NoSQL Databases

an overview

Who? Why?

- During studies: Excited by simplicity
- Crawler Project:
  - 100 Million records
  - Single server
  - 100+ QPS
  - Initially: Limited query options
  - Now: Query them all
  - Experimented with all of them as a backend

What types of database are there?

- SQL
    - Relational (MySQL, Postgres, Oracle, DB2)
- NoSQL
    - Key Value Stores (Membase, Voldemort)
    - Document Databases (CouchDB, MongoDB, Riak)
    - Wide Column Stores (Cassandra, HBase, Hypertable)
    - Graph Databases (Neo4j)
    - Datastructure Servers (Redis)

What do they often have in common

- Most of them:
  - Not 100% ACID compliant (but fast!)
  - Standardized interfaces (http, protocol buffers, ...)
  - Schema free
  - Open source

- The distributed ones:
  - Eventual consistency
  - Scaling is easy (no, really!)

Key - Value stores

simple and fast

Key Value Stores

- Data model is an associative array (aka: hash / dictionary / ...)

KEY	VALUE
"/user/john/profile"	"{ age: 42, friends: ['joanne', 'jose'], avatar: 'icon234.png'}"
"users:online"	122
"/top_companies/acquia.php"	"<HTML><LOREM>ipsum</LOREM>...</HTML>"
"server:build-1:packages"	"rubygems|java|tomcat"
"server:build-1:last-launch"	"Thu Oct 06 19:38:29 +0200 2011"

⇧

logic in the key

Key Value Stores

- Don't want to know what the "value" part is supposed to be

KEY	VALUE
"/user/john/profile"	11010101010110100101010010101010
"users:online"	101001010010110101101001010100101
"/top_companies/acquia.php"	11010111011100101010011101011010
"server:build-1:packages"	11110101101001110101001110101010
"server:build-1:last-launch"	111101010010001001010010101010110

Key Value Stores

Examples:

- MemcacheDB
- Membase
- Project Voldemort
- Scalaris
- (Kyoto + Tokyo) Cabinet
- Redis (can do way more)
- Berkley DB
- HandlerSocket for MySQL (can also do a bit more)
- Amazon S3

- Note: A lot of the other databases can be used as a key-value store

Document databases

know what you're talking about

Document databases

- Data model is still an associative array

KEY	DOCUMENT
X	Y

Document databases

- Difference: servers know about your values

KEY	DOCUMENT
"jane@hotmail.com"	"{ age: 42, friends: ['malroy@gmail.com'], avatar: 'icon-234.png' }"

"john@example.org"	"{
	age: 33,
	highscores: {
	'sim-garden': [
	{1317930201: 131232,
	time-played: 320}
	]
	}
	}"

"malroy@gmail.com"	"{ age: 51, friends: ['jane@hotmail.com']}"

Document databases

KEY	DOCUMENT
"bob@builder.com"	"{ age: 23, friends: ['joanne@aol.com', 'jose@bigcorp.com'], avatar: 'kitten-141.png' }"
	"{ age: 42, friends: ['malroy@gmail.com'], avatar: 'icon-234.png' }"

	"{
	age: 33,
	highscores: {
	'sim-garden': [
	{1317930201: 131232,
	time-played: 320}
	]
	}
	}"

"malroy@gmail.com"	"{ age: 51, friends: ['jane@hotmail.com']}"

No common schema

Document databases

Nested data types

"john@example.org"	"{
	age: 33,
	highscores: {
	'sim-garden': [
	{1317930201: 131232,
	time-played: 320}
	]
	}
	}"

Document databases

References by key

(not enforced by database)

"malroy@gmail.com"	"{ age: 51, friends: ['jane@hotmail.com']}"

Document Databases

"Relations" by embedding:

```
"{
title: "The cake is a lie",
timestamp: 1317910201,
body: "Lorem ipsum sit dolor amet. Yadda [...] Thanks."
comments': [
        {
        author: "bob@builder.com",
        timestamp: 1317930231
        text: "First!"
        },
        {
        author: "janedoe@example.com",
        timestamp: 1317930359
        text: "Bob, you're an idiot!"
        }
        ]
    }
}"
```

Document Databases

Server side modifications:

Counters

Document Databases

Server side modifications:

```
@database.domains.update("acquia.com", "{cms: 'drupal'}")
```

Document Databases

Query for data

```
db.companies.find({ "city" : "Boston" } );
```

Document Databases

Examples:

- CouchDB
- MongoDB
- Terrastore
- OrientDB
- Riak

Wide column stores

bigdata is calling

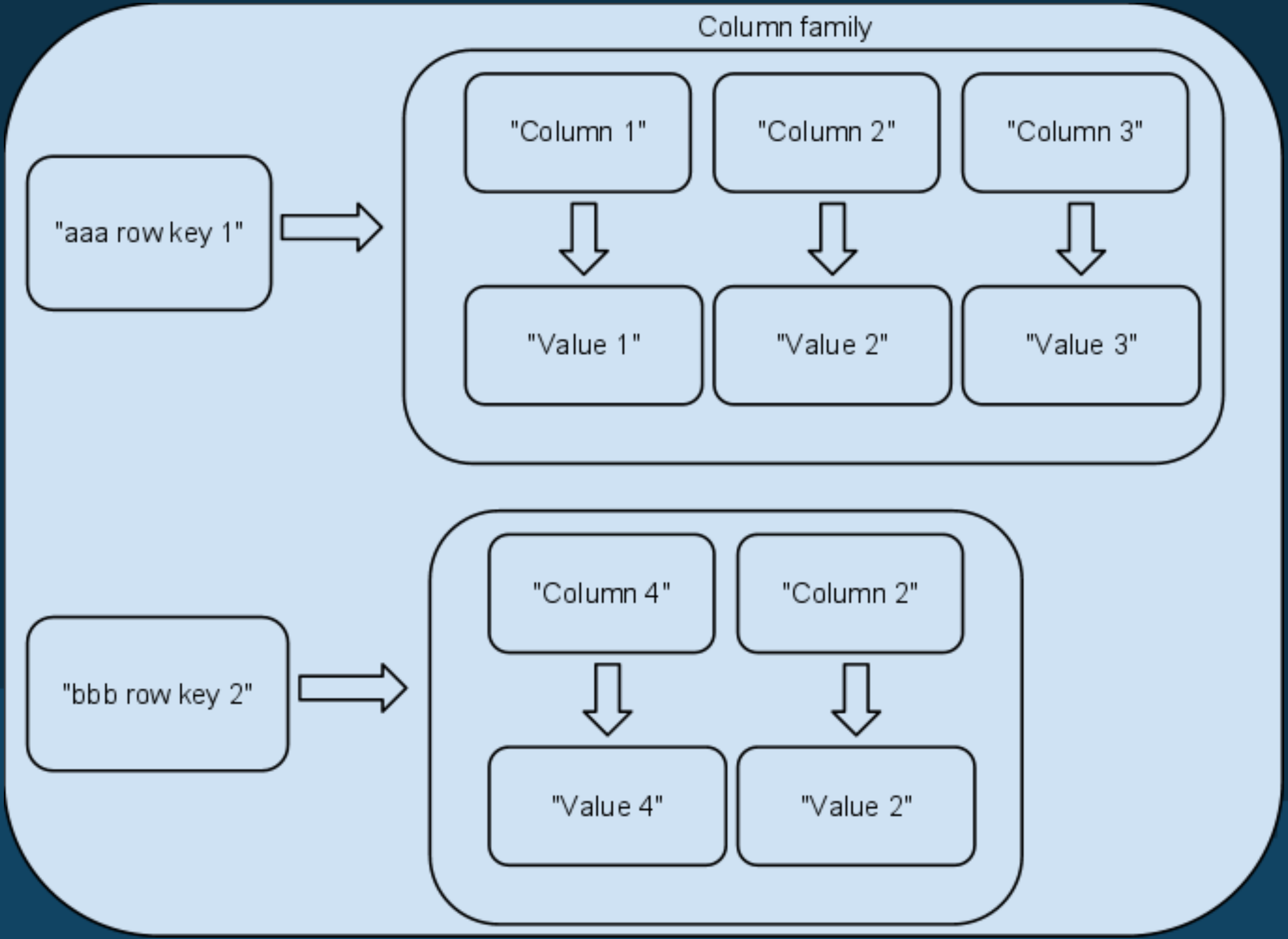
Wide column stores

- Data model is ... weird

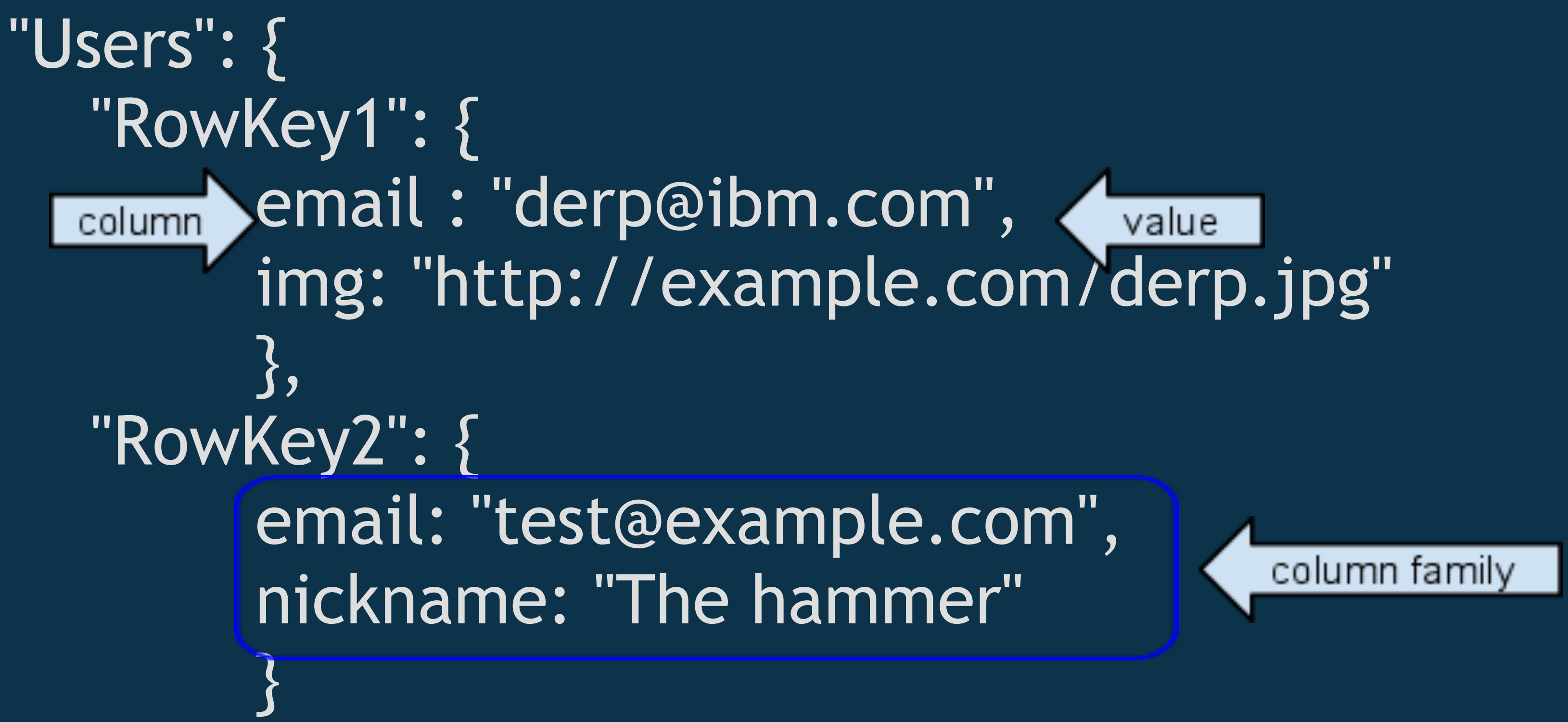
("**a sparse, distributed**, persistent **multidimensional** sorted map") *

* Google's BigTable Paper

Wide Column Stores

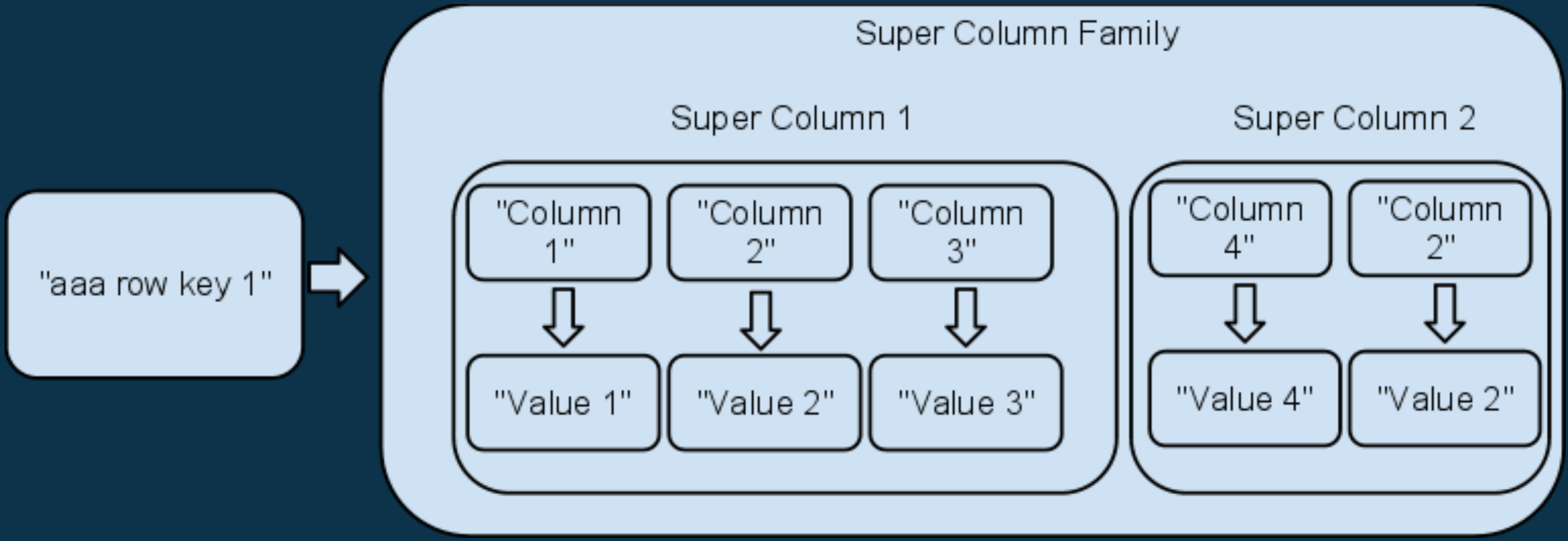


Wide Column Stores



```
"Users": {
   "RowKey1": {
         email : "derp@ibm.com",
         img: "http://example.com/derp.jpg"
         },
   "RowKey2": {
         email: "test@example.com",
         nickname: "The hammer"
         }
```

Wide Column Stores


"aaa row key 1"

Super Column Family

Super Column 1

"Column 1"

"Value 1"

"Column 2"

"Value 2"

"Column 3"

"Value 3"

Super Column 2

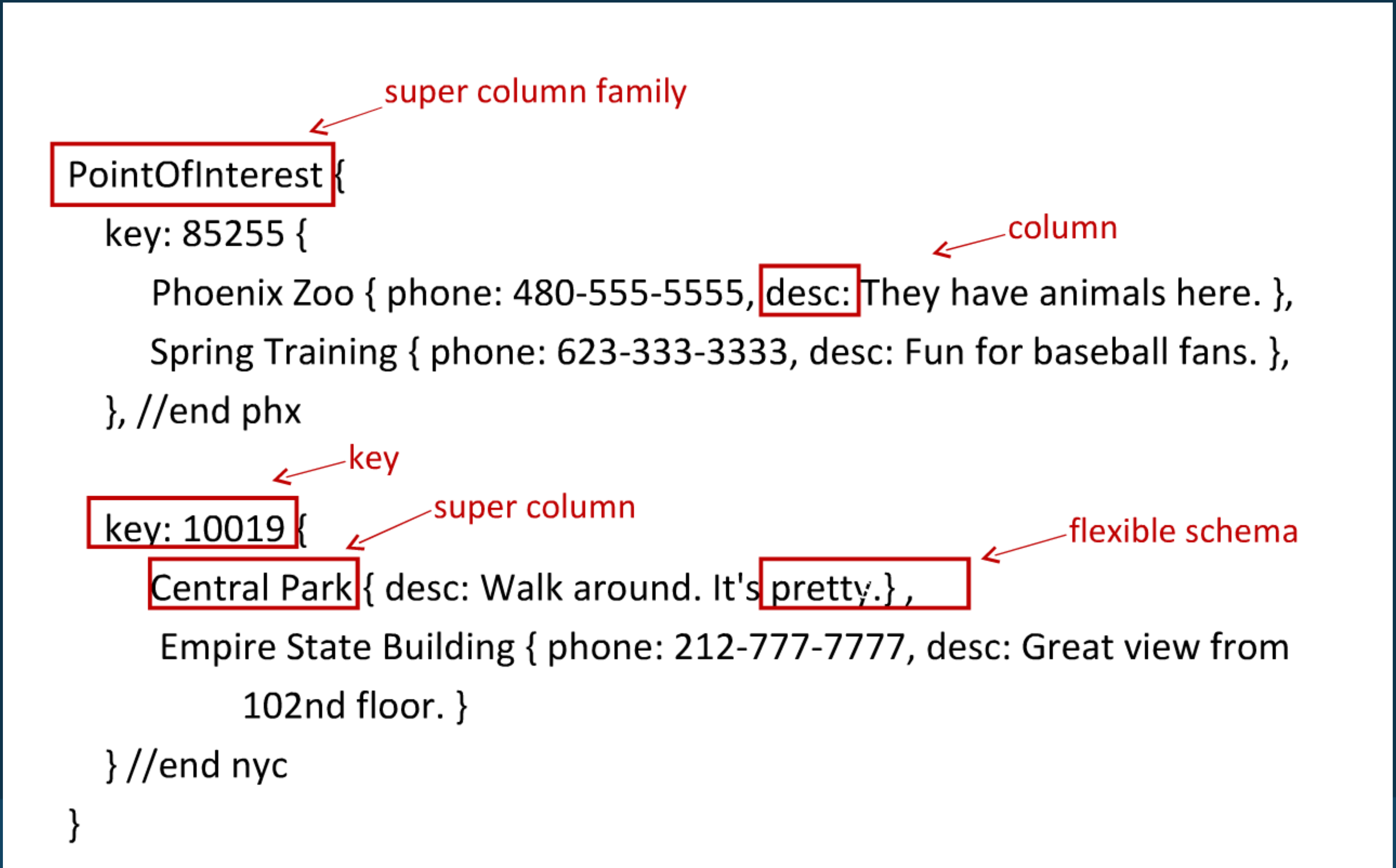
"Column 4"

"Value 4"

"Column 2"

"Value 2"

Wide Column Stores



super column family
column
key
super column
flexible schema

```
PointOfInterest {
   key: 85255 {
      Phoenix Zoo { phone: 480-555-5555, desc: They have animals here. },
      Spring Training { phone: 623-333-3333, desc: Fun for baseball fans. },
   }, //end phx

   key: 10019 {
      Central Park { desc: Walk around. It's pretty.} ,
       Empire State Building { phone: 212-777-7777, desc: Great view from
            102nd floor. }
   } //end nyc
}
```

eben hewitt - the cassandra data model:
http://www.slideshare.net/ebenhewitt/cassandra-datamodel-4985524

Wide Column Stores

Examples:

- Cassandra
- HBase
- Hypertable

Note: All of those target multi-machine scalability

Graph Databases

your DB is now in a relationship

Graph Databases

Data model usually consists of:

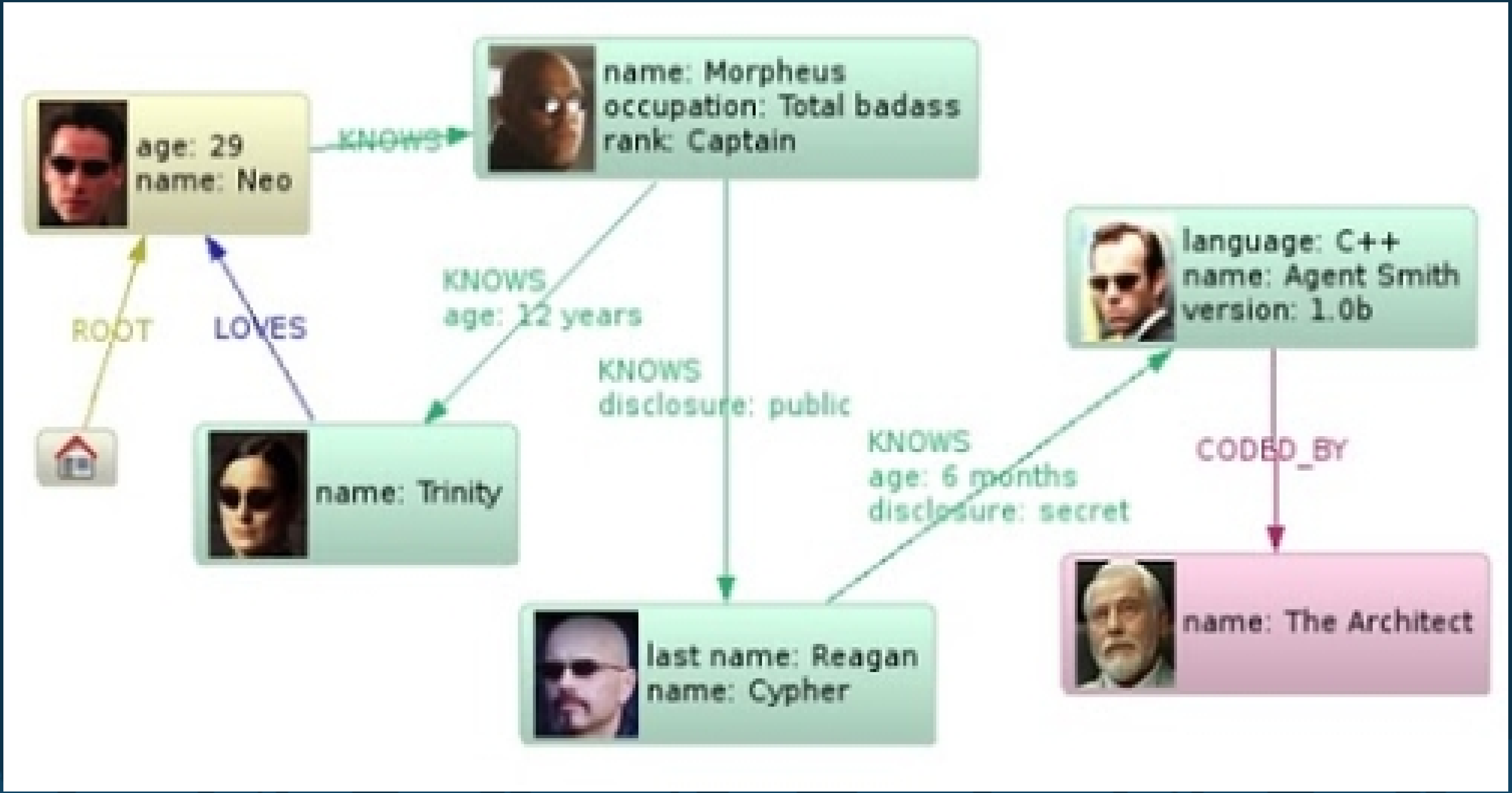
Nodes

Relationships

Properties

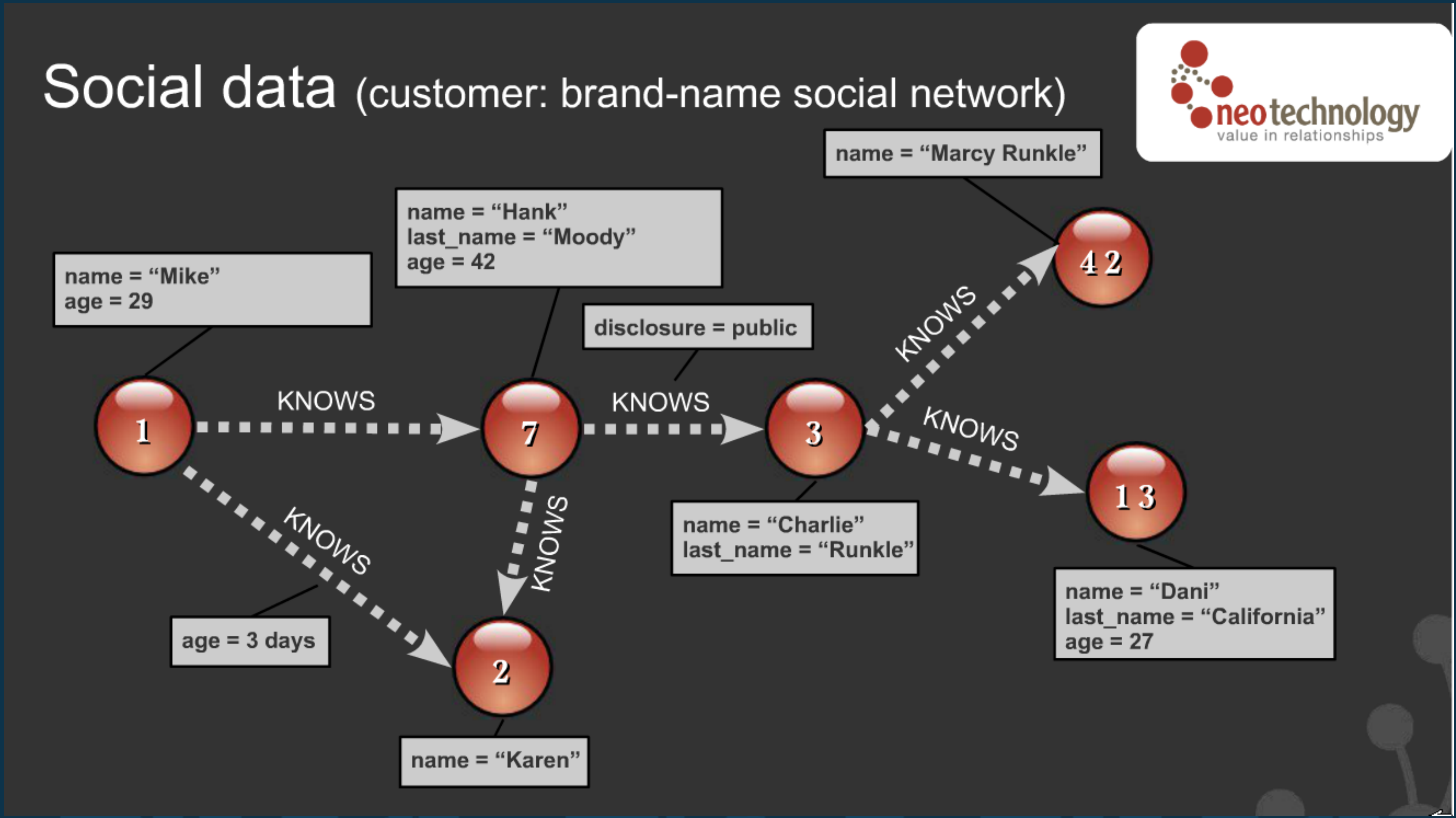
Note: They can have billions of those on a single machine!

Graph Databases



source: neo4j wiki

Graph Databases



Social data (customer: brand-name social network)

name = "Mike"
age = 29

name = "Hank"
last_name = "Moody"
age = 42

disclosure = public

name = "Marcy Runkle"

name = "Charlie"
last_name = "Runkle"

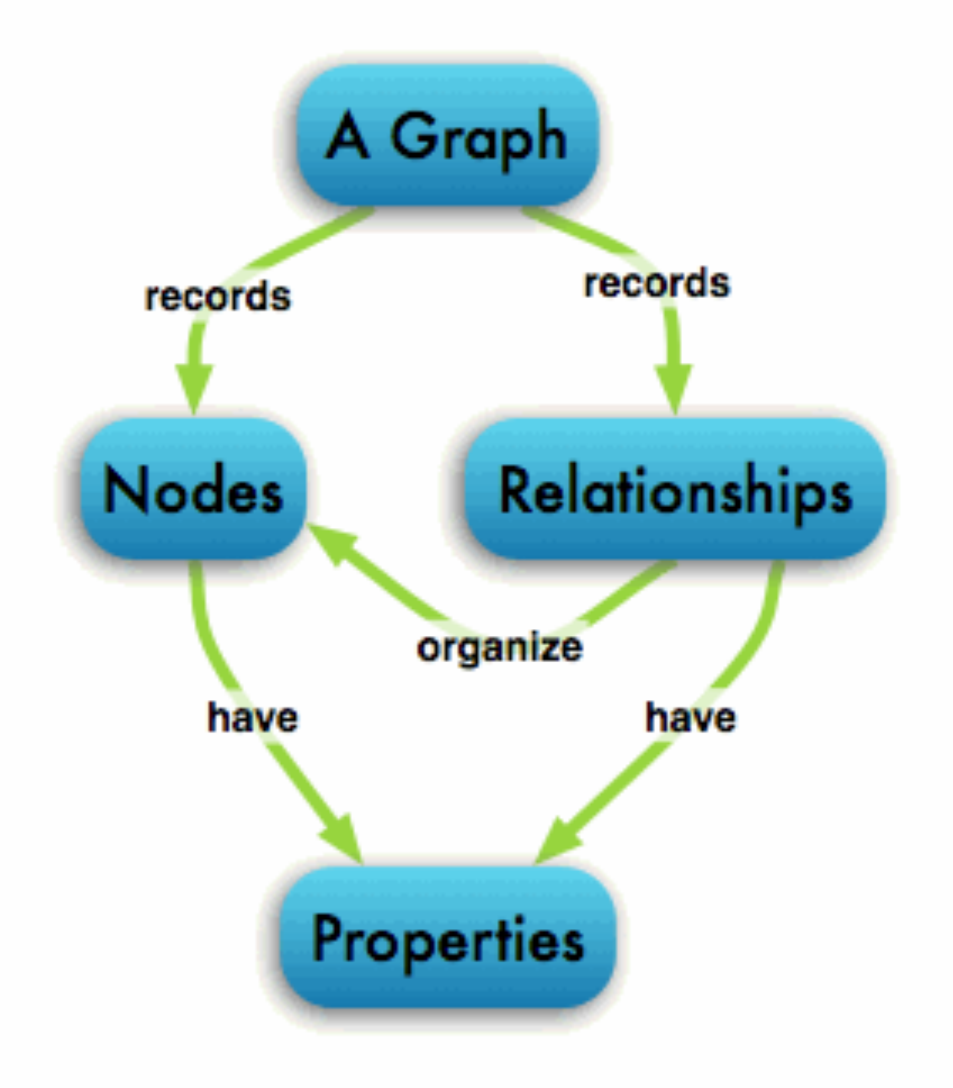
name = "Dani"
last_name = "California"
age = 27

age = 3 days

name = "Karen"

http://www.slideshare.net/peterneubauer/neo4j-5-cool-graph-examples-4473985

Graph Databases



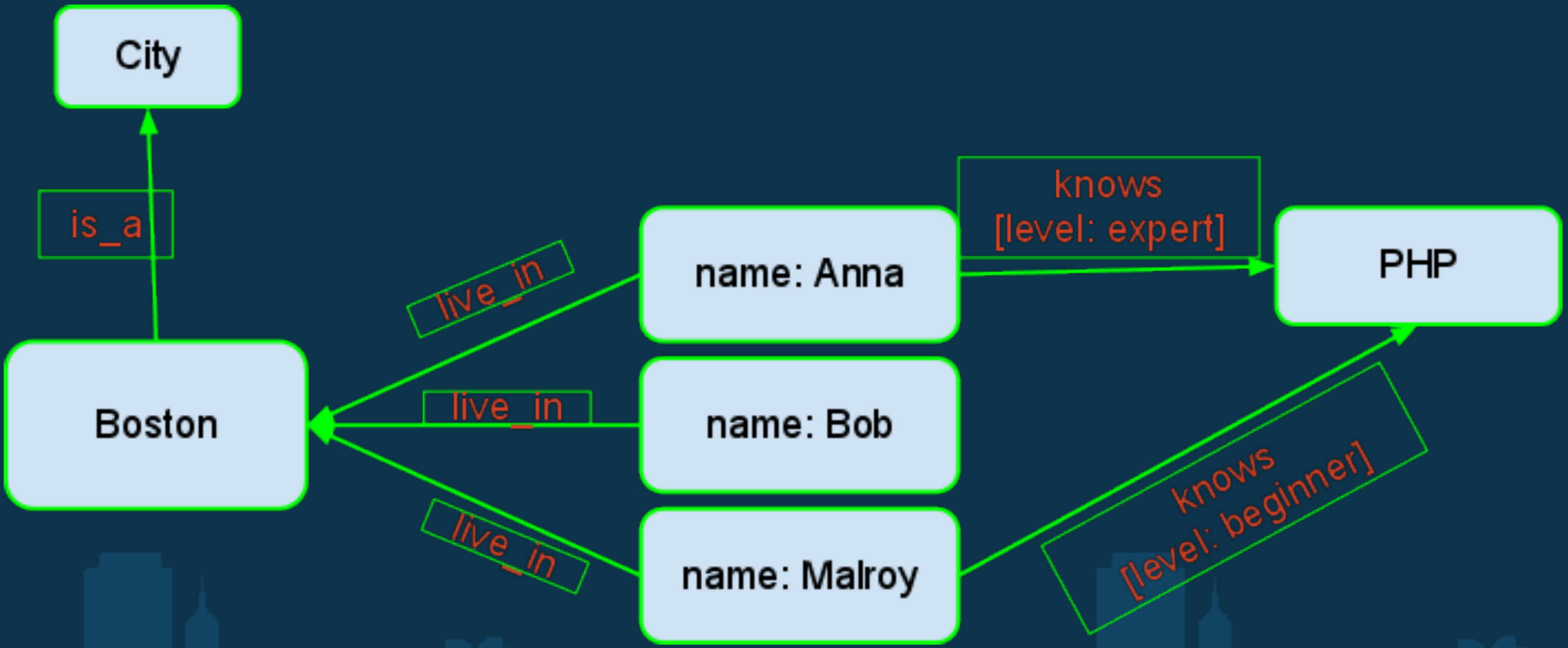
Graph Databases

Traversal:

1. start at a node A
2. Collect all connected nodes if they:
    1. have a certain property on themselves
    2. have a certain property on their relationship to node A

Graph Databases

Traversal:
"All Bostonians that know PHP"



Graph databases

"How do I find my first node to start the traversal from?"

Graph databases

Examples:

- Neo4J
- Sones

Data structure servers

aka: Redis

Data structure servers (redis)

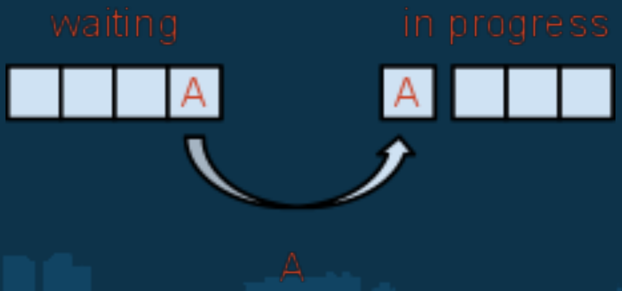
Data schema:

- Strings
- Hashes
- Lists
- Sets
- Sorted sets.

Data structure servers (redis)

Functionality for Lists:

- push/pop (blocking or non-blocking, from left or right)
- trim (-> capped lists)
  - example: a simple log buffer for the last 10000 messages:
  -
  - def log(message)
  -   @redis.lpush(:log_collection, message)
  -   @redis.ltrim(:log_collection, 0, 10000)
  - end
- brpoplpush()



Data structure servers (redis)

Functionality for Strings:

- decrement/increment (integers + soon float)
- getbit,setbit,getrange,setrange (-> fixed length bitmaps?)
- append (-> grow the bitmaps)
- mget/mset (set/get multiple keys at once)
- expire (great for caching, works for all keys)

```
@redis.incr(:counter_acquia_com, 1)
@redis.setbit(:room_vacancy, 42, 0) #guest moved in room 42
@redis.setbit(:room_vacancy, 42, 1) #guest moved out
```

Data structure servers (redis)

Functionality for Hashes:

- decrement/increment (integers + soon float)
  - visitor counter?
- hexists (determine if a field exists)
  - check if e.g. this customer is a credit card number in the system (server side!)

Data structure servers (redis)

Functionality for Sets:

- server side intersections, unions, differences
  - *Give me all keys in the set "customers:usa" that are also in the set "customers:devcloud"*
  - *What is the difference between the sets "sales-leads" and "already-called"*
    - result can be saves as a new set
- "sorted sets"
  - sets with a score
  - score can be incremented/decremented
  - server side intersections and unions available

Data structure servers (redis)

Pub/Sub:

- A simple publish subscribe system
- publish(channel, message)
- subscribe(channel) / unsubscribe(channel)
    - also available: subscribe to a certain pattern
        - psubscribe(:alert_channel, "prio:high:*")

          ```
          {|message|
              send_sms(@on_call, message)
          }
          ```

Data structure servers (redis)

Using "redis-benchmark" on my MBP:

```
GET: 69930.07 requests per second
SET: 70921.98 requests per second
INCR: 71428.57 requests per second
LPUSH: 70422.53 requests per second
LPOP: 69930.07 requests per second
SADD: 70422.53 requests per second
SPOP: 74626.87 requests per second
```

Search in NoSQL

Where's Waldo?

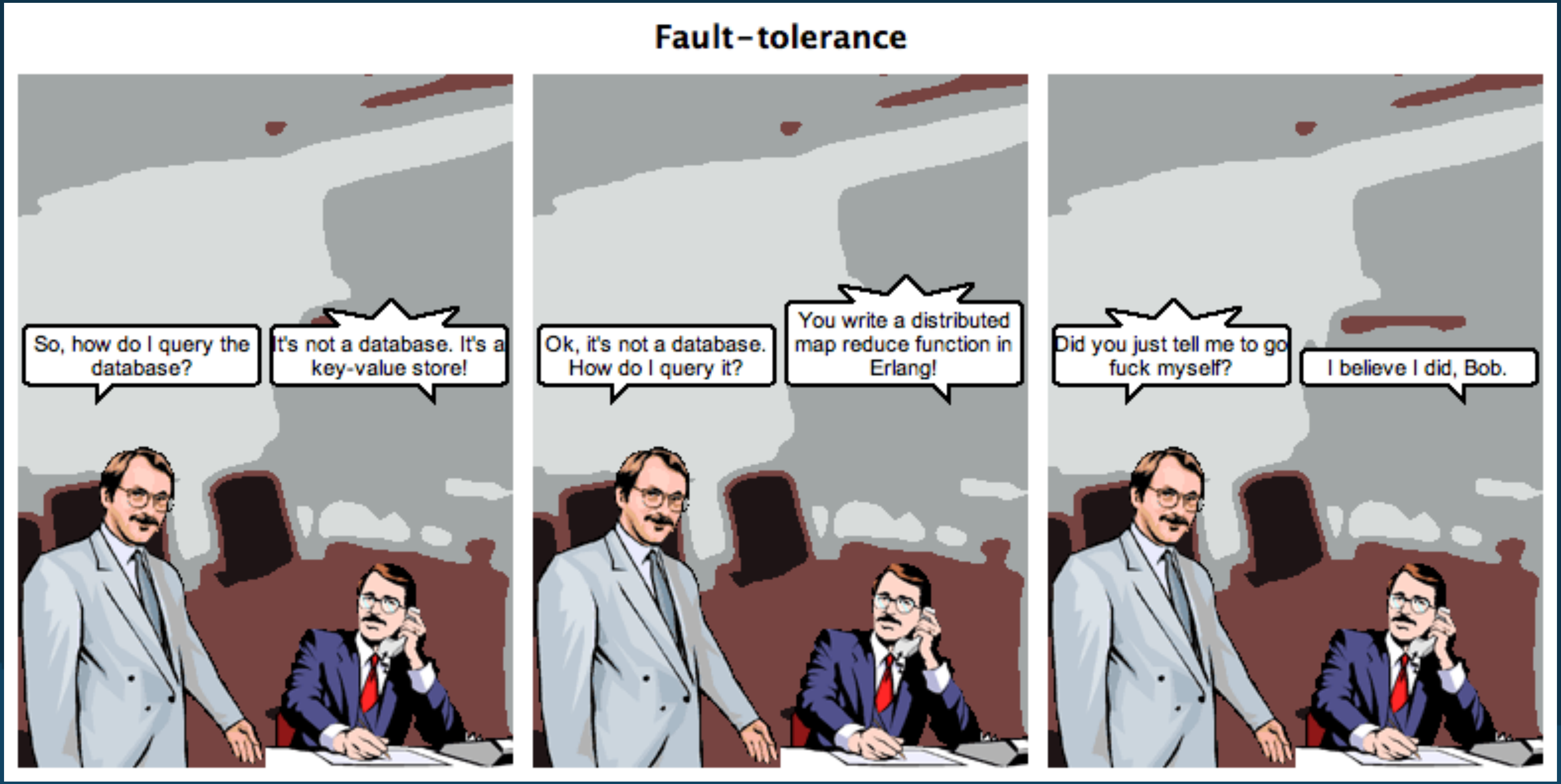
How can I get my data?

Access by known key (most of them)

```
db.get("domains:acquia.com")
db.get("users:john")
```

How can I get my data?

Map-Reduce (CouchDB, Riak, MongoDB)


Fault−tolerance
So, how do I query the database?
It's not a database. It's a key-value store!
Ok, it's not a database. How do I query it?
You write a distributed map reduce function in Erlang!
Did you just tell me to go fuck myself?
I believe I did, Bob.

How can I get my data?

Map-Reduce (example: where do my customers come from?)

Map:

```
function(doc) {
  if (doc.Type == "customer") {
    emit(doc.country, 1);
  }
}
```

Reduce:

```
function (key, values) {
    return sum(values);
}
```

How can I get my data?

Secondary Indexes (e.g. Riak, Cassandra, MongoDB)

MongoDB:

```
db.users.find({last_name: 'Smith'})
```

How can I get my data?

Graph traversal (Graph databases)

Chose your poison: SPARQL/Gremlin/Blueprint/...

How can I get my data?

External search services

- Elastic Search has CouchDB Integration (+unofficial MongoDB)
- "Solandra" allows you to save your Solr index to Cassandra
- "Riak Search" got integrated into Riak

Personal favorites

- Riak (scales really nicely over several servers)
- Redis (fast and useful)
- MongoDB (annoying to scale, but fast for smaller things, really nice querying options)
- Elasticsearch (clutter free and easily scalable search)

Links

nosql.mypopescu.com

"My curated guide to NoSQL Databases and Polyglot Persistence"

www.nosqlweekly.com

"A free weekly newsletter featuring curated news, articles, new releases, jobs etc related to NoSQL."