Exercise 1:
A **First Touch** On **NoSQL** Servers

1. Work to do

This practical exercise is intended to make you reason about the differences of principles between "classic" RDBMS and NoSQL servers for building and managing data. DO NOT expect to acquire an extensive practice of the use of the DBMS used here. Rather expect to be able to:

- Point out some of their principles concerning: data model, design process, internal and external data management, architecture;
- Compare these principles with those of classic RDBMS.

Therefore we propose an exercise organized into two parts where you are supposed to run some tests and answer control questions for making sure that you are achieving the expected objective of the exercise.

2. Building and Querying a NoSQL Oriented Database

In this exercise you will populate and query a NoSQL database using data coming from the Deezer music service. In particular, you will use data related with the rock band Muse. In Deezer, Muse has the ID 705. Thus, the following links retrieve Muse's albums and information about similar rock bands.

- http://api-v3.deezer.com/2.0/ artist/705/albums
- http://api-v3.deezer.com/2.0/artist/705/related

For simplifying the experiment, we have included the data obtained by these requests in the files of the folder **CouchDB/Deezer** of the Practice .zip.

3. TO DO

3.1 Creating and populating a database

- Using curl on a Terminal: create the database Deezer

```
curl -X PUT http://localhost:5984/deezer
```

- Then you populate the Deezer database with the provided documents by issuing the following commands:

```
curl -T "MuseAlbums.json" http://127.0.0.1:5984/deezer/muse_albums

curl -T "MuseRelatedArtist.json" http://127.0.0.1:5984/deezer/muse_related_artists
```

-

- Open **Fouton** on your browser:

```
http://127.0.0.1:5984/_utils/index.html
```

- Access and observe the database **Deezer** on **Fouton**.

3.2 Querying a database

Execute the following queries:

- **Query 1**
  Retrieve the name and the web page of the groups that are similar to the rock band Muse.

```
Map
function(doc) {
    var artists = doc.data;
    if(doc._id == "muse_related_artists") {
       for(var i in artists)
           emit(artists[i].name, artists[i].link);
    }
}
```

- **Query 2**
  Compute the total number of the albums produced by the rock band Muse (requires to check the ***reduce check button*** in Fouton).

```
Map
function(doc) {
    var artists = doc.data;
    if(doc._id == "muse_related_artists") {
       for(var i in artists)
           emit('muse_albums', 1);
    }
}

Reduce

function(keys, values, rereduce) {
     return sum(values);
}
```

3.3 REQUIREMENTS

- CouchDB
- cURL: Tool for transferring data to a server using the protocol HTTP (among others)
- **CouchDB.zip**: Contains the data for this practice. We have retrieved these data using the following APIs:[1]

[1] Wait the next courses on RESTful services for implementing an application that can extract data from these providers

- **Deezer simple API**
  http://www.deezer.com/fr/developers/simpleapi
- **Allociné**
  http://wiki.gromez.fr/dev/api/allocine

4. Control questions (TO ANSWER)

1) Is it possible to represent data under the classic relational model (see the 1 Normal Form)?
2) Compare the notion of key in a relational schema with respect to the notion of key in key-value NoSQL approaches.
3) Why is it necessary to define views for querying a database in CouchDB? Is this a way of integrating data? Justify.

5. REFERENCES

- Introduction guide of GAE (Google App Engine)
- CouchDB, the definitive guide
- NoSQL
- DataStore