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

Practice 02:

Key-Value Stores: Riak, Redis

Martin Svoboda

Faculty of Mathematics and Physics, Charles University in Prague

Outline

- Key-value store tutorials
  - **Riak**
  - **Redis**
- Assignment 2

Riak Tutorial

Key-Value Store

Client Connection

- **SSH and SFTP access**
  - **nosql.ms.mff.cuni.cz:42222**
  - Login: as usual
  - Password: as usual
- Tools
  - PuTTY
  - WinSCP
- Change your password
  - `passwd`

Cluster Initialization

- **Home directory**
  - `/home/holubova/riak-1.4.2/rel`
- **Cluster initialization**
  - Do not execute the following commands, our cluster is already running!
  - Starting individual nodes
    - `./riak/bin/riak start`
      `./riak2/bin/riak start`
      `./riak3/bin/riak start`
  - Joining nodes together
    - `./riak2/bin/riak-admin join -f riak@127.0.0.1`
      `./riak3/bin/riak-admin join -f riak@127.0.0.1`

Cluster Maintenance

- Check cluster status
  - `./riak/bin/riak-admin test`
  - `./riak/bin/riak-admin status`
  - `./riak/bin/riak-admin status | grep ring_members`

Database Usage

- Client
  - `http://localhost:10002/`
- HTTP requests
  - **GET** – select
  - **POST** – insert
  - **PUT** – update
  - **DELETE** – delete
- Tools
  - `curl`

CURL Command

- Curl parameters
  - **-X method**
    - Available methods: GET, POST, PUT, DELETE, …
  - **-H header**
    - Specification of additional HTTP headers
  - **-d data**
    - Data to be sent to the server using the POST method
  - **-i**
    - Include headers in the output

Data Manipulation

- **Insert new object**
  - Prefix all your bucket names with your login name
  - ```
    curl -i
    -H "Content-Type: application/json"
    -d '{"name" : "Ivan Trojan"}'
    http://localhost:10002/riak/login_actors/trojan
    ```
  - Note that a bucket will be created automatically (when missing)
- **List all the buckets**
  - ```
    curl http://localhost:10002/riak?buckets=true
    ```

Data Manipulation

- **Get bucket properties**
  - `curl http://localhost:10002/riak/login_actors/`
  - `curl http://localhost:10002/riak/login_actors/ | python -mjson.tool`
- **Retrieve particular object**
  - `curl http://localhost:10002/riak/login_actors/trojan`
- **List all bucket keys**
  - `curl http://localhost:10002/riak/login_actors?keys=true`

Data Manipulation

- **Update particular object**
  - ```
    curl –i -X PUT
    -H "Content-Type: application/json"
    -d '{"name" : "Ivan Trojan", "yearOfBirth" : "1964"}'
    http://localhost:10002/riak/login_actors/trojan
    ```
  - Check the result
- **Create another new object**
  - Use automatically generated key
  - ```
    curl -i
    -H "Content-Type: text/plain"
    -d "Jitka Schneiderova"
    http://localhost:10002/riak/login_actors/
    ```
  - Check the result

Data Manipulation

- **Delete particular object**
  - `curl -i -X DELETE http://localhost:10002/riak/login_actors/trojan`
  - Check the result

Data Manipulation

- **Insert new objects**
  - ```
    curl -i
    -H "Content-Type: application/json"
    -d '{"name" : "Jiri Machacek"}'
    http://localhost:10002/riak/login_actors/machacek
    ```
  - ```
    curl -i
    -H "Content-Type: application/json"
    -d '{"name" : "Ivan Trojan"}'
    http://localhost:10002/riak/login_actors/trojan
    ```
  - ```
    curl -i
    -H "Content-Type: application/json"
    -d '{"name" : "Samotari"}'
    http://localhost:10002/riak/login_movies/samotari
    ```
    ```

Data Manipulation

- **Create new links**
  - Links allow us to model relationships among objects
  - ```
    curl
    -H "Content-Type: text/plain"
    -H 'Link: </riak/login_actors/machacek>;
        riaktag="actor"'
    -d 'Jakub'
    http://localhost:10002/riak/login_movies/samotari
    ```
  - ```
    curl
    -H "Content-Type: text/plain"
    -H 'Link: </riak/login_actors/trojan>;
        riaktag="actor"'
    -d 'Ondrej'
    http://localhost:10002/riak/login_movies/samotari
    ```
  - Multiple links can be specified within one header

Data Manipulation

- **Check the created links**
  - `curl -v http://localhost:10002/riak/login_movies/samotari`
- **Traverse existing links**
  - `curl -i http://localhost:10002/riak/login_movies/samotari/login_actors,actor,1`
  - Restriction to a particular bucket: `login_actors`
  - Restriction to a particular link tag: `actor`
  - `_` wildcard to represent any bucket or tag
  - `0/1` to include the step in the result
  - More navigation steps are possible

Riak References

- **Documentation**
  - http://docs.basho.com/riak/latest/

Redis Tutorial

Key-Value Store

Getting Started

- **Home directory**
  - `/home/holubova/redis-2.6.16/`
- **Database connection**
  - Server is already running
  - Open client connection
    - `./src/`**`redis-cli`**
  - Basic commands
    - **SELECT** – selects a particular database (via its number)
      - Use the assigned database for your data
      - Otherwise prefix all key names using your login name
    - **QUIT** – closes the connection

Basic Commands

- **Basics**
  - EXPIRE, TTL, …
- **Strings**
  - SET, GET, INCR, DECR, APPEND, STRLEN, GETRANGE, …
- **Lists**
  - LPUSH, RPUSH, LINSERT, LPOP, RPOP, LREM, LRANGE, LINDEX, LLEN, …
- **Sets**
  - SADD, SREM, SISMEMBER, SUNION, SINTER, SDIFF, SUNIONSTORE, …, SCARD, SMEMBER, …

Basic Commands

- **Sorted sets**
  - ZADD, ZREM, ZINCRBY, ZCARD, ZCOUNT, ZRANGEBYSCORE, ZUNIONSTORE, ZINTERSTORE, ZDIFFSTORE, …
- **Hashes**
  - HSET, HMSET, HGET, HMGET, HGETALL, HKEYS, HVALS, HDEL, HEXISTS, HLEN, …

Redis References

- **Commands**
  - http://redis.io/commands
- **Data types**
  - http://redis.io/topics/data-types-intro
- **Documentation**
  - http://redis.io/documentation

Assignment 2

Key-Value Stores: Riak, Redis

Assignment 2

- **Choose any real-world database domain**
  - E.g. cinema tickets, flight booking, study system, …
- **Create non-trivial applications**
  - Data + operations + comments
  - **Riak**: standard shell script
  - **Redis**: script with commands
- **Submit your solution** by e-mail
  - svoboda@ksi.mff.cuni.cz

Assignment 2

- **Riak**
  - Create at least three buckets for objects of different kinds (prefix bucket names with your login name)
  - Insert, update, delete and retrieve non-trivial data
  - Define and traverse non-trivial links
- **Redis**
  - Assume objects of at least three different kinds
  - Use at least two advanced data structures (lists, sets, sorted sets and hashes)
  - Insert, update, delete and retrieve non-trivial data