



## **Base De Données NoSQL**

# TP CouchDB & MapReduce

## Introduction et prise en main de CouchDB

---

### 1. Qu'est-ce que CouchDB ?

**CouchDB** (Apache CouchDB) est une **base de données NoSQL orientée documents**.

#### Caractéristiques principales :

- Base de données **orientée documents**
  - Documents stockés en **JSON**
  - Chaque document possède un **identifiant unique** (**\_id**) et une **révision** (**\_rev**)
  - Accès aux données via une **API REST**
  - Support natif de **MapReduce** pour l'indexation et l'agrégation
  - Réplication facile entre bases de données
- 

### 2. CouchDB comme base de données orientée documents

Dans CouchDB :

- Une **base de données** est une collection de documents
- Un **document** est un objet JSON structuré
- Il n'y a **pas de schéma fixe** imposé

#### Exemple de document :

```
{
  "_id": "movie:10098",
  "title": "Inception",
  "year": 2010,
  "actors": [
    { "first_name": "Leonardo", "last_name": "DiCaprio" },
    { "first_name": "Joseph", "last_name": "Gordon-Levitt" }
  ]
}
```

Chaque document peut avoir une structure différente, ce qui rend CouchDB très flexible.

---

### 3. CouchDB et l'API REST (CRUD)

CouchDB expose **toutes ses fonctionnalités via une API REST** basée sur HTTP. Les opérations classiques **CRUD** sont directement associées aux méthodes HTTP :

Opération	HTTP	Description
Create	PUT / POST	Créer une base ou un document
Read	GET	Lire une base ou un document
Update	PUT	Mettre à jour un document
Delete	DELETE	Supprimer un document

Cela permet d'utiliser CouchDB depuis n'importe quel langage capable de faire des requêtes HTTP.

---

### 4. Lancement de CouchDB avec Docker

Pour simplifier l'installation, CouchDB est lancé dans un **conteneur Docker**.

```
docker run -d --name couchdbdemo \  
-e COUCHDB_USER=sarah \  
-e COUCHDB_PASSWORD=sarah \  
-p 5984:5984 couchdb
```

#### Explication :

- -d : exécution en arrière-plan
  - --name couchdbdemo : nom du conteneur
  - COUCHDB\_USER / PASSWORD : identifiants admin
  - -p 5984:5984 : exposition du port CouchDB
  - couchdb : image officielle
- 

### 5. Vérification du bon fonctionnement

```
curl -X GET http://sarah:sarah@localhost:5984
```

Réponse :

```
{{"couchdb":"Welcome","version":"3.5.1","git_sha":"44f6a43d8","uuid":
:"586f67f207bcabbeaed56279c4e971af","features":["access-ready","part
itioned","pluggable-storage-engines","reshard","scheduler"],"vendor"
:{"name":"The Apache Software Foundation"}}
```

---

## 6. Interface graphique Fauxton

CouchDB fournit une interface web appelée **Fauxton** :

`http://localhost:5984/_utils`

Elle permet :

- Créer des bases de données
  - Visualiser les documents
  - Créer des vues MapReduce
  - Tester les requêtes
- 

## 7. Création d'une base de données (ressource **films**)

```
curl -X PUT http://sarah:sarah@localhost:5984/films
```

Réponse :

```
{"ok":true}
```

Affichage de la base :

```
curl -X GET http://sarah:sarah@localhost:5984/films
```

---

## 8. Insertion de documents dans **films**

### Insertion simple

```
curl -X PUT http://sarah:sarah@localhost:5984/films/doc \
-H "Content-Type: application/json" \
-d '{"cle":"valeur"}
```

Le document est alors visible dans l'interface graphique.

---

### Insertion depuis un fichier JSON

```
curl -X POST http://sarah:sarah@localhost:5984/films \  
-H "Content-Type: application/json" \  
-d @movie:10098.json
```

### Insertion en masse (\_bulk\_docs)

```
curl -X POST http://sarah:sarah@localhost:5984/films/_bulk_docs \  
-H "Content-Type: application/json" \  
-d @films_couchdb.json
```

\_bulk\_docs permet d'insérer plusieurs documents en une seule requête.

---

## 9. MapReduce dans CouchDB

### Définition générale

**MapReduce** est un paradigme de programmation permettant de traiter de grandes quantités de données en deux phases :

1. **Map** : transformation des documents en paires (clé, valeur)
2. **Reduce** : agrégation des valeurs associées à une même clé

Dans CouchDB, MapReduce est utilisé pour :

- Créer des index
  - Effectuer des statistiques
  - Réaliser des agrégations
- 

## 10. Exemple 1 : nombre de films par année

### Fonction Map

```
function (doc) {  
  emit(doc.year, doc.title);  
}
```

Pour chaque document film, on émet :

- clé : l'année
- valeur : le titre du film

### Fonction Reduce

```
function (keys, values) {  
  return values.length;  
}
```

Le reduce compte le nombre de films par année.

---

## 11. Exemple 2 : nombre de films par acteur

### Fonction Map

```
function (doc) {  
  for (var i = 0; i < doc.actors.length; i++) {  
    emit(  
      {  
        "prenom": doc.actors[i].first_name,  
        "nom": doc.actors[i].last_name  
      },  
      doc.title  
    );  
  }  
}
```

Chaque acteur reçoit une entrée pour chaque film auquel il participe.

### Fonction Reduce

```
function (keys, values) {  
  return values.length;  
}
```

On obtient le nombre de films par acteur.