



Modern Database Systems

Practicals: Riak

Doc. RNDr. Irena Holubova, Ph.D.

holubova@ksi.mff.cuni.cz

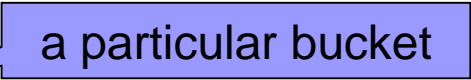
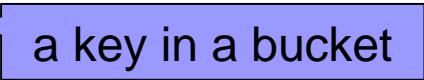
Key-value store



Riak

- Open source, distributed database
 - First release: 2009
 - Implementing principles from Amazon's Dynamo
- OS: Linux, BSD, Mac OS X, Solaris
- Language: Erlang, C, C++, some parts in JavaScript
- Built-in MapReduce support
- Stores keys into **buckets** = a namespace for keys
 - Like tables in a RDBMS, directories in a file system, ...
 - Have a set of common properties for its content
 - e.g., number of replicas

Riak Usage

- HTTP – default interface
 - GET (retrieve), PUT (update), POST (create), DELETE (delete)
 - Other interfaces: Protocol Buffers, Erlang interface
- We will use HTTP via curl (**curl --help**)
- Keys and buckets in Riak:
 - Keys are stored in buckets (= namespaces) with common properties
 - `n_val` – replication factor
 - `allow_mult` – allowing concurrent updates
 - ...
 - If a key is stored into a non-existing bucket, it is created
 - Keys may be user-specified or generated by Riak
- Paths:
 - `/riak/<bucket>` 
 - `/riak/<bucket>/<key>` 

Riak Usage

- Note: If you use the school installation, note that anyone can access anything. So:
 - Use your own bucket names (check if they exist)
 - Prefix all the bucket names with your <login>
 - Save all your command

Basic operations on objects

- Create: POST or PUT methods
 - Inserts a key-value pair into a given bucket
 - Key is specified manually, or will be generated automatically
- Read: GET method
 - Retrieves a key-value pair from a given bucket
- Update: PUT method
 - Updates a key-value pair in a given bucket
- Delete: DELETE method
 - Removes a key-value pair from a given bucket

HTTP API

■ cURL tool

- Allows to transfer data from / to a server using HTTP (or other supported protocols)

■ Options

- `-X command, --request command`
 - HTTP request method to be used (GET, ...)
- `-d data, --data data`
 - Data to be sent to the server (implies the POST method)
- `-H header, --header header`
 - Extra headers to be included when sending the request
- `-i, --include`
 - Prints both headers and (not just) body of a response

Riak Usage – Examples

Working with Buckets

- List all the buckets:

```
curl http://localhost:10011/riak?buckets=true
```

- Get properties of bucket `foo`:

```
curl http://localhost:10011/riak/foo/ |  
python -mjson.tool
```

pretty-print JSON output

- Get all keys in bucket `foo` (check the difference):

```
curl http://localhost:10011/riak/foo?keys=true
```

- Change properties of bucket `foo`:

```
curl -X PUT http://localhost:10011/riak/foo  
-H "Content-Type: application/json"  
-d '{"props" : { "n_val" : 4 } }'
```

Riak Usage – Examples

Working with Data

HTTP POST

- Storing a plain text into bucket `foo` using a generated key:

```
curl -i -H "Content-Type: plain/text"
  -d "My text" http://localhost:10011/riak/foo/
```

- Storing a JSON file into bucket `<login>_artists` with key `Bruce`:

```
curl -i -H "Content-Type: application/json"
  -d '{"name": "Bruce"}'
  http://localhost:10011/riak/<login>_artists/Bruce
```

- Getting an object:

```
curl
  http://localhost:10011/riak/<login>_artists/Bruce
```

HTTP GET

Riak Usage – Examples

Working with Data

- Updating an object:

HTTP PUT

```
curl -i -X PUT -H "Content-Type: application/json"  
  -d '{"name":"Bruce", "nickname":"The Boss"}'  
  http://localhost:10011/riak/<login>_artists/Bruce
```

```
curl http://localhost:10011/riak/<login>_artists/Bruce
```

- Deleting an object:

HTTP DELETE

check the value

```
curl -i -X DELETE  
  http://localhost:10011/riak/<login>_artists/Bruce
```

```
curl http://localhost:10011/riak/<login>_artists/Bruce
```

Riak Links

- Allow to create relationships between objects
 - Like, e.g., foreign keys in relational databases, or associations in UML
- Attached to objects via `Link` header

- Add albums and links to the performer:

```
curl -H "Content-Type: text/plain"  
  -H 'Link: </riak/<login>_artists/Bruce>; riaktag="performer" '  
  -d "The River"  
  http://localhost:10011/riak/<login>_albums/TheRiver
```

```
curl -H "Content-Type: text/plain"  
  -H 'Link: </riak/<login>_artists/Bruce>; riaktag="performer" '  
  -d "Born To Run"  
  http://localhost:10011/riak/<login>_albums/BornToRun
```

```
curl -v http://localhost:10011/riak/<login>_albums/TheRiver
```

-v = more talkative

check the links

Riak Links

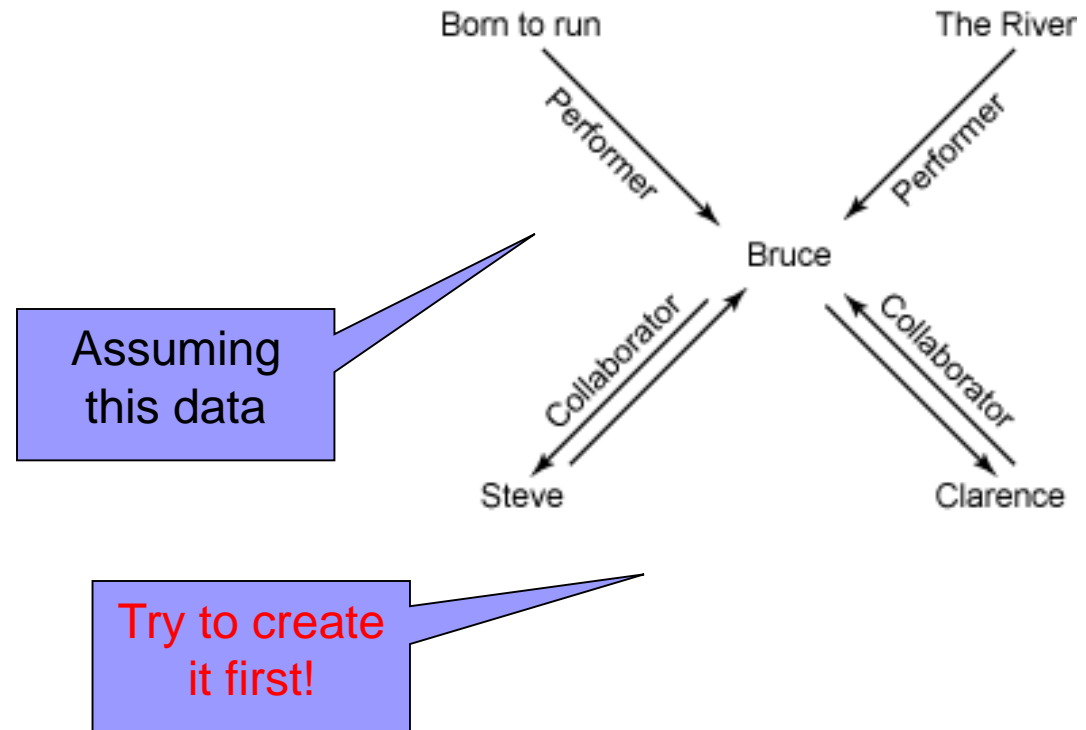
- Find the artist who performed album The River

```
curl -i
```

```
http://localhost:10011/riak/<login>_albums/TheRiver/<login>_artists,performer,1
```

- Restrict to bucket <login>_artists
- Restrict to tag performer
- 1 = include this step to the result

Riak Links



- Which artists collaborated with the artist who performed The River

```
curl -i
```

```
http://localhost:10011/riak/<login>_albums/TheRiver/<login>_artists,  
_,0/<login>_artists,collaborator,1
```

- _ = wildcard (any relationship)
- 0 = do not include this step to the result



(Optional) Assignment

- Chose your unique problem domain
 - E.g., the results of football matches of various teams
- For your selected problem domain think about an application that uses Riak buckets and Riak links
- Submit a script with respective commands for Riak + explanatory comments

References

- Riak KV: <https://docs.riak.com/riak/kv/2.1.4/>
- Riak KV Search:
<https://docs.riak.com/riak/kv/2.1.4/developing/usage/search/>