Course NDBI040: **Big Data Management and NoSQL Databases**

Practice 04:

Document Stores: MongoDB

Martin Svoboda

15. 12. 2015

Faculty of Mathematics and Physics, Charles University in Prague

Outline

- Document stores
  - **MongoDB**
- Assignment 4

MongoDB

Document Store

Installation

- **Download**
  - https://www.mongodb.org/downloads
  - Windows, Linux, Mac OS X, Solaris
- **Installation**
  - https://docs.mongodb.org/manual/installation/

Connection

- SSH and SFTP access
  - **nosql.ms.mff.cuni.cz:42222**
  - Login and password: as usual
- Tools
  - PuTTY and WinSCP
- Directory
  - **`/home/svoboda/mongodb/mongodb-3.2.0/`**
- Client
  - `./bin/`**`mongo`**

Database

- Help
  - **`help`**
- Create (and select) a new database
  - **`use`** `name`
    - Use your login name as a name for your database
- List existing databases
  - `show` **`dbs`**

Collections

- Create new collections
  - `db.createCollection("actors")`
  - `db.createCollection("movies")`
- List existing collections
  - `show collections`
  - `db.getCollectionNames()`

Insertion

- Insert new documents
  - `db.actors.insert({name : "Ivan Trojan"})`
  - ```
    db.actors.insert(
      {_id : 2, name : "Jitka Schneiderova"}
    )
    ```
  - `db.actors.save({name : "Jiri Machacek"})`
  - ```
    db.actors.save(
      {_id : ObjectId(), name : "Sasa Rasilov"}
    )
    ```
- Retrieve available documents
  - `db.actors.find()`

Updates

- Update existing documents
  - ```
    db.actors.update(
      {name : "Ivan Trojan"},
      {$set : {yearOfBirth : 1964}}
    )
    ```
  - ```
    db.actors.save(
      {_id : 2}, {$set : {yearOfBirth : 1973}}
    )
    ```
  - ```
    db.actors.update(
      {_id : 2}, {$set : {_id : ObjectId()}}
    )
    ```
    - Note that document identifiers are immutable
    ```

Removals

- Remove an existing document
  - `db.actors.remove({_id : 2})`
- Insert new documents
  - ```
    db.actors.update(
      {name : "Jitka Schneiderova"},
      {$set : {yearOfBirth : 1973}},
      {upsert : true}
    )
    ```
  - `db.movies.save({title : "Samotari"})`
    ```

References

- Link actors and movies
  - ```
    db.actors.update(
      {name : "Ivan Trojan"},
      {$push : {movies : "Samotari"}}
    )
    ```
  - ```
    db.actors.update(
      {name : "Ivan Trojan"},
      {$set : {"movies.0" : ObjectId("56…a4")}}
    )
    ```
    - Assume that `"56…a4"` is a document id of our movie

Querying

- Retrieve relevant documents
  - `db.actors.find({movies : {$exists : true}})`
    - Actors with specified movies
  - `db.actors.find({yearOfBirth : {$gt : 1960}})`
    - Actors that have year of birth greater than 1960
  - `db.actors.find({}, {name : 1, _id : 0})`
    - All actors projected to their names only
  - `db.actors.find().sort({name : -1})`
    - All actors sorted using their names in a descending order
  - `db.actors.find().sort({name : 1}).limit(2)`
    - The first two actors according to their names

Indices

- Explore evaluation plans
  - `db.actors.find(`
    `{movies : ObjectId("56…a4")}`
    `)`**`.explain()`**
- Create a new index
  - `db.actors.`**`createIndex(`**`{movies : 1}`**`)`**

References

- Manual
  - https://docs.mongodb.org/v3.0/
  - **CRUD operations**
    - https://docs.mongodb.org/v3.0/core/crud-introduction/

Assignment 4

Document Stores: MongoDB

Assignment 4

- **Create a non-trivial real-world application** that would work with MongoDB
  - Choose any database domain
    - E.g. cinema tickets, flight booking, study system, …
  - Create a commented script with MongoDB statements
- **Submit your solution** by e-mail
  - svoboda@ksi.mff.cuni.cz
  - Deadline: 5. 1. 2016

Assignment 4

- **Requirements**
  - Create a database with at least 3 collections
  - Insert about 5 documents into each of them
    - Use embedded documents and arrays
    - Use references
  - Define at least 2 indices
  - Perform at least 5 meaningful updates and removals
  - Express at least 5 document retrieval queries
    - Use projection, sorting, limit, …