MongoDB

An Introduction

Who's Waldo

- Geek
- Problem solver
- Systems Engineer
- @gwaldo
- github.com/gwaldo

Caveats

- No affiliation with 10gen.
- Language

What is MongoDB?

- “MongoDB (from "humongous") is a scalable, high-performance, open source NoSQL database.“

Features! Quick!

- JSON Document-oriented
    - (actually BSON)

Features! Quick!

- Indexing

Features! Quick!

- Querying

```
db.geeks.find({name:"waldo"}, {sexy: 1})
```

Features! Quick!

- Replication

Features! Quick!

- Sharding

More Features

- Journaling

More Features

- Write Concern

More Features

- ‘Stored Procedures’

```
function addNumbers( x , y ) {
   return x + y;
}
```

```
db.system.js.save({_id:"addNumbers",
value:function(x, y){ return x + y; }});
```

More Features

- ‘Stored Procedures’

```
> db.eval('addNumbers(17, 25)');
42
```

More Features

- ...

More Features

- Seriously, ...

MongoDB is a ________ Database

- Document
- Open-Source
- High-Performance
- Horizontally-Scalable
- Full-Featured

MongoDB is a Document Database

- Not .pdf or .doc files
- JSON objects
- Associative Arrays (PHP array, Python Dict, Ruby & Perl Hash, etc)

MongoDB is an Open-Source Database

- https://github.com/mongodb
- AGPL license
- Originated and Sponsored by 10gen
- Commercial licenses available
- Contributions welcome

MongoDB is a High-Performance Database

- C++
- Extensive use of MMapped-files
- Runs in ALL THE PLACES
- Data serialized as BSON

MongoDB is a Horizontally-Scalable Database

- Replication
- Sharding

- Both Dead-Simple

MongoDB is a Full-Featured Database

- Rich Querying
- Real-time Aggregation
- Traditionally Consistent
- Geospatial

“NoSQL?”

- Non-Relational
- Flexible (if any) Schema
- not-ACID-ic
- Does not use SQL
    - uses a JSON Query style

Platforms

- 64- and 32-bit
  - Don’t use 32-bit
- *nix, Mac, & Windows
- Binary, source, and package managers

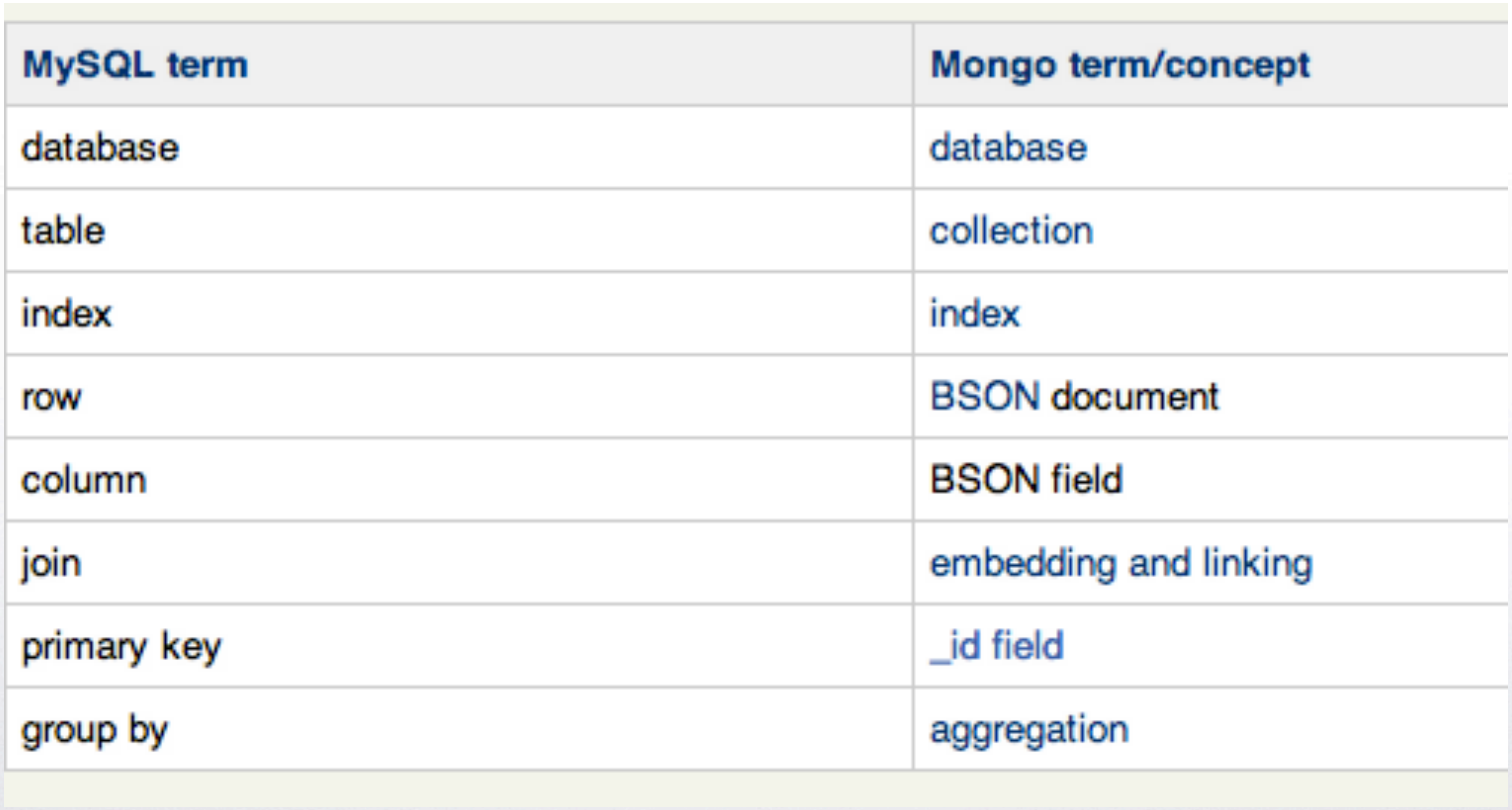
Officially SupportedLanguages

- MongoDB Supported:
    - C & C++
    - Erlang
    - Haskell
    - Java
    - JavaScript
    - .NET
    - Perl
    - PHP
    - Python
    - Ruby
    - Scala

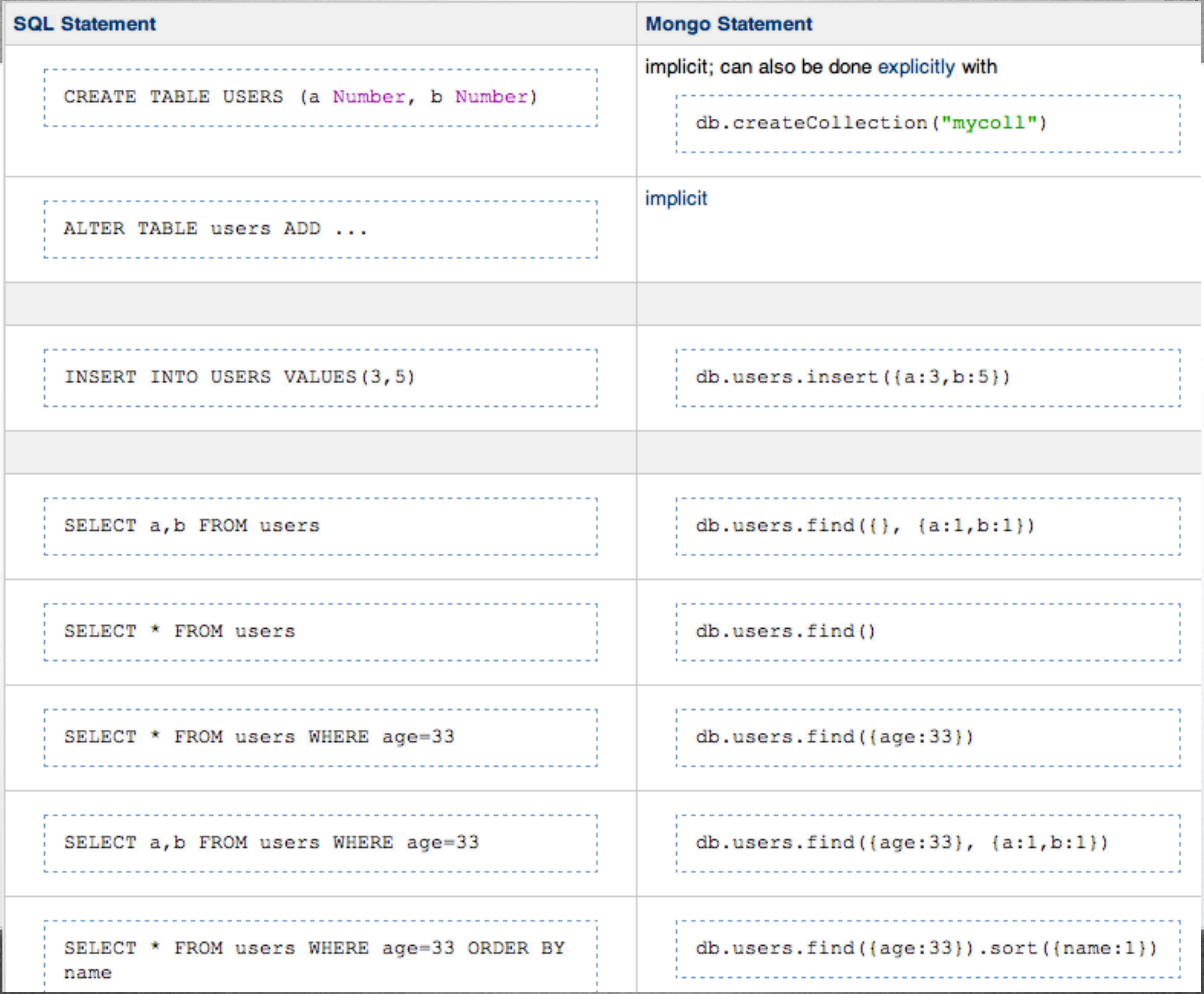
Community-Supported Languages

- Too many for me to list
- http://www.mongodb.org/display/DOCS/Drivers

SQL to MongoDB



MySQL term	Mongo term/concept
database	database
table	collection
index	index
row	BSON document
column	BSON field
join	embedding and linking
primary key	_id field
group by	aggregation



SQL Statement	Mongo Statement
`CREATE TABLE USERS (a Number, b Number)`	implicit; can also be done explicitly with `db.createCollection("mycoll")`
`ALTER TABLE users ADD ...`	implicit
`INSERT INTO USERS VALUES(3,5)`	`db.users.insert({a:3,b:5})`
`SELECT a,b FROM users`	`db.users.find({}, {a:1,b:1})`
`SELECT * FROM users`	`db.users.find()`
`SELECT * FROM users WHERE age=33`	`db.users.find({age:33})`
`SELECT a,b FROM users WHERE age=33`	`db.users.find({age:33}, {a:1,b:1})`
`SELECT * FROM users WHERE age=33 ORDER BY name`	`db.users.find({age:33}).sort({name:1})`

What it's Good at

- Archiving & Event Logging
- Documents / Content Management
- Gaming
- Mobile & Location Services
- Agile Development
- Real-Time Stats / Analysis

What it's Bad at

- Complex Transactional (Banking & Accounting)
- Traditional Data Warehousing
- Where you absolutely need SQL (complex joins)

On Joins

- Data Design
    - by-ref
    - by copy
- Separate queries in app logic

Caveats & Gotchas

- Global Write Lock
    - On Writes, read first

Caveats & Gotchas

- Queries are case-sensitive
  - var test1 = db.test.find({'tags': 'jquery'}).count();
  - var test2 = db.test.find({'tags': 'jQuery'}).count();
  - test1 == test2; // Output is false - they do not query for the same information

Caveats & Gotchas

- Don’t store numbers as strings
  - {'count': 102}; // 'count' is stored as an int
  - {'count': "102"}; // 'count' is stored as a string

Caveats & Gotchas

- Document sizes capped at 16MB
  - Not a problem for much of the world...
  - but for the rest, GridFS

Other tips

- Unless speed is paramount,
    - getLastError
- Use .limit() when using .find()
- When doing mass-updates, narrow the search AMAP

Cool Tools

- `mongostat`
- db.serverStatus()
- db.stats() & db.<collection_name>.stats()
- db.printReplicationInfo()
- db.printSlaveReplicationInfo
- <query>.explain()

What to watch for

- Page Faults
- Index Misses
- Queue Length

Massive Cop-Out

_id

- primary key
- Automatically created
- Automatically indexed
- you may override if desired

ObjectID

- special 12-byte value
- Guaranteed unique across cluster

Shell vs Drivers

- Don’t build your apps in the shell

Officially SupportedLanguages

- MongoDB Supported:
  - C & C++
  - Erlang
  - Haskell
  - Java
  - JavaScript
  - .NET
  - Perl
  - PHP
  - Python
  - Ruby
  - Scala

Now what?

- Lots of concepts introduced

Now what?

- Documentation is awesome
- 10gen.com/presentations
- “Snail in a Turtleneck” blog
  - Kristina Chodorow

Now what?

- Play!
- Talk about it!