Introducción al mundo NoSQL

César D. Rodas
crodas@php.net
http://crodas.org/

Free Software Asunción
Asunción, Paraguay

acerca de ...

- Desarrollador
- Estudiante
- Zelote de
  - Open Source
  - PHP
  - MongoDB
- PECL Developer
- ... y algunas cosas mas

Agenda

- NoSQL
- Redis
- CouchDB
- MongoDB

NoSQL, ¿Qué es?

NoSQL es un término usado en informática para agrupar una serie de almacenes de datos no relacionales que no proporcionan garantías ACID. Normalmente no tienen esquemas fijos de tablas ni sentencias "join".

Wikipedia

NoSQL, ¿Qué es?

- Como término nació a principios del 2009
- Como concepto, es algo viejo
- Conjunto de bases de datos que:
  - No provee SQL
  - Los datos no tienen relación
  - Generalmente escalan horizontalmente

“Problemas”de las bases de datos relacionales

- SQL
  - Lenguaje interpretado
  - Mucha funcionalidad
- ACID ofrece demasiado
- La manera más fácil de escalar es verticalmente

“CAP”y “BASE”

- Teorema de CAP, puedes tener dos cosas:
  - Consistency
  - Availability
  - Partition tolerance
- BASE, el reemplazo de ACID
  - Basically Available
  - Soft state
  - Eventually consistent

Algunas bases de datos NoSQL

REDIS

Redis

- In-memory Key-value store
- Muy rápido, en mi notebook:
  - 20K inserts en 0.6 segundos
  - 20K selects en 1 segundo
- Soporta operaciones con las claves
- Los valores pueden ser:
  - bytes
  - Sets
  - Tuplas
- Operaciones atómicas (Push, Pop, ranges, etc)
- Clientes para varios lenguajes
- Replicación a disco

¿Documentos?

http://www.flickr.com/photos/beglen/152027605/

¿Documentos?

- "Objetos sin funciones"
- Arrays (en PHP)
- Tuples (en Python)
- Hashes (en Ruby y Perl)
- Sin esquema

Documentos!

```
$collection[$id] = array(
    "title" => "PHP rules",
    "tags" => array("php", "web"),
    "body" => "... PHP rules ...",
    "comments" => array(
        array("author" => "crodas", "comment" => "Yes it does"),
    ),
    "visits" => array("200.10.228.34", "200.10.228.2"),
);
```

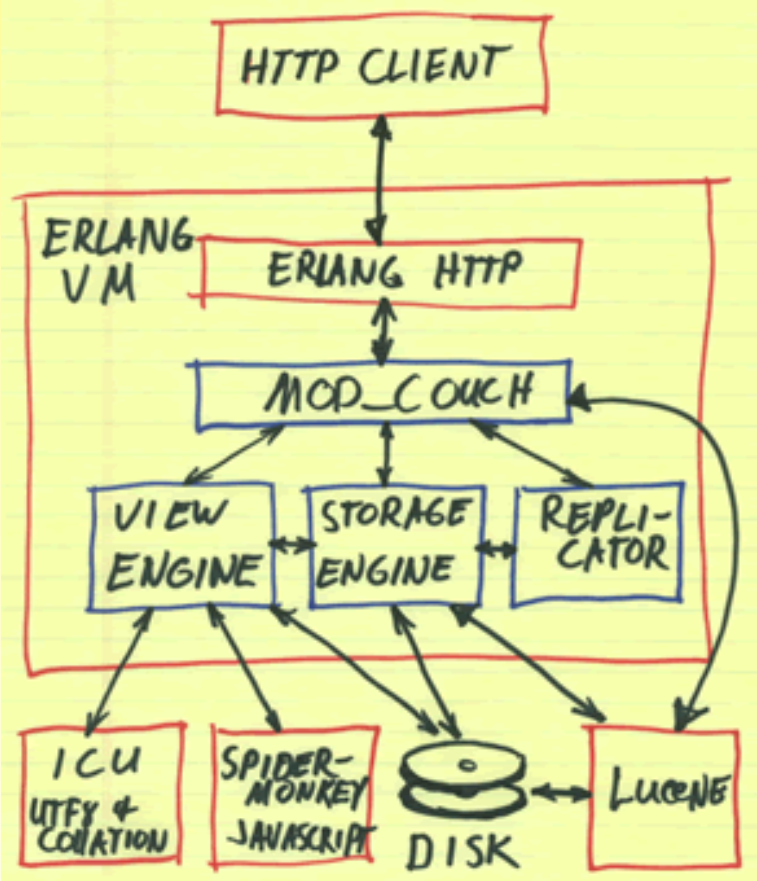

CouchDB
relax

CouchDB

- Proyecto Apache
- JSON (no XML :-)
- Interfaz RESTful (+/-)
  - Estándar (++)
  - Http es lento (--)
- MVCC
- Safe IO (append only b-tree)
- Multi-master
- Excelente Interfaz de administración

CouchDB

- Javascript incrustado
- No soporta indexes (--)
- Map/Reduce para realizar queries (+/-)
- Soporta guardar archivos
- Fácil de escalar (con proxies convencionales)
- Distribuido y concurrente por naturaleza (Erlang)


HTTP CLIENT
ERLANG VM
ERLANG HTTP
MOD_COUCH
VIEW ENGINE
STORAGE ENGINE
REPLI-CATOR
ICU
UTFY & COLLATION
SPIDER-MONKEY
JAVASCRIPT
DISK
LUCENE


mongoDB

MongoDB

- <EN>Mongo</EN> != <ES>Mongo</ES>
- Orientada a documento
- Rápida, escalable, fácil de usar
- Soporta índices
  - Simples
  - Compuestos
  - Geo-spatial
- Auto-sharding
- Cliente PECL
- *El "MySQL"del NoSQL*

MongoDB

- Soporta sub-documentos
- Updates in-places
- Javascript incrustrado
- Map/Reduce para procesamiento off-line
- Todo es un “documento”

http://bit.ly/mongodb-php

La diversión

MongoDB - Operaciones

- Select
  - $gt, $lt, $gte, $lte, $eq, $neq: >, <, >=, <=, ==, !=
  - $in, $nin
  - $size, $exists
  - $where: Any javascript expression
  - group()
  - limit()
  - skip()
- Updates
  - $set
  - $unset
  - $push
  - $pull
  - $inc

pecl install mongo

MongoDB - Connección

```
/* connects to localhost:27017 */
$connection = new Mongo();

/* connect to a remote host (default port) */
$connection = new Mongo( "example.com" );

/* connect to a remote host at a given port */
$connection = new Mongo( "example.com:65432" );

/* select some DB (and create if it doesn't exits yet) */
$db = $connection->selectDB("db_name");

/* select a "table" (collection) */
$table = $db->getCollection("table");
```

FROM SQL to MongoDB

MongoDB - Count

```
/* SELECT count(*) FROM table */
$collection->count();

/* SELECT count(*) FROM table WHERE foo = 1 */
$collection->find(array("foo" => 1))->count();
```

MongoDB - Consultas

```
/*
 * SELECT * FROM table WHERE field IN (5,6,7) and enable=1
 * and worth < 5
 * ORDER BY timestamp DESC
 */

$collection->ensureIndex(
    array('field'=>1, 'enable'=>1, 'worth'=>1, 'timestamp'=>-1)
);

$filter  = array(
        'field' => array('$in' => array(5,6,7)),
        'enable' => 1,
        'worth' => array('$lt' => 5)
    );
$results = $collection->find($filter)->sort(array('timestamp' => -1));
```

MongoDB - Paginaciones

```
/*
 * SELECT * FROM table WHERE field IN (5,6,7) and enable=1
 * and worth < 5
 * ORDER BY timestamp DESC LIMIT $offset, 20
 */
$filter  = array(
        'field' => array('$in' => array(5,6,7)),
        'enable' => 1,
        'worth' => array('$lt' => 5)
    );

$cursor = $collection->find($filter);
$cursor->sort(array('timestamp' => -1))->skip($offset)->limit(20);

foreach ($cursor as $result) {
    var_dump($result);
}
```

Diseñando documentos

Simple multi-blog system

MongoDB - Collections

- Blog
- Post
- User
- Comment

Documentos "blog"

```
$blog = array(
    "user"  => <userref>,
    "title" => "Foo bar blog"
    "url" => array(
        "foobar.foobar.com",
        "anotherfoo.foobar.com",
    ),
    "permissions" => array(
        "edit" => array(<userref>, <userref>)
    );
    "post" => 1,
    ...
);
```

Documentos "post"

```
$post = array(
    "blog" => <blogref>,
    "author" => <userref>,
    "uri"    => "/another-post",
    "title" => "Another post",
    "excerpt" => "bla, bla, bla",
    "body" => "bar, foo bar, foo, bar",
    "tags" => array("list", "of tags"),
    "published" => true,
    "date" => <mongodate>,
);
```

otros documentos

```
$comment = array(
    "post" => <postref>,
    "name" => "foobar",
    "email" => "foo@bar.com",
    "comment" => "Hello world",
    "date" => <mongodate>,
);

$user = array(
    "user" => "crodas",
    "email" => "crodas@php.net",
    "password" => "a1bad96dc68f891ca887d50eb3fb65ec82123d05",
    "name" => "Cesar Rodas",
);
```

Índices

```
$blog_col->ensureIndex(array("url" => 1));

$post_col->ensureIndex(array("blog" => 1, "date" => -1));
$post_col->ensureIndex(array("blog" => 1, "uri" => 1), array("unique" => 1));

$comment_col->ensureIndex(array("post" => 1, "date" => -1));

$user_col->ensureIndex(array("user" => 1), array("unique" => 1));
$user_col->ensureIndex(array("email" => 1), array("unique" => 1));
```

Eso es todo :-)

Agreguemos emociones :-)

Let's shard it!

Estrategia

- Shard la coleción "blog"
- Shard la coleción "user"
- Las otras coleciones estan aisladas
  - Crear “namespaces”para post and comments (foo.post, foo.comments)
  - Modificar "blog" para agregar referencia hace el “namespaces”
- MongoDB hará el trabajo por nosotros

Configurando MongoDB

```
/*  Asumiendo que tienes MongoDB ejecutandose
 *  en un entorno distribuido (sharded), esto
 *  se hace una sola vez.
 *
 *  $admin es una coneccion a la DB "admin"
 */
$admin->command(array(
    "shardcollection" => "blog",
    "key" => array("uri" => 1),
));

$admin->command(array(
    "shardcollection" => "user",
    "key" => array("user" => 1),
    "unique" => true,
));
```

Nueva colección "blog"

```
$blog = array(
    "user"  => <userref>,
    "title" => "Foo bar blog"
    "url" => array(
        "foobar.foobar.com",
        "anotherfoo.foobar.com",
    ),
    "permissions" => array(
        "edit" => array(<userref>, <userref>)
    );
    "post" => 1,
    "namespace" => "3h3g3k3",
    "database"  => "34fs321",
);
```

Selecionar el namespace correcto, y consultar :-)

Preguntas?

Gracias hackers!

@crodas

crodas.org