

# MongoDB

You got a Database in my Key-Value Store!

Mathias Stearn  
@mathias\_mongo



Berlin Buzzwords – 7 June 2010

Find this presentation at [bit.ly/mongodb-berlin-buzzwords](http://bit.ly/mongodb-berlin-buzzwords)  
Can people in the back read this line?

## Warning: MongoDB is not fully buzzword compliant!!!

- Binary protocol not HTTP/REST
- BSON not “pure” JSON
- C++ not Erlang
- Queries use Indexes not MapReduce
- B-Trees not Hashing

## Inserting

```
db.users.insert( {  
  _id: 'mstearn',  
  company: '10gen',  
  name: { first: 'Mathias',  
          last: 'Stearn' },  
  likes: [ 'Bier', 'Ampelmännchen', 'Python' ],  
  posts: 42  
  
  addresses: [  
    { street: '17_W_18th_St', floor: '8',  
      city: 'New_York', state: 'NY', zip: '10019' },  
    { street: '123_Fake_St', country: 'Elbonia' }  
  ]  
})
```

## Querying

```
db.users.findOne({_id: 'mstearn'})  
db.users.find({company: '10gen'})  
db.users.find({posts: {$gte: 40, $lt: 50}})  
db.users.find({'name.last': 'Stearn'})  
  
db.users.ensureIndex({knows: 1})  
  
db.users.find({likes: 'Bier'})  
db.users.find({likes: {$in: ['Bier', 'Beer']}})  
db.users.find({likes: {$all: ['Bier', 'Milch']}})  
db.users.find({likes: /^Py/})  
db.users.find().sort({posts: -1}).skip(10).limit(10)
```

## Pretty Queries with MongoMagic

### Shameless self-promotion

<http://github.com/RedBeard0531/MongoMagic/>

```
db.users.find( M._id == 'mstearn' )  
db.users.find( M.company == '10gen' )  
db.users.find( 40 <= M.posts < 50 )  
db.users.find( M.name.last == 'Stearn' )  
db.users.find( M.likes.IN( 'Bier', 'Beer' ) )  
db.users.find( M.likes.ALL( 'Bier', 'Milch' ) )  
db.users.find( M.likes.STARTSWITH( 'Py' ) )
```

## Selecting Fields

```
db.posts.find({}, {comments: 1})
```

```
db.posts.find({}, {comments: 0})
```

```
db.posts.find({}, { 'comments.by': 1})
```

```
db.posts.find({}, {comments: $slice: 10})
```

```
db.posts.find({}, {comments: $slice: -10})
```

```
db.posts.find({}, {comments: $slice: [10, 10]})
```

## 2D Geospatial Queries

```
db.zips.insert({_id: '10011', loc: [43, -74]})
```

```
db.zips.ensureIndex({loc: '2d'})
```

```
db.zips.find({loc: {$near: [43, -74]}})
```

```
var box = [[x1, y1], [x2, y2]]  
db.zips.find({loc: {$within: {$box: box}}})
```

```
var circle = [[x,y], radius]  
db.zips.find({loc: {$within: {$center: circle}}})
```

## Updating

```
db.posts.insert({_id: ObjectId(123),  
                 by: 'mstearn',  
                 title: 'Why_is_MongoDB_Awesome?',  
                 body: 'It_just_is_MASSIVE TYPO',  
                 tags: [] })
```

```
db.posts.update({_id: ObjectId(123)},  
               {$set: {body: 'It_just_is' }})
```

```
db.posts.update({_id: ObjectId(123)},  
               {$addToSet: {tags: 'Citation_Needed' }})
```

```
db.tags.update({_id: 'Citation_Needed'},  
               {$inc: {count: 1}},  
               {upsert: true})
```



## Updating in an Array

```
db.posts.update({_id: ObjectId(123)},  
               {$push: {comments:  
                        {id: ObjectId(987),  
                          by: 'joe_schmoe',  
                          text: 'I_Agree'}}})
```

```
db.posts.update({_id: ObjectId(123), comments.id: ObjectId(987)},  
               {$inc: {comments.$.votes: 1}})
```

## Removing

```
db.stuff.remove({_id: SOME_ID})  
db.stuff.remove({any: query})
```

## Sidenote: Primary Keys

- Always called “\_id”
- Automatically indexed
- Can be any type (except array)
- If you don't supply one, one will be created for you
- ObjectId(“DEADBEEF C0FFEE ABBA BADFEE”)
  - 4-Byte Timestamp (second)
  - 3-Byte Machine ID
  - 2-Byte PID
  - 3-Byte Incrementing Counter
  - Total: 12 Bytes (not 24)

## Replication

- Replication: Just Do It!
  - No durability guarantees without clean shutdowns
- Master-Slave not ~~Potentially~~ Eventually Consistent
- Automatic fail-over
  - Replica Pairs (Deprecated)
  - Replica Sets (Coming in 1.6)
- Slaves do an initial sync then pull operations
- Initial sync can be skipped if starting from snapshot

## Replication Live Demo (If there's time)

### Setup

```
rm -rf /tmp/{master,slave}
mkdir /tmp/{master,slave}
./mongod --master --dbpath /tmp/master --port 2000 > /tmp/master/log &
./mongod --slave --dbpath /tmp/slave --port 5000 --source localhost:2000 > /tmp/slave/log &
multitail /tmp/{master,slave}/log
```

### Master Shell

```
mongo --port 2000
db.foo.insert({_id:1, count:1})
db.foo.update({_id:1}, {$inc: {count:1}})
db.foo.find()
use local
db.oplog.$main.find({op: {$ne: 'n'}})
```

### Slave Shell

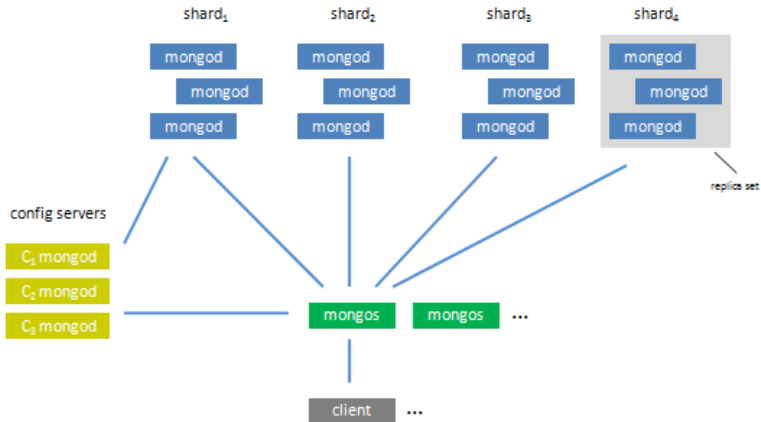
```
mongo --port 5000
db.foo.find()
```

## Sharding: Preface

- You (probably) don't need sharding!
- Wordnik.com has 1.5TB in over 5 Billion docs
  - Sustained 100,000 inserts per second during loading
  - Sustained 250,000 fetches per second during testing
  - Production Queries are 4x faster than old MySQL setup
- Speed and Scalability are different things
  - But you only need scalability if you're too slow

## Sharding: Details

- No single point of failure
- Automatic range-based partitioning
- Your app connects to a *mongos* rather than a *mongod*
- You pick which collections are sharded
- You pick a shard-key for those collections
- No other changes are necessary in your app
- We handle the rest!





# Questions?

## Links

- <http://bit.ly/mongodb-berlin-buzzwords>
- <http://github.com/RedBeard0531/MongoMagic>
- <http://try.mongodb.org> (Try mongo in your browser)
- <http://www.mongodb.org>
- #mongodb on irc.freenode.net
- mongodb-user on google groups

## Contact

- [mathias@10gen.com](mailto:mathias@10gen.com)
- @mathias\_mongo ← follow me!