

CouchDB tricks

[CouchDb](#) is a NoSQL database based on JSON documents rather than records. We do not use SQL to query it. It uses key/value pairs rather than structured relational data.

The semi-structured data model.

<http://docs.couchdb.org/en/2.0.0/intro/tour.html>

Add an admin user

```
curl -X PUT http://admin:admin@couchdb:5984/config/admins/fabien -d '{"secret":'
curl fabien:secret@couchdb:5984
```

Create and delete a Database

```
curl -X PUT http://fabien:secret@couchdb:5984/plankton
curl -X DELETE http://admin:admin@couchdb:5984/plankton
curl -X PUT http://fabien:secret@couchdb:5984/songs
```

! If no members are defined, the database is public.

Add a document in database

```
curl -X PUT http://fabien:secret@couchdbv2:5984/songs/1 -d '{"title": "Guitar tune", "artist": "Johnny", "album": "Best of Johnny"}'
```

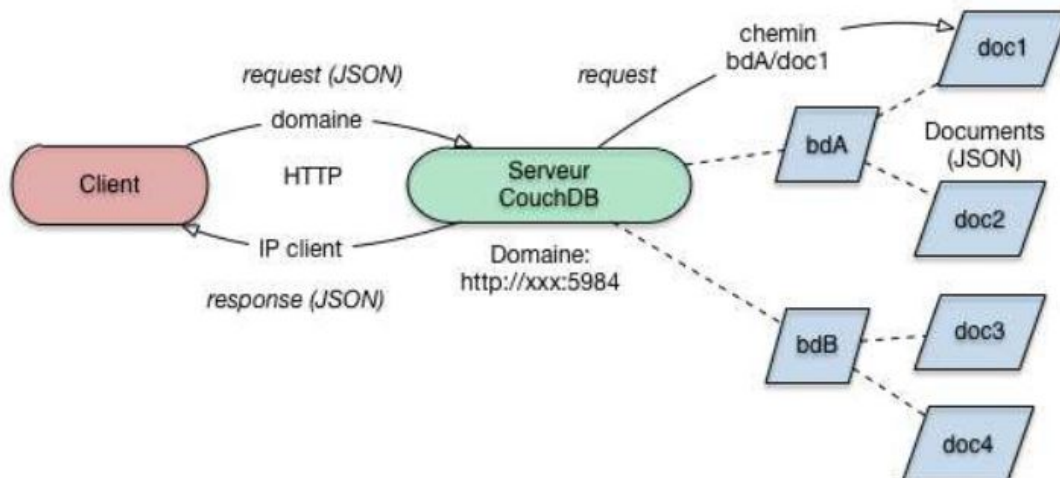
```
curl -X POST http://fabien:secret@couchdbv2:5984/test_database/00a271787f89c0ef2e10e88a0c0001f4 -d '{"foo": "bar"}'
```

```
curl -X PUT http://fabien:secret@couchdbv2:5984/test_database/5 -d '{"_id": "5", "item": "figues", "prices": { "Super U": 9.99, "Galeries Lafayette": 9.99, "Intermarché": 1.79 } }'
```

Replicate a database

```
curl -X POST -H 'Content-Type: application/json' http://fabien:secret@couchdb:5984/_replicate -d '{"source": "http://fabien:secret@couchdb:5984/test_database", "target": "http://fabien:secret@couchdb1:5984/test_database"}'
```

Distribute a database



CouchDB tricks

Creating per-user Databases script

```
#!/bin/bash
# Creating per-user Databases
# See more at https://wiki.apache.org/couchdb/

usage () {
    echo "usage: $0 username password" ;
}

if [ $# -lt 2 ] ; then
    usage; exit
else

# Set the url to couchDB server: adminuser:passwd@host:port
COUCH=http://fabien:secret@couchdb:5984

# Create the user:
curl -HContent-Type:application/json \
    -vXPUT $COUCH/_users/org.couchdb.user:$1 \
    --data-binary '{"_id": "org.couchdb.user:'$1'", "name": "'$1'", "roles": [], "type": "user", "password": "'$2'"}'

# Create the user DB
curl -vX PUT $COUCH/$1

# Update the DB security object
curl -vX PUT $COUCH/$1/_security \
    -Hcontent-type:application/json \
    --data-binary '{"admins":{"names":[],"roles":[]},"members":{"names":["'$1'"],"roles":[]}}'

fi
```

Auto increment IDs with PHP

<http://aravindavk.in/blog/couchdb-auto-increment-ids-with-php/>

CouchDB Lounge

[CouchDB Lounge](#) est une application mandataire (proxy) qui permet de partitionner et de [grappeler](#) des instances CouchDB. Lounge se compose de deux modules :

Dumbproxy s'occupe des « simples » requêtes GET et PUT visant à obtenir un document ; le second, **smartproxy** répartit les requêtes nécessitant de recourir à des vues. Il traite toutes les requêtes qui n'utilisent pas le mécanisme de vues de CouchDB. C'est un module de [nginx](#), serveur mandataire frontal (en anglais, reverse proxy) HTTP très performant. Puisque ce serveur est le point d'entrée de votre infrastructure, et parce qu'il intègre les mécanismes de mandataire, vous pouvez le configurer pour apporter la sécurité que vous désirez, le chiffrement, la répartition de charge, la compression des flux et, bien sûr, l'antémémoire (en anglais, cache) pour soulager vos bases de données.

smartproxy, traite uniquement les requêtes sur les vues CouchDB qu'il distribue entre les nœuds de la grappe afin que les performances soient fonction de la puissance de celle-ci.

CouchDB tricks

INDEX

Données autosuffisantes: a business card contains all the important information, right there on the cardstock. We call this “self-contained” data

Le théorème de Brewer, 'il est impossible sur un système [informatique](#) de [calcul distribué](#) de garantir en même temps (c'est-à-dire de manière **synchrone**) les trois contraintes suivantes:

La cohérence (Consistency)

Tous les clients de la base de données voient les mêmes données, même en cas de mises à jour concurrentes.

La disponibilité (Availability)

Tous les clients de la base de données peuvent accéder à une version des données.

La résistance au morcellement (Partition tolerance)

La base de données peut être divisée et répartie sur plusieurs serveurs.

B-Tree

CouchDB utilise une structure de données particulière pour indexer ses documents et ses vues. Il s'agit d'un B-tree, ou arbre B (ça sonne moins bien...). Les B-tree, structurent et trient les données sous forme d'un arbre équilibré. Cela permet une exécution des opérations d'insertion et de suppression en temps logarithmique.

Documents

CouchDB est une base de données orientée document. Il faut distinguer deux types de documents :

- Documents : Il s'agit de l'entité porteuse d'information.
- Design Documents : Ces documents sont un peu spéciaux puisqu'ils décrivent des fonctions qui peuvent être exécutées par CouchDB.

DocID

Tous les documents stockés sur CouchDB possèdent un *DocID*. Il s'agit d'un identifiant qui permet de distinguer un document de manière unique. Deux documents ne peuvent pas posséder le même DocID, sinon ils seront considérés comme un seul document et le second à être enregistré sera une mise à jour du premier. Cet identifiant peut être forcé par l'utilisateur, ou défini par CouchDB.

Il existe deux types spéciaux de DocID :

- *_design/* : désignent les Design Documents
- *_local/* : désignent les documents qui ne seront pas répliqués sur d'autres bases

Futon (Fauxton)

the built-in administration interface. http://couchdb:5984/_utils/

CouchDB tricks

Clustering couchdb V2

- q - The number of shards.
- n - The number of copies there is of every document. Replicas.

```
echo "deb https://apache.bintray.com/couchdb-deb xenial main" | sudo tee -a /etc/apt/sources.list  
curl -L https://couchdb.apache.org/repo/bintray-pubkey.asc | sudo apt-key add -  
sudo apt-get update && sudo apt-get install couchdb
```