NoSQL Databases
To SQL or not to SQL

Nick Dimiduk (@xefyr)
Committer, PMC
http://n10k.com

2015-12-02

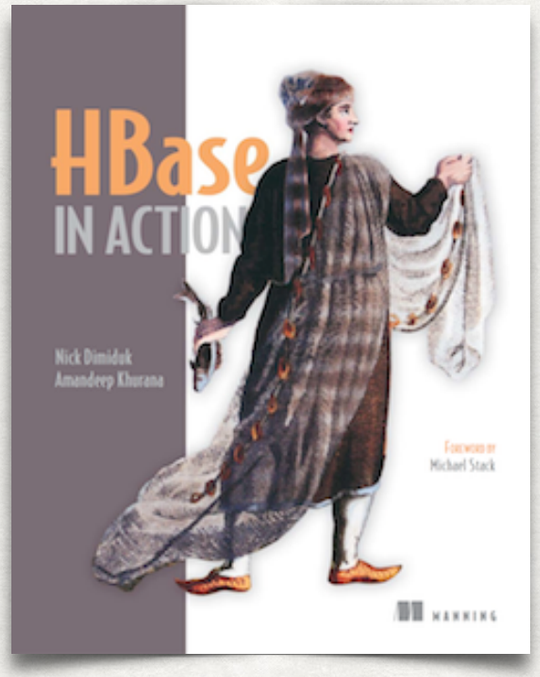
Apache HBase (@hbase)
http://hbase.apache.org

Apache Phoenix (@ApachePhoenix)
http://phoenix.apache.org

APACHE HBASE
CC BY SA
APACHE PHOENIX

About Me

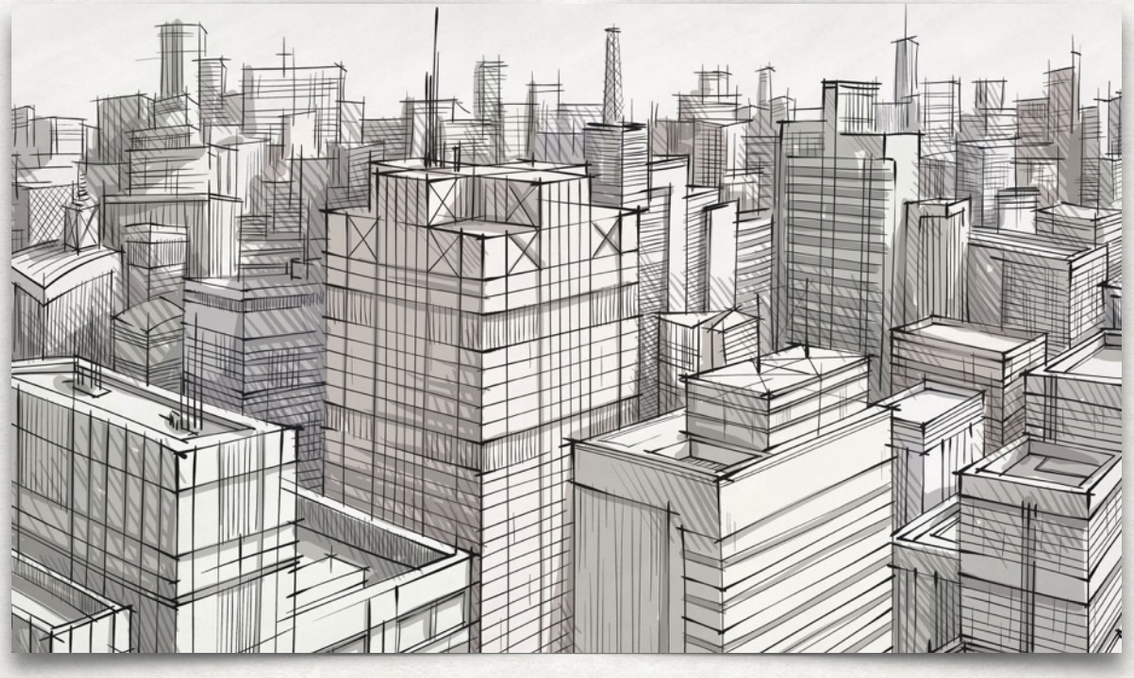
- Working with NoSQL and Hadoop since 2008
- Committer and PMC Member
  - Apache HBase
  - Apache Phoenix
- Release manager for HBase 1.1 releases



CC BY SA

Agenda

- Part 1: What is NoSQL?
- Part 2: Data at Scale
- Part 3: NoSQL in Depth with Apache HBase
- Part 4: Bring SQL Back with Apache Phoenix



APACHE HBASE

CC BY SA

APACHE PHOENIX

Part 1

What is NoSQL?

APACHE HBASE

CC BY SA

APACHE PHOENIX

What is NoSQL?

"... a mechanism for storage and retrieval of data that is modeled in means **other than the tabular relations used in relational databases**."

https://en.wikipedia.org/wiki/NoSQL

WIKIPEDIA
The Free Encyclopedia

N★SQL

"... often more characteristics apply such as: **schema-free, easy replication support, simple API, eventual consistency / BASE (not ACID), a huge amount of data**."

http://nosql-database.org

CC BY SA

What is NoSQL?

cc BY SA

What is NoSQL?

CC BY SA

The Right Tool For The Job

http://www.flickr.com/photos/zakh/337938459/

cc BY SA

What Kinds Of Other Tools?

- Difficult to model with relational algebra
  - graphs, timeseries
- Performance gaps in RDBMS implementation
  - full-text search
- "Inside-out" requirements
  - multi-master, distributed, offline applications
- Ease of development
  - JavaScript/JSON/HTTP vs SQL/ODBC
- In-memory latency requirements
- Some **other** requirement more important than relationships/constraints

APACHE HBASE
CC BY SA
APACHE PHOENIX

What NoSQL is Not

- NoSQL is not **a standard**
  - there is no ANSI NoSQL-2014 definition
  - API, not SQL; different for each tool
- NoSQL is not **THE new database**
  - specialized tools for specific purposes
  - results in "polyglot storage"
- NoSQL is not **a silver bullet ***
  - how to integrate into existing data workflows?
  - are the tradeoffs acceptable?

* http://www.cs.nott.ac.uk/~pszcah/G51ISS/Documents/NoSilverBullet.html *

CC BY SA

Key-Value Stores

CC BY SA

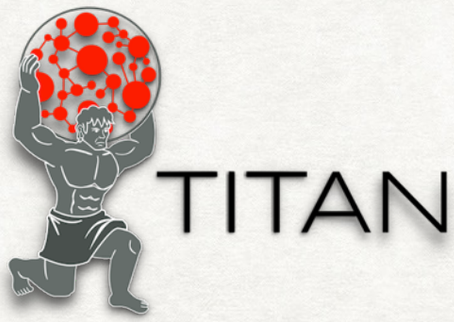
Document Stores

cc BY SA

Bigtable Databases

cc BY SA

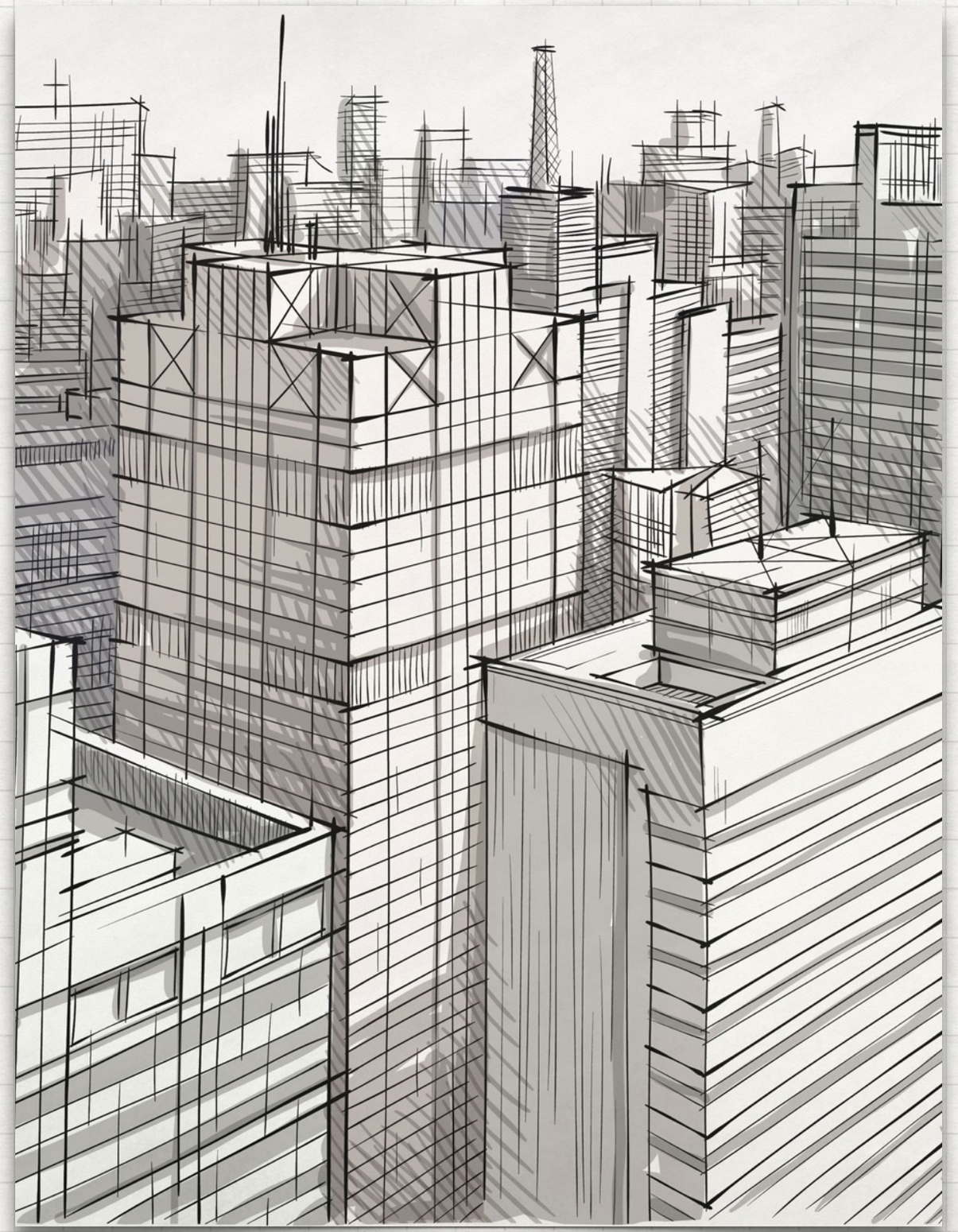
Graph Databases



cc BY SA

Part 2

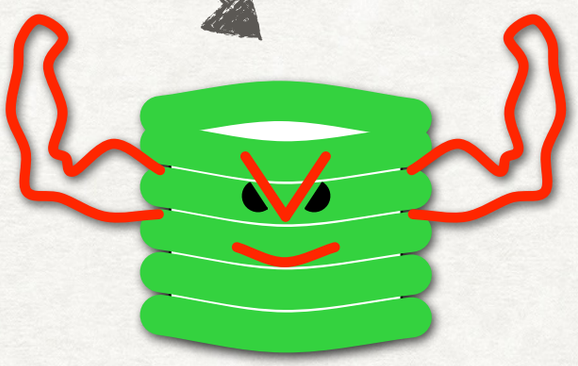
Data at Scale

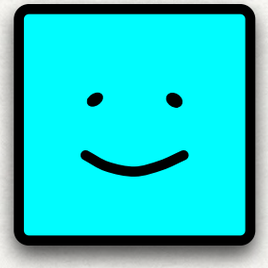


CC BY SA

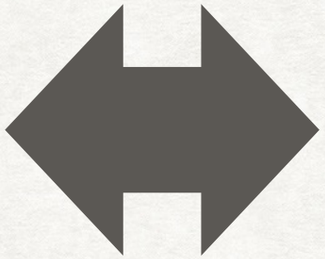
Typical Application

Database Server





App Server

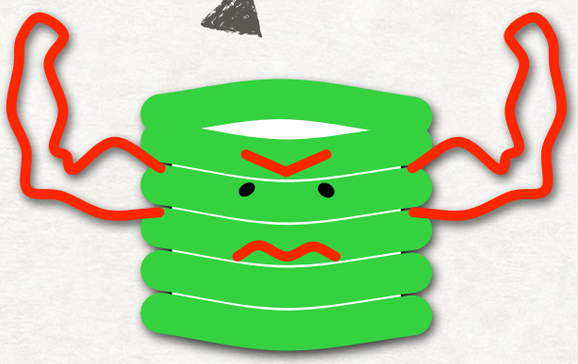


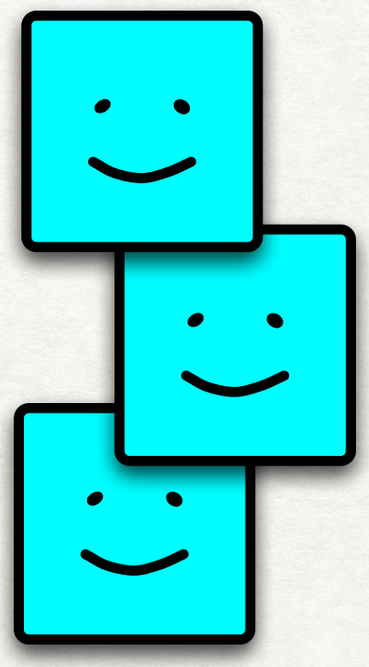
Village People

CC BY SA

Growing Pains

Database Server





App Servers

API

Villages of People

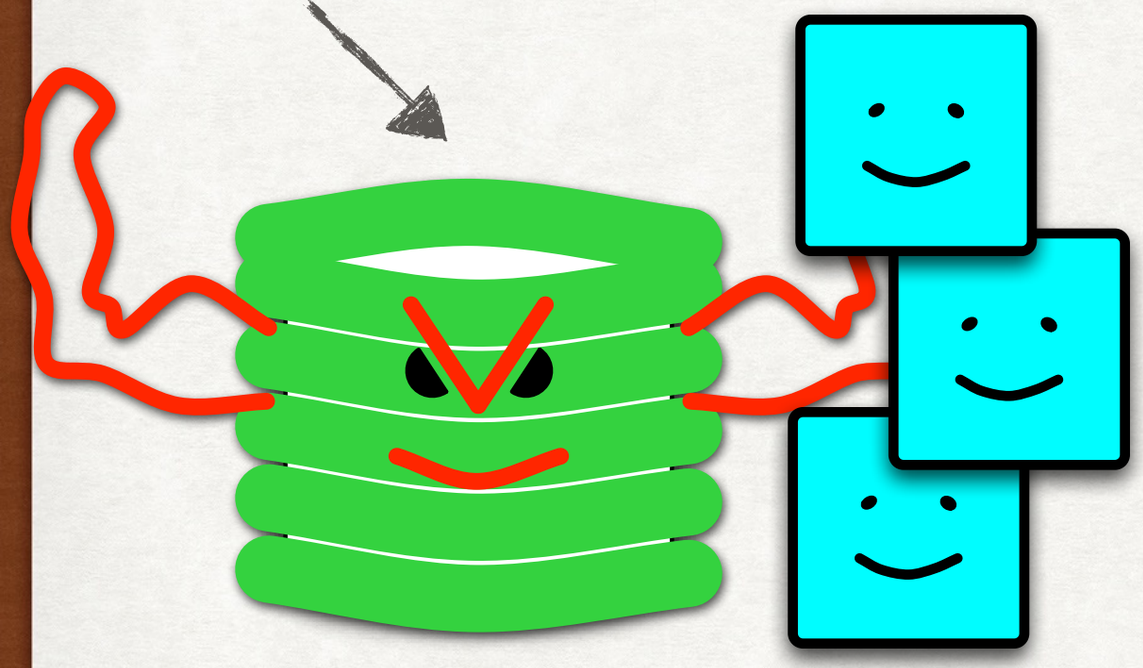
APACHE HBASE

cc BY SA

APACHE PHOENIX

Vertical Scaling

BIGGER
Database Server



App Servers

API

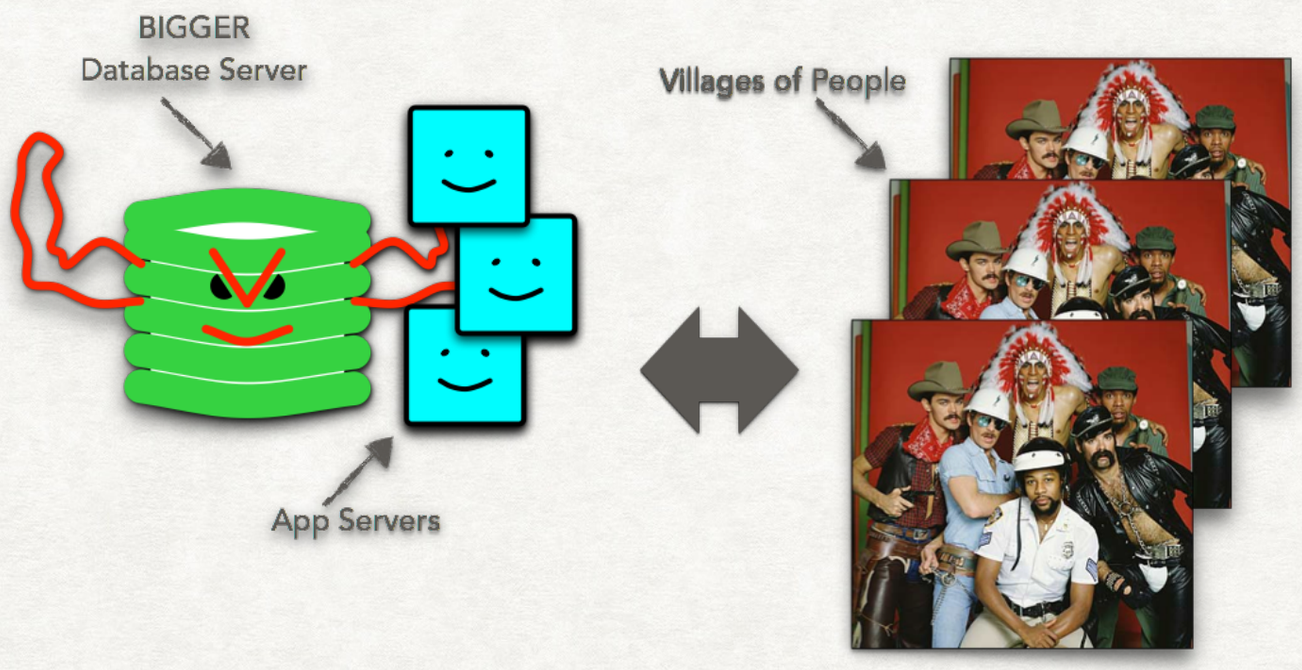
Villages of People

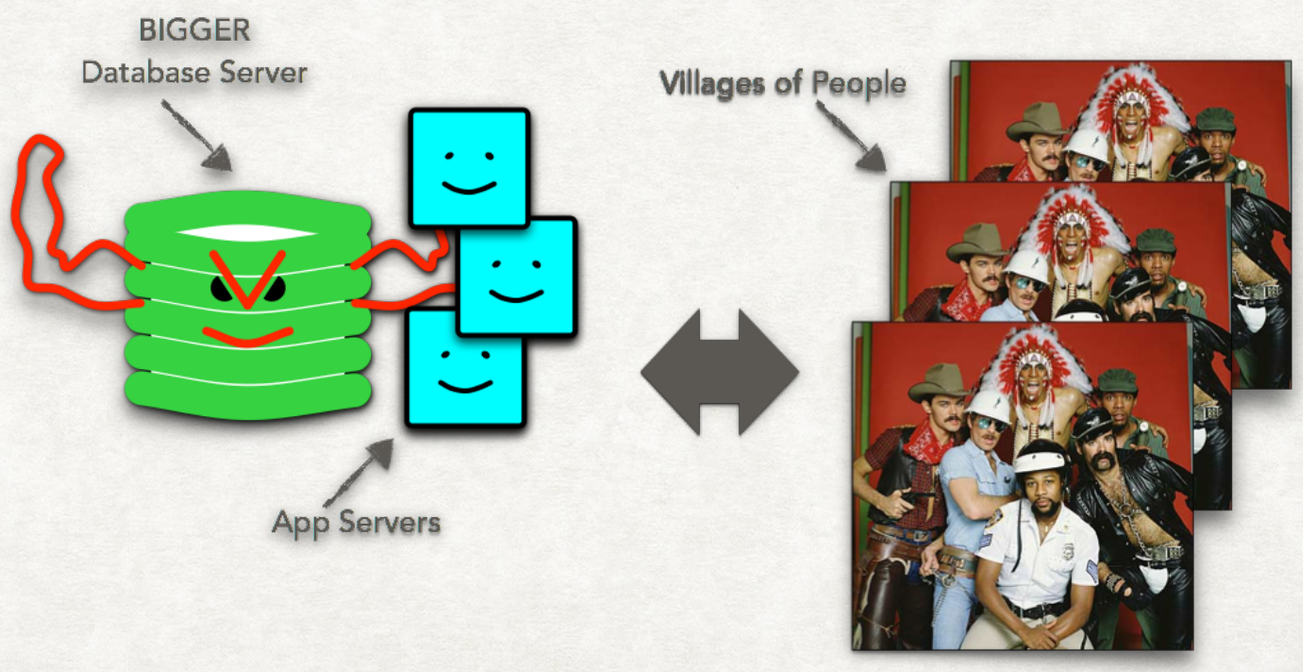
APACHE HBASE

CC BY SA

APACHE PHOENIX

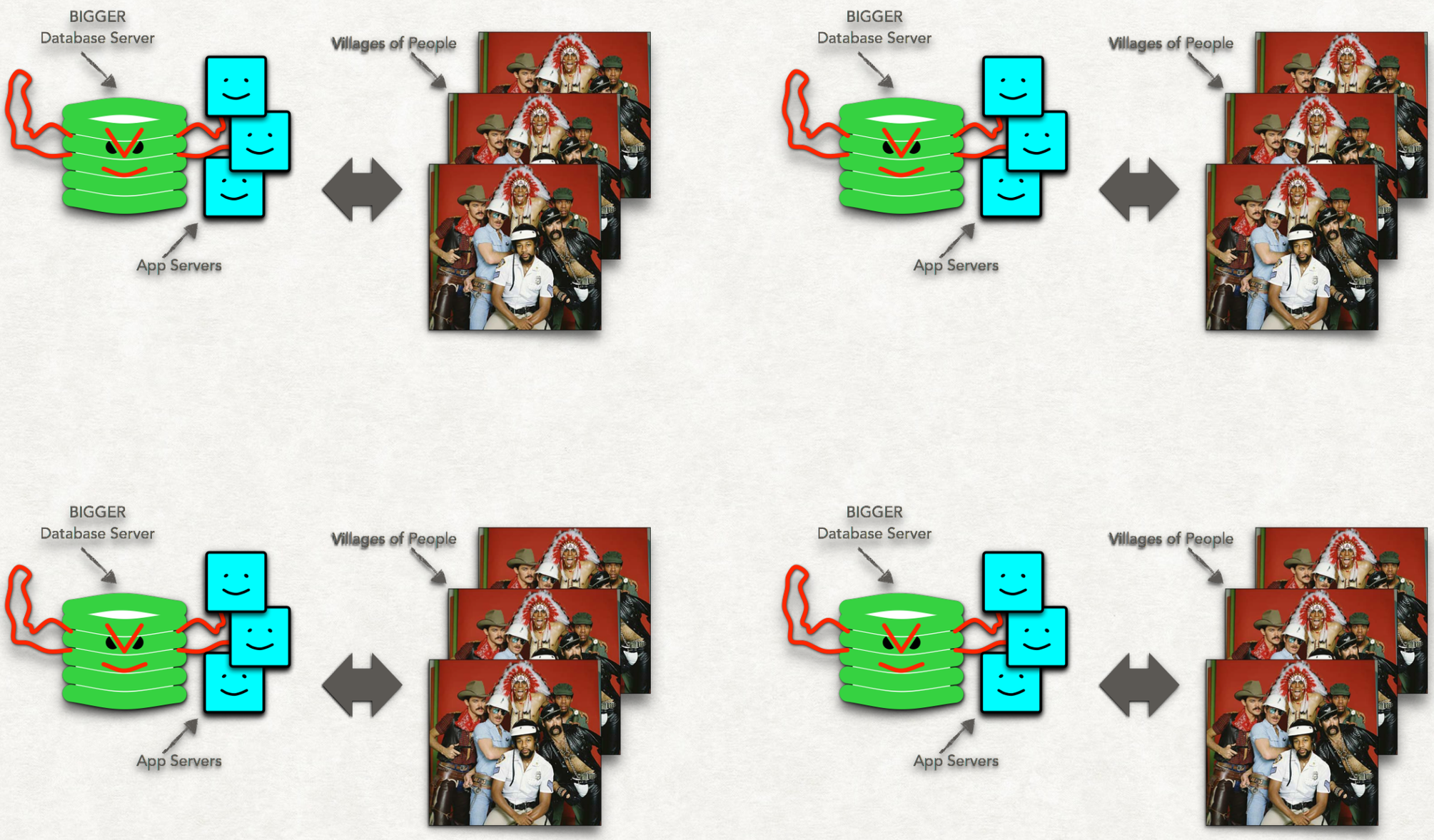
Application Partitioning





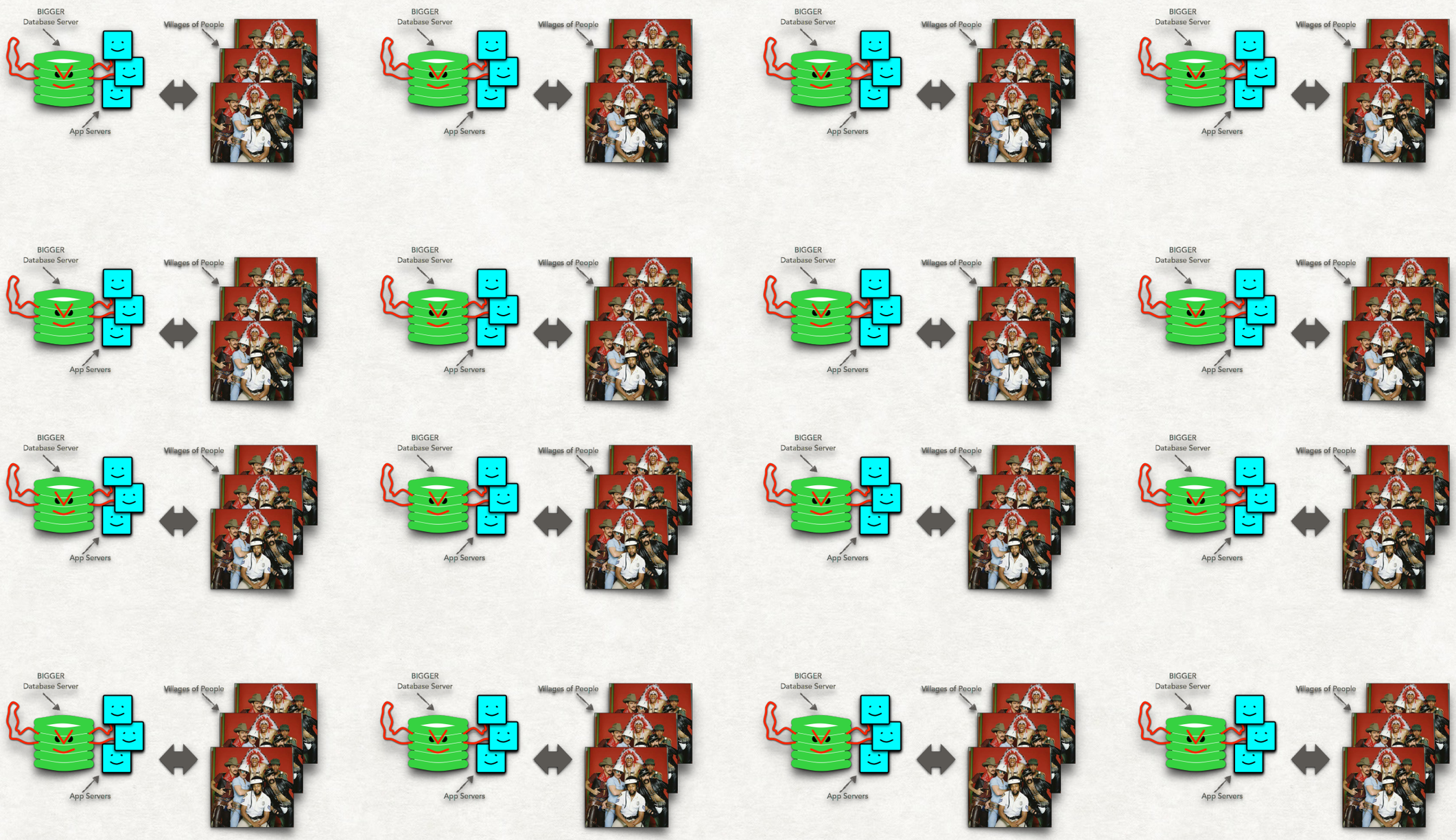
CC BY SA

Application Partitioning



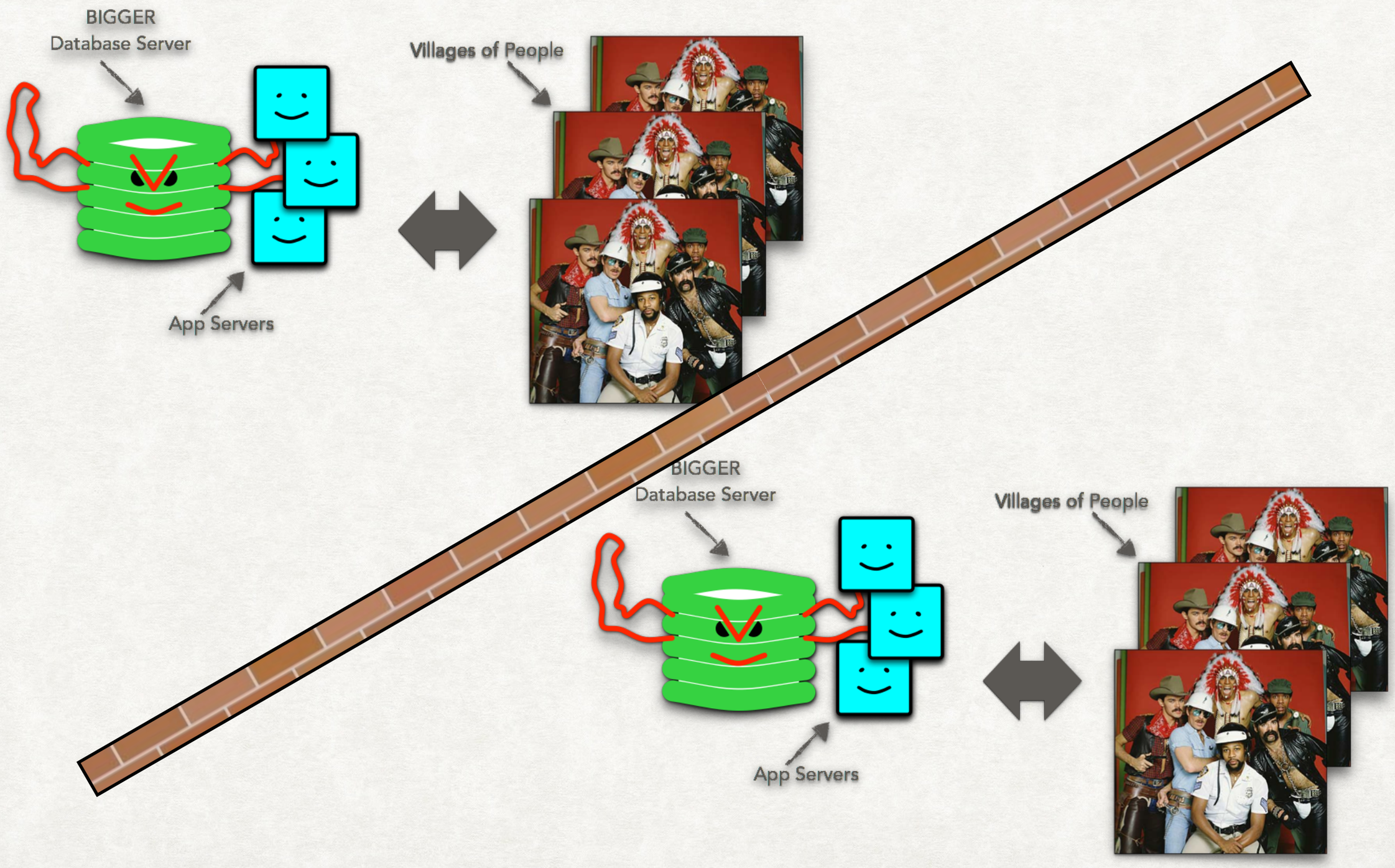
CC BY SA

Application Partitioning



cc BY SA

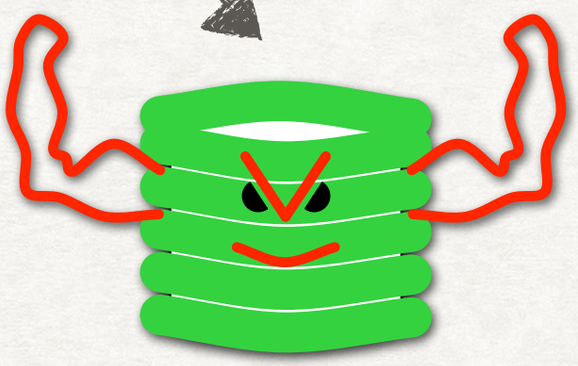
Physical Partitioning

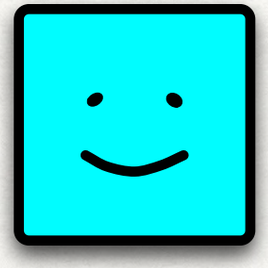


CC BY SA

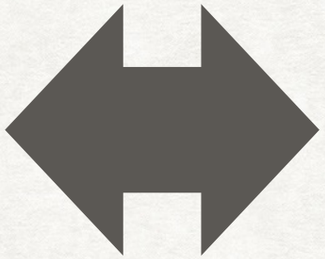
Typical Application

Database Server





App Server

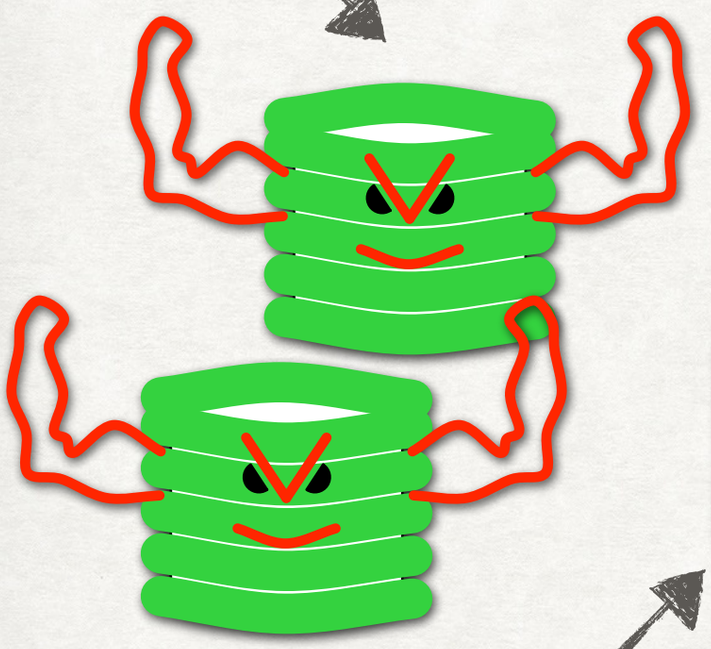


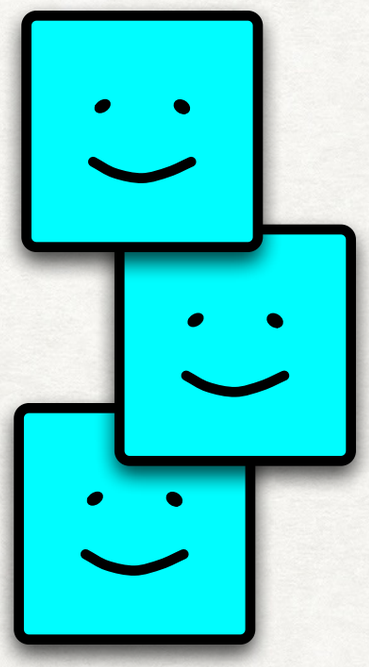
Village People

CC BY SA

Horizontal Scaling

Database Servers





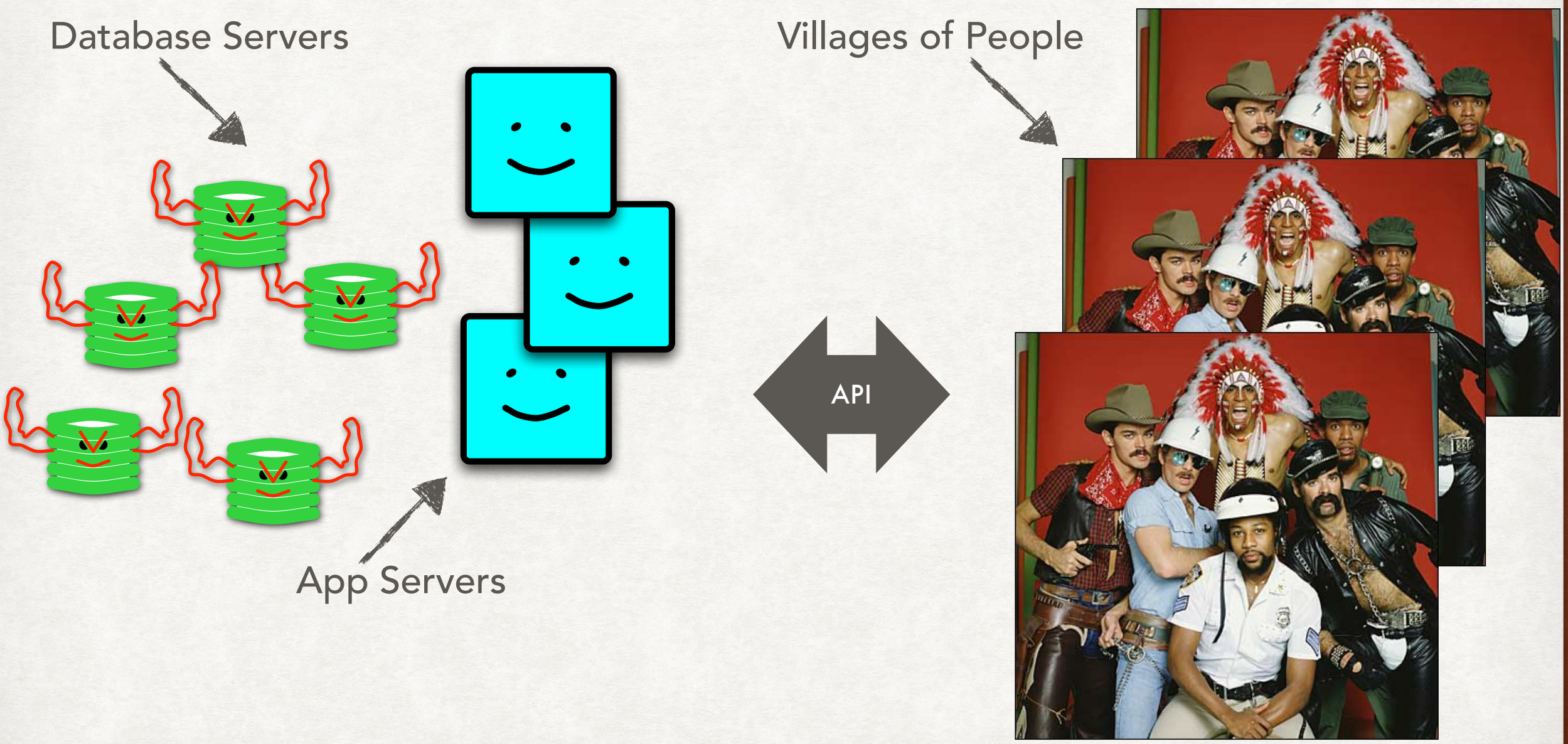
App Servers

API

Villages of People

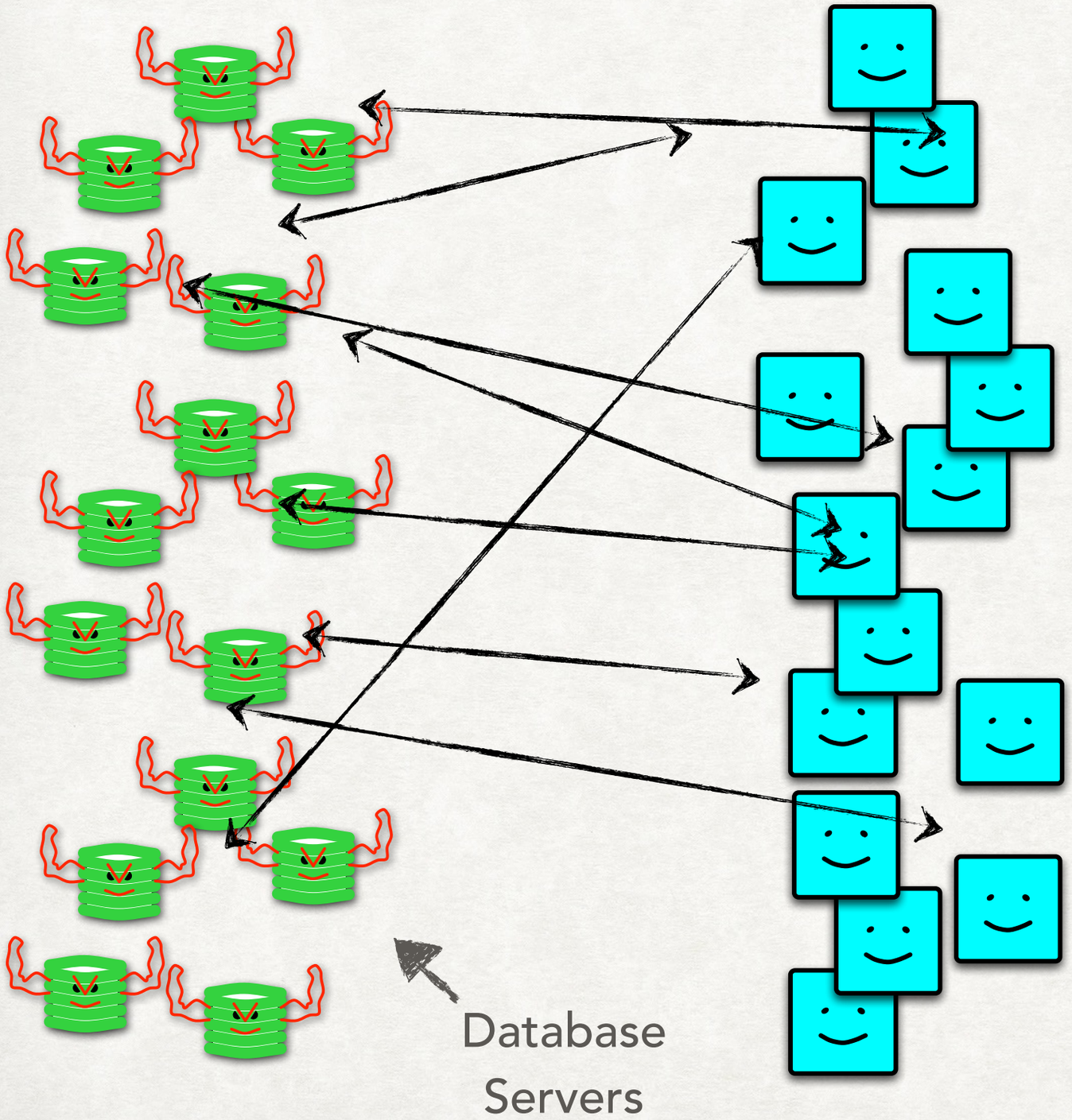
CC BY SA

Horizontal Scaling



CC BY SA

Horizontal Scaling

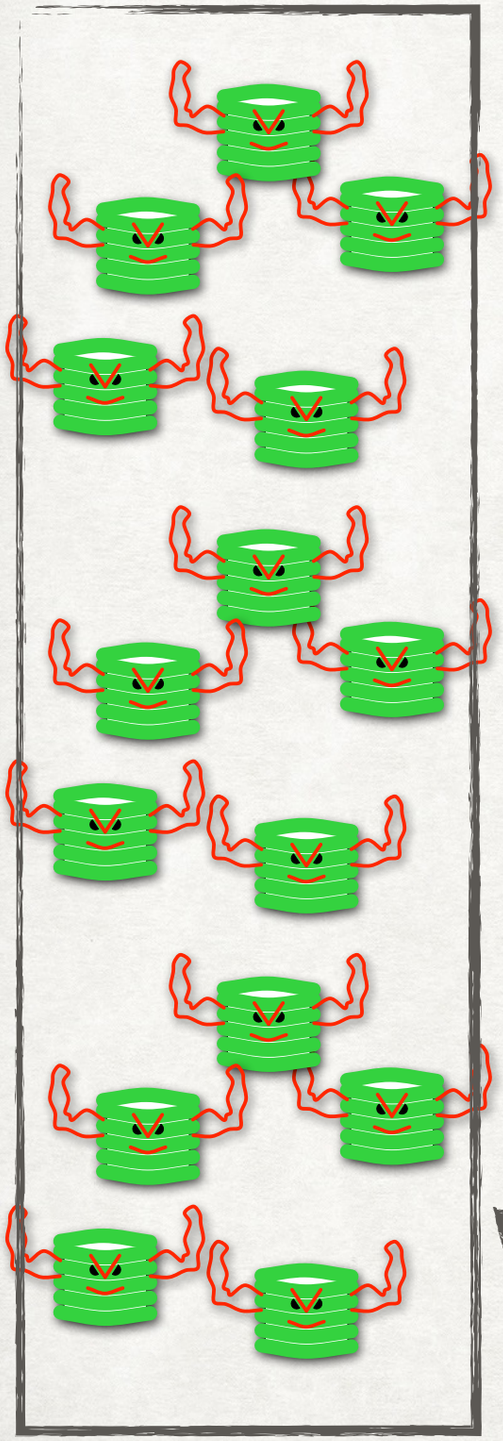


API

Villages of People

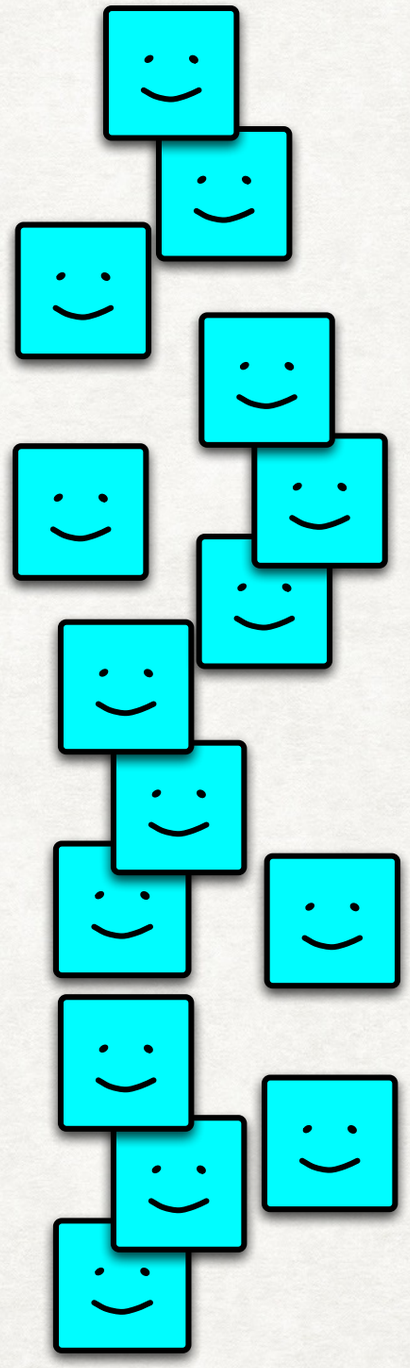
APACHE HBASE
BY SA
APACHE PHOENIX

Distributed Database



Database Cluster

API



API

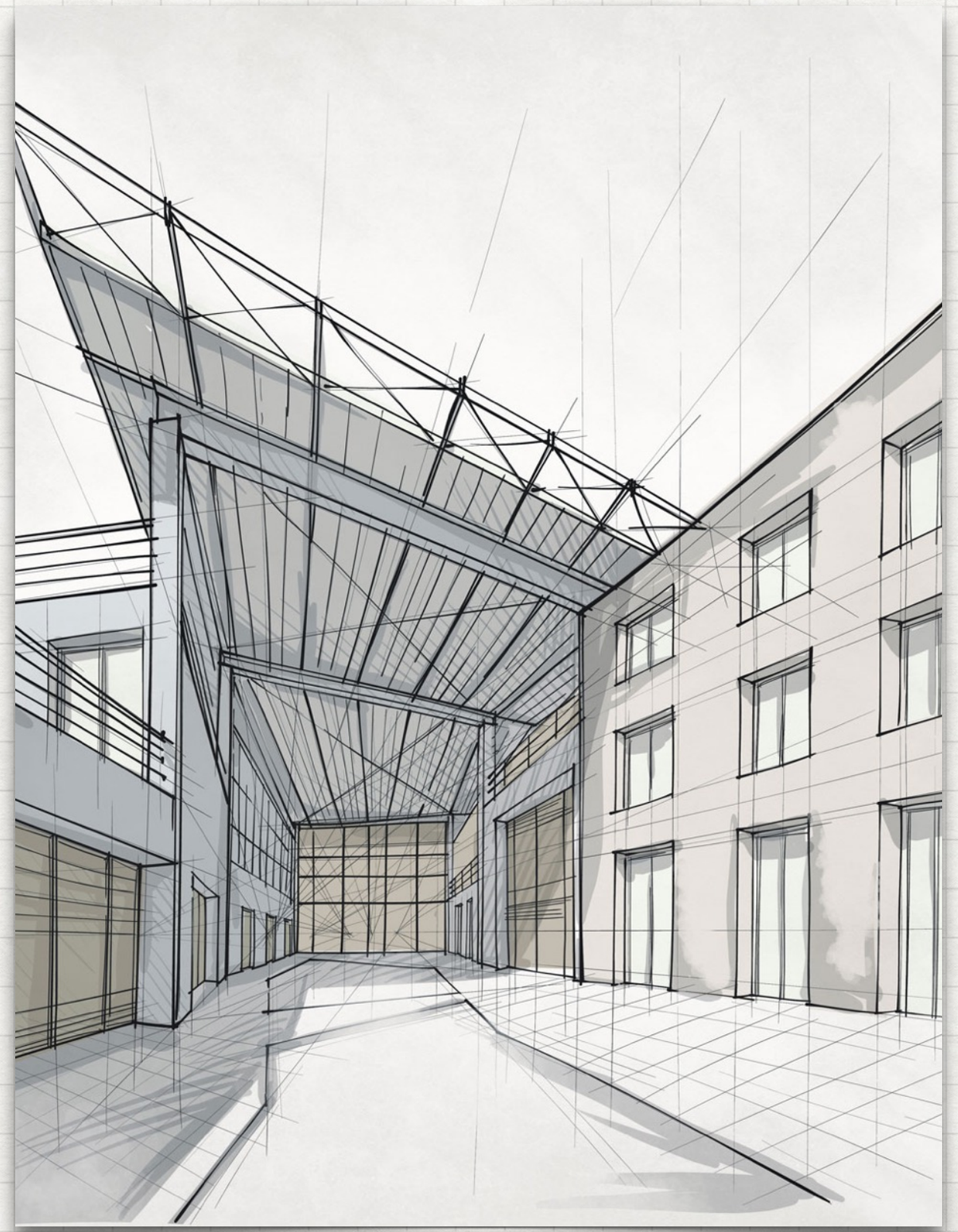
App Servers

Villages of People

CC BY SA

A Short Detour:

Computer Science!



APACHE HBASE

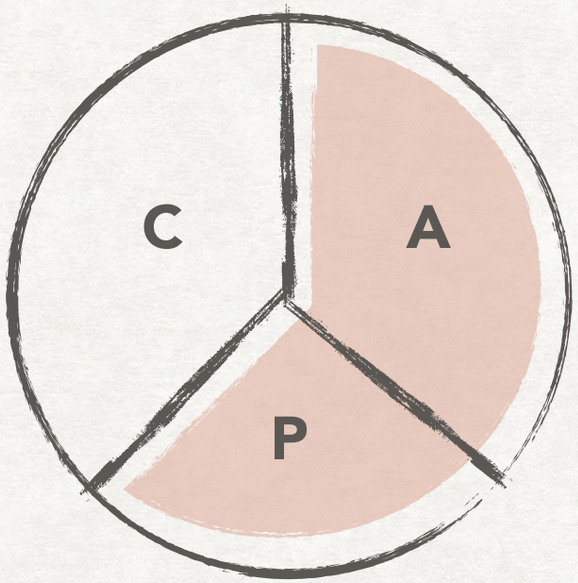
CC BY SA

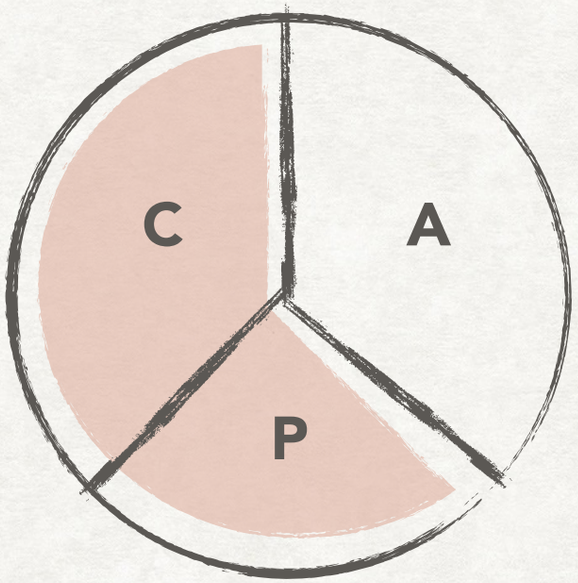
APACHE PHOENIX

Brewer's CAP Conjecture/Theorem

For strict definitions of **C**onsistency, **A**vailability, Tolerance to Network **P**artitions:

- Define expectations when Bad Things[tm] happen
- Distributed storage focus on either CP or AP





http://www.cs.berkeley.edu/~brewer/cs262b-2004/PODC-keynote.pdf
http://www.infoq.com/articles/cap-twelve-years-later-how-the-rules-have-changed

APACHE HBASE
CC BY SA
APACHE PHOENIX

How we Distribute

Quorum Operations

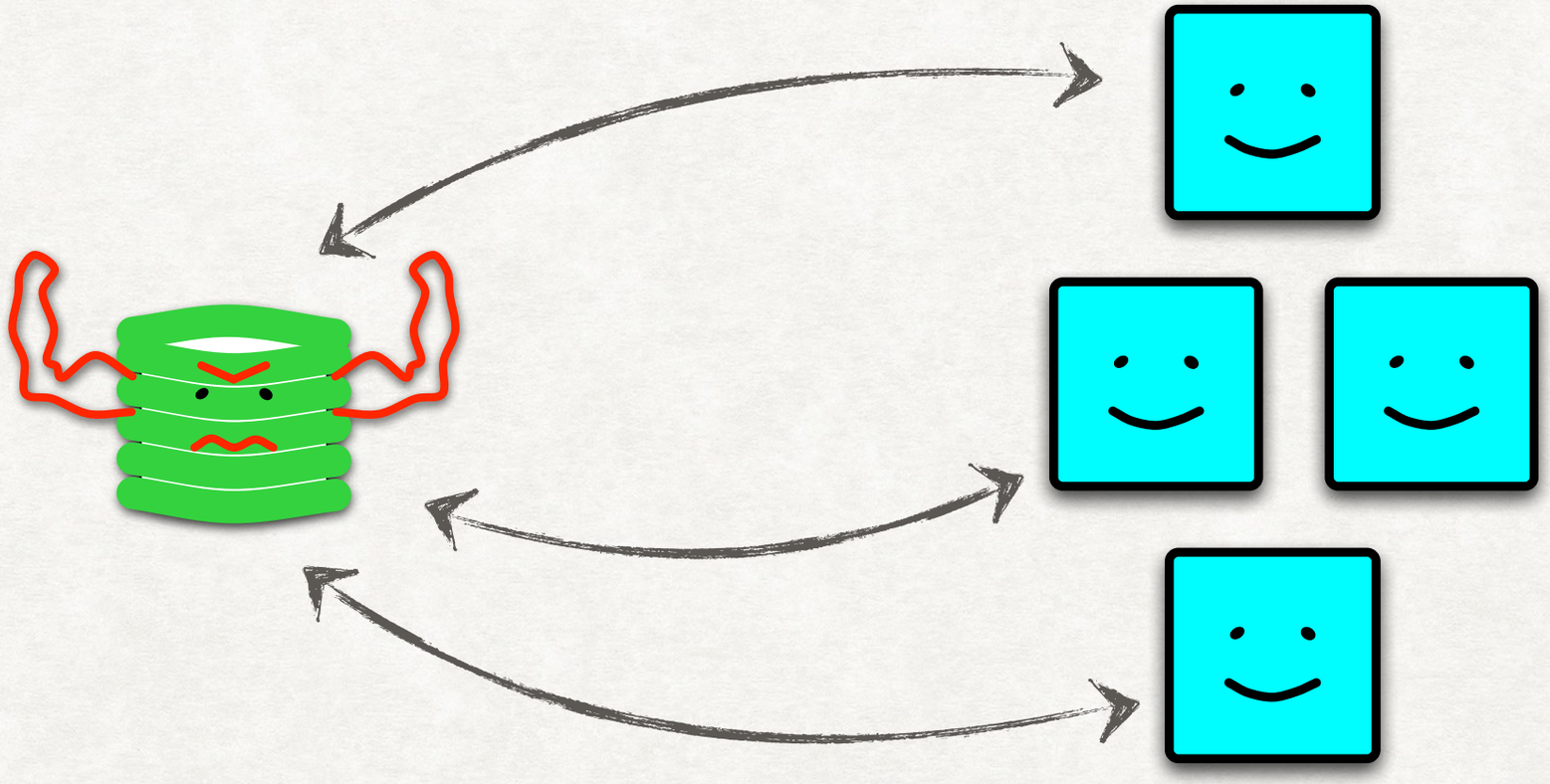
- Group of members agree on every change
- Odd number of members, majority wins
- Member failure transparent to clients
- Agreement latency increases with quorum size

Federated Responsibility

- Regions operate independently
- Some "oracle" defines region boundaries
- Single member is authority within region
- Authority failure requires recovery

CC BY SA

Growing Pains



cc BY SA

Dimensions of Growing Pains

Scaling user load

- problem: increasing service load to 1000's or millions of clients
- solution: replication

Scaling data volume

- problem: increasing amount of data under management
- solution: partitioning

Data durability

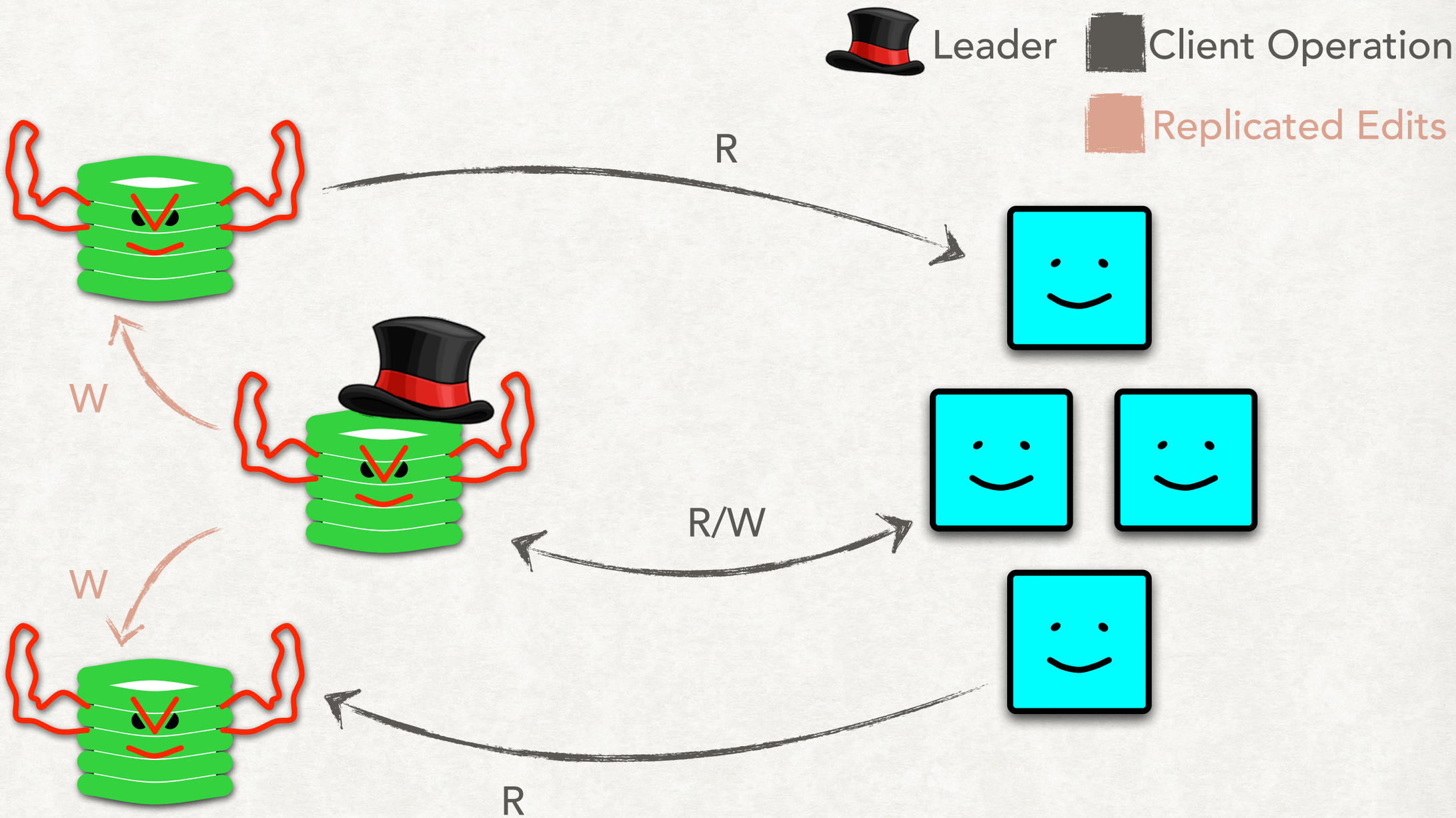
- problem: retaining data in the presence of member failure
- solution: replication

Service availability

- problem: serving requests in the presence of member failure
- solution: replication, partitioning

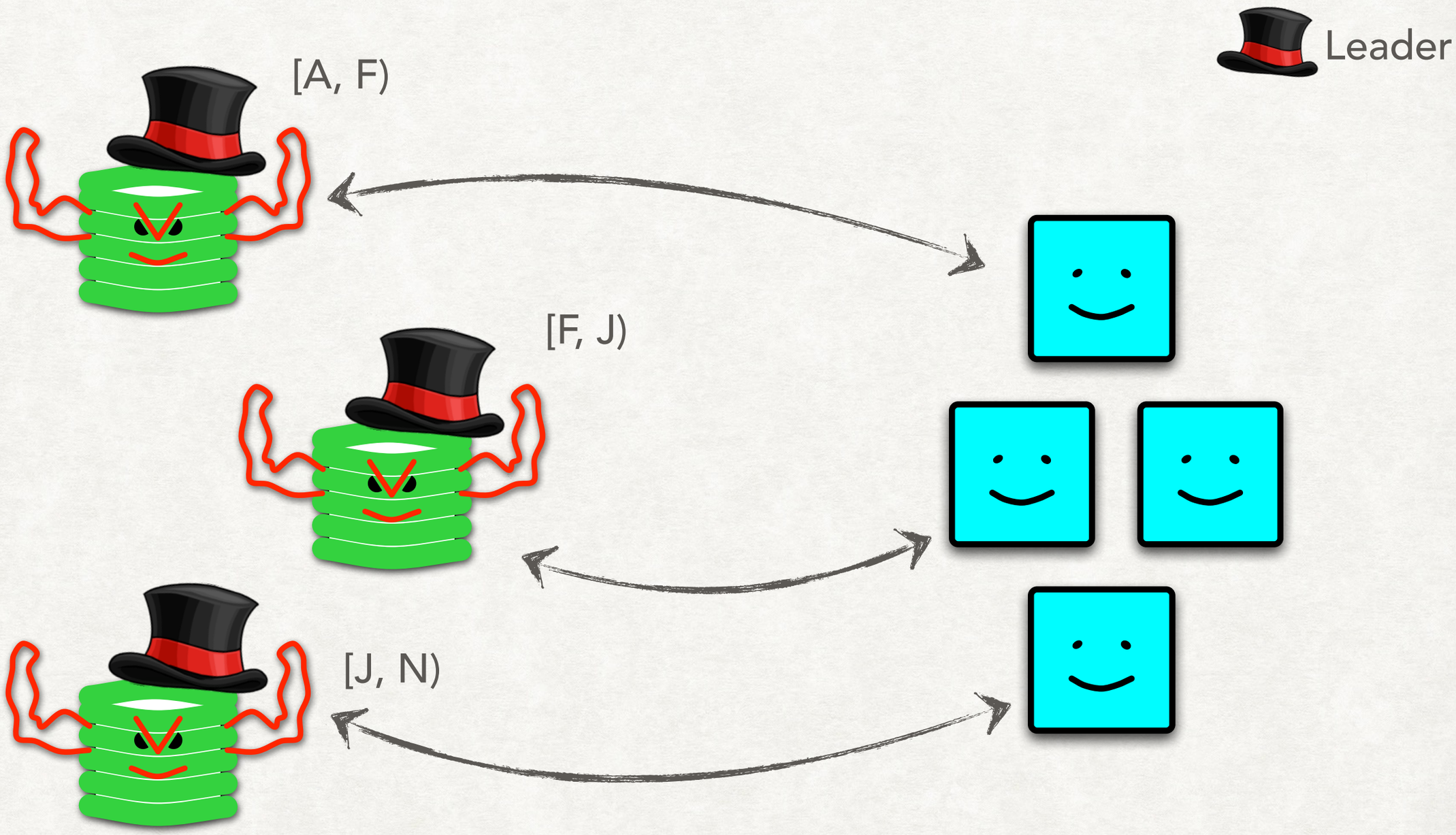
CC BY SA

Scaling User Load with Replication



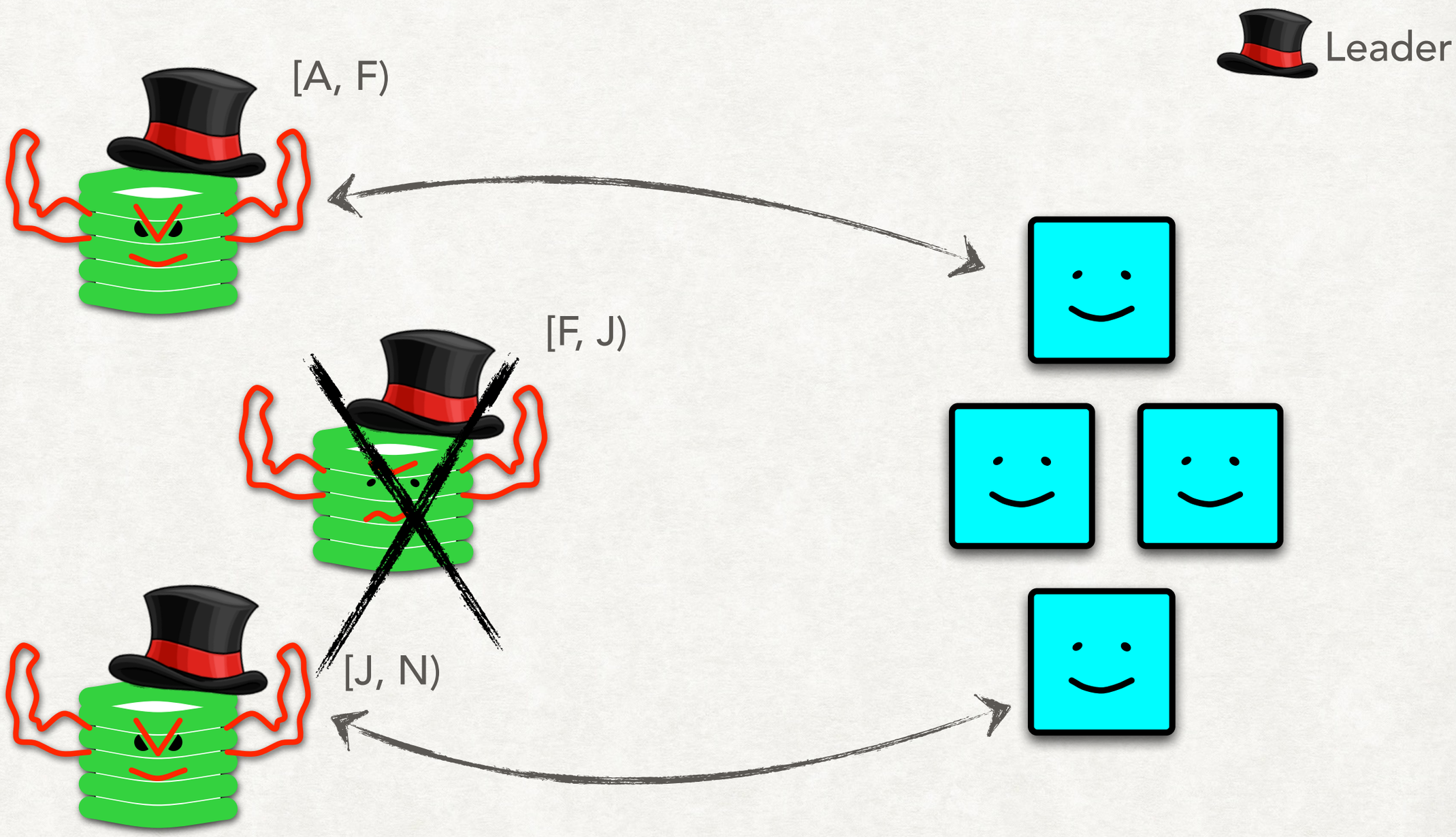
APACHE HBASE
CC BY SA
APACHE PHOENIX

Scaling Data Volume with Partitioning



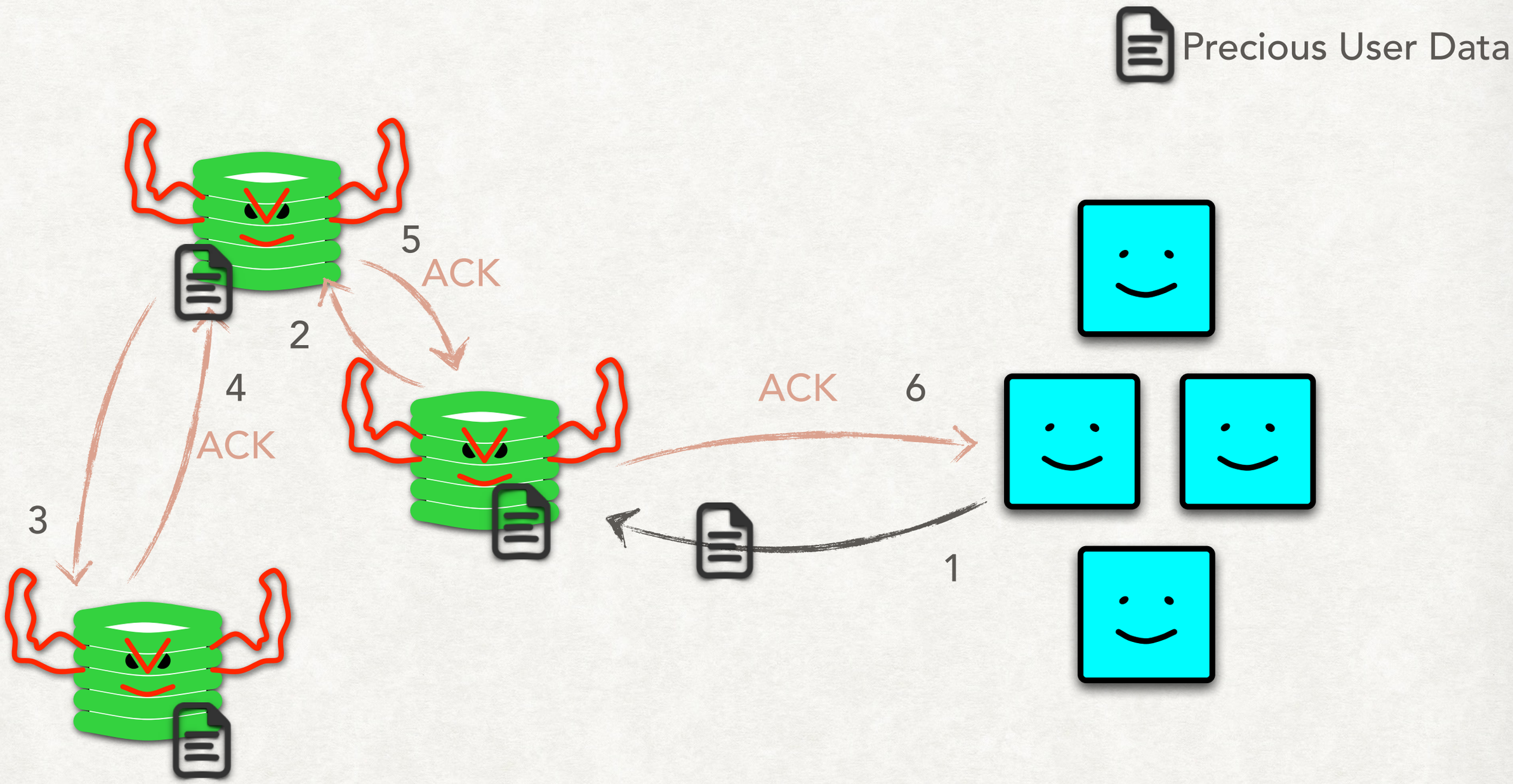
CC BY SA

Availability with Partitioning



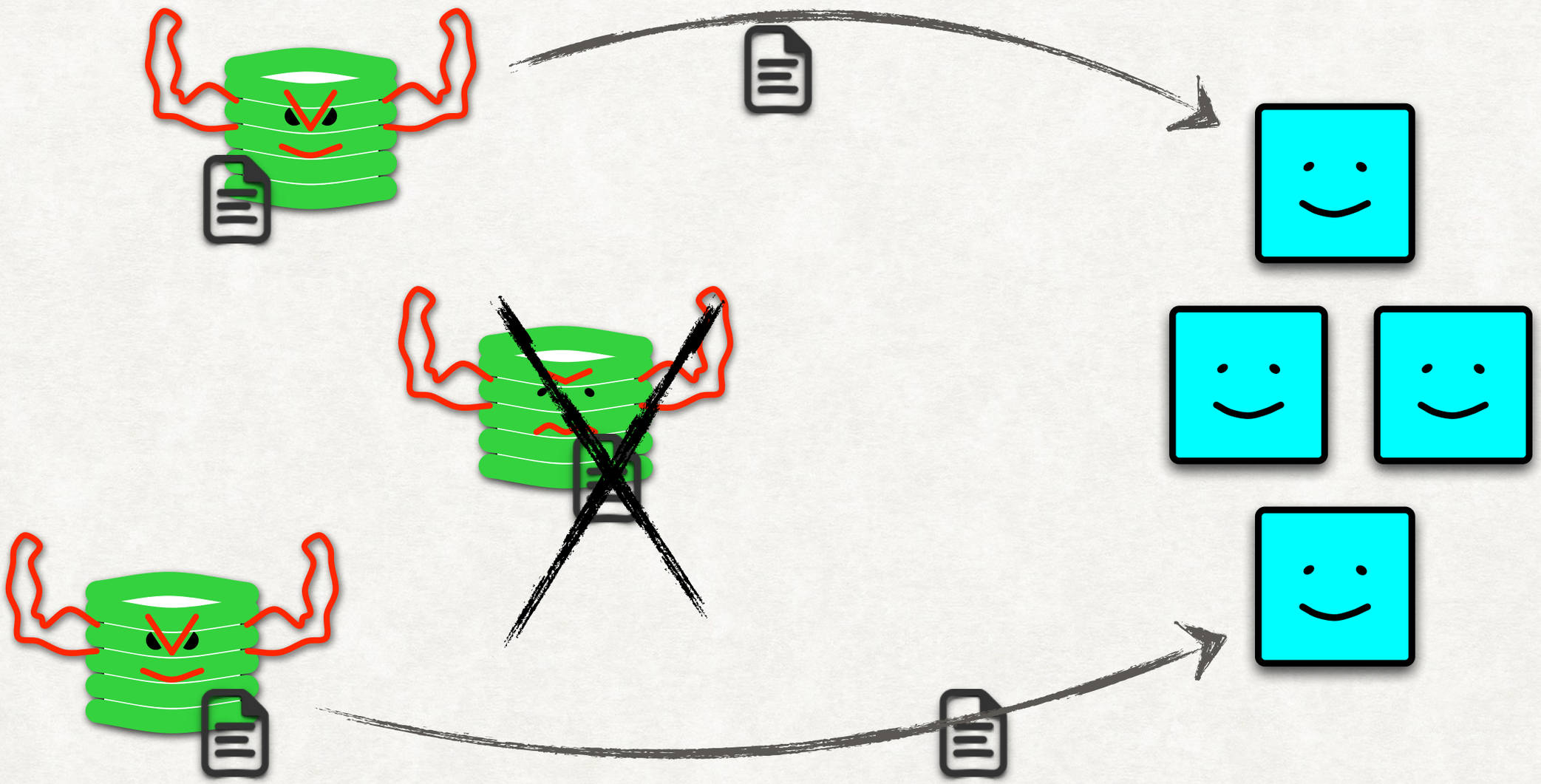
APACHE HBASE
CC BY SA
APACHE PHOENIX

Durability with Replication



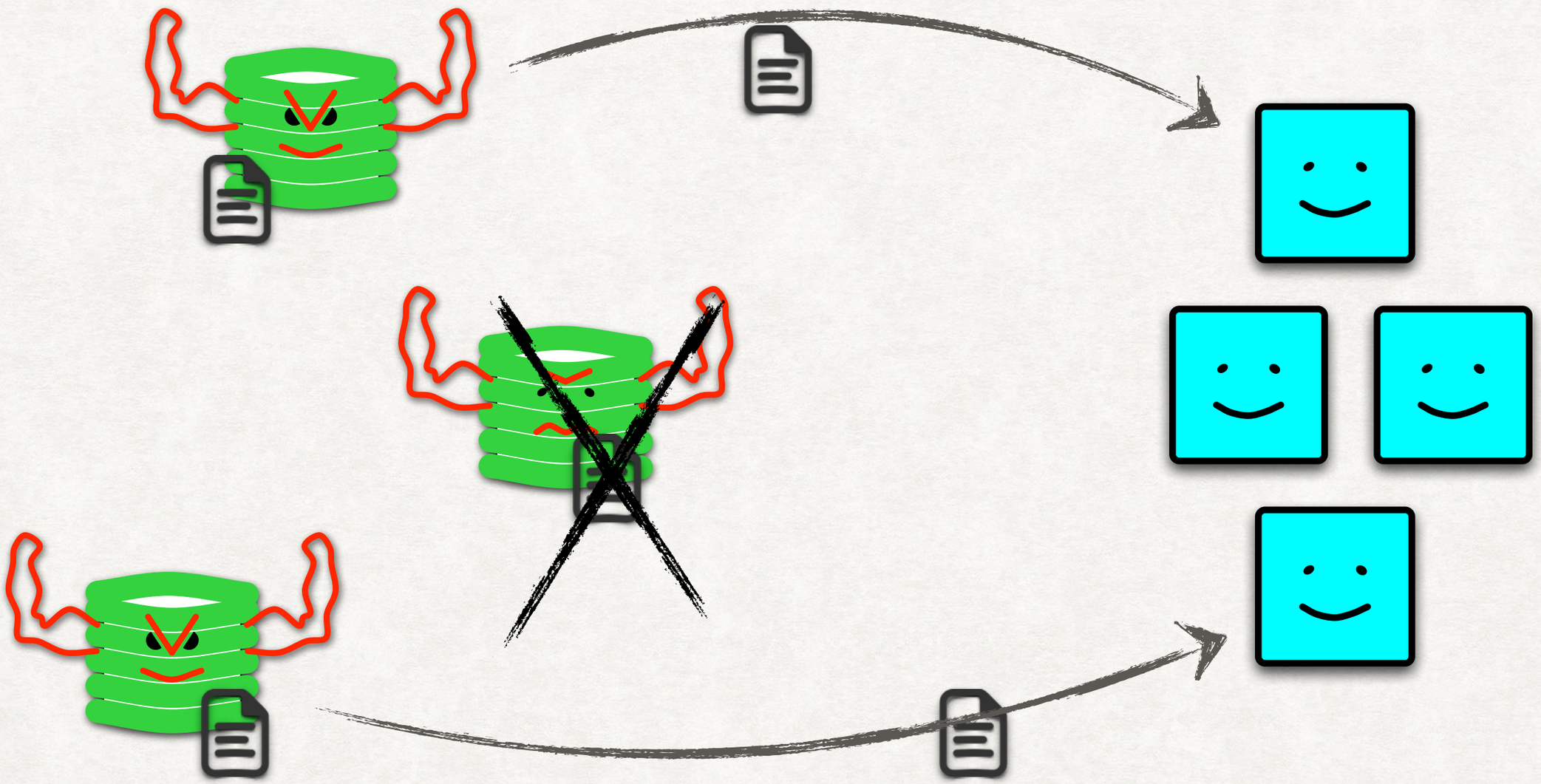
APACHE HBASE
CC BY SA
APACHE PHOENIX

Durability with Replication



CC BY SA

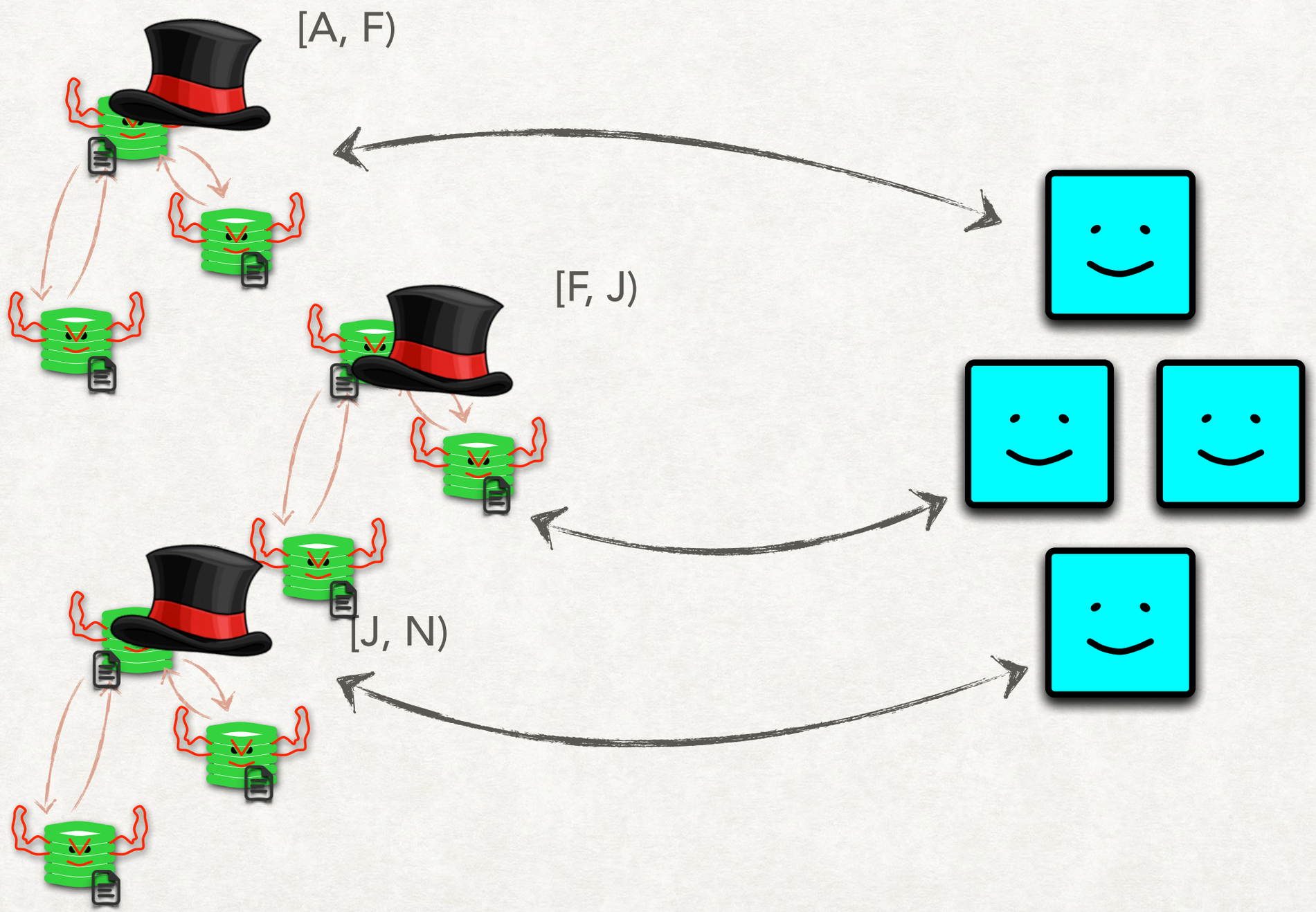
~~Durability~~Availability with Replication



CC BY SA

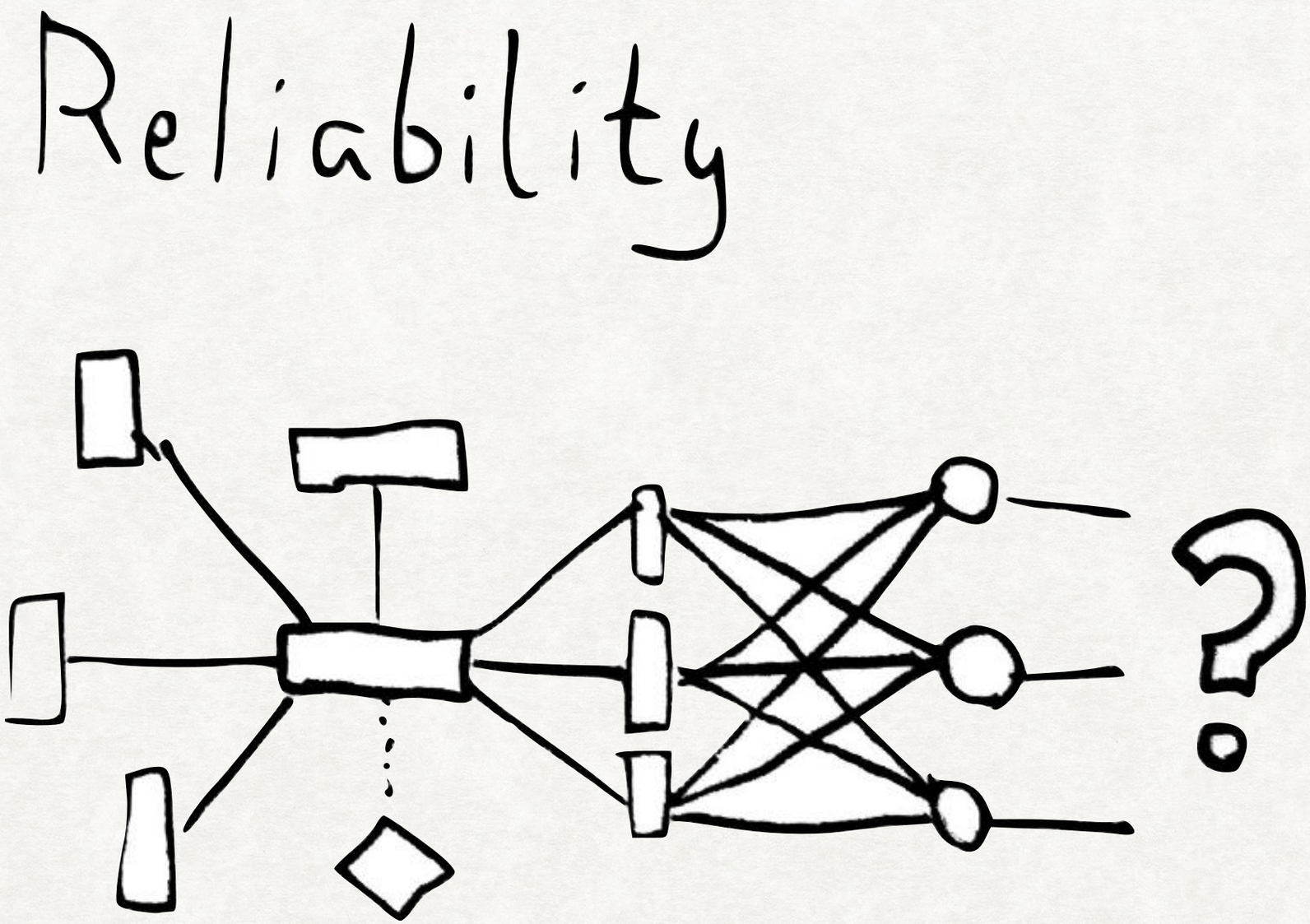
APACHE HBASE
CC BY SA
APACHE PHOENIX

Durable, Available Partitions



APACHE HBASE
CC BY SA
APACHE PHOENIX

Empirical Evidence

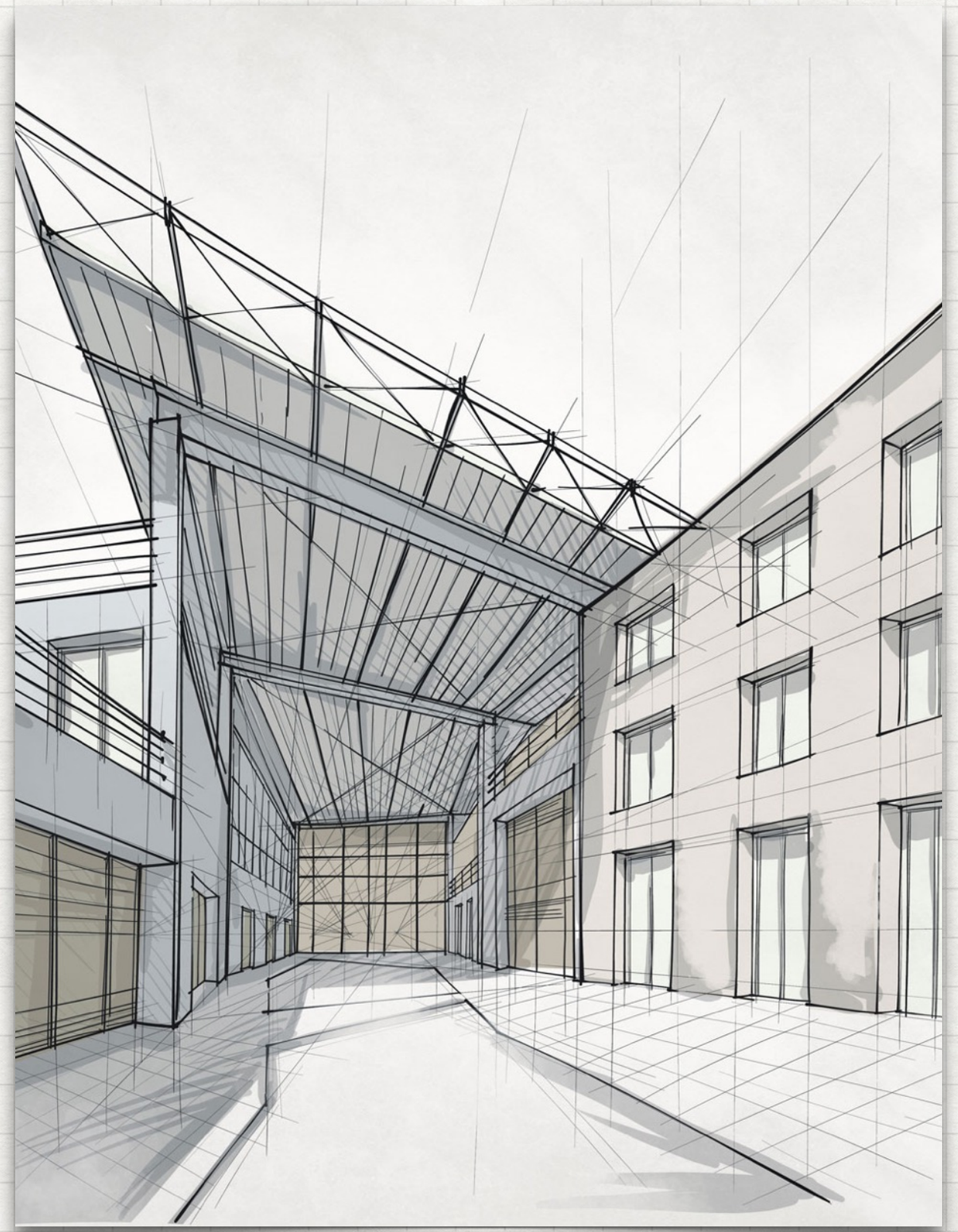


https://aphyr.com/posts/
281-call-me-maybe-carly-rae-jepsen-and-the-perils-of-network-partitions

APACHE HBASE
CC BY SA
APACHE PHOENIX

Part 3

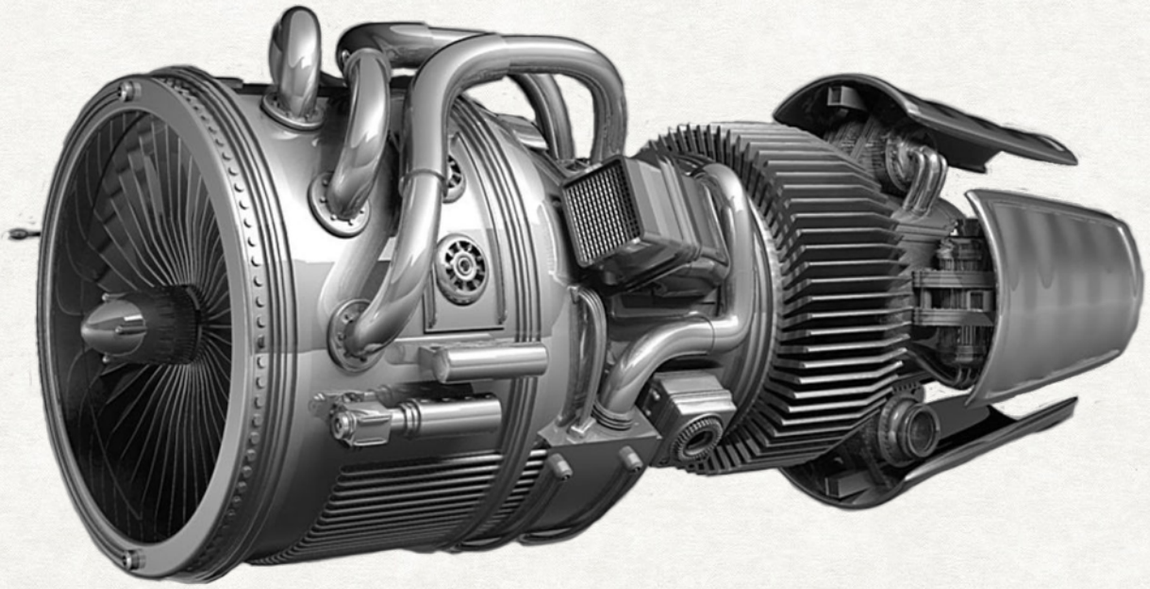
NoSQL In Depth With Apache HBase



cc BY SA

What is Apache HBase?

A high performance horizontally scalable bytestore for Big Data, suitable as the store of record for *mission critical data*.



CC BY SA

What is Apache HBase?

"Bigtable: A Distributed Storage System for Structured Data"

- Chang, Fay, et al. OSDI'06
- "Bigtable is a sparse, distributed, persistent multi-dimensional sorted map."
- http://research.google.com/archive/bigtable.html

Key Features include:

- Distributed storage over a cluster of machines
- Random, online, read and write access
- Schema-less data model
- Self-managed partitions

CC BY SA

Apache HBase Dependencies

Apache Hadoop Distributed Filesystem (HDFS)

- Distributed, fault-tolerant, throughput-optimized data storage
- The Google File System, 2003, Ghemawat et al.
- http://research.google.com/archive/gfs.html

Apache Zookeeper (ZK)

- Distributed, available, reliable coordination system
- The Chubby Lock Service …, 2006, Burrows
- http://research.google.com/archive/chubby.html

CC BY SA

Data Model

Table A

Column Families

Rows

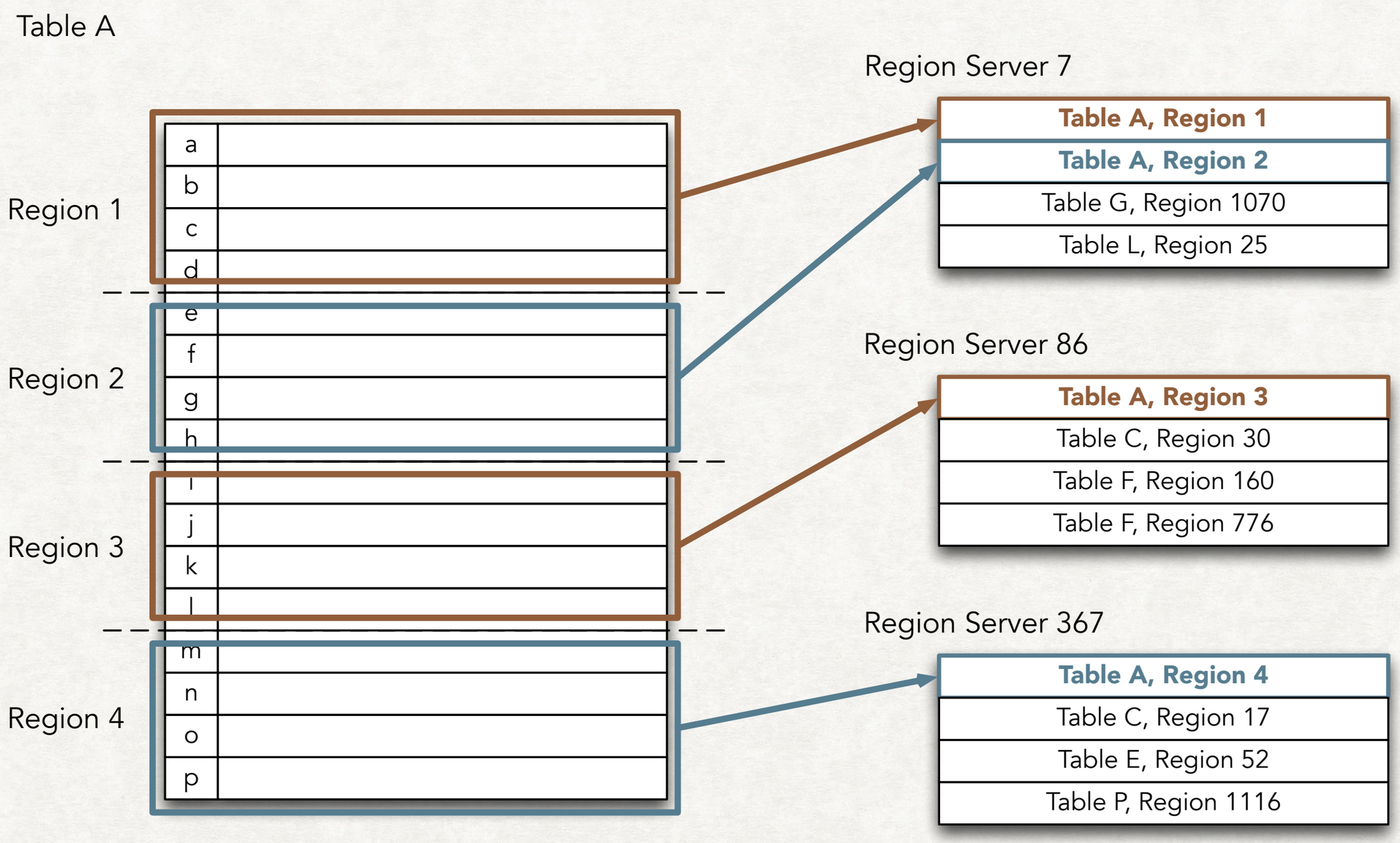
rowkey	column family	column qualifier	timestamp	value
a	cf1	"bar"	1368394583	7
			1368394261	"hello"
		"foo"	1368394583	22
			1368394925	13.6
			1368393847	"world"
	cf2	"2011-07-04"	1368396302	"fourth of July"
		1.0001	1368387684	"almost the loneliest number"
b	cf2	"thumb"	1368387247	[3.6 kb png data]

APACHE HBASE

CC BY SA

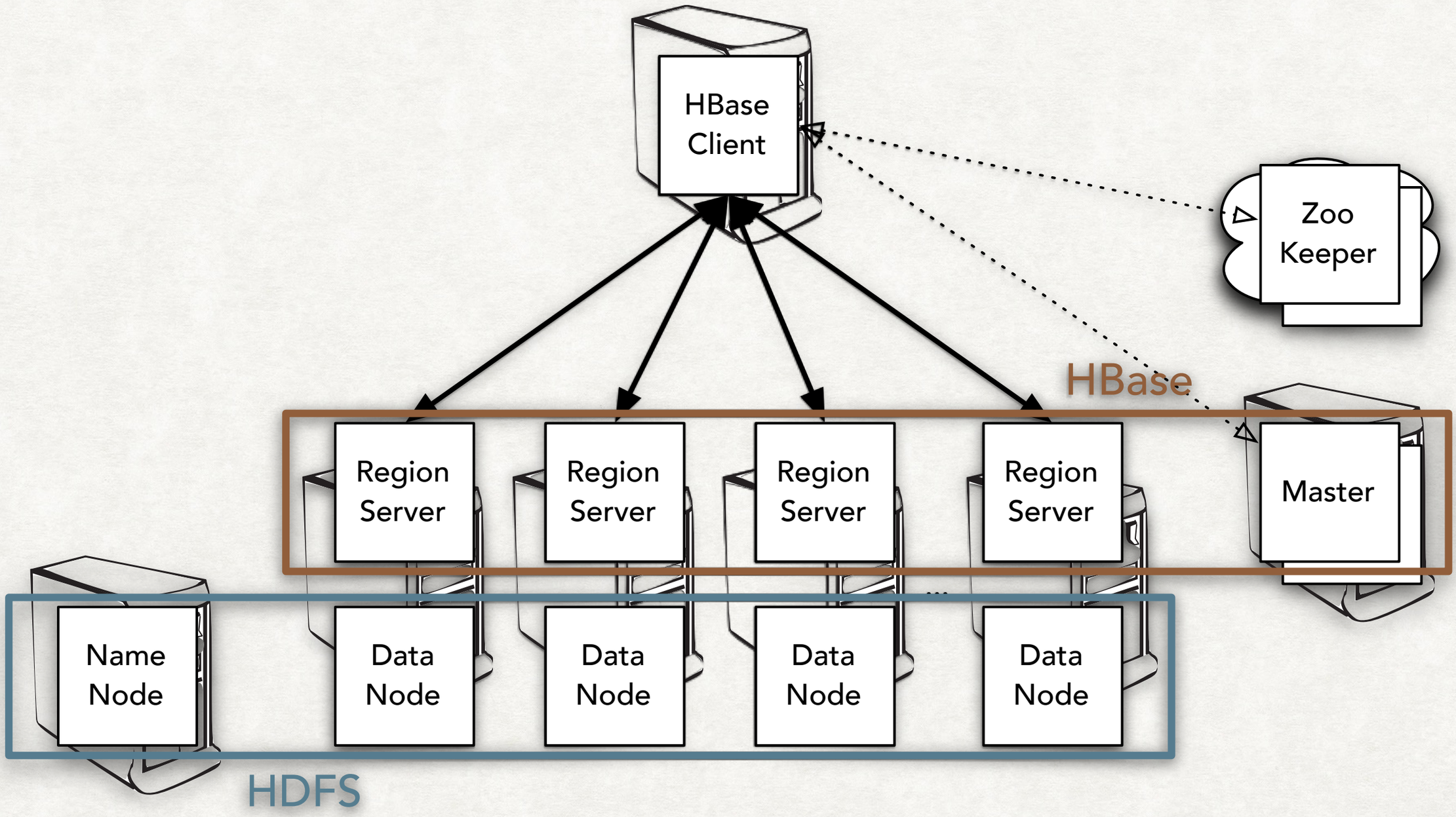
APACHE PHOENIX

Logical Architecture



APACHE HBASE
CC BY SA
APACHE PHOENIX

Physical Architecture



APACHE HBASE
CC BY SA
APACHE PHOENIX

User API

{rowkey => {family => {qualifier => {version => value}}}}

- Think: nested TreeMap (Java), OrderedDictionary (C#), OrderedDict (Python)

Basic data operations: GET, PUT, DELETE

SCAN over range of key-values

- benefit of the sorted rowkey business
- this is how you implement any kind of "complex query" *

GET, SCAN support Filters

- Push application logic to RegionServers

INCREMENT, APPEND, CheckAnd{Put,Delete}

- Server-side, atomic data operations, can be contentious!

CC BY SA

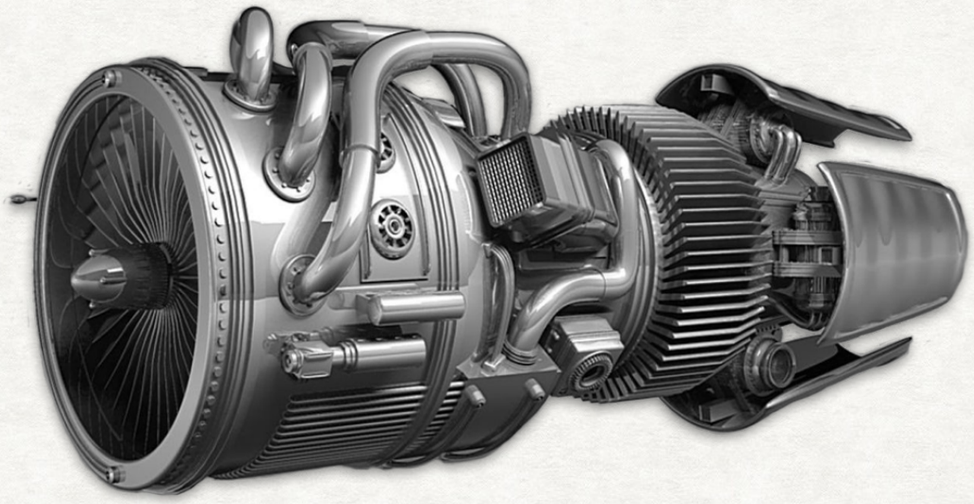
Part 4

Bringing SQL Back With Apache Phoenix

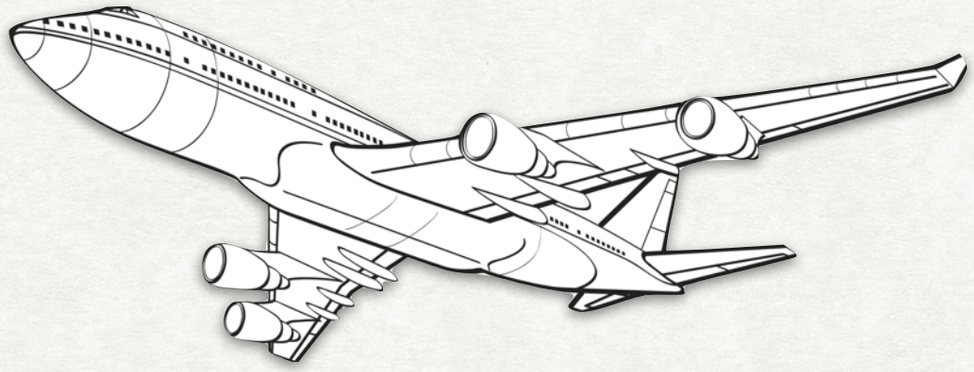
CC BY SA

What is Apache Phoenix?

==



==



CC BY SA

What is Apache Phoenix?

A relational database layer for Apache HBase

- Query Engine
  - Transforms SQL queries into native HBase API calls
  - Push as much work as possible into HBase region servers
- Metadata repository
  - Typed access to data stored in HBase tabled
- Thick and thin JDBC Driver

Originally Developed at Salesforce

A growing community

APACHE HBASE

CC BY SA

APACHE PHOENIX

Phoenix puts SQL back in NoSQL

Familiar SQL interface (DDL, DML, &c.)

- Read only materialized views on existing tables
- Dynamic columns extend schema at runtime
- Flashback queries using versioned schema

Data Types

Secondary Indices

Joins

Query planning and optimization

Multi-tenancy

APACHE HBASE

CC BY SA

APACHE PHOENIX

Phoenix puts SQL back in NoSQL

Accessing HBase data with Phoenix can be substantially easier than directly through the HBase API:

```
SELECT * FROM foo WHERE bar > 30
```

Versus

```
HTable t = new HTable("foo");
RegionScanner s = t.getScanner(new Scan(…,
    new ValueFilter(CompareOp.GT,
        new CustomTypedComparator(30)), …));
while ((Result r = s.next()) != null) {
     // blah blah blah Java Java Java
}
s.close();
t.close();
```

Your BI tool probably can't do this

(And we didn't include error handling…)

APACHE HBASE

CC BY SA

APACHE PHOENIX

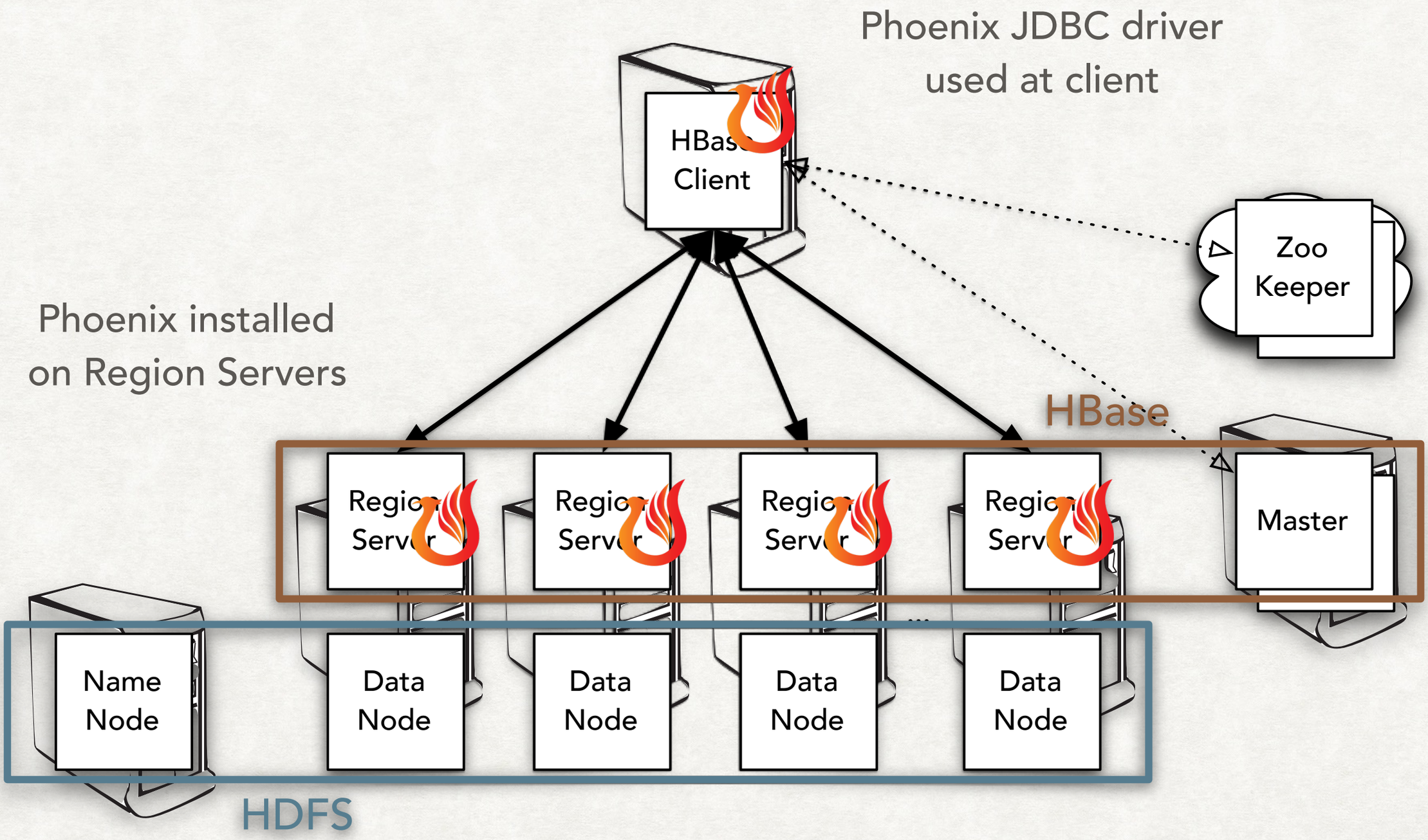
Phoenix puts SQL back in NoSQL

Accessing HBase data with Phoenix can be substantially faster than direct HBase API use:

- Phoenix parallelizes queries based on stats; HBase does not know how to chunk queries beyond scanning an entire region
- Phoenix pushes processing to the server - most "by hand" API accesses do not use coprocessors
  - A huge difference for aggregation queries
- Phoenix supports and uses secondary indexes
- Phoenix uses "every trick in the book" based on various factors: the HBase version, metadata and query, reverse scans, small scans, skip scans, etc.

CC BY SA

Phoenix + HBase Architecture



CC BY SA

Thanks!

Nick Dimiduk (@xefyr)
Committer, PMC
http://n10k.com

2015-12-02

Apache HBase (@hbase)
http://hbase.apache.org

Apache Phoenix (@ApachePhoenix)
http://phoenix.apache.org

CC BY SA