Files

- Access method layer offers an abstraction of data on disk: a file of records residing on multiple pages
  - A number of *fields* are organized in a *record*
  - A collection of records are organized in a *page*
  - A collection of pages are organized in a *file*

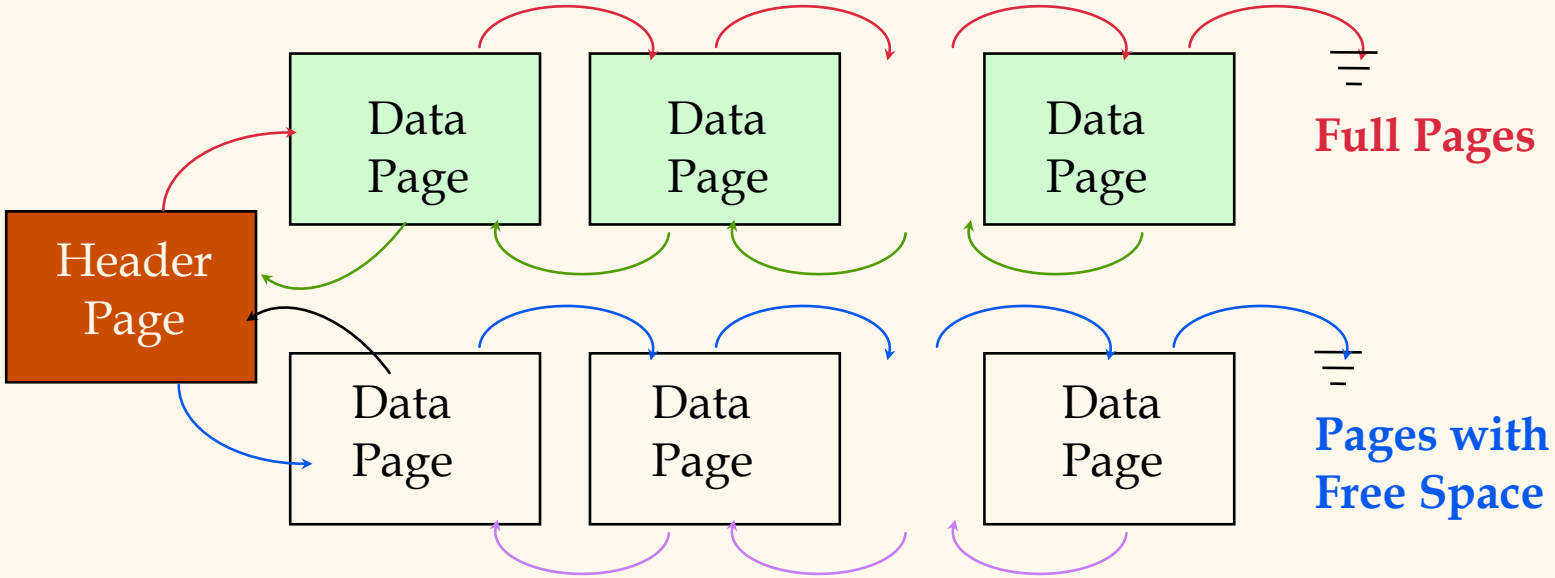
Files of Records

- Page or block is OK when doing I/O, but higher levels of DBMS operate on *records* and *files of records*.
- FILE: A collection of pages, each containing a collection of records. Must support:
  - insert/delete/modify record
  - read a particular record (specified using *record id*)
  - scan all records (possibly with some conditions on the records to be retrieved)

Unordered (Heap) Files

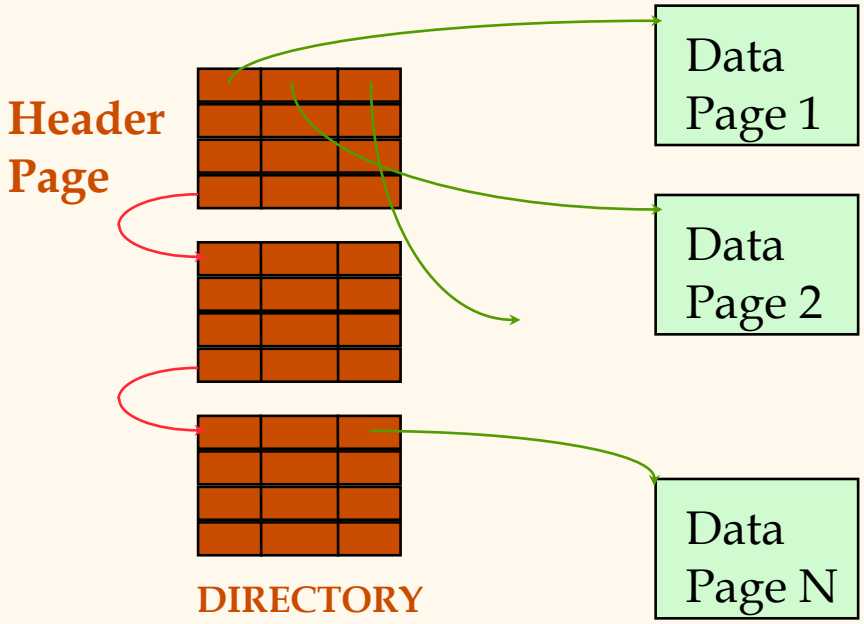
- Simplest file structure contains records in no particular order.
- As file grows and shrinks, disk pages are allocated and de-allocated.
- To support record level operations, we must:
  - keep track of the *pages* in a file
  - keep track of *free space* on pages
  - keep track of the *records* on a page
- There are many alternatives for keeping track of this.

Heap File Implemented as a List



- (heap file name, header page id) stored in a known place.
- Two doubly linked lists, for full pages & pages with space.
  - Each page contains 2 `pointers' plus data.
- Upon insertion, scan the list of pages with space, or ask disk space manager to allocate a new page

Heap File Using a Page Directory

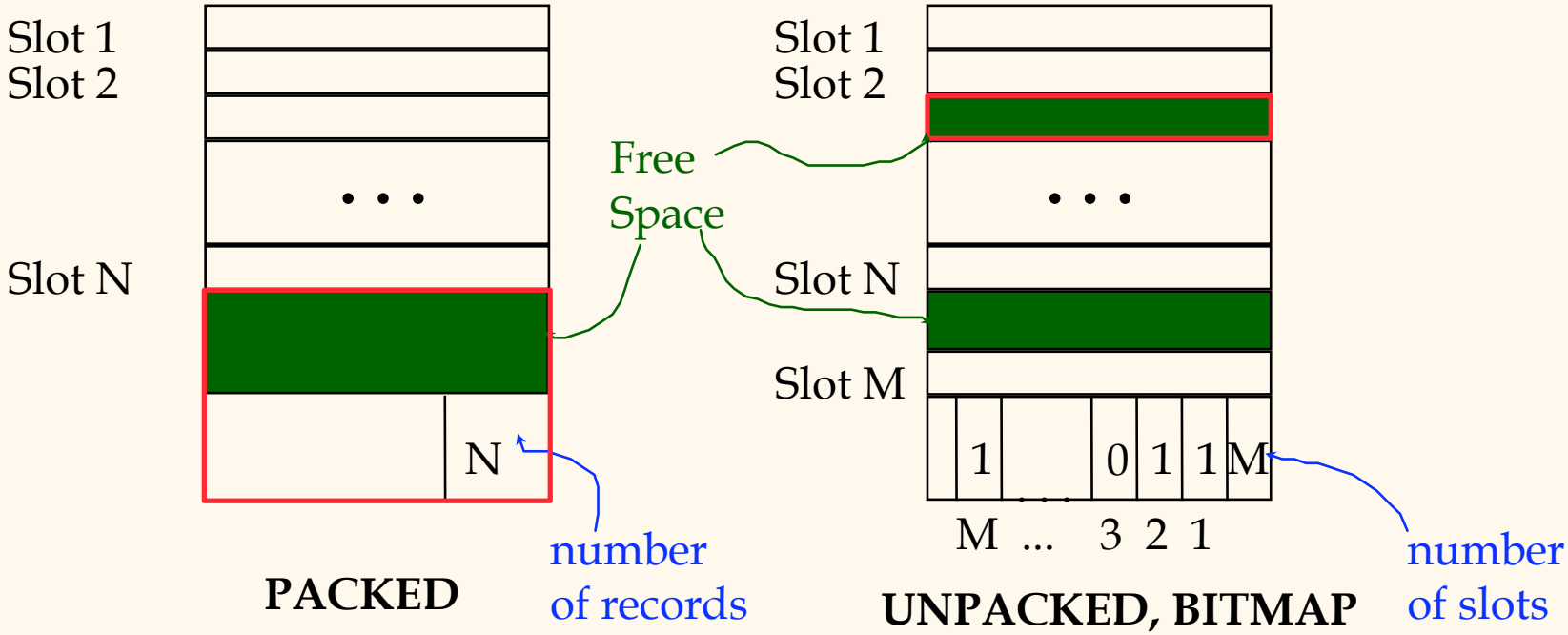


- A directory entry per page; it can include the number of free bytes on the page.
- The directory is a collection of pages; linked list implementation is just one alternative.
  - *Much smaller than linked list of all HF pages*!

Page Format

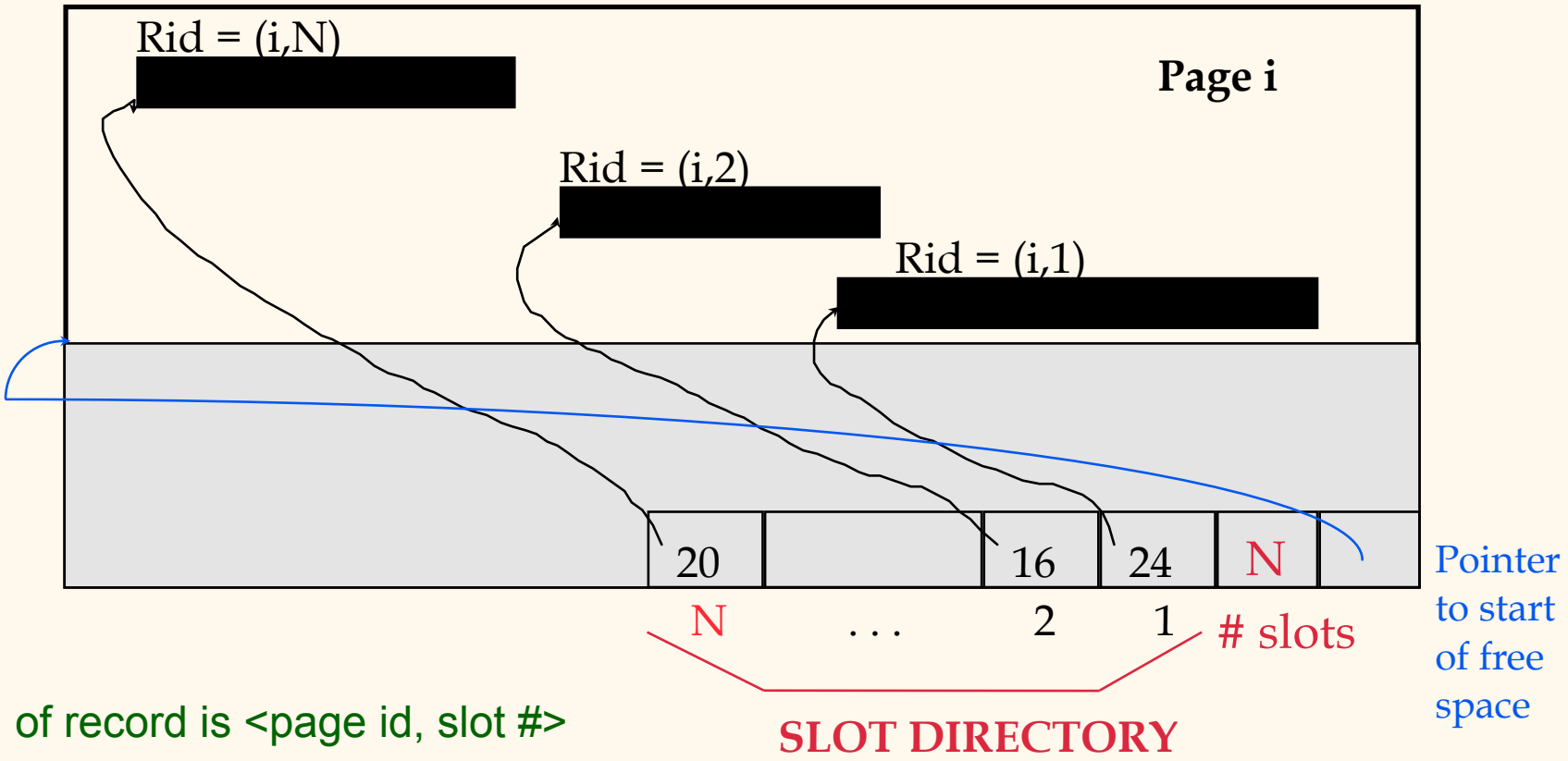
- How to store a collection of records on a page?
- Consider a page as a collection of slots, one for each record.
- A record is identified by rid = <page id, slot #>
- Record ids (rids) are used in indexes (Alternatives 2 and 3).

Page Format: Fixed Length Records



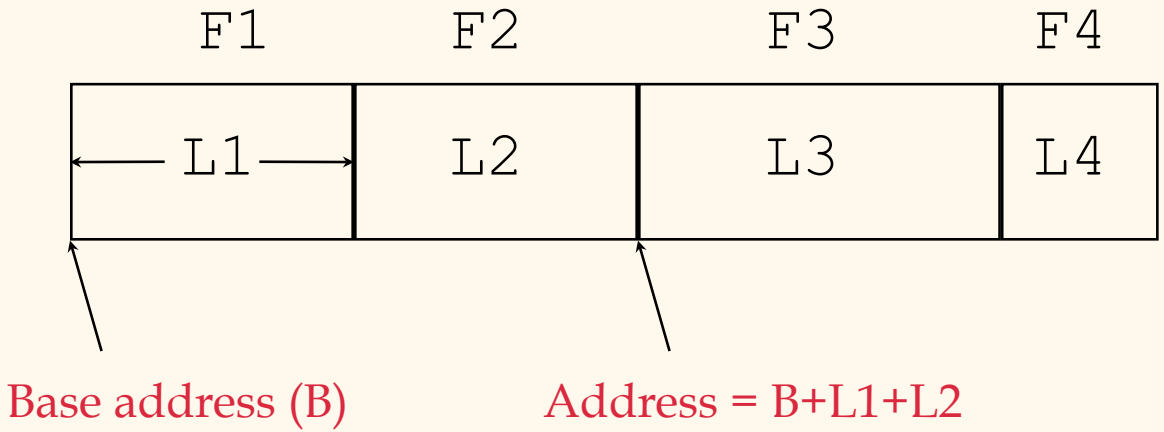
- *Moving records for free space management changes rid! May not be acceptable.*

Page Format: Variable Length Records



- *Can move records on page without changing rid; so, attractive for fixed-length records too. (**level of indirection**)*

Record Format: Fixed Length

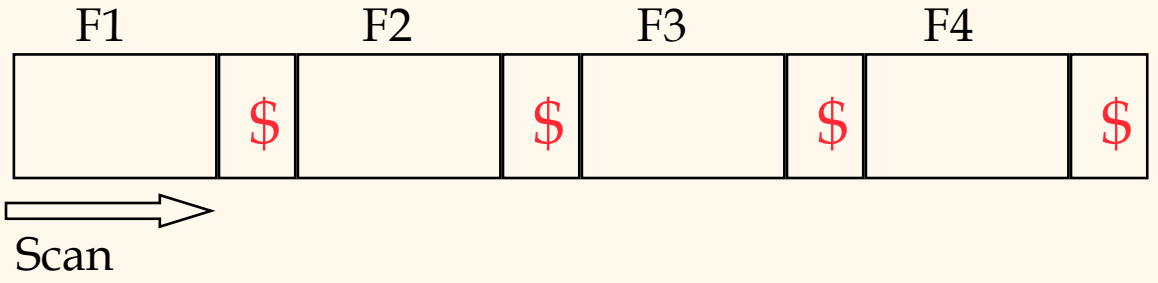


- Information of a *record type* e.g., the number of fields and field types is stored in the *system catalog.*
- Fixed length record: (1) the number of fields is fixed, (2) each field has a fixed length.
- Store fields consecutively in a record.
- Finding *i'th* field does not require scan of record.

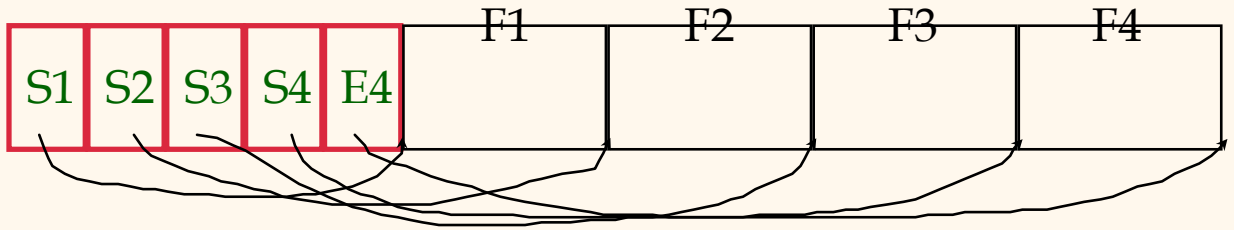
Record Format: Variable Length

- Variable length record: (1) number of fields is fixed, (2) some fields are variable length
- Two alternatives:

Fields Delimited by Special Symbols



Array of Field Offsets



☛ Second offers direct access to i'th field, efficient storage of *NULLs* ; small directory overhead.

System Catalogs

- For each index:
  - structure (e.g., B+ tree) and search key fields
- For each relation:
  - name, file name, file structure (e.g., Heap file)
  - attribute name and type, for each attribute
  - index name, for each index
  - integrity constraints
- For each view:
  - view name and definition
- Plus statistics, authorization, buffer pool size, etc.

☛ *Catalogs are themselves stored as relations*!