RAPPORT DU DEVOIR
Base de données non traditionnelles

Réalisé par Mamadou Alpha BAH & Manel REKIK

I. Synthèse du cours

1. No-SQL :

Les Not Only SQL se sont des systèmes de base de données non traditionnelles qui désignent les bases de données qui ne sont pas fondées sur l'architecture classique des bases de données relationnelles, ils permettent utilisation combinée de SQL avec autres mécanismes de recherche d'information, leurs architectures permettent de résoudre les problèmes de performances en matière d'évolutivité de gérer du Big Data. Ils sont utiles pour accéder à de grandes quantités de données non structurées ou de données stockées à distance sur plusieurs serveurs.

Voici les différents types d'implémentation de NoSQL :

- *Les bases **clef/valeur**, permettent de stocker des informations sous forme d'un couple clef/valeur (une chaîne de caractère, un entier ou un objet sérialisé).*
- *Les bases orientées **colonnes**, ressemble aux bases de données relationnelles, car les données sont sauvegardées sous forme de ligne avec des colonnes ; exemple : Cassandra(utilisé par Twitter)*
- *Les bases orientées **document**, représente les informations sous forme d'objet XML ou JSON pour pouvoir récupérer simplement des informations structurées, exemple: MongoDB(utilisé par Sourceforge)*
- *Les bases orientées **graphe**, présentent les données sous forme de noeud et de relation. Cette structure permet de récupérer simplement des relations complexes.*

Avantages ;

- *Les bases de données non relationnelles NoSQL fournissent une excellente performance et scalabilité que les bases de données traditionnelles.*
- *Ils n'ont pas de schéma de bases avec les contraintes sur les champs. Cela apporte de la flexibilité dans la gestion des données et la rapidité.*
- *Elle permet d'adapter l'architecture des données en fonction de la charge en distribuant sur un ou plusieurs serveurs pour traiter des requêtes et gérer les gros volumes de données .*

Inconvénients :

- *N'a pas de jointures sur les ensembles de données dû à la non-formalisation des données.(Les données ne doivent pas être formalisées pour pouvoir être stockées dans un seul document).*
- *les bases de données NoSQL ont abandonné certaines fonctionnalités proposées par défaut par les bases relationnelles comme les transactions ou encore les vérifications d'intégrités.*

2. Ontologies :

L'ontologie est une représentation partagée et une base de connaissances dans un domaine donné qui présente une collection de termes, de concepts et des relations entre ces derniers. Elle permet l'étude sémantique donc de structurer, documenter et lier sémantiquement selon le sens un ensemble de données et de trouver les données correspondantes à une requête dans une base de données ainsi que d'avoir facilement le vocabulaire correspondant à un domaine de données demandé.

Avantages :

Elle permet la une mise en valeur des relations de sens entre les termes par l'expression des liens sémantiques décrivant les composants de la connaissance : les concepts et les relations entre les concepts.

Inconvénients :

- *C'est une stratégie coûteuse, moins connue que d'autres systèmes de base de données et pas encore appliquée au stade de développement industriel*
- *En utilisant ce système on aura des difficultés de partager les mêmes conceptualisations et des difficulté de préciser la sémantique d'un ensemble de mots.*

Exemples des outils de développement d'ontologies : *selon le site* *https://www.cairn.info/revue-documentaliste-sciences-de-l-information-2007-1-page-81.htm*

Les outils informatiques – éditeurs ou outils de développement d'ontologies – sont relativement nombreux mais tous ou presque sont issus d'instituts de recherche ou de laboratoires d'universités. Aucun de ces outils n'est encore parvenu au stade de développement industriel et de la commercialisation. Parmi les plus connus, on peut citer ONTOEDIT (Ontology Editor), Protege 2000 du Stanford Research Institute, ONTOLINGUA, OILED de l'Université de Manchester, WEBODE du Laboratoire d'intelligence artificielle de Madrid, Differential Ontology Editor (DOE) de l'Institut national de l'audiovisuel. Ces outils utilisent des extracteurs de terminologie (à partir d'un corpus spécifique) qui sont eux-mêmes des produits universitaires comme NOMINO, TERMINO ou LEXTER.

3. Spark :

Spark est un framework qui permet de traiter de manière complexe des données de types variés. Il se situe donc plutôt au niveau des étapes Map et Reduce qu'au niveau de l'infrastructure en elle-même. Spark se combine très bien avec des clusters Hadoop (en HDFS) et permet d'appliquer des algorithmes complexes sur des données issues de ces clusters, faire des requêtes et appliquer des algorithmes d'apprentissage (machine learning). Spark utilise uniquement les parties nécessaires d'un jeu de données trop grand pour pouvoir le traiter.

Le RDD, ou Resilient Distributed Dataset est la base de Spark :

- *Dataset : Il s'agit d'un jeu de données qui se parcourt comme une collection.*
- *Distributed : Cette structure est distribuée afin d'être découpée pour être traitée dans les différents nœuds.*
- *Resilient : Il est résilient, car il pourra être relu en cas de problème.*

Le RDD est simple à créer et peut être obtenu à partir de multiples sources :

- *Une collection (List, Set), transformée en RDD*
- *Un fichier local ou distribué (HDFS) dont le format est configurable: texte, SequenceFile Hadoop, JSON*
- *Une base de données*

On peut simplement exporter le contenu d'un RDD dans un fichier, dans une base de données ou dans une collection.

Avantages :

- *Développer une application pour le traitement des données distribuées est beaucoup plus facile avec Spark. En effet, Spark offre une riche interface de programmation d'application " application programming interface (API)" pour développer des applications Big Data.*
- *Spark permet de stocker les données intermédiaires en mémoire. D'autre part, il possède un moteur d'exécution avancé.*
- *Spark intègre un ensemble de librairies pour le traitement par lots, analyse interactive, traitement en streaming, apprentissage automatique et théorie des graphes.*

***Inconvénients** :*

- *peut adapté à la fouille de données*
- *Le problème de la création d'un index pour une collection*

- *Le problème de variable globale inaccessible en cours de travail*

Algorithmes pour implémenter le spark :

- *Classification:NaiveBayes*
- *clustering:LDA*
- *évaluation:BinaryClassificationMetrics*
- *features : Normalizer*
- *fpm : frequent pattern mining : FPGrowth*
- *algèbre linéaire : Vector, BlockMatrix*
- *aléatoire : RandomRDDs*

4. Blockchain:

La blockchain (chaîne de bloques) est une technologie qui permet de stocker et transmettre des données d'une manière sécurisée. Elle présente une grande base de données qui contient les échanges réalisés par les utilisateurs. Les informations contenues dans les blocs (transactions, titres de propriétés, contrats, actions) sont protégées par des procédés cryptographiques qui empêchent les utilisateurs de les modifier.
L'implémentation de la blockchain:

- *Deux personnes s'accordent sur une transaction*
- *Grace à la blockchain, la transaction est encryptée et validée par consensus*
- *Elle est ensuite inscrite puis verrouillée dans le dernier bloc de la blockchain*
- *Enfin la blockchain est répliquée dans tous les nœuds du réseau.*

Exemples de la blockchain : Blockchain Ethereum, Blockchain Bitcoin, Blockchain privée/Blockchain publique, Blockchain de consortium

Avantages :

- *Données infalsifiables : une fois une donnée est inscrite sur une blockchain décentralisée ; elle devient infalsifiable et non modifiable*
- *Avec le smart-contract, on peut programmer un échange sans intermédiaire*

- *Sécurité et rapidité : les transactions sont sécurisées*
- *Décentralisation ce qui peut empêcher la collusion, la censure, la vente de données…*

Inconvénients:

- *Problème de scalabilité*
- *La blockchain ne supprime pas tous les intermédiaires*
- *L'énergie consommé :*Le système de vérification des transactions *consomme plus d'énergie*

II. *SITE DES REPAS SCOLAIRE*

1. *Introduction:*

Ce devoir est une application consistant à mettre à profit toutes les connaissances que nous avons acquises tout au long du cours de base de données non traditionnelles. Il met particulièrement en évidence les systèmes No Sql avec l'utilisation de mongodb et la visualisation des données avec la librairie d3.

*Notre application présente des données ouvertes ("open data") des menus de cuisine centrale de la Communauté d'Agglomération du Pays de Saint-Malo ("**Saint-Malo Agglomération**") fournies sur https://data.stmalo-agglomeration.fr/explore/dataset/menus-de-la-cuisine-centrale-2016/api/?disjunctive.entree_info&disjunctive.plat_info&disjunctive.garniture_info&disjunctive.laitier_info&disjunctive.dessert_info. Nous avons choisis ces données car ce sont données réelles objectives et très riches pour un bon exercice de mongodb.*

Dans la suite de ce rapport nous allons présenté le but de l'application ainsi que ses différentes parties et nous allons finir par une conclusion résumant les compétences acquises .

2. Le but de l'application:

Le but de cette application est de présenter les menus de cuisine, leurs services, leurs constitutions ainsi que les dates aux quelles ils ont été servis en proposant différents types de visualisation des données. Mais aussi de voir le nombre de menus servis par date, le nombre de fois qu'une entrée, un plat principal ou un dessert a été servi. Cela permet par la suite de faire des analyses sur les menus, plats, entrées et desserts les plus consommés mais aussi de voir l'année la plus rentable et les services les plus favorables. Pour plus d'informations, le lien de la vidéo de présentation du devoir est accessible sur https://youtu.be/yS1WrpLHcSk .

3. Les différentes fonctionnalités:

Notre application est composée d'une partie serveur fait en node.js et une partie client constitué de fichiers html et javascript dans les quels nous utilisons la librairie d3. L'importance d'utiliser node comme serveur est que nous pouvons créer notre propre serveur et le manipuler à notre guise. Ainsi dans le serveur nous avons fait la connexion de la base de données mongo et créé des sockets pour la communication instantanée entre le serveur et le client.

4. La liste des commandes (menus):

Cette fonctionnalité nous affiche sur une page, la liste de tous les menus servis avec un système de pagination de 100 menus par page. Le but de cette fonctionnalité est d'avoir une vision d'ensemble sur les menus de la base. Pour ce faire nous avons développé:

- *sur le serveur, une méthode de récupération des données avec le système de pagination*

```
const findCommandes = function(db, col, page, callback) {
  let perPage = 100;
  let p = 1;
  if (page != undefined) {
    p = parseInt(page);

    console.log("Current page in findCommandes", p);
  }
  console.log(perPage, p);
  const collection = db.collection(col);
  let total = 0;
  collection.find().count().then(function(count){
    console.log(count);
    total = count;
  });

  collection.find().project({fields: 1})
  .limit(perPage)
  .skip(perPage * (p-1))
  .sort({ "fields.date": -1 }).toArray(function(err, docs) {
    if (err) {
      assert.equal(err, null);
    }
    console.log("Found the following records");
    //console.log(docs)
    let total_count = total / perPage;
    let results = {
      page: p,
      total: Math.floor(total_count),
      docs: docs
    }
    console.log(results);
    callback(results);
  });
}
```

- *Sur le client, nous utilisons les données renvoyées pour construire la page.*

5. Les Desserts:

Cette fonctionnalité présente tous les desserts de la base et le nombre de fois que chacun d'eux a été servis. Ainsi grâce aux agrégations de mongodb nous avons réussi à récupérer les données nécessaires à la page.

- *Sur la page, chaque ligne montre à gauche le dessert et à droite le nombre de fois qu'il a été servi avec une couleur de fond décrivant sa fréquence.*
- *Au survole de chaque ligne, nous avons fait une petite animation en indexant le dessert et en le colorant en bleu*

```
const findPlats = function(db, col, options, callback) {
  const collection = db.collection(col);
  collection.aggregate(
    [
      {
        $group :
          {
            _id : options,
            count: { $sum: 1}
          }
      }
    ]).toArray(function(err, docs) {
      if (err) {
        assert.equal(err, null);
      }
      console.log("Found the following records");
      //console.log(docs)
      callback(docs);
    });
}
```

Nous avons développé la méthode ainsi que les autres de sorte à les réutiliser avec plusieurs bases de données, collections et les conditions de groupe.

6. Les entrées:

Pour les entrées, nous avons utilisé l'outil pack de d3 pour construire des bulles contenu dans un élément svg avec les données renvoyées par le serveur.

- *Chaque bulle correspond à une itération de la donnée que nous manipulons et est colorée.*
- *Au survol de chaque bulle, le label change par le nom de l'entrée*
- *Le premier clique d'une bulle la zoome tandis que le deuxième la dé-zoome et ramène le graphe à l'état initial.*

7. Les Plats principaux :

Ils sont affichés dans un panel avec le nombre de fois qu'ils sont servis dans des menus.

- *Le survol de chaque ligne la colore en bleu*
- *le clique de chaque ligne affiche une boîte de dialogue avec un tableau contenant le(s) menu(s) dans le(s) quel(s) le plat a été servi.*

8. Le nuage des plats:

Dans cette page, les données sont représentées sous forme de nuage des mots. Pour ce faire nous utilisons une librairie supplémentaire qui s'appelle d3.layout.cloud.js.

- *Les plats sont colorés avec dix(10) couleurs et sont aléatoirement roté de 45 degrés*
- *La taille des mots est en fonction de l'attribut* ***count*** *(le nombre de fois qu' un plat a été servi)*

Cette représentation rend vit aux données et met en évidence les plats les plus consommés.

L'histogramme des dates:

Dans cette page, nous avons créé un histogramme qui montre de façon très simple le nombre de menus consommés par date. Cela nous permet de faire une étude comparative entre les différentes dates pour voire par exemple en quelle année il y'a eu plus de profit.

- *L'axe des abscisse est représenté par les* ***dates*** *et celui des ordonnés par les* ***counts***
- *Les bandes sont animées et colorées du plus bleu foncé au plus bleu claire*
- *les bandes font une translation de bas en haut.*

9. Conclusion:

En résumé, ce devoir nous a non seulement permis de mettre en application toutes les connaissances que nous avons acquises tout au long de ce cours mais aussi de les approfondir avec les notions d'agrégations en mongodb, les graphes avec d3 et la combinaison avec certaines librairies comme bootstrap. En fin nous avons aussi découvert l'avantage des systèmes NoSql par rapport aux autres systèmes de gestions de base de données.