Getting Started With MongoDB

Agenda

- MongoDB Overview
- Mongo Test Drive
- Mongo Data Model
- CRUD Operations
- Working With Files

MongoDB Overview

Data Store for JSON Objects

MongoDB Overview

Data Store for JSON Objects

```
{
  "Name" : "Rose Tyler"
}
```

JSON Objects

- A JSON Object is a collection of key/value pairs
- Keys are simple strings
- Values can be: Numbers, Strings, Arrays, Other Objects, and more

JSON Examples

- {
  “name”: “The Doctor”,
  “age”: 900
  }

- {
  “race”: “human”,
  “body parts” : [“head”, “legs”, “arms”, “eyes”]
  }

MongoDB Overview

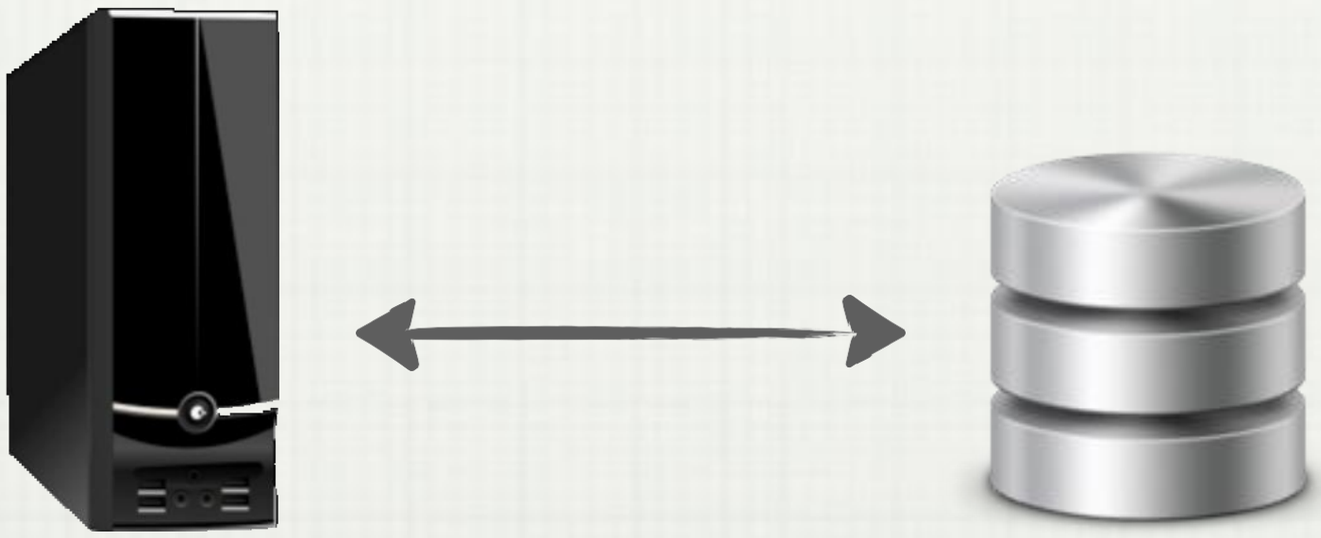
- A Document Oriented Database (No SQL)

Keeping It Simple

Keeping It Simple

- No Transactions
- No Joins

Application Architecture



SERVER

DB

What Can Mongo Do For You

- Create and store objects
- Arrange them in collections
- Retrieve them later

Q & A

Mongo Test Drive

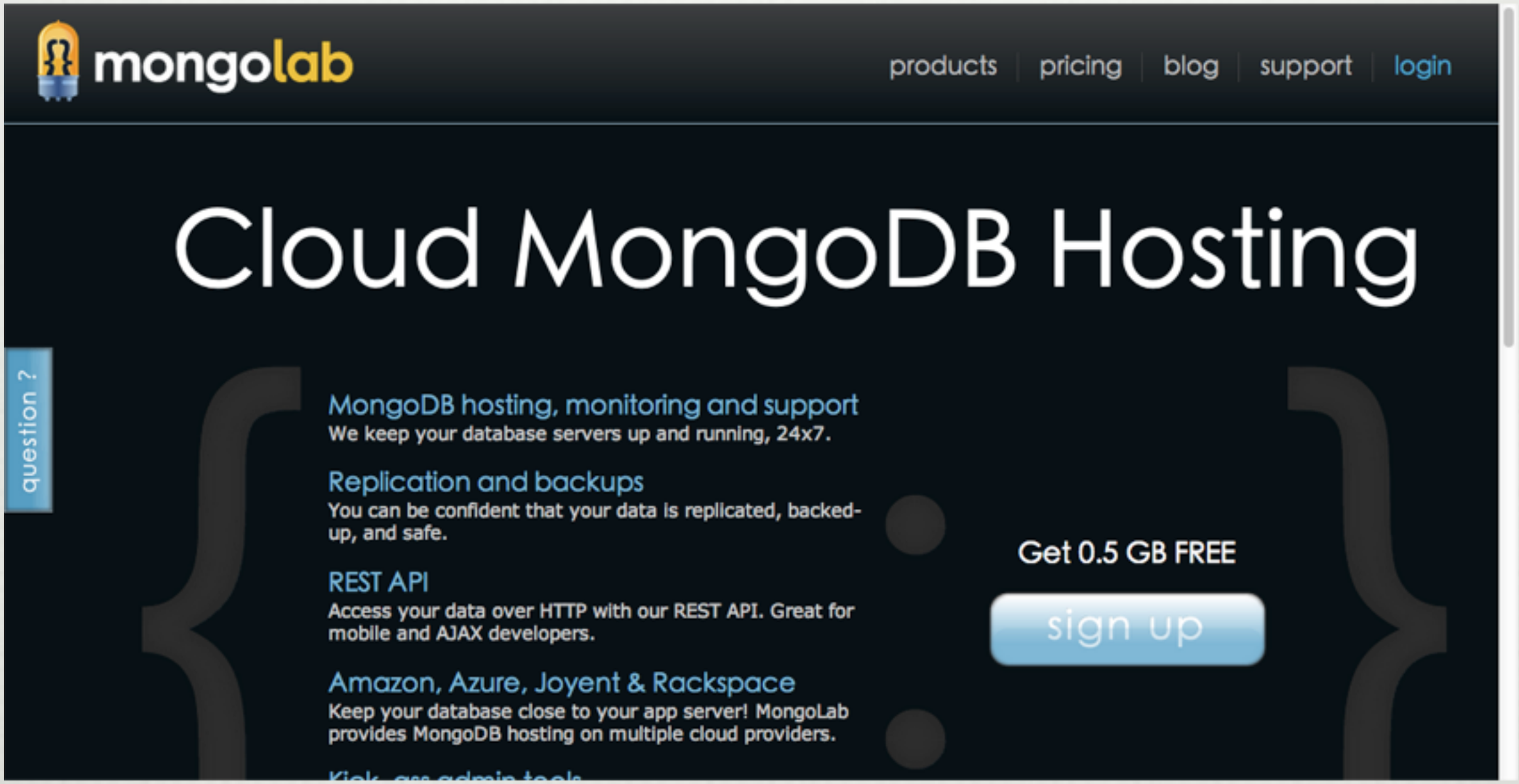
Create MongoLab Account And Start Using The DB

Install mongo Client

- Download mongo from: http://www.mongodb.org/downloads
- Extract zip file
- Run mongo

MongoLab

Webpage Screenshot



https://mongolab.com/home

Database Dashboard

Webpage Screenshot

mongolab

products | pricing | blog | support | logout

{ user: "ynonperek@gmail.com", account: "ynonp" }

Home

Databases

Clone existing | Create new

DATABASE	PLAN	CLOUD	COLLECTIONS	DOCUMENTS	DATA SIZE	FILE SIZE
cc	Free	EC2 us-east-1	8	24	1.35 KB	16 MB
liand	Free	EC2 us-east-1	9	85	19.54 KB	16 MB
rdb	Free	EC2 us-east-1	8	77	1.63 MB	16 MB

question ?

Remote Connections

Create new

[None at this time]

Dedicated Clusters

Create new

[None at this time]

MongoLab Home

This is home base for your Mongol
Each of your databases that are ho
of our multi-tenant servers appear
"Databases."

If you have any replica set clusters
dedicated VMs, they will appear un
"Clusters." If you would like to add
please contact us about our dedica

To connect to your database(s), us
connection info that is supplied on
database's admin page.

https://mongolab.com/home

Create New Database

- ☐ Choose database name
- ☐ Choose provider
- ☐ Choose plan (free is good)
- ☐ Create a DB user

Database Dashboard

Webpage Screenshot

mongolab

products | pricing | blog | support | logout

{ user: "ynonperek@gmail.com", account: "ynonp" }

Home

Database: testdb

Delete database

To connect using the shell:

```
$ mongo ds045007.mongolab.com:45007/testdb -u <dbuser> -p <dbpassword>
```

To connect using a driver via the standard URI (what's this?):

```
mongodb://<dbuser>:<dbpassword>@ds045007.mongolab.com:45007/testdb
```

mongod v2.2.2

question ?

Collections | Users | Stats | Tools

Collections

Open API view | Add

[None at this time]

System Collections

NAME	DOCUMENTS	SIZE

Connect to your database

Use the connection information on the top of the "Collections" tab of this page with either a:

(1) MongoDB interactive shell

If you want to use MongoDB's shell, you need to download MongoDB.

We recommend that you use our open-source tool, mongoctl, to help you (Mac OSX, Linux and Unix only).

To install MongoDB using mongoctl, follow these steps (you may need to first install pip):

```
$ sudo pip install mongoctl
$ mongoctl install-mongodb
```

To connect to your database, paste your URI into the following command:

```
$ mongoctl connect <YOUR-URI>
```

https://mongolab.com/databases/testdb

Connecting To The DB

- There are two options to work with your new DB
  - You can use the web console
  - You can use the command line console
- Let’s start with the web.

Demo: Creating Documents

- Create a few documents on the web console
- Update the data
- Delete some of them
- Search by fields

Mongo Data Model

- Let’s model A blog post in a blog app
- What’s The Data ?
- How Should You Save It ?

Cool MongoDB Design

```
{
  “title”: “Mongo 101”,
  “author” : “ynonp”,
  “comments” : [
    { “author” : “...”, “content” : “...” },
    { “author” : “...”, “content” : “...” }
  ],
  “tags” : [ “funny”, “informative”],
  “content” : “...”
}
```

Q & A

Lab

- Create a DB for musical info
- Create a collection called albums
- Add info for 3 albums you like, including:
  - Album Name, Artist, Tracks, Release Date, Genres
  - Tracks is an array of objects
  - Genres is an array of strings

CRUD Operations

Create, Read, Update and Destroy Data

Mongo CRUD

- Create is called insert
- Read is called find
- Update is called update
- Destroy is called remove

Mongo CRUD

- From a developer's perspective, MongoDB operations are the same through the driver and through the console
- In both cases, operations look like function calls or method invocations
- We'll use mongo shell for the rest of this chapter

Inserting Data

- Use the command *insert* or *save* to insert a new object
- `db.collection.insert( obj );`
- `db.collection.insert( array );`

Inserting Data

- Inserting to a new collection creates the collection
- Inserting an object with an _id key, it is used as the object's id (and must be unique).

Reading Data

- *find* and *findOne* perform read operations
- Both take a query
- *find* returns a cursor
- *findOne* returns an object

Optional: Fields to fetch

- `db.collection.find( <query>, <projection> )`

Query Document

- An empty (or missing) query document returns everything
- `db.collection.find({})`
- `db.collection.find()`

Query Document

- Each key/value pair in the query document imposes a condition on the results (objects that match).
- `db.movies.find({ “genre” : “indie” });`
- `db.books.find({“pages” : { “$gt” : 100 }});`

Query Document

Query Object

- Each key/value pair in the query document imposes a condition on the results (objects that match).
- `db.movies.find({ “genre” : “indie” });`
- `db.books.find({“pages” : { “$gt” : 100 }});`

Query Document

- A compound query means a logical AND on the conditions.
- db.inventory.find(

```
    {
        “type” : “snacks”,
        “available” : { “$lt” : 10 }
    });
```

Quiz: What Is Returned

- {
  "publisher" :
  "DC"
  }

from	alterego	publisher
Earth	Bruce Wayne	DC
Earth	Peter Parker	Marvel
Krypton	Clark Kent	DC

Quiz: What Is Returned

- {
  "publisher" :
  "DC",
  "from" :
  "Earth"
  }

from	alterego	publisher
Earth	Bruce Wayne	DC
Earth	Peter Parker	Marvel
Krypton	Clark Kent	DC

More Queries

- You can use “$or” to have an OR expression
- {

```
    “$or” : [
      { “type” : “food” },
      { “type” : “drinks” }
    ]
  }
```

Sub Documents

- If your document has sub-documents, it's possible to query by a full sub document or look for a partial match
- Full sub-document query means *subdocument is exactly as* specified in the query
- Example:

```
{ ISBN : { "ISBN-10" : "1906465592",
           "ISBN-13" : "978-1906465599" }}
```

Sub Documents

- A partial query matches all objects that *have* at least the required field (but may contain more)
- Example:

```
{
  "language.primary" : "english"
}
```

- Value of *language* is an object, and it *has a field* called *primary*

Arrays

- You can use an *exact array match* by providing the full array in the query
- Example:

```
{
  tags : [ “funny”, “cute”, “cats” ]
}
```

Arrays

- You can query for an array that *has at least* one element matching the query
- Example:

```
{ “tags” : “funny” }
```

Arrays

- If you have a *subdocument as the element* of an array, it's possible to query by its fields using the dot notation.

- Examples:

  { "tracks.4.name" : "Rose Mary Stretch" }

  { "tracks.name" : "Rose Mary Stretch" }

Query Operators

- Complex queries are performed with special operators.
- These are reserved words starting with a $
- Some of them: $gt, $gte, $lt, $lte, $ne, $in, $nin, $all, $or, $not

Comparator Queries

- Value for key a is greater than 10
  `{ "a" : { "$gt" : 10 }}`
- Value for key b is not 7
  `{ "b" : { "$ne" : 7 }}`
- Value for key name is greater (dictionary sort) than 'bird'
  `{ "name" : { "$gt" : "bird" }}`

Queries: $in, $nin

- Use $in to specify a choice from multiple options
- Value for grade is 85, 90 or 100

```
{ "grade" : { "$in" : [ 85, 90, 100 ] } }
```

- Value for fruit is neither apple nor banana

```
{ "fruit" : { "$nin" : ["apple", "banana" ] } }
```

Quiz: What Is Selected

- { “reads” : { “$gt” : 10 },
  “author” : { “$nin” : [“admin”, “manager”, “boss”] } }

author	reads	title
admin	99	How To Use Mongo
Joe	120	How To Make Money
Jim	8	Windows Manual

Queries: $all

- Select objects with array containing all elements
- Example:

```
{ “tags” : { “$all” : [ “funny”, “cats” ] } }
```

More Query Operators

- “$size” - array has a specific number of elements
- “$exists” - field present or missing
- Example:

  { “friends” : { “$size” : 7 } }

  { “producer” : { “$exists” : false } }

Aggregation

- count() - returns how many objects found
- distinct() - returns all distinct values for a key
- Example:

db.posts.distinct(“tags”)

Resources

- Queries Cheat Sheet
  https://github.com/vunb/backend-nodejs/blob/master/slides/Module%205/mongodb_qrc_queries.pdf

Q & A

Lab

- Using the previously defined musical DB. Query for:
- Albums released after/before 2008
- Albums with 7 tracks
- Albums by a specific genre
- Albums by a specific track name
- Display ALL different genres in the DB

Update

- Update operations modify existing data in the DB
- Mongo supports two update commands: *update()* and *save()*
- Update is the more general (and complex)

Update

- The general form for update is:

- db.collection.update(
  <query>, <update>, <options>)

Which Entries to update

What to do with them

Update

- The second argument to *update()* is an operator object
- It tells update what to do with the data
- Some keys you can use: “$set”, “$inc” “$push”, “$pushAll”, “$addToSet”, “$pop”, “$pull”, “$pullAll”

Update: set

- $set modifies a value or add a new value
- Example:

```
db.posts.update(
  { title: “Why Is Your Cat Unhappy” },
  { $set : { “archived” : true } }
);
```

Quiz: $set

- What happens here ?

```
db.cats.update(
  { color: “white” },
  { “$set” : { “owners” : [“John”, “Jim”] } }
);
```

Quiz: $set

- Update owners array of the first cat with white color
- If you want to update all objects, use multi

```
db.cats.update(
  { color: “white” },
  { “$set” : { “owners” : [“John”, “Jim”] } }
  { multi : true }
);
```

Update: inc

- $inc increases a numeric value
- Example:

```
{ "$inc" : { "age" : 11 } }
```

Quiz: $inc

- What happens here ?

```
db.songs.update(
  { “title” : “Killing Lies” },
  { “$inc” : { “plays” : 1 } }
);
```

Update: push and pushAll

- push() and pushAll() add items to an existing array
- If they array did not exists, it is created
- Example:

```
db.creatures.update(
  { name: “The Doctor” },
  { “$push” : { companions : “Rose Tyler” } }
)
```

Update: addToSet

- The $addToSet adds a new item only if it wasn’t already in the array
- Example:

  { “$addToSet” : { “tags” : “funny” } }

Update: pop

- pop removes items of an array
- Use a value of 1 to remove the last element
- Use a value of -1 to remove the first element
- Example:

```
{ “$pop” : { “companions” : 1 } }
```

Update: pull

- Remove a specific item from an array.
- Can use $pullAll to remove all matching elements
- Example:

```
{ “$pull” : { “companions” : “Rose Tyler” } }
```

Updating with save()

- The second update operation is *save()*
- takes a document:
  - If the document has an id - update it
  - If not, insert it to the DB

Deleting Data

- *remove()* deletes objects from a collection
- Takes a query and possibly a <justOne> arguments
- Examples:

```
db.posts.remove({ “author” : “Father Angelo” })

db.music.remove({ “genres” : “pop” })

db.posts.remove({ “tags” : “funny” }, 1 );
```

Q & A

Lab

- From the previous music database:
  - Add a new album with 4 tracks
  - Add a new track to that new album
  - Set property “plays” on all albums to 6
  - Increase it by 4 only for “indie” albums
  - Delete all “indie” music