Storage

Database Systems: The Complete Book

UBDB Seminars

Mondays @ 10:30 AM in Davis 113A

Feb 15: Rethinking the Database for the Data Science Era
Zack Ives (UPenn)

Feb 22: Large-Scale Machine Learning With The SimSQL System
Chris Jermaine (Rice)

Mar 21: Approximate lifted inference with probabilistic databases
Wolfgang Gatterbauer (CMU)

April 18: Title TBD
Ihab Ilyas (Waterloo)

http://odin.cse.buffalo.edu/seminar/index.html

Recap

R	A	B
	1	1
	1	2
	2	3
	3	4
	1	5
	3	6
	2	7
	1	8

Recap

R	A	B
	1	1
	1	2
	2	3
	3	4
	1	5
	3	6
	2	7
	1	8

$\gamma_A(R)$	A	Groups
	1	**{** <1,1> <1,2> <1,5> <1,8> **}**
	2	**{** <2,3> <2,7> **}**
	3	**{** <3,4> <3,6> **}**

Recap

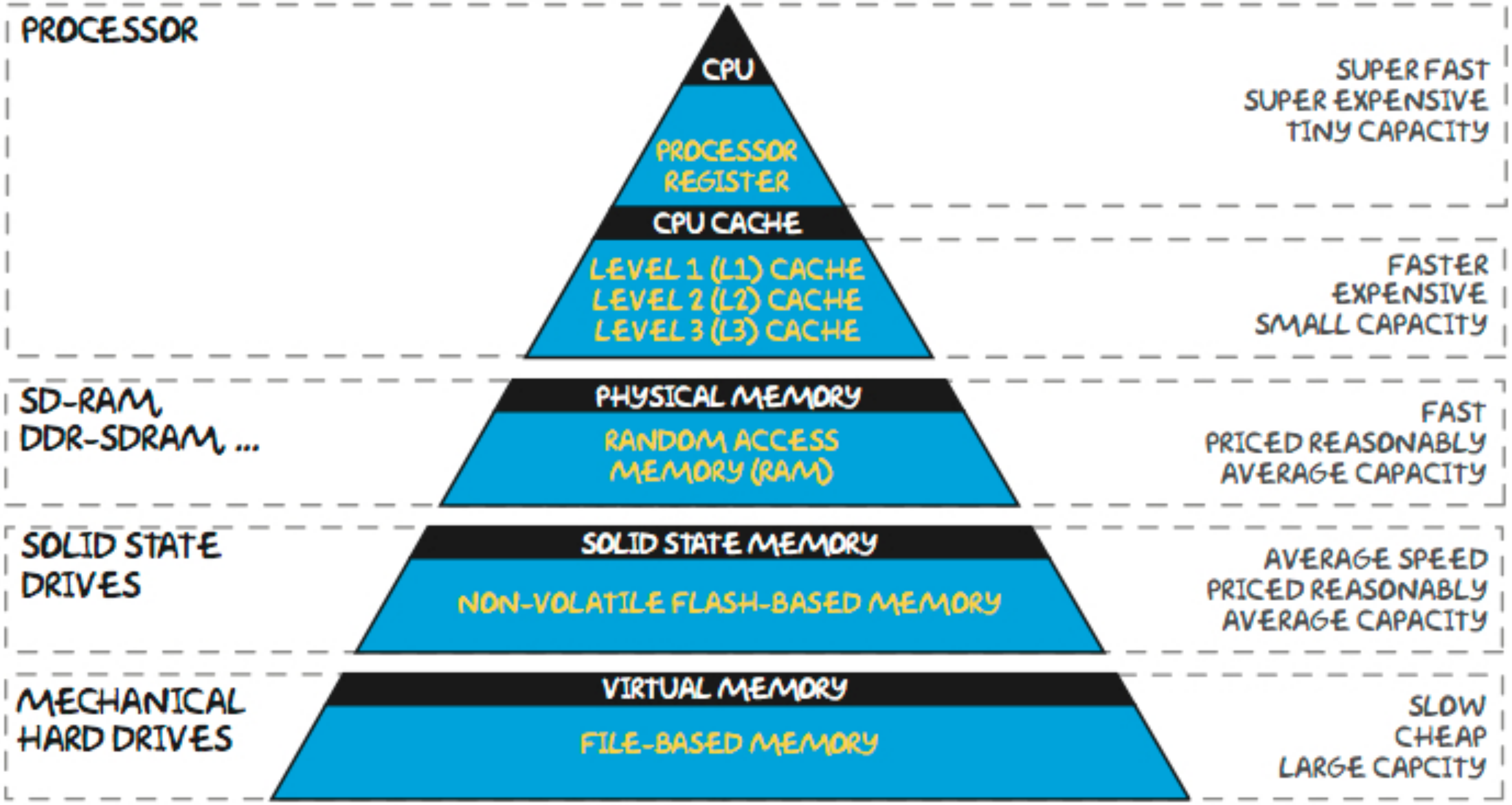
R	A	B
	1	1
	1	2
	2	3
	3	4
	1	5
	3	6
	2	7
	1	8

$\gamma_{A}(R)$	A	Groups
	1	{ <1,1> <1,2> <1,5> <1,8> }
	2	{ <2,3> <2,7> }
	3	{ <3,4> <3,6> }

$\gamma_{A,\ SUM(B)}(R)$	A	SUM(B)
	1	16
	2	10
	3	10

The Memory Hierarchy

Fast (but small)



Big (but slow)

Storage

- How do we…

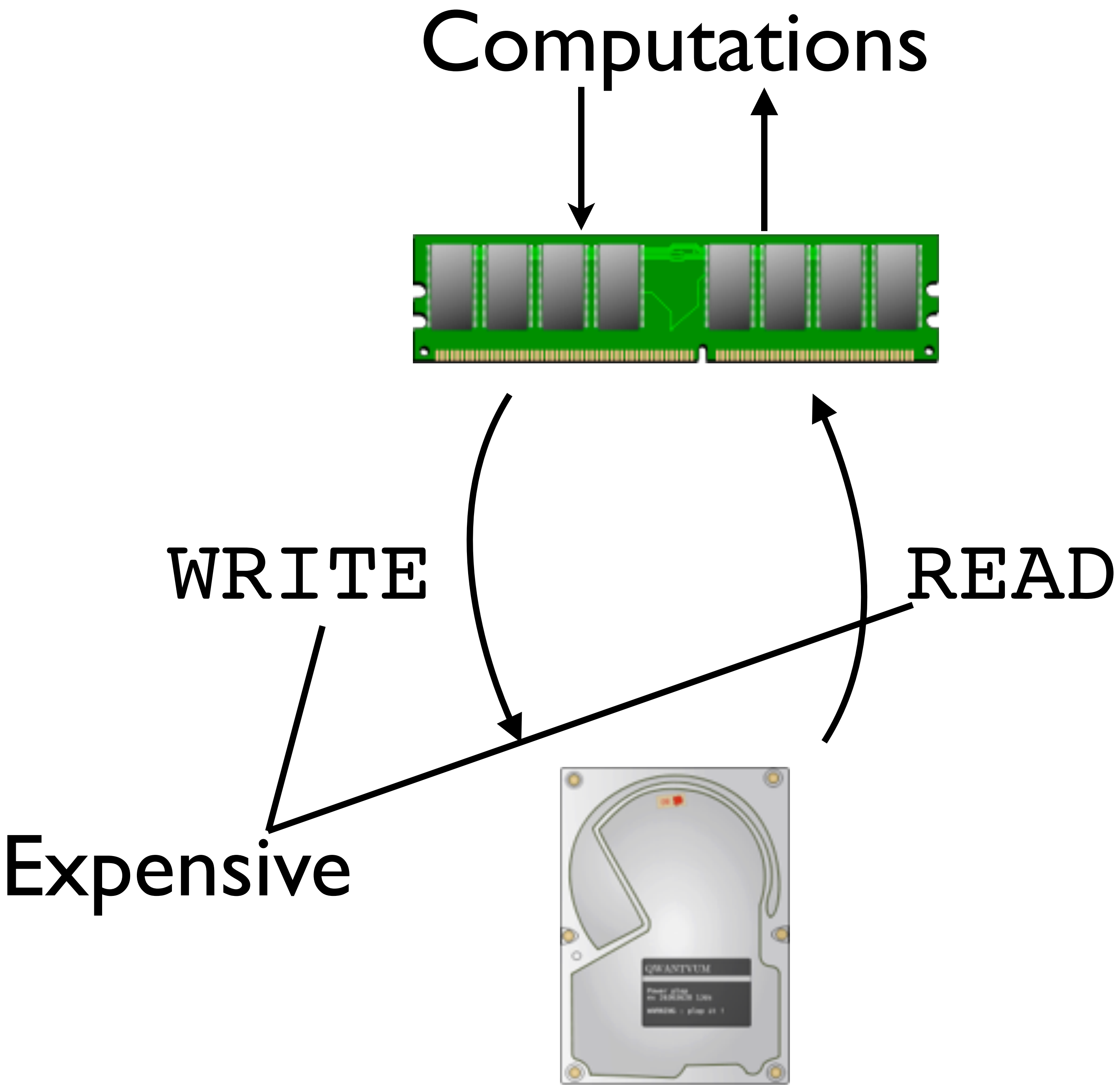
Buffer Manager

  - …optimize across the memory hierarchy?
  - …use the right data access pattern for the storage medium we’re using?

File Manager

  - …organize data to minimize access costs?
  - …organize data to minimize storage costs?

The IO Problem



Computations

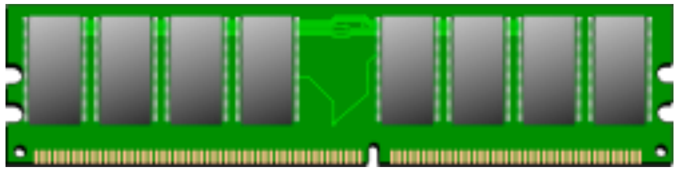
WRITE

READ

Expensive

Why not use just RAM?

- RAM is more expensive than HD
    - 200 MB/$ vs 10 GB/$
- RAM is smaller
    - 128 GB vs 10 TB
- RAM is volatile



Are in-memory databases still viable? (Hint: Yes)

In-Memory DBs

- Why use In-Memory DBs?
    - Faster processing (especially for random access)
- How can we provide persistence?
    - … with respect to local failures (crashes)
    - … with respect to global failures (hurricanes)
- How do we provide scale?
    - Some DBs need TB/PB/EB of space.

Select Bottlenecks

High Latency if source is disk!

```
def Select(predicate, source)
  while(source.hasMoreTuples)
    tuple = source.readTuple()
    if(predicate(tuple))
      output(tuple)
```

Where is output stored?

IO + Buffering

```
def Select(predicate, source)
  while(source.hasMoreTuples)
    in_buffer = source.fetch()
    while(in_buffer.hasMoreTuples)
      tuple = in_buffer.readTuple()
      if(predicate(tuple))
        out_buffer.output(tuple)
      if(out_buffer.isFull)
        out_buffer.flush()
```

Data Organization

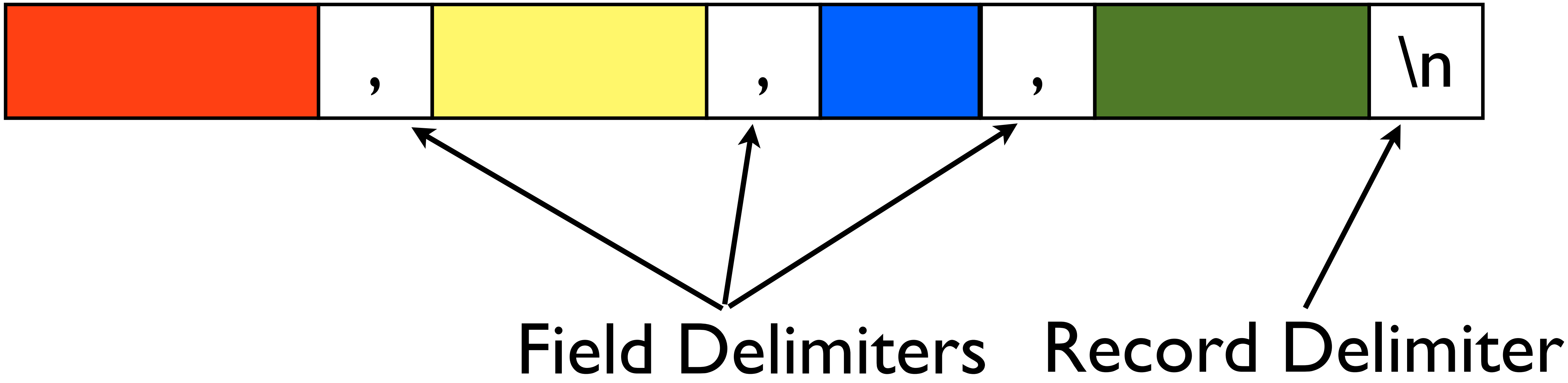
- How do we store data?
    - How are records represented on-disk? (Serialization)
    - How are records stored within a page?
    - How are pages organized in a file?
    - What other metadata do we need?
- Our solutions must also be persisted to disk.

Files and Data

- A **File** is a collection of pages
    - A **Page** is a collection of records
        - A **Record** is a data value (e.g., a tuple)
- We need an infrastructure to ensure that records we need are in memory.
- We need some way to organize and store files, pages, and records.

How is data laid out in a record?

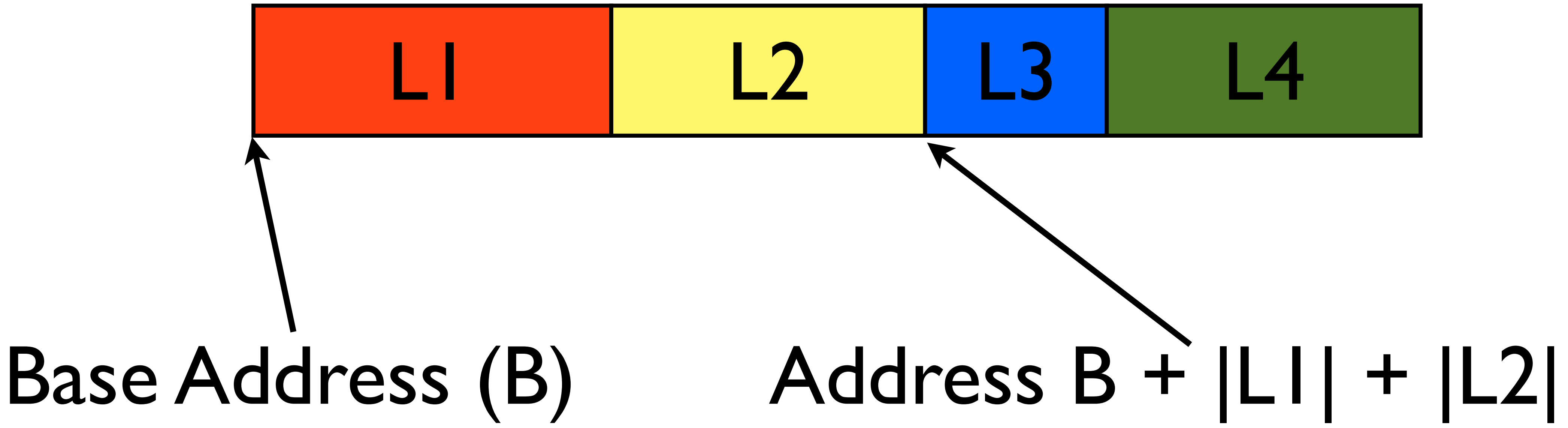
Record (Tuple) Formats



What are some advantages/disadvantages of storing records this way?

Record (Tuple) Formats

Record information stored in a **System Catalog**



L1

L2

L3

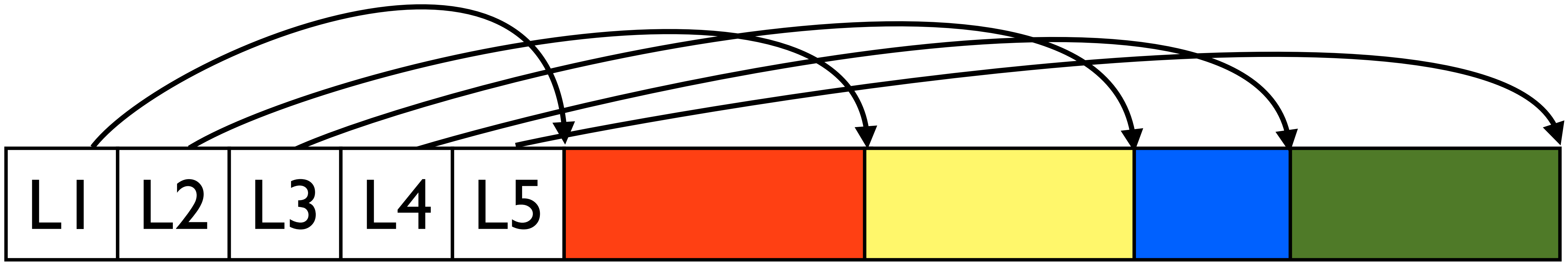
L4

Base Address (B)

Address B + |L1| + |L2|

What are some advantages/disadvantages of storing records this way?

Record (Tuple) Formats



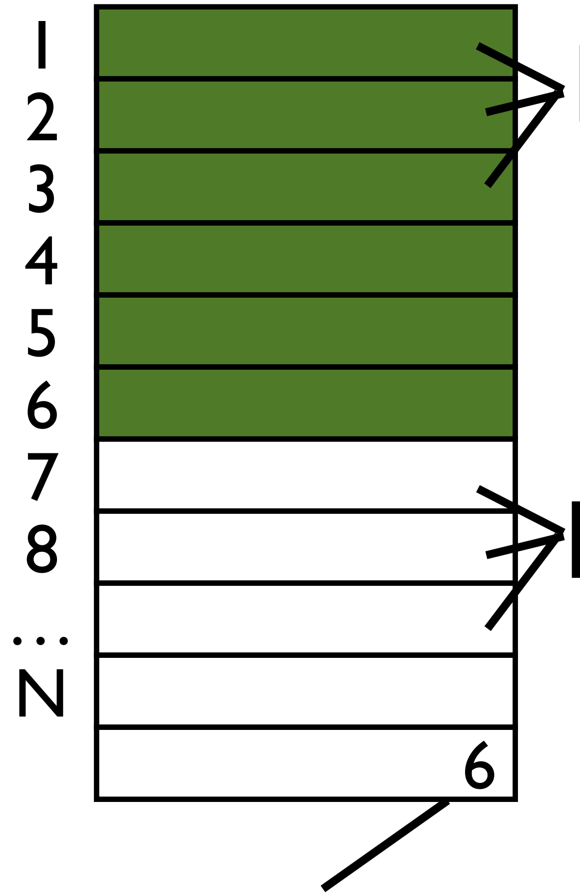
Array of Field Offsets

What are some advantages/disadvantages of storing records this way?

How are records laid out in a page?

Page Formats

Packed

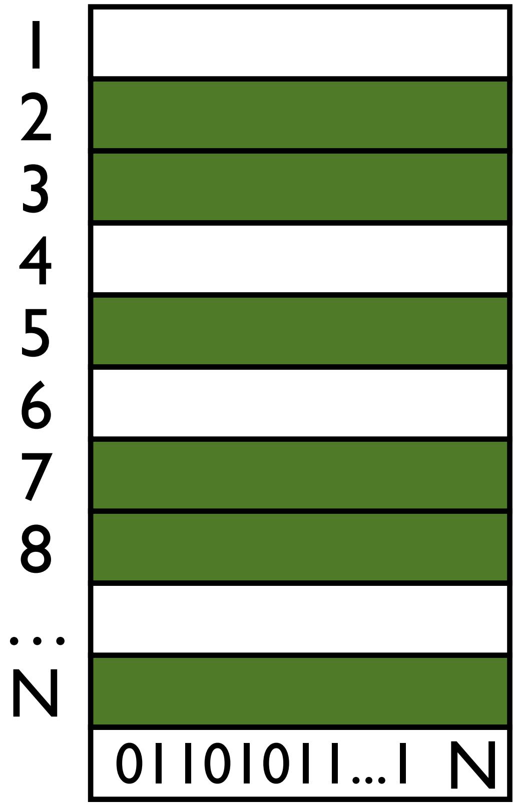


Data Records

Free Space

Number of records

Unpacked, Bitmap

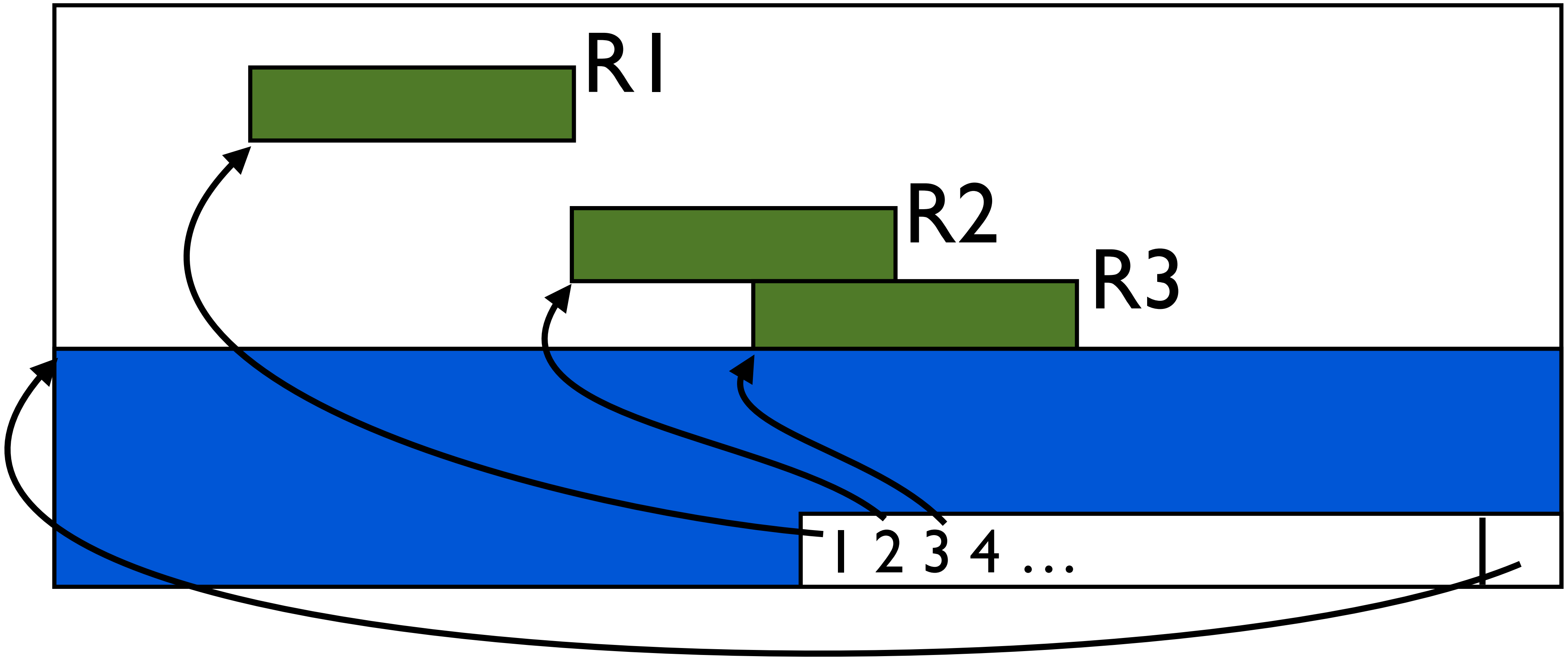


Bit array of occupied slots
(and size of page)

What are advantages/disadvantages of these formats?

Page Formats

Variable Size Records



Pointer to start of free space

What are advantages/disadvantages of this format?

Files of Records

File: A collection of pages of records that must support:

Read a record

Insert/Delete/Update a record

Scan all records

Unordered (Heap) Files

Store records in no particular order

Disk pages are allocated/freed as file grows and shrinks

Support for record level operations by:

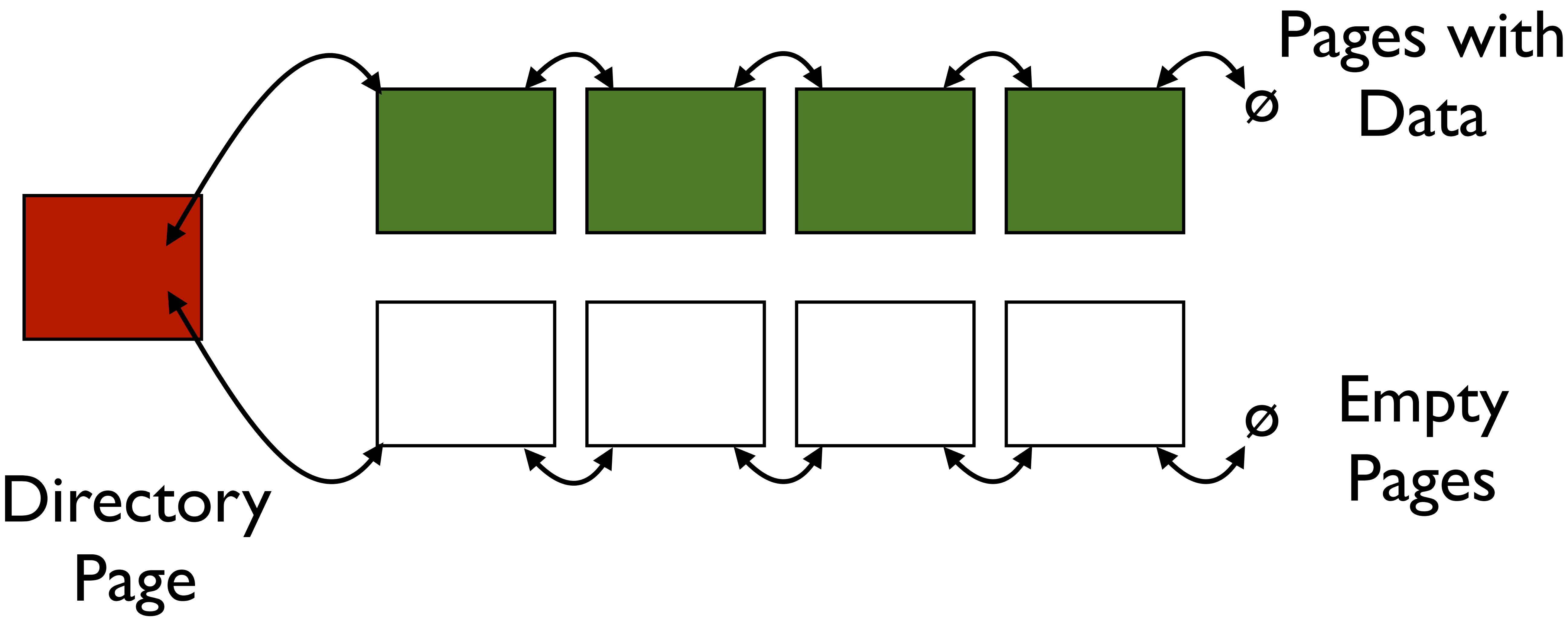
Keeping track of pages in the file

Keeping track of free space in each page

Keeping track of records on each page

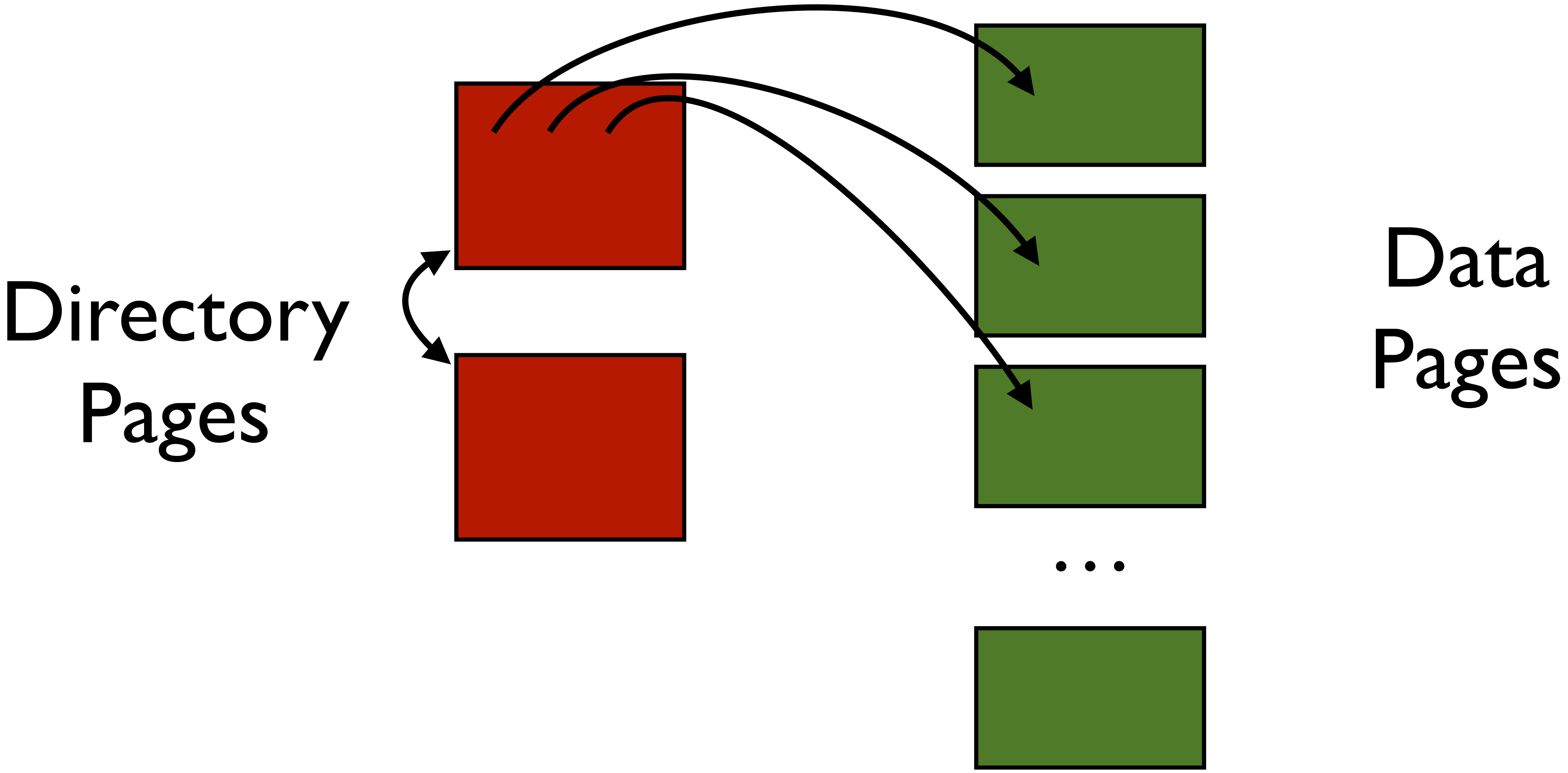
This data must be stored somewhere!

Unordered (Heap) Files

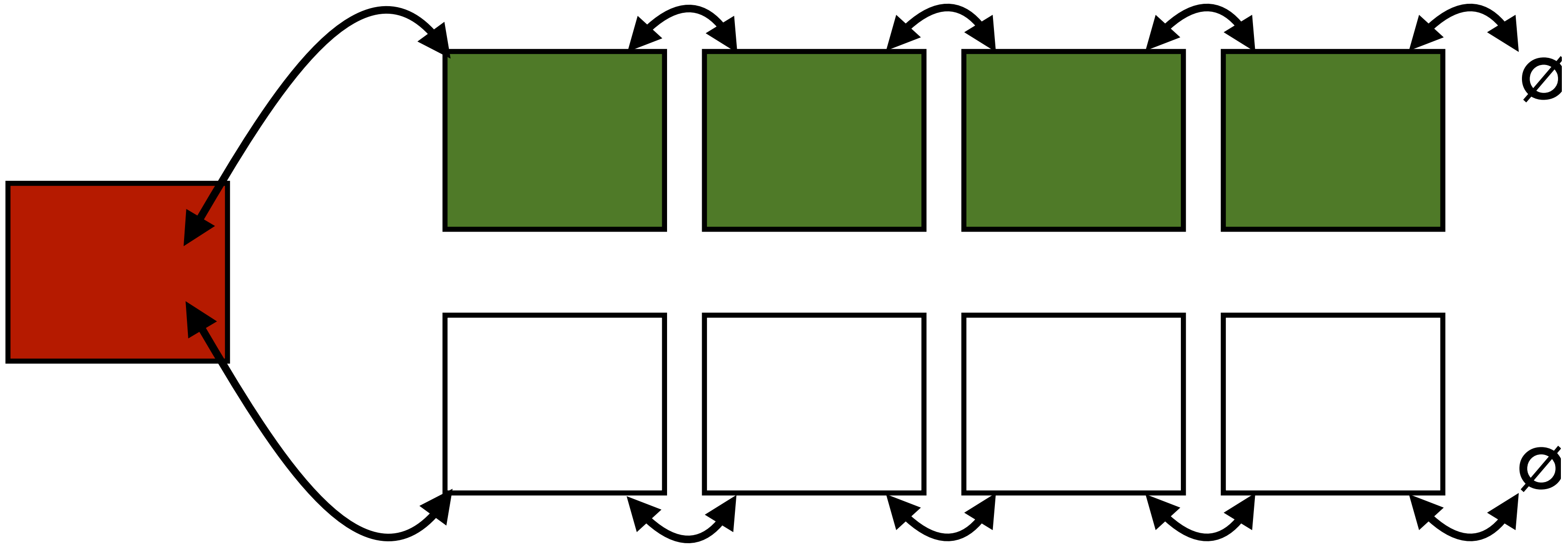


Each page contains 2 pointers plus data

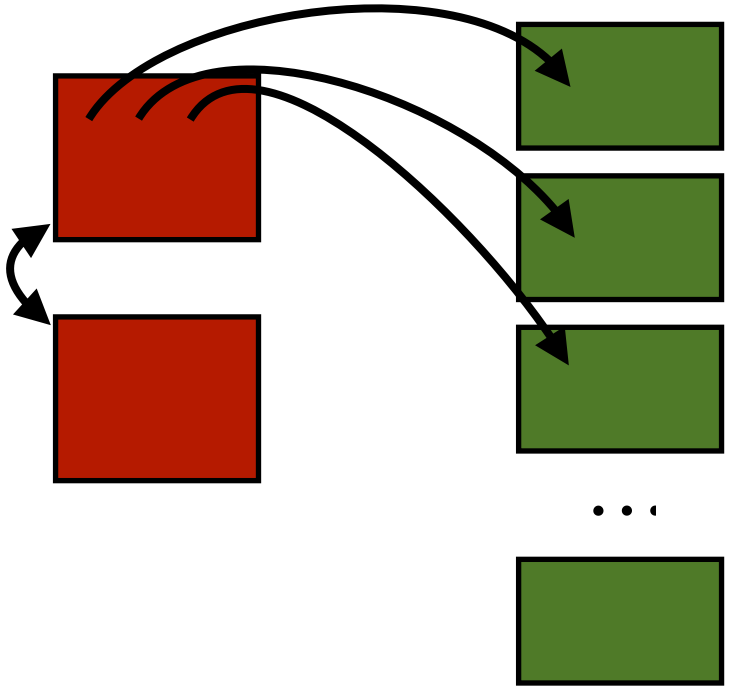
Unordered (Heap) Files



Directories are a collection of pages (e.g., a linked list)
Directories point to all data pages
(entries can include # of free pages)



What are the advantages and disadvantages of each?



IO + Buffering

```
def Select(predicate, source)
  while(source.hasMoreTuples)
    in_buffer = source.fetch()
    while(in_buffer.hasMoreTuples)
      tuple = in_buffer.readTuple()
      if(predicate(tuple))
        out_buffer.output(tuple)
      if(out_buffer.isFull)
        out_buffer.flush()
```

IO + Buffering

Generalize & Standardize!

Have a component that handles buffering!

The Buffer Manager

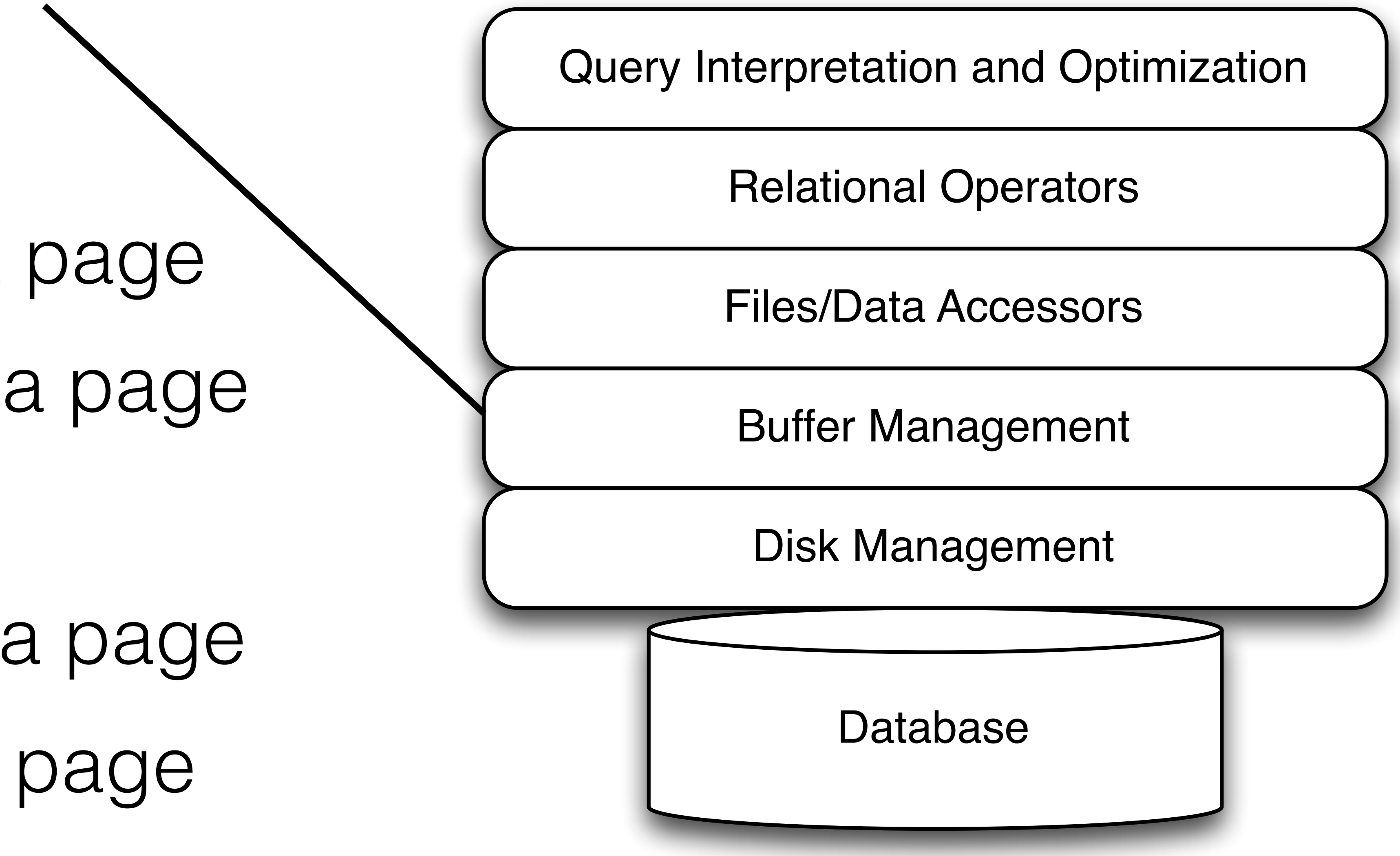
API

Allocate a page

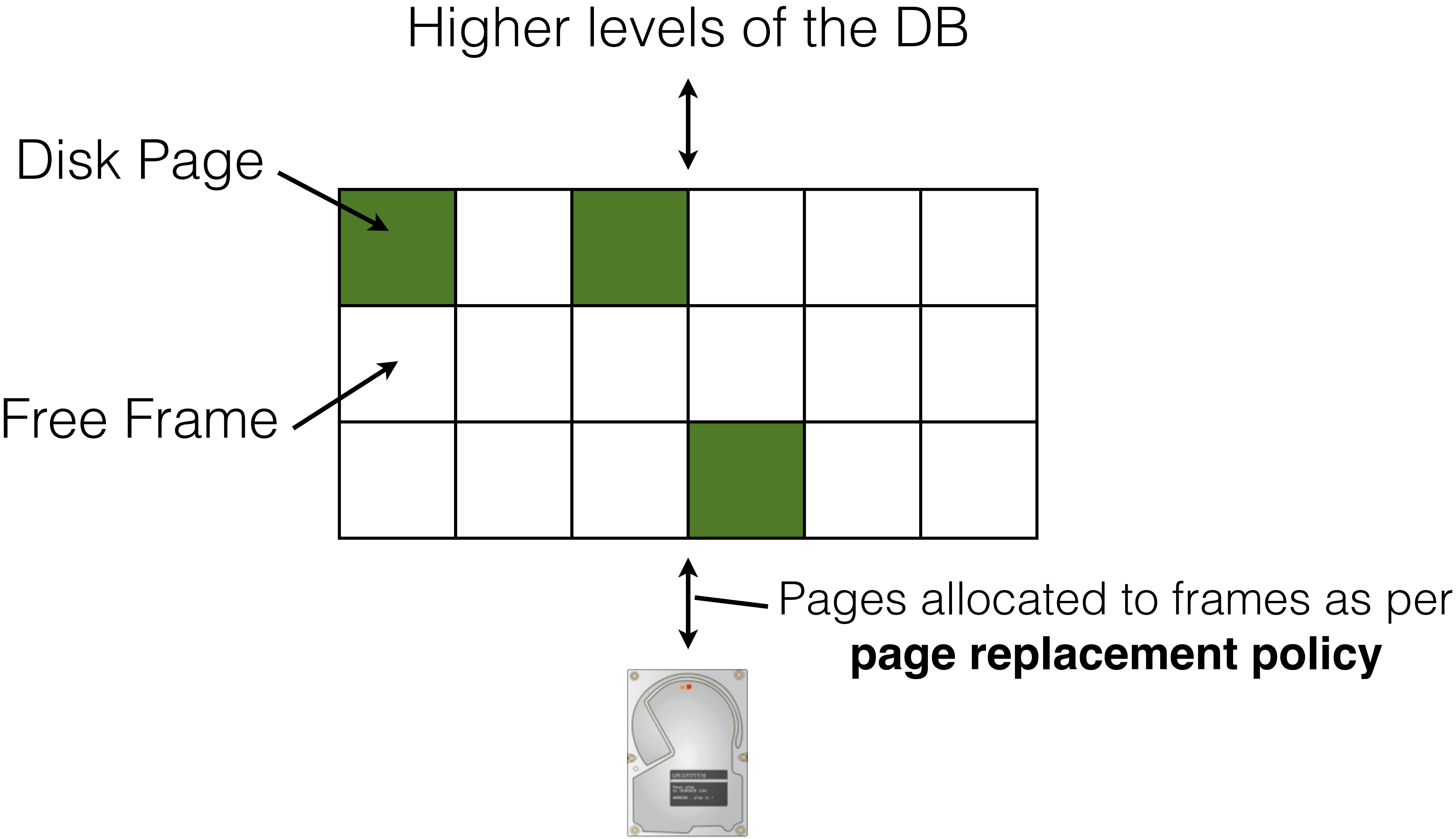
Deallocate a page

Read from a page

Write to a page



The Buffer Manager



Higher levels of the DB

Disk Page

Free Frame

Pages allocated to frames as per **page replacement policy**

Pinned Pages

- Pinning a page indicates that it is being used.
- The requestor must unpin the page when done.
  - The requestor must also indicate whether the page has been modified (with a ‘dirty’ bit)
  - Dirty pages must be written to disk
- Pages may be requested multiple times
  - Use a pin count (reference count) to keep track.
- Concurrency Control/Recovery may require other operations when replacing a frame.

Buffer Replacement

- Frames are chosen for replacement by a **buffer replacement policy**.
  - (e.g., LRU, MRU, Clock)
- Policy can have a big impact!
  - Depends on the access pattern.
- What is a worst-case scenario for LRU?

Hmmm… this sounds awfully familiar…

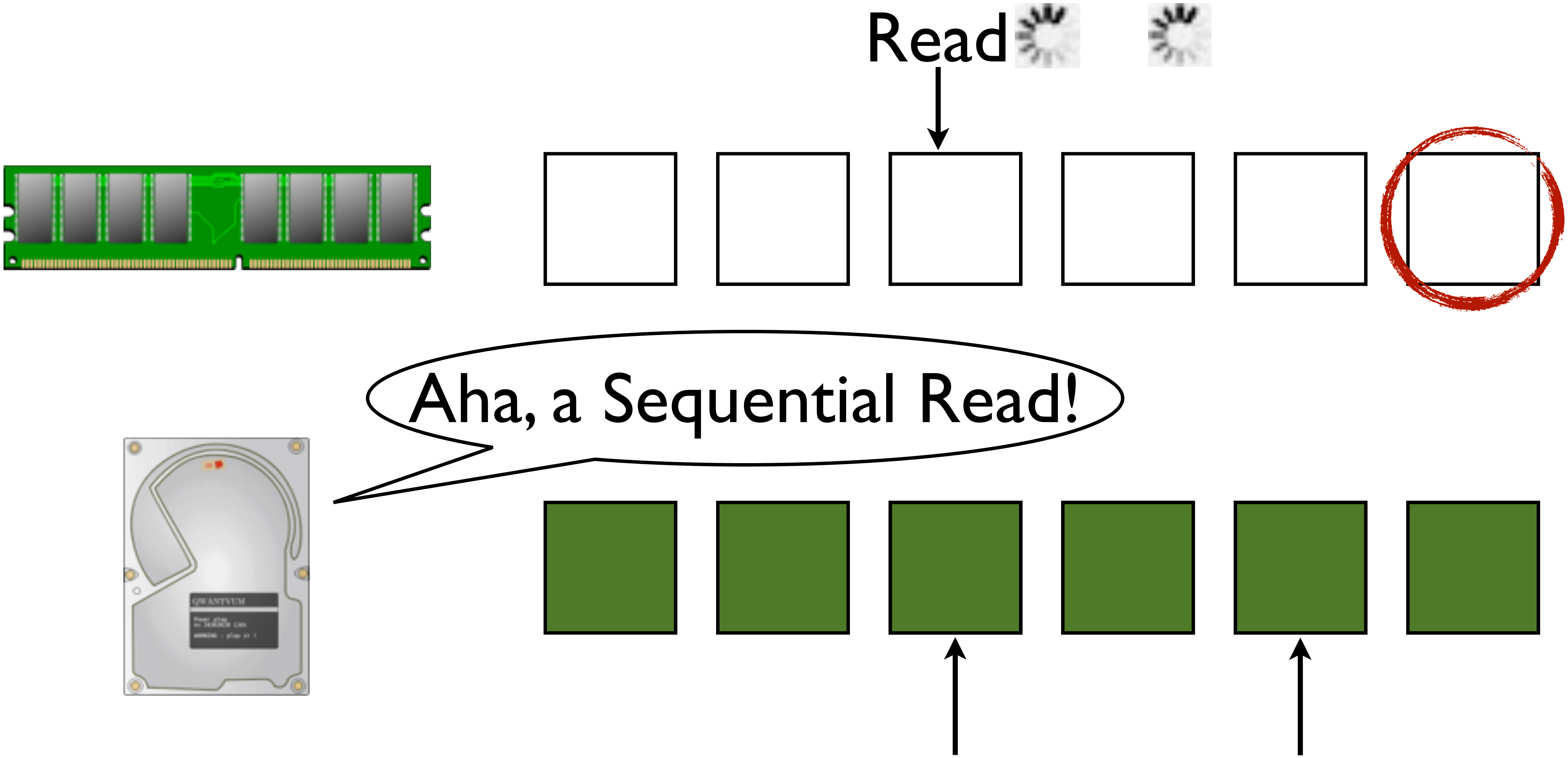
Hey… Oliver!

This sounds a lot like virtual memory!

Buffer Managers vs Virtual Memory

- Not a huge difference
  - Many lightweight DBs use VMem as a buffer manager!
- Reasons to implement an explicit buffer manager:
  - Control when and how paging happens.
    - e.g., better/more efficient prefetching.
  - Control what gets paged in/out.
    - e.g., better knowledge of data access patterns.
  - Integrate additional memory layers (e.g., Network)

Example-OS Paging



Read

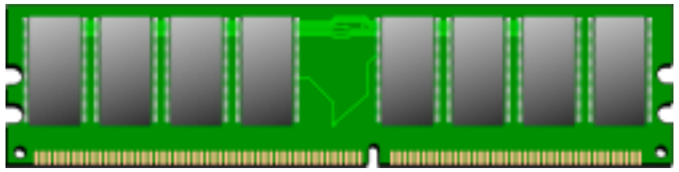
Aha, a Sequential Read!

Index: 'Enterprise officers in this range'

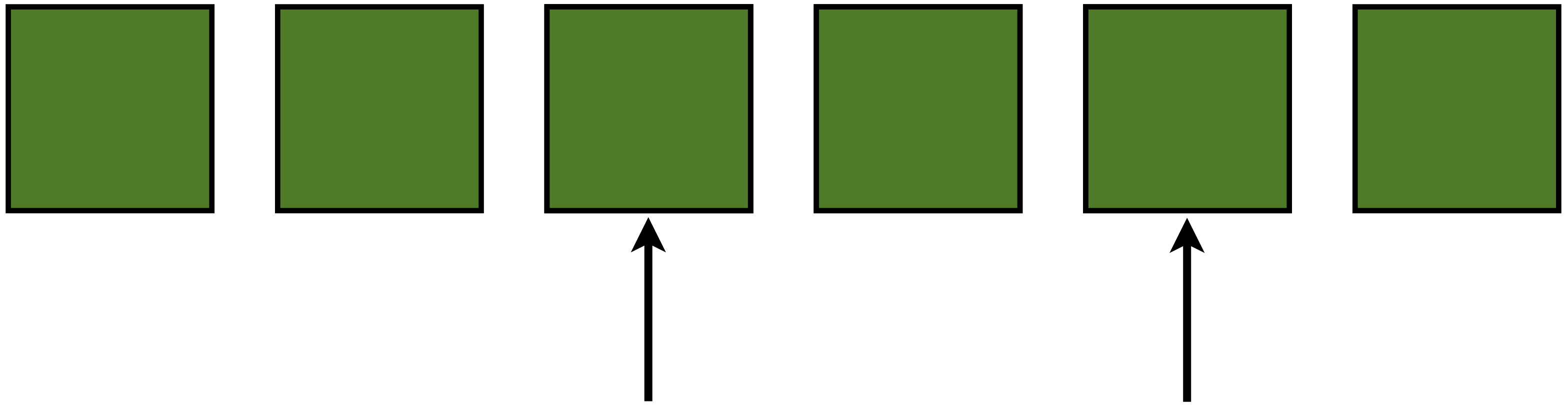
image credit: openclipart.org

Example-DB Paging

Read ↓



Read in precisely what you need.



Index: ‘Enterprise officers in this range’

image credit: openclipart.org

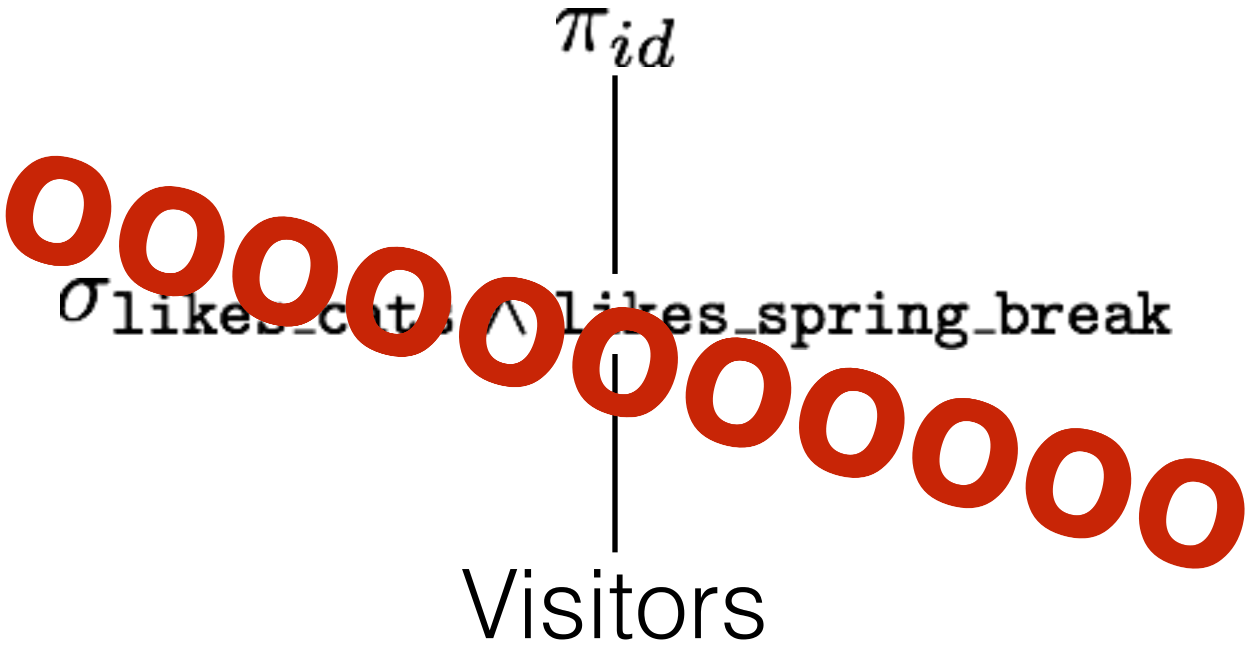
Time permitting…

The record is the main unit of computation…

… but what if the records are really really really big

```
CREATE TABLE visitor(
  id big_int,
  ip int,
  age int,
  gender enum,
  …
  region string,
  country string,
  city string,
  …
  likes_cats bool,
  likes_spring_break bool,
  likes_cookies bool,
  …
);
```

Google, Facebook, Amazon, etc… have log files and customer information tables with 100s or 1,000s of columns.



π_{id}

$\sigma_{\texttt{likes\_cats} \wedge \texttt{likes\_spring\_break}}$

Visitors

[Slooooooooooooooow]