

CouchDB

Team 1

Plan

- Introduction
- Structure
- Orienté document
- API REST
- Design documents
 - Views (map & reduce)
- "Eventual consistency"
- Réplication
- Conclusion
- Questions

Introduction

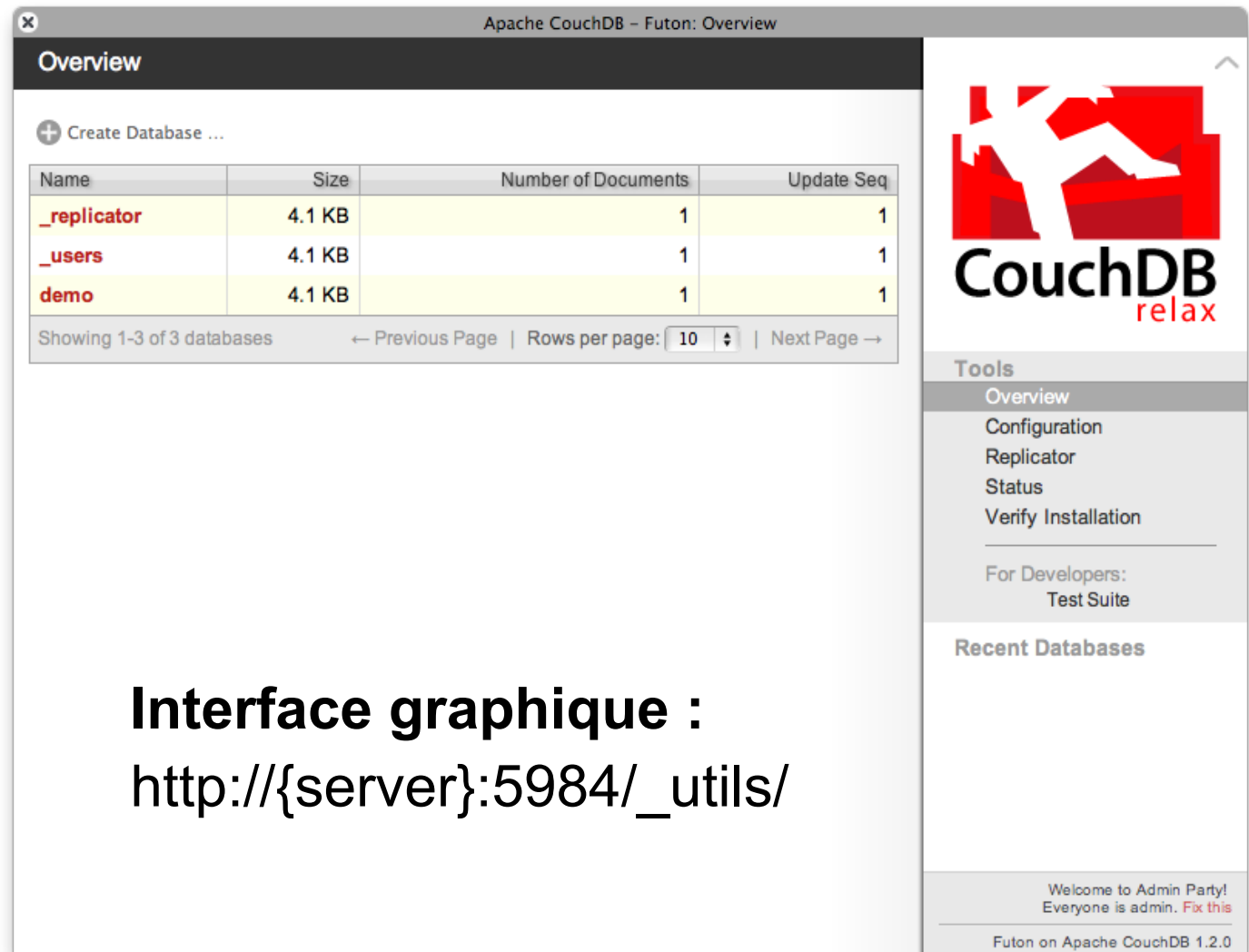
CouchDB

- **Ecrit en:** Erlang
- **Version initiale :** 2005
- **Type NoSQL :** Orienté document
- **Format :** JSON
- **Points principaux:** "eventual consistency", réplication, map-reduce
- **License:** Apache
- **Protocol:** HTTP/REST



Introduction

Futon




Apache CouchDB – Futon: Overview

Overview

+ Create Database ...

Name	Size	Number of Documents	Update Seq
_replicator	4.1 KB	1	1
_users	4.1 KB	1	1
demo	4.1 KB	1	1

Showing 1-3 of 3 databases ← Previous Page | Rows per page: 10 | Next Page →


CouchDB
relax

Tools

- Overview
- Configuration
- Replicator
- Status
- Verify Installation

For Developers:
Test Suite

Recent Databases

Welcome to Admin Party!
Everyone is admin. [Fix this](#)

Futon on Apache CouchDB 1.2.0

Interface graphique :
http://{server}:5984/_utils/

Structure

CouchDB



database

1

N



document

Orienté document

"En règle générale, un document correspond à peu de choses près à l'instance de l'objet correspondant dans votre langage de programmation." **CouchDB - Definite guide**

```
{  
  "_id": "39ccfc9709437b79b1ab88a7300f379e",  
  "_rev": "1-bfd9bc1339410d7edd1d77354038529e",  
  "appID": "39ccfc9709437b79b1ab88a7300ea576",  
  "type": "badge",  
  "name": "badge_name",  
  "description": "badge_description",  
  "URLBadge": "http://test.com/test.png",  
}
```

peut être choisi
par le client ou
par le server

choisi par le
serveur pour le
contrôle de
concurrence



Même type de document != Même structure

API REST

- **Créer une base donnée**

PUT `http://127.0.0.1:5984/my_database`

- **Créer un document**

POST `http://127.0.0.1:5984/my_database`

```
{  
  "_id": "my_badge1",  
  "name": "badge_name",  
  "type": "badge",  
  ...  
}
```

API REST

- **Informations sur le document créé**

GET `http://127.0.0.1:5984/my_database/my_badge1`

```
{  
  "_id": "my_badge1",  
  "_rev": "3-104988593fd7fc98e476fb55ca7bc46f",  
  "name": "badge_name",  
  "type": "badge",  
  ...  
}
```

- **Suppression du document**

DELETE `http://127.0.0.1:5984/my_database/my_badge1`

Design documents

Documents spéciaux, dans lesquels certaines fonctions sont définies, notamment pour répondre aux requêtes.

doit être
_design/{name}



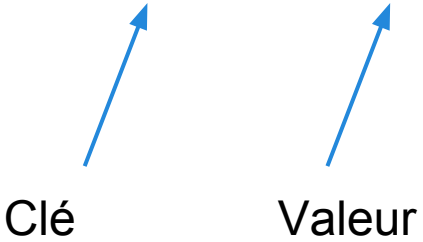
```
{
  "_id": "_design/badges",
  "_rev": "11-5d65c62dd55a3b02729d4ebd873a86f2",
  "views": {
    "my_view": {
      "map": "function (doc) {...}"
    }
  },
  "shows": {
    "my_show": "function (doc, req) {...}"
  }
}
```

Design documents (views)

Map : Construction d'une liste de clés/valeurs

Reduce : Réduction de la liste (à une valeur). Par exemple pour compter le nombre d'entrées.

```
"my_view": {  
  "map": "function (doc) {  
    if (doc.type="badge") {  
      emit(doc.name, doc);  
    }  
  }"  
}
```



Clé

Valeur

GET http://127.0.0.1:
5984/my_database/
_design/badges/_view/my_view

```
{ "total_rows": 2, "offset": 0, "  
  rows": [ { "id": "my_badge1", "key":  
    badge_name, "value": { "_id":  
      "my_badge1", "_rev": "3-  
104988593fd7fc98e476fb55ca7bc46  
f", "name": "badge_name", "type":  
      "badge", ...  
    } },  
    { "id": "my_badge2", "key": ... }  
  ] }
```

Design documents (views)

Des requêtes sur une view peuvent être faites sur le paramètre key.

`http://127.0.0.1:5984/my_database/_design/badges/_view/my_view`

`?key="badge_name1"`

`?startkey="b"&endkey="d"` // Tous les résultats commençant par "b" et "c"

Replication

- Deux copies (ou plusieurs) de la même base de données sur des serveurs distants.
- Haute disponibilité
- Une modification sur une des bases est transmise à son homologue (bidirectionnelle)
- Synchronisation grâce au versionning
- L'attribut spécial `"_conflicts":true` est ajouté aux documents en conflits. C'est à l'application de les gérer.

Eventual Consistency

La consistance

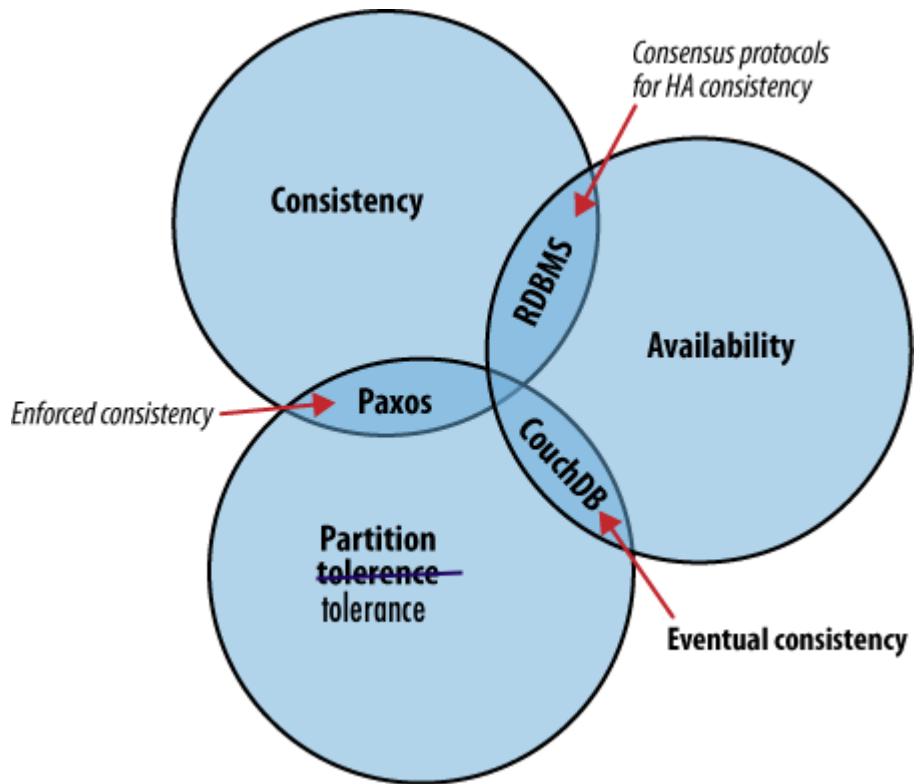
Tous les clients de la base de données voient les mêmes données, même en cas de mises à jour concurrentes.

La disponibilité

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

La tolérance au partitionnement

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



Conclusion

Nos observations :

- API REST agréable à utiliser et ne nécessite pas d'apprentissage.
- Id auto-incrément difficilement réalisable.
- Documentation en ligne encore très restreinte, beaucoup de points théoriques mais très peu d'exemples pratiques.

Références

THE reference



<http://guide.couchdb.org/>

Références

- THE map/reduce presentation : [Map/reduce in CouchDB](#)
- THE reduce example : [Writing a Reduce Function in CouchDb](#)
- THE wiki : [CouchDB wiki](#)

Questions

