CSCI-UA:0060-02

Database Design & Web Implementation

Professor Evan Sandhaus
sandhaus@cs.nyu.edu
evan@nytimes.com

Lecture #26: MongoDB Part Dos
Administrivia

- Homework
  - HW 9 Is Going Out
- Do Some Reading
  - Check Out Python and MongoDB Book
On The Menu

- MongoDB - All Day Long
  - Up and running on i6
  - Inserting Data
  - Querying Data
Mongo: The Parts

- mongod - The database server
- mongo - The database client
- There are - of course - clients for many programming languages.
## Mongo The Basics By Analogy

<table>
<thead>
<tr>
<th>MySQL</th>
<th>Mongo</th>
</tr>
</thead>
<tbody>
<tr>
<td>database</td>
<td>database</td>
</tr>
<tr>
<td>table</td>
<td>collection</td>
</tr>
<tr>
<td>row</td>
<td>document</td>
</tr>
<tr>
<td>column</td>
<td>field</td>
</tr>
<tr>
<td>primary key</td>
<td>primary key</td>
</tr>
</tbody>
</table>
JSON - Remember Me?

```json
{
  "numeric_key" : 9,
  "string_key" : "Hello",
  "bool_true_key" : true,
  "bool_false_key" : false,
  "null_key" : null,
  "array_key" : [true,"2","three"],
  "dictionary_key" : {
    "a" : "there"
  }
}
```
Every document inserted into a collection is assigned a unique _id field if no such field is specified.

_id is generated based on a number of factors (time, previous inserts, current machine).
## Document Example

<table>
<thead>
<tr>
<th>artist_name</th>
<th>album_name</th>
<th>release_year</th>
</tr>
</thead>
<tbody>
<tr>
<td>Green Day</td>
<td>Dookie</td>
<td>1993</td>
</tr>
<tr>
<td>Green Day</td>
<td>Uno</td>
<td>2012</td>
</tr>
<tr>
<td>Green Day</td>
<td>Dos</td>
<td>2012</td>
</tr>
<tr>
<td>Green Day</td>
<td>Tre!</td>
<td>2013</td>
</tr>
<tr>
<td>Lady Gaga</td>
<td>Born This Way</td>
<td>2012</td>
</tr>
</tbody>
</table>
Table To Documents

[
{
    "artist_name" : "Green Day",
    "albums" : [
        {"album_name" : "Dookie", "release_year" : 1993},
        {"album_name" : "Uno", "release_year" : 2012},
        {"album_name" : "Dos", "release_year" : 2011},
        {"album_name" : "Tre!", "release_year" : 2013}
    ]
},
{
    "artist_name" : "Lady Gaga",
    "albums" : [
        {"album_name" : "Born This Way", "release_year" : 2011}
    ]
}]

Thursday, May 2, 13
## Transforming A Complete Database

<table>
<thead>
<tr>
<th>professor_id</th>
<th>first_name</th>
<th>last_name</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Evan</td>
<td>Sandhaus</td>
</tr>
<tr>
<td>2</td>
<td>Tara</td>
<td>Bobiak</td>
</tr>
<tr>
<td>3</td>
<td>Matt</td>
<td>Boggie</td>
</tr>
<tr>
<td>4</td>
<td>Jim</td>
<td>Boehmer</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>class_id</th>
<th>class_name</th>
<th>professor_id</th>
<th>enrollment</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Intro To Database</td>
<td>1</td>
<td>116</td>
</tr>
<tr>
<td>2</td>
<td>Knowledge Management</td>
<td>1</td>
<td>42</td>
</tr>
<tr>
<td>3</td>
<td>Italian Opera</td>
<td>2</td>
<td>39</td>
</tr>
<tr>
<td>4</td>
<td>Advertising in the 21st century</td>
<td>2</td>
<td>50</td>
</tr>
<tr>
<td>5</td>
<td>Project Management</td>
<td>3</td>
<td>111</td>
</tr>
<tr>
<td>6</td>
<td>Advanced Angry Birds</td>
<td>NULL</td>
<td>86</td>
</tr>
</tbody>
</table>
To Documents!

professors = [{
    "_id" : 1,
    "first_name" : "Evan",
    "last_name" : "Sandhaus"
}, {
    "_id" : 2,
    "first_name" : "Tara",
    "last_name" : "Bobiak"
}, {
    "_id" : 3,
    "first_name" : "Matt",
    "last_name" : "Boggie"
}, {
    "_id" : 4,
    "first_name" : "Jim",
    "last_name" : "Boehmer"
}]

classes = [{
    "_id" : 1,
    "name" : "Intro to Database",
    "professor_id" : 1,
    "enrollment" : 116
}, {
    "_id" : 2,
    "name" : "Knowledge Management",
    "professor_id" : 11,
    "enrollment" : 42
}, {
    "_id" : 3,
    "name" : "Italian Opera",
    "professor_id" : 21,
    "enrollment" : 39
}, {
    "_id" : 4,
    "name" : "Advertising in the 21st Century",
    "professor_id" : 21,
    "enrollment" : 50
}, {
    "_id" : 5,
    "name" : "Project Management",
    "professor_id" : 31,
    "enrollment" : 111
}, {
    "_id" : 6,
    "name" : "Advanced Angry Birds",
    "enrollment" : 86
}]

Thursday, May 2, 13
Can We Improve this

- Joins take 2 queries
  - SQL Has Select - Mongo Has Find
  - Let's try it out.
- Let's get it down to one.
Down to One Document

classes = [{
  "_id" : 1,
  "name" : "Intro to Database",
  "professor_id" : 1,
  "enrollment" : 116,
  "professor" : {
    "first_name" : "Evan",
    "last_name" : "Sandhaus"
  }
},
  {
  "_id" : 2,
  "name" : "Knowledge Management",
  "professor_id" : 11,
  "enrollment" : 42,
  "professor" : {
    "first_name" : "Evan",
    "last_name" : "Sandhaus"
  }
},
  {
  "_id" : 3,
  "name" : "Italian Opera",
  "professor_id" : 21,
  "enrollment" : 39,
  "professor" : {
    "first_name" : "Tara",
    "last_name" : "Bobiak"
  }
},
  {
  "_id" : 4,
  "name" : "Advertising in the 21st Century",
  "professor_id" : 21,
  "enrollment" : 50,
  "professor" : {
    "first_name" : "Tara",
    "last_name" : "Bobiak"
  }
},
  {
  "_id" : 5,
  "name" : "Project Management",
  "professor_id" : 31,
  "enrollment" : 111,
  "professor" : {
    "first_name" : "Matt",
    "last_name" : "Boggie"
  }
},
  {
  "_id" : 6,
  "name" : "Advanced Angry Birds",
  "professor_id" : 21,
  "enrollment" : 86
}]}
MongoDB - Quick Review

- Use a database
  - use database_name
- Insert record
  - db.collection.insert({<document>});
- Show all databases
  - show dbs
- Show all collections
  - show collections
Finding

- **Basic**
  - `db.collection.find()`

- **Exact Match**
  - `db.collection.find({"k1" : "v1"})`

- **Comparison:**
  - `db.collection.find({"k1" : {$lt : "v1"}})`
  - `$lt, $lte, $gt, $gte, $ne`

- **Negation**
  - `db.collection.find({"k1" : {$not : {expr}}})`

- **Membership**
  - `db.collection.find({"k1" : {$in : ["v1","v2",...,"vn"]}})`

- **Existence Testing**
  - `db.collection.find("k1" : {$exists : true})`
Finding In Arrays

```javascript
{  
  "artist_name" : "Green Day",
  "albums" : [  
    {"album_name" : "Dookie", "release_year" : 1993},
    {"album_name" : "Uno", "release_year" : 2012},
    {"album_name" : "Dos", "release_year" : 2011},
    {"album_name" : "Tre!", "release_year" : 2013}
  ]
}
```

- To match any item in the array:
  - `db.artist.find({"albums.album_name" : "Uno"})`;

- To match a specific index:
  - `db.artist.find({"albums.1.album_name": "Uno"})`
    - matches
  - `db.artist.find({"albums.0.album_name": "Uno"})`
    - matches nothing
It's not just finding

- Deletion
  - `db.collection.delete(query_document)`
- Other operations also take query documents
  - Too much for the time we have.
Projecting

- Return all fields
  - `db.collection.find({},{})`
- Return only specified field (and _id)
  - `db.collection.find({},{"k1" : 1})`
- Return all but specific field
  - `db.collection.find({},{"k1" : 0})`
- Return only specific fields (without _id)
  - `db.collection.find({},{"k1" : 1, "_id" : 0})`
More Finding

- With Limit
  - `db.collection.find().limit(n)`

- With Sorting
  - `db.collection.find().sort(\{"key_1" : 1|-1, "key_2" : 1|-1\})`

- Count
  - `db.collection.find().count()`
Being Distinct

- Get all unique values for specific key in collection.
  - `db.collection.distinct("k1")`
- Get number of distinct values
  - `db.collection.distinct("k1").length`
  - This is different syntax than count
Let's Do some Querying

- What classes does Evan teach?
- How many classes have enrollments of greater than 100?
- Who teaches either "Intro to Database" or "Project Management".
- How many classes are offered?
- How many classes have professors?
- How many classes have no professors?
Batch Import

- mongoimport --db presidents --collection presidents --type json --file prez.json
Presidential Questions

- What are all distinct presidential parties?
- How many presidents were Democrats, Republicans?
- Who is the most recent president?
Homework First Steps