

MongoDB

Or how I learned to stop worrying and love the database

Mathias Stearn

10gen

N*SQL Berlin – October 22th, 2009

- 1 What is MongoDB?
 - Document Oriented
 - JavaScript Enabled
 - Fast, Scalable, Available, and Reliable
- 2 What Makes Mongo Special?
 - Native Language Integration
 - Rich Data Types
 - Atomic Modifiers
 - Dynamic Queries
- 3 Summary
 - Use Cases
 - Links

- Organized into Databases and Collections (like Tables)
- Document Oriented
- JSON-like (BSON)
- Schemaless
- Dynamic, Strong Typing
- Database can “reach into” objects

```
db.people.insert({
  _id: "mstearn",
  name: "Mathias Stearn",
  karma: 42,
  active: true,
  birthdate: new Date(517896000000),
  interests: ["MongoDB", "Python", "Ampelmännchen"],
  subobject: {foo: "bar"}
});
```

JavaScript used for:

- Shell and Documentation
- (Very) Advanced Queries
- “Group By” Queries
- MapReduce (new)

```

db.users.find({$where: "this.a + this.b >= 42"});
db.posts.group(
  { key: "user"
  , initial: {count:0, comments:0}
  , reduce: function(doc,out){
      out.count++;
      out.comments += doc.comments.length; }
  , finalize: function(out){
      out.avg = out.comments / out.count; }
  });
  
```

- Master-Slave replication for Availability and Reliability
 - Replica-Pairs support auto-negotiation for master
- Auto-Sharding for Horizontal Scalability
 - Distributes based on specified field
 - Currently alpha
- MMAP database files to automatically use available RAM
- Asynchronous modifications

Official

- Python
- Ruby
- C++
- Perl
- PHP
- Java

Community Supported

Erlang, C#, ColdFusion, *and More*

JSON

- String (UTF8)
- Double
- Object (hash/map/dict)
- Array
- Bool
- Null / Undefined

Extras

- Date
- Int32 / Int64
- OID (12 bytes)
- Binary (with type byte)

- \$set
- \$inc
- \$multiply (soon)
- \$push / \$pushAll
- \$pull / \$pullAll

```
db.posts.update({_id:SOMEID}, {$push:{tags:"mongodb"}})
db.tags.update({_id:"mongodb"}, {$inc:{count:1}},
               {upsert:true})
```

Simple

```
db.posts.findOne({ user: "mstearn" });  
  
var cursor = db.posts.find({ user: "mstearn" });  
cursor.forEach(function() {  
    doSomething(this.text);  
});
```

Sorted

```
db.posts.find(  
  { user: "mstearn" }  
) .sort({timestamp:-1})
```

Paginated

```
db.posts.find(  
  { user: "mstearn" }  
) .sort({timestamp:-1}) .skip(10) .limit(10);
```

Simple Tag Search

```
db.posts.find(  
  { user: "mstearn"  
    , tags: "mongo"  
  }  
) .sort({timestamp:-1}) .skip(10) .limit(10);
```

Complex Tag Search

```
db.posts.find(  
  { user: "mstearn"  
    , tags: {$in: ["mongo", "mongodb"]}  
  }  
) .sort({timestamp:-1}) .skip(10) .limit(10);
```

Nested Objects

```
db.posts.find(  
  { user: "mstearn"  
    , tags: {$in: ["mongo", "mongodb"]}  
    , comments.user: "mdirolf"  
  }  
) .sort({timestamp:-1}) .skip(10) .limit(10);
```

Regular Expressions

```
db.posts.find(  
  { user: "mstearn"  
    , tags: {$in: ["mongo", "mongodb"]}  
    , comments.user: "mdirolf"  
    , text: /windows/i  
  }  
) .sort({timestamp:-1}) .skip(10) .limit(10);
```

Ranges

```
db.posts.find(  
  { user: "mstearn"  
    , tags: {$in: ["mongo", "mongodb"]}  
    , comments.user: "mdirolf"  
    , text: /windows/i  
    , points: {$gt: 10, $lt 100}  
  }  
) .sort({timestamp:-1}) .skip(10) .limit(10);
```

Arbitrary JavaScript

```
db.posts.find(  
  { user: "mstearn"  
    , tags: {$in: ["mongo", "mongodb"]}  
    , comments.user: "mdirolf"  
    , text: /windows/i  
    , points: {$gt: 10, $lt 100}  
    , $where: "this.a + this.b >= 42"  
  }  
) .sort({timestamp:-1}) .skip(10) .limit(10);
```

Good

- Websites
- Complex Objects
- High-Volume Low-Value
- Real-time analysis

Bad

- Not Spelled S-Q-L
- Complex Transactions
- Business Intelligence

- <http://www.mongodb.org>
- <http://jira.mongodb.org>
- <http://www.github.com/mongodb/>

- #mongodb on irc.freenode.net
- mongodb-user on google groups
- mathias@10gen.com