

Document-oriented databases

Martin Hrdlička

A database is not the synonym for
a relational database.

NoSQL

- Scalability
- Distributive
- Performance
- Flexibility
- Domain Complexity

Scalaris
Project Valdemort
Dynomite
Key-value
Cassandra
Redis
Tokyo Cabinet

CouchDB
Lotus Notes
Document
SimpleDB
MongoDB
Riak
XML



BigTable
HBase
Column-oriented
FluidDB
LucidDB

Neo4j
InfoGrid
Graph-oriented
Dryad
VertexDB
RDF

What is a document?

A document (noun) is a bounded physical **representation of a body of information** designed with **the capacity** (and usually intent) **to communicate**

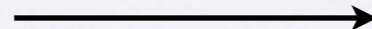
Source: [wikipedia.org](https://en.wikipedia.org/wiki/Document)

What is a document?

A document (noun) is a bounded physical **representation of a body of information** designed with **the capacity** (and usually intent) **to communicate**

Source: wikipedia.org

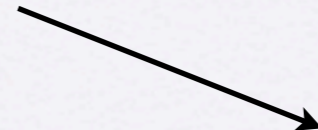
A document is a structure



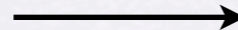
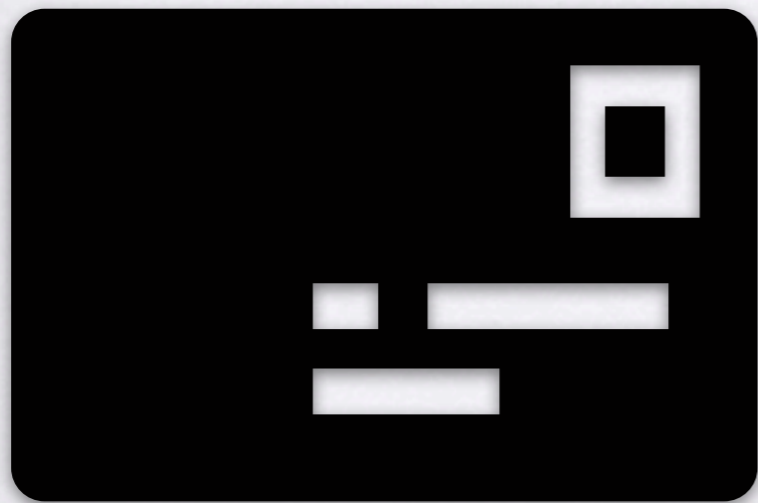
Author
Title
Text
...



First name
Last name



Chapters
Paragraphs



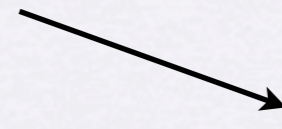
Recipient
Stamps
Text
...



Name
Address



Postcode
Street
Town
Country
...



Price
Picture

Document-oriented stores

- CouchDB
- MongoDB
- SimpleDB
- Riak
- Lotus Notes
- XML databases

XML, JSON, YAML, ...

```
{  
  "title": "Presentation of document-oriented databases systems",  
  "presenter": {  
    "uco": 208297  
    "name": "Martin Hrdlicka",  
  },  
  "session": {  
    "start": "26.11.2009 12:00"  
    "end": "26.11.2009 13:00"  
  },  
  "tags": ["couchdb", "mongodb", "document databases"]  
}
```

JSON

XML

```
<?xml version="1.0" encoding="utf-8"?>  
<event>  
  <title type="string">Presentation of document-oriented database systems</title>  
  <presenter>  
    <uco type="integer">208297</uco>  
    <name type="string">Martin Hrdlicka</string>  
  </presenter>  
  <session>  
    <start type="date">26.11.2009 12:00</start>  
    <end type="date">26.11.2009 13:00</end>  
  </session>  
  <tags type="array">  
    <tag type="string">couchdb</tag>  
    <tag type="string">mongodb</tag>  
    <tag type="string">document databases</tag>  
  </tags>  
</event>
```

What are document-oriented stores and why should we use them?

- Schema-less
- Distributive
- Scalability
- Replication



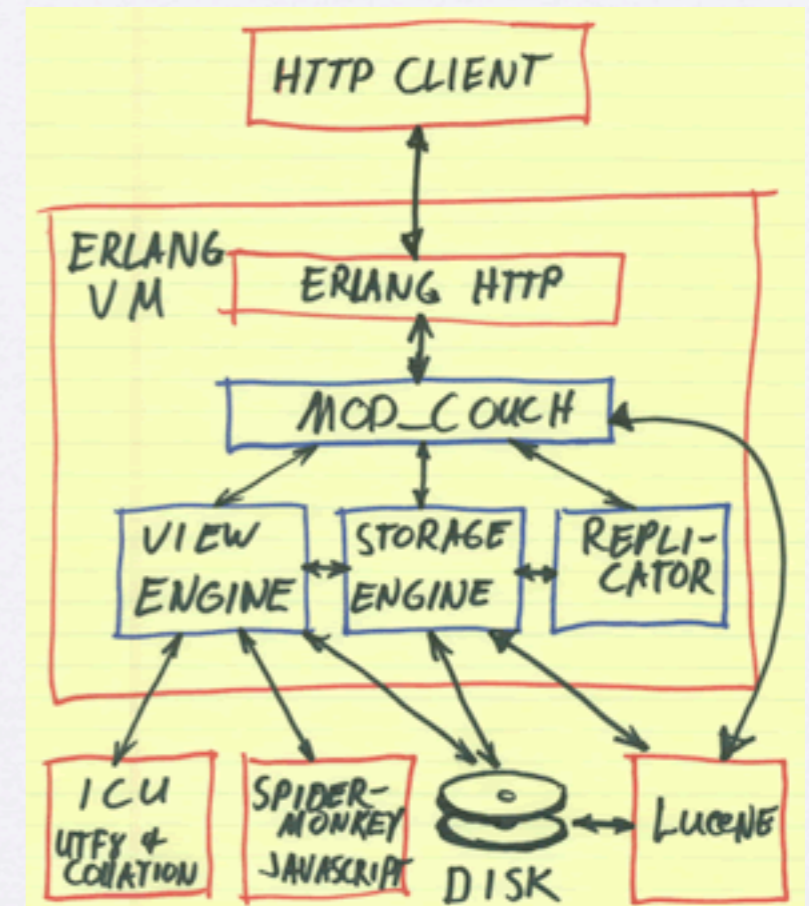
mongoDB



CouchDB
relax

CouchDB

- Robust, highly concurrent, fault-tolerant
- HTTP protocol and REST API
- MapReduce system for querying
- Incremental replication
- P2P and multi-master replication
- ACID, MVCC
- Modular system, Futon web interface



Couch Document

- Semi structured data
- JSON
- Revisions
- Binary attachments
- Full updates

```
{
  "_id": "e34ae5e9ff56453e81351d7cdf51fd58",
  "_rev": "2-967a00dff5e02add41819138abb3284d",
  "_attachments": {
    "reigatexslt2.xml": {
      "stub": true,
      "content_type": "text/xml",
      "length": 7687,
      "revpos": 2
    }
  }
  "title": "Presentation of document-oriented databases systems",
  "presenter": {
    "uco": 208297
    "name": "Martin Hrdlicka",
  },
  "session": {
    "start": "26.11.2009 12:00"
    "end": "26.11.2009 13:00"
  },
  "tags": ["couchdb", "mongodb", "document databases"]
}
```

Couch Views

- Static and predefined
- Incremental indexing
- Called design documents (prefix `_design`)

Documents

```
{ ...  
  "tags": ["mongodb", "couchdb"]  
},  
{ ...  
  "tags": ["couchdb"]  
}
```



Map

```
function(doc) {  
  if (doc.tags) {  
    doc.tags.forEach(function(tag) {  
      emit(tag, 1)  
    })  
  }  
}
```



Reduce

```
function(keys, values, rereduce) {  
  return sum(values);  
}
```



Result

```
{  
  "couchdb": 2  
  "mongodb": 1  
}
```




Close To Metal

HTTP REST API

Create

PUT /my_db/doc_id

Read

GET /my_db/doc_id

Update

PUT /my_db/doc_id

Delete

DELETE /my_db/doc_id

HTTP tools

- load balancing
- cluster

Who uses it





mongoDB

MongoDB

- Between key-value store and traditional relational databases
- Binary protocol
- Dynamic queries
- Fail-over, replication
- Indexes for inner-objects (embedded documents)
- Auto-sharding, GridFS

Mongo document

- JSON-like format Binary JSON (BSON)
- Data-types (String, Date, Integer, ...)
- Binary data (max. 4 MB, 4 MB > GridFS)
- Dynamic queries

Mongo Queries

- Dynamic
- Based on JSON-like syntax
- Querying for inner-objects

Conditional Operators : <, <=, >, >=

```
// field > value
db.collection.find({
  "field" : { $gt: value }
});
```

```
// field < value
db.collection.find({
  "field" : { $lt: value }
});
```

```
// field >= value
db.collection.find({
  "field" : { $gte: value }
});
```

Value in an Embedded Object

```
db.collection.find({
  "presenter.name" : "Martin Hrdlicka"
});
```

```
limit();
```

```
db.students.find().limit(10);
```

Who uses it



CouchDB vs. MongoDB vs. MySQL

	CouchDB	MongoDB	MySQL
Data Model	Document-Oriented (JSON)	Document-Oriented (BSON)	Relational
Data Types	?	string, int, double, boolean, date, bytearray, object, array, others	link
Large Objects (Files)	Yes (attachments)	Yes (GridFS)	no???
Replication	Master-master (with developer supplied conflict resolution)	Master-slave	Master-slave
Object(row) Storage	One large repository	Collection based	Table based
Query Method	Map/reduce of javascript functions to lazily build an index per query	Dynamic; object-based query language	Dynamic; SQL
Secondary Indexes	Yes	Yes	Yes
Atomicity	Single document	Single document	Yes - advanced
Interface	REST	Native drivers	Native drivers
Server-side batch data manipulation	?	Yes, via javascript	Yes (SQL)
Written in	Erlang	C++	C
Concurrency Control	MVCC	Update in Place	Update in Place

Source: <http://www.mongodb.org/display/DOCS/MongoDB,+CouchDB,+MySQL+Compare+Grid>

Use cases

- Web applications
- Analytics tools
- CRM, Warehouse
- Caching
- Etc.

Real World?

Pragmaticcraft

	Memcache	CouchDB	TokyoTyrant	MongoDB	PostgreSQL	MySQL (MyISAM)
Persistent	N	Y	Y	Y	Y	Y
Schema replication	Y	Y	Y	Y	N	Y [4]
Easy to install	Y	Y	Y	Y	Y	Y
Easy to use	Y	N	Y	Y	Y	Y
Well-documented	Y	N	Y	Y	Y	Y
Console	N	Y	Y	Y	Y	Y
Fetch by id	Y	Y	Y	Y	Y	Y
Fetch by query	N	Y	Y	Y	Y	Y
Fetch by substring	N	Y	Y	Y	Y	Y
Fetch by subset	N	Y [1]	Y [2]	Y	Y	Y
Fetch count	N	Y	Y	Y	Y	Y
Fetch min/max	N	Y [1]	Y [2]	Y	Y	Y
Data types	N	N	N	Y	Y	Y
Increment/decrement	Y	Y [1]	Y [2]	Y	Y	Y
Push/pop value	N	Y [1]	Y [2]	Y	Y	N
Index a column	N	Y	Y	Y	Y	Y
Virtual filesystem	N	N	N	Y	N	N
Sensible import/export	N	Y	Y	Y	Y	Y
Multi-master replication	N	Y	Y	Y	Y [3]	Y [3]
Master-slave replication	N	Y	Y	Y	Y [3]	Y [3]
Transactions	N	Y	Y	N	Y	Y
Extensible	N	Y	Y	Y	Y	Y
Proven	Y	N	N	N	Y	Y
Well-understood & common	Y	N	N	N	Y	Y
Insert one (rows/sec)	3,293	235	6,204	2,376	891	6,488
Retrieve one (rows/sec)	1,438	404	5,787	1,047	3,848	1,378
Query one (rows/sec)		237	2,793	1,047	3,848	1,378
Insert many (rows/sec)	3,293	1,620	6,204	1,018	5,457	5,774
Find all (rows/sec)		9,394	3,882	3,458	19,830	18,854
Score (bigger is better)	41%	65%	86%	86%	87%	89%
Pros:	N/A	Flexible	Quick, specialized	Easy, complete	Safe, simple	Safe, simple
Cons:	Not persistent	Very slow, trickier	Fewer features	Slower than DB	Schema replication	Schema replication
Conclusion:	Not an option	Probably not	For performance	For general purpose	Grampa is still spry	Quirky kid grew up

"Non-relational data stores for OpenSQL Camp" - Igal Koshevoy - 2009-11-14

Pragmaticraft

	Memcache	CouchDB	TokyoTyrant	MongoDB	PostgreSQL	MySQL (MyISAM)
Persistent	N	Y	Y	Y	Y	Y
Schema replication	Y	Y	Y	Y	N	Y [4]
Easy to install	Y	Y	Y	Y	Y	Y
Easy to use	Y	N	Y	Y	Y	Y
Well-documented	Y	N	Y	Y	Y	Y
Console	N	Y	Y	Y	Y	Y
Fetch by id	Y	Y	Y	Y	Y	Y
Fetch by query	N	Y	Y	Y	Y	Y
Fetch by substring	N	Y	Y	Y	Y	Y
Fetch by subset	N	Y [1]	Y	Y	Y	Y
Fetch count	N	Y	Y	Y	Y	Y
Fetch min/max	N	Y [1]	Y	Y	Y	Y
Data types	N	N	Y	Y	Y	Y
Increment/decrement	Y	Y [1]	Y	Y	Y	Y
Push/pop value	N	Y [1]	Y	Y	Y	Y
Index a column	N	Y	Y	Y	Y	Y
Virtual filesystem	N	N	Y	Y	Y	Y
Sensible import/export	N	Y	Y	Y	Y	Y
Multi-master replication	N	Y	Y	Y	Y	Y
Master-slave replication	N	Y	Y	Y	Y	Y
Transactions	N	Y	Y	Y	Y	Y
Extensible	N	Y	Y	Y	Y	Y
Proven	Y	N	N	N	Y	Y
Well-understood & common	Y	N	N	N	Y	Y
Insert one (rows/sec)	3,293	235	6,204	2,376	891	6,488
Retrieve one (rows/sec)	1,438	404	5,787	1,047	3,848	1,378
Query one (rows/sec)		237	2,793	1,047	3,848	1,378
Insert many (rows/sec)	3,293	1,620	6,204	1,018	5,457	5,774
Find all (rows/sec)		9,394	3,882	3,458	19,830	18,854
Score (bigger is better)	41%	65%	86%	86%	87%	89%
Pros:	N/A	Flexible	Quick, specialized	Easy, complete	Safe, simple	Safe, simple
Cons:	Not persistent	Very slow, trickier	Fewer features	Slower than DB	Schema replication	Schema replication
Conclusion:	Not an option	Probably not	For performance	For general purpose	Grampa is still spry	Quirky kid grew up



"Non-relational data stores for OpenSQL Camp" - Igal Koshevoy - 2009-11-14

Pragmaticcraft

	Memcache	CouchDB	TokyoTyrant	MongoDB	PostgreSQL	MySQL (MyISAM)
Persistent	N	Y	Y	Y	Y	Y
Schema replication	Y	Y	Y	Y	N	Y [4]
Easy to install	Y	Y	Y	Y	Y	Y
Easy to use	Y	N	Y	Y	Y	Y
Well-documented	Y	N	Y	Y	Y	Y
Console	N	Y	Y	Y	Y	Y
Fetch by id	Y	Y	Y	Y	Y	Y
Fetch by query	N	Y	Y	Y	Y	Y
Fetch by substring	N	Y	Y	Y	Y	Y
Fetch by subset	N	Y [1]	Y	Y	Y	Y
Fetch count	N	Y	Y	Y	Y	Y
Fetch min/max	N	Y [1]	Y	Y	Y	Y
Data types	N	N	Y	Y	Y	Y
Increment/decrement	Y	Y [1]	Y	Y	Y	Y
Push/pop value	N	Y [1]	Y	Y	Y	Y
Index a column	N	Y	Y	Y	Y	Y
Virtual filesystem	N	N	Y	Y	Y	Y
Sensible import/export	N	Y	Y	Y	Y	Y
Multi-master replication	N	Y	Y	Y	Y	Y
Master-slave replication	N	Y	Y	Y	Y	Y
Transactions	N	Y	Y	Y	Y	Y
Extensible	N	Y	Y	Y	Y	Y
Proven	Y	N	N	N	Y	Y
Well-understood & common	Y	N	N	N	Y	Y
Insert one (rows/sec)	3,293	235	6,204	2,376	891	6,488
Retrieve one (rows/sec)	1,438	404	5,787	1,047	3,848	1,378
Query one (rows/sec)		237	2,793	1,047	3,848	1,378
Insert many (rows/sec)	3,293	1,620	6,204	1,018	5,457	5,774
Find all (rows/sec)		9,394	3,882	3,458	19,830	18,854
Score (bigger is better)	41%	65%	86%	86%	87%	89%
Pros:	N/A	Flexible	Quick, specialized	Easy, complete	Safe, simple	Safe, simple
Cons:	Not persistent	Very slow, trickier	Fewer features	Slower than DB	Schema replication	Schema replication
Conclusion:	Not an option	Probably not	For performance	For general purpose	Grampa is still spry	Quirky kid grew up

Your use case is more important here!

"Non-relational data stores for OpenSQL Camp" - Igal Koshevoy - 2009-11-14

Should start asking
yourself to your

Philosophy

Design

Strategy

Requirements/Goals

Product

Useful links

Apache CouchDB

<http://couchdb.apache.org>

CouchDB: The Definitive Guide

<http://books.couchdb.org/relax/>

Interactive CouchDB

<http://labs.mudynamics.com/wp-content/uploads/2009/04/icouch.html>

MongoDB

<http://www.mongodb.org>

MongoMapper (ODM tool for Ruby)

<http://github.com/jnunemaker/mongomapper>

Why I think Mongo is to Databases what Rails was to Frameworks

<http://railstips.org/2009/12/18/why-i-think-mongo-is-to-databases-what-rails-was-to-frameworks>

Useful links

NoSQL

<http://en.wikipedia.org/wiki/NoSQL>

“Relational Databases”, Comm. of the ACM 35,4 (April 1992), 16,18.

<http://home.pipeline.com/~hbaker1/letters/CACM-RelationalDatabases.html>

Anti-RDBMS: A list of distributed key-value stores

<http://www.metabrew.com/article/anti-rdbms-a-list-of-distributed-key-value-stores/>

NoSQL East

<http://nosqleast.com>

NoSQL Discussion

<http://groups.google.com/group/nosql-discussion>

NoSQL Databases

<http://nosql-databases.org/>

Questions?

Goodbye!