RAM ⟺ EM

↑

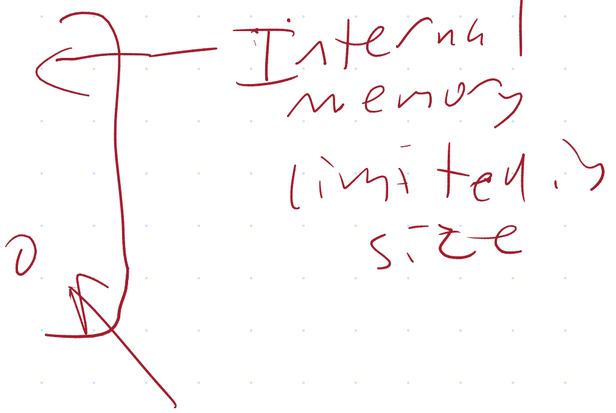
O(1) Array

↑

Internal : O(1) access

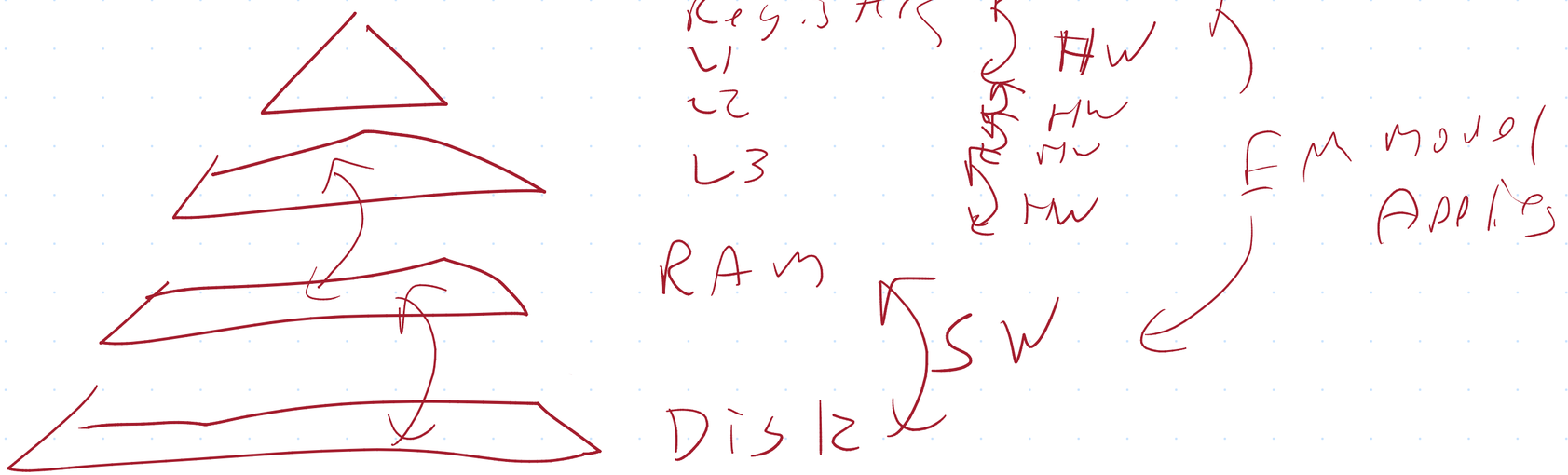
External : No access

↳ Copy data for I/O

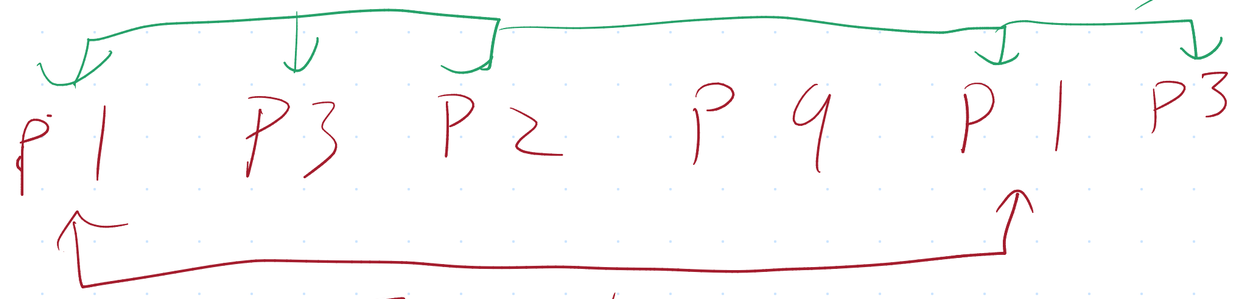


- Runtime cost
- I/O cost
- Mem reqd

- Runtime vs


Registers
L1
L2
L3
RAM
Disk
HW
HW
HW
HW
SW
FM model
Apps

Temporal locality

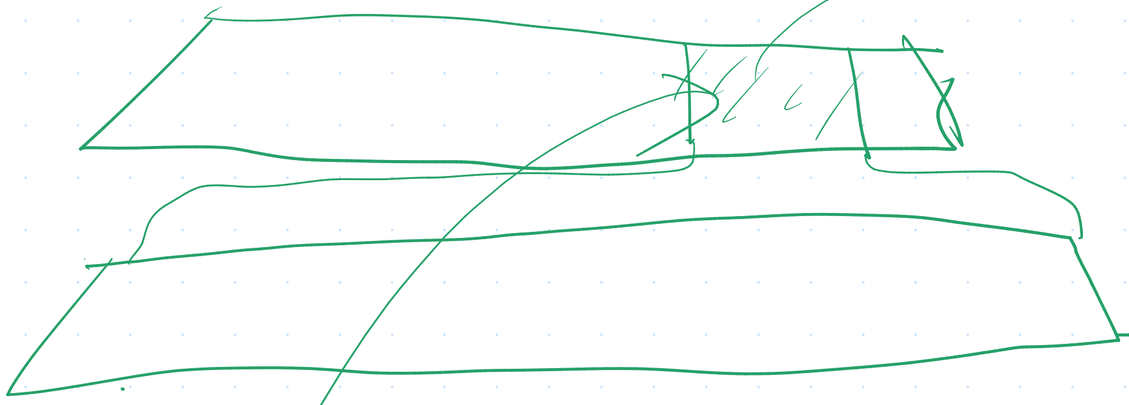


Temporal locality

Spatial locality
↳ Turns into temporal locality at lower levels
↳ Enables prefetching

← Caching means that sufficiently local reads only need I/O

Spatial locality



Cache Replacement → LRU
↳ Least Recently Used

pro
- Eviction
  ↳ log(N)

Second Chance
↳ Sweep over frames
  ↳ If used bit set
    unset it
  ↳ else
    ↳ evict page

pro
- One bit per frame
- don't need timestamps or heaps
- Lighter weight
- Mostly behaves like LRU

con
- Eviction may be linear

Writeback
↳ Enforcing Atomicity

Tidy Data

- Each cell has one thing
- Each record describes one "entity"
- Each data value lives in exactly one place

No: Redundancy
↳ More consistent
↳ Easy to find/search

Relations/Tables

- Project: changes the columns of a table
- Selection (filter): removes rows
- Join: combines tables: pairing records using a join condition
- Union: combines rows of tables
- Aggregation / G.B.: statistics

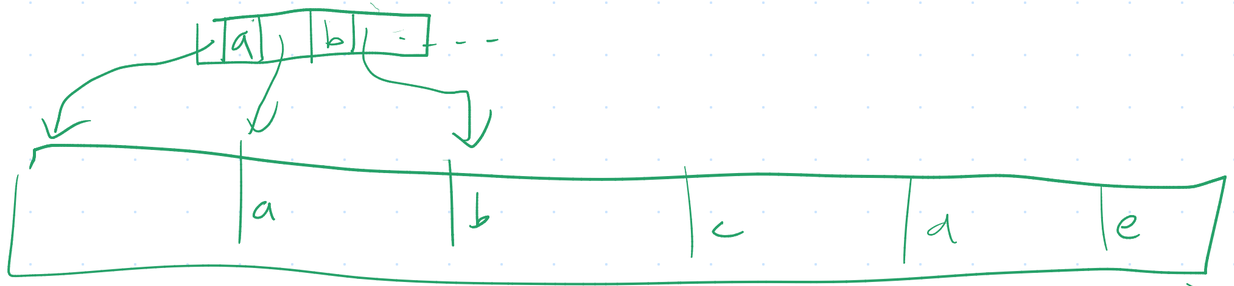
relational algebra

Serialization

- Field → Array of bits
- Record → Array of bits
- Array of records → Page (Array of bits)

Tree Based Data layout → Sort

↳ Fp-Table



access any record with 1 IO

O(N) memory

(Very small constant)

Con

hard to modify

↳ ISAM

multiple layers

access any record w/ $\log_k(N)$ IOs

k: fairly large (size of page)

↳ B+ Tree

↳ ISAM, but leave up to 50% space in each data & dir page

- ↳ If page fills up: Split it
- ↳ If page falls below 50%
  - ↳ steal records from adj pages
  - ↳ merge it adj pages already at 50%

forces a like change to parent

↳ $O((\log_{\frac{K}{2}}(N))$ IOs to access

Hash

T

$Hash(R) = 2$

Write-Optimized

LSM

production { FP Table w/ sorted on-disk array

original { ISAM Index ↖ p5

w/ ℓ levels

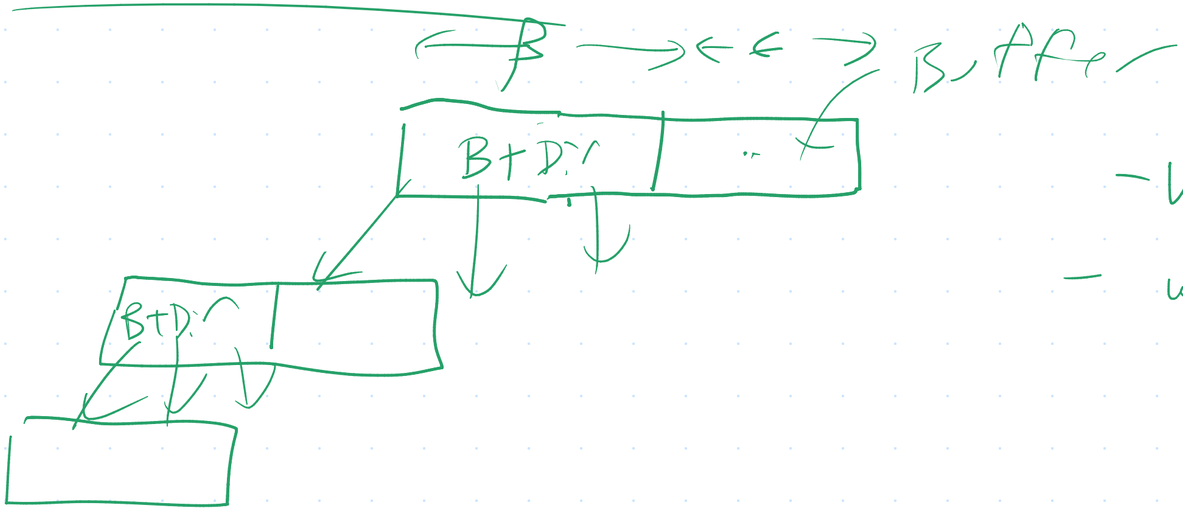
No record is copied $> \ell$ times

↻ write amplification $\leq \ell$

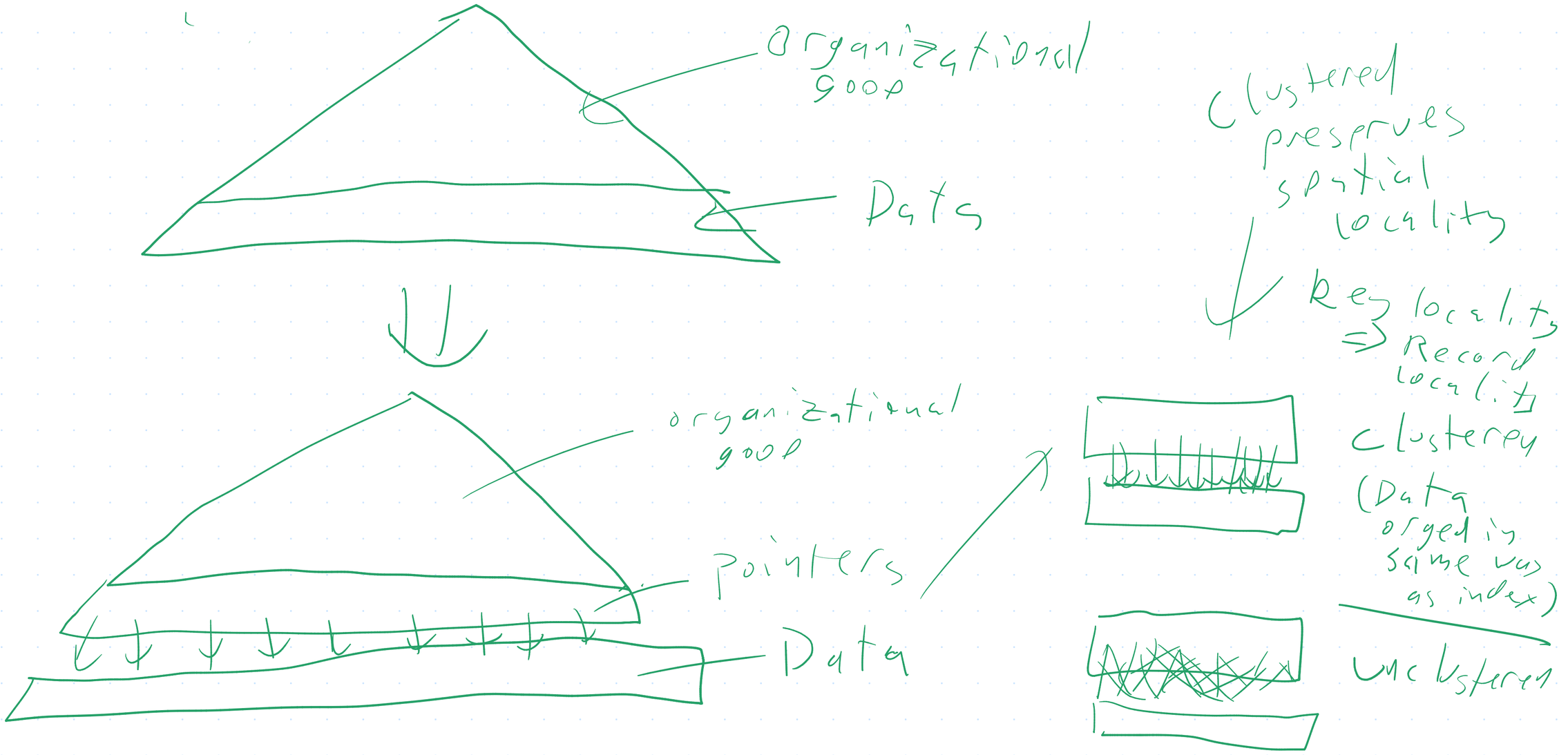
Read amplification $\leq \ell$

Good for range scans

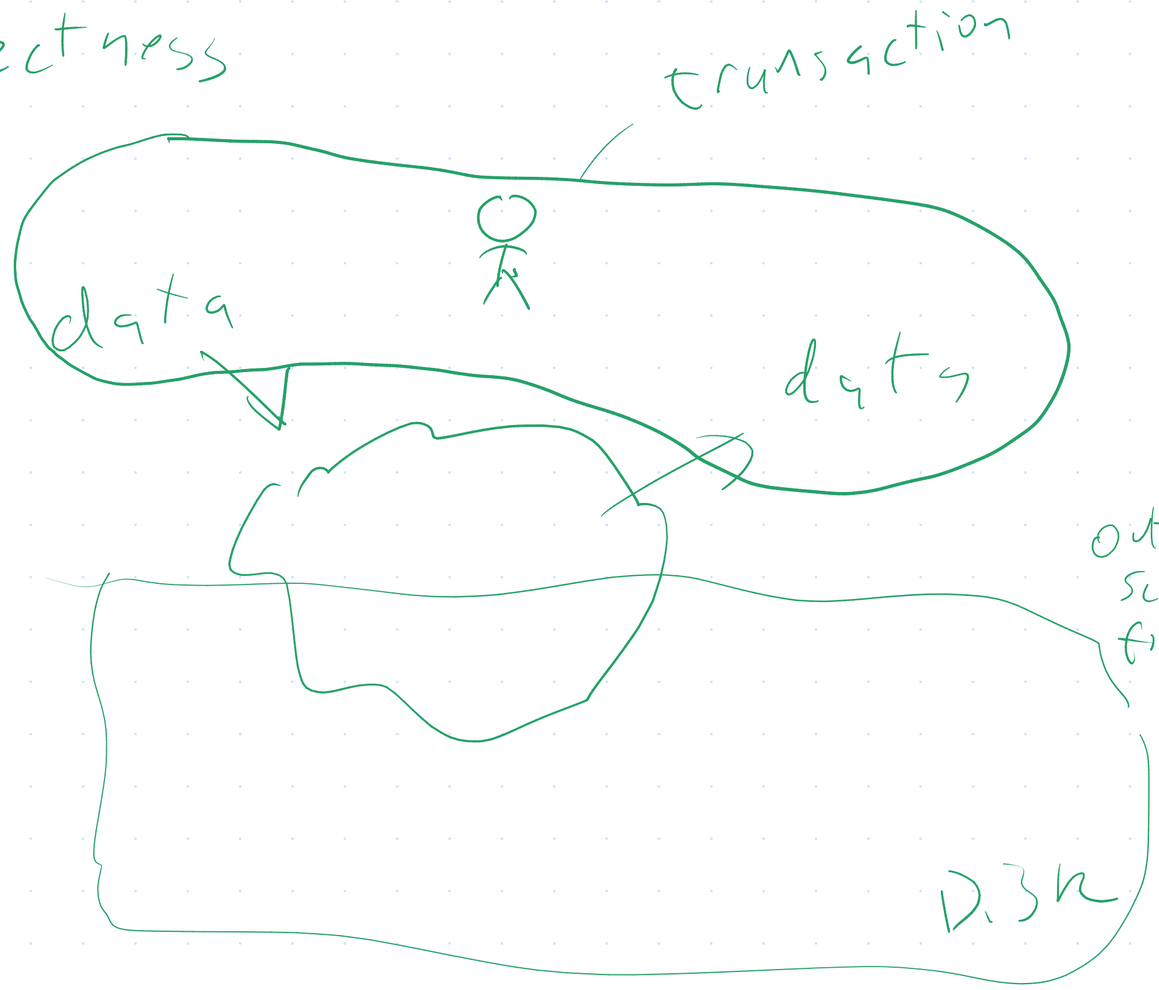
B-ε Tree



- writes go to buffer in root
- when buffer full, empty buffer & write to next level
  - ↳ repeat as needed


organizational
good
Data
organizational
good
pointers
Data
clustered
presprues
spatial
locality
key locality
⇒ Record
locality
clustered
(Data
orged in
same way
as index)
unclustered

Correctness



out of scope for 550

- **Atomicity** → Transactions effects are all visible or not visible at all
  - Commit / Abort
- **Consistency** → effects of xact are checked for ~~sanity~~
- **Isolation** → Each xact can pretend that nothing else is running
- **Durability** → When xact commits all its effects are "safe"

Atomicity + Durability

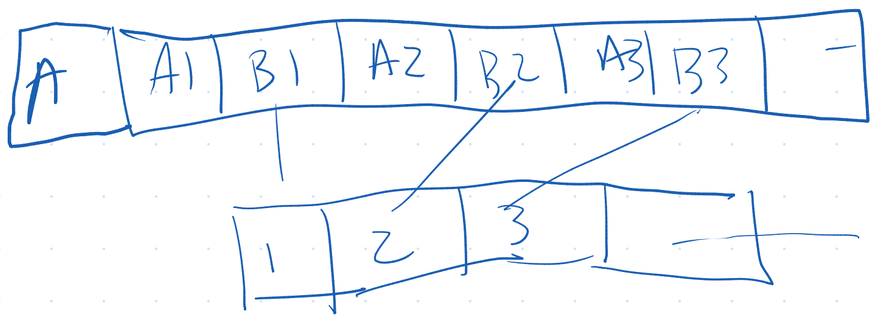
↳ Big problems: HW failure → RAID
Disaster → Replicating data.

↳ Power failure or crash → If writing many pages not every write may be successful

↳ Assumption: One page write will be atomic

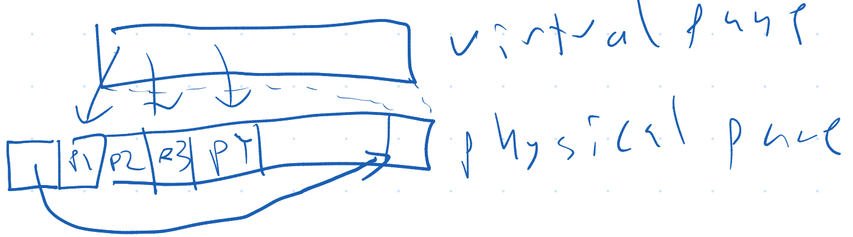
Data Duplication

Twin Block



- Flush edits to disk
- Flip version (commit)
- Copy new data over old
- Start editing old blocks

Shadow Paging



(Copy on Write)

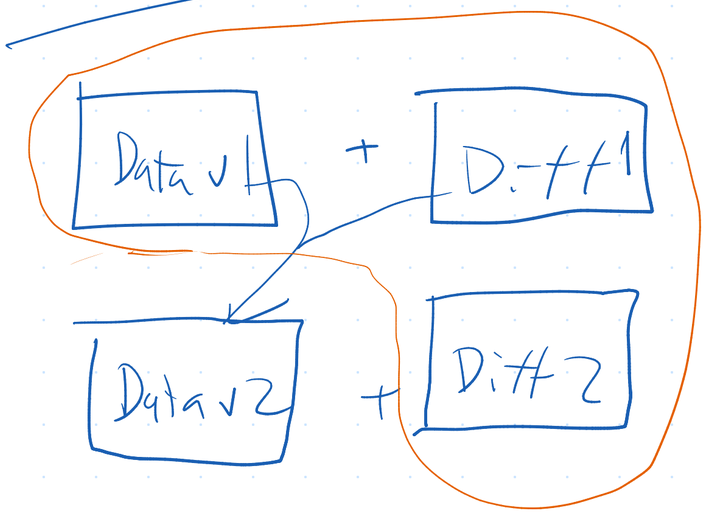
- write out page lookup table
- Flush edits to disk
- update version to point to new lookup table

pro / Less redundancy

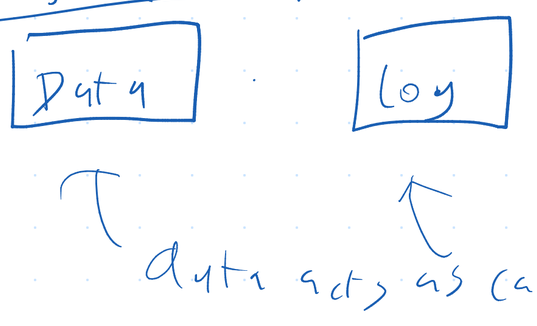
con / Lower preservation of spatial locality

Diff-based

Diff file



Log (WAL)



Atomicity

- Abort + Limited Mem
  - May need to write changes to disk before COMMIT
- Undo logging
  - Log prev version before write
    - To undo: replay log in reverse

Algorithms

- Sorting (2 pass sort)
  + optimization for bigger initial runs

- Joins (2 pass hash
  Block Nested Loop
  Sort-Merge)

Projects

0. Paged data access
1. BufferMgr
2. Record layout
3. Indexes
4. ESAM
5. LSM