinivasan	Comp. Sci.	65000
record 1	12121	Wu	Finance	90000
record 2	15151	Mozart	Music	40000
record 11	98345	Kim	Elec. Eng.	80000
record 4	32343	El Said	History	60000
record 5	33456	Gold	Physics	87000
record 6	45565	Katz	Comp. Sci.	75000
record 7	58583	Califieri	History	62000
record 8	76543	Singh	Finance	80000
record 9	76766	Crick	Biology	72000
record 10	83821	Brandt	Comp. Sci.	92000

7

Free Lists

- Store the address of the first deleted record in the file header.
- Use this first record to store the address of the second deleted record, and so on
- Can think of these stored addresses as **pointers** since they "point" to the location of a record.

8

Free Lists

- More space efficient representation: reuse space for normal attributes of free records to store pointers. (No pointers stored in in-use records.)

9

Free Lists

header				
record 0	10101	Srinivasan	Comp. Sci.	65000
record 1				
record 2	15151	Mozart	Music	40000
record 3	22222	Einstein	Physics	95000
record 4				
record 5	33456	Gold	Physics	87000
record 6				
record 7	58583	Califieri	History	62000
record 8	76543	Singh	Finance	80000
record 9	76766	Crick	Biology	72000
record 10	83821	Brandt	Comp. Sci.	92000
record 11	98345	Kim	Elec. Eng.	80000

10

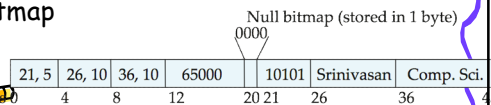
Variable-Length Records

- Variable-length records arise in database systems in several ways:
 - Storage of multiple record types in a file.
 - Record types that allow variable lengths for one or more fields such as strings (**varchar**)
 - Record types that allow repeating fields (used in some older data models).

11

Variable-Length Records

- Attributes are stored in order
- Variable length attributes represented by fixed size (offset, length), with actual data stored after all fixed length attributes
- Null values represented by null-value bitmap



12

Variable-Length Records: Slotted Page Structure

- **Slotted page** header contains:
 - number of record entries
 - end of free space in the block
 - location and size of each record

Variable-Length Records: Slotted Page Structure

- Records can be moved around within a page to keep them contiguous with no empty space between them; entry in the header must be updated.
- Pointers should not point directly to record — instead they should point to the entry for the record in header.

Data Structures vs File Structures

- Both involve:
 - Representation of Data + Operations for accessing data
- Difference:
 - Data structures: deal with data in the main memory
 - File structures: deal with the data in the secondary storage

File Structure in Computer

Data Storage in Computer

- type: Semiconductors
- Properties: Fast, expensive, volatile, small
- type: disks, tapes
- properties: Slow, cheap, stable, large

File and File Operations

- A file is a collection of data stored on mass storage like hard disk, CD etc.
- File data consist of records (student information) and each record contains number of fields (ID, Name etc.).

File and File Operations

- Operations on files can be viewed as either a **retrieval operation**, where the record is selected from the file according to specific **search conditions** but not changed, or
- an **update operation** which also involves a search and changes the file.
- The search may be for a specific record, a group of records or a specific location in the file.

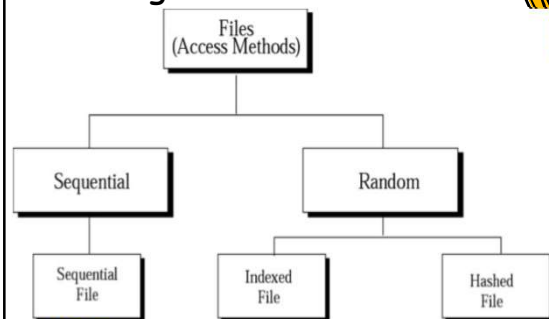
19

File Operations

- Remove / Update a certain item.
- Order the data items according to a certain criterion,
- merge of files.
- Creation of new files from existing files.
- create, open, and close operations

20

File Organization and Access



21

File Organization Method

- The process that involves how data/information is stored so file access could be as easy and quickly as possible.
- Three main ways of file organization:
 - Sequential
 - Index-Sequential
 - Random

22

Sequential file organization

- All records are stored in some sort of order (ascending, descending, alphabetical).
- The order is based on a field in the record.
 - For example a file holding the records of employeeID, date of birth and address.
 - The employee ID is used as the key and records are stored in groups accordingly

23

Sequential File Organization

- Records are conceptually organized in a sequential list.
- The actual storage might or might not be sequential (on tape or on disk)

24

Sequential File Organisation

File Header
Record 1
Record 3
Record 4
Record 2
Record N

Sequential File Organisation

25

Sequential File Organisation

- In Ordered sequential files, it is easy to locate and read from the file in order of key value.
- A search is on key values is efficient and relatively fast if a binary search algorithm is used.

26

Sequential File Organisation

- inserting and deleting from ordered sequential files requires several manipulations and is not very efficient.
- Sequential files can also be unordered.
- These are called Heap files. Unordered files are very easy to insert new records.

27

Sequential File Organisation

- The new record is added to the end of the file.
- Locating a record however, requires a linear search which is not efficient particularly with large files.

28

Advantages of Seq

- Simple file design
- Very efficient when most of the records must be processed e.g. Payroll
- Very efficient if the data has a natural order
- Can be stored on inexpensive devices like magnetic tape.

29


Disadvantages of Seg

- Entire file must be processed even if a single record is to be searched.
- Transactions have to be sorted before processing
- Overall processing is slow.

30

Index-Sequential organization


- The records are stored in some order but there is a second file called the index-file that indicates where exactly certain key points are.
- Can not be used with sequential access method.



31

Index-Sequential organization


- In this type of file organisation an index table or file is created.
- The index file contains key value(s) that can be matched with key values in one or more records.
- The index also contains the disk address of the record.



32

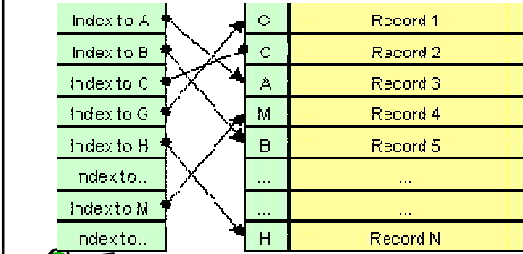
Index-Sequential organization

- To locate a record, the index is accessed first using some search algorithm.
- When the key value is found a pointer in the index file contains the address of the matching record(s).
- New records can be easily added to or deleted from the record file.




33

Index-Sequential organization



The diagram illustrates an indexed file organization. On the left, an index table lists keys: 'Index to A', 'Index to B', 'Index to C', 'Index to G', 'Index to H', 'index to ..', 'Index to M', and 'index to ..'. On the right, a record table lists records: 'Record 1', 'Record 2', 'Record 3', 'Record 4', 'Record 5', '...', '...', and 'Record N'. Arrows show the mapping: 'Index to A' points to 'Record 1'; 'Index to B' points to 'Record 2'; 'Index to C' points to 'Record 3'; 'Index to G' points to 'Record 4'; 'Index to H' points to 'Record 5'; 'Index to M' points to 'Record N'. Ellipses in the index table indicate other keys and their corresponding records.


Indexed File Organisation



34

Advantages


- Provides flexibility for users who need both type of accesses with the same file.
- Faster than sequential.



35

Disadvantages

- Extra storage space for the index is required



36

Random file organization

- The records are stored randomly but each record has its own specific position on the disk (address).
- With this method no time could be wasted searching for a file.
- Instead it jumps to the exact position and access the data/information.



37

Advantages of Random

- Records are quickly accessed (i.e. there is fast access to records).
- Files are easily updated (i.e. adding, deleting, and amending the records is easily achieved).
- The method does not require the use of indexes, hence saving space.



38

Advantages of Random

- Transactions do not need to be sorted before being updated.
- New records can be easily inserted into a random file



39

DisAdvantages of Random

- Data may be accidentally erased or overwritten unless special precautions are taken.
- Random files are less efficient in the use of storage space compared to sequentially organized files.



40

DisAdvantages of Random

- Expensive hardware and software resources are required.
- Relatively complex when programming.
- System design based on random file organization is complex and costly.



41

File Access

- The way by which information/data can be retrieved.
- There are three methods of file access:
 - Direct Access
 - Sequential Access
 - Index Access



42

Direct Access

- the information/data stored on a device can be accessed randomly and immediately irrespective to the order it was stored.
- The data with this access method is quicker than sequential access.
- This is also known as random access method.
 - For example Hard disk, Flash Memory

43

Sequential Access

- The information/data stored on a device is accessed in the exact order in which it was stored.
- Sequential access methods are seen in older storage devices such as magnetic tape.

44

Index Access:

- In this method an index is created which contains a key field and pointers to the various block.
- To find an entry in the file for a key value, we first search the index and then use the pointer to directly access a file and find the desired entry.

45

Index Access

- With large files, the index file itself may become too large to be kept in memory.
- One solution is to create an index for the index file.
- The primary index file would contain pointers to secondary index files, which would point to the actual data items.

46

Questions



47