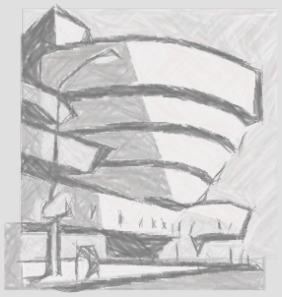
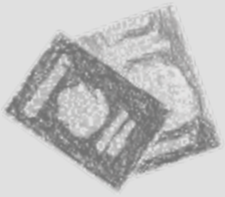
ACID Transactions at the PB Scale with MarkLogic Server

A talk by Nuno Job,
Welcome to Berlin Buzzwords 2011!


portugal, new york, toronto, san francisco, london

83 08 09 10 11



past.

MarkLogic

present.

stuff I like




HELLO
my name is

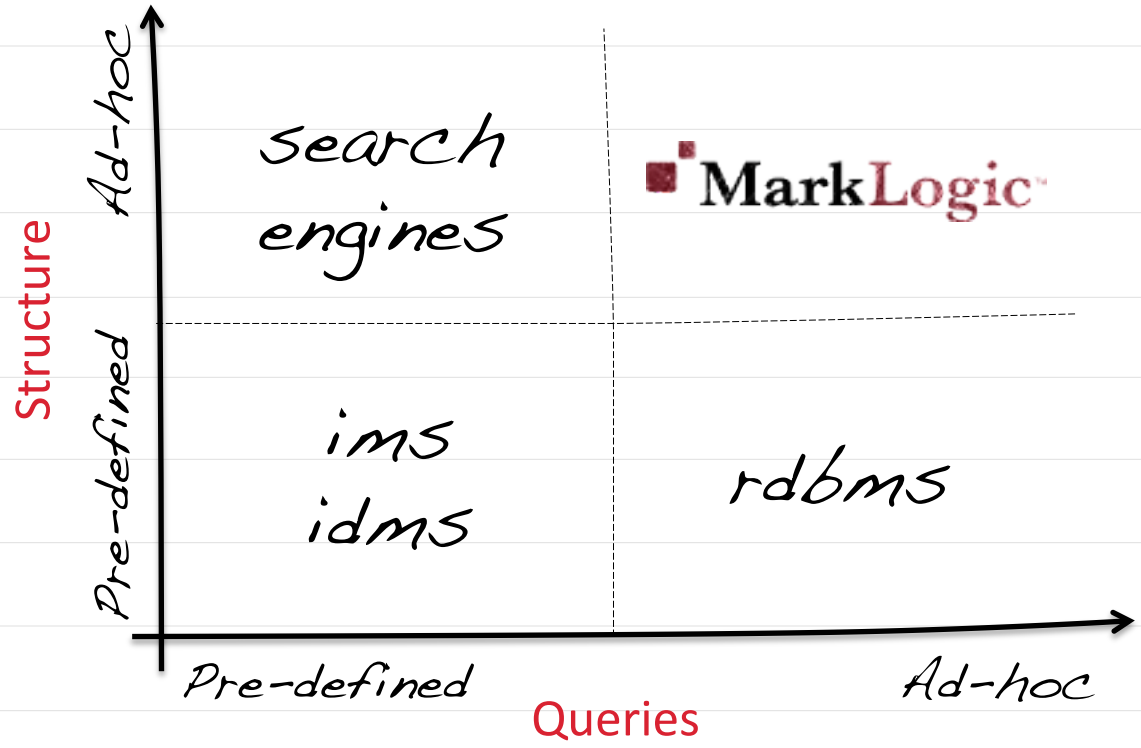
Nuno

@dscape

foreseeable future.

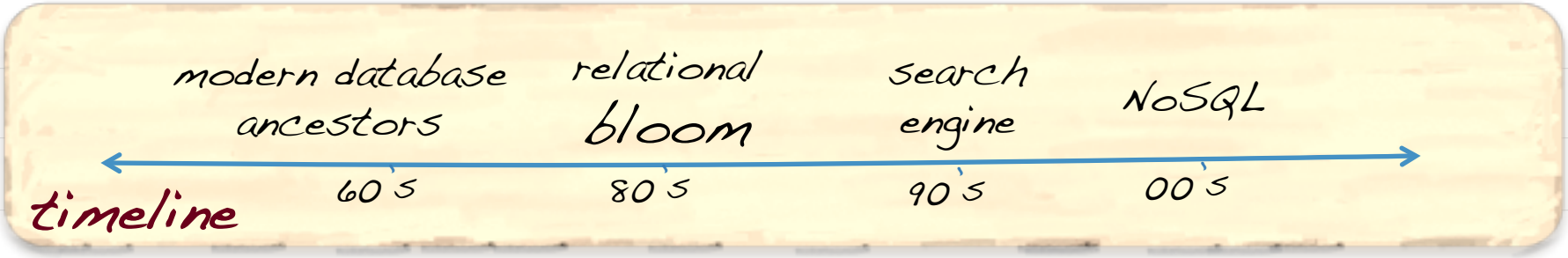
What is MarkLogic!?

the idea



dinosaurs are fun!

(oo)—',--



a database for unstructured information

unstructured
schema-less*
easy evolution.
xml or json.

c++ core
~ pb scale

also stores:
text and binaries

features
acid, backups
replication, query
language (xquery).

native search
a database built
on a search engine?

!! stop shredding your data
start storing data as is

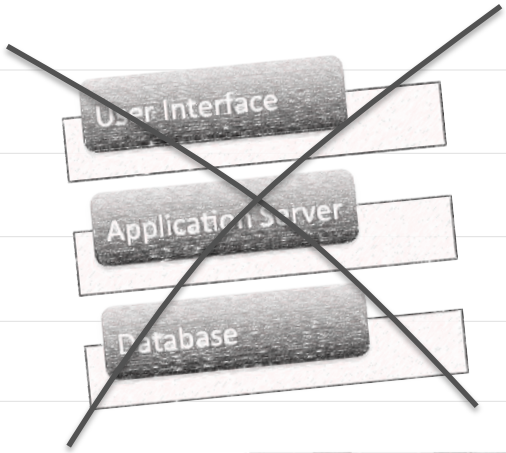
no tables, rows, columns
thinkin' documents
uris? looks like a filesystem

* they have this universal index thing.
an inverted index that is structure aware

application server

oysters at neptune tonight?

XQuery:
dynamic, functional programming language

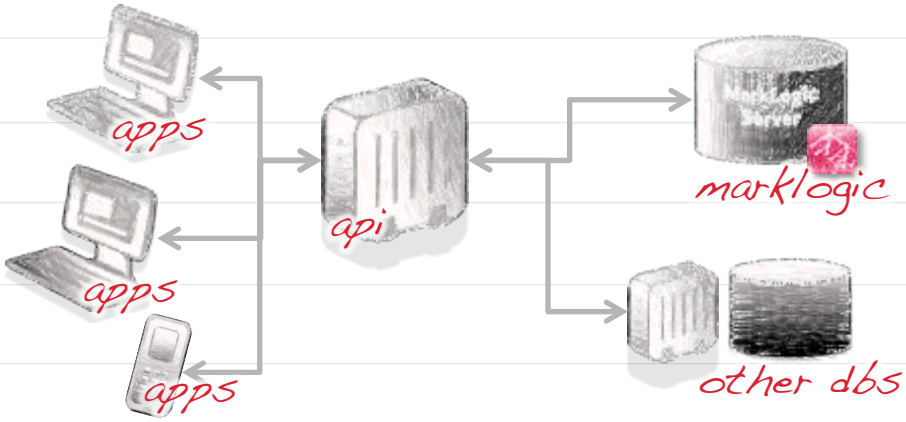


single tier:
- no boundaries between languages
- smaller stack
- king is dead!
  long live the king!

features:

- easy geospatial
- http client
- facets
- alerting
- store applications in the database
- url rewriting

!! stop exposing your database
start exposing your data



github.com/dscape/rewrite
(for rails like routing, session later on)

In MarkLogic we were thinkin'

thinkin' documents. json or xml?

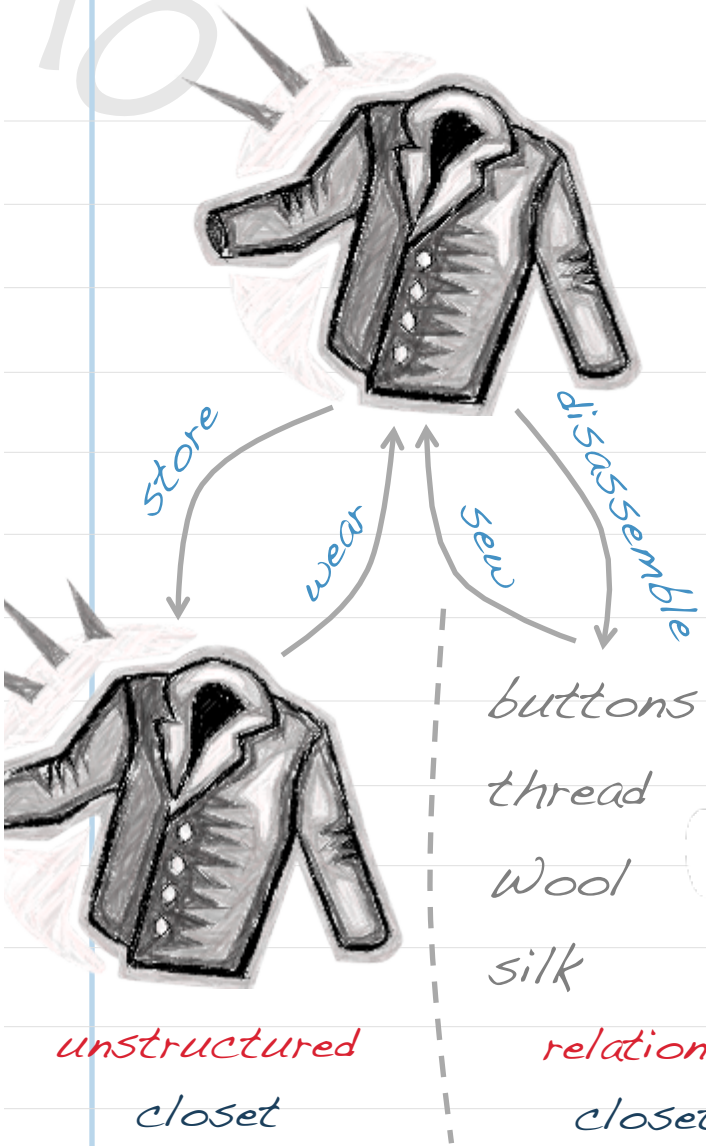
- ✓ Unstructured Information — *flexible*
- ✓ Multiple TBs to PB scale — *big*
- ✓ Sub-second response times — *fast*
- ✓ Data immediately durable — *realtime*
- ✓ Mix of complex database queries, alerting, full text search, transformations, geospatial, and real time analytics.

ever wonder what that lotus flower video-clip is all about?

love to talk about this

What is unstructured information?

unstructured?



To be great, be whole; blank verse
Exclude nothing, line
exaggerate nothing that is not you.
Be whole in everything.
Put all you are
Into the smallest thing you do.
So, in each lake, the moon shines
Because it blooms up above.

Ricardo Reis, Odes

author

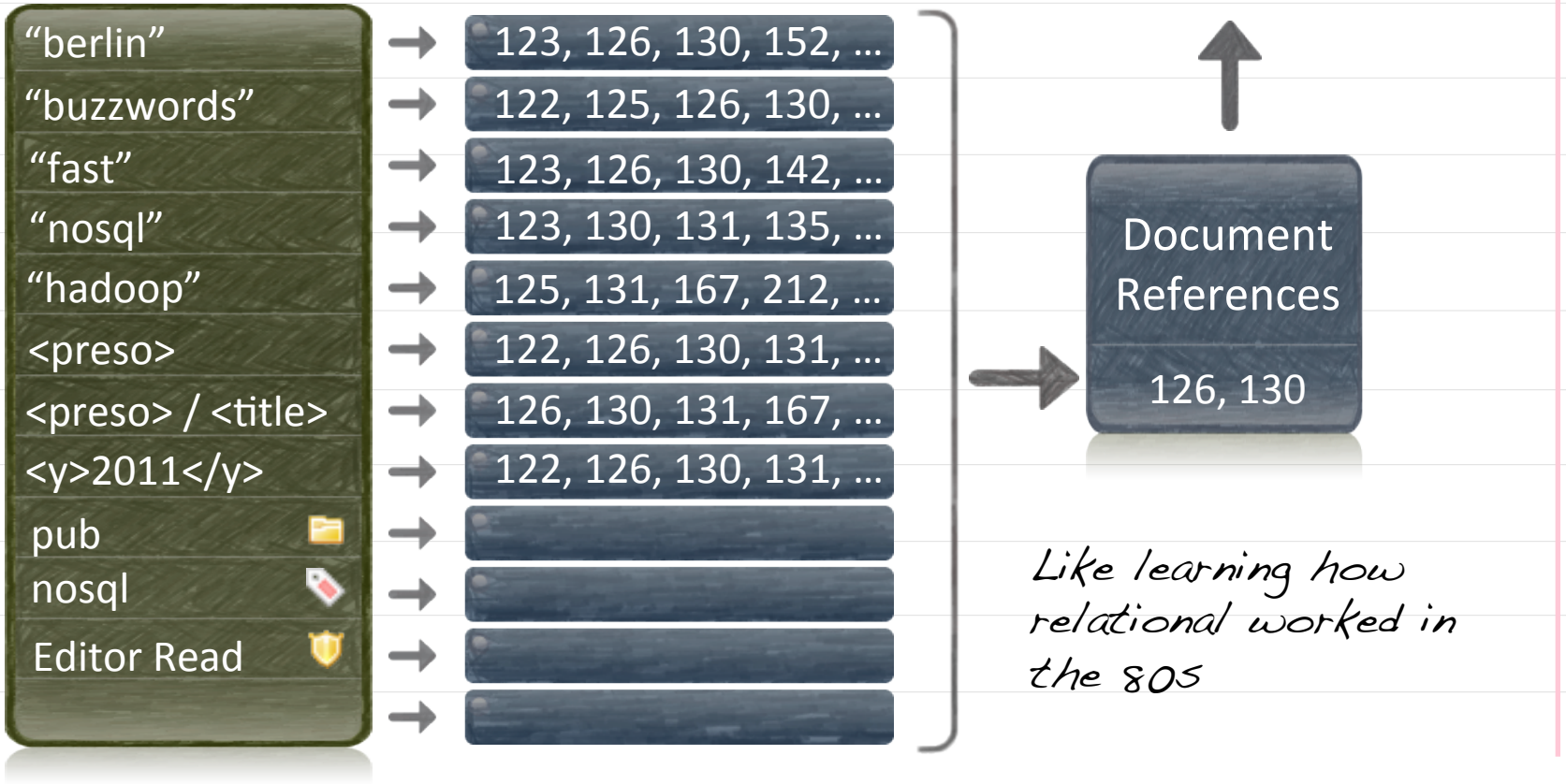
title

poem

universal index

universal index:
an inverted index that understands structure, organization, and security

slides:
http://www.slideshare.net/cbiow/mark-logic-strangeloop-2010



Like learning how relational worked in the 80s

“It is not the strongest of the species that survives, nor the most intelligent that survives. It is the one that is the most

adaptable to change.”

- Charles Darwin

ACID vs. CAP

ACID

Helps

- Easy to reason about data
- Guaranteed persistent state

Hurts

- Hard to scale horizontally
- Hard to assure high availability

Atomicity

Either all operations of the transaction are correctly executed or none is.

Consistency

Database will remain in a consistent state after the transaction commits.

Isolation

In a concurrent transactional system transactions are unaware of each other.

Durability

After a transaction completes, changes persist even if the system fails.

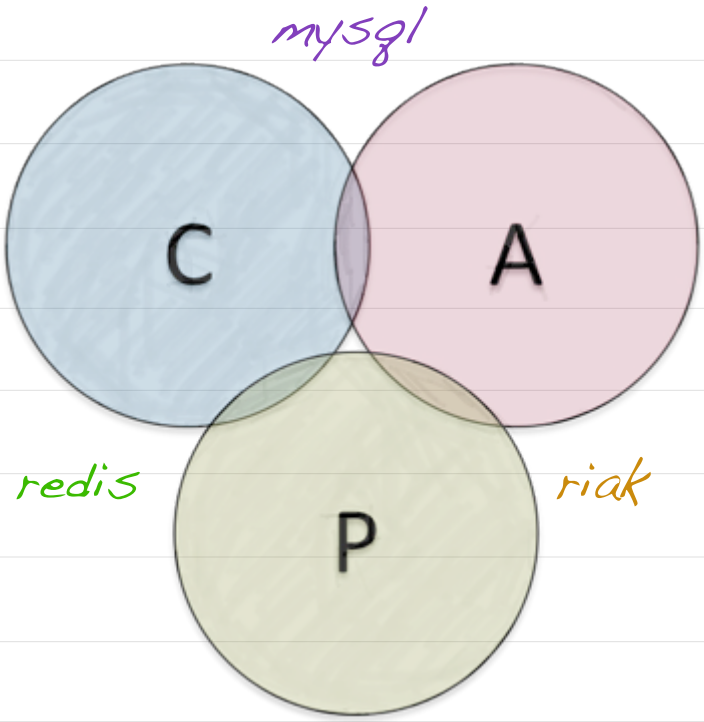
CAP

Consistency
Each client always has the same view of the data.

Availability
All clients can always read and write.

Partition Tolerance
System works well across physical network partitions.

Pick Two!



credit: blog.nahurst.com/visual-guide-to-nosql-systems

“It’s naive to explain NoSQL with CAP... for x tending to infinite it's like stating that in the world there are just 3 databases.”

- Salvatore Sanfillipo, @antirez

“There is a magic bullet!
It's called relaxing the requirements.”

- Evan Weaver, @evan

engineering is not
about science
or
imagination

it's about using
science *and*
imagination to design
solutions people need

today

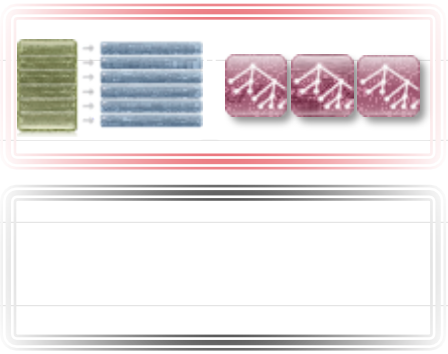
so, how do you do acid transactions at scale?

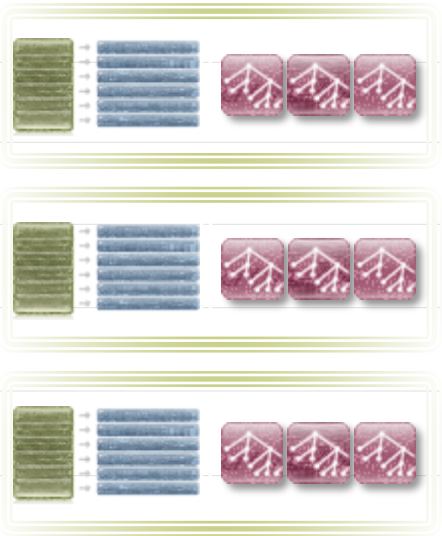
scaling an inverted index

journaled! durability ++

memory

disk





Ingestion is limited to a size where indexes are manageable

On query both in memory and on disk stands behave the same -> transparent to the developer

Means:
Fast ingestion with transactions!

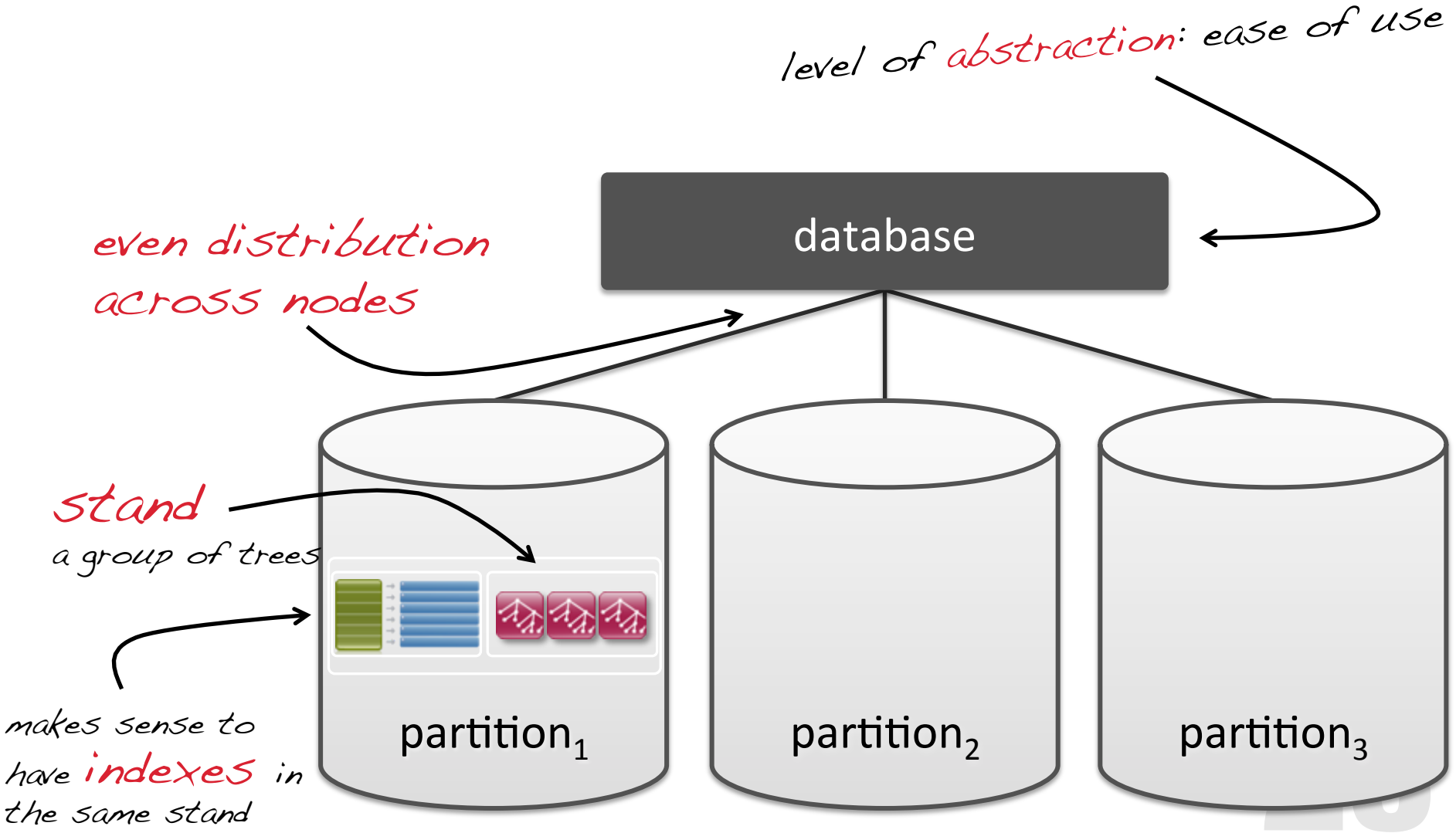
Log-Structured Merge-Tree (LSM-Tree)

Zero-latency ingestion and Indexing

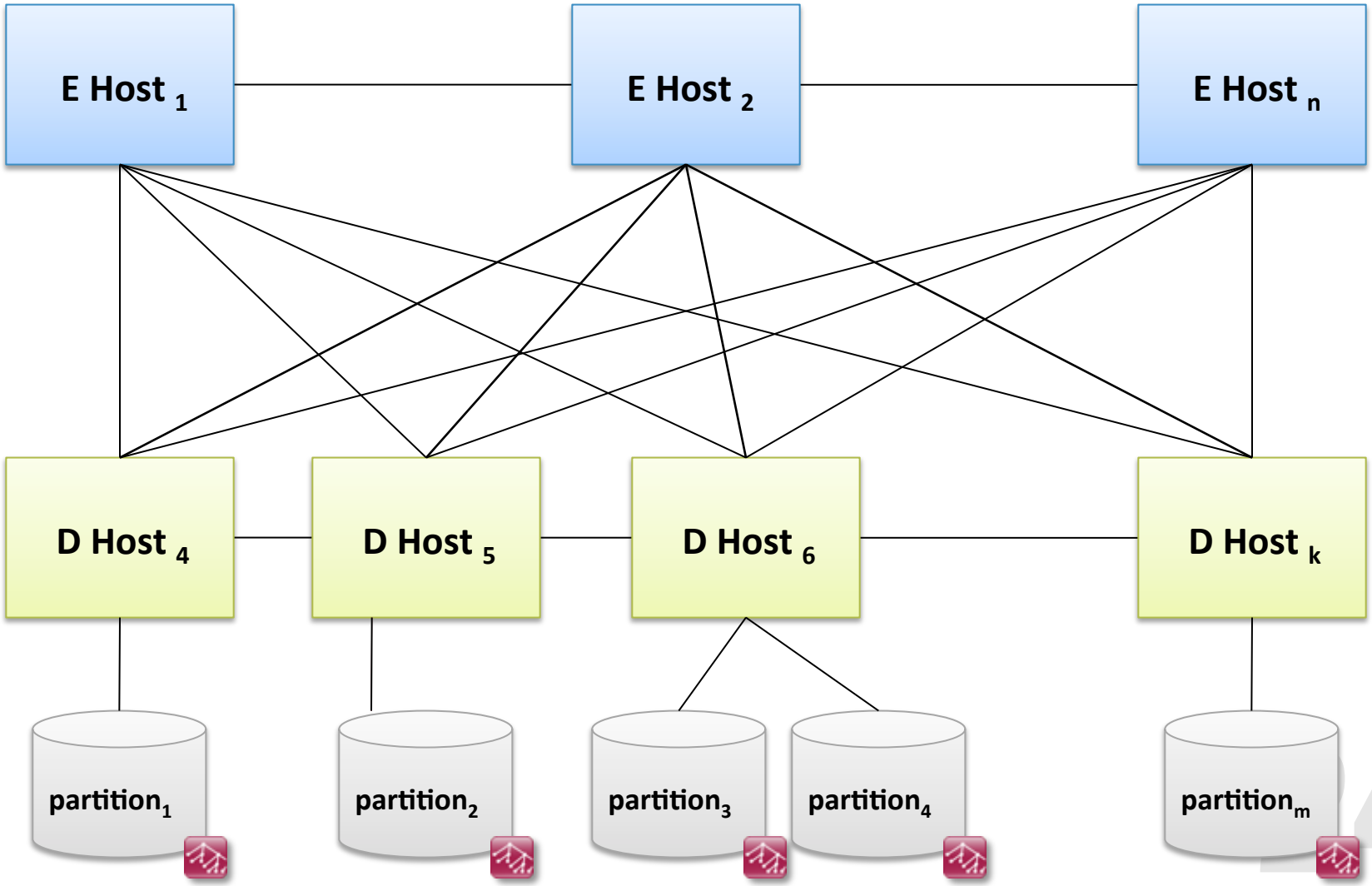
“You cannot take a car, grow it 10 times and expect to get a mining truck.”

- Ivan Pepelnjak, @ioshints

divide and conquer



shared nothing cluster



What about Locking?

MVCC

26

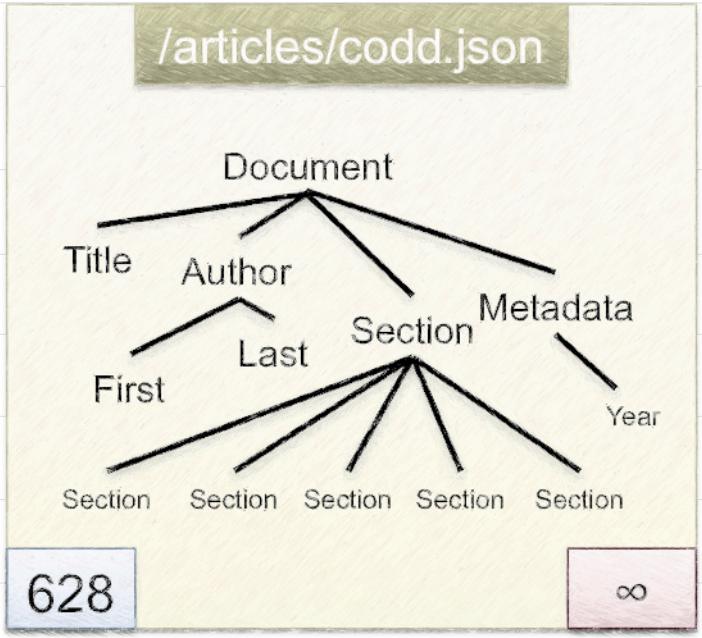
Append only database

High Throughput

- Queries don't require locks
- Queries and Updates do not conflict

ACID

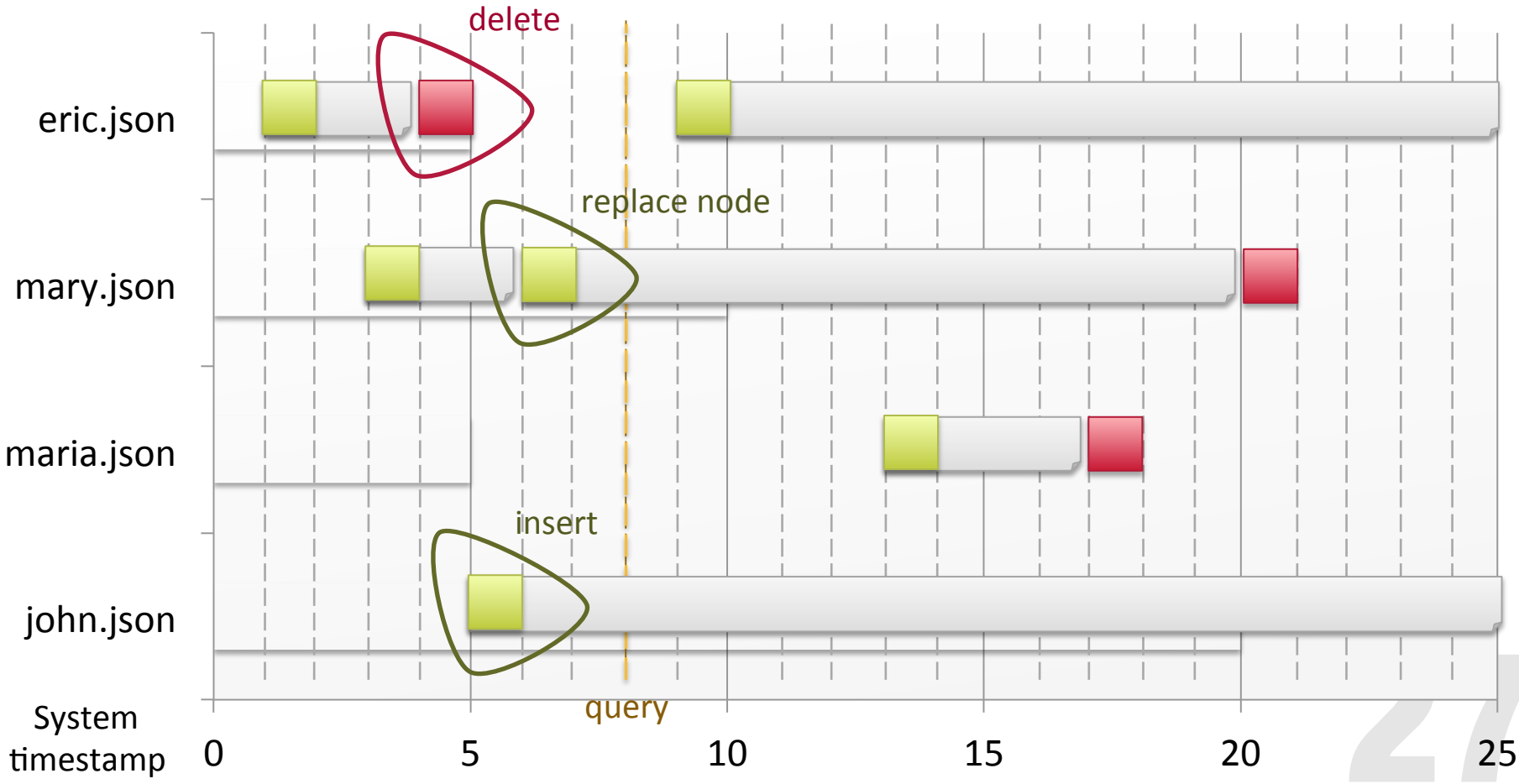
- Cluster consistency: 2-phase commit



Time Machine?

mvcc

queries never lock!



How does the 2-phase commit work?

Journal A
Insert fragment 123 “/foo.json”
Distributed Begin, A added(123), B added(234)
Commit, timestamp 1, added (123)
Distributed End
Insert fragment 345 “/foo.json”
Commit, timestamp 2, added (345), deleted (123)
Distributed Begin, A added(123), B added(234)
Commit, timestamp 3, deleted (34567)
Distributed End

Journal B
Insert fragment 234 “/bar.json”
Prepare
Commit, added (123)
Prepare
Commit, deleted (234)

doesn't lock documents, locks uris

insert “/foo.json”, { “foo”: “” }
delete “/foo.json”
insert “/bar.json”, { “bar”: “” }
delete “/bar.json”
“stuff”

Shard A

ID	✔	✗	URI
123	1	2	/foo.json
345	2	3	/foo.json

Shard B

ID	✔	✗	URI
234	1	3	/bar.json

What about Availability?


developer.marklogic.com
THE power of
IMAGINATION
makes us infinite.
awesome project btw...
365q.ca
MarkLogic

“You have database problem. You research blog and HN. You start use NoSQL product. Now you not know anymore if you have problem.”

- Devops BORAT, @devops_borat

Questions?

@dscape