Administrivia

- Homework
  - HW 8 Is Due Tonight at Midnight in Some Time Zone (this may help [http://en.wikipedia.org/wiki/Anywhere_on_Earth](http://en.wikipedia.org/wiki/Anywhere_on_Earth))

- Do Some Reading
  - Check Out Python and MongoDB Book
On The Menu

- Homework Questions
  - PHP, MySQL, HTML, CSS
- Return To JSON
- MongoDB
  - Up and running on i6
  - Inserting Data
  - Querying Data
“YOU MUST UNLEARN WHAT YOU HAVE LEARNED”
Mongo The Vitals

- Released in 2009 (!)
- Developed and maintained by 10Gen (right here in NYC)
- It is a NoSQL Document Database
- It's out there in a BIG BIG way
  - [http://www.mongodb.org/about/production-deployments/](http://www.mongodb.org/about/production-deployments/)
Mongo: The Parts

- mongod - The database server
- mongo - The database client
- There are - of course - clients for many programming languages.
Mongo Up An Running on i6

- [http://cs.nyu.edu/courses/spring13/CSCI-UA.0060-001/MongoDB_on_i6_sp13_bothSections.php](http://cs.nyu.edu/courses/spring13/CSCI-UA.0060-001/MongoDB_on_i6_sp13_bothSections.php)
# 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>
What is a Document?

- A document is a collection of key-value pairs that looks an awful lot like JSON.
- JSON? - Let's review.
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"
    }
}
```
## 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
      }
    ]
  }
]
```
**MongoDB - Delving Right In**

- Setup server and connect to it
  - `> use music`
    - Switches to database creating it if it does not exist
  - `> db.artist.insert({<document>});`
    - Inserts document into collection "artist" creating collection if it does not exist.
- `> db.artist.find()`
  - List all the items inserted into the specified collection.
_id

- 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).
Mongo Client Useful Commands

- **show dbs**
  - Show all databases on server

- **show collections**
  - Show all collections for currently active database
Mongo DB Finding

- SQL Has Select - Mongo Has Find
- Formal Syntax
  - `find(<query_document>,<projection_document>)`
Query Documents

- Query Document
  - Document of form `{key_1 : <test>, key_2 : <test>}`
  - E.G.
    - `{artist_name : 'Green Day'}`
    - `{albums.release_year : 2012}`
    - `{albums.release_year : {$lt : 2000}}`
      - `$lt, $lte, $gt, $gte, $ne`
    - `{artist_name : {$in : ['Green Day','Lady Gaga']}}`

- More than 20 results type it.
Projection Document

- Projection document has the following syntax
  - \{'field_1' : 0|1, 'field_2' : 0|1,... 'field_n' : 0|1\}
  - E.G.
    - \{'artist_name' : 1, '_id' : 0\} - Show the artist name hide the id.
### 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>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Intro To Database</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>Knowledge Management</td>
<td>1</td>
</tr>
<tr>
<td>3</td>
<td>Italian Opera</td>
<td>2</td>
</tr>
<tr>
<td>4</td>
<td>Advertising in the 21st century</td>
<td>2</td>
</tr>
<tr>
<td>5</td>
<td>Project Management</td>
<td>3</td>
</tr>
<tr>
<td>6</td>
<td>Advanced Angry Birds</td>
<td>NULL</td>
</tr>
</tbody>
</table>

Thursday, May 2, 13
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
},
    {
    "_id" : 2,
    "name" : "Knowledge Management",
    "professor_id" : 1
},
    {
    "_id" : 3,
    "name" : "Italian Opera",
    "professor_id" : 2
},
    {
    "_id" : 4,
    "name" : "Advertising in the 21st Century",
    "professor_id" : 2
},
    {
    "_id" : 5,
    "name" : "Project Management",
    "professor_id" : 3
},
    {
    "_id" : 6,
    "name" : "Advanced Angry Birds"
}]
```
Can We Improve this

- Joins take 2 queries
- Let's get it down to one.
Batch Import

- `mongoimport --db presidents --collection presidents --type json --file prez.json`