# Fixed-Length Records File Organization

Those Records which has Fixed size like let us Consider a file of 'Account records' for our bank DataBase

↓Table Name

```
type deposit = record
                 account-number : char(10);
                 Branch-name : char(22);
                 Balance: real;
               end
```

Attributes

let assume 1 Char = 1 byte

1 Real = 8 bytes

So our One Records becomes

Account_Numb Char 10 × 1 = 10 byte

Branch-Name Char 22 × 1 = 22 byte

Balance : Real 8 × 1 = 8 bytes

40 bytes

| | | |
|---|---|---|
| A-101 | Punjabi Bagh | 1800 |
| A-102 | Sadar Bazar | 7800 |
| A-103 | Pharganj | 8100 |
| A-104 | Rani Bagh. | 6900 |
| A-105 | Rohini East | 8102 |



Then we can follow an approach.

→ use a first 40 bytes for the first Record

→ the next 40 bytes for the second Record.

But there are two problems with this Simple approach.

① Deleting the Record from this will create an empty block which may occur waste of Memory space

So to over come this Problem we must fill this block with

record of the file, Or we must have Way of marking Deleted Records so that can be ignored.

→ If block size is less than 40 bytes.

So we have to store Records in 2 or more block according the blocks size. But for this kind of problem Write & Read Operations become tedius.

To Save the space of Deleted Record

One Approach
Move every Record up to the deleted record block.

It takes More time & effosts to move every Record till end of file

| | | |
|---|---|---|
| A-101 | Punjabi Bagh | 1800 |
| A-102 | Sadar Bazar | 7800 |
| A-103 | Phar Ganj | 8100 |
| | | |
| A-105 | Rohini East | 8102 |
| A-106 | Rohini West | 6280 |
| ! | | |
| ! | | |
| ! | | |
| ! | | |
| A-19 | Chandni Chowk | 7940 |
| A-20 | Rani Bagh. | 80000 |

2nd Approach.

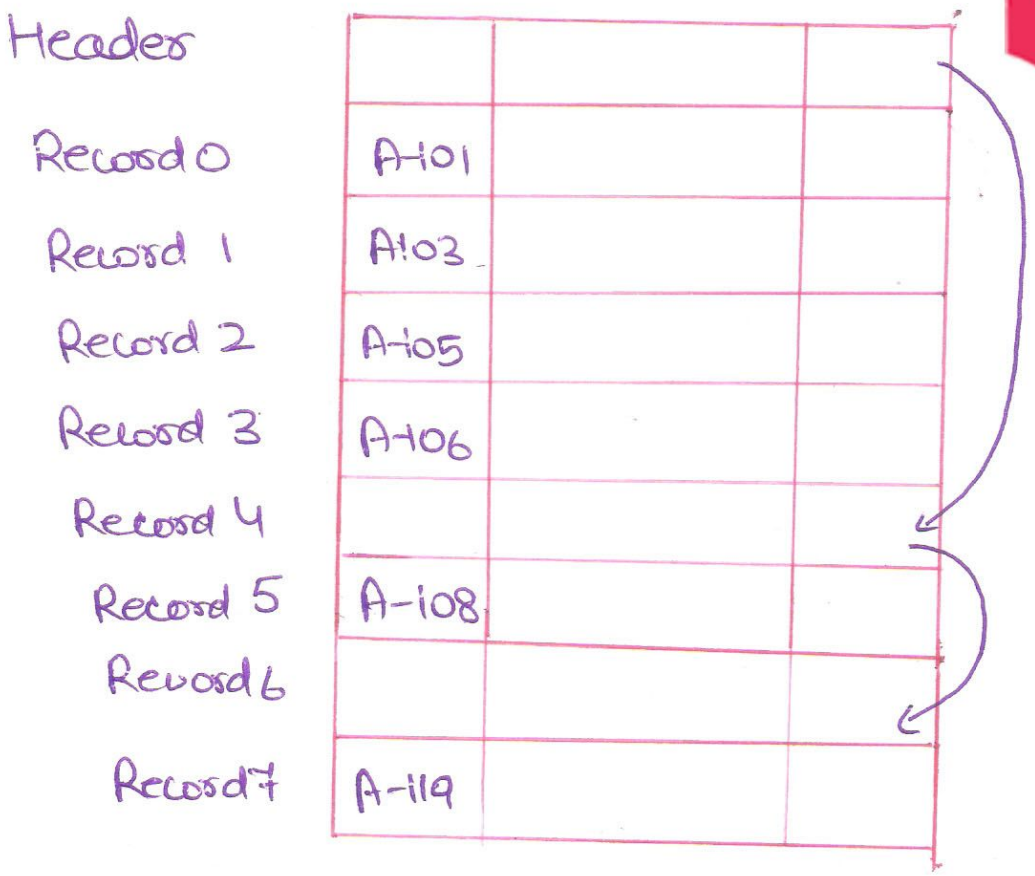
Move the Last Record to the Delcted Record Back.

Need Additional block Accesses.

New Approach :- At the beginning of the file, we allocate a certain number of bytes as a 'File Header'.

This File Header contains a variety of information about the file with the address of the records whose ~~addr~~ Contents are deleted.

We use this first Record to store the address of the second available Record, and so on.




→ We can assume these Stored addresses as pointers as they point to the location of a record.

→ The deleted Records thus form a linked list or a free list.

→ On Insertion of a new Record, we use Record pointed to by the header & we change the header pointer to point to next available record.

→ If no space is available, we add new record to the end of the file.