Introduction

สถาบัน ไอเอ็มซี

What is MongoDB ?

Humongous

hu·mon·gous

/(h)yo͞o'mäNGgəs,-'məNG-/

adjective NORTH AMERICAN *informal*

huge; enormous.
"a humongous steak"

What is MongoDB ?

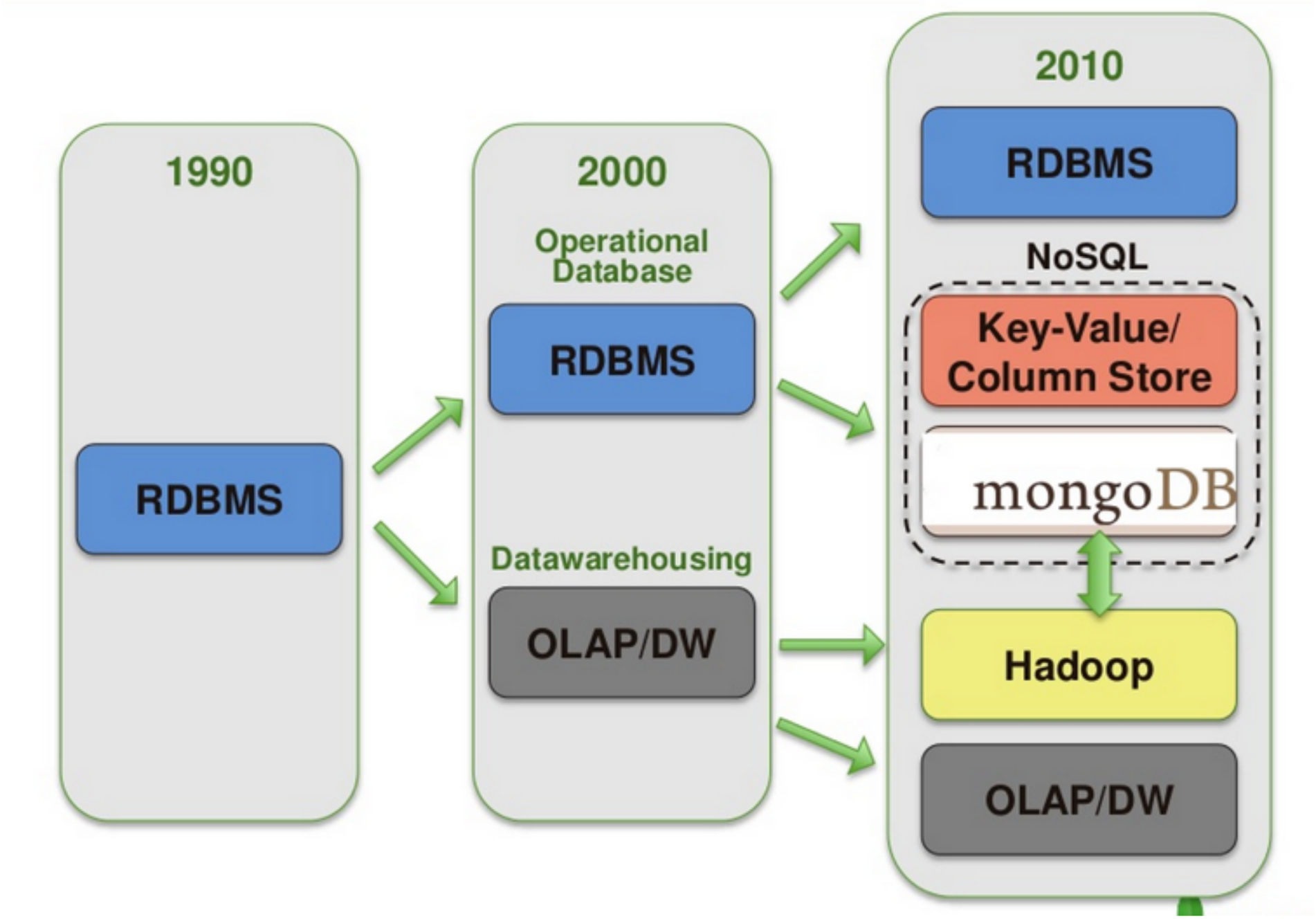
Non-relational database

Store **JSON** document

Schema less

Not support Joins

Evolution of database



What is NoSQL ?

Not Only SQL

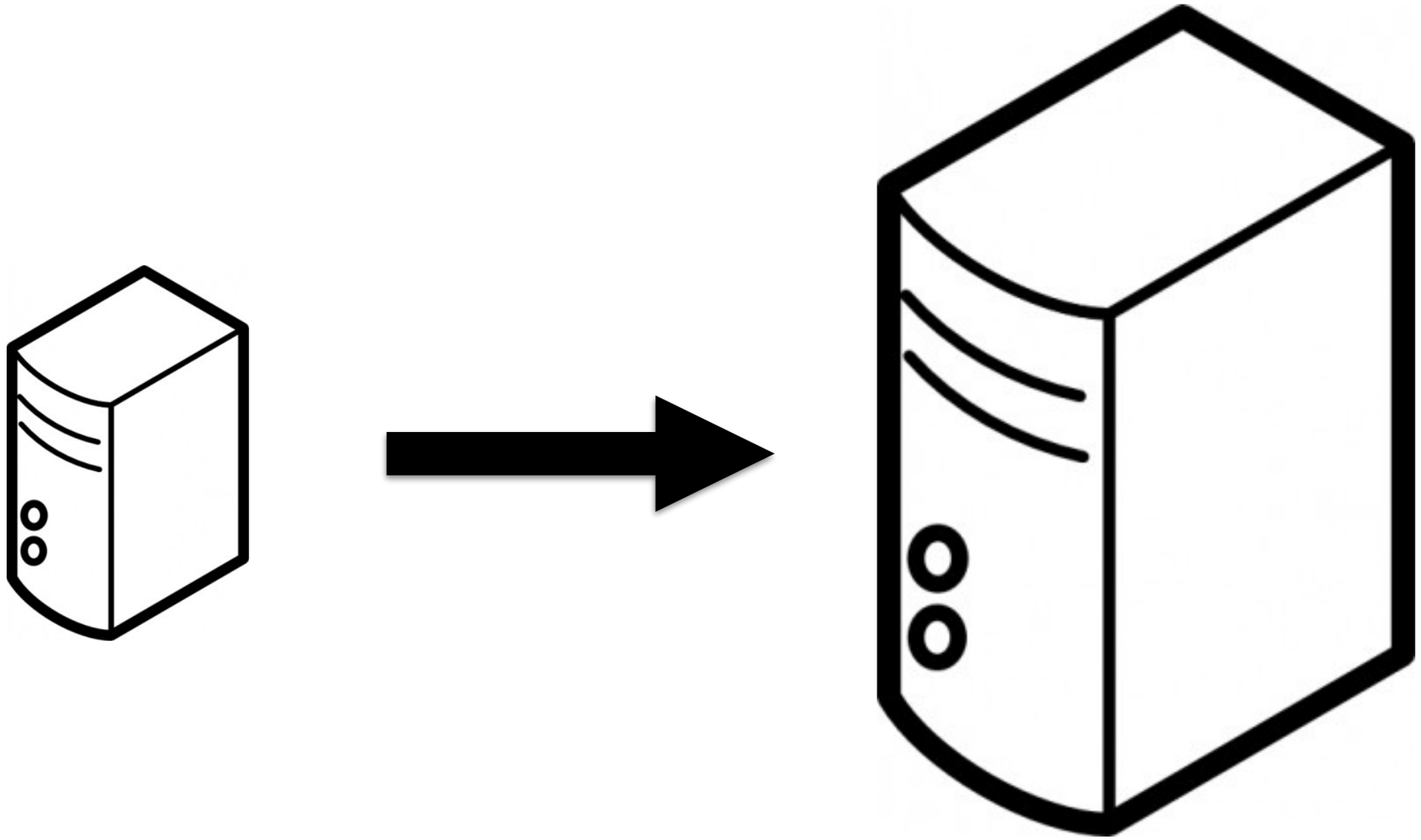
Non-relational database

No joins

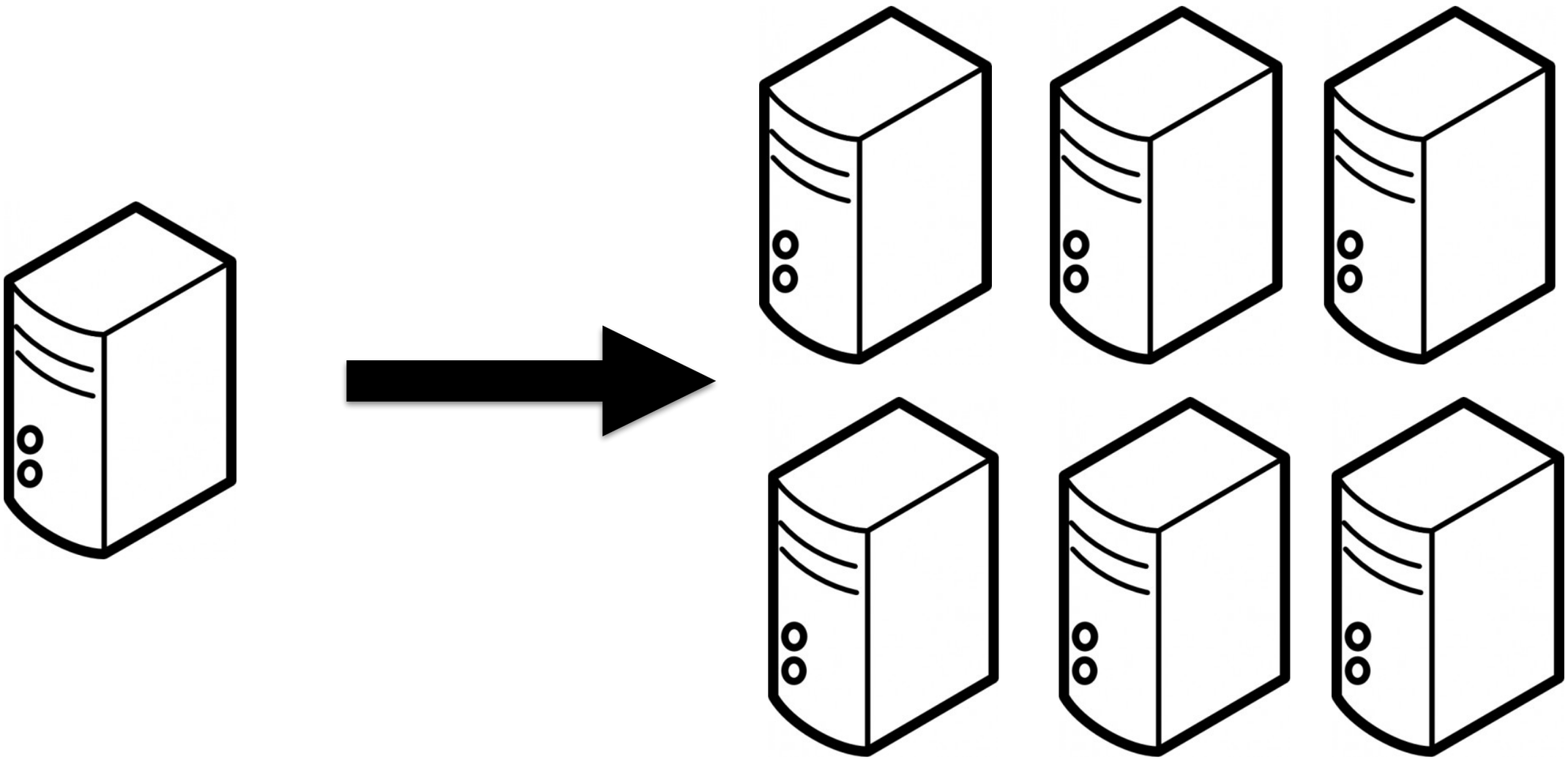
No complex transactions

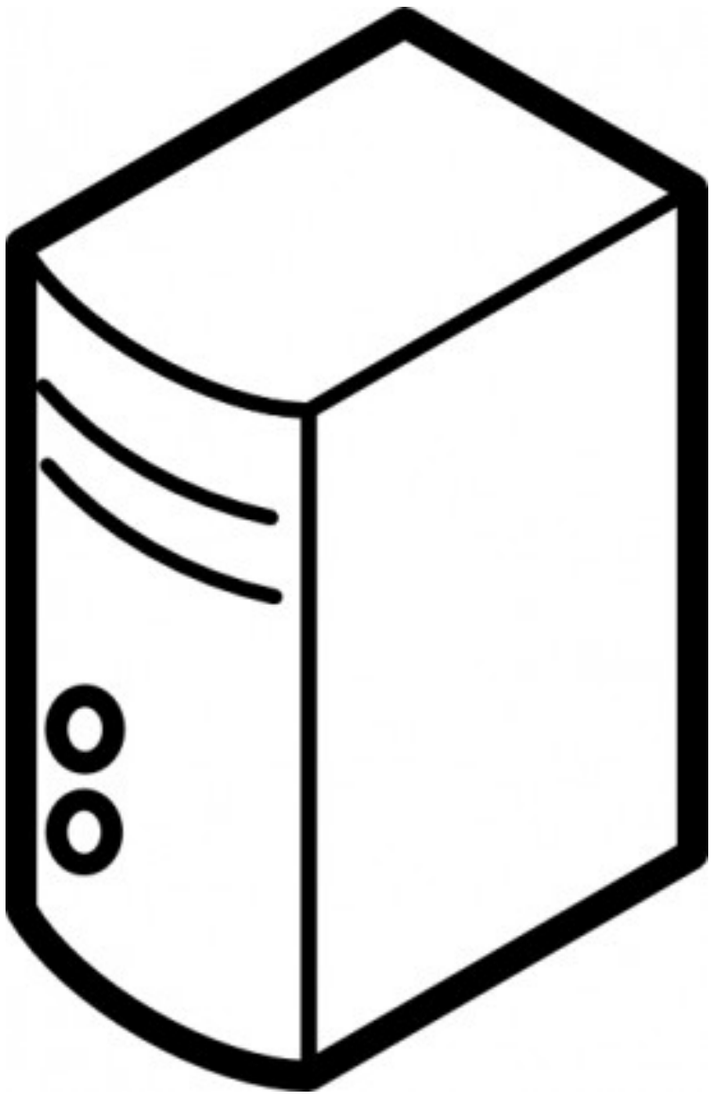
Focus on horizontal scalable

Vertical scaling

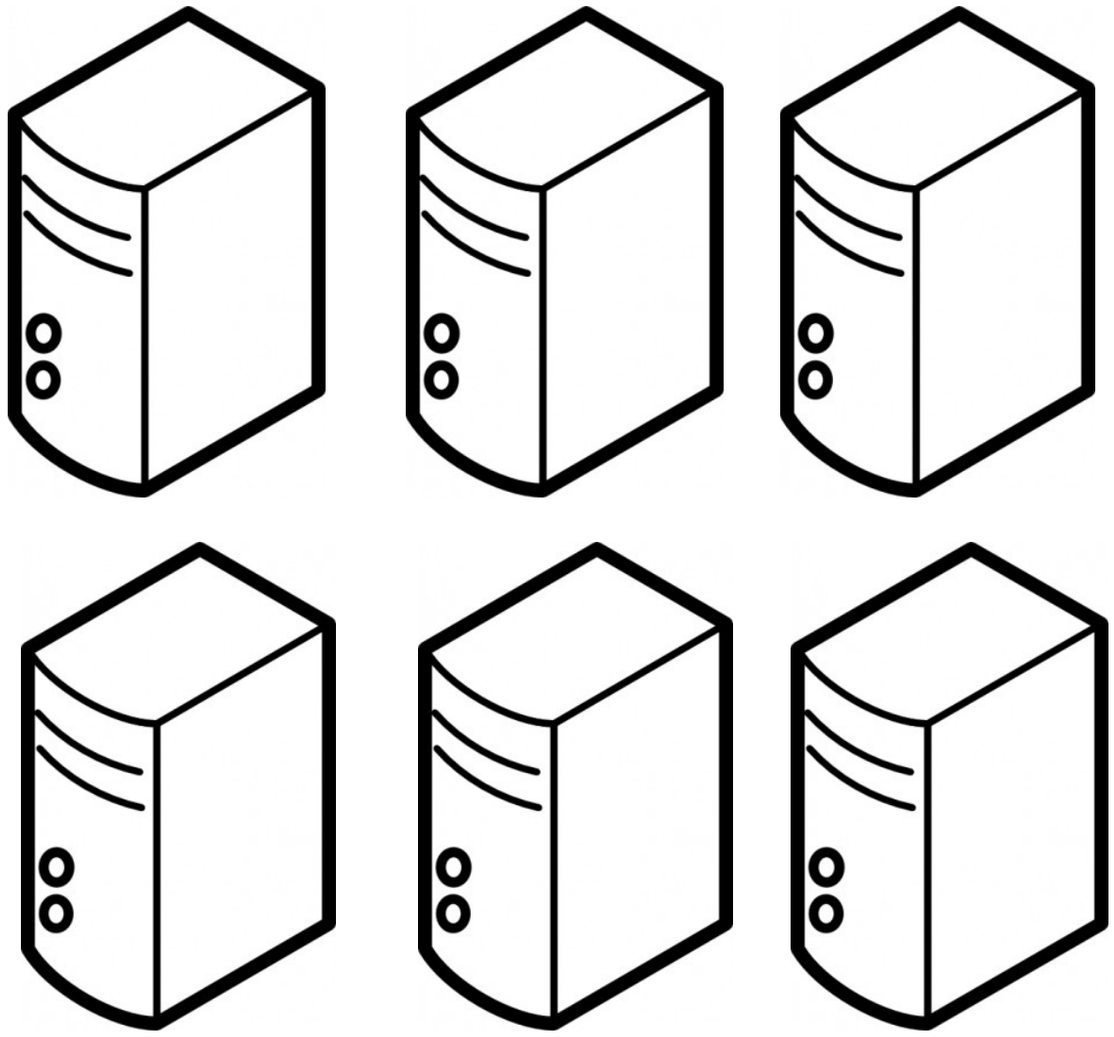


Horizontal scaling





VS



Characteristics of NoSQL

Non-relational

Open-source

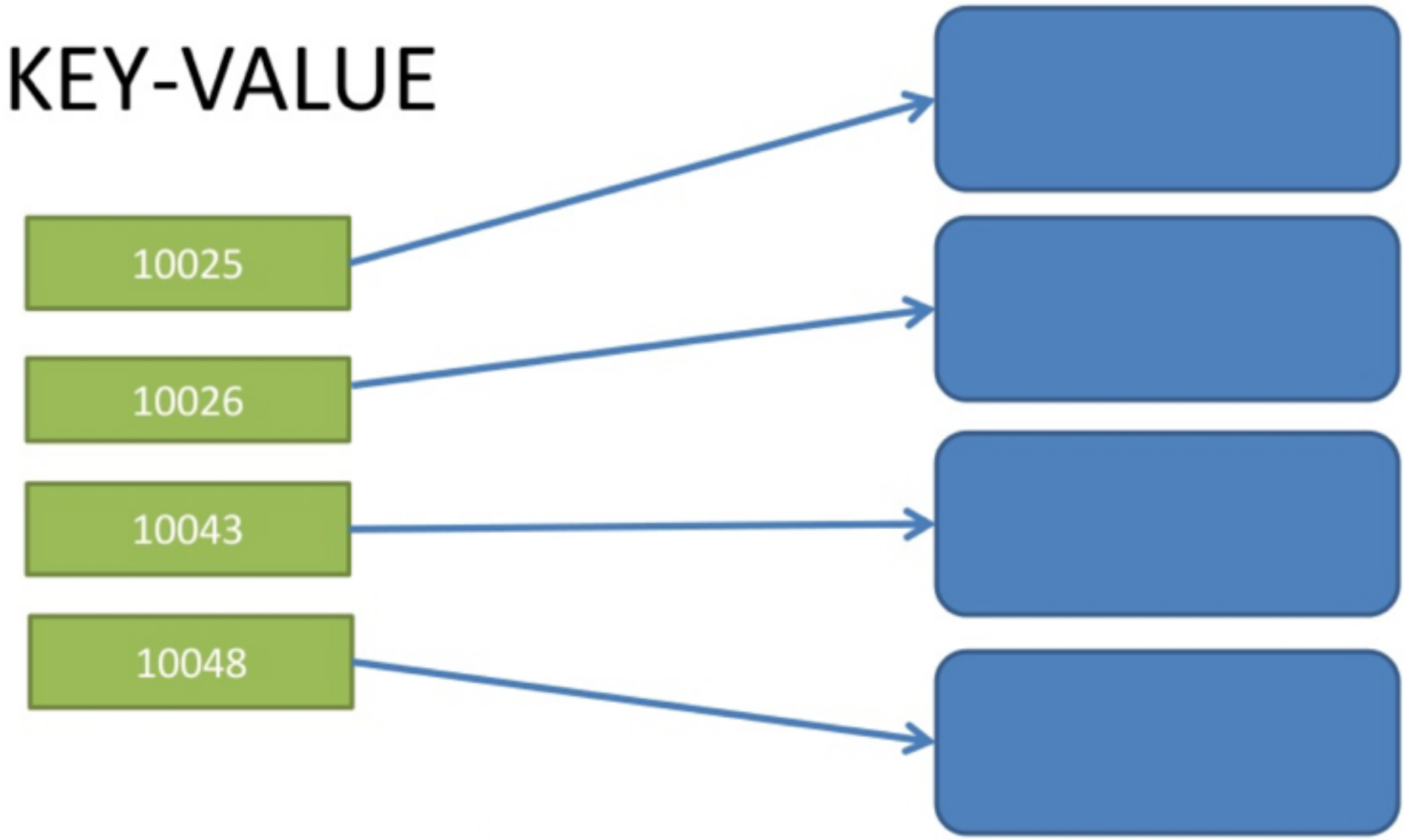
Cluster-friendly

Schema less

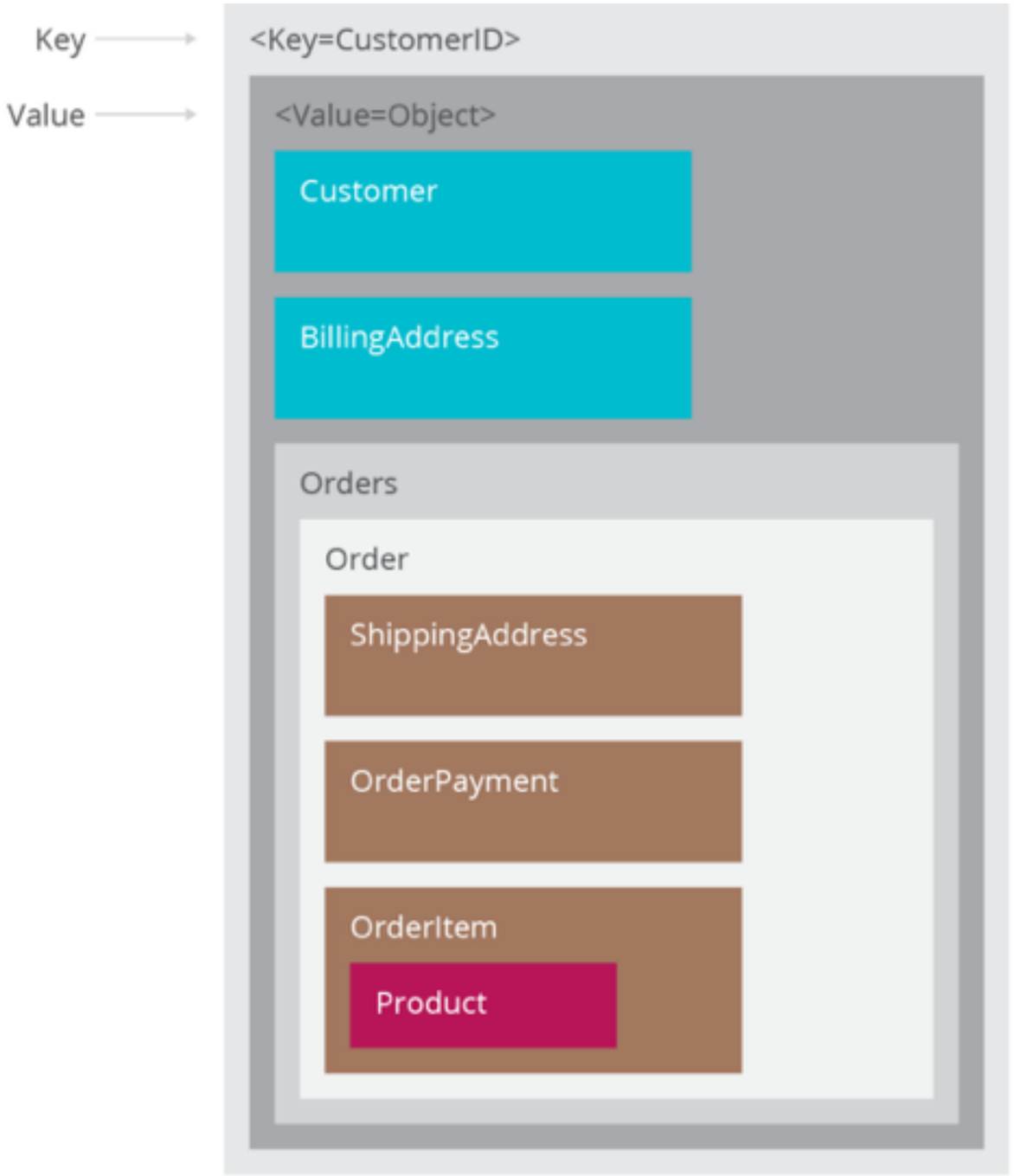
For new web app

Data model of NoSQL

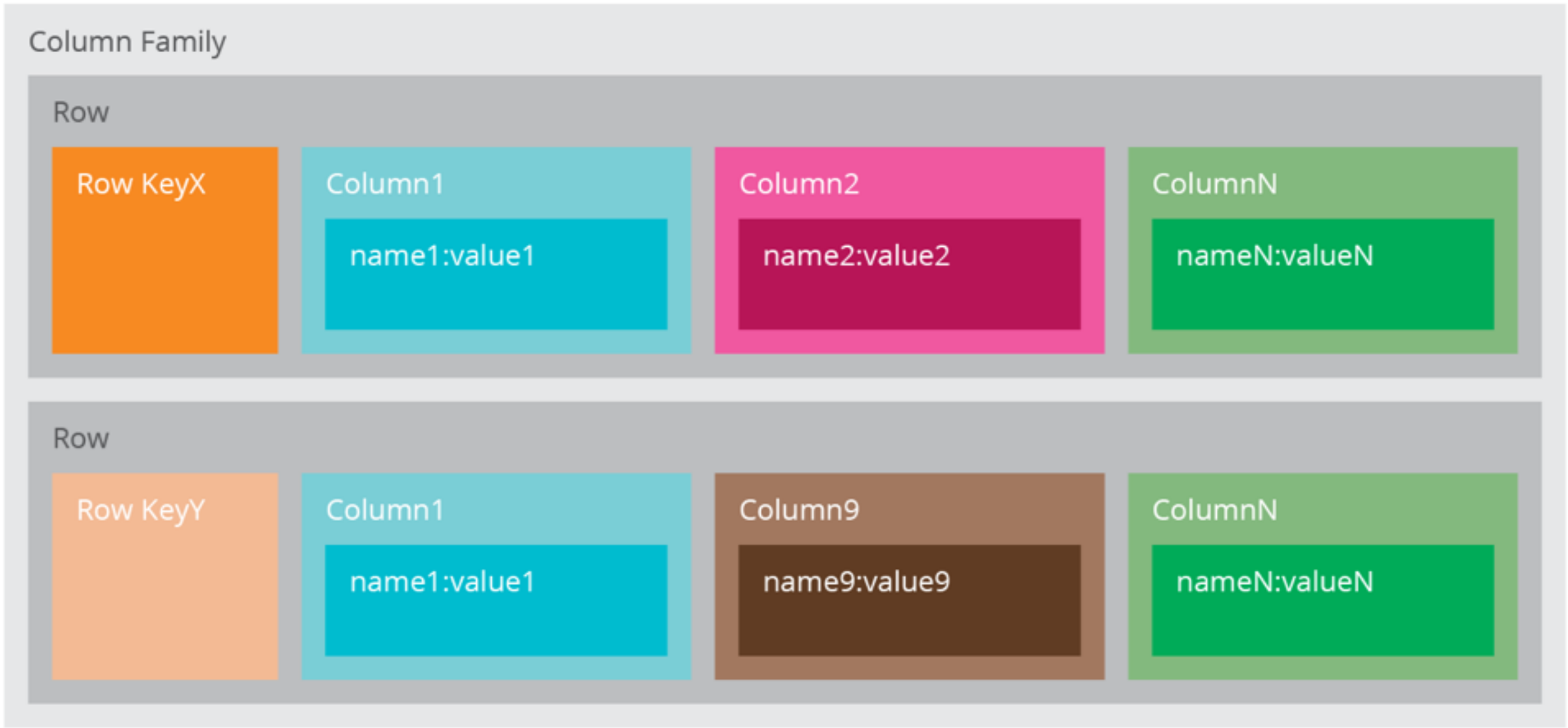
Key-value



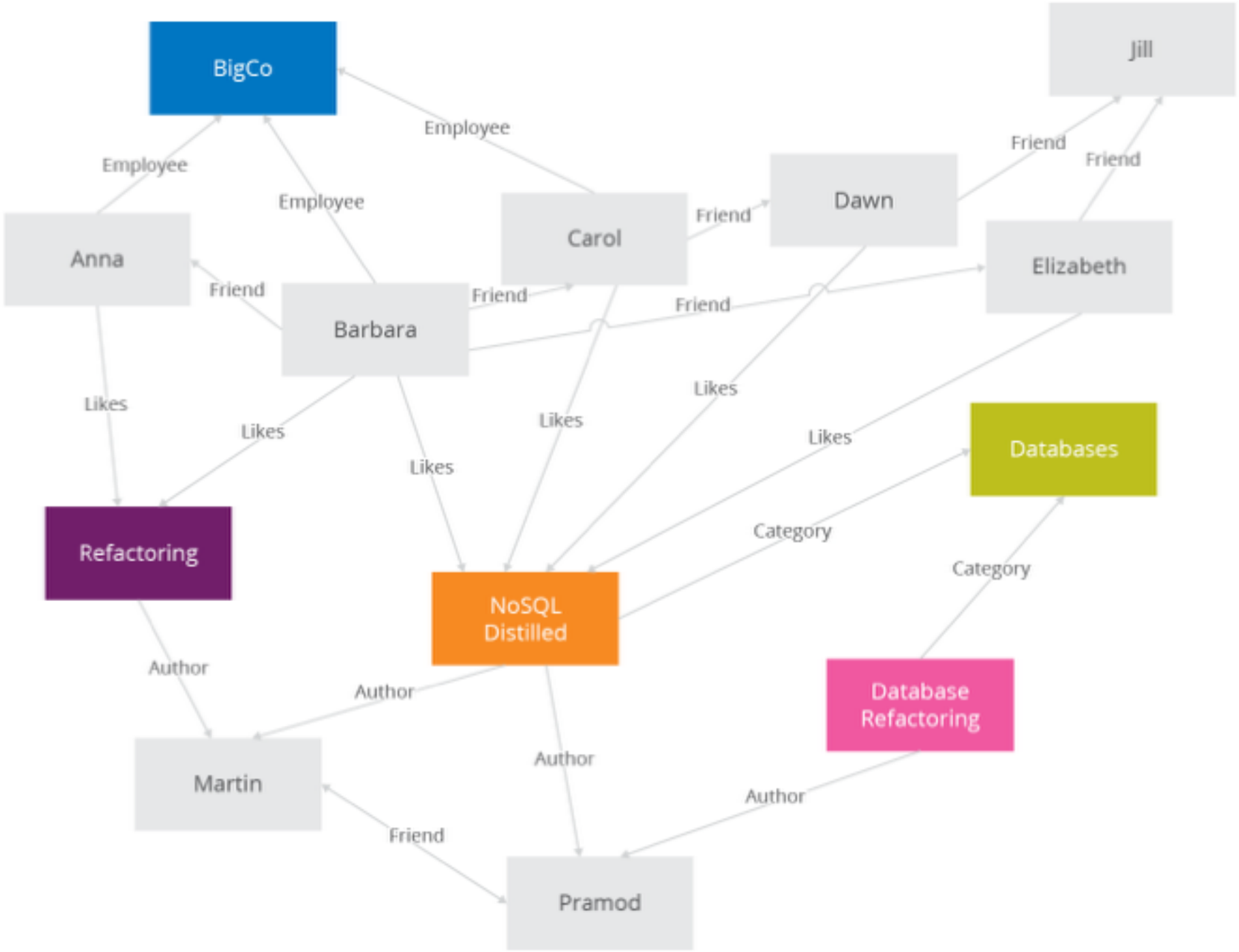
Document


Key
Value
<Key=CustomerID>
<Value=Object>
Customer
BillingAddress
Orders
Order
ShippingAddress
OrderPayment
OrderItem
Product

Column-family



Graph

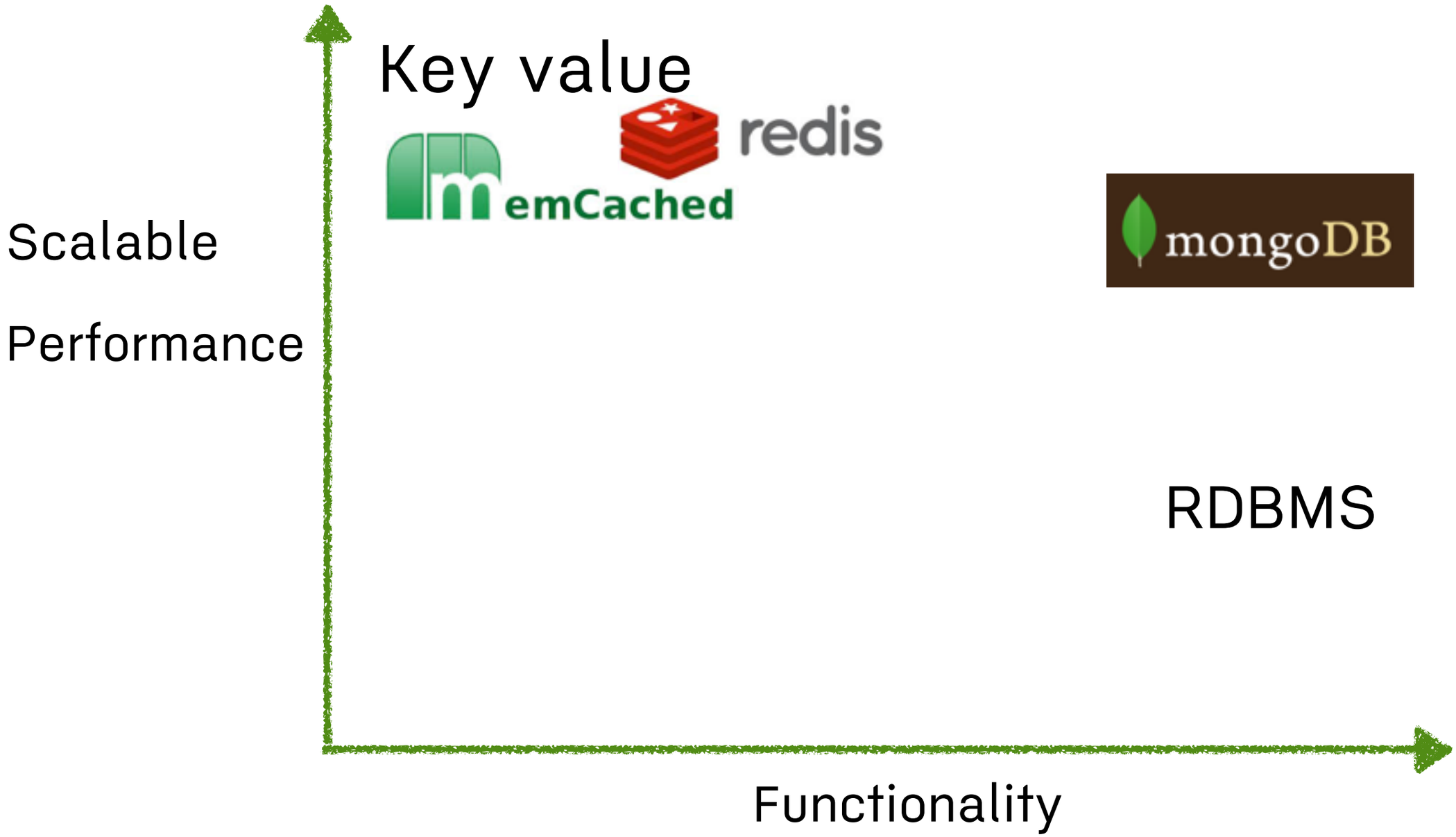

BigCo
Jill
Employee
Employee
Employee
Friend
Friend
Anna
Carol
Dawn
Friend
Elizabeth
Friend
Barbara
Friend
Likes
Likes
Likes
Likes
Likes
Likes
Databases
Refactoring
Category
Category
NoSQL
Distilled
Author
Author
Database
Refactoring
Author
Author
Martin
Friend
Pramod

Data model of NoSQL

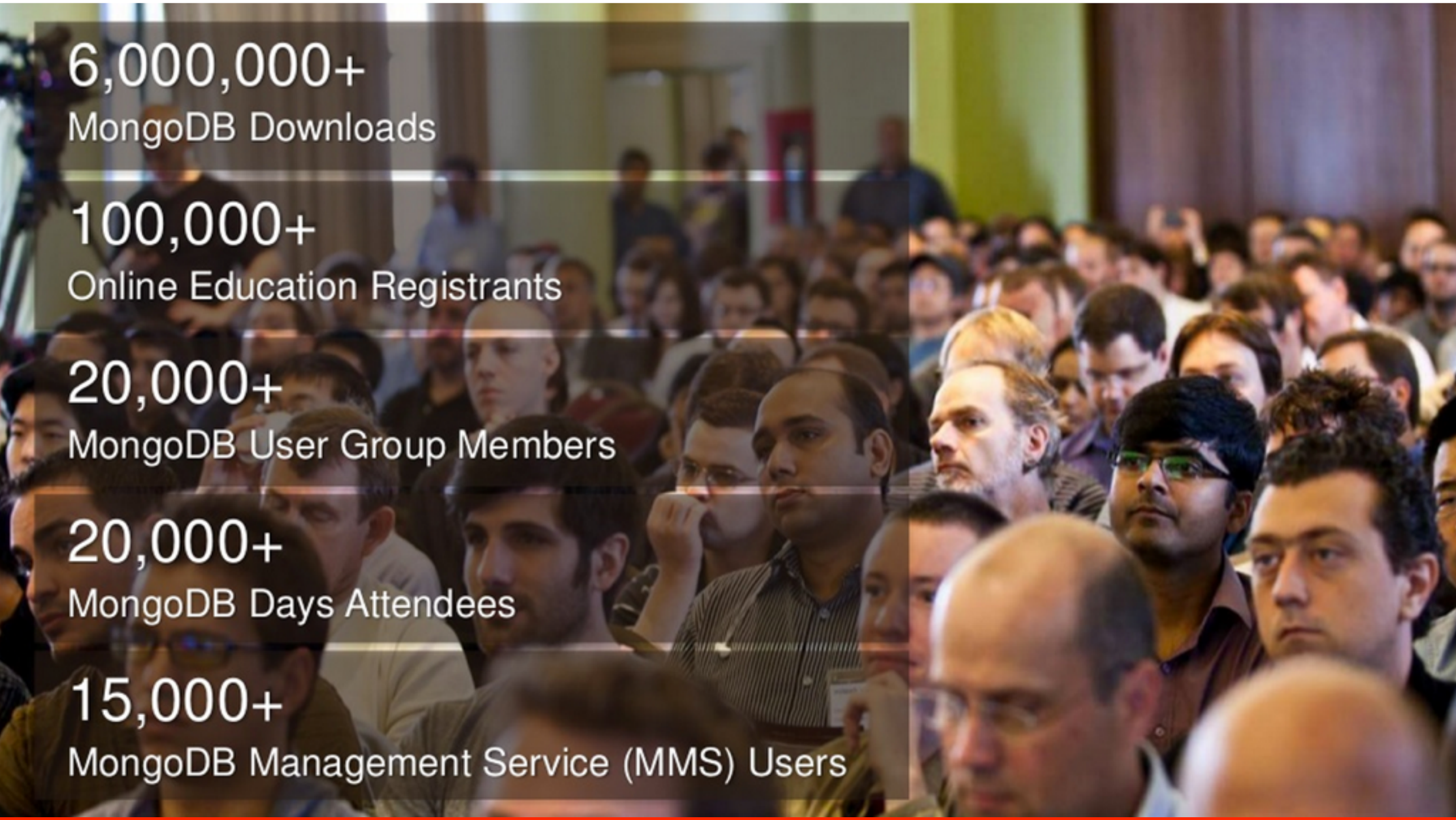
Document Database	Graph Databases
   	 
Wide Column Stores	**Key-Value Databases**
   	    

@cloudtxt http://www.aryannava.com

NoSQL


Key value
redis
emCached
mongoDB
RDBMS
Scalable
Performance
Functionality

Community of MongoDB



303 systems in ranking, April 2016

Rank Apr 2016	Rank Mar 2016	Rank Apr 2015	DBMS	Database Model	Score Apr 2016	Score Mar 2016	Score Apr 2015
1.	1.	1.	Oracle	Relational DBMS	1467.53	-4.48	+21.40
2.	2.	2.	MySQL	Relational DBMS	1370.11	+22.39	+85.53
3.	3.	3.	Microsoft SQL Server	Relational DBMS	1135.05	-1.45	-14.07
4.	4.	4.	MongoDB	Document store	312.44	+7.11	+33.85
5.	5.	5.	PostgreSQL	Relational DBMS	303.73	+4.10	+35.41
6.	6.	6.	DB2	Relational DBMS	184.08	-3.85	-13.56
7.	7.	7.	Microsoft Access	Relational DBMS	131.97	-3.06	-10.22
8.	8.	8.	Cassandra	Wide column store	129.67	-0.66	+24.78
9.	9.	↑ 10.	Redis	Key-value store	111.24	+5.02	+16.69
10.	10.	↓ 9.	SQLite	Relational DBMS	107.96	+2.19	+5.67
11.	11.	↑ 14.	Elasticsearch	Search engine	82.58	+2.41	+17.92
12.	12.	↓ 11.	SAP Adaptive Server	Relational DBMS	73.32	-3.33	-13.37
13.	13.	13.	Teradata	Relational DBMS	72.26	-1.81	+2.00
14.	14.	↓ 12.	Solr	Search engine	66.02	-3.35	-15.98
15.	15.	15.	HBase	Wide column store	51.49	-0.92	-9.65
16.	16.	↑ 17.	Hive	Relational DBMS	49.08	-1.43	+6.33
17.	17.	↓ 16.	FileMaker	Relational DBMS	46.10	-1.83	-5.72

http://db-engines.com/en/ranking_trend

One size fit all !!

What is JSON ?

JavaScript Object Notation

Key-value

JSON document

```
{
   “name”: “Somkiat”,
   “age”: 30,
   “province”: “Bangkok”
   “country”: “Thailand”
}
```

JSON document

```
{
   “name”: “MongoDB book,
   “page”: 300,
   “author”: [“author 1”, “author 2”]
}
```

Schema less

Two documents don’t have the same schema !!

MongoDB Book

```
{
   “name”: “MongoDB book,
   “page”: 300,
   “author”: [“author 1”, “author 2”]
}
```

NoSQL Book

```
{
   “name”: “NoSQL book,
   “page”: 200,
   “author”: [“author 1”, “author 2”],
   “discount”: 100
}
```

Installation

https://www.mongodb.com/download-center?jmp=nav#community

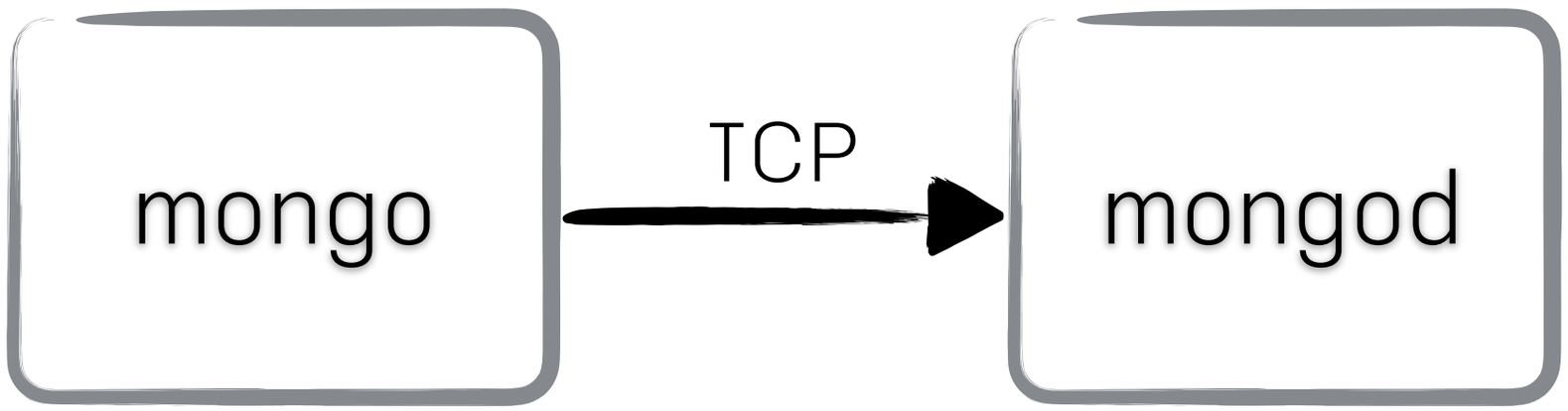
Supported platforms

Platform	3.2	3.0	2.6	2.4	2.2
Amazon Linux	✓	✓	✓	✓	✓
Debian 7	✓	✓	✓	✓	✓
Fedora 8+			✓	✓	✓
RHEL/CentOS 6.2+	✓	✓	✓	✓	✓
RHEL/CentOS 7.0+	✓	✓	✓		
SLES 11	✓	✓	✓	✓	✓
SLES 12	✓				
Solaris 64-bit	✓	✓	✓	✓	✓
Ubuntu 12.04	✓	✓	✓	✓	✓
Ubuntu 14.04	✓	✓	✓		
Microsoft Azure	✓	✓	✓	✓	✓
Windows Vista/Server 2008R2/2012+	✓	✓	✓	✓	✓
OSX 10.7+	✓	✓	✓	✓	

Let’s start

mongod = MongoDB database

mongo = MongoDB shell



Start MongoDB

$mongod --dbpath d:\mongodb\data

```
[initandlisten]
[initandlisten] ** WARNING: soft rlimits too low. Number o

 [HostnameCanonicalizationWorker] Starting hostname canoni

 [initandlisten] Initializing full-time diagnostic data ca
.data'
 [initandlisten] waiting for connections on port 27017
```

Mongo shell

$mongo

```
MongoDB shell version: 3.2.6
connecting to: test
Server has startup warnings:
2016-05-11T21:59:35.256+0700 I CONTROL  [initandlisten]
2016-05-11T21:59:35.256+0700 I CONTROL  [initandlisten]
** WARNING: soft rlimits too low. Number of files is 256
, should be at least 1000
>
```

RDBMS vs MongoDB concept

RDBMS	MongoDB
database	database
table	collection
row	document/BSON
column	field
index	index
table joins	embedded documents and linking
primary key	primary key
aggregation	aggregation pipeline

Create database

```
>use store
```

Drop database

```
>db.dropDatabase()
```

Insert new document

```
>db.book.insert(
   {
       "name": "Book 01",
       "page": 250
   }
)
```

Count all documents

```
>db.book.count()
```

Query for all documents

```
>db.book.find()
```

Query by a field

>db.book.find({"name": "Book 01"})

Count by query

```
>db.book.find( {"name": "Book 01"} ).count()
```

Query with operator

```
>db.book.find( {"page":{ $gt: 200} } )
>db.book.find( {"page":{ $lt: 250} } )
```

Query with limit

>db.book.find().limit(2)

Query with skip

```
>db.book.find().limit(1).skip(1)
```

Sort query results

```
>db.book.find().sort({"page": 1})
>db.book.find().sort({"page": -1})
```

Update a document

```
>db.book.update(
    { "name" : "Book 01" },
    {
      $set: { "author": "Somkiat" },
      $currentDate: { "lastModified": true }
    }
)
```

Update a document

```
>db.book.update(
    { "name" : "Book 01" },
      {
        $set: { "author": "Somkiat" },
        $currentDate: { "lastModified": true }
      }
)
```

Update a document

```
>db.book.update(
    { "name" : "Book 01" },
    {
      $set: { "author": "Somkiat" },
      $currentDate: { "lastModified": true }
    }
)
```

Update multiple documents

```
>db.book.update(
    {"page": { $gt: 200 }},
    {
      $set: { "author": "New" },
      $currentDate: { "lastModified": true }
    },
    { multi: true}
)
```

Replace a document

```
>db.book.update(
    { "name" : "Book 01" },
    {
      "name": "Update book 01",
      "page": 500
    }
)
```

Remove all documents

```
>db.book.remove( { } )
```

Remove some documents

```
>db.book.remove({"name": "Book 02"})
```

Remove only one document

>db.book.remove({ }, { **justOne: true** })

Drop a collection

```
>db.book.drop()
```

Workshop

Create a post

Add a comment to post

01_blog.txt

Create database

```
>use blog
```

Create a new post

```
>post =
{    _id: "1",
     topic: "Hello my blog",
     text: "this is my first blog",
     tag: [ "intro", "mongodb" ],
     author: "somkiat",
     date: new Date()
}

>db.posts.insert(post)
```

Add a comment to post

```
>comment =
{    text: "Cool post!",
     author: "somkiat",
     date: new Date()
}
```

```
>db.posts.update(
    { _id: "1"},
    {
        $push: { comments : comment }
    }
)
```

Query data

```
>db.posts.find()
>db.posts.find().pretty()
```

Workshop with query

Post by author

Last 10 posts

Posts from 1 May 2016

Posts with a tag

Paging

Post by author

```
>db.posts.find( {author:"somkiat"} )
```

Last 10 posts

```
>db.posts.find().sort( {date: -1} ).limit(10)
```

Posts from 1 May 2016

```
>may_1_2016 = new Date(2016, 5, 1)

>db.posts.find(
    {
        date: { $gte:  may_1_2016}
    }
)
```

Posts with tag

```
>db.posts.find( {tag:"mongodb"} )
```

Paging

```
>page = 0
>page_size = 10

>db.posts.find().limit(page_size).skip( page *
page_size )
```

Analyze Query Performance

Posts with tag

>db.posts.find({tag:"mongodb"}).explain()


```
"queryPlanner" : {
        "plannerVersion" : 1,
        "namespace" : "blog.posts",
        "indexFilterSet" : false,
        "parsedQuery" : {
                "tag" : {
                        "$eq" : "mongodb"
                }
        },
        "winningPlan" : {
                "stage" : "COLLSCAN",
                "filter" : {
                        "tag" : {
                                "$eq" : "mongodb"
                        }
                },
                "direction" : "forward"
        },
        "rejectedPlans" : [ ]
},
```

Create new index

```
>db.posts.createIndex({ tag: 1 })
```

Posts with tag

>db.posts.find({tag:"mongodb"}).explain()


```
"winningPlan" : {
        "stage" : "FETCH",
        "inputStage" : {
                "stage" : "IXSCAN",
                "keyPattern" : {
                        "tag" : 1
                },
                "indexName" : "tag_1",
                "isMultiKey" : true,
                "isUnique" : false,
                "isSparse" : false,
                "isPartial" : false,
                "indexVersion" : 1,
                "direction" : "forward",
                "indexBounds" : {
                        "tag" : [
                                "[\"mongodb\", \"mongodb\"]"
                        ]
                }
        }
},
```


Drop index

```
>db.posts.dropIndex({ tag: 1 })
```

Aggregation

03_aggregation.txt

```
SELECT by_user, count(*)
FROM mycol
GROUP BY by_user
```

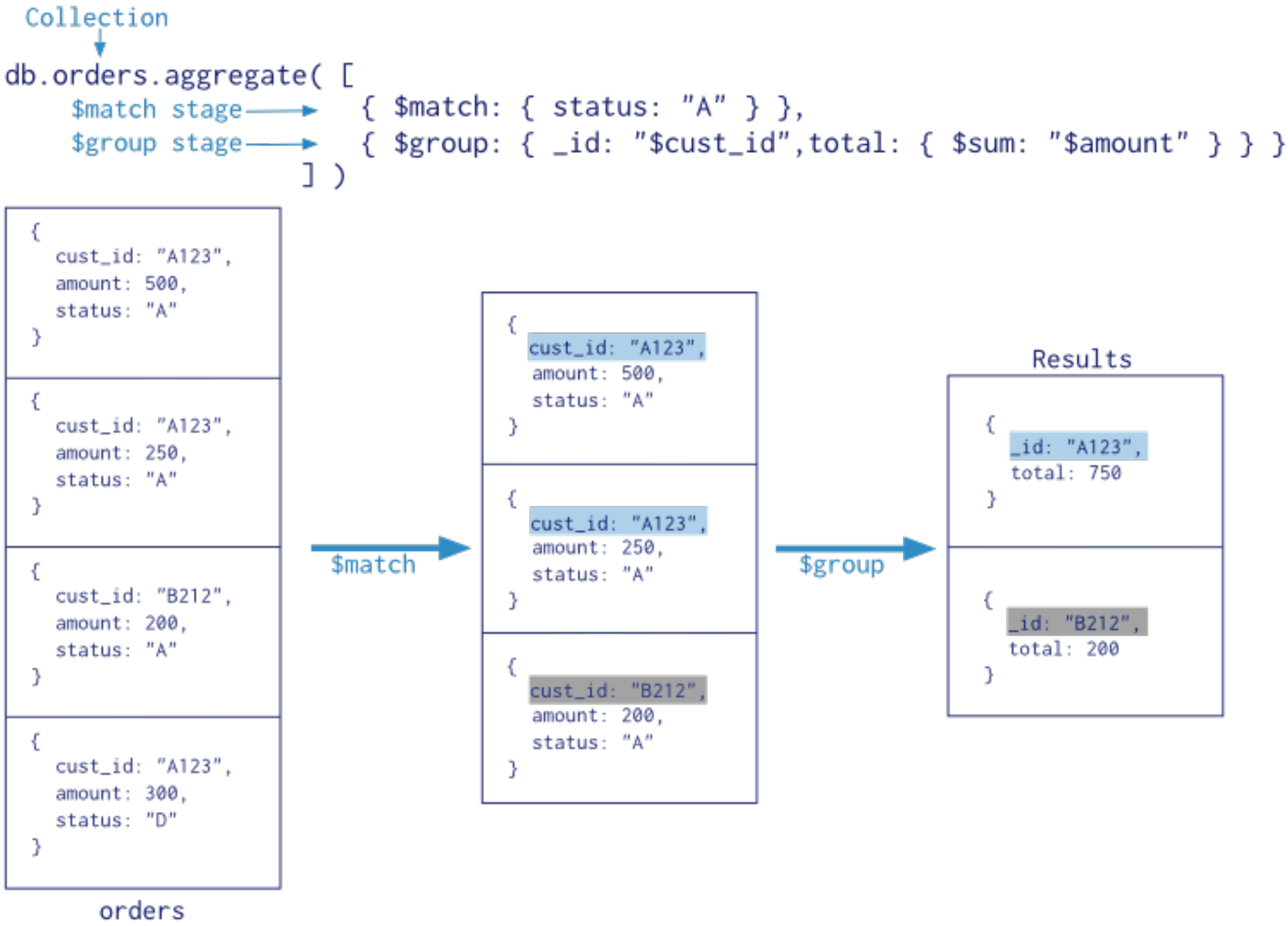
Count number of book by user

```
>db.book.aggregate(
[
   {
     $group :
       { _id : "$by_user",
         num_tutorial : {$sum : 1}
       }
   }
])
```

SQL vs Aggregation

SQL	Aggregation
SELECT	$project
WHERE	$match
GROUP BY	$group
HAVING	$match
ORDER BY	$sort
LIMIT	$limit
SUM()	$sum
COUNT()	$sum
join	$lookup

Aggregation pipeline



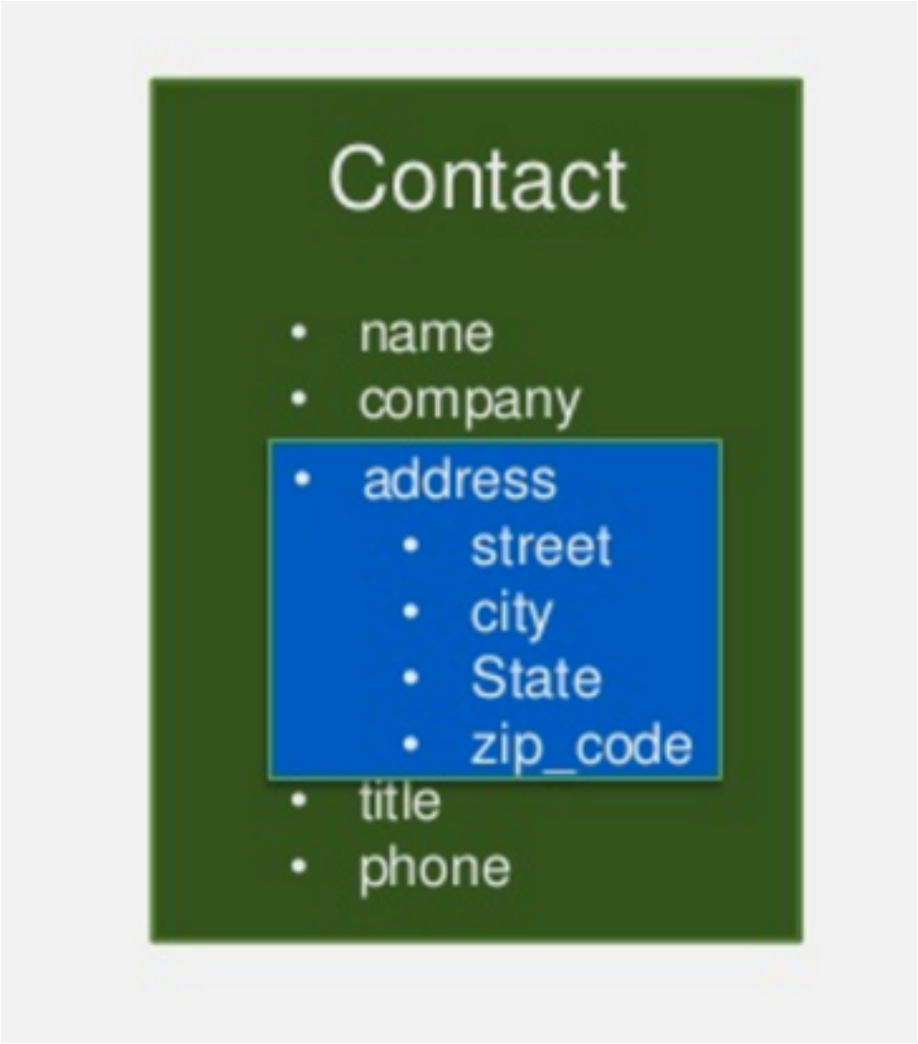
Workshop

Working with order information

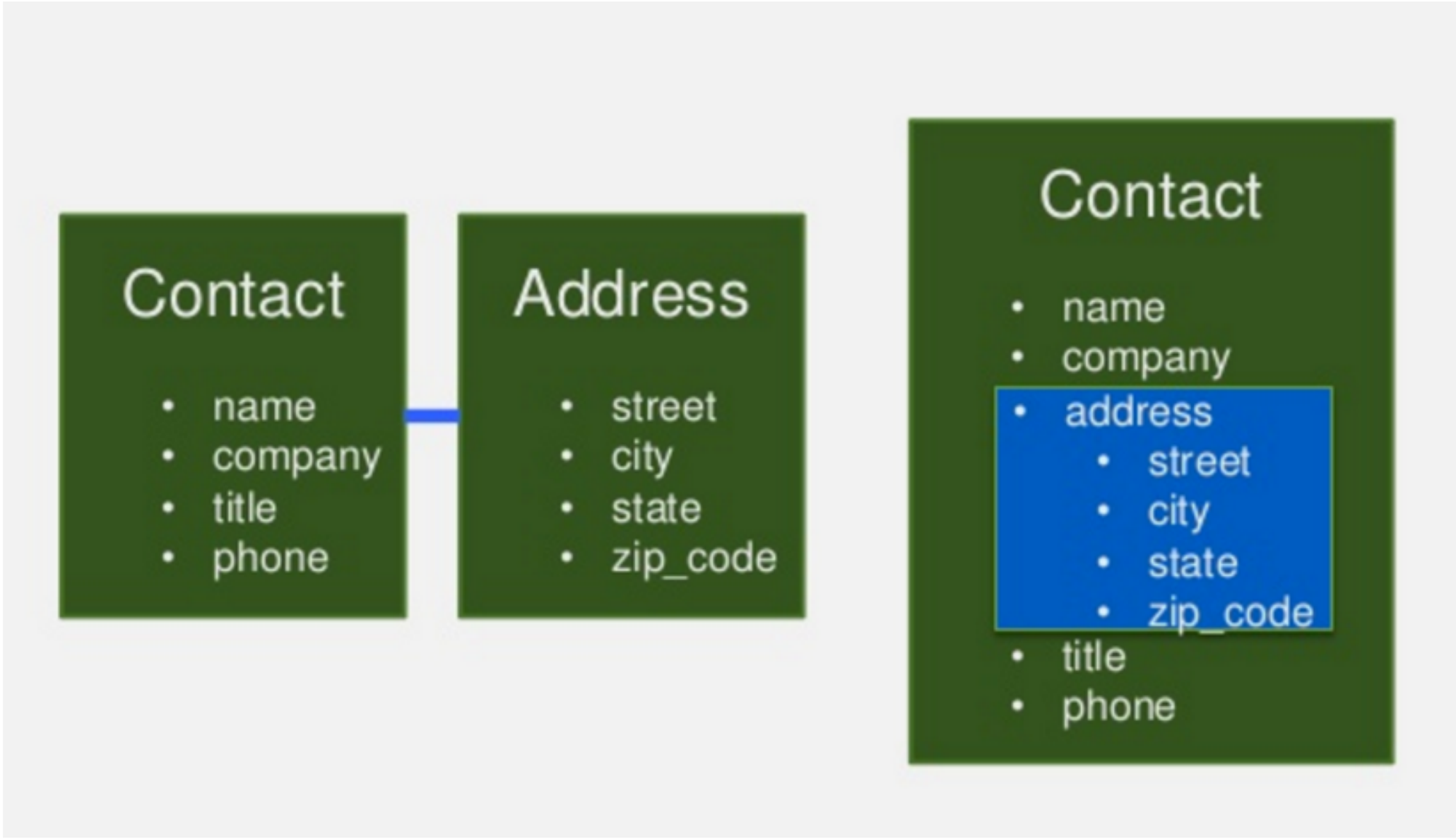
04_order.txt

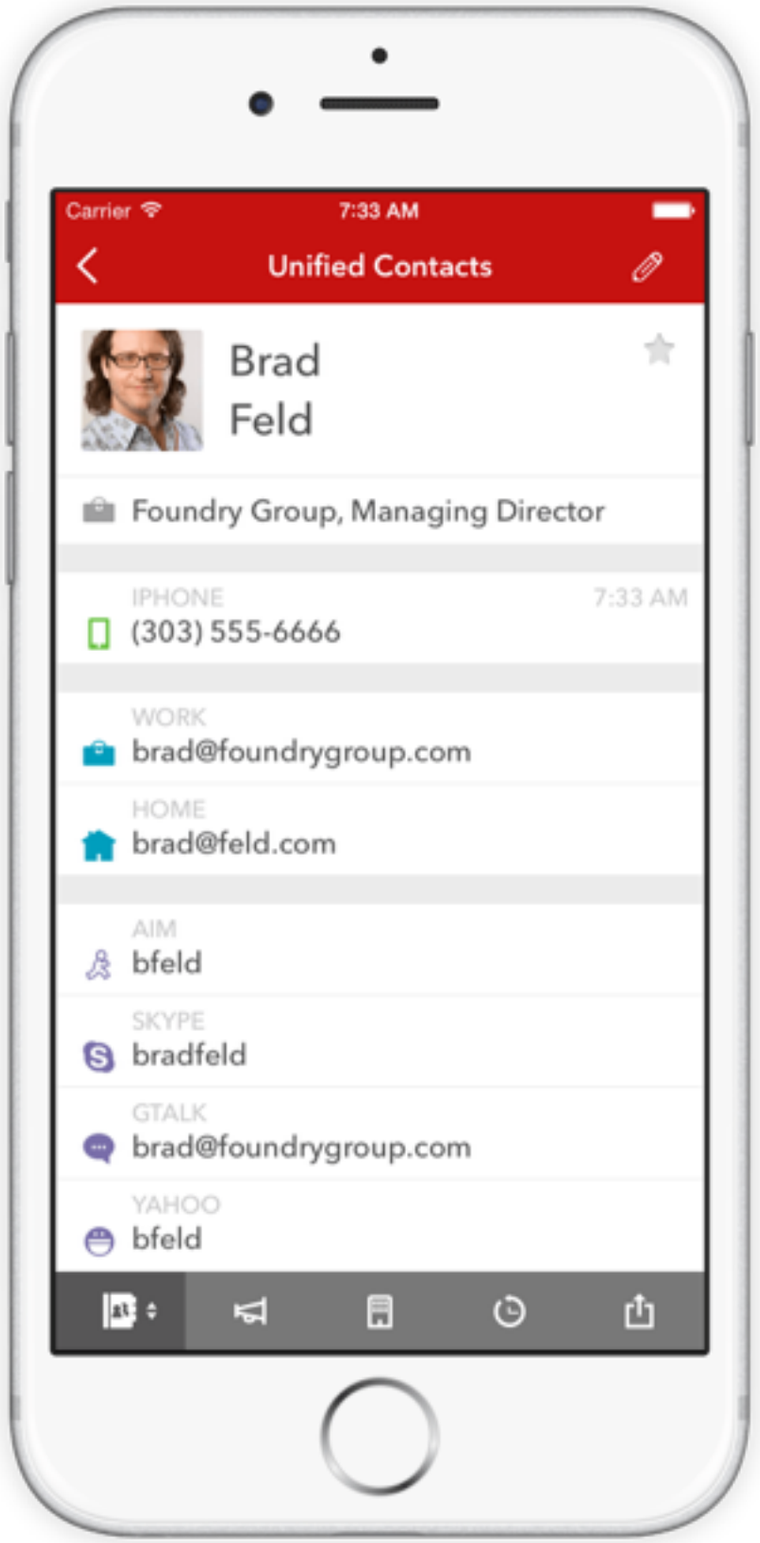
Data Modeling

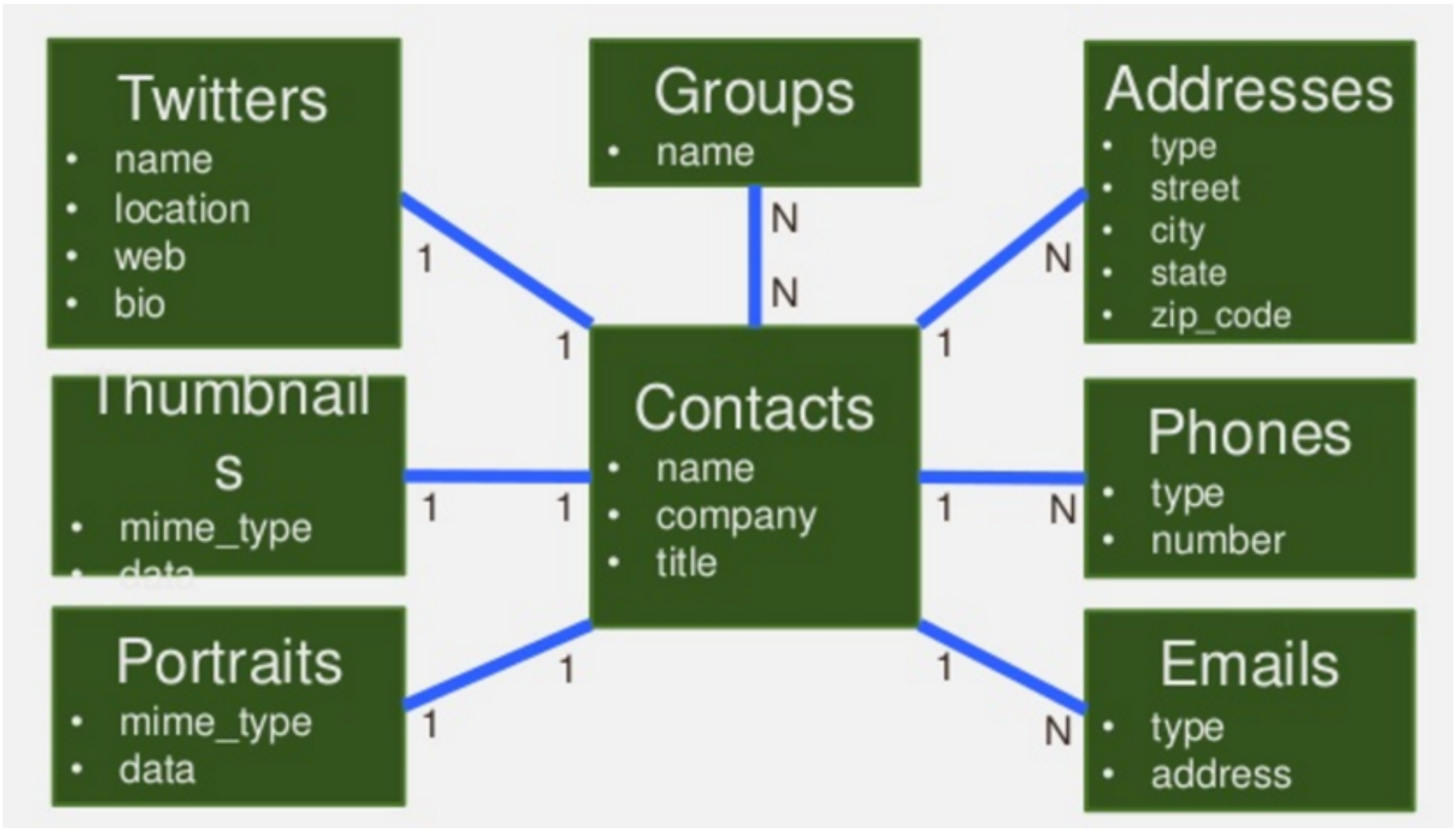
Embedding


Contact
name
company
address
street
city
State
zip_code
title
phone

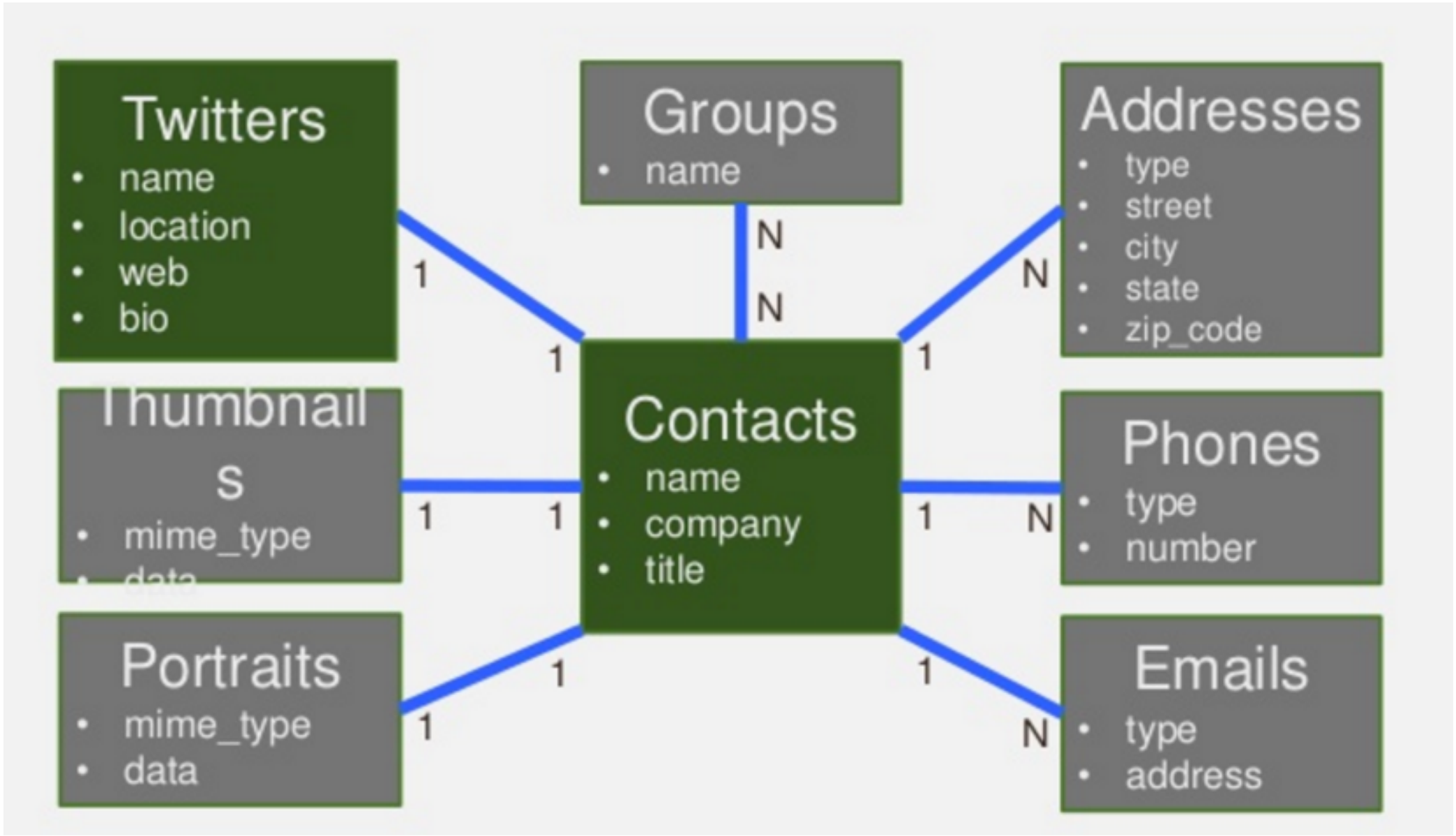
แตกต่างกันอย่างไร ? ทำไม ?




Carrier
7:33 AM
Unified Contacts
Brad
Feld
Foundry Group, Managing Director
IPHONE
7:33 AM
(303) 555-6666
WORK
brad@foundrygroup.com
HOME
brad@feld.com
AIM
bfeld
SKYPE
bradfeld
GTALK
brad@foundrygroup.com
YAHOO
bfeld

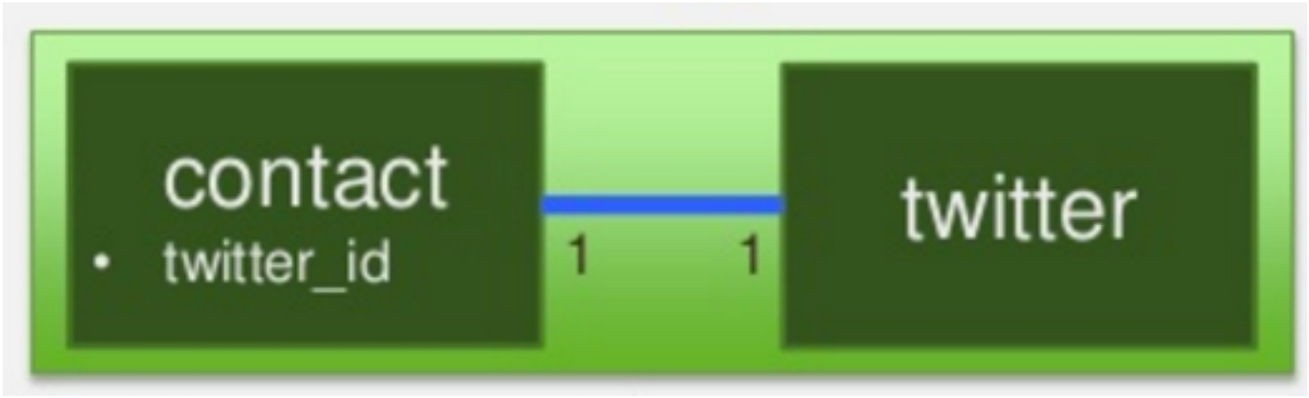

Twitters
• name
• location
• web
• bio
Groups
• name
Addresses
• type
• street
• city
• state
• zip_code
Thumbnail
s
• mime_type
• data
Contacts
• name
• company
• title
Phones
• type
• number
Portraits
• mime_type
• data
Emails
• type
• address
1
N
N
1
1
N
1
1
1
1
1
N
1
N

One-to-One

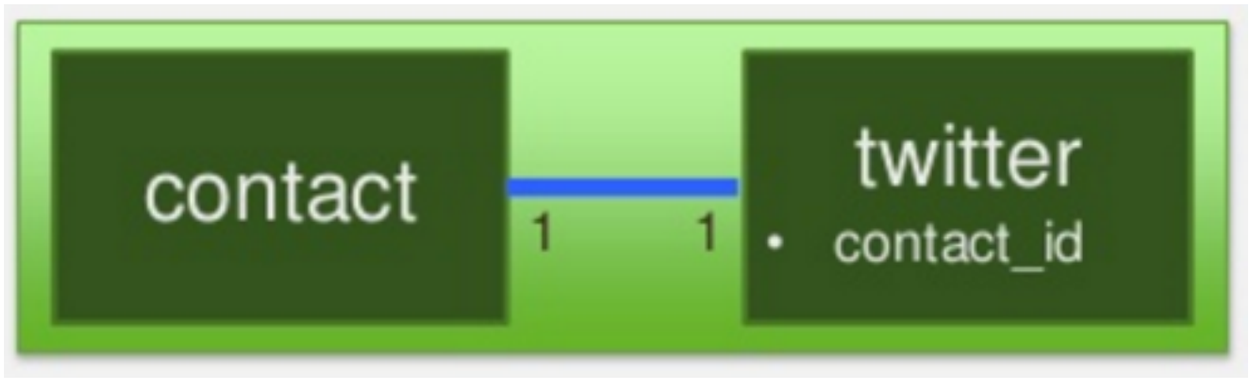

Twitters
• name
• location
• web
• bio
Groups
• name
Addresses
• type
• street
• city
• state
• zip_code
Thumbnail
s
• mime_type
Contacts
• name
• company
• title
Phones
• type
• number
Portraits
• mime_type
• data
Emails
• type
• address
1
N
N
1
1
N
1
1
1
1
1
1
1
N

ออกแบบได้อย่างไรบ้าง ?

One-to-One :: 1

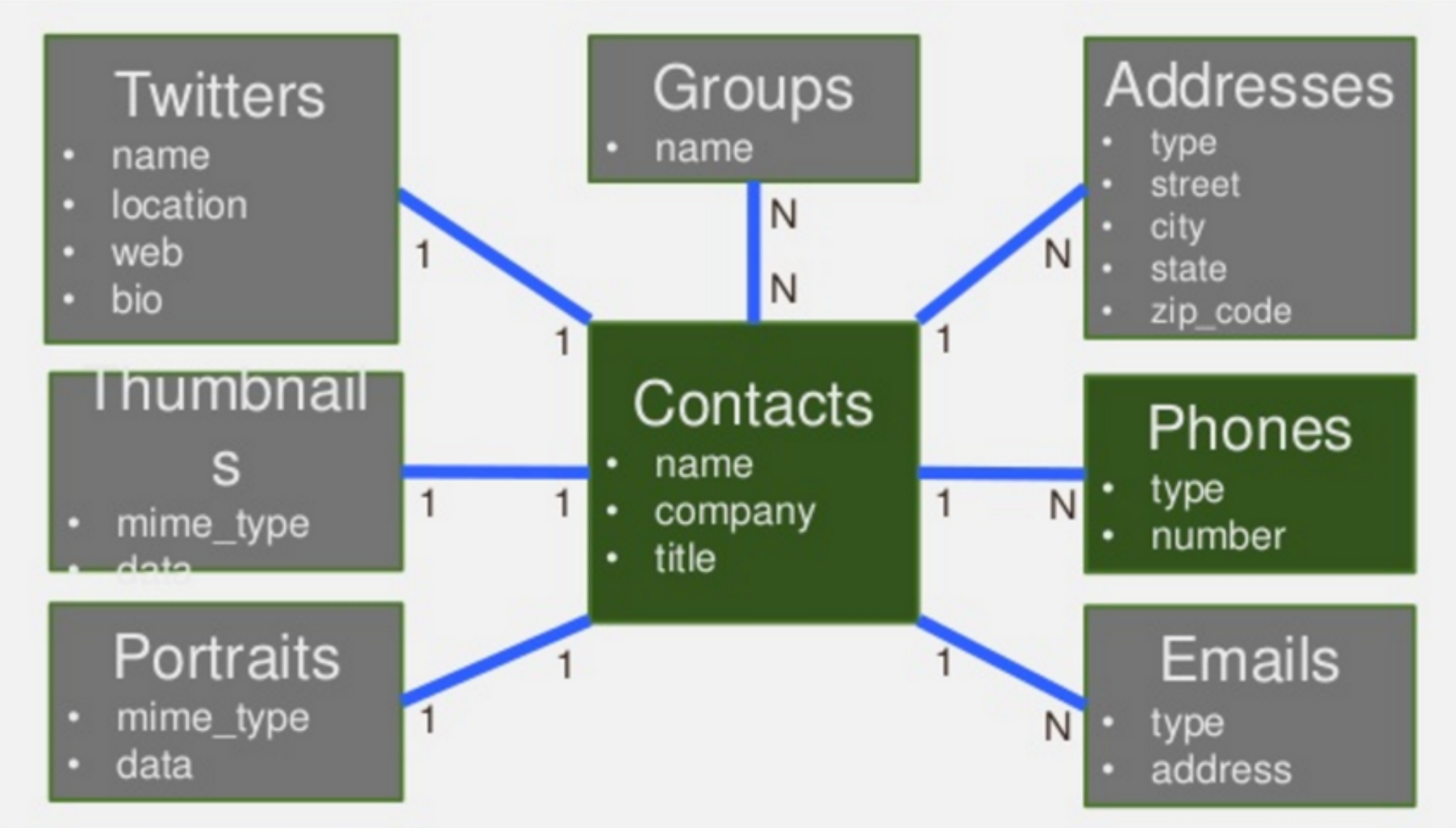


One-to-One :: 2



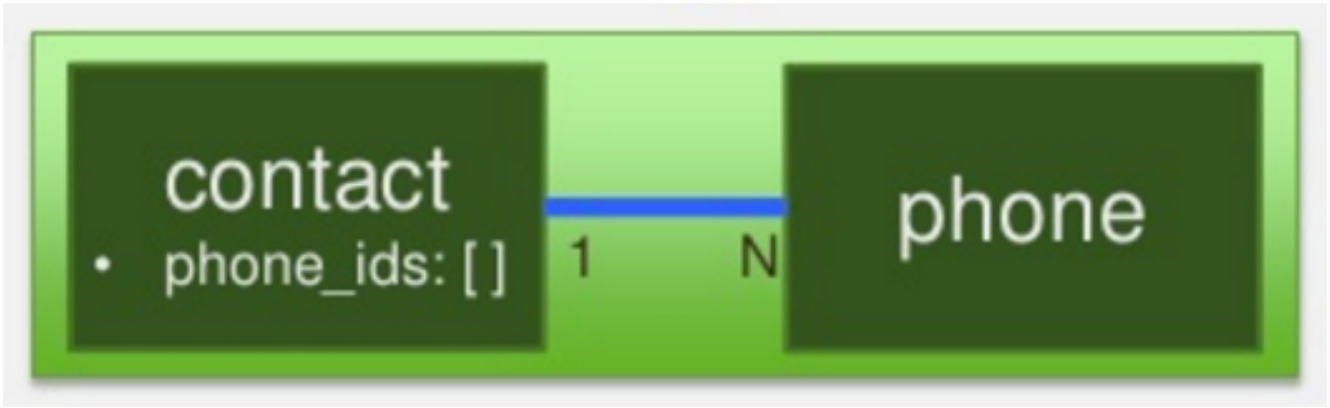
One-to-One :: 3

One-to-Many

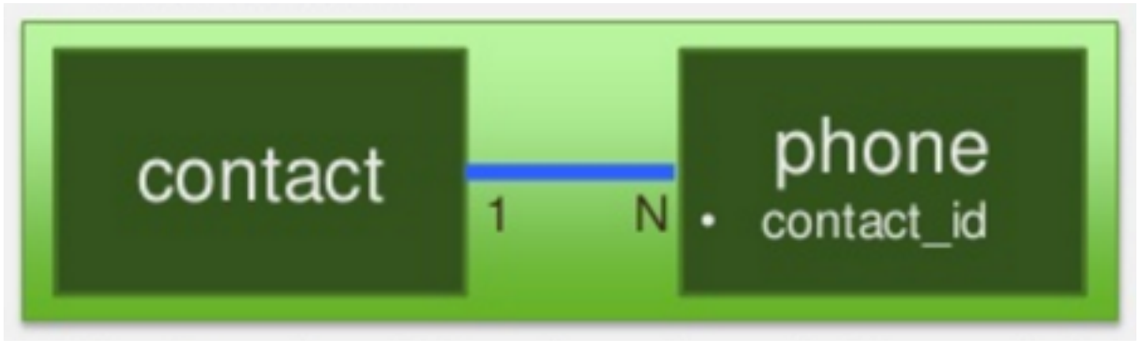

Twitters
• name
• location
• web
• bio
Groups
• name
Addresses
• type
• street
• city
• state
• zip_code
Thumbnail
s
• mime_type
• data
Contacts
• name
• company
• title
Phones
• type
• number
Portraits
• mime_type
• data
Emails
• type
• address
1
N
N
1
1
1
1
1
1
1
1
N
N
1

ออกแบบได้อย่างไรบ้าง ?

One-to-Many :: 1

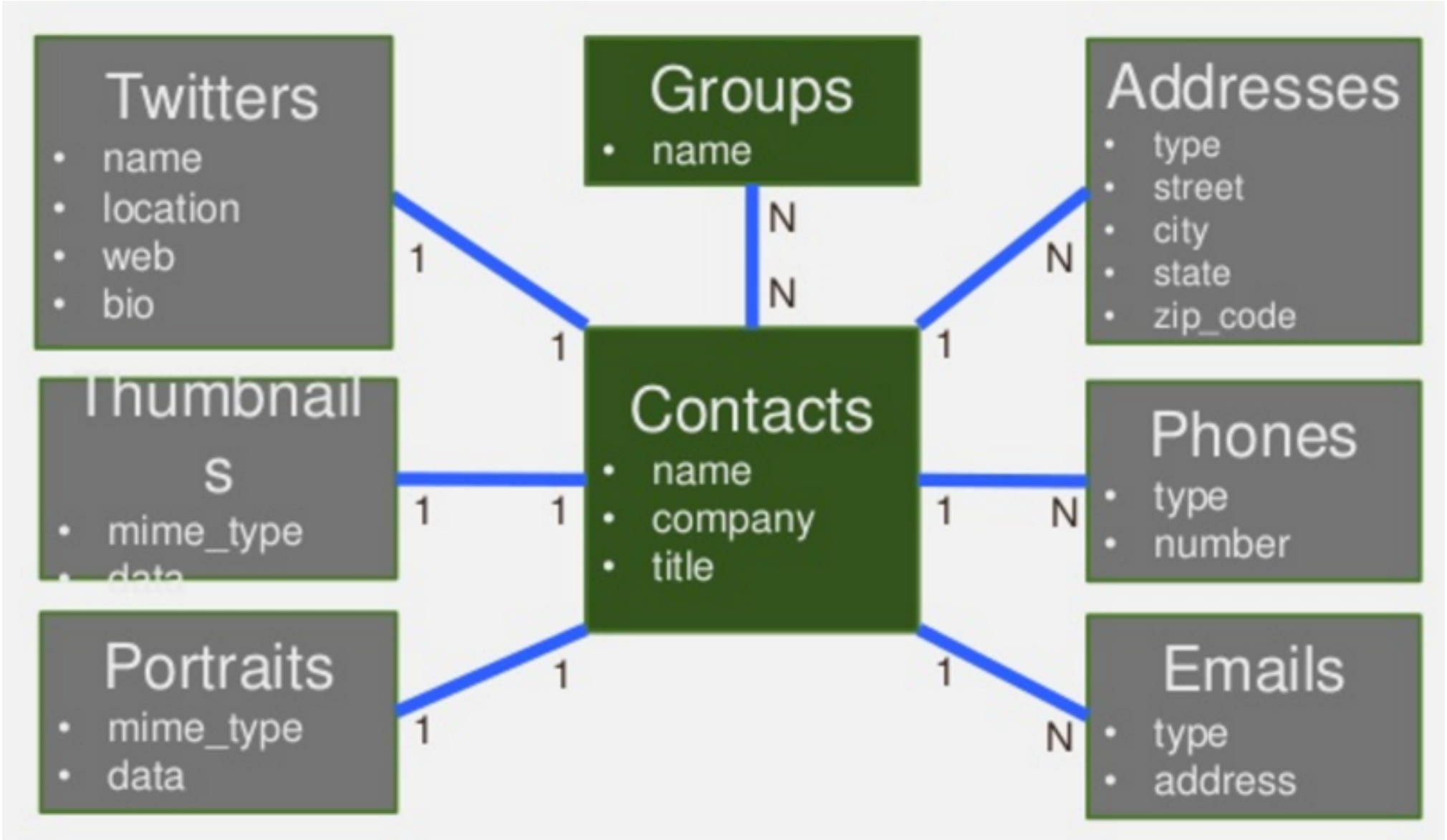


One-to-Many :: 2

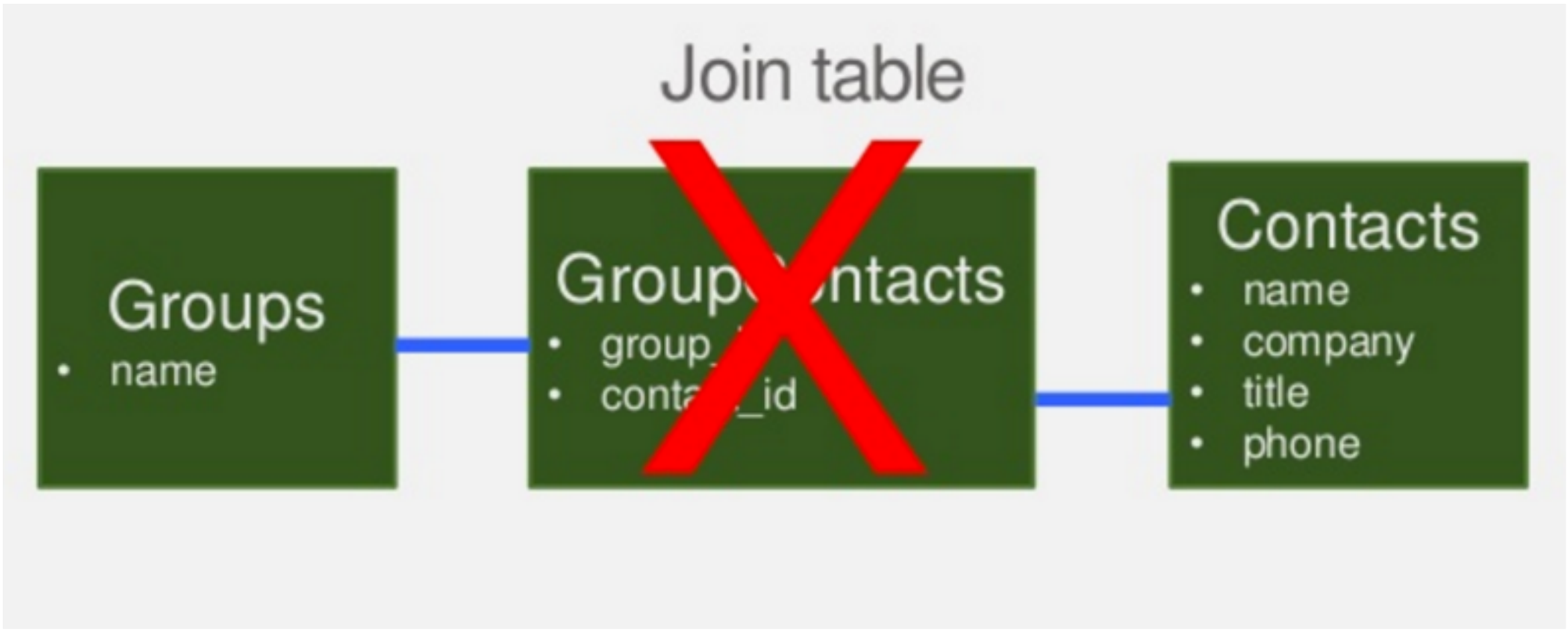


One-to-Many :: 3

Many-to-Many


Twitters
• name
• location
• web
• bio
Groups
• name
Addresses
• type
• street
• city
• state
• zip_code
Thumbnail
s
• mime_type
• data
Contacts
• name
• company
• title
Phones
• type
• number
Portraits
• mime_type
• data
Emails
• type
• address
1
N
N
1
1
1
1
1
1
1
1
N
1
N

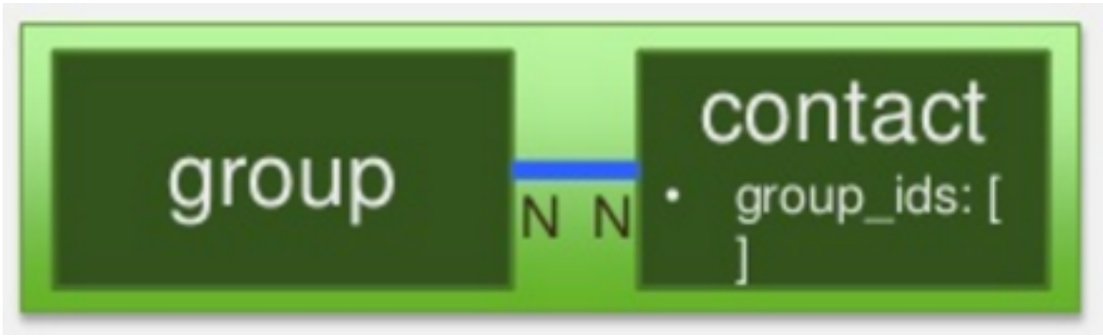
Many-to-Many


Join table
Groups
• name
Group
• group_
• conta
_id
Contacts
• name
• company
• title
• phone

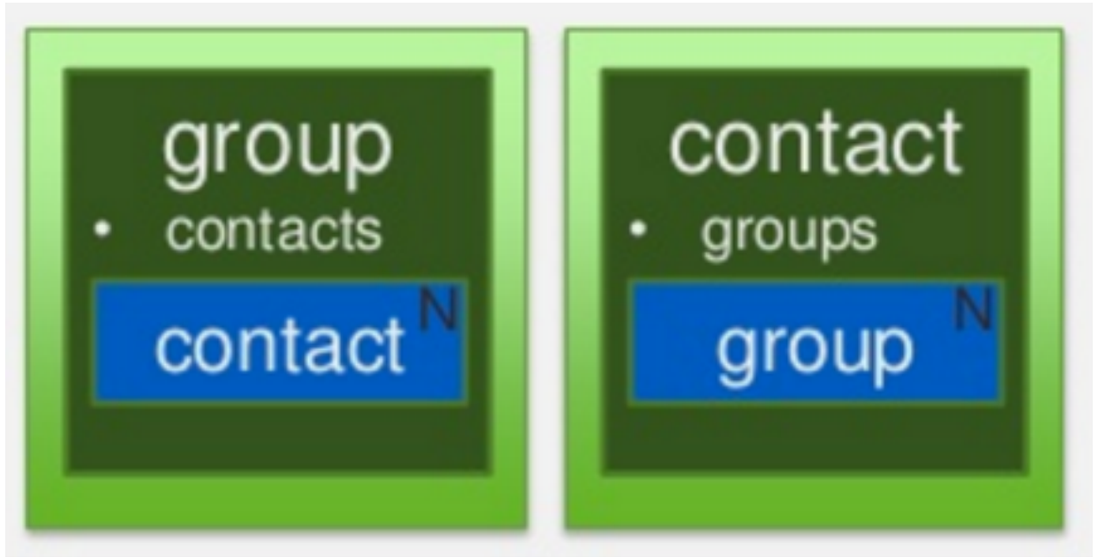
ออกแบบได้อย่างไรบ้าง ?

Many-to-Many :: 1

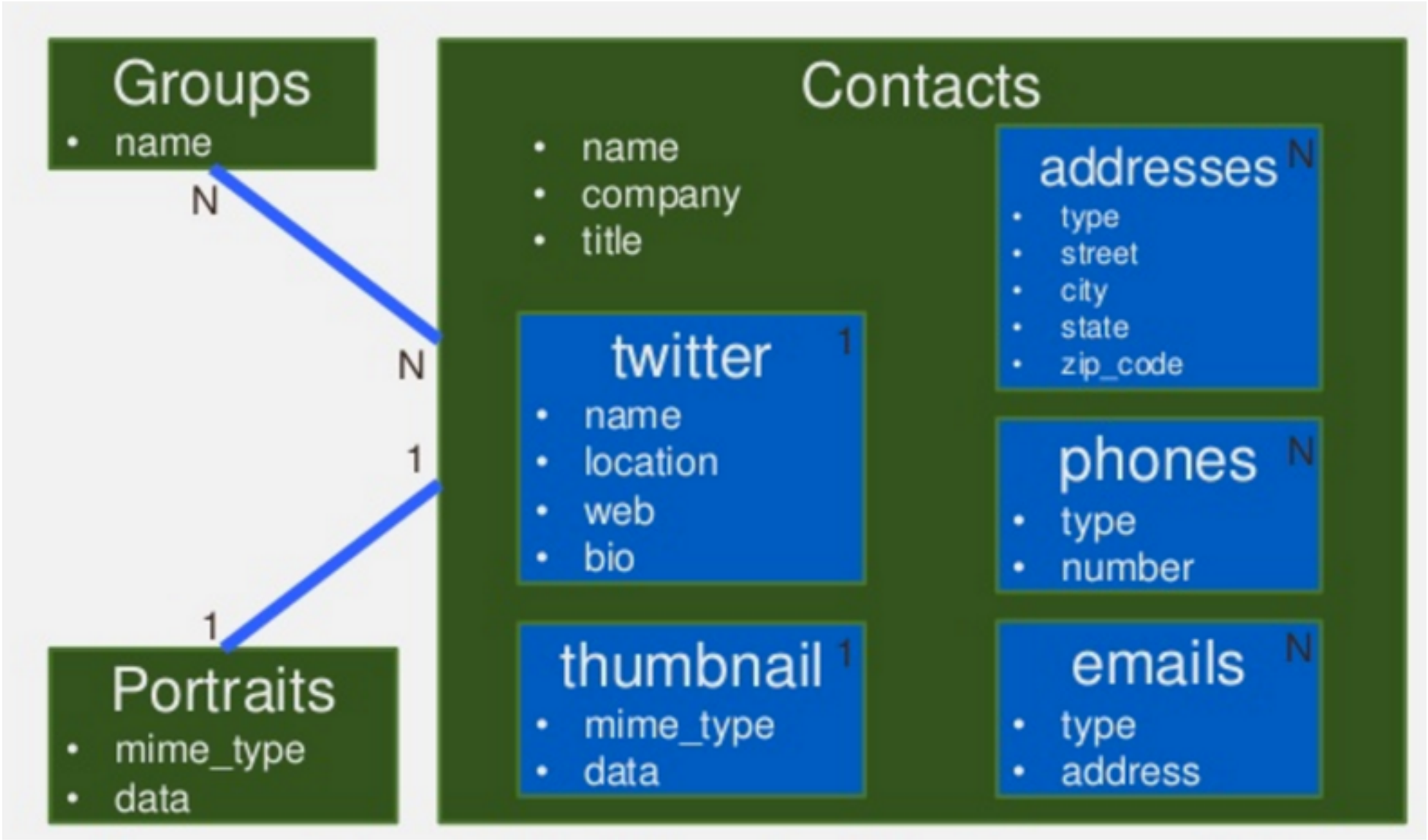
Many-to-Many :: 2



Many-to-Many :: 3



สุดท้ายแล้ว


Groups
• name
N
N
1
1
Portraits
• mime_type
• data
Contacts
• name
• company
• title
twitter
1
• name
• location
• web
• bio
thumbnail
1
• mime_type
• data
addresses
N
• type
• street
• city
• state
• zip_code
phones
N
• type
• number
emails
N
• type
• address

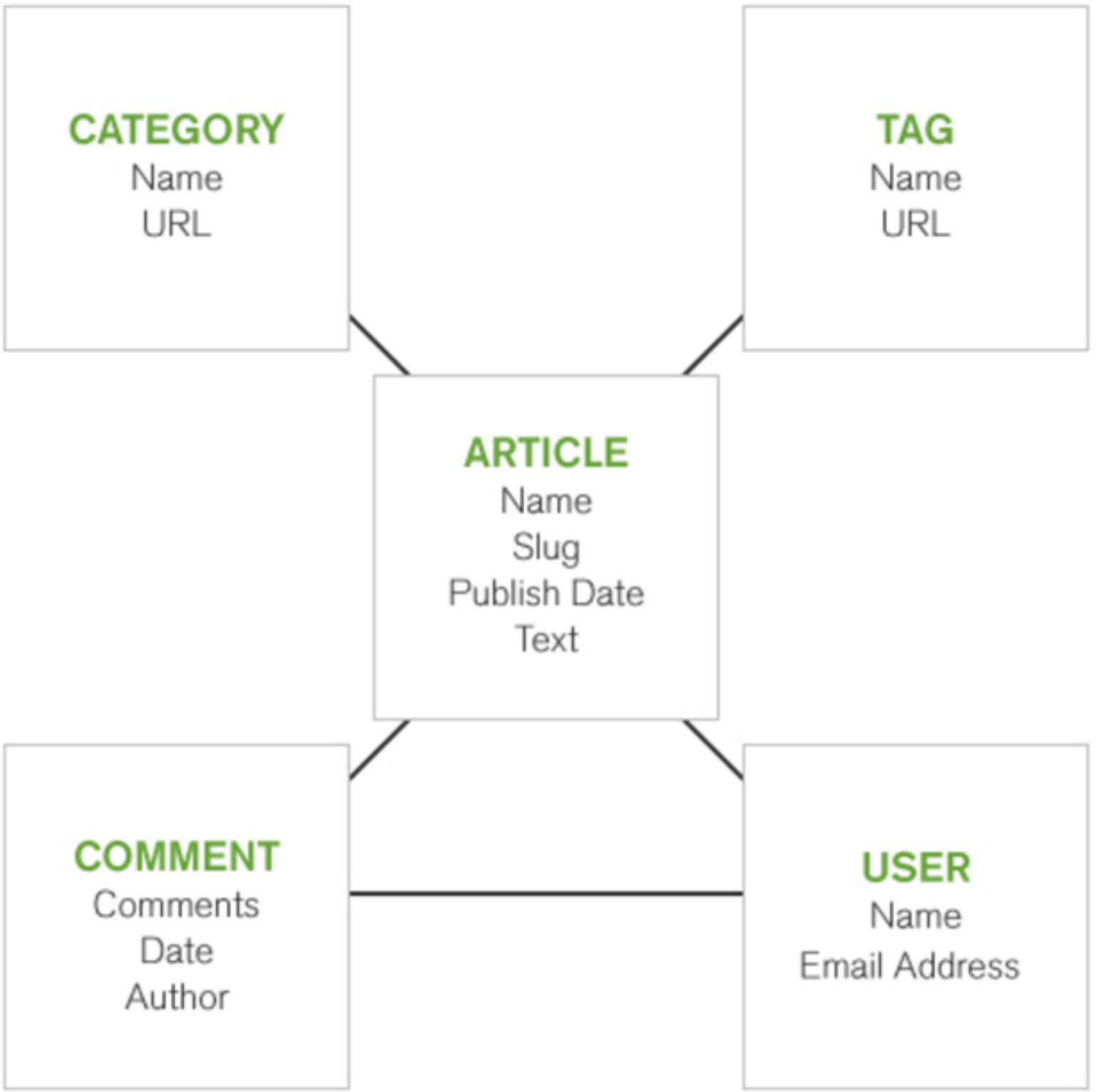
Workshop

Working with contact information

05_contact.txt

ระบบตัวอย่าง

ออกแบบด้วยแนวคิดของ RDBMS ?


CATEGORY
Name
URL
TAG
Name
URL
ARTICLE
Name
Slug
Publish Date
Text
COMMENT
Comments
Date
Author
USER
Name
Email Address

เทคนิคในการออกแบบ NoSQL

1. Denomalization

Query data volume หรือ I/O per query

vs.

Total data volume

1. Denormalization

Processing complexity

vs.

Total data volume

2. Aggregation

Minimization of one-to-many relationship

Reduction of joins

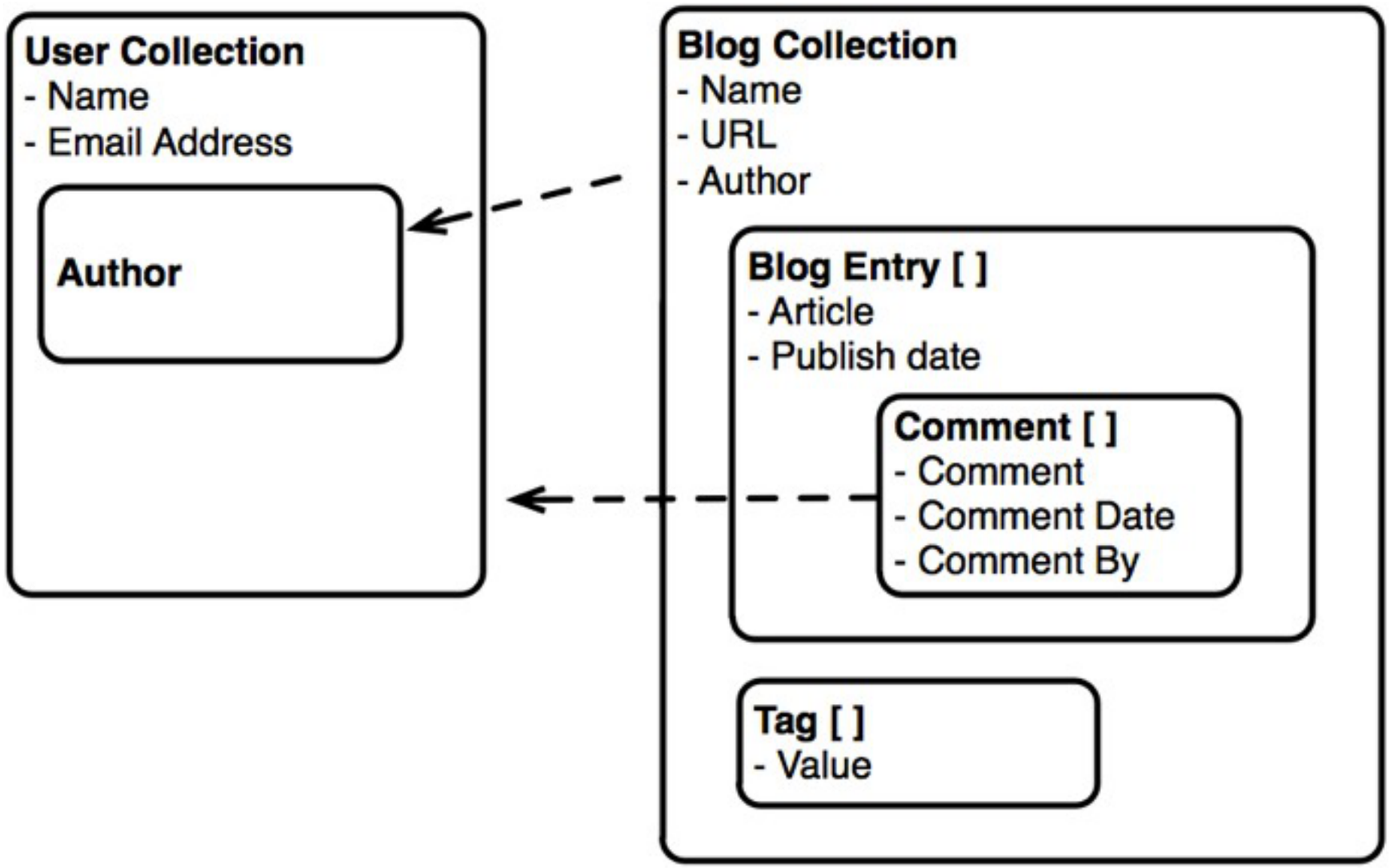
2. Aggregation

Minimization of one-to-many relationship

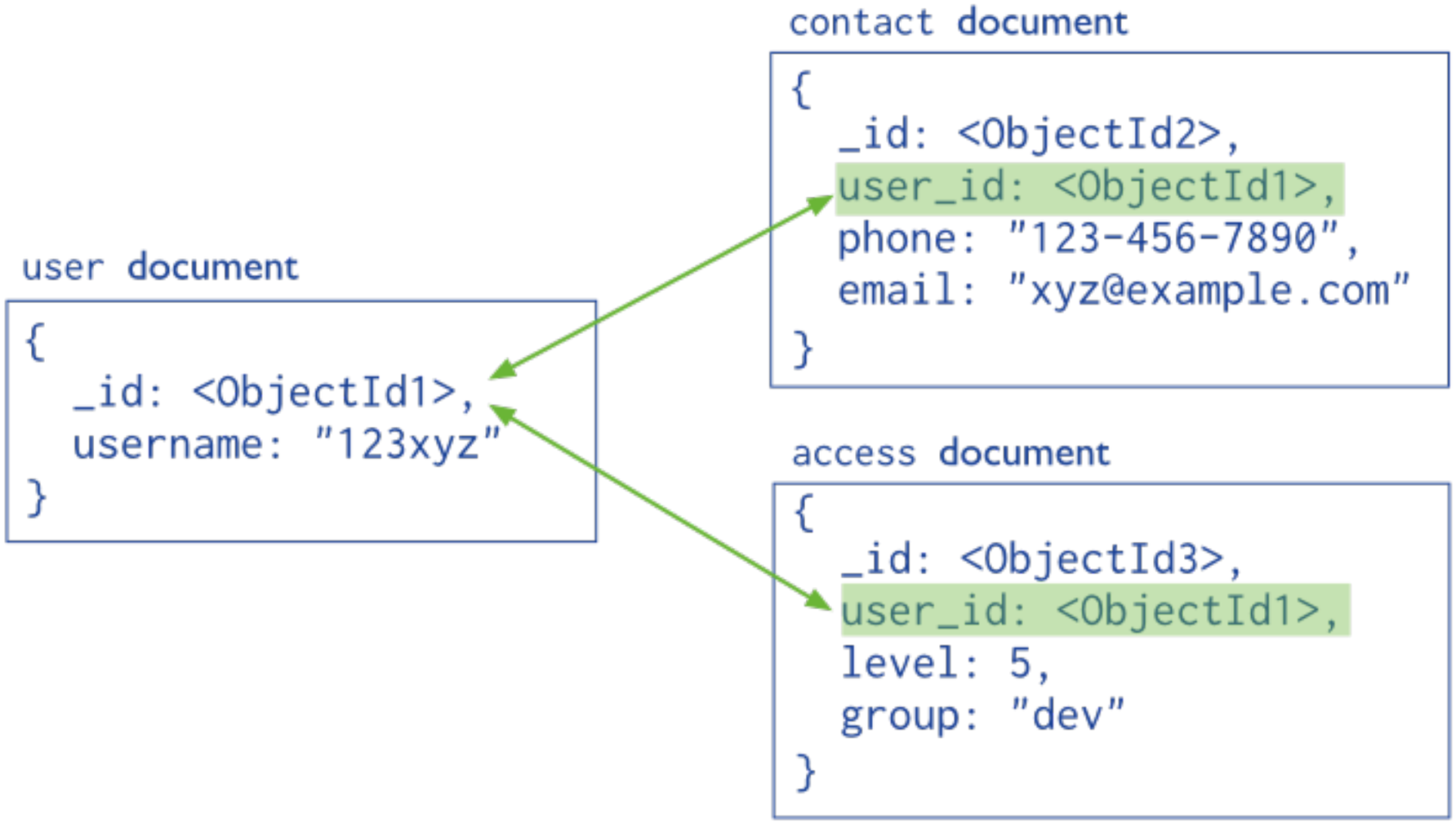
Reduction of joins

Nested or Embedded entity

Embedded document



Normalized data model



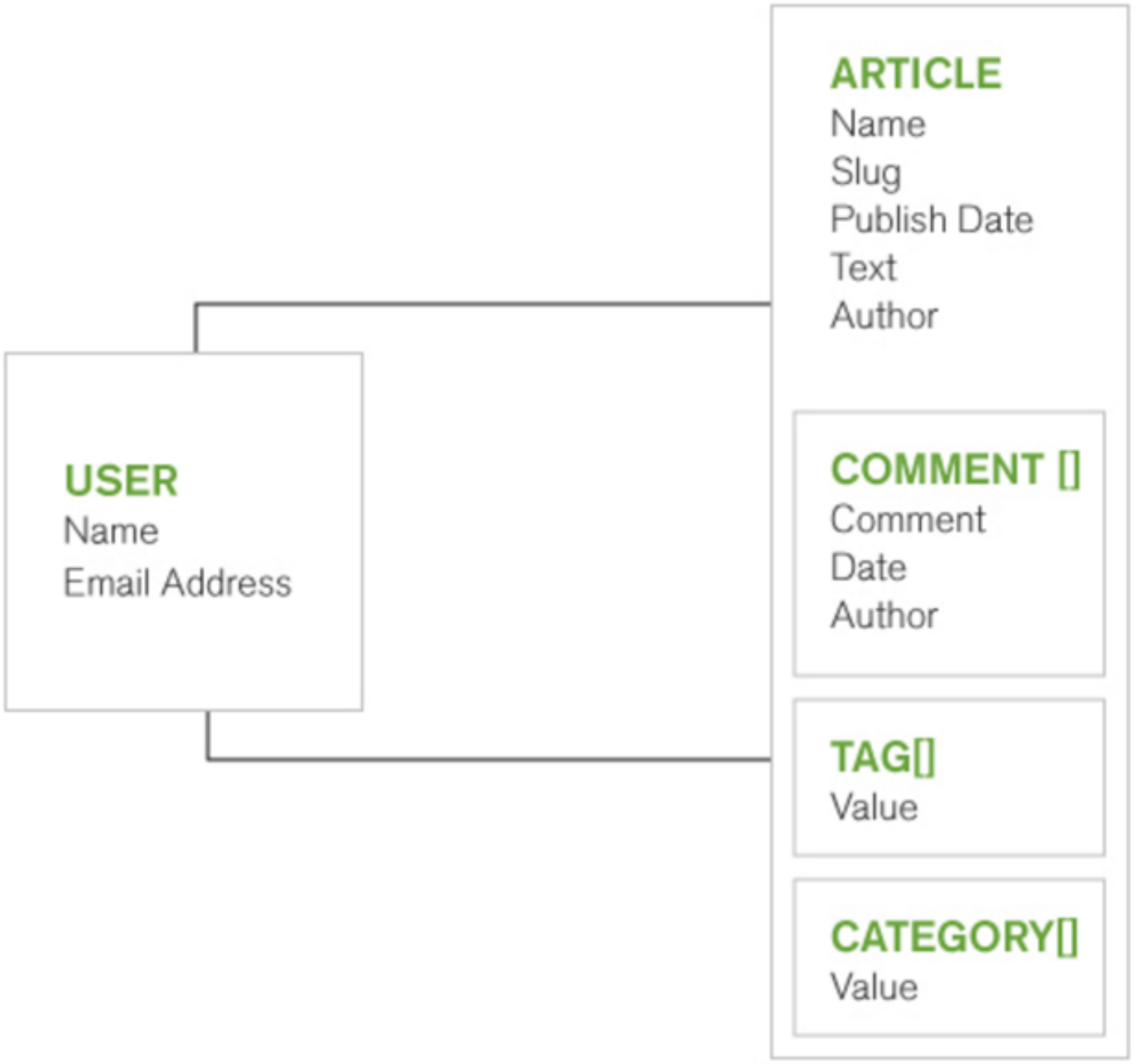
Embedded document

```
{
  _id: <ObjectId1>,
  username: "123xyz",
  contact: {
              phone: "123-456-7890",
              email: "xyz@example.com"
            },
  access: {
              level: 5,
              group: "dev"
            }
}
```

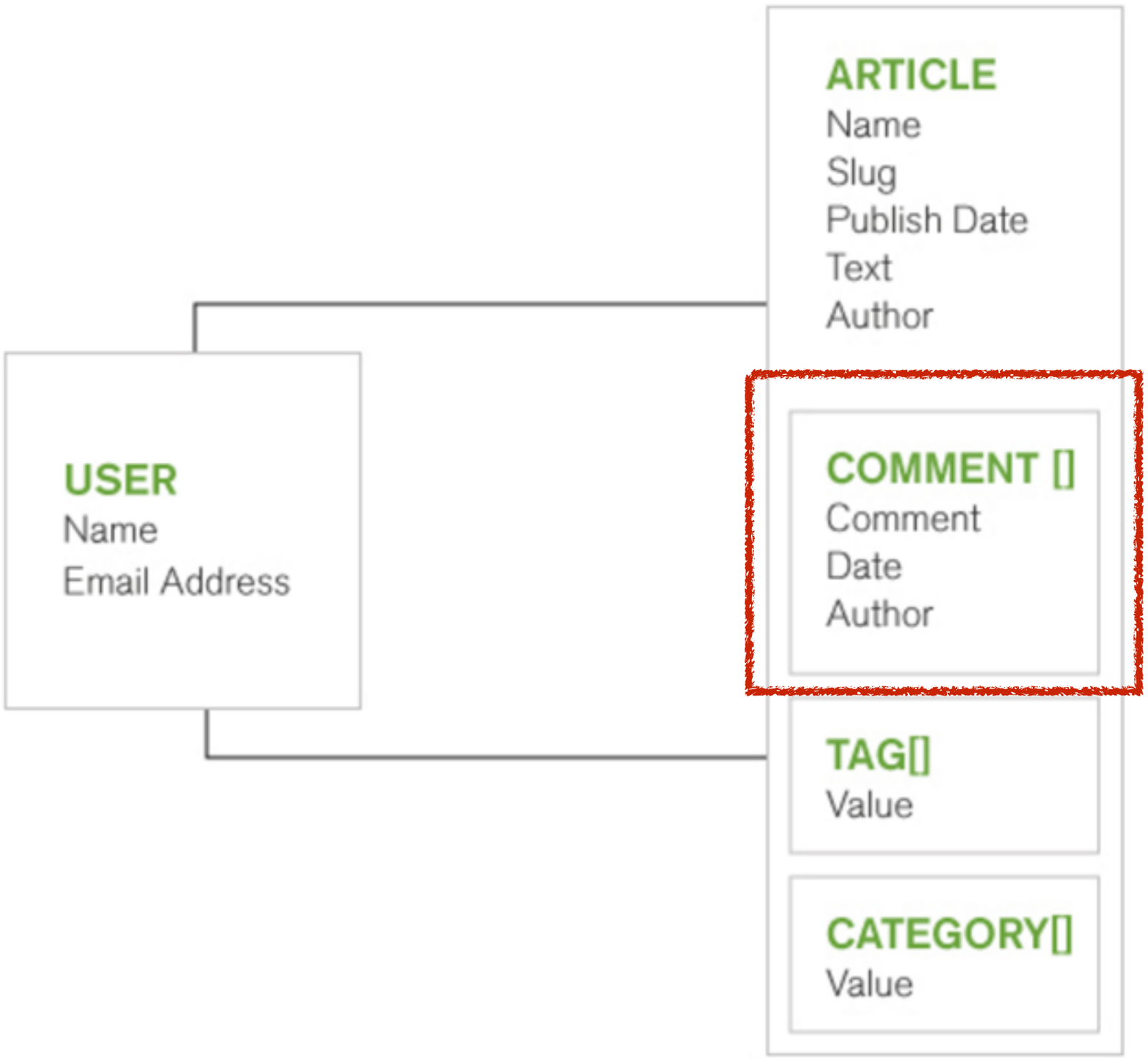
Embedded sub-document

Embedded sub-document

ออกแบบด้วยแนวคิดของ NoSQL

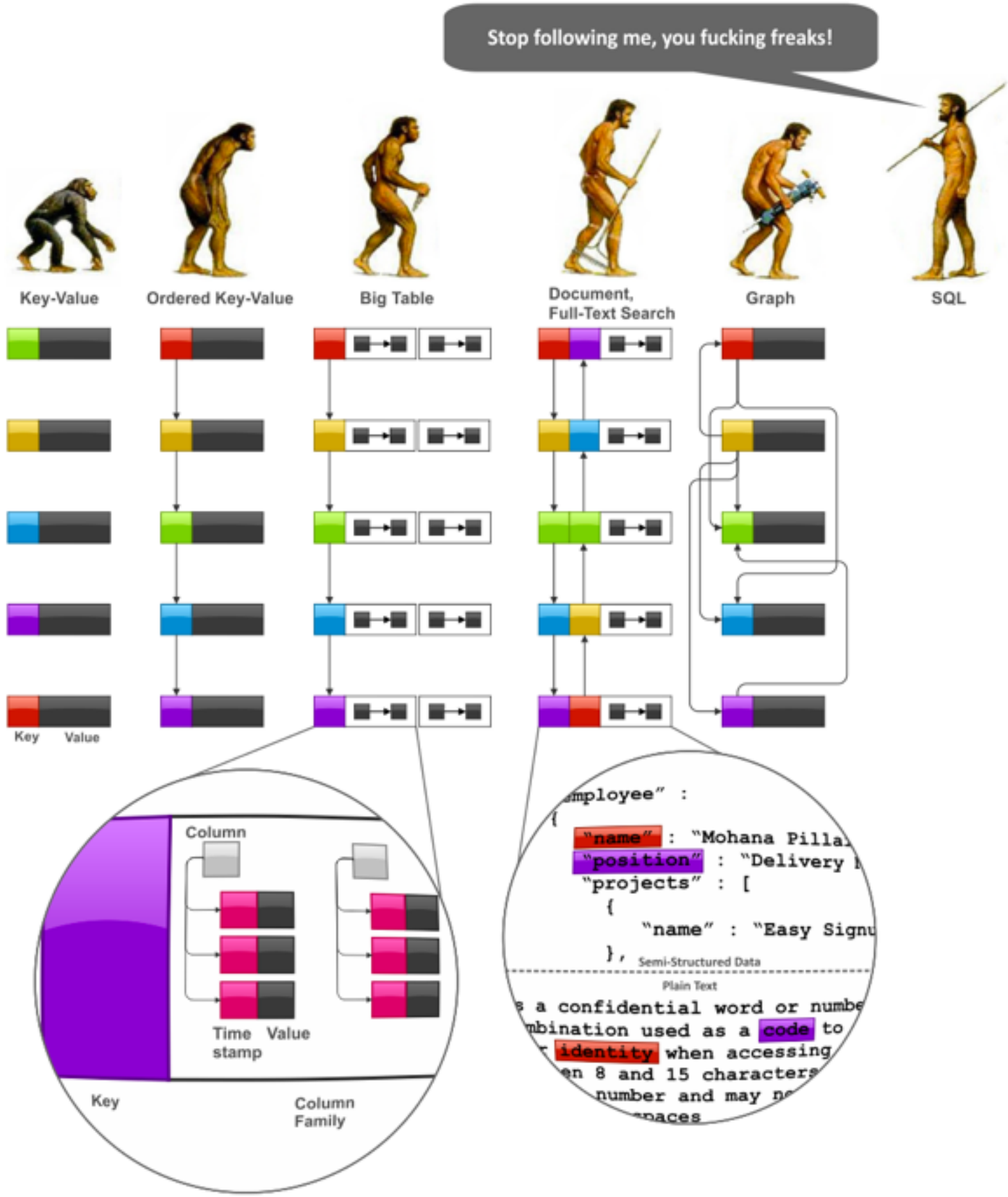

ARTICLE
Name
Slug
Publish Date
Text
Author
COMMENT []
Comment
Date
Author
TAG[]
Value
CATEGORY[]
Value
USER
Name
Email Address

ปัญหาที่อาจจะเกิดขึ้น ?


ARTICLE
Name
Slug
Publish Date
Text
Author
USER
Name
Email Address
COMMENT []
Comment
Date
Author
TAG[]
Value
CATEGORY[]
Value

แก้อย่างไรดี ?

Embedding vs Referencing


Stop following me, you fucking freaks!
Key-Value
Ordered Key-Value
Big Table
Document,
Full-Text Search
Graph
SQL
Key
Value
Column
Time stamp
Value
Key
Column Family
"name" : "Mohana Pilla
"position" : "Delivery
"projects" : [
"name" : "Easy Signu
},
Semi-Structured Data
Plain Text
a confidential word or numbe
bination used as a code to
identity when accessing
en 8 and 15 characters
number and may n
paces

คำแนะนำสำหรับการออกแบบ

Iterate schema design development

Measure performance

Find bottlenecks

Embedding by default (90/10)

Referencing when you need more scaling

Normalize vs Denormalize

Flexible for development and change

KEEP CALM

AND

use the right tool for the right job

Replication

What is Replication ?

Synchronizing data across multiple servers

Redundancy

Increase data availability

No Single Point of Failure (SPoF)

Why Replication ?

To keep data safe

High availability of data (24*7)

Disaster recovery

No downtime

Read scaling

Working with MongoDB

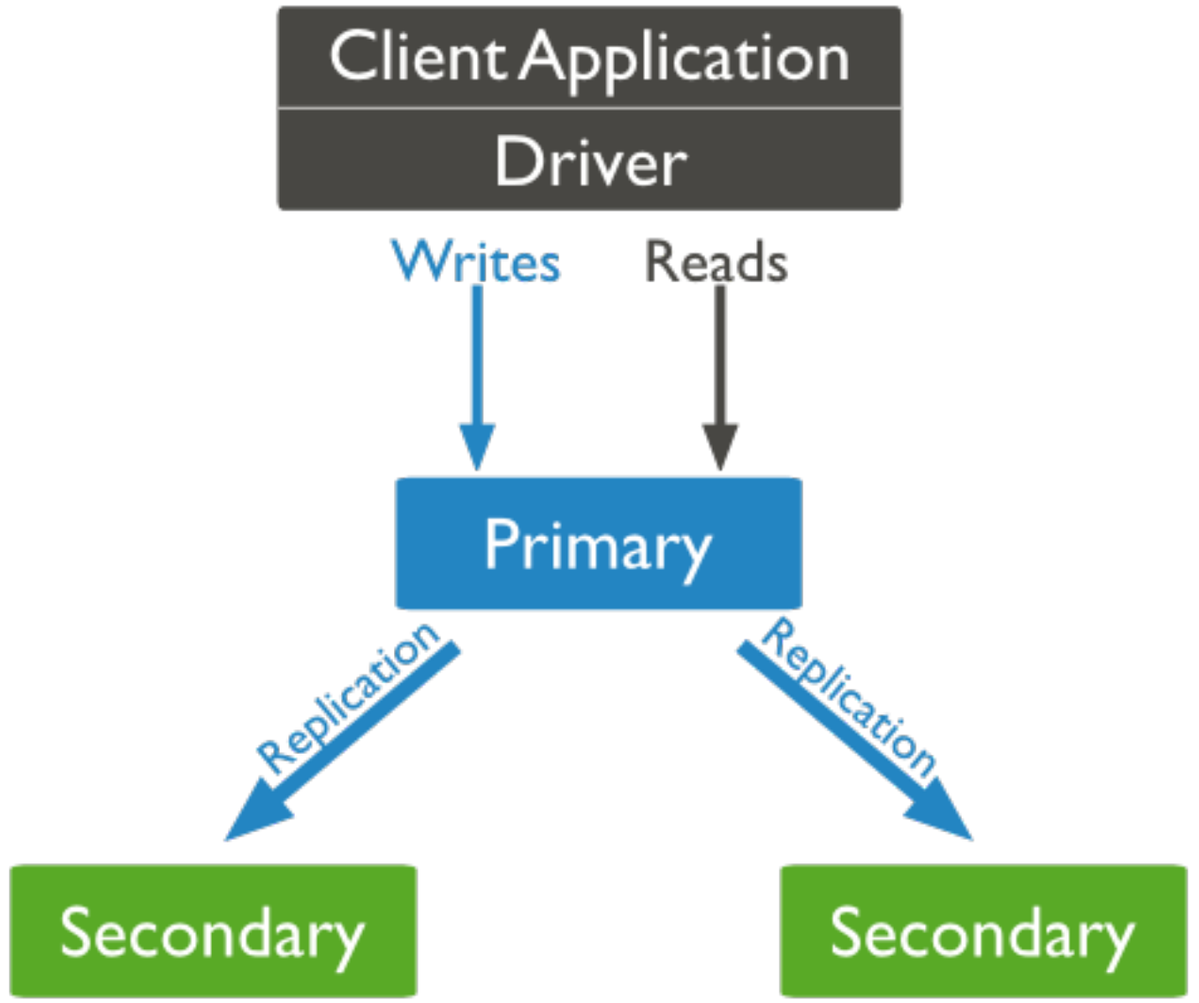
Group of **mongod** instances

Minimum 3 nodes are required !!

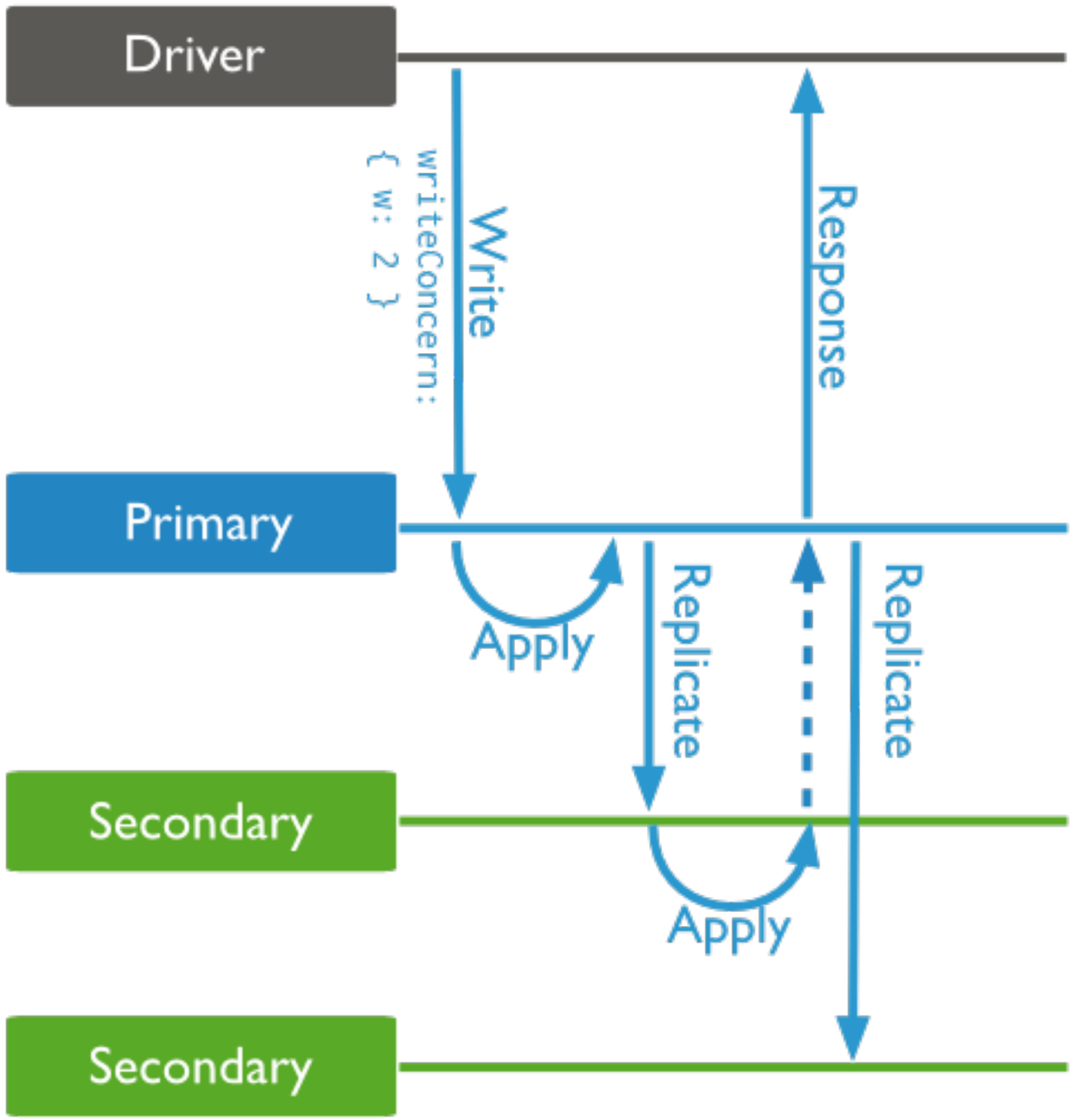
One node is **Primary node**

Other nodes is **Secondary node**

Working with MongoDB



Working with MongoDB



Replica set features

A cluster of N nodes

Anyone node can by primary

All write operations goes to primary

Automatic failover

Automatic recovery

Consensus election of primary

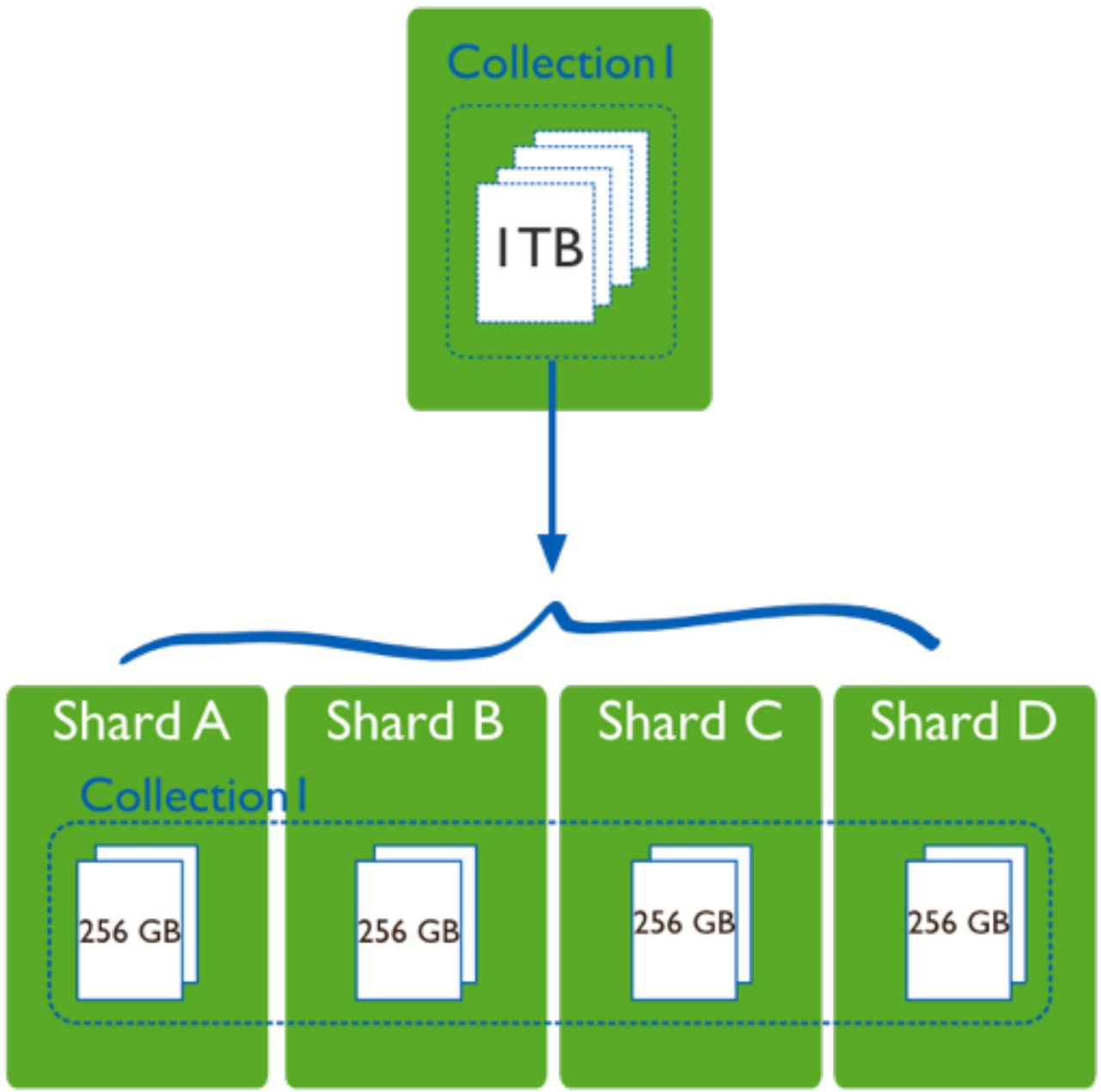
Workshop

Setup a replica set

06_replica.txt

Sharding

What is Sharding ?



What is Sharding ?

The process of storing data records across multiple machine

Approach for data growth

Why Sharding ?

In replication all writes go to master

Latency sensitive queries
still go to master

Single replica set has limitation of 12 node

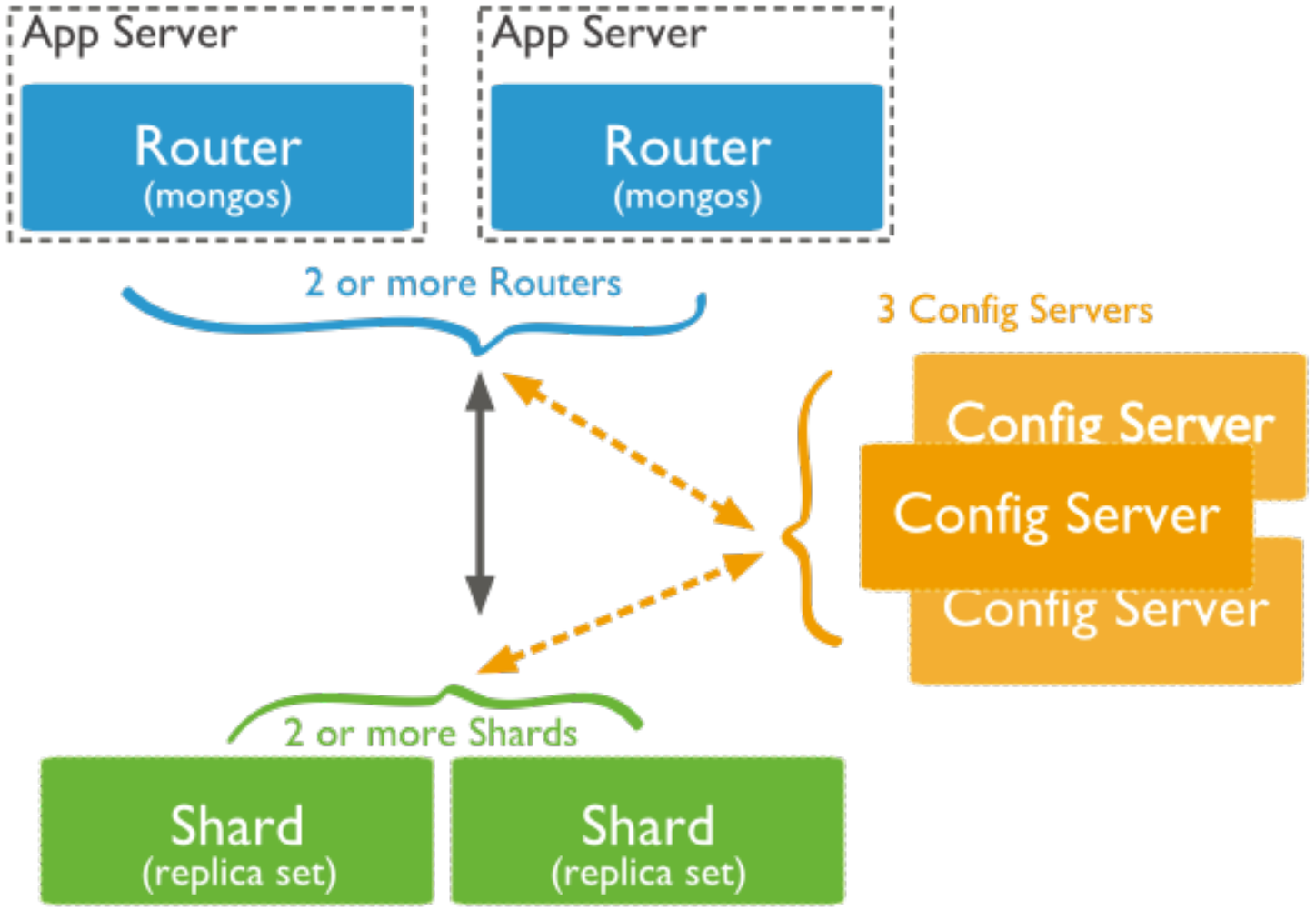
Why Sharding ?

Memory can not be large enough

Local disk is not big enough

Vertical scaling is too expensive

Sharding in MongoDB



Workshop

Working with sharding

07_sharding.txt

Backup and Restore

Create backup

$mongodump

```
writing blog.posts to
writing store.book to
done dumping store.book (3 documents)
done dumping blog.posts (1 document)
```

Create backup

$mongodump

```
dump
├── blog
│   ├── posts.bson
│   └── posts.metadata.json
└── store
    ├── book.bson
    └── book.metadata.json
```

Mongodump usage

$mongodump --host **HOST_NAME** –port **PORT_NUMBER**

HOST_NAME = 127.0.0.1

PORT = 27017

Mongodump usage

$mongodump --dbpath **DB_PATH**
--out **BACKUP_DIRECTORY**

Mongodump usage

$mongodump --collection **COLLECTION**
--db **DB_NAME**

Restore data

$mongorestore

Monitoring tools

mongostat

mongotop

https://docs.mongodb.com/manual/administration/monitoring/

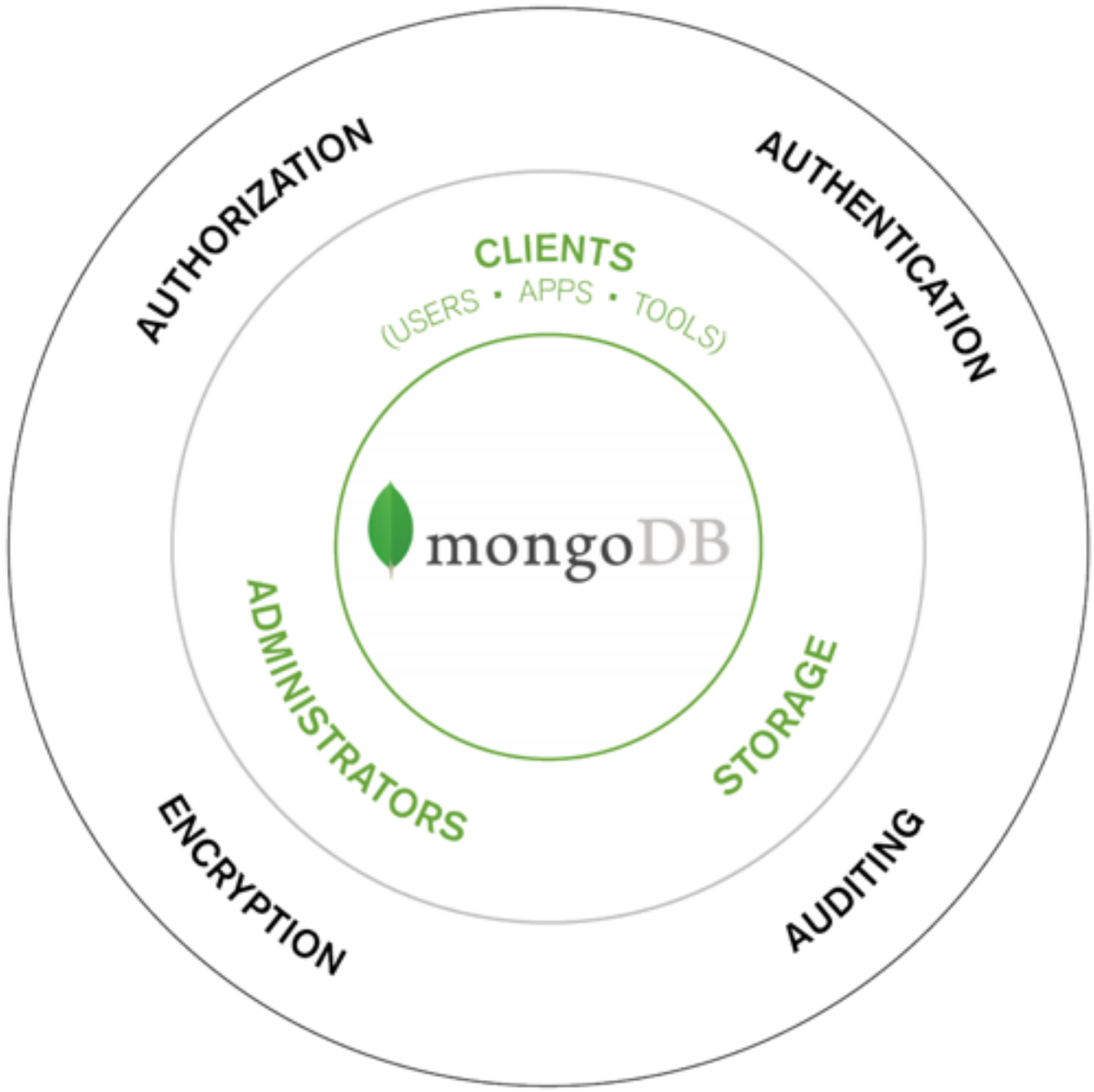
Monitoring tools

Ganglia
Motop
mtop

Munin
Nagios

https://docs.mongodb.com/manual/administration/monitoring/

Security


AUTHORIZATION
AUTHENTICATION
CLIENTS
(USERS • APPS • TOOLS)
mongoDB
ADMINISTRATORS
STORAGE
ENCRYPTION
AUDITING

Security checklist

Enable access control

Enforce authentication

Configure rule-based access control

Encrypt communication

https://docs.mongodb.com/manual/administration/security-checklist/

Security checklist

Limit network exposure

Audit system activity

Encrypt and protect data

Run MongoDB with a dedicated user

https://docs.mongodb.com/manual/administration/security-checklist/

Security checklist

Run MongoDB with secure configuration

Audit system activity

Encrypt and protect data

Run MongoDB with a dedicated user

https://docs.mongodb.com/manual/administration/security-checklist/

Workshop

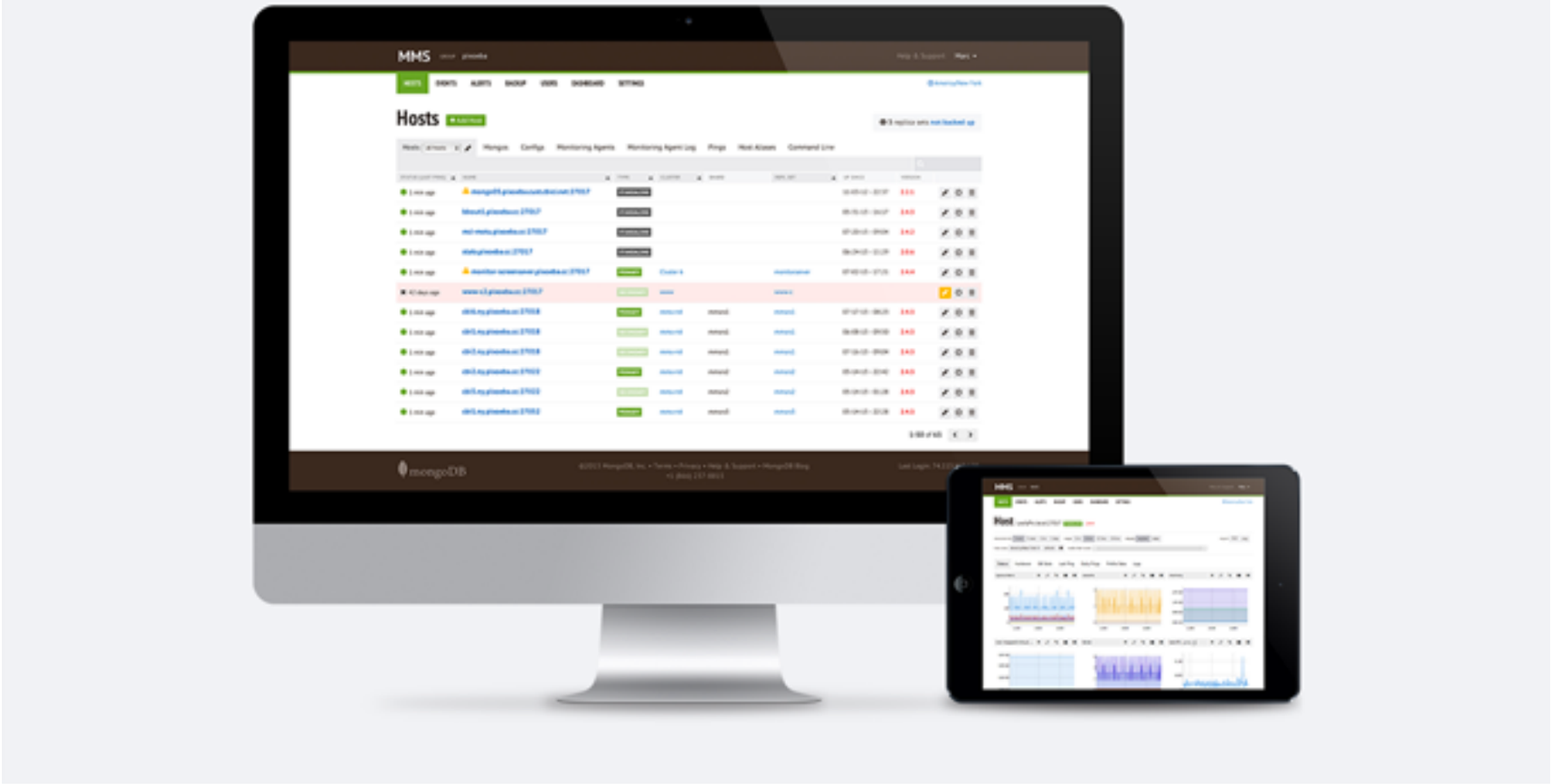
Secure MongoDB configuration

08_security.txt

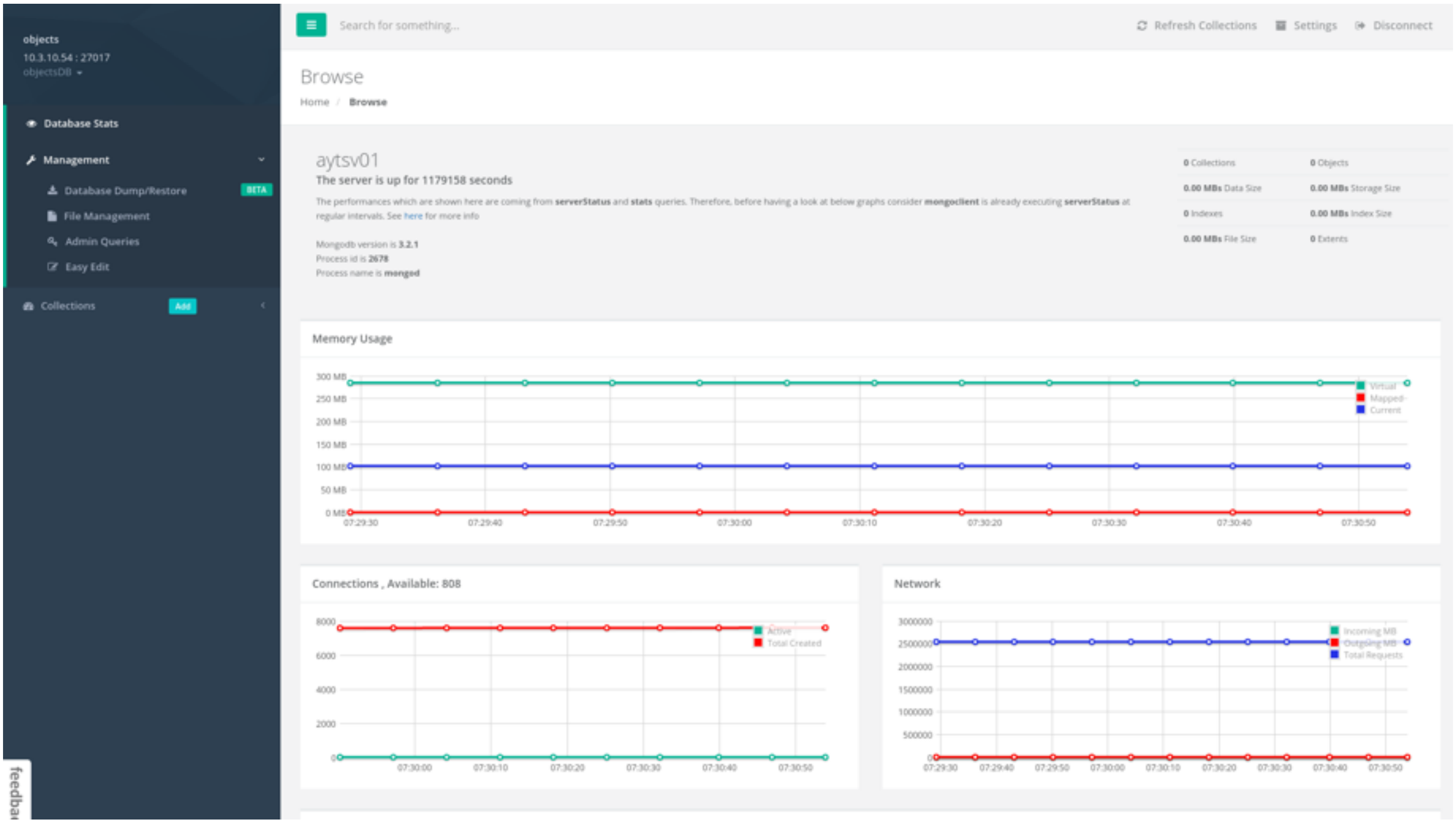
Tools for Administrator

https://docs.mongodb.com/ecosystem/tools/administration-interfaces/

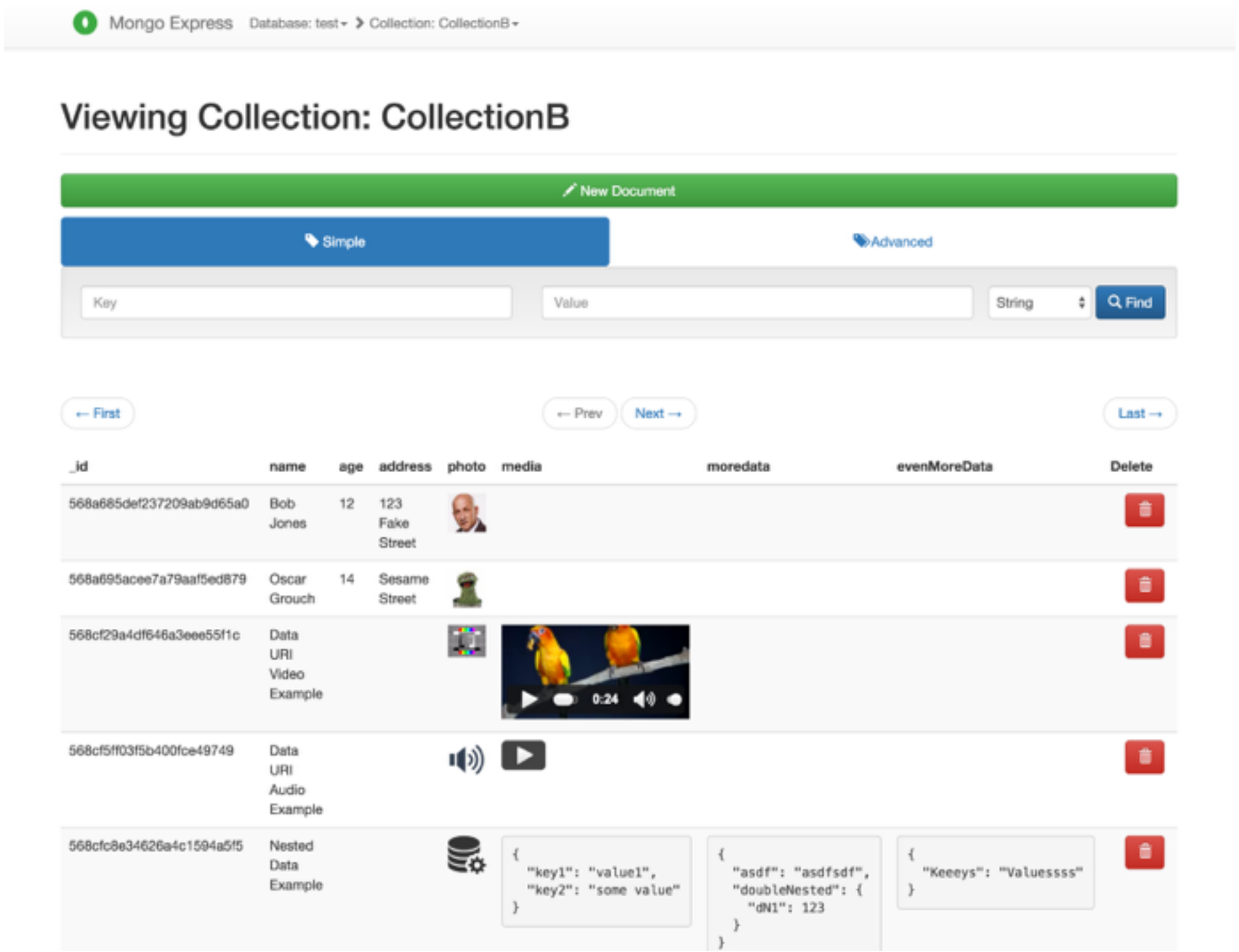
MongoDB cloud manager



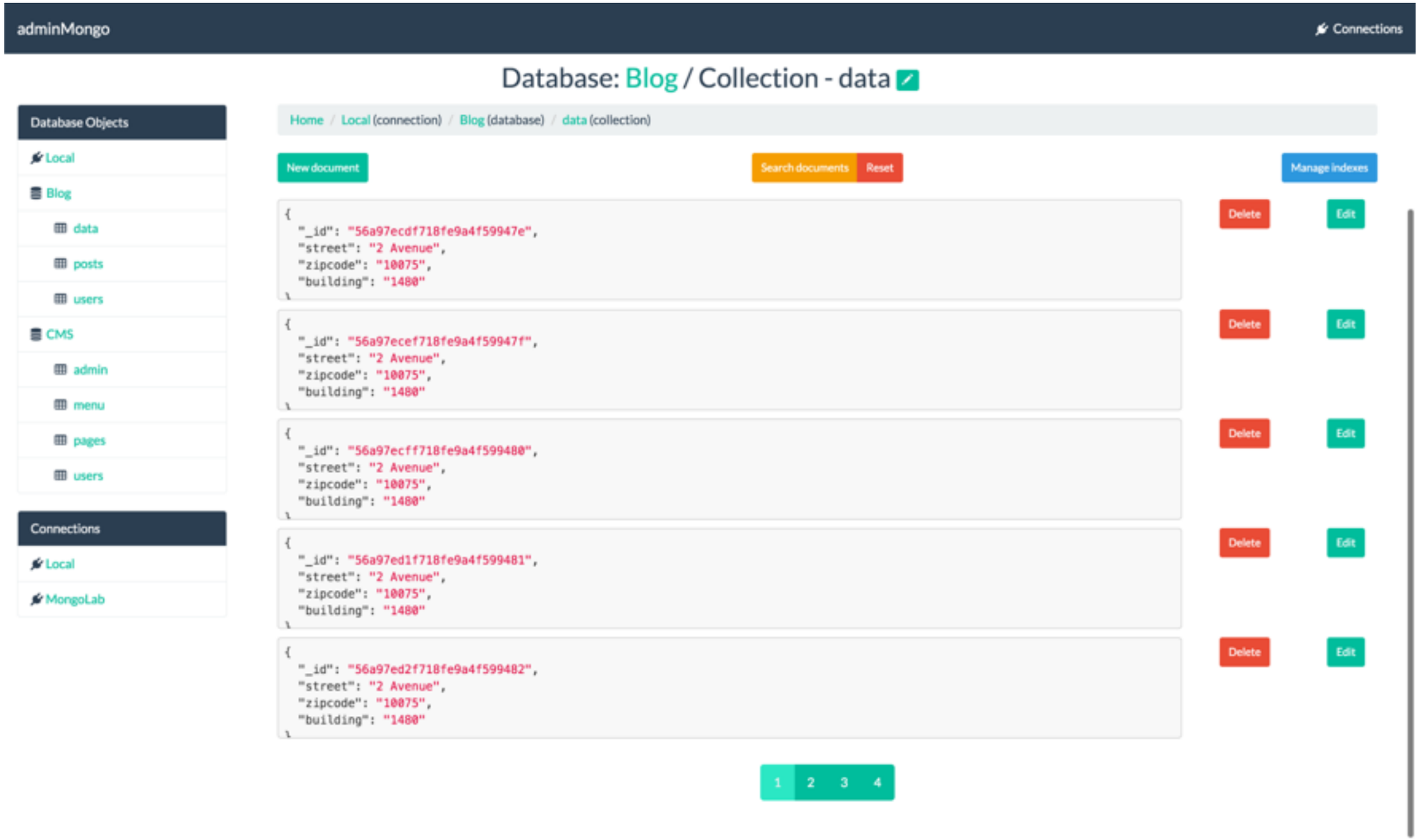
Mongoclient



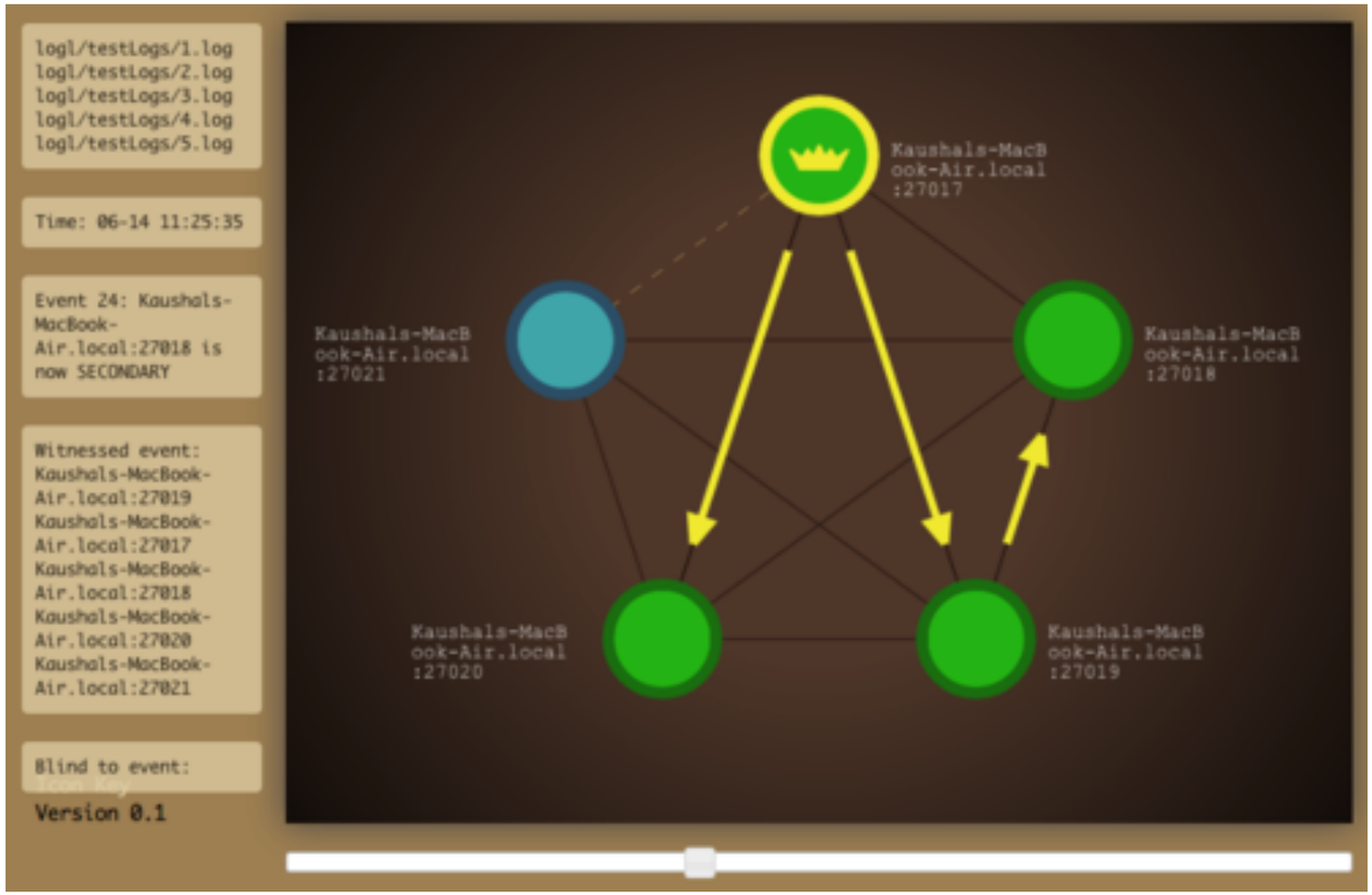
Mongo-express



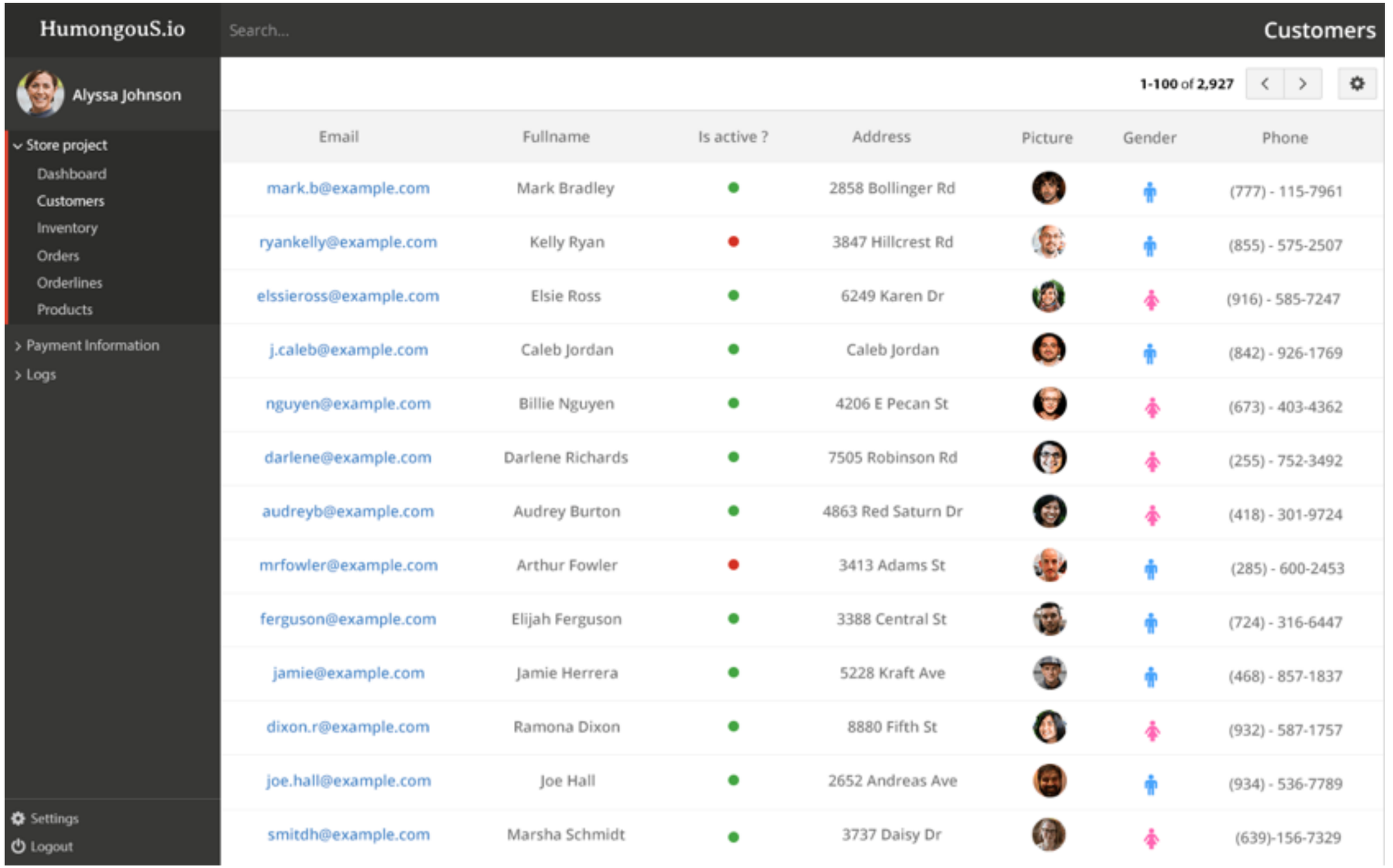
adminMongo



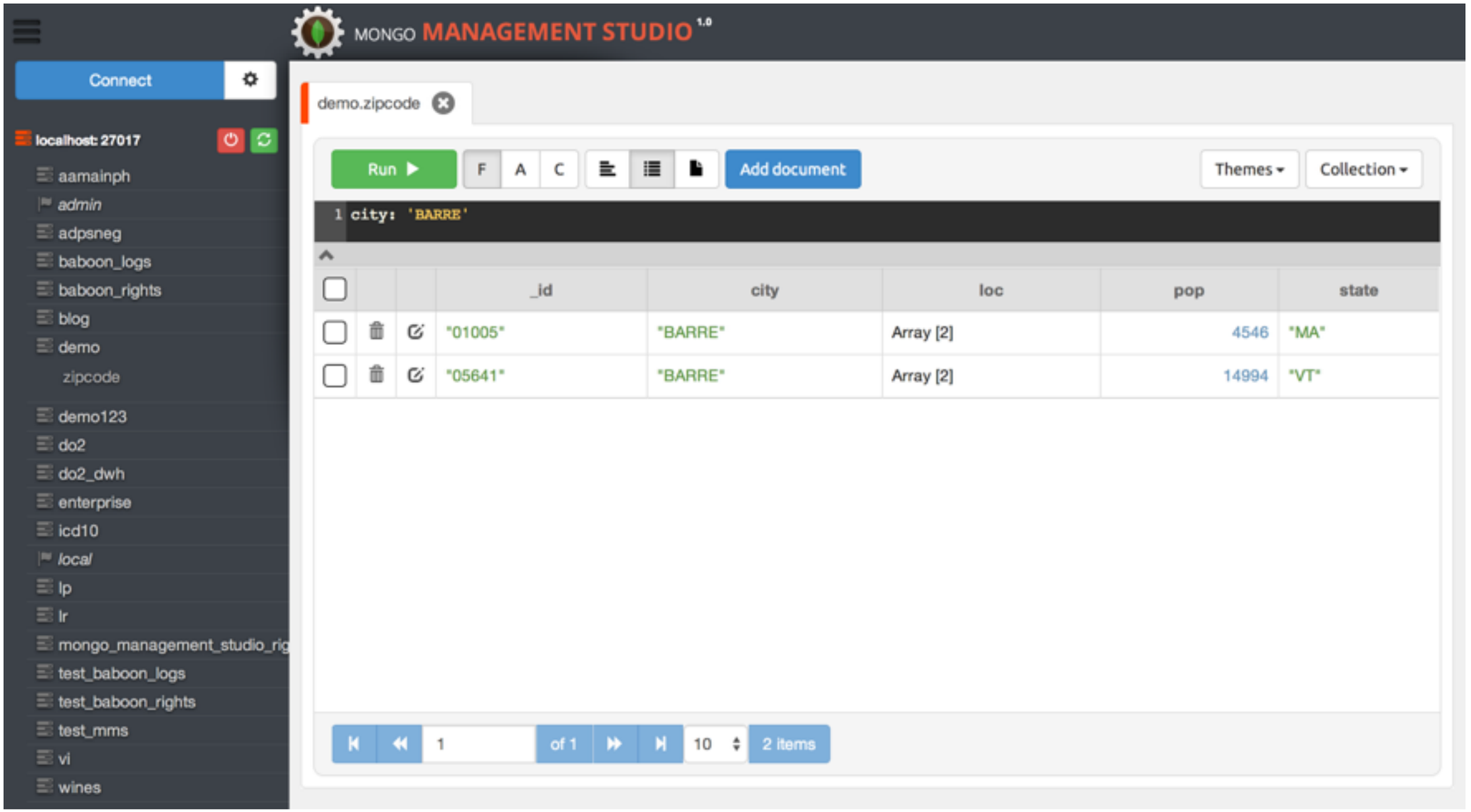
Edda


logl/testLogs/1.log
logl/testLogs/2.log
logl/testLogs/3.log
logl/testLogs/4.log
logl/testLogs/5.log
Time: 06-14 11:25:35
Event 24: Kaushals-MacBook-Air.local:27018 is now SECONDARY
Witnessed event:
Kaushals-MacBook-Air.local:27019
Kaushals-MacBook-Air.local:27017
Kaushals-MacBook-Air.local:27018
Kaushals-MacBook-Air.local:27020
Kaushals-MacBook-Air.local:27021
Blind to event:
Version 0.1
Kaushals-MacBook-Air.local:27017
Kaushals-MacBook-Air.local:27021
Kaushals-MacBook-Air.local:27018
Kaushals-MacBook-Air.local:27020
Kaushals-MacBook-Air.local:27019

HumongouS.io



Mongo Management Studio



Resources

https://github.com/up1/course-introduction-mongodb