



# MongoDB Advanced Topics

César D. Rodas

*[crodas@php.net](mailto:crodas@php.net)*

*<http://crodas.org/>*



Yahoo! Open Hack Day 2010  
São Paulo, Brasil

# Who is this fellow?

- Paraguayan
- Zealot of
  - Open Source
  - PHP
  - MongoDB
- PECL Developer
- ... and few other things



# I'd like to thanks to...

- 10gen
- Yahoo!
- My Brazilian friends



# Agenda

- Introduction to MongoDB
- MongoDB Queries
- Real life example:
  - Designing data model for a Wordpress.com like site
  - Optimize our data model to run in sharded environment



# MongoDB

- <EN>Mongo</EN> != <PT>Mongo</PT>
- Document oriented database
- Fast, Scalable, Easy to use
- Support Indexes
  - Simple
  - Compound
  - Geo spatial
- Support sharding
- PECL client





# Documents?



# It's just an `Array()`



**In fact everything is an  
Array() in MongoDB**





<http://bit.ly/mongodb-php>



# The fun part

# MongoDB - Operations



## • Select

- `$gt`, `$lt`, `$gte`, `$lte`, `$eq`, `$neq`: `>`, `<`, `>=`, `<=`, `==`, `!=`
- `$in`, `$nin`
- `$size`, `$exists`
- `$where`: Any javascript expression
- `group()`
- `limit()`
- `skip()`

## • Update

- `$set`
- `$unset`
- `$push`
- `$pull`
- `$inc`



# pecl install mongo

# MongoDB - Connection



```
/* 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 exists 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 - Queries



```
/*  
* 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 - Pagination



```
/*
 * 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);
}
```



# Designing data structure

Simple multi-blog system

# MongoDB - Collections

- Blog
- Post
- User
- Comment



# Blog document

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



# Post document

```
$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>,  
);
```



# Missing docs.

```
$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",  
);
```



# About indexes



```
$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));
```



**That's it.**





# Hum...



# Until now, nothing new



# Let's shard it!

# Strategy



- ① Shard Blog collection
- ① Shard User collection
- ① Other collections are isolated
  - Create namespaces for post and comments (foo.post, foo.comments)
  - Modify Blog collection, add info about "namespace" (and DB).
- ① MongoDB will distribute our DB and namespaces

# Configuring the sharding

```
/* Assuming you have already MongoDB running
 * on a sharded env., you should do this once.
 *
 * $admin is a connection to the "admin" DB.
 */
```

```
$admin->command(array(
    "shardcollection" => "blog",
    "key" => array("uri" => 1),
));
```

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



# New blog document

```
$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",  
);
```





**Select the DB and namespace  
to use "post" and "comments"**



# Questions?





# Thank you hackers!



# MongoDB is looking from Brazilian help



**@crodas**

**crodas.org**