

TP3 — Découverte et manipulation de CouchDB

Base de Données NoSQL orientée documents

Nom : Haiballa El Varougou

Année : 2025–2026

Table des matières

1	Introduction	2
2	Origines du MapReduce : matrice de liens et calcul distribué	2
2.1	Énoncé	2
2.2	Modélisation des documents	2
2.3	Calcul de la norme des lignes	3
2.4	Produit matrice–vecteur	3
3	Installation et déploiement de CouchDB	3
3.1	Création du volume Docker	3
3.2	Lancement du conteneur CouchDB	4
3.3	Interface Web	4
3.4	Test de connexion HTTP	4
4	Création et gestion des bases	4
4.1	Création d’une base	4
4.2	Insertion de documents	4
4.3	Lecture d’un document	4
4.4	Mise à jour d’un document	4
5	Vues MapReduce	5
5.1	Comptage des films par année	5
5.2	Indexation par acteurs	5

1 Introduction

Apache CouchDB est une base de données NoSQL orientée documents, reposant sur trois piliers :

- le stockage de données au format JSON ;
- une interface d'accès via API REST ;
- un moteur MapReduce intégré pour l'analyse des données.

Sa capacité de réplication incrémentale ainsi que sa tolérance aux pannes font de CouchDB une solution bien adaptée aux architectures distribuées et aux systèmes multi-nœuds.

Ce TP présente l'installation de CouchDB, la création et la manipulation de bases de données, ainsi que la définition de vues MapReduce pour l'analyse de données.

2 Origines du MapReduce : matrice de liens et calcul distribué

2.1 Énoncé

On considère une matrice M de dimension $N \times N$ représentant les liens entre un très grand nombre de pages web. Chaque coefficient M_{ij} correspond à l'importance du lien allant de la page P_i vers la page P_j .

1. Proposer un modèle de documents permettant de représenter cette matrice sous forme NoSQL (inspiration : algorithme PageRank).
2. Décrire un traitement MapReduce permettant de calculer la norme de chaque ligne $\|M_i\|$.
3. Définir un traitement MapReduce calculant le produit $\varphi = M \cdot W$, où W est un vecteur de dimension N accessible en mémoire.

2.2 Modélisation des documents

La matrice étant très grande et creuse, seuls les liens existants sont stockés. Chaque document représente une ligne de la matrice, c'est-à-dire une page source P_i et ses liens sortants.

```
{
  "_id": "page:Pi",
  "type": "page",
  "i": 123,
  "url": "https://example.org/pi",
  "outlinks": [
    { "j": 17, "w": 0.42 },
    { "j": 93, "w": 0.10 }
  ]
}
```

Chaque élément `outlinks` correspond à un coefficient non nul M_{ij} .

2.3 Calcul de la norme des lignes

La norme d'une ligne est définie par :

$$\|M_i\| = \sqrt{\sum_{j=1}^N M_{ij}^2}$$

Avec une représentation creuse, seules les valeurs présentes sont prises en compte.

Map Pour chaque lien sortant, on émet le carré du poids.

```
function (doc) {
  if (doc.type !== "page" || !doc.outlinks) return;
  doc.outlinks.forEach(l => emit(doc.i, l.w * l.w));
}
```

Reduce La fonction de réduction consiste à sommer les valeurs.

Reduce = `_sum`

La norme est ensuite obtenue par racine carrée du résultat.

2.4 Produit matrice–vecteur

On souhaite calculer :

$$\varphi_i = \sum_{j=1}^N M_{ij} w_j$$

Le vecteur W est supposé accessible globalement.

Map

```
function (doc) {
  if (doc.type !== "page" || !doc.outlinks) return;
  var acc = 0;
  doc.outlinks.forEach(l => {
    if (W[l.j] !== undefined) {
      acc += l.w * W[l.j];
    }
  });
  emit(doc.i, acc);
}
```

Reduce La réduction consiste à sommer les contributions partielles.

Reduce = `_sum`

3 Installation et déploiement de CouchDB

3.1 Création du volume Docker

```
docker volume create couchdb_data
```

3.2 Lancement du conteneur CouchDB

```
docker run \
  --name couchdb \
  -e COUCHDB_USER=admin \
  -e COUCHDB_PASSWORD=admin \
  -p 5984:5984 \
  -v couchdb_data:/opt/couchdb/data \
  -d couchdb
```

3.3 Interface Web

L'interface graphique Fauxton est accessible à l'adresse suivante :

http://localhost:5984/_utils/

3.4 Test de connexion HTTP

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

4 Création et gestion des bases

4.1 Création d'une base

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

4.2 Insertion de documents

Insertion simple

```
curl -X POST http://admin:admin@localhost:5984/films \
  -H "Content-Type: application/json" \
  -d '{"title":"Inception","year":2010}'
```

Insertion en lot

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

4.3 Lecture d'un document

```
curl -X GET http://admin:admin@localhost:5984/films/<id>
```

4.4 Mise à jour d'un document

```
curl -X PUT http://admin:admin@localhost:5984/films/<id> \
  -H "Content-Type: application/json" \
  -d '{"_rev":"1-xxx","title":"Dune","year":2021}'
```

5 Vues MapReduce

5.1 Comptage des films par année

Map

```
function (doc) {  
  emit(doc.year, 1);  
}
```

Reduce

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

5.2 Indexation par acteurs

Map

```
function (doc) {  
  if (doc.actors) {  
    doc.actors.forEach(a => emit(a, doc.title));  
  }  
}
```

Reduce

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