Storing Variable length Data/records.

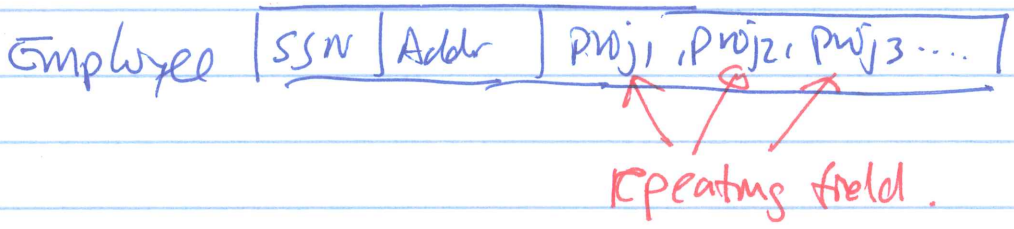
Need for variable size records

(1) Data item has variable size

eg: string.

(2) Repeating field.

(eg: representing a many-many relationship inside a record.



(3) Variable formate records

eg: XML:

```
<Name>
   John Smith
</Name>
   :
```

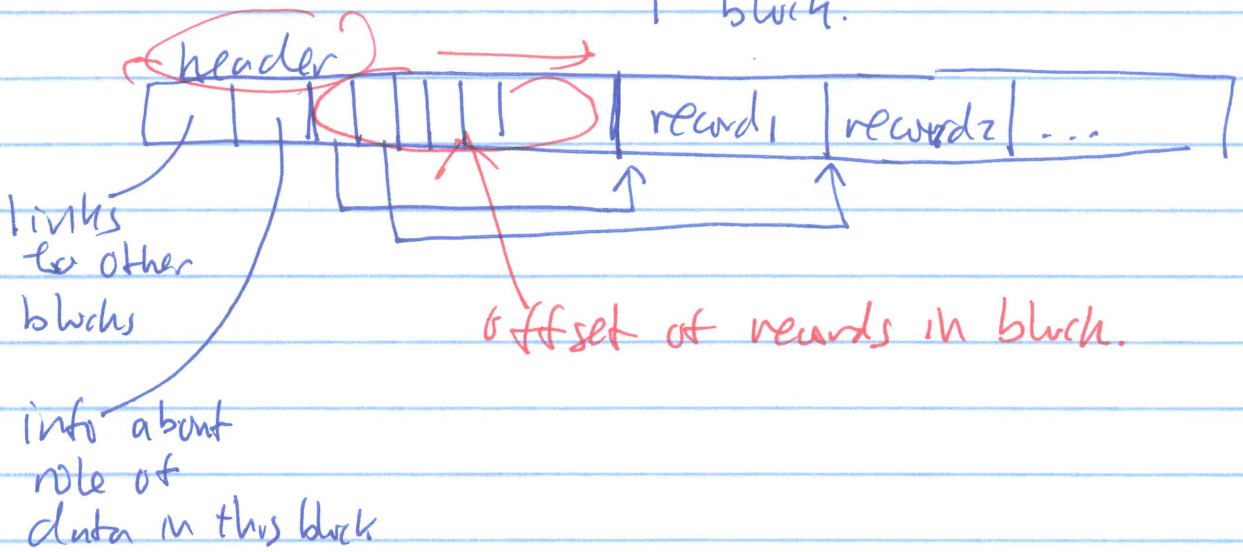
(4) Large size fields

"Binary Large Objects" (BLOB)

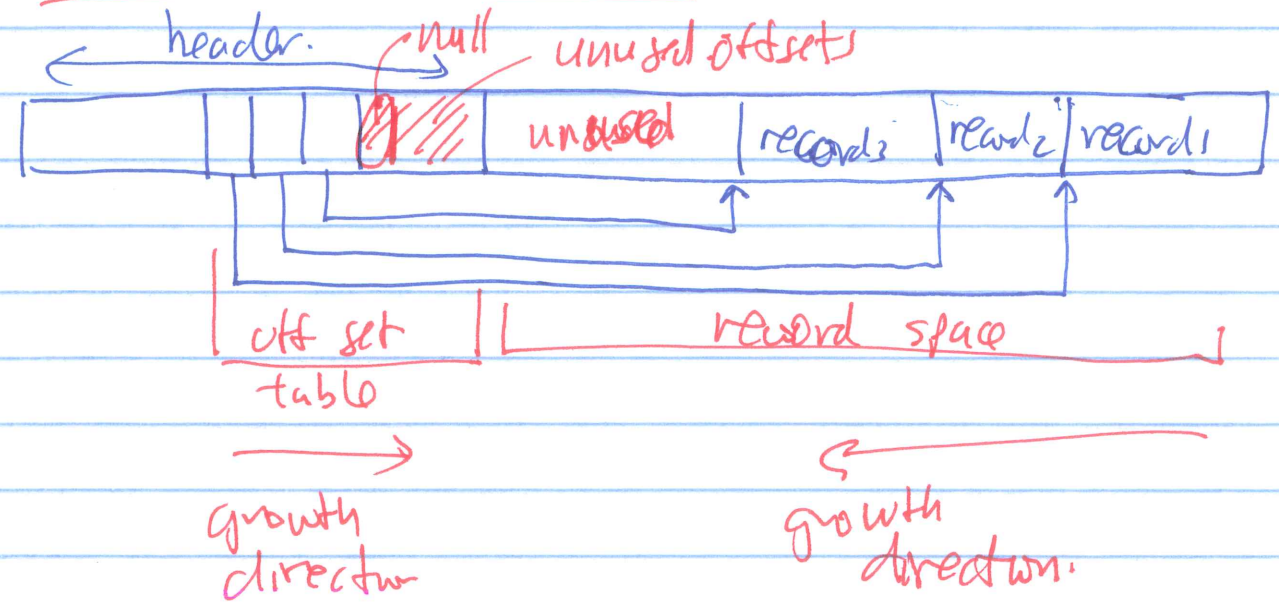
eg: movie! song!

Storing variable sized records in a block:

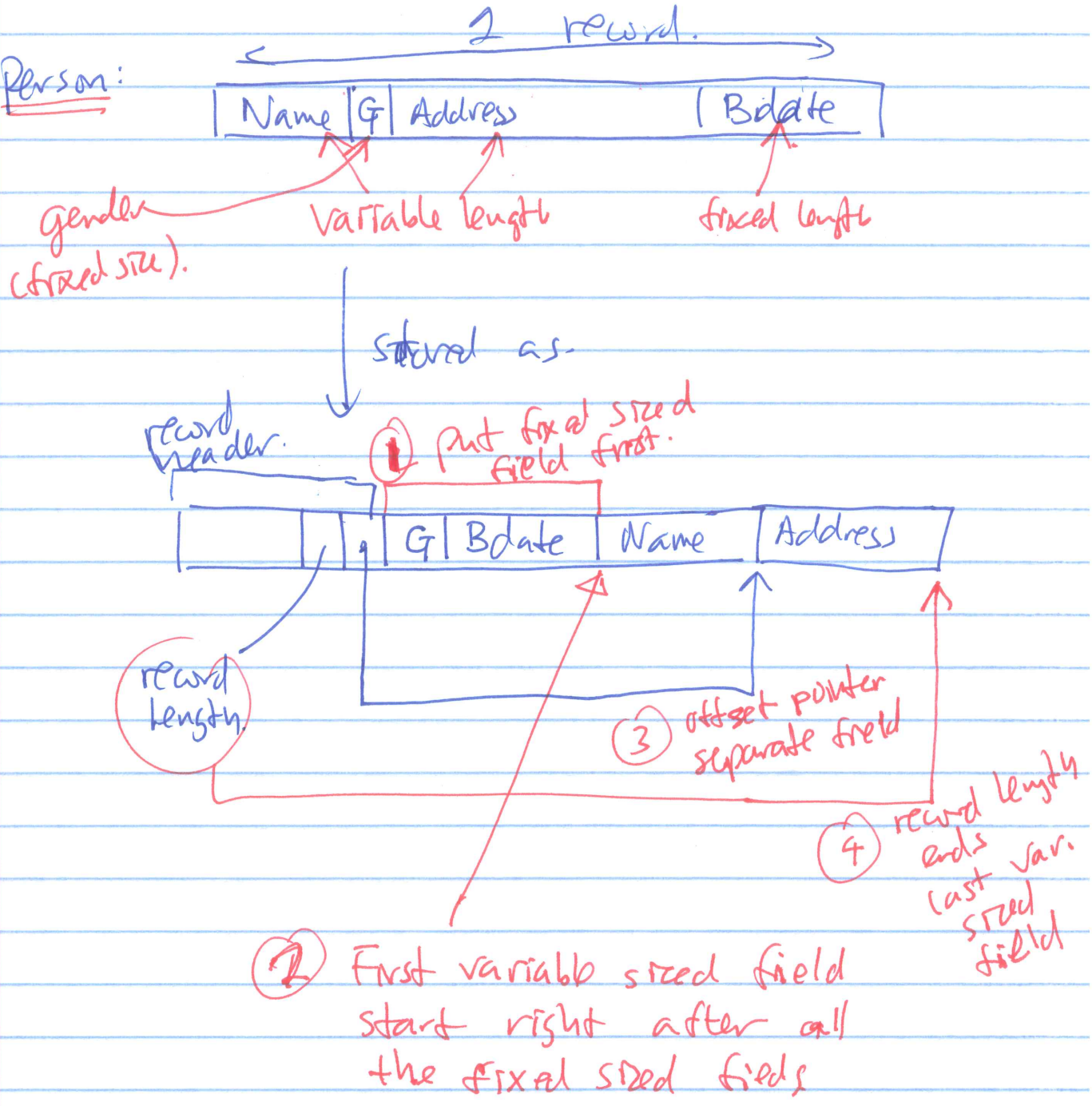
- Recall: Fixed sized records:



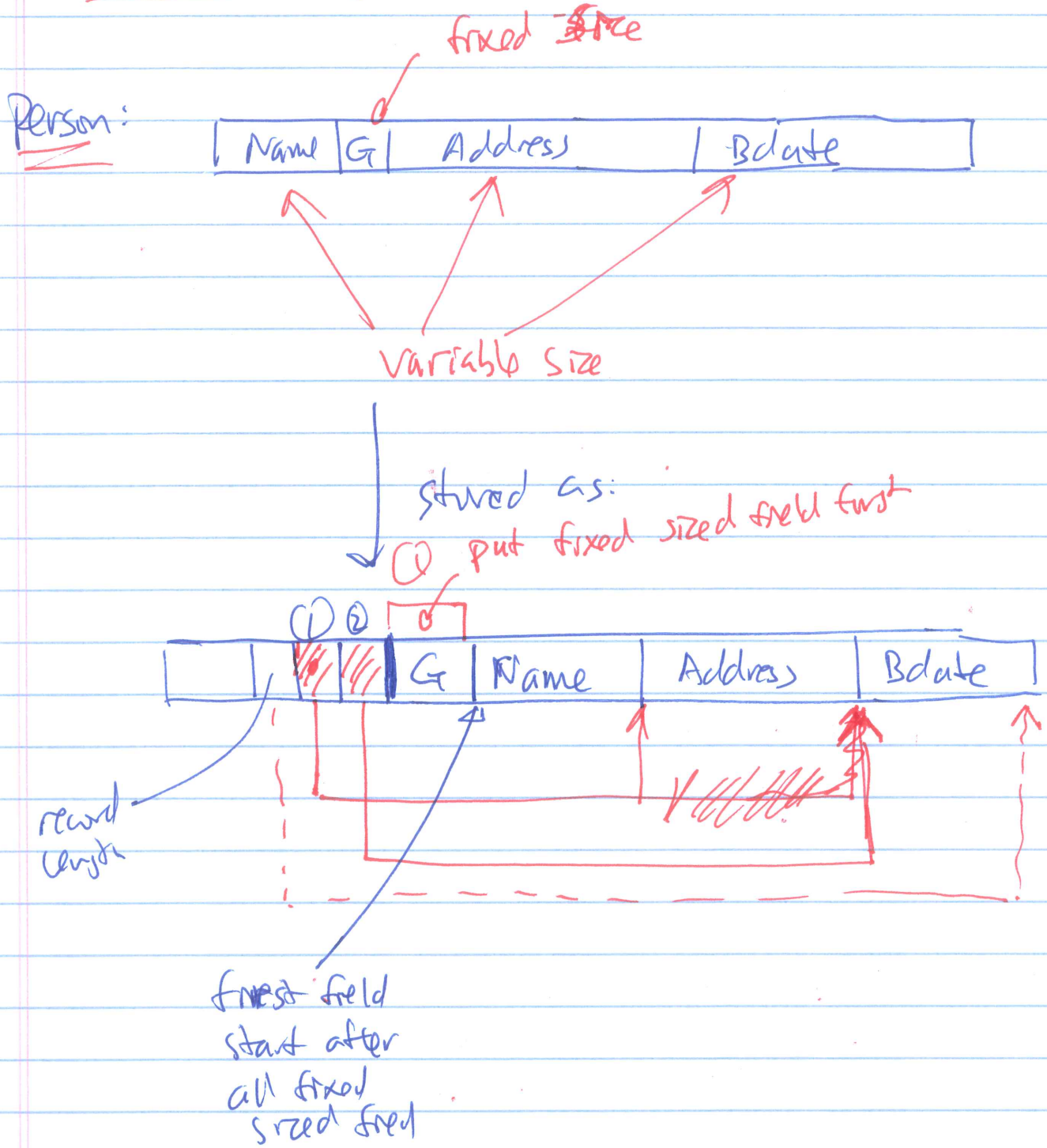
- Block with variable length Records:



Storing a variable sized record itself



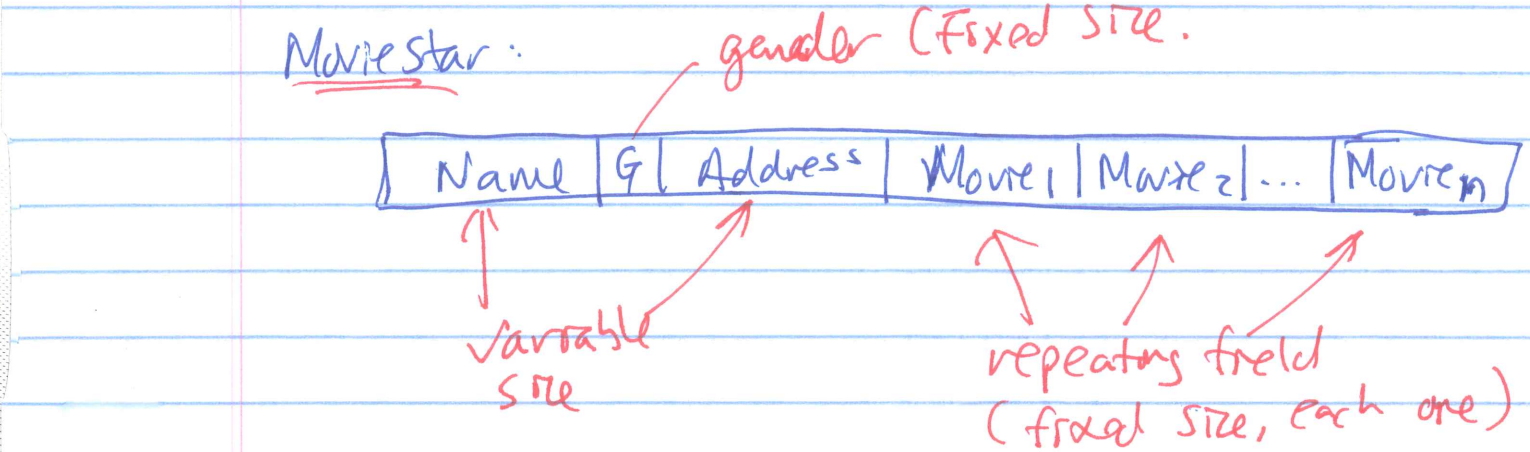
Another example:



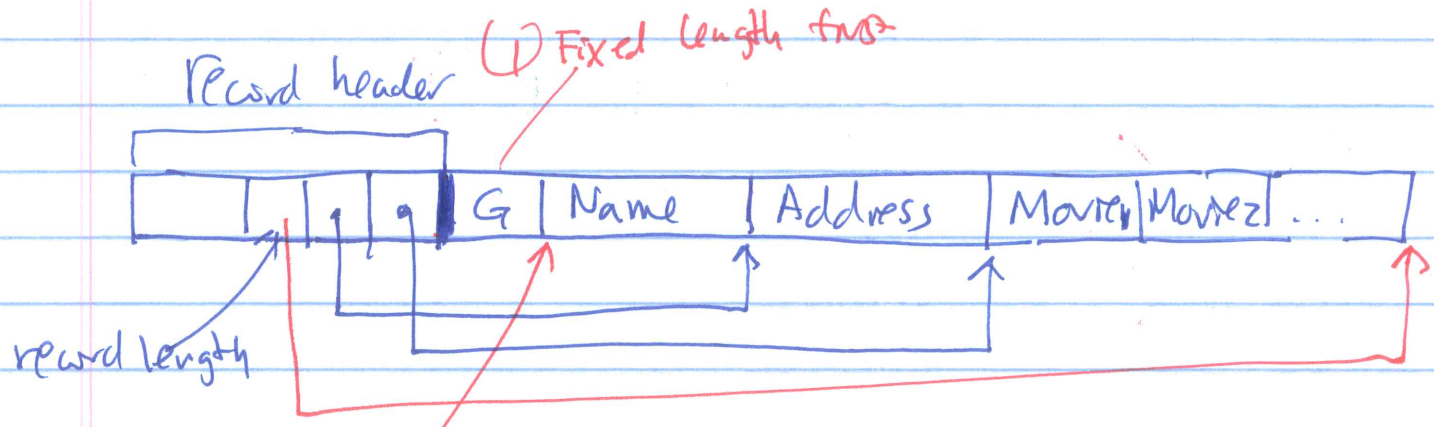
Storing records with repeating fields

Example:

MovieStar:

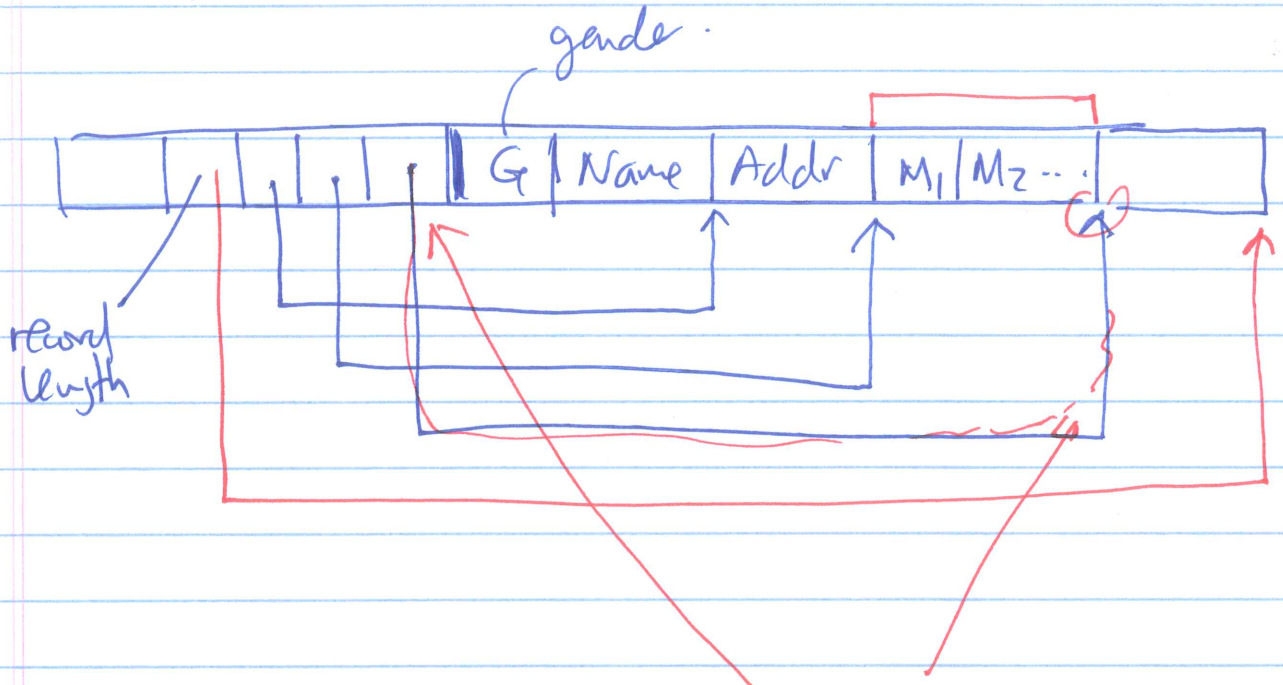


stored as:



First variable length field starts (is known!). Implicit.

Note: If you have MORE variable/repeating field, the END can always be detected by the info. in the record header:



This index/pointer marks the end of a variable / repeating field

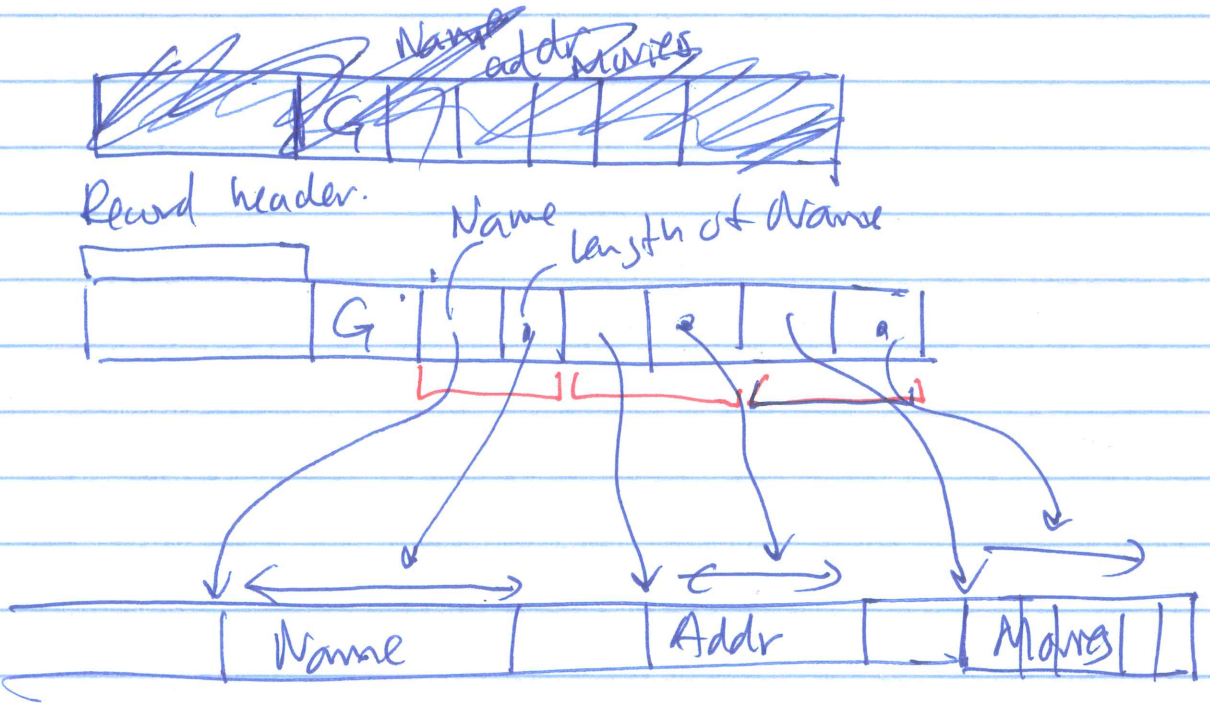
Alternative Storage Structure for variable length and repeating fixed records:

(1) Use 2 separate records.

(A) a fixed length records with data + pointers to variable length fields.

(B) The variable length field(s) are stored separately.

eg:



Advantages:

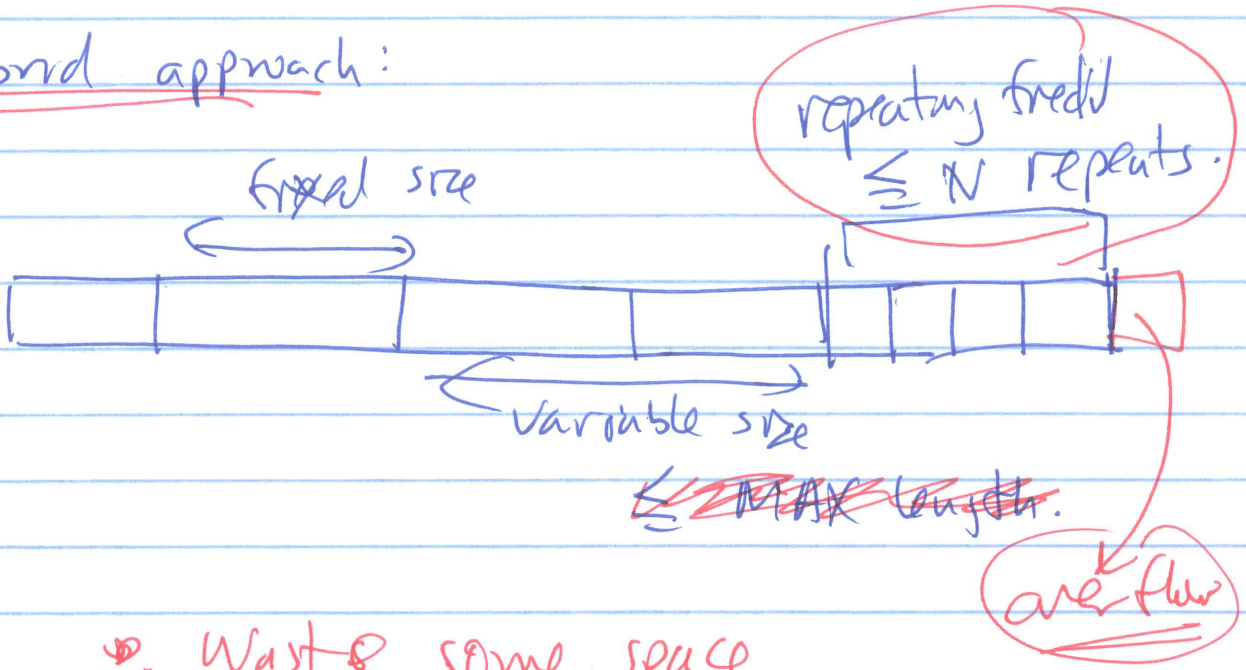
- The record is fixed length

  → allow record to be moved more easily.

Disadvantage:

- More disk access to access Variable length field (Need 2 disk accesses)

Hybrid approach:



- Waste some space
- faster access for small # of repeats.

Storing variable FORMAT records

- When you store a record whose structure is unknown / variable

You must:

(1) Define an ontology.
(Naming scheme)

Eg: N = Name
R = Restaurant owned.

S = String
I = Integer.

(2) Store the meaning of a field (encoded with the ontology)

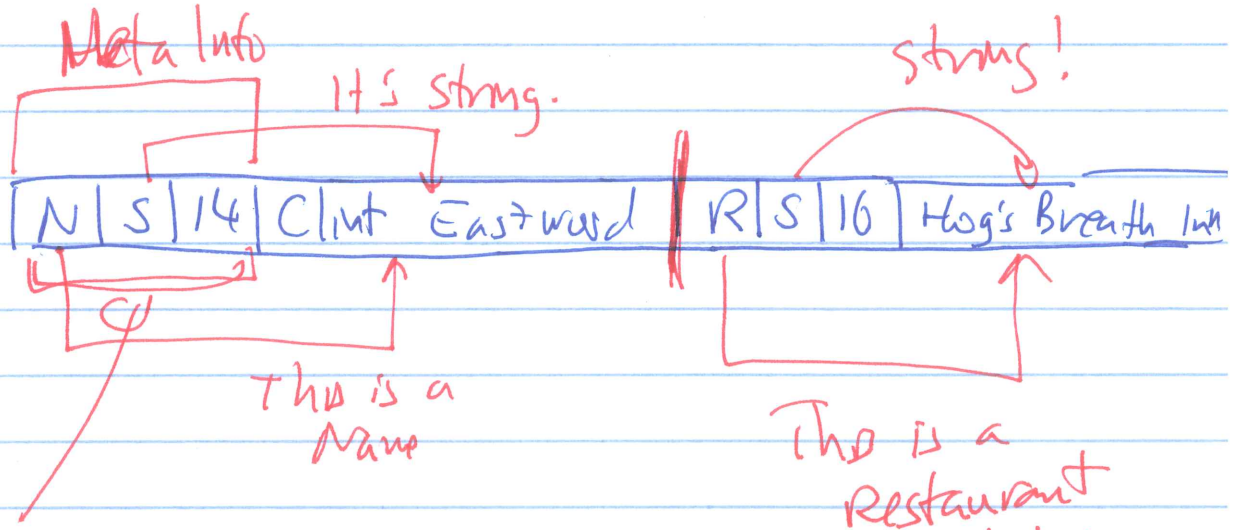
along with the value of the field.

Typical ~~entry~~ format of an entry for ONE field

Meta information

Meaning of the field	DataType used to store values of this field	Length (in # bytes)	Value of the field

Eg:



Assume every field is 1 byte in length.

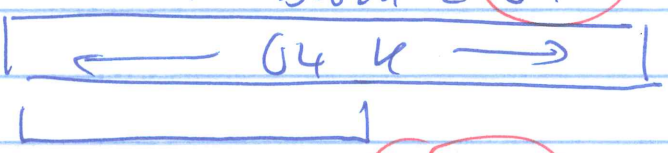
Records that are larger than a block

- Spanning record = a record that is stored inside (> 1) block.

NB: records that are smaller than a block CAN span > 1 block :

eg: To store records efficiently.

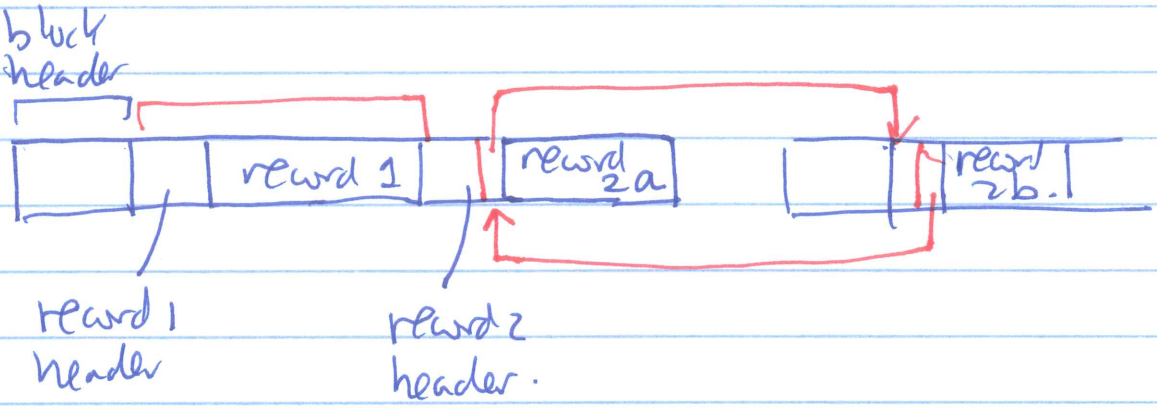
1 block = 64 k



1 record = 33 k.

⇒ Waste too much space if we store 1 record in one block !!!

Storing a record that is spread over 2 blocks:



Record Header stores:

(1) A bit to indicate whether the record is whole / fragmented.

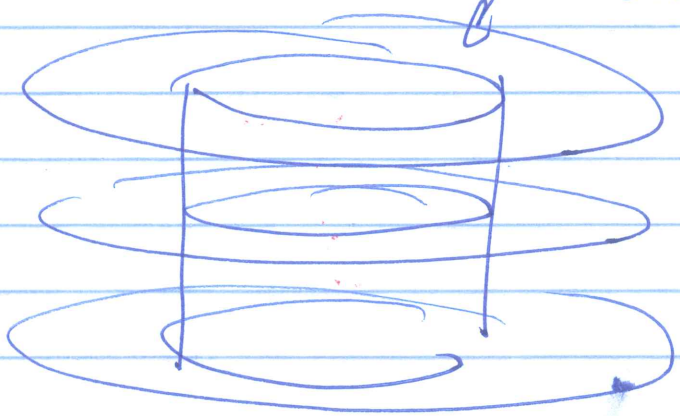
(2) (2 pieces): A bit to tell if fragment is:

first fragment
or Last fragment

(Use a fragment # if more)

(3) Pointers to previous and next fragments. (In each fragment).

Storing BLOBs:



- Retrieve only portion of the BLOB
  (eg: Movie: enough to play
  at the display rate)