**Static Header**

|  |  |
| --- | --- |
| Field | Size (Bytes) |
| File Header (“CS”) | 2 |
| Timestamp | 8 |
| #Records ( Logically present) | 4 \*(changed) |
| #Columns | 1 |
| Table Name | 32 |
| Size of Record | 2 |
| Data Head | 4 |
| Data End | 4 |
| First Record (stores first record number) | 4 (Since #records takes 8 bytes) \* |
| Last Record (stores last active record number) | 4 \* |
| Total Record ( #records + free records) | 4 \* |
|  |  |
|  |  |
| Total | 85 |

Fixing some maximum values:  
  
No of Columns : (takes 1 Byte, hence) 255

No of Records : (takes 8 Bytes, hence) 2^(32)-1

Size of Record : (takes 2 Bytes, hence) 65535

Buffer Space :

The Column structure can be like this:

struct Column

{

char col\_name[COLUMN\_NAME\_SIZE];

char data\_type; // this contains both type (1st 3bits) and constraint(2nd 3bits)

char index; // present or not ( may be extended as column ordering)

char size; // length of the data

~~char frac\_part\_size; // nonzero if the data type is real, zero otherwise~~(cancelled)

/\* int auto\_increment; \*/

};

lets assume COLUMN\_NAME\_SIZE = 32, no column cannot be more than 32 characters.

**Note:**

PLEASE MAINTAIN THIS ORDERING OF STRUCTURE SO THAT **fwrite** or **write** function

WRITES YOUR ARRAY OF COLUMNS IN THIS ORDER.

‘size’ is not the data type size, datatype size is fixed as we discussed. It is the length. Like if

for an integer data if length is 5, the max value it can contain is 99999.

size is declared as char for consuming less memory.

Since the length of integer data was restricted to (1^10 – 1), so length is maximum 9, which can be stored in a byte. As well as, string data has maximum length 255, that can also be saved in a byte.

Similarly for fractional part, we allow maximum 6 digit for precision.

Hence, Size of Column = 32 + 1 + 1 + 1 = 35 bytes.

**\*Change**

**For size attribute,**

**integer, date, time : 1st 4 bits ( AND with 240 and 4 right shift to get the size)**

**Real : 1st 4 bits (integral part); last 4 bits( decimal part, AND with 15 )**

**string : total 8 bits**

**Final Metadata:**  
  
Static Part of Header : 85 Bytes

Dynamic Part: (#column \* 36) bytes

DATA\_HEAD = DATA\_HEAD\_SIZE + DATA\_END\_SIZE + FIRST\_RECORD\_SIZE + LAST\_RECORD\_SIZE + TOTAL\_REC\_SIZE + (#column \* 36) + BUFFER SPACE

= 20 + (#column \* 36) + BUFFER SPACE

DATA\_END = DATA\_HEAD + total\_record \* BLOCK\_SIZE

**Record Structure**

Since Each Block stores a previous record number (8 Bytes), next record number (8 Bytes) and a record, so

BLOCK\_SIZE = 8 + 8 + 65535 = 65551

Static Header (85 Bytes)

Block 1

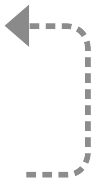
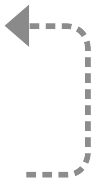
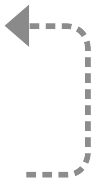
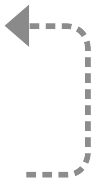
Block 2

Block 3

Dynamic Part (Columns + Buffer)

Next

pointer



Prev

Pointer