N° d'ordre :
N° de série:

RÉPUBLIQUE ALGÉRIENNE DÉMOCRATIQUE ET POPULAIRE
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique

---

UNIVERSITE ECHAHID HAMMA LAKHDAR - EL OUED
FACULTÉ DES SCIENCES EXACTES
Département D'Informatique

Mémoire de Fin D'étude
Présenté pour l'obtention du Diplôme de

MASTER ACADEMIQUE

Domaine : **Mathématique et Informatique**
Filière : **Informatique**
Spécialité : **Systèmes Distribués et Intelligence Artificielle**

Présenté par :
- **Hamidatou Hana**
- **Djouadi Soumaia**

Thème

Une méthode de classification dans un environnement BIG DATA sous Hadoop

Soutenue le 16-06- 2022 Devant le jury:

M.	Naoui Mohammed Anour	MCA	Président
M.	Yagoub Mohammed Amine	MCB	Rapporteur
M.	Retima Farida	MCB	Encadreur

Année Universitaire: 2021/2022

Dédicace

A'exemple de courage et de patience... mon cher père.

A'Celle qui a donné mon bonheur et mon réconfort sur son bonheur... mon paradis qui était absent... l'âme pure de ma mère.

A'mon charmant mari.

qui m'a donné espoir et patience... mon amour et mon coeur

A' Ceux qui m'aident et me soutiennent au quotidien... mes sœurs, frères et leurs épouses, ainsi qu'à mes deuxièmes père et mère

A'toute la famille et amis

Tout le monde me souhaite bonne chance

Je vous dédie ce travail...

DJOUADI SOUMAIA

Dédicace

Au symbole de l'amour et de la d´evotion à ma mére bien-aim´ee

A' celui qui m'a donn´e la bienveillance, le sacrifice et l'altruisme, mon cher père.

A' ceux qui m'aident et me soutiennent chaque jour ... mon cher mari.

A' celui qui m'a donn´e espoir et patience ... mon amour et mon Coeur…iline

A'toute la famille et amis

tous ceux qui m'aiment sinc`erement et cordialement

Je vous d´edie ce travail

HAMIDATOU HANA

Remerciement

Tout d'abord nous remercions notre Dieu qui nous a donné la force et la volonté pour élaborer ce travail.

Nous adressons nos sincères remerciements à notre encadrant

Dr.« Retima Farida »

Qui était chargé de fournir des conseils et des orientations

Pour chaque petit et chaque grand dans ce mémoire

Nous adressons nos sincères remerciements à tous les enseignants du département d'informatique

Université Echahid Hamma Lakhdar

El-Oued.

Aussi à nos collègues de la promotion 2021-2022. On remercie également tous ceux qui ont participé de près ou de loin à élaborer ce travail.

Soumaia - Hana

Résumé

En conséquence de l'explosion quantitative des données numériques au des dernières années, Le traitement intensif de ces données occupe une place significative dans notre vie quotidienne.

Ce projet de fin d'étude repose, principalement, sur l'étude d'une méthode de classification des données (selon des critères de classification) dans un environnement de Big Data. Nous intéresserons particulièrement aux technologies Hadoop avec ses composantes HDFS et MapReduce afin de réaliser une classification parallèles et distribués sur de gros volumes de données pour améliorer les performances de traitement dans un tel environnement.

Mots clés :

Big Data, Hadoop, Système Distribué, MapReduce, classification.

ملخص

نتيجة للانفجار الكمي للبيانات الرقمية في السنوات الأخيرة ، تحتل المعالجة المكثفة لهذه البيانات مكانًا مهمًا في حياتنا اليومية.

يعتمد مشروع نهاية الدراسة هذا بشكل أساسي على دراسة طريقة تصنيف البيانات (وفقًا لمعايير التصنيف) في بيئة البيانات الضخمة. سنكون مهتمين بشكل خاص بتقنيات Hadoop بمكونات HDFS و MapReduce من أجل إجراء تصنيف متوازي وموزع على كميات كبيرة من البيانات لتحسين أداء المعالجة في مثل هذه البيئة.

كلمات مفتاحية :

البيانات الضخمة ، Hadoop، النظام الموزع ، MapReduce، التصنيف.

Abstrait

As a result of the quantitative explosion of digital data in recent years, the intensive processing of these data occupies a significant place in our daily lives.

This end-of-study project is mainly based on the study of a data classification method (according to classification criteria) in a Big Data environment. We will be particularly interested in Hadoop technologies with its HDFS and MapReduce components in order to perform parallel and distributed classification on large volumes of data to improve processing performance in such an environment.

Keywords :

Big Data, Hadoop, Distributed System, MapReduce, classification.

TABLE DES MATIÈRES

Chapitre 4 : Implémentation et Analyses

TABLE DES FIGURE

Page

Chapitre 1 : Big Data

Chapitre 2 : Hadoop

Chapitre 3 : Conception

Chapitre 4 : Implémentation et Analyses

Liste des Algorithmes

Liste des tableaux

INTRODUCTION GÉNÉRALE

INTRODUCTION GÉNÉRALE

Ces dernières années ont été marqué par une explosion quantitatives croissante des données numériques, cela a créé un nouveau type de données dites les méga donnée ; un terme générique pour désigner un vaste ensemble de données. Ce phénome peut apporter plusieurs avantages dans tous les domaines, à savoir : l'industrie, la commerce, le marketing, etc. comme il peut générer d'autres problèmes qui concerne, particulièrement, le stockage et le traitement de ces données.

Par conséquence, la croissance des volumes de données traitées dépasse largement les limites des solutions traditionnelles à savoir les entrepôts de données et leurs modèles de traitements. Au fait, bien que les capacités de stockage des disques durs aient augmenté massivement au fil des dernières années, la vitesse d'accès aux données ne l'est pas encore. En plus, aucune solution classique ne permet de traiter rapidement et efficacement les grands volumes de données.

Notre projet de fin d'étude se repose, principalement, sur l'étude des méthodes et des technologies du Big Data. Nous nous intéresserons particulièrement aux technologies Hadoop avec ses composantes HDFS et MapReduce et l'application de la méthode de classification k-means sur ces deux derniers

Structure du mémoire

Le contenu de ce mémoire est organisé comme suit:

Chapitre 1 : présentation le concept du big data avec ses définitions, ses caractéristique, ses fonctionnements, ses technologies et les domaines dans lesquels on l'utilise.

Chapitre 2 :présentation de Framwork Hadoop, les caractéristique de ses principaux composants HDFS et MapReduce.

Chapitre 3 :présentation de la méthode de classification k-means et nous avons expliqué en détail ce que nous allons faire.

Chapitre 4 : nous présentons les différentes étapes de l'implémentation et la réalisation de notre travail

Chapitre : 1

Big DATA

1. Introduction :

Actuellement le monde est conforté à une augmentation incroyable du volume des données avec ces divers types structurées et non structurées, provenant de sources internes et externes diverses et multiples. Le problème qui se pose maintenant est que la croissance des données dépasse la puissance des SGBD traditionnelle, à cause du volume des données, la diversité des données, et le temps du traitement qui ne doit pas dépasser la durée acceptable par l'utilisateur. Par conséquent, les développeurs ont le défi de développer des technologies qui permet de traiter d'énormes quantités de données en peu de temps, ce qu'on appelle le BigData.

Dans cette partie, nous étudions la signification de Big Data et son importance, les domaines d'utilisation, ses outils et quelques technologies du Big Data.

2. Big Data:

2.1. Émergence de Big data:

Le Big data est considéré comme le dernier cri en matière de High Tech, il a résulté de la révolution de la technologie de l'information. Il a certainement une grande influence sur le présent et la future à cause de son importance dans le domaine de la communication et de traitement de données et la recherche scientifique.

La quantité d'information traitée par le micro-processeur se voit doublé chaque 18 mois environ selon la loi de Moore.

Il convient également de signaler que la capacité de stockage des données sur le disque dur évolue plus rapidement que la capacité de traitement de données sur le micro-processeur (loi de Kryder).

Le volume de données stocké est en croissance exponentielle. La capture, l'intégration et l'exposition de l'information était assez difficile. Les données non structurées prennent une importance considérable, cela est dû à l'évolution des réseaux sociaux et à leurs exigences.

Exemple : chaque appel téléphonique exige une masse de données ayant rapport à le lieu de l'abonné, son numéro, le temps de communication et le tarif unitaire. Un opérateur de télécommunications typique générera quelques téraoctets de données détaillées d'appels chaque mois. Nous constatons également une amélioration considérable dans le traitement du son (flac), de la vidéo (mkv) qui exigent plus de mémoire de stockage et de vitesse de traitement.

Cette masse énorme de données doit être stockée, et le cas échéant analysée et exploitée correctement pour répondre aux besoins des différents opérateurs industriels, économiques ou sociaux. En bref c'est l'ère de l'information.

Le Big data est une technologie révolutionnaire qui peut provoquer des perturbations aux niveaux économique, scientifique et culturelle. Cela est dû à l'importance des changements et des améliorations qu'il impose dans ces différents domaines et qui exigent une nouvelle réadaptation [2].

2.2. Définition :

Le terme de Big Data (parfois appelées « données massives » désigne une nouvelle discipline qui se situe au croisement de plusieurs domaines : statistiques, technologie, base de données et métiers (marketing, finance, etc.).

Le Big data a pour objectif d'exploiter des volumes de données qui sont en croissance exponentielle et qui deviennent difficiles à travailler avec des outils classiques de gestion de base de données ou de la gestion de l'information. Il a aussi pour objectif de traiter rapidement des données complexes [5].

Étant un objet complexe polymorphe, sa définition varie selon les communautés qui s'intéressent en tant qu'utilisateur ou fournisseur de services.

Les données du Big Data proviennent de sources diverses, et peuvent donc prendre plusieurs formes. On distingue plusieurs catégories principales. Lorsque les données pouvant être stockées et traitées dans un format fixe et bien défini, on parle alors de données " structurées ". Grâce aux nombreuses avancées réalisées dans le domaine de l'informatique, des techniques permettent aujourd'hui de travailler efficacement avec ces données.

Cependant, même les données structurées peuvent poser problème à cause de leur volume massif. Alors que le volume d'un ensemble atteint désormais plusieurs zettabytes, le stockage et le traitement représentent de véritables défis.

Les données dont le format ou la structure sont inconnus, quant à elles, sont considérées comme des données " non structurées ". Ce type de données présente de nombreux défis en termes de traitement et d'exploitation, au-delà de leur volume massif.

Enfin, les données " semi-structurées " sont à mi-chemin entre ces deux catégories. Il peut s'agir par exemple de données structurées en termes de format, mais n'étant pas clairement définies au sein d'une base de données [9].

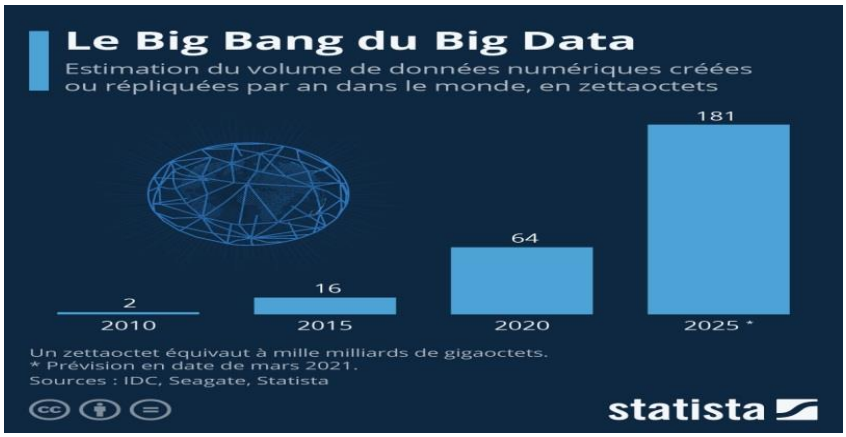


Figure 1.1: Le Big Bang du Big data

3. Caractéristiques du Big data :

Le big data, est un ensemble de données numériques produites par l'utilisation de nouvelles technologies qui sont caractérisées par les 5V et devient difficile à traiter et à analyser en utilisant les outils classiques de gestion de base de données .

Les 5V représentent ´: le volume, la vélocité , la variété la véracité et la valeur des données [8]. Et on peut considérer aussi la caractéristique de variabilité.

Figure 1.2: Les Caractéristiques du Big data

a) Volume : il représente la quantité massive de données générées.

b) Vélocité : signifie la rapidité de la collection et de l'analyse des données.

c) Variété ´ : le type et le format de données générés sont différentes.

d) Véracité : correspond à la fiabilité de données. La précision et la qualité des données sont très importantes. Donc, il faut s'assurer que les données du big data sont vraies pour pouvoir prendre des décisions.

e) Valeur : est le V le plus important. l'objectif est de créer des valeurs pour les entreprises et les clients en transformant toutes les données en valeurs exploitables.

f) Variabilité : faisant référence aux différentes manières dont le Big Data peut être utilisé et formaté.

4. Gestion de Big Data:

Gestion de Big Data ou bien Big Data Management est une nouvelle discipline dans laquelle les techniques, les outils et les plates-formes de gestion de données y compris le stockage, le prétraitement, le traitement et la sécurité peuvent être appliqués. [3]

Le rôle de la gestion des données est assure un haut niveau de qualité des données et aide les entreprises à faire face à la quantité des données qui grandit.

a) Stockage des données :

Stocker les données on pétaoctets de façon distribuée utilise les services de Cloud. Lestockage consiste trois opérations principales (Clusting, Réplication, indexing).

b) Prétraitement :

Avant l'analyse de Big Data, on a besoin de vérifier la qualité des données et réparer les données au traitement par l'application des étapes suivantes (nettoyage des données, transformation, intégration, transmission, réduction).

c) Traitement :

C'est l'aptitude de traiter un grand volume de données quel que soit le type ou bien la structure et l'emplacement de ces données. Ce traitement peut être classification ou prédiction.

d) Sécurité:

Pour sécuriser un grand volume de données, plusieurs algorithmes de sécurité sont apparus pour la confidentialité, intégrité, et la disponibilité.

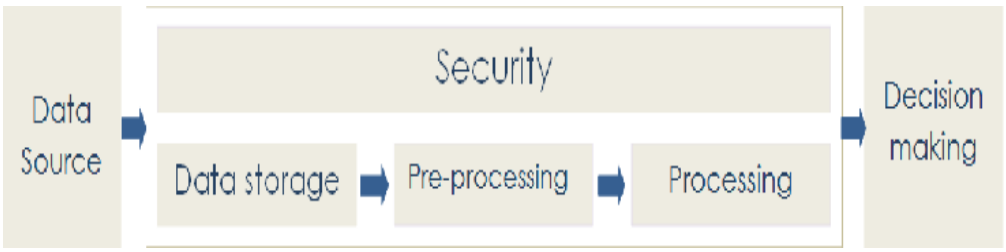


Figure 1.3 : Les étapes de gestion de Big Data.

5. Domaines d'applications du Big Data:

Les grands domaines du big data sont [10]:

5.1. Le Big Data au service des gouvernements :

Le Big Data est utilisé depuis sa naissance par les pouvoirs publics (Les enquêtes criminelles, élections…etc.)

Ainsi, Le Big Data est utilisée pour le service de la finance, de la banque, et de l'assurance. Il est estimé que le gain tiré de l'utilisation du Big Data par les banques et les institutions financières représenterait 10 % de revenus annuels supplémentaires.

5.2. Dans le domaine de l'assurance :

Le Big Data a déjà pris sa place. Grâce à toutes les données qu'elles possèdent aujourd'hui, les compagnies d'assurance peuvent désormais mieux percevoir le risque de chacun de leurs clients et donc adapter leurs clients avec leurs offres.

5.3. Dans le domaine de finance :

Grâce au Big Data, il permet de développer le « trading à haute fréquence » permettant de passer plusieurs centaines d'ordres d'achat et/ou de vente en quelques secondes, voire en millisecondes, tout en optimisant la prise en compte des risques. Cela permet d'acheter pour revendre dans un laps de temps suffisamment court pour éviter une évolution potentiellement négative du marché durant l'opération. On peut donc affirmer que le Big Data a permis de révolutionner le secteur financier.

5.4. Dans le domaine des banques :

Les banques traditionnelles sont aussi confrontées au Big Data. Le développement des services en ligne leur offre une meilleure connaissance de leurs clients, ce qui induit un changement dans la relation de la banque avec ses clients.

Par ailleurs, le Big Data permet aux banques de lutter contre la fraude. Elles sont désormais capables de surveiller l'intégralité des transactions par carte bancaire et d'être alertées lorsqu'un utilisateur réalise un paiement inhabituel (notamment en termes de montant).

5.5. Le big data touche tous les domaines

Le Big Data couvre de nombreux domaines d'applications telles que l'industrie, les sciences, la publicité, le marketing, le commerce, le transport, l'éducation, télécom, etc. (comme illustrées par la (Figure 1.4)

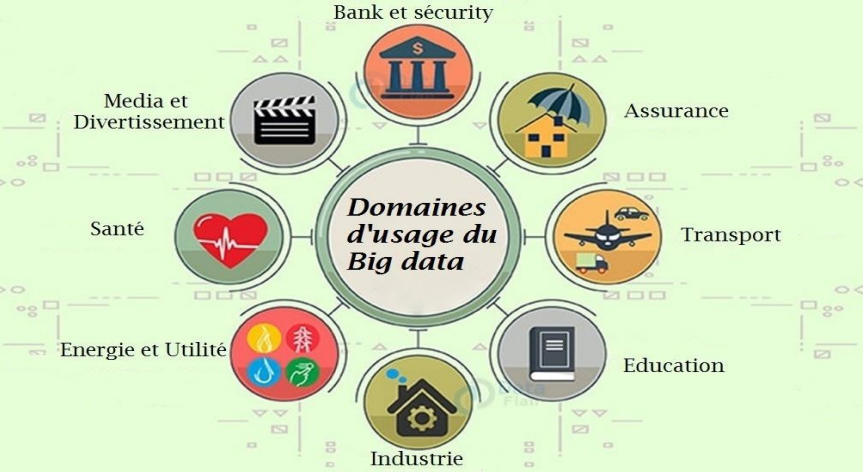


Figure 1.4 : Les domaines de Big Data.

6. Fonctionnement du Big Data :

Le Big Data offre de nouvelles perspectives, qui ouvrent de nouvelles opportunités et favorisent de nouveaux Business Model. Son adoption implique trois actions principales :

- **Intégrer:**

Le Big Data rassemble des données provenant de sources et d'applications disparates. Les mécanismes d'intégration des données classiques, comme ETL (extraire, transformer et charger) ne sont généralement pas à la hauteur. Pour analyser un grand énorme de données à l'échelle de téraoctets, voire de pétaoctets, il est nécessaire d'adopter de nouvelles stratégies et technologies.

Lors de la phase d'intégration, il faut importer les données, les traiter et assurer qu'elles sont formatées et disponibles sous une forme que les analystes peuvent les exploiter.

- **Gérer:**

Le Big Data nécessite du stockage. La solution de stockage peut se trouver dans le cloud, sur site, ou les deux à la fois. Vous pouvez stocker vos données sous la forme de votre choix et imposer à ces jeux de données vos exigences de traitement, ainsi que les moteurs de traitement nécessaires. Nombreux sont ceux qui choisissent leur solution de stockage en fonction de l'endroit où sont hébergées leurs données. Le cloud est de plus en plus adopté, car il prend en charge vos besoins informatiques actuels et laisse la possibilité d'augmenter les ressources en fonction des besoins.

- **Analyser:**

Une fois que les données massives ont été correctement stockées dans un lieu sécurisé et facilement accessible, il faut les analyser. Cette étape vient avant leur utilisation. L'exploration des données permet de pouvoir les utiliser afin de prendre des décisions qui s'imposent.[12]

7. Les Technologies de Big Data:

Nous avons cité quelques technologies utilisent le Big Data:

- **HADOOP :** est un framework mis au point par l'Apache Software Foundation afin de mieux généraliser l'usage du stockage et traitement massivement parallèle et distribuée de MapReduce et de Google File System. C'est une solution de Big Data très largement utilisée pour effectuer des analyses sur de très grands nombres de données.

- **Bases NoSQL:** Les bases de données relationnelles ont une philosophie d'organisation des données bien spécifiques, avec notamment le langage d'interrogation SQL, le principe d'intégrité des transactions (ACID), et les lois de normalisation. Elles sont utiles pour gérer les données qualifiées de l'entreprise, mais elles ne sont pas du tout adaptées au stockage de très grandes dimensions et au traitement ultra rapide. Les bases NoSQL implémentent des systèmes de stockage considérés comme plus performants que le traditionnel SQL pour l'analyse de données en masse .[11]

- **Stockage "In-Memory" :** Pour des analyses encore plus rapide, les traitements directement en mémoire sont une solution. Est une technologie bien qu'encore trop coûteuse mais il est vrai pour être généralisée .[13]

8. Avantages et inconvénients de l'utilisation du Big Data : [14]

L'utilisation du Big Data présente les avantages suivants :

-La réduction des coûts ;

-La création de produits et services améliorés ou nouveaux pour répondre aux différents besoins des clients ;

-La possibilité d'avoir des retours en temps réel ;

-Une meilleure connaissance du marché

Partant, on peut être sûr que l'utilisation du Big Data peut être très bénéfique pour les organisations. Cependant, le Big Data est aussi à l'origine de certaines des grandes problématiques de ce début de siècle, principalement la confidentialité et le respect de la vie privée.

Ainsi, un certain nombre d'inconvénients associés au Big Data sont à considérer :

-La confidentialité des données.

-La sécurité des données stockées mise à mal par les risques d'espionnage numérique.

-La manipulation des données.

-Les données personnelles doivent être collectées dans un but bien précis, explicite et légitime selon la loi « informatique et libertés » de 1995.

9. Architecture du Big Data:

Il existe deux principaux types d'architectures Big data :

L'architecture Lambda a été imaginée afin de faire simultanément du traitement de type batch (traitement par block de données) et du traitement en temps réel (de manière continu). L'architecture Lambda permet de fusionner le traitement par bloc de données (batch) et les nouvelles données entrées (temps réel). Elle offre des vues complètes sur l'ensemble des données.

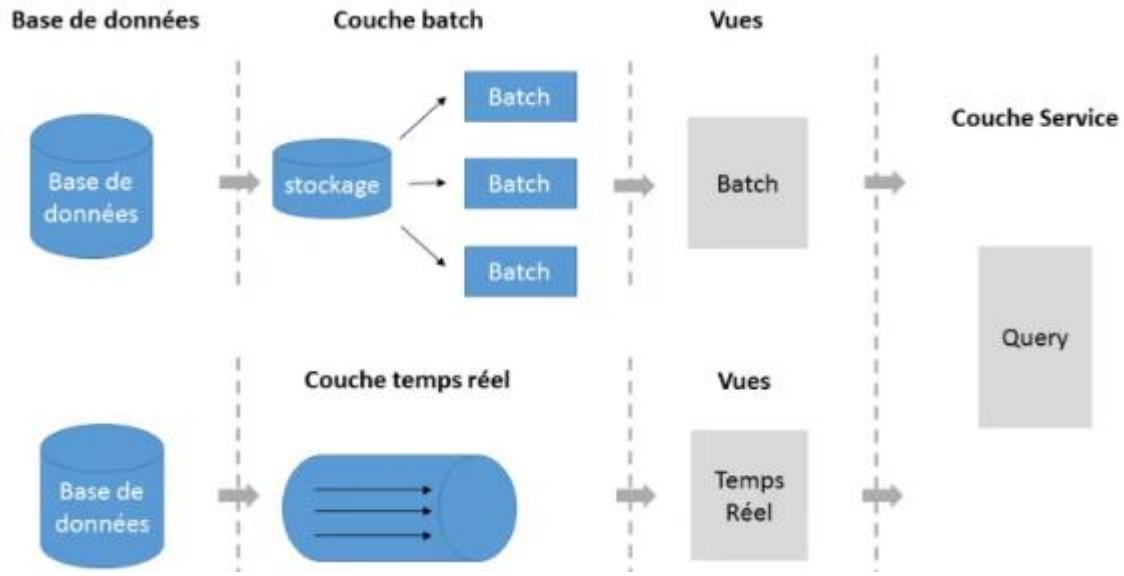


Figure 1.5: Architecture Lambda**.**

L'architecture KAPPA a été pensée pour pallier la complexité de l'architecture Lambda. Elle repose sur le principe de fusion de la couche temps réel et batch, ce qui la rend moins complexe que l'architecture Lambda. Ne permettant pas le stockage de manière permanente. Cette architecture est faite pour le traitement des données.

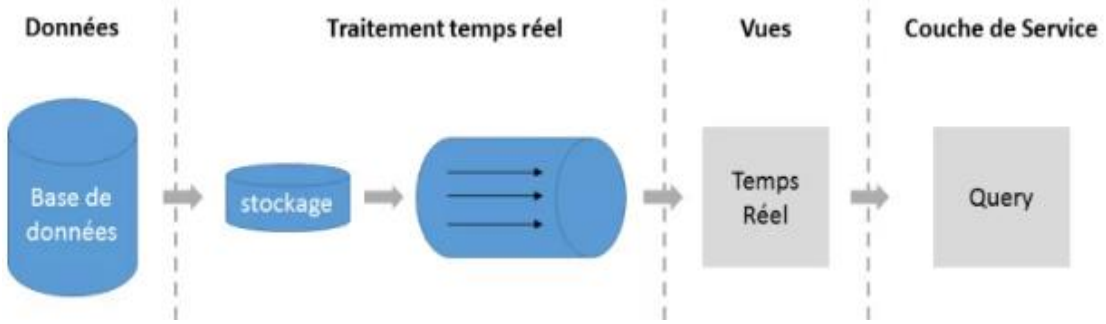


Figure 1.6: Architecture Kappa**.**

Chacune de ces architectures permet de répondre à un besoin spécifique. Le choix du modèle architectural le plus adapté dépend de besoins, les infrastructures existantes, les objectifs, et de contexte.

L'architecture de Big data est composé de 4 grandes parties : intégration, data processing et stockage, sécurité et opération comme le montre le schéma si dessous [6] :

Figure 1.7: Architecture du Big data.

- ✓ **Intégration :** consiste à charger le volume de données au sein du stockage.
- ✓ **Stockage de données (Data Storage) :** le stockage des données massives et volumineuses dans lequel il faut faire très attention à réduire les coûts de stockage au sein de l'entreprise.
- ✓ **Manipulation de données (Data Processing) :** il s'agit de la manipulation et du traitement de données (comme le Map Reduce).
- ✓ **Sécurité :** sert à l'autorisation, l'authentification et la protection des données.
- ✓ **Opérations :** pour la gestion, le monitoring et les tâches planifiées.

10. Les enjeux du Big data :

Les grands enjeux du donnés massives sont : [15]

1. Garantir la qualité des informations : la qualité des données doit être une priorité pour ne pas fausser les stratégies découlant de leur traitement.

2. Optimiser le traitement des données : pour de nombreux experts, le traitement des données est l'un des enjeux les plus importants du Big Data. En effet, les informations arrivent en masse et se présentent sous divers formats.

3. Mettre en relation tous les métiers : le Big Data implique de faire travailler ensemble avec de différents professionnels dans le but d'atteindre des objectifs précis.

4. Assurer la sécurité : la sécurité a une importance particulière, car elle engage la responsabilité et la réputation de l'information.

11. Conclusion :

Le Big Data est en train de devenir une nouvelle technologie dans tous les domaines, surtout avec le besoin d'apporter des mécanismes avancés d'acquisition, de gestion et d'analyse des données volumineuses.

Dans ce chapitre nous avons survolés les principaux concepts de Big Data. Nous avons commencé par présenter l'émergence de big data et sa définition. Ensuite nous avons expliqué ses caractéristiques, son fonctionnement ainsi que les différents domaines dans lesquels elles sont utilisées. Dans le prochain chapitre nous présenterons en détail la plateforme Hadoop comme une solution efficace pour gérer et traiter des données volumineuse.

Chapitre : 2

HADOOP

1. Introduction :

Dans l'approche traditionnelle, les entreprises stockent et traitent le Big Data sur des ordinateurs, c'est-à-dire que les développeurs utilisent les bases de données et leurs ordinateurs personnels pour la manipulation des données. Cette approche donne des résultats effectifs avec des petites quantités des données où le nombre des traitements est limité. Mais si on a des données volumineuses, il est difficile de traiter ces données dans un temps acceptable. Google donne la solution pour résoudre ce problème avec l'algorithme MapReduce qui divise une tâche aux plusieurs parties et affecte chaque partie à une machine, puis collecte les résultats de chaque tâche pour donner la solution global grâce à un système de gestion des fichiers distribués s'appelle HDFS. À partir de la solution proposée les développeurs de Google invitent un projet open source appelé Hadoop.

Dans ce chapitre, nous aborderons quelques généralités sur Hadoop, tout en commençant par une définition de tel plate-forme, ensuite son architecture et finalement son fonctionnement.

2. Qu'est-ce qu'Hadoop ?

Hadoop est une plate-forme qui fournit à la fois des capacités de stockage et de calcul de manière distribué et parallèle. Hadoop a d'abord été conçu pour résoudre un problème d'évolutivité qui existait dans le projet Apache Nutch (visant à construire un moteur de recherche open source). À l'époque, Google avait publié des articles qui décrivent son nouveau système de fichiers distribué (le Google File System (GFS)), et Map- Reduce, (cadre de calcul pour le traitement parallèle).

Hadoop peut être considéré comme un système de traitement de données évolutif permettant de stocker et de traiter par lots de très grandes quantités de données, c'est-à-dire des mégadonnées. Il est idéal pour le stockage à grande échelle et l'analyse de type ad-hoc de grandes quantités de données.

Le principe repose sur l'exécution du traitement répartie multi noeuds pour augmenter les capacités de calculs et de stockage afin de traiter de très grandes quantités de données comme le montre la figure 2.1 [1].

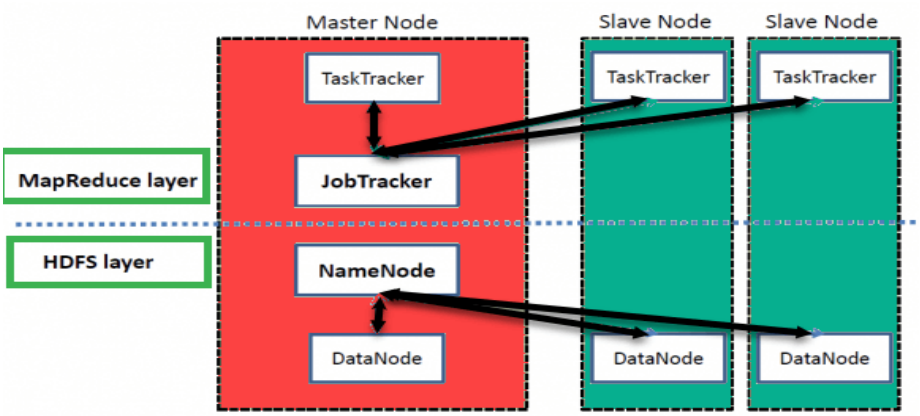


Figure 2.1 : High-level Hadoop architecture

3. Historique:

Doug Cutting a créé Hadoop. En 2002 doug cutting et mike caferella de google travaillent sur projet apache nutch, ils ont des difficultés de stockages de données et des couts élevés.

En 2003, google publier GFS(google file system). En 2004, la firme américaine publier un livre blanc sur Mapreduce. Les deux dernières publications ont fortement influencé la genèse de hadoop.

En 2005, doug cutting quitte google pour rejoindre yahoo. En 2006, il décidé de rejoindre yahoo avec le projet Nutch et les idées basées sur les premiers travaux de google (traitement et stockage des données distribuées). En 2008 Hadoop est proposé par yahoo comme un projet open source.

L'idée de logo de hadoop inspiré par le jouet préféré de son premier fils (un éléphant jaune), l'idée sera confortée par son ami. En 2008, il devient le système le plus rapide à trier un terabyte de données sur un cluster de 900 noeuds en seulement 209 secondes.

La version 2.2 est lancée en 2013. La version 3.0 est lancée en 2017. [16]

4. Les composants d' Hadoop :

Le noyau d'Hadoop est constitué d'une partie de stockage : HDFS (Hadoop Distributed File System), et d'une partie de traitement appelée MapReduce. Hadoop fractionne les fichiers en gros blocs et les distribue à travers les nœuds du cluster. Pour traiter les données, il transfère le code à chaque nœud et chaque nœud traite les données dont il dispose. Cela permet de traiter l'ensemble

des données plus rapidement et plus efficacement que dans une architecture supercalculateur plus classique qui repose sur un système de fichiers parallèle où les calculs et les données sont distribués via les réseaux à grande vitesse.

Le Framework Hadoop de base se compose des modules suivants **:**

- **Hadoop Common .**
- **Hadoop YARN .**
- **Hadoop Distributed File System (HDFS), système de fichiers.**
- **Hadoop MapReduce .**

4.1. Hadoop Common : Il utilise des bibliothèques Java standards entre chaque module.

4.2. Le YARN (Yet Another Resource Negotiator): Comme son nom l'indique, il s'agit d'un négociateur de ressources. Il permet de planifier des tâches, de gérer les ressources et de surveiller les noeuds de clusters et les autres ressources [1].

4.3. Le Hadoop Distributed File System (HDFS): est utilisé pour le stockage de données. Il est comparable à un système de fichier local sur un ordinateur classique.

Toutefois, ses performances sont nettement supérieures. Le HDFS délivre par ailleurs une excellente élasticité. Il est possible de passer d'une machine unique à plusieurs milliers d'entre elles très facilement [17].

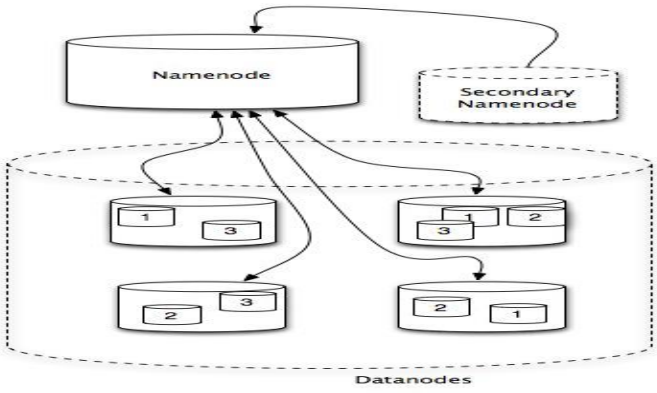


Figure 2.2: Hadoop Distributed File System

4.3.1. Architecture de machines HDFS :

- **NameNode :**

Il Gère l'espace de noms, l'arborescence du système de fichiers et les métadonnées des fichiers et des répertoires.

- **SecondaryNameNode :**

Il gère l'historique des modifications dans le système de fichiers, et permet la continuité du fonctionnement du cluster en cas de panne du NameNode principal

- **DataNode :**

Il permet de stocker et restituer les blocs de données [4].

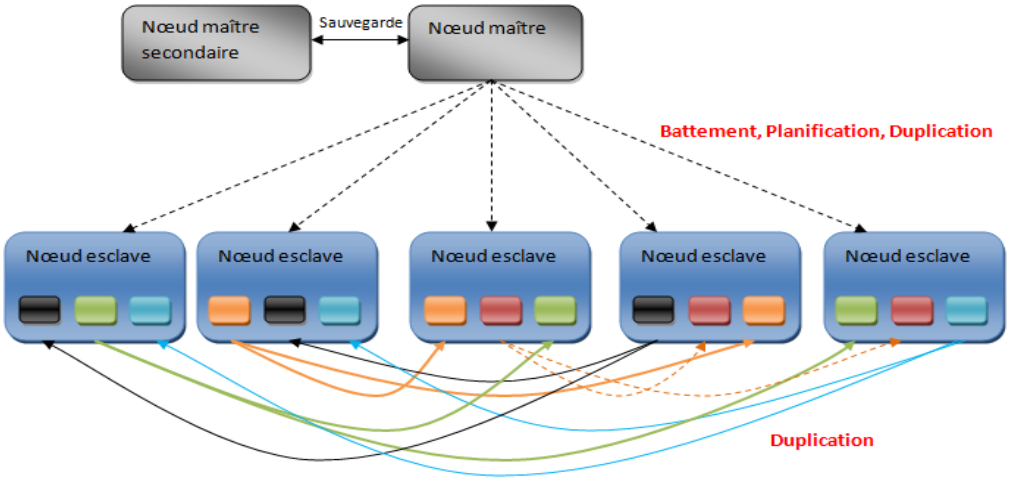


Figure 2.3: Architecture De Machines HDFS

4.3.2. Les avantages de HDFS :

HDFS présente plusieurs avantages distincts :

- Ce système de fichiers est réparti sur des centaines ou des milliers de serveurs, chaque nœud stockant une partie du système de fichiers. Pour éviter le risque de perdre des données, chaque donnée est stockée dans trois emplacements. Il est également très efficace pour le traitement des flux de données.

-Pour les grands ensembles de données, de l'ordre de plusieurs gigabytes ou terabytes, ce système de fichier distribué fonctionne également très bien. Il est donc très adapté au Big Data. Des

dizaines de millions de fichiers peuvent être supportés sur une seule instance. Par ailleurs, cela assure la cohérence et l'intégrité des données pour éviter tout désagrément.

-HDFS évite également la congestion de réseau en privilégiant le déplacement des opérations plutôt que le déplacement des données. Ainsi, les applications peuvent accéder aux données à l'endroit où elles sont entreposées. Dernier point fort : sa portabilité. Il peut fonctionner sur différents types de hardware sans aucun problème de compatibilité.

-Plus qu'une base de données, ce système de fichier distribué se présente comme une Data Warehouse. Il est impossible de déployer un langage de requête sur HDFS, et les données sont accessibles par le biais de fonction de mapping et de réduction (Hadoop MapReduce). Les données adhèrent à un modèle de cohérence simple et robuste.

4.4. Le Module Hadoop MapReduce : MapReduce ou Hadoop MapReduce est un modèle de programmation qui sert à calculer de gros volumes de données en parallélisant les calculs sur différents nœuds d'un cluster. On parle de calculs distribués.

Plus concrètement, MapReduce consiste, de base, en deux fonctions :

Map() pour distribuer le travail sur les nœuds du cluster ;

Reduce() pour agréger le résultat de chaque nœud en un unique résultat [18].

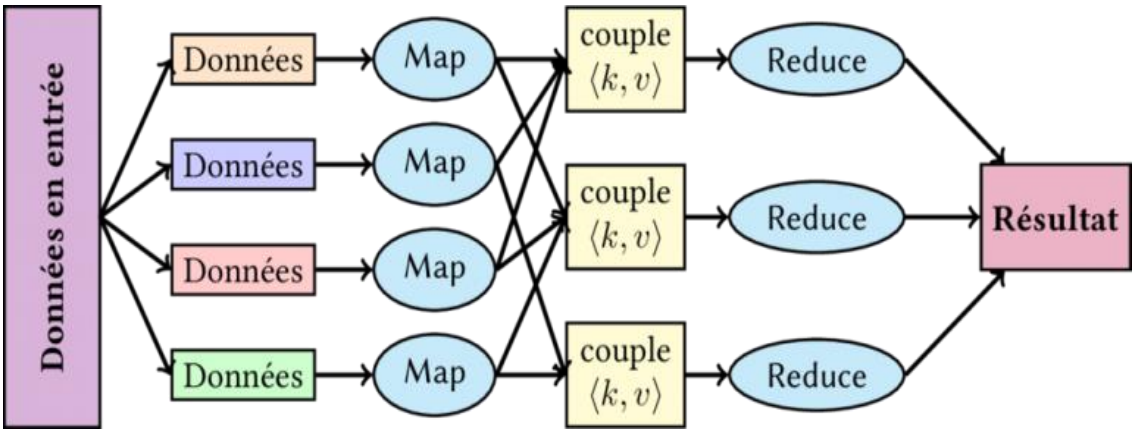


Figure 2.4: Schéma simplifié du fonctionnement de MapReduce

Ce Framework fonctionne exclusivement sur le concept de (clé, valeur).

4.4.1. La phase Map : [19]

Dans cette phase, le fichier de données à traiter a déjà été partitionné dans le HDFS ou le système de fichiers distribué du cluster. A chaque partition des données est affectée une tâche map. Ces tâches Map se sont en réalité des fonctions qui transforment la partition à laquelle chacune est assignée en paires de clé/valeurs. La façon dont les données d'entrée sont transformées en clé/valeurs est à la discrétion de l'utilisateur. Attention, pour ceux qui travaillent dans le développement des bases de données, le terme « clé » peut générer de la confusion. Les clés générées ici ne sont pas les « clés » dans le sens de « clé primaire » des bases de données relationnelles, elles ne sont pas uniques, ce sont juste des nombres, des identifiants arbitraires qui sont affectés aux valeurs de la paire.

La spécificité cependant est qu'à toutes les valeurs identiques de la partition sont affecté la même clé. Dans la figure 2.5, prenons l'illustration du comptage de mots dans une pile de 3 documents. Chaque document dans la pile est une partition stockée dans un nœud du cluster. La fonction map est définie de la façon suivante : M(k,v) = List (k,v) avec k la clé, qui est ici le mot contenu dans le document, et v, la valeur, qui représente ici la référence du document dans lequel est situé la clé. La figure suivante illustre l'exécution du traitement Map.

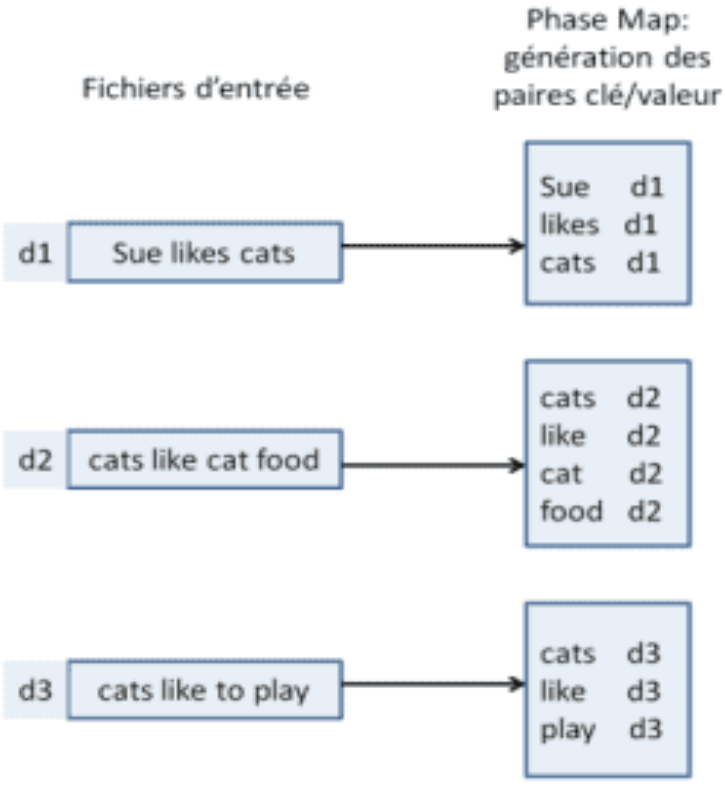


Figure 2.5: Exécution De La Phase Map()

4.4.2. La Phase Shuffle :

Une fois que toutes les tâches Map sont achevées, la phase Shuffle démarre. Cette phase consiste d'une part à trier par clé, toutes les paires clé/valeurs générées par la phase Map, et d'autre part à regrouper dans une liste pour chaque clé, l'ensemble de ses valeurs éparpillées à travers les nœuds auxquels a été assignée la fonction Map.

Formellement, les paires de clés/valeurs à cette phase ont la forme suivante : (k,[v1,v2,v3,……..,vn]) où (k,v1),(k,v2),(k,v3),…,(k,vn) sont les paires de clé/valeurs, avec k la clé venant de toutes les tâches Map. La figure ci-après illustre la phase Shuffle **[19].**

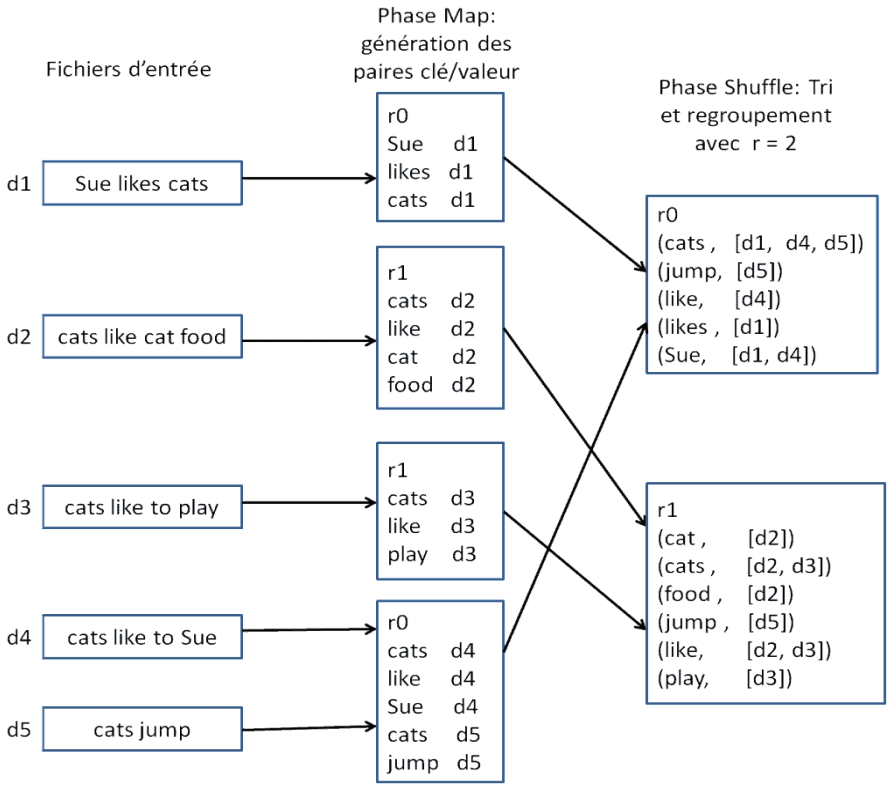


Figure 2.6 : Exécution de la phase Shuffle.

4.4.3. La Phase Reduce :

La phase Shuffle s'achève avec la construction des fichiers contenant les listes de clés/valeurs qui vont servir d'arguments à la fonction Reduce. Le but de cette phase est d'agréger les valeurs des clés reçues par le Shuffle et de joindre verticalement l'ensemble des fichiers pour obtenir le résultat final. L'utilisateur définit dans la fonction Reduce l'agrégat qu'il veut utiliser, par exemple la somme, le comptage,…etc., et ce qu'il souhaite faire des résultats : soit les

afficher à l'aide d'une instruction « print », soit les charger dans une base de données ou soit les envoyer à un autre job MapReduce.

Prenons l'exemple précédent et supposons qu'on souhaite à l'aide du Reduce compter le nombre d'occurrence de chaque mot dans l'ensemble de la pile des 3 documents. La figure ci-après illustre le résultat. **[19]**

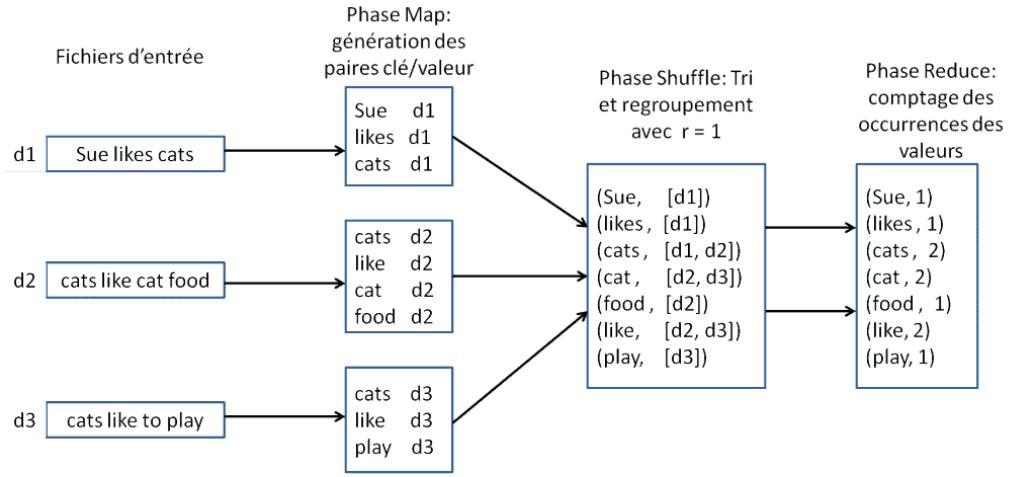


Figure 2.7: Exécution de la phase reduce.

- Exécution de MapReduce dans un cluster hadoop

Le traitement MapReduce écrit par l'utilisateur s'appelle un job MapReduce dans la terminologie du cluster Hadoop et s'exécute en 7 étapes :

1. Au départ, l'utilisateur configure le Job MapReduce : il écrit la fonction Map, la fonction Reduce, spécifie le nombre r de tâches Reduce, le format de lecture du fichier d'entrée, le format de sortie des r fichiers Reduce, éventuellement la taille des blocs du fichier d'entrée et le facteur de réplication [1]. Une fois que tout cela est fait et qu'il déclenche l'exécution du job, l'orchestrateur de ressources (YARN ou Kubernetes) démarre les r tasktrackers dans des containers au niveau des noeuds qui vont effectuées les r tâches Reduce que l'utilisateur a spécifié.

2. Le HDFS partitionne le fichier d'entrée en blocs de taille fixe, généralement 64 Mo par bloc (sauf si l'utilisateur a spécifié une taille de bloc différente à la première étape). Ensuite, le HDFS réplique ces blocs selon le facteur de réplication définie par l'utilisateur (3 par défaut) et les distribue de façon redondante dans des nœuds différents dans le cluster. Le fait de partitionner le fichier d'entrée en blocs de taille fixe permet de répartir de façon équilibrée la

charge de traitement parallèle entre les nœuds du cluster, ce qui permet aux tâches de s'achever à peu près au même moment dans l'ensemble des nœuds du cluster.

3. Par défaut, l'orchestrateur déclenche M tasktrackers sur les M nœuds de données dans lesquels ont été répartis les M partitions du fichier d'entrée pour exécuter les tâches Map (soit un tasktracker Map pour chaque bloc de fichier). Après, étant donné que chaque tasktracker s'exécute dans un container au niveau du noeud, il n'est pas exclu que plusieurs containers soient déclenchés dans le même noeud. Chaque tasktracker lit le contenu de sa partition par rapport au format d'entrée spécifié par l'utilisateur, le transforme par le processus de hachage définie dans la fonction Map en paires de clés/valeurs. Ce processus de hachage s'effectue en mémoire locale du nœud.

4. Périodiquement, dans chaque nœud, les paires de clés/valeurs sont sérialisées dans un fichier sur le disque dur local du nœud. Ensuite ce fichier est partitionné en r régions (correspondant aux r tâches Reduce spécifiées par l'utilisateur) par une fonction de hachage qui va assigner à chaque région une clé qui correspond à la tâche Reduce à laquelle elle a été assignée. Les informations sur la localisation de ces régions sont transmises à l'orchestrateur, qui fait suivre ces informations aux r tasktrackers qui vont effectuer les tâches Reduce.

5. Lorsque les r tasktrackers Reduce sont notifiés des informations de localisation, ils utilisent des appels de procédures distantes (protocole RPC) pour lire depuis le disque dur des nœuds sur lesquels les tâches Map se sont exécutées, les régions des fichiers Map leur correspondant. Ensuite, ils les trient par clé. Notez au passage que le tri s'effectue en mode batch dans la mémoire du tasktracker Reduce. Si les données sont trop volumineuses, alors cette étape peut augmenter de façon significative le temps total d'exécution du job.

6. Les tasktrackers Reduce itèrent à travers toutes les données triées et pour chaque clé unique rencontrée, ils la passent avec sa valeur à la fonction Reduce écrite par l'utilisateur. Les résultats du traitement de la fonction Reduce sont alors sérialisés dans le fichier r_i (avec i l'indice de la tâche Reduce) selon le format de sortie spécifié par l'utilisateur. Cette fois-ci, les fichiers ne sont pas sérialisés dans le disque dur du nœud tasktracker, mais dans le HDFS, ceci pour des raisons de résilience (tolérance aux pannes).

7. Le job s'achève là, à ce stade, les r fichiers Reduce sont disponibles et Hadoop applique en fonction de la demande de l'utilisateur, soit un « Print Ecran », soit leur chargement dans un SGBD, soit alors leur passage comme fichiers d'entrée à un autre job MapReduce.

Le facteur de réplication c'est le nombre de fois qu'un bloc de fichier est répliqué dans le cluster par le HDFS. Il est définit par l'utilisateur et par défaut est égal à 3. Par exemple, un facteur de réplication égal à 3 signifie que chaque bloc de fichier est répliqué 3 fois dans 3 différents nœuds du cluster par le HDFS. **[19]**

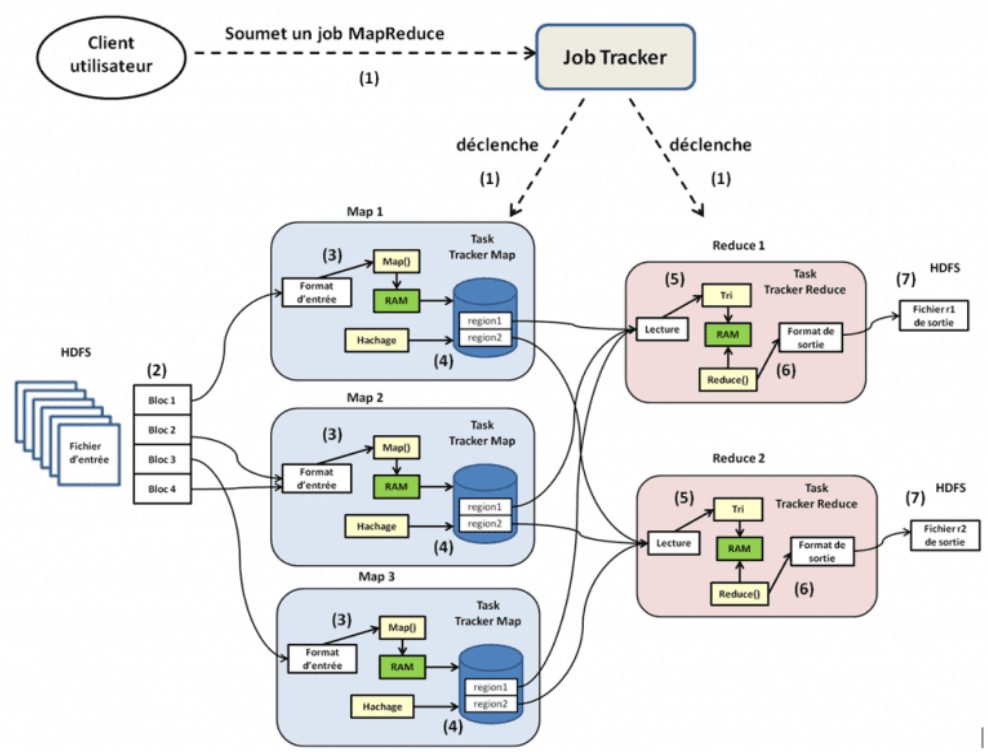


Figure 2.8: étapes d'exécution d'un job MapReduce dans un cluster Hadoop.

Où, La couleur jaune traduit les traitements, la couleur verte représente la RAM, le blanc représente les opérations d'accès aux données et les cylindres bleus représentent les fichiers Map.

4. 5. Les caractéristiques de MapReduce:

*Le modèle de programmation du MapReduce est simple mais très expressif. Bien qu'il ne possède que deux fonctions, map() et reduce(), elles peuvent être utilisées pour de nombreux types de traitement des données. Il est indépendant du système de stockage et peut manipuler de nombreux types des variables.

*Le système découpe automatiquement les données en entrée en bloc de données de même taille. Puis, il planifie l'exécution des tâches sur les nœuds disponibles.

*Il fournit une tolérance aux fautes à grain fin grâce à laquelle il peut redémarrer les nœuds ayant rencontré une erreur ou affecter la tâche à un autre nœud.

*La parallélisation est invisible à l'utilisateur afin de lui permettre de se concentrer sur le traitement des données.

*Avec MapReduce, Hadoop est également très flexible.

*La vitesse est également l'un des principaux caractéristiques [8].

5. Les avantages d'hadoop :

Hadoop présente de nombreux avantages, et les fonctionnalités suivantes rendent Hadoop particulièrement adapté à la gestion et à l'analyse de Big Data :

— Évolutivité : Hadoop permet d'augmenter et de réduire l'infrastructure matérielle sans avoir à modifier les formats de données. Le système redistribuera automatiquement les données et les tâches de calcul pour s'adapter aux changements de matériel.

— Rentabilité : Hadoop apporte un calcul massivement parallèle aux serveurs de base, conduisant à une réduction importante du coût par téraoctet de stockage, ce qui rend le calcul massivement parallèle abordable pour le volume croissant de big data.

— Flexibilité : Hadoop est libre de tout schéma et capable d'absorber n'importe quel type de données provenant d'un nombre quelconque de sources. En outre, différents types de données provenant de sources multiples peuvent être agrégés dans Hadoop pour une analyse plus approfondie.

— Tolérance aux pannes : les données manquantes et les échecs de calcul sont courants dans les analyses de Big Data. Hadoop peut récupérer les données et les échecs de calcul causés par une panne de noeud ou une congestion du réseau [7].

6. Les différents outils de l'écosystème Hadoop:

L'écosystème Hadoop regroupe une large variété d'outils Big Data open source**.** Ces divers outils complémentent Hadoop et améliorent sa capacité de traitement Big Data.

- ✓ **Apache Hive** est un entrepôt de données pour le traitement de grands ensembles de données stockées dans le système de fichiers de Hadoop
- ✓ **Apache Zookeeper** automatise les basculements et réduit l'impact d'un NameNode défaillant.
- ✓ **Apache HBase** est une base de données non relationnelle open-source pour Hadoop
- ✓ **Apache Sqoop** est un outil en ligne de commande pour la migration des données entre Hadoop et les bases de données relationnelles
- ✓ **Apache HCatalog** est un outil de stockage et de gestion de tableaux permettant de trier les données provenant de différents outils de traitement des données [20].

7.Conclusion

Dans ce chapitre, nous avons détaillé le Hadoop, ses composants principaux HDFS et MapReduce et son mode de fonctionnement. Il s'agit d'un outil polyvalent pour les entreprises qui traitent de grandes quantités de données.

Le prochain chapitre représente nos contributions pour la classification du big data en utilisant le Framework hadoop.

Chapitre : 3

Conception

1. Introduction :

Dans ce chapitre nous allons présenter la solution au problème posé. Nous présentons Une méthode de classification dans un environnement BIG DATA sous Hadoop.

Les algorithmes de clustering sont l'un des algorithmes d'apprentissage non supervisé les plus populaires en apprentissage automatique. L'idée de ce type d'algorithme est de regrouper les éléments de données en plusieurs groupes en fonction de la similarité entre ces éléments. Ces algorithmes sont utilisés dans de nombreux domaines tels que les projets d'exploration de données, la détection de modèles, l'analyse d'images, l'informatique médicale et autres.

Il existe de nombreux algorithmes de clustering avec des objectifs similaires et de différentes manières de travailler, telles que la connectivité, le point central, la distribution, le clustering basé sur la densité, etc. Notre solution proposé est basé sur l'algorithme de K-means, qui est l'un des algorithmes de clustering à base de points centraux les plus connus. Nous visons dans ce chapitre à appliquer cet algorithme de clustering sous hadoop. Le but de notre solution est d'accélérer le calcul et introduire le parallélisme pour classifier une grande énorme de données.

2. Définition K-means :

K-Means est un algorithme simple d'apprentissage non supervisé utilisé pour résoudre les problèmes de clustering. Il suit une procédure simple consistant à classer un ensemble de données dans un nombre de clusters, défini par la lettre « **k**» qui est fixé au préalable.

On positionne ensuite les clusters comme des points. On associe tous les observations ou points de données au cluster le plus proche, calculés et ajustés. Puis, le processus recommence en utilisant les nouveaux ajustements jusqu'à ce qu'un résultat souhaité soit atteint.

L'algorithme de clustering K-means est déployé pour découvrir des groupes qui n'ont pas été explicitement définis.

Algorithme K-means

Entrée :

- K le nombre de cluster à former
- Le Training Set (matrice de données)

DEBUT

Choisir aléatoirement K points (une ligne de la matrice de données). Ces points sont les centres des clusters (nommé centroïd).

REPETER

Affecter chaque point (élément de la matrice de donnée) au groupe dont il est le plus proche au son centre

Recalculer le centre de chaque cluster et modifier le centre

JUSQU'A CONVERGENCE

OU (stabilisation de l'inertie totale de la population)

FIN ALGORITHME

Algorithme 1 : algorithme classique K-means

- **Explication les étapes de l'algorithme K-means**

- On précise le nombre de clusters. Nous sélectionnons k points au hasard dans l'ensemble de points (k nombre de clusters). Nous mesurons la distance entre chaque point et k points pour les clusters.

La mesure de distance se fait par des équations mathématiques (il ya plusieurs méthodes pour calculer la distance). Parmi ces méthodes les plus connue est la distance euclidienne.

Distance Euclidienne:

En mathématiques, la distance euclidienne entre deux points dans l'espace euclidien est la longueur d'un segment de droite entre les deux points. Elle peut être calculée à partir des coordonnées cartésiennes des points en utilisant le théorème de Pythagore, donc parfois appelée distance de Pythagore

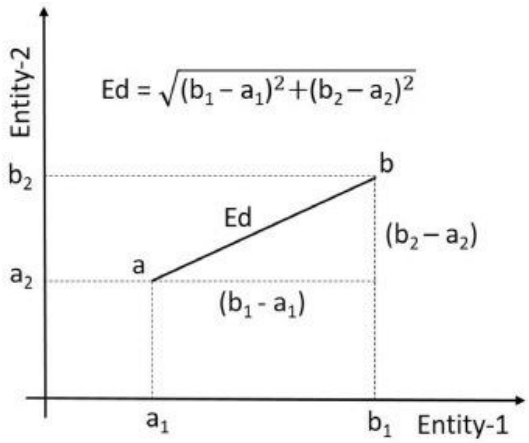


Figure 3.1 : Utilisation du théorème de Pythagore pour calculer la distance euclidienne bidimensionnelle

- placer chaque point avec le cluster le plus proche.

- Nous calculons la moyenne des points de chaque groupe, puis répétons les étapes

- Si les clusters ne changent pas à la dernière étape, on s'arrête et ce sont les clusters.

3. Motivation d'utiliser k-means :

Le k-means est l'algorithme de clustering le plus simple. Il permet de réaliser des analyses non supervisées, de regrouper les individus ayant des caractéristiques similaires. C'est surement la méthode la plus connue quand on doit visualiser rapidement des groupes d'individus.

Le k-means n'est pas une méthode de clustering hiérarchique. Comparé à d'autres méthodes, les temps de calcul sont donc bien plus faibles. Concrètement il peut donc être utilisé sur des volumes de données importants.

- Il est facile à comprendre.
- Il est flexible : L'algorithme K-means s'adapte aux divers changements de vos données. En cas de souci, l'ajustement du segment de cluster permettra d'apporter rapidement des modifications nécessaires à l'algorithme.
- Il est disponible dans la majorité des outils : k-means est l'algorithme le plus répandu. On peut le trouver évidemment dans les outils de Data Science (R, Python, DSS, Spark,…)

4. Présentation de l'approche proposée :

4.1. L'architecture globale :

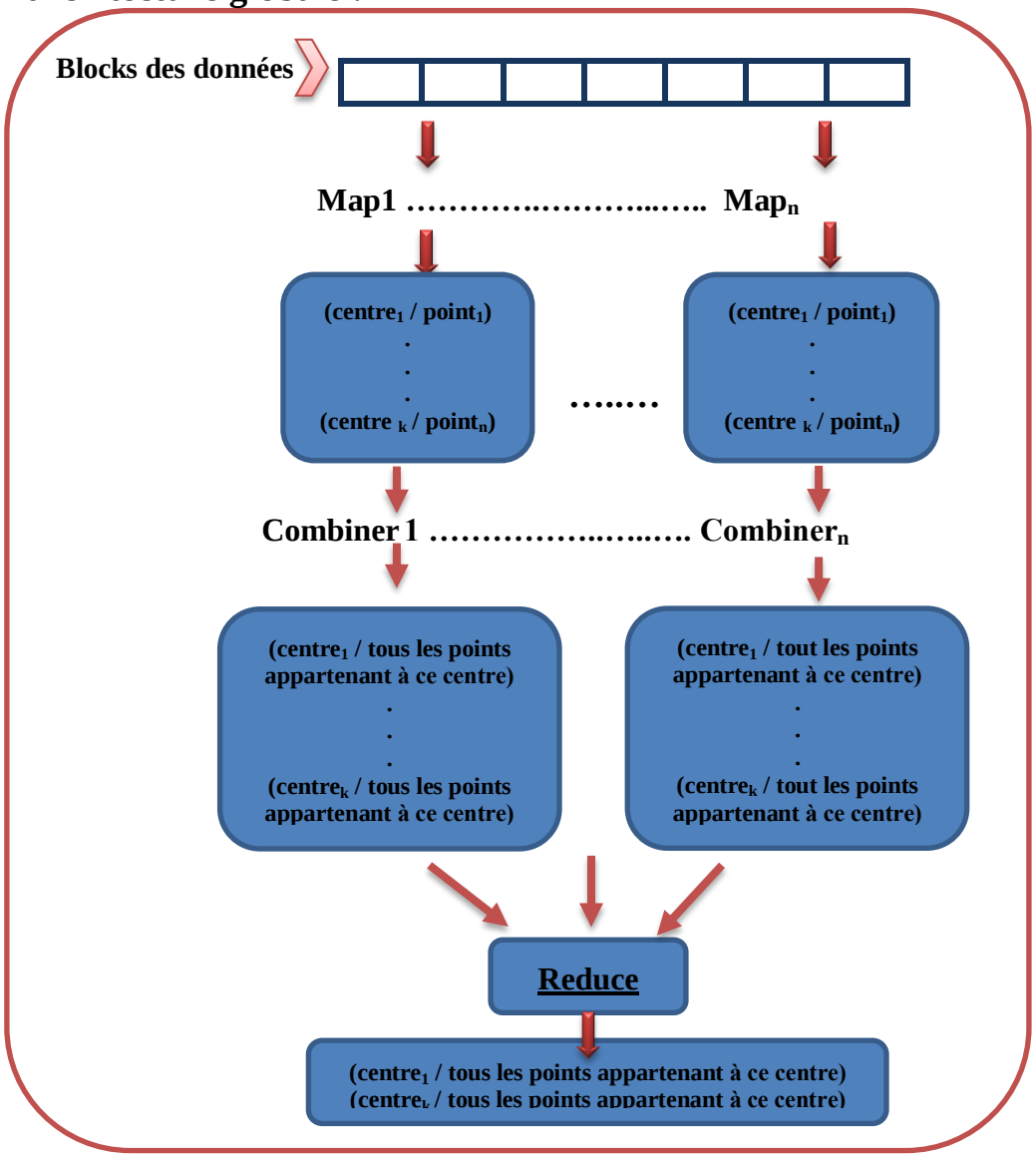


Figure 3.2 : l'architecture de l'approche proposée.

La solution proposée PKMeans (Parallel K-Means) nécessite un type d'un fonctionnement parallèle basé sur MapReduce. La fonction Map exécute la procédure d'attribution de chaque point de donnée au centre le plus proche tandis que la fonction Reduce exécute la procédure de mise à jour des nouveaux centres. Afin de diminuer le coût de la communication réseau, une fonction Combiner est développée pour traiter la combinaison partielle des valeurs intermédiaires avec la même clé (center) pour la sortie de chaque tâche Map.

4.2 L'architecture Détaillée

Le principe de MapReduce est très simple, il consiste à découper une tâche traitant un gros volume de données en plusieurs tâches traitant chacune un sous-ensemble de ces données. MapReduce comporte deux phases Map et Reduce. Dans map, les tâches sont envoyées à tous les nœuds. Chaque nœud traite un ensemble de données associées. Dans Reduce, les résultats sont combinés pour former le résultat final du traitement.

Pour la solution proposée, les fonctions Map, Combiner et Reduce sont détaillées comme suit :

Map : C'est la première étape de la programmation MapReduce.

- Tout d'abord, la fonction map est lancée pour chaque bloc de données. Il calcule la distance entre chaque point de données et chaque centre choisi. Il produit un ensemble des paires (centre/point), Où le centre est le centroïd le plus proche et le point c'est le point de données :

C.-à-d. : placer chaque point avec le cluster le plus proche.

- ✓ L'entrée : un ensemble des points de données d'un $block_x$ et l'ensemble des centres initiaux.
- ✓ Sortie : un ensemble des paires ($centre_i$ /$point_j$)

Combiner : Il s'agit d'une étape facultative dans le modèle MapReduce. Il est utilisé pour améliorer les performances des fonctions MapReduce. Les entrées de l'étape Combiner sont les sorties de la fonction Map. Cette fonction vise à collecter tous les points de données du même cluster. Cette opération est appliquée de manière parallèle à chaque block parmi tous les blocks résultants du map.

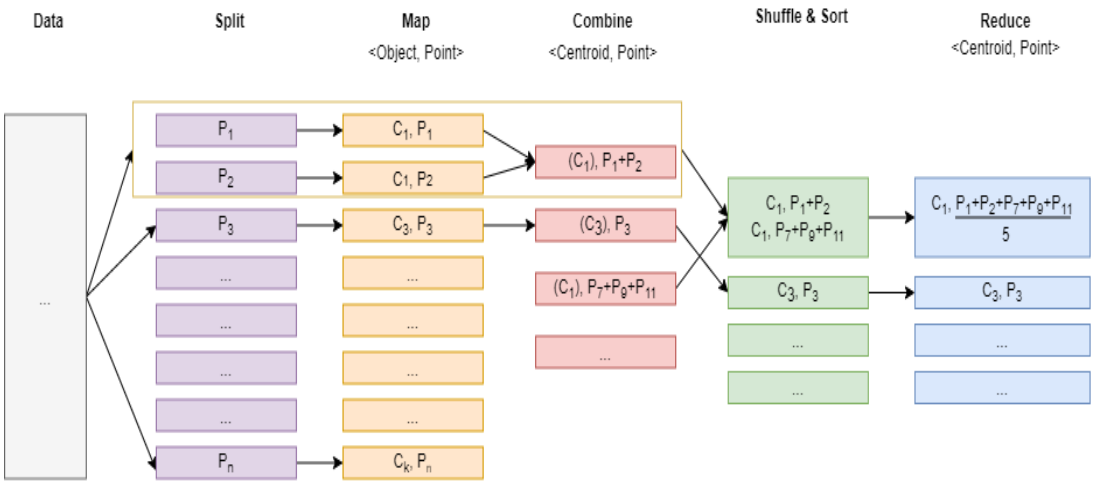
Combiner nous permet de l'utilisation d'un combinateur pour réduire la quantité de données à transmettre au Reducer.

- ✓ L'entrée : un ensemble des paires ($centre_i$ /$point_j$) de la fonction Map pour un $block_X$
- ✓ Sorties : un ensemble des paires ($centre_i$ /tous les points d'un $block_X$ appartiennent à ce centre i).

Reduce : Les entrées de cette étape sont les sorties du processus Combiner.

- À ce stade, la fonction Reduce fusionne tous les blocs qui sont obtenus par Combiner pour faire un classement final entre eux.
  - ✓ L'entrée : un ensemble des paires ($centre_i$ /tous les points d'un $block_X$ appartiennent à ce centre i) résultant pour chaque Combiner
  - ✓ Sortie : un ensemble des paires ($centre_i$ /tous les points appartiennent à ce centre i)

Un nouveau centre pour chaque cluster→ finalement, on a arrivé à un classement final ($centre_i$ /tous les points appartiennent à ce centre i)



Où, $P_{1,n}$ représentent les points de donnée, et $C_{1,k}$ représentent les centres.

Figure 3.3: Application d'algorithme K-means sur MapReduce

- Les algorithmes K-means sur MapReduce :

1) K-means basée sur MapReduce

centroids = k random sampled points from the dataset.
do:
Map:
- Given a point and the set of centroids.
- Calculate the distance between the point and each centroid.
- Emit the point and the closest centroid.

Reduce:
- Given the centroid and the points belonging to its cluster.
- Calculate the new centroid as the aritmetic mean position of the points.
- Emit the new centroid.

prev_centroids = centroids. // **Old center**
centroids = new_centroids.// **New center**
while prev_centroids – centroids < threshold.

Algorithme 2 : K-means basée sur MapReduce

Comme dans l'algorithme classique K-means à la première étape, les centres des clusters sont échantillonnés au hasard à partir de l'ensemble de points de données. La fonction Map prend en entrée un point de données et la liste des centres. Après, il calcule la distance entre le point et chaque centre et affecte le point au centre le plus proche. La fonction Reduce collecte tous les points appartenant à un cluster et calcule le nouveau centre. À la fin de chaque étape, il trouve une nouvelle approximation entre le nouveau et l'ancien centre. Cet algorithme répète jusqu'à ce que la distance entre chaque centre d'une étape précédente et les centres correspondants de l'étape actuelle soit inférieure à un seuil donné.

2) Mapper :

Le mapper calcule la distance entre le point de données et chaque centre. Puis il affecte chaque point au centre le plus proche. Cette fonction retourne une paire (centre, point)

```
class MAPPER
method MAP (file_offset, point)
    min_distance = POSITIVE_INFINITY
    closest_centroid = -1// the nearest center
    for all centroid in list_of_centroids
        distance = distance(centroid, point)
        if (distance < min_distance)
            closest_centroid = index_of(centroid)
            min_distance = distance
    EMIT(closest_centroid, point)
```

Algorithme 3 : Class Mapper

Combiner :

À chaque étape, nous devons collecter les points de données appartenant à un cluster.

```
class COMBINER
  method COMBINER(centroid_index, list_of_points)
    point_sum.number_of_points = 0// the number of points belonging to the cluster
    point_sum = 0// the sum of the points belonging to the  cluster
    for all point in list_of_points:
        point_sum += point
        point_sum.number_of_points += 1
    EMIT(centroid_index, point_sum)
```

Algorithme 4 : Class Combiner

Reducer:

Dans cette fonction, nous pouvons collecter tous les points et calculer le total nombre des points affectés au même cluster. Par conséquent, nous pouvons obtenir les nouveaux centres qui sont utilisés pour la prochaine itération.

```
class REDUCER
   method REDUCER(centroid_index, list_of_point_sums)
      number_of_points = partial_sum.number_of_points // the number of partial points
                                                          intended for Combine x
       point_sum = 0
        for all partial_sum in list_of_partial_sums:
         point_sum += partial_sum // the total sum of all points
           belonging to a cluster
         point_sum.number_of_points +=  partial_sum.number_of_points // the total number
                                                          of points belonging to a cluster
      centroid_value = point_sum / point_sum.number_of_points //calculate the arithmetic
                                                          mean (the new center)
      EMIT(centroid_index, centroid_value)
```

Algorithme 5: Class Reducer

6. Conclusion :

Dans ce chapitre, nous considérons le problème de classification des données dans un environnement de Big Data. Ce problème pose un grand défi à cause de volume croissant des informations. Pour résoudre ce problème, nous proposons un algorithme de clustering de manière parallèle et distribuée basé sur k-means et MapReduce.

Dans le prochain chapitre, Les résultats expérimentaux démontrent que l'algorithme proposé peut bien traiter efficacement un grand ensemble de données.

Chapitre : 4

Implémentation et Analyses

1. Introduction :

Dans ce chapitre, nous présentons l'aspect de la mise en œuvre de notre application. Nous avons commencé par la présentation de l'environnement matériel sur lequel notre système a été développé, les langages de programmation et les outils utilisés.

Enfin, nous donnons une description de notre système en appuyant sur des résultats expérimentaux sur hadoop pour le but de montrer l'efficacité de notre solution.

2. Environnement de Développement :

Dans cette partie nous allons citer l'environnement matériel (Hardware) et logiciel (Software) utilisés.

2.1. Environnement matériel :

Figure 4.1 : Environnement logiciel utilisé.

2.2. Environnement logiciel :

2.2.1. Le langage Python:

Python est le langage de programmation open source le plus employé par les informaticiens. Ce langage s'est propulsé en tête de la gestion d'infrastructure, d'analyse de données ou dans le domaine du développement de logiciels. En effet, parmi ses qualités, Python permet notamment aux développeurs de se concentrer sur ce qu'ils font plutôt que sur la manière dont ils le font. Il a libéré les développeurs des contraintes de formes qui occupaient leur temps avec les langages plus anciens. Ainsi, développer du code avec Python est plus rapide qu'avec d'autres langages.

Les principales utilisations de Python par les développeurs sont :

- la programmation d'applications .
- la création de services web.
- la génération de code.
- la méta-programmation.

Techniquement, ce langage servira surtout pour le scripting et l'automatisation (interaction avec les navigateurs web).

On différencie deux versions : Python 2 et Python 3. Python 2, l'ancienne version propose des mises à jour jusqu'en 2020. Python 3 est la version actuelle. Son interpréteur est plus efficace, ainsi que son contrôle de concurrence.

Figure 4.2 : Python

2.2.2. Docker

Docker est une plateforme permettant de lancer certaines applications dans des conteneurs logiciels.

Selon la firme de recherche sur l'industrie 451 Research, « Docker est un outil qui peut empaqueter une application et ses dépendances dans un conteneur isolé, qui pourra être exécuté sur n'importe quel serveur ». Il ne s'agit pas d'un virtualisation, mais de conteneurisation, une forme plus légère qui s'appuie sur certaines parties de la machine hôte pour son fonctionnement. Cette approche permet d'accroître la flexibilité et la portabilité d'exécution d'une application, laquelle va pouvoir tourner de façon fiable et prévisible sur une grande variété de machines hôtes, que ce soit sur la machine locale, un cloud privé ou public, une machine nue, etc.

Techniquement, Docker étend le format de conteneur Linux standard, LXC, avec une API de haut niveau fournissant une solution pratique de virtualisation qui exécute les processus de façon isolée. Pour ce faire, Docker utilise entre autres LXC, cgroups et le noyau Linux lui-même[1]. Contrairement aux machines virtuelles traditionnelles, un

conteneur Docker n'inclut pas de système d'exploitation, mais s'appuie au contraire sur les fonctionnalités du système d'exploitation fournies par la machine hôte.

La technologie de conteneur de Docker peut être utilisée pour étendre des systèmes distribués de façon à ce qu'ils s'exécutent de manière autonome depuis une seule machine physique ou une seule instance par nœud. Cela permet aux nœuds d'être déployés au fur et à mesure que les ressources sont disponibles, offrant un déploiement transparent et similaire aux PaaS pour des systèmes comme Apache Cassandra, Riak, ou d'autres systèmes distribué.

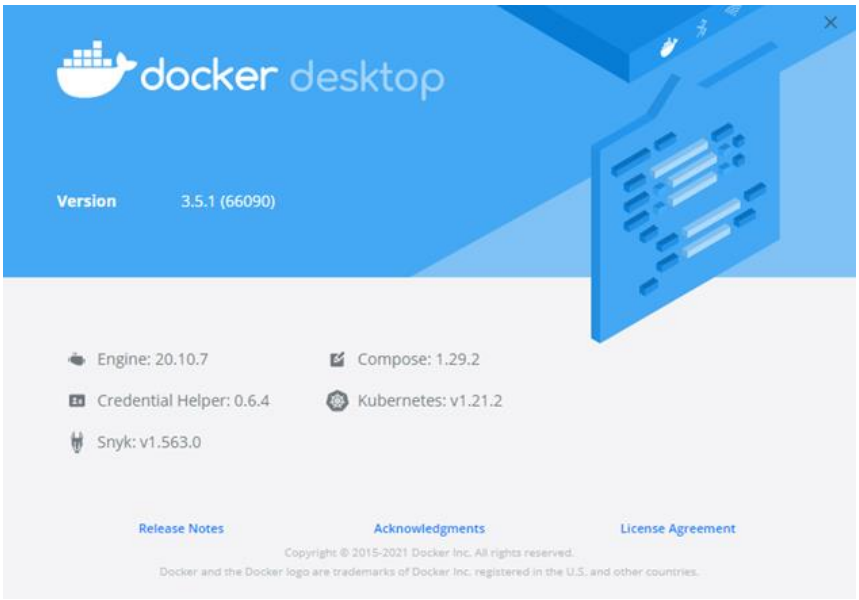


Figure 4.3 : Docker

2.2.3. Visual Studio :

Visual Studio Code est un éditeur de code simplifié, qui est gratuit et développé en open source par Microsoft. Il fonctionne sous Windows, mac OS et Linux. Il fournit aux développeurs à la fois un environnement de développement intégré avec des outils permettant de faire avancer les projets techniques, de l'édition, à la construction, jusqu'au débogage.

Les fonctionnalités proposées par Visual Studio Code sont nombreuses. On retrouve notamment:

- ✓ **La prise en charge de plusieurs centaines de langage de programmation**, telles que C, C#, C++, CSS, HTML, Java, JavaScript, JSON, Markdown, PHP, Powershell, Python, TypeScript, YAML….
- ✓ **IntelliSense**, une fonction de complétion intelligente du code،
- ✓ **Un débogueur intégré** pour accélérer votre boucle d'édition, de compilation et de suppression des bugs.

- ✓ **Une interface d'édition**, qui intègre des raccourcis clavier, des sélections multiples, un enregistrement automatique de votre travail, une fonction rechercher/remplacer, le formatage du code source‹…
- ✓ **Peek**, une fonction qui permet de parcourir rapidement le code source et de naviguer entre les fichiers‹
- ✓ **Les commandes Git intégrées** ainsi que la gestion du contrôle des sources (SCM).

Visual Studio Code permet également aux développeurs de créer et d'utiliser des extensions grâce à son API, afin de personnaliser leur utilisation de l'outil. Il est livré avec un support pour JavaScript, TypeScript et Node.js.

Figure 4.4 : Visual Studio Code

2.3. Gestion de Base de Données (phpMyAdmin)

2.3.1. XAMPP :

XAMPP est un ensemble de logiciels permettant de mettre en place un serveur web local, un serveur FTP et un serveur de messagerie électronique. Il s'agit d'une distribution de logiciels libres (X (cross) Apache MariaDB Perl PHP) offrant une bonne souplesse d'utilisation, réputée pour son installation simple et rapide. Ainsi, il est à la portée d'un grand nombre de personnes puisqu'il ne requiert pas de connaissances particulières et fonctionne, de plus, sur les systèmes d'exploitation les plus répandus.

Il est distribué avec différentes bibliothèques logicielles qui élargissent la palette des services de façon notable : OpenSSL, Expat (analyseur syntaxique de fichiers XML) PNG, SQLite, zlib… ainsi que différents modules Perl et Tomcat. Nombre de ces extensions étant inutiles aux débutants, une version allégée - version lite - est en conséquence aussi proposée.

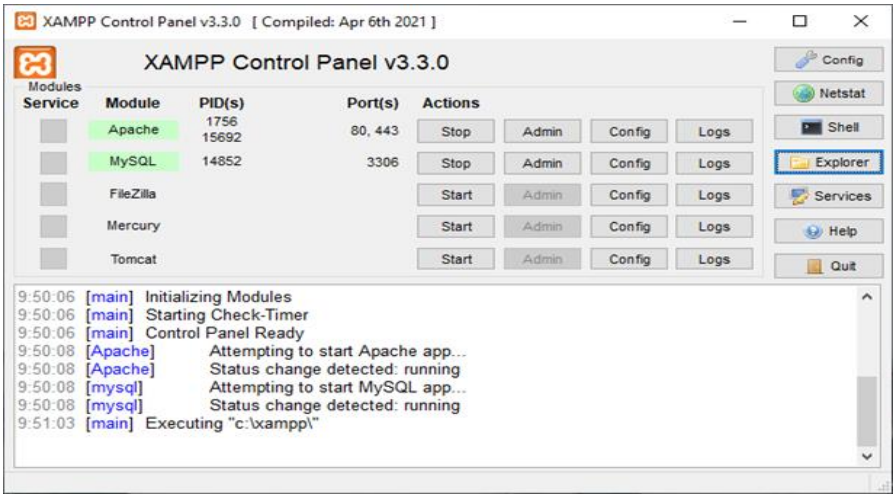


Figure 4.5 : XAMPP**.**

2.3.2. MySQL :

MySQL est un système de gestion de bases de données relationnelles (SGBDR) propriétaire, gratuit, performant, très populaire, multi-threadé, multi-utilisateurs…

MySQL est principalement un *serveur* de bases de données. Pour s'y connecter localement ou à distance, on utilise un *client*. Il peut s'agir de la commande mysql, ou couramment d'un script PHP. Il faudra dans ce cas installer le module php-mysql qui permet à PHP de communiquer avec un serveur MySQL.

3. Dataset :

Après avoir examiné un groupe d'échantillons pour les patients diabétiques, tels que:

- l'ensemble de données provient à l'origine de l'Institut national du diabète et des maladies digestives et rénales. En particulier, tous les patients ici d'au moins 21 ans d'origine indienne Pima.

- L'ensemble de données représente 10 ans (1999-2008) de soins cliniques dans 130 hôpitaux américains et réseaux de prestation intégrés. Il comprend plus de 50 caractéristiques représentant les résultats des patients et des hôpitaux.

Nous avons profité de ces données pour créer des échantillons aléatoires à travers les valeurs des analyses médicales nécessaires.

3.1. Définitions du diabète et du prédiabète :

Le diabète est un trouble métabolique caractérisé par la présence d'une hyperglycémie due à une réduction de la sécrétion d'insuline ou de l'action de l'insuline, ou des deux. L'hyperglycémie chronique liée au diabète est associée à des complications micro vasculaires à long terme assez spécifiques affectant les yeux, les reins et les nerfs, ainsi qu'à un risque accru de maladies cardio vasculaires (MCV). Les critères diagnostiques du diabète sont basés

sur les seuils de glycémie associés à la maladie et les maladies micro vasculaires, la rétinopathie en particulier.

Le terme « prédiabète » fait référence à un taux de glycémie à jeun anormal, à une altération de la tolérance au glucose ou à une hémoglobine glycosylée (HbA1c) entre 6,0 % et 6,4 %, ce qui expose les personnes à un risque élevé de diabète et de complications liées à la maladie.

3.2. Classification du diabète :

La majorité des cas de diabète peuvent être globalement classés en deux catégories : le diabète de type 1 et le diabète de type 2. Certains cas échappent toutefois à cette classification. Le diabète gestationnel découle intolérance au glucose qui se manifeste ou est détectée pour la première fois pendant la grossesse.

- **Diabète de type 1:**

Ce type de diabète apparaît en général chez le sujet jeune mais peut se développer à tout âge. L'étiologie exacte reste inconnue mais une pathologie auto-immune détruisant les cellules béta du pancréas est souvent évoquée, ainsi que des facteurs environnementaux et certains virus ou bactéries. Le pancréas ne produit plus du tout ou pas assez d'insuline ce qui provoque les symptômes classiques d'hyperglycémie: soif, polyurie et polydipsie, perte de poids involontaire. Ces patients nécessitent un apport exogène d'insuline pour vivre.

- **Diabète de type 2 :**

Il peut apparaître à tout âge mais se développe généralement chez les adultes d'âge moyen ou les personnes âgées qui peuvent souffrent déjà d'un syndrome métabolique (surpoids, obésité, dyslipidémie, hypertension, etc.). L'étiologie est inconnue mais il apparaît plus fréquemment dans certaines ethnies ou après un diabète gestationnel.

Ce type de diabète est souvent asymptomatique et peut se développer silencieusement pendant plusieurs années et provoquer déjà des complications. Parfois certains signes sont présents tels que: infections fréquentes et guérison lente, syndrome des ovaires.

- **Diabète gestationnel :**

Ce diabète apparaît pendant la grossesse. L'intolérance au glucose se développe en raison d'une sécrétion insuffisante de l'insuline dans le cadre d'une résistance accrue à son action pendant la grossesse. Ce diabète est généralement asymptomatique, d'où l'importance du dépistage qui doit être systématique chez toutes les femmes entre la 24ème et la 28ème semaine de grossesse. Certains facteurs de risque sont associés à son apparition.

Ces patientes ont souvent besoin d'un traitement à l'insuline ainsi que d'une surveillance étroite de leur glycémie pendant la grossesse.

MÉTHODES DIAGNOSTIQUES	GLYCÉMIE (mmol/L)	INTERPRÉTATION
Glycémie plasmatique à jeun A jeun ou « régime 0 calories » durant au moins 8h	< 5,6	Valeur normale
	5,6 – 6,9	Pré-diabète
	≥ 7.0	Diabète (ou diabète gestationnel si femme enceinte)
Glycémie postprandiale Test de tolérance au glucose : - Prise orale de 75g de sucre - 2h plus tard, ad. glycémie	< 7,8	Tolérance au glucose normale
	7,8 – 11,0	Pré-diabète
	≥ 11,1	Diabète
Glycémie (à tout moment) chez des patients symptomatiques (CLINIQUE: Polyurie, polydipsie, perte pondérale inexpliquée)	≥ 11,1	Diabète
Glycémie veineuse entre la 24e et 28e semaine de grossesse après prise orale de 75g de sucre (HGPO)	à jeun : ≥ 5,1 **ou** 1h post HGPO : ≥ 10 **ou** 2h post HGPO : ≥ 8,5	Diabète gestationnel (une seule valeur anormale suffit à poser le diagnostic)
HbA1c (Hémoglobine glycosylée ou glyquée)	4,4 – 5,7%	Valeur normale
	5,7 – 6,4 %	Pré-diabète
	≥ 6,5 %	Diabète

Table 4.1: Les méthodes diagnostiques

Le diagnostic ne peut être posé sur le résultat d'un test de glycémie capillaire.

Le diagnostic de diabète nécessite une mesure de la glycémie veineuse et doit être confirmée par une deuxième mesure prise un autre jour. Cette attestation pas nécessaire si la glycémie est supérieure à 11,1 mmol/L et que le patient est symptomatique ou en cas de diabète gestationnel.

3.3. Dataset:

Nous utilisons des datasets avec 100, 1000, 10000,100000 points (patients diabétiques) sur hadoop. Pour chacun d'eux, nous avons un jeu de données:

- des points avec deux dimensions (HepGe, HbA1c) et 3 centres (trois classes).

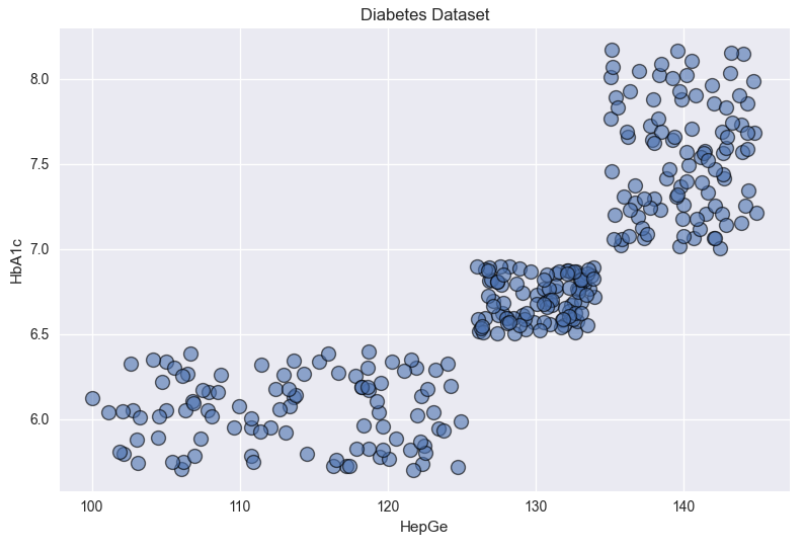


Figure 4.6 : illustration de distribution des 300 échantillons de patients diabétiques

- **HepGe** :(hyperglycémie provoqué orale) est un examen de dépistage du diabète sucré (diabète gestationnel, diabète de type 1 et diabète de type 2). Il consiste en l'absorption d'une quantité standard de glucose par voie orale avec suivi de la réponse physiologique de l'organisme (glycémie et insulinémie). Il se réalise sur ordonnance dans un laboratoire médical.

- **HbA1c** : (hémoglobine glyquée) est le pourcentage d'hémoglobine ayant fixé du sucre dans le sang. Elle est le reflet de la glycémie moyenne (taux de sucre dans le sang) des 3 derniers mois précédant le dosage en laboratoire.

Un exemple de la façon de mesurer la distance entre un point et un centre avec deux dimensions :

p = [107.728, 6.003] et C=[107.275, 5.807]

Distance = sqrt ((107.275 - 107.728)^2) + (5.807 - 6.003)^2)

Distance = sqrt (0.205 + 0.0384) = sqrt (0.2436)

Distance = 0.4935

On note la répartition des échantillons de patients en grands groupes. Pour chaque jeu de données, nous exécutons l'algorithme 10 fois. Pour chaque exécution, le seuil (Threshold) est défini sur 0,0001 et l'itération maximale est définie sur 50.

Considérant que l'algorithme k-means est sensible aux centroids initiaux et que nous avons utilisé une initialisation aléatoire. Nous allons montrer dans ce tableau le temps d'exécution des itérations.

Nombre d'échantillons	Temps d'exécution
100	9.1137 s
1000	25.1982 s
10000	26.6614 s
100000	27.3514 s

Table 4.2: le temps d'exécution selon le nombre d'échantillons en utilisant MapReduce et k-means

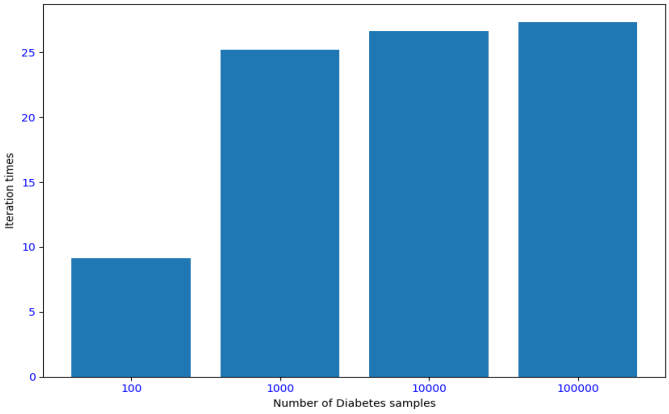


Figure 4.7 : graphique montrant le temps d'exécution selon le nombre d'échantillons en utilisant MapReduce et k-means

Lorsque le nombre est très petit, 100 échantillons, le temps consommé dans le traitement est très petit et le nombre d'étapes est petit car les groupes sont presque pré-divisés et clairs. Les échantillons 1000, 10000 et 100000 sont approximativement proches en termes de temps d'exécution et de nombre des itérations.

La méthode du coude utilise la somme de la distance au carré (SSE) pour choisir une valeur idéale de k (nombre de clusters) en fonction de la distance entre les points de données et leurs clusters attribués. Nous choisissions une valeur de k où le SSE commence à s'aplatir et nous voyons un point d'inflexion. Une fois visualisé, ce graphique ressemblerait un peu à un coude.

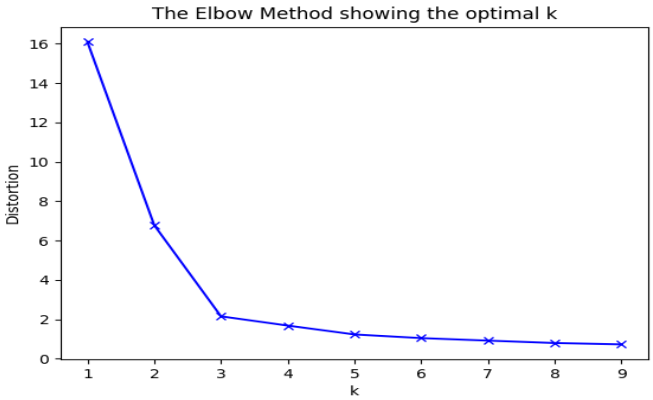


Figure 4.8 : Elbow point

Le graphique ci-dessus montre que k = 3 est probablement un bon choix pour le nombre de clusters. Il y a des situations où le graphique ne ressemble pas à un coude, cela rend les choses très difficiles pour choisir la valeur de k.

Le code pour générer le graphique ci-dessus et le composant de modélisation de cet algorithme est fourni ci-dessous dans l'implémentation.

```python
from sklearn.cluster import KMeans
from sklearn import metrics
from scipy.spatial.distance import cdist
import numpy as np
import matplotlib.pyplot as plt

def elbow(dataset):
    x,y = np.loadtxt(dataset, delimiter=",", unpack=True)

    # create new plot and data
    plt.plot()
    X = np.array(list(zip(x, y))).reshape(len(x), 2)
    colors = ['b', 'g', 'r']
    markers = ['o', 'v', 's']

    # k means determine k
    distortions = []
    K = range(1,10)
    for k in K:
        kmeanModel = KMeans(n_clusters=k).fit(X)
        kmeanModel.fit(X)
        distortions.append(sum(np.min(cdist(X, kmeanModel.cluster_centers_, 'euclidean'), ax
    # Plot the elbow
    plt.plot(K, distortions, 'bx-')
    plt.xlabel('k')
    plt.ylabel('Distortion')
```

Figure 4.9 : package sklearn.cluster

Nous appelons une bibliothèque ou un package **sklearn.cluster**, nous attribuons l'appel à **K-Means** pour toutes les fonctions de classification de k-means.

La distorsion dans la graphique « Elbow point» est la somme de la distance au carré entre chaque point et le centre dans un groupe.

4. Présentation des Interfaces Graphiques :

4.1. Interface d'accueil

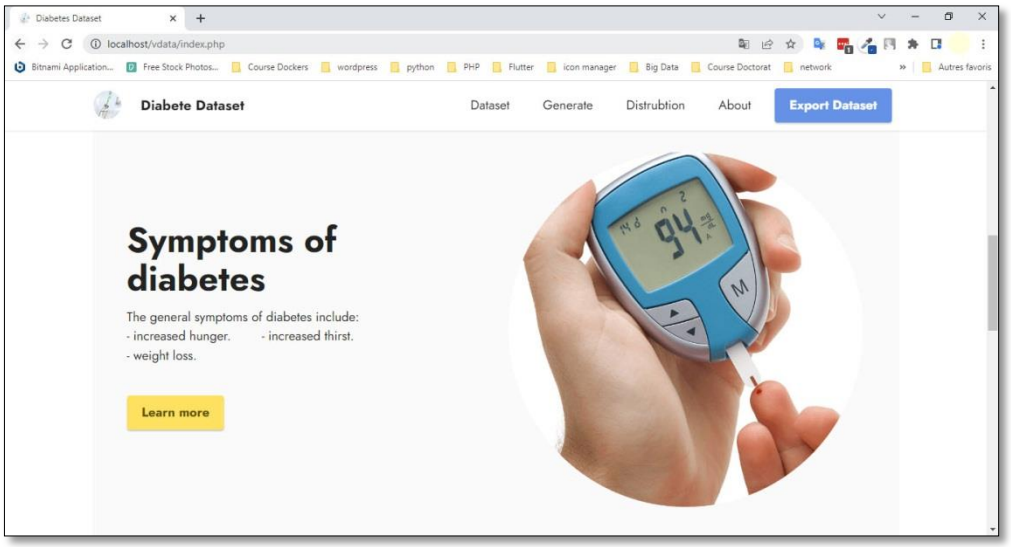


Figure 4.10 : Interface d'accueil

4.2. Dataset diabetes sample :

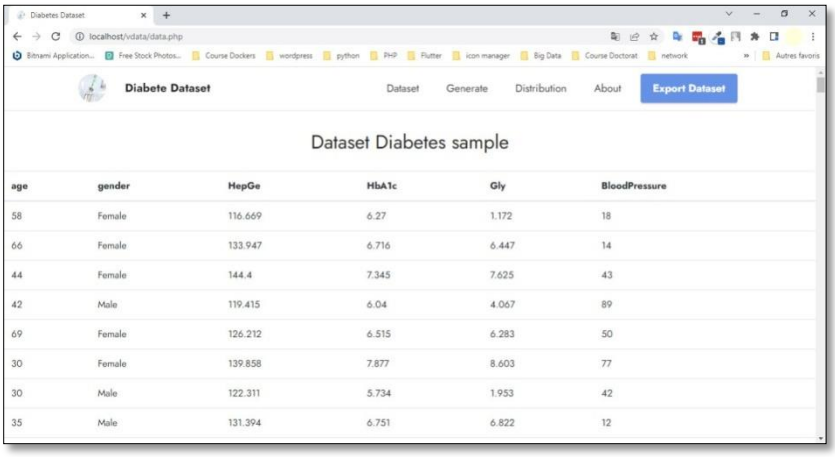


Figure 4.11 : dataset diabetes sample

4.3. Scatter chart diabetes :

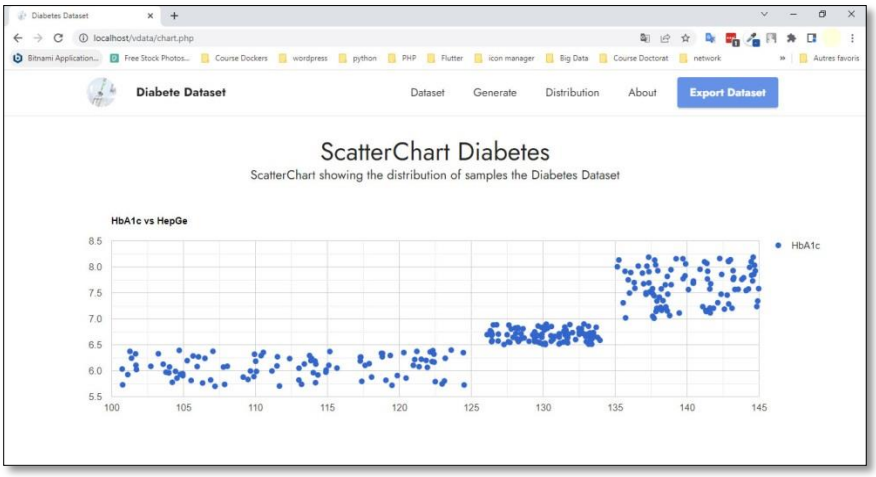


Figure 4.12 : scatter chart diabetes

4.4. Interface d'introduction

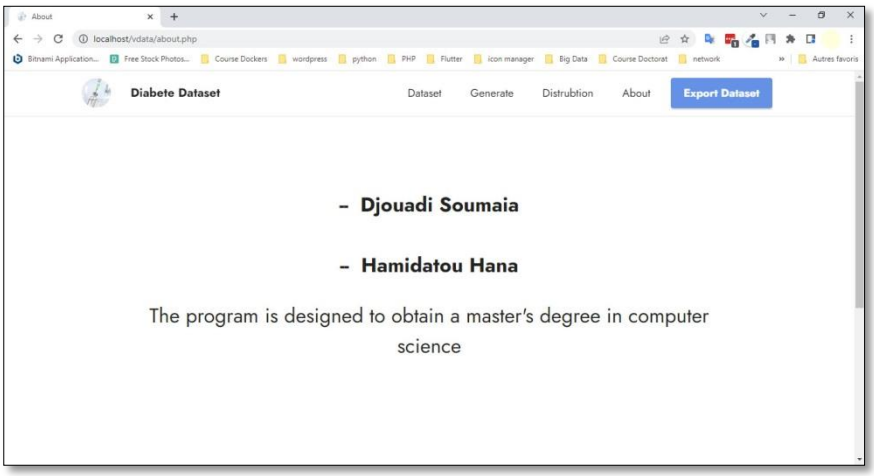


Figure 4.13 : Interface d'introduction

5. Mise en œuvre et expérimentation :

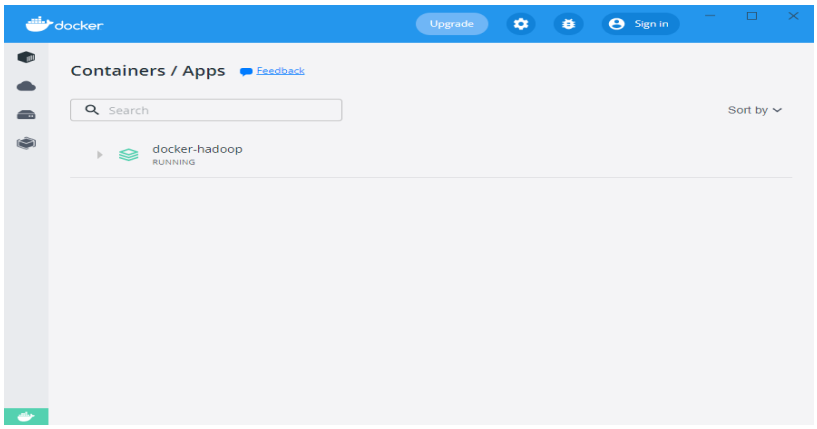
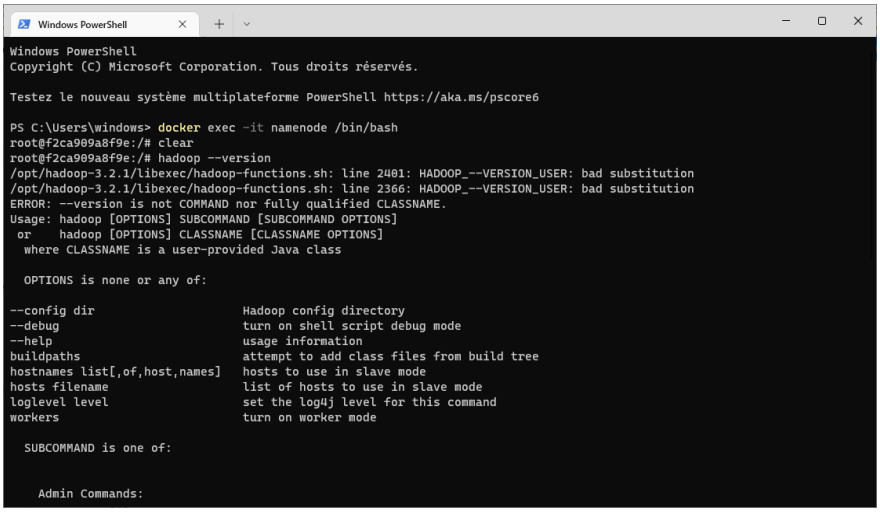
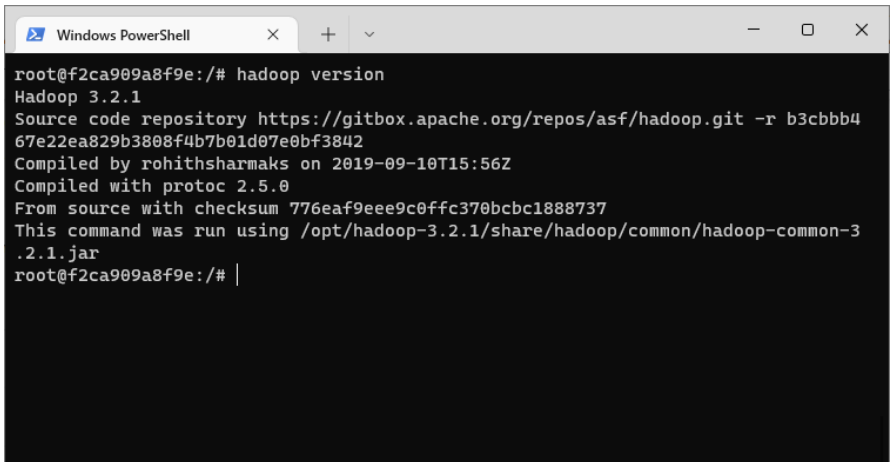


Figure 4.14 : Interface docker

Cette fenêtre montre l'application **Docker** permettant de lancer certaines applications dans des conteneurs logiciels.





Les deux images précédentes affichent la version d'hadoop installé (hadoop 3.2.1)

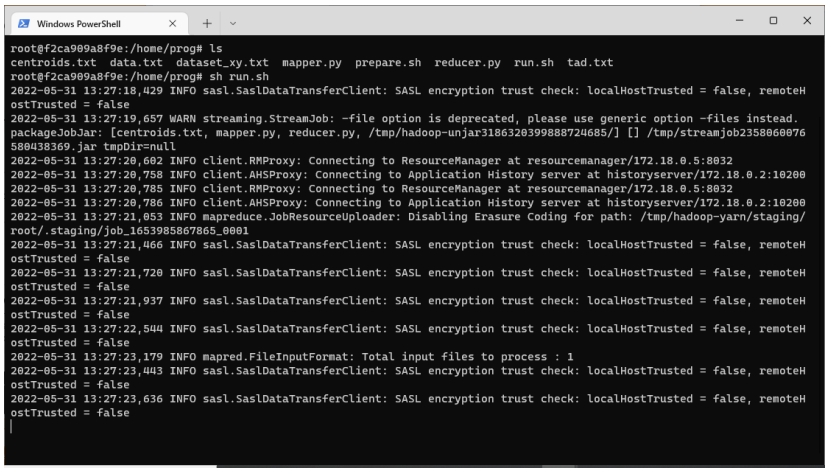


Figure 4.15 : l'exécution de MapReduce

Cette fenêtre en dessus montre l'exécution de MapReduce et lance à la fois le mapper, combiner et reducer.

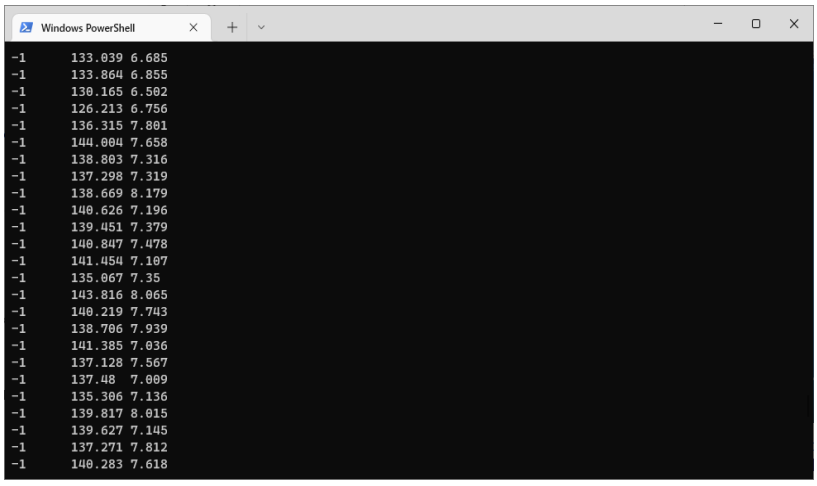


Figure 4.16 : résultat de mapper

Cette fenêtre en dessus montre le résultat de mapper.

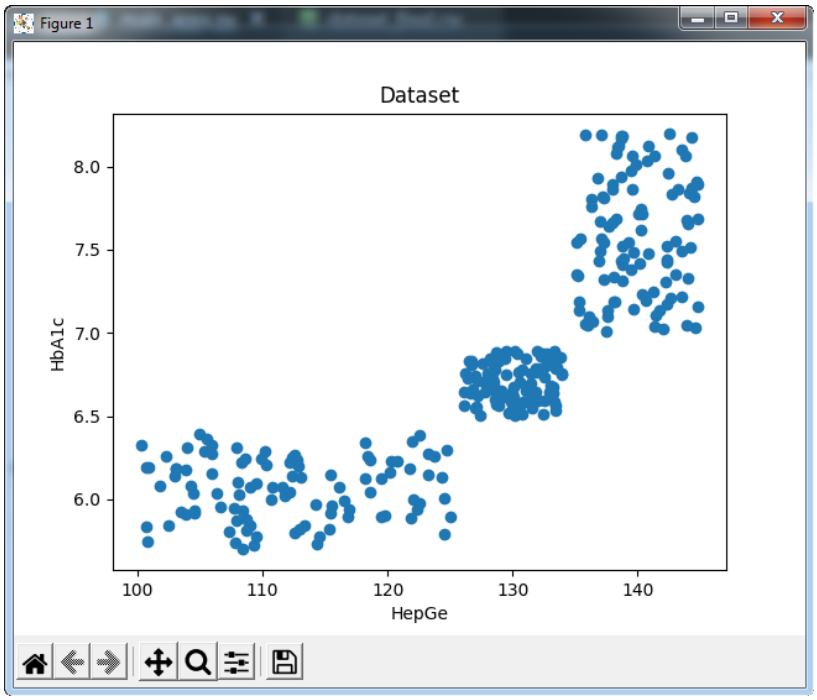


Figure 4.17 : la distribution des échantillons du DATASET

Cette image en dessus montre la distribution des échantillons du DATASET.

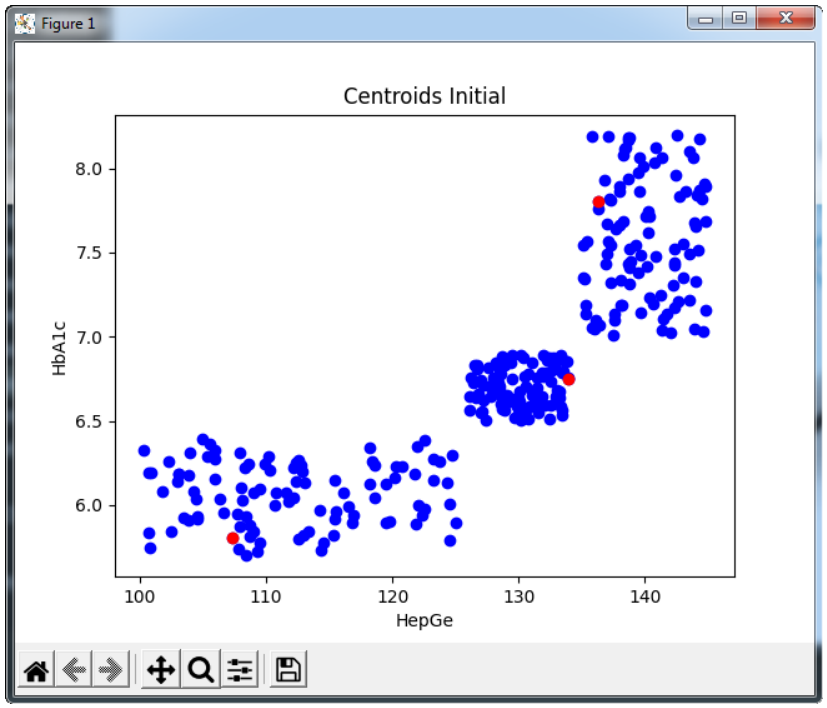


Figure 4.18 : les centres initiaux

L'image en dessus montre la distribution des échantillons du DATASET et les points rouges montrent les centres initiaux.

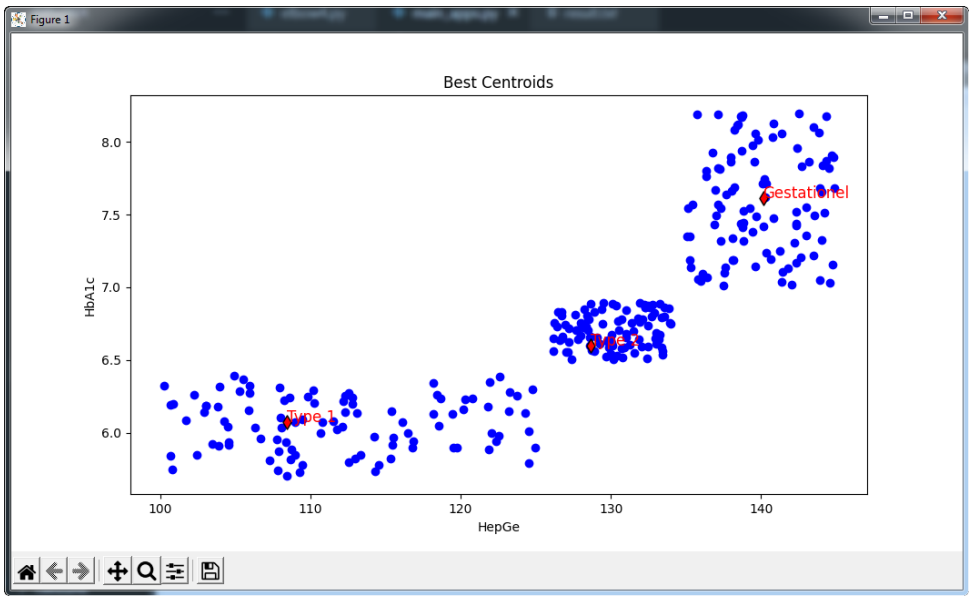


Figure 4.19 : les centres finals

Cette image montre la distribution des échantillons du DATASET après l'exécution d'algorithme de K-means et les points rouges montrent les centres finals.

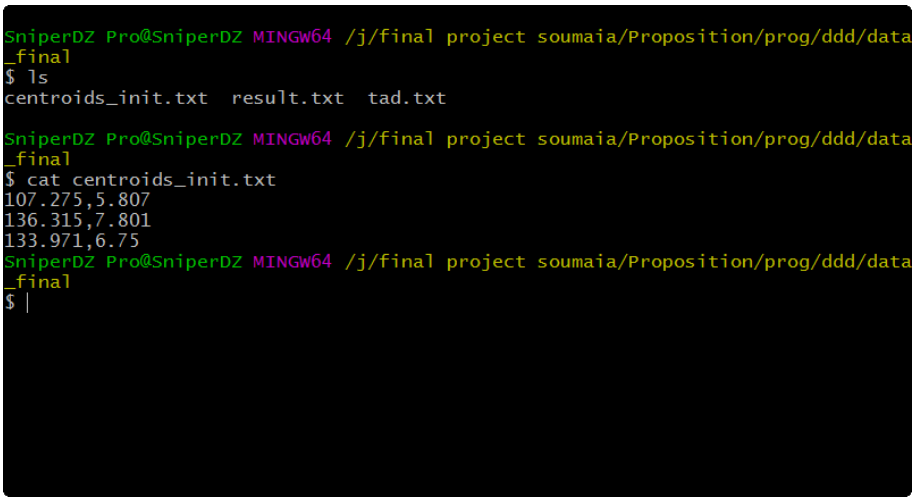


Figure 4.20 : les valeurs des centres initiaux

Cette fenêtre en dessus montre les valeurs des centres initiaux que nous avons choisis aléatoirement à partir des points d'échantillonnage.

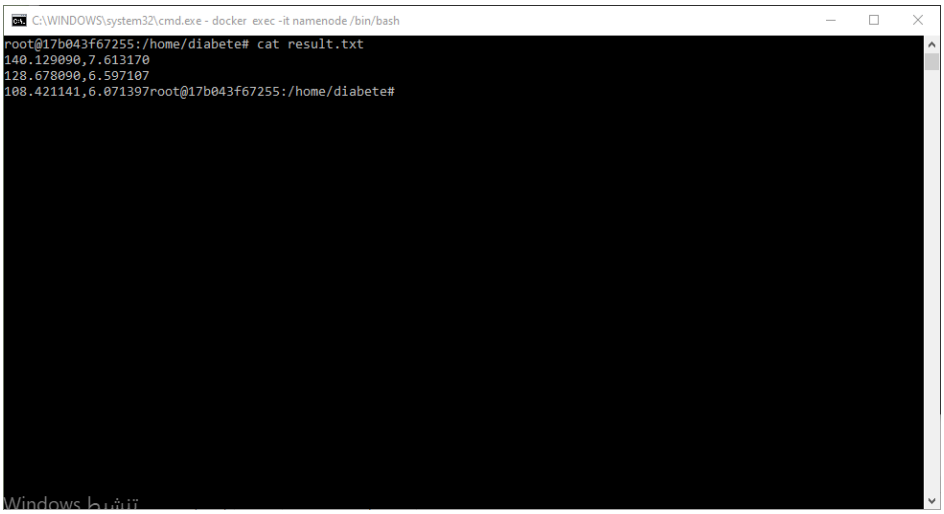


Figure 4.21 : les valeurs des centres finals

Cette fenêtre montre les valeurs de nouveaux centres.

6. conclusion

Ce chapitre a été consacré à la présentation du l'environnement de développement en termes de différents outils et les technologies utilisés. Nous avons simulé la solution proposée pour résoudre le problème de classification des données. Comme nous l'avons montré dans ce chapitre l'efficacité de notre proposition pour minimiser le temps d'exécution de manière distribué et parallèles pour classifier un grand nombre des données.

CONCLUSION GÉNÉRALE

CONCLUSION GÉNÉRALE

Depuis quelques années le nombre de données numériques créées explose**,** provenant des moteurs de recherches, des mails, des réseaux sociaux, des applications mobiles, des objets connectés, de capteurs, des achats en ligne, etc… Ces volumes d'informations récoltés, qui ne peuvent pas être traités par des outils classiques et auxquels les entreprises sont confrontées**,** forment le Big Data.

La classification de ces grandes quantités des données a attiré beaucoup d'attention de la recherche. Pour résoudre ce problème, de nombreux algorithmes de classification ont été proposés au cours des dernières décennies.

Les volumes croissants d'informations résultant par le progrès de la technologie, rend la méthode de classification à très grande échelle de données une tâche difficile. Afin de résoudre le problème, de nombreux chercheurs tentent de concevoir des algorithmes de classification parallèles efficaces. Dans ce mémoire, nous proposons un algorithme de classification parallèle k-means basé sur MapReduce, qui est une technique de programmation parallèle très puissante.

Dans ce mémoire, nous avons décrit les expérimentations permettant l'analyse de performance de la solution proposée. Ces expérimentations sont menées pour classifier des personnes diabètes.

Les résultats expérimentaux démontrent que l'algorithme proposé peut bien évoluer et traiter efficacement de grands ensembles de données.

Perspectives :

Plusieurs améliorations et extensions peuvent être envisagées pour enrichir la solution proposée :

- Notre travail futur consiste à appliquer la solution proposée sur un véritable ensemble de données et sur un grand nombre de données.

- on peut résoudre ce même problème par des autres méthodes de classification sur hadoop. Le but est de faire une comparaison en termes de temps d'exécution entre notre solution proposée et les solutions qui utilisent des autres méthodes de classification.

Bibliographies

Bibliographies

[1]-ALEX HOLMES, Hadoop in Practice, MANNING SHELTER ISLAND,2012.

[2]- Evan Stibbs , "Big Data,Big Innovation", 2014.

[3]-Hao Zhang, Gang Chen, In-Memory Big Data Management and Processing : A Survey.

[4]-Ludovic Denoyer et Sylvain Lamprier, Certificat Big Data, Introduction à MapReduce/Hadoop et Spark.

[5]-Medfouni Hayet. Validation de clustering des donn_ees dans un contexte big data, memoire de master. Universit_e Larbi Ben M'hidi Oum El Bouaghi, 2017/2018.

[6]-Medfouni Hayet. Validation de clustering des donn_ees dans un contexte big data,memoire de master. Universit_e Larbi Ben M'hidi Oum El Bouaghi, 2017/2018.

[7]-MERABET Mohamed,Un ordonnancement efficace des tâches pour les applications Big data, Mémoire Doctorat informatique, UNIVERSITE DEDjillali Liabès de Sidi Bel Abbès,2018/2019.

[8]-Mlle MOUZAIA Chahinas epouse REDJDAL Mlle ABBAS Célia. Le Contrôle d'Accés au Big Data.Cas d'Etude : Internet des Objets, Universite A/Mira de Bejaia, Memoire de Master. 2016/2017.

[9] -[consulte le 24-02-2022]. https ://www.lebigdata.fr/denition-big-data.

[10]- [consulte le 25-02-2022].https://www.redsen-consulting.com/fr/inspired/data-analyse/cas-dutilisation-big-data.

[11]- [consulté le 15-12-2021].http://fr.wikipedia.org/wiki/Entrep%C3%B4t_de_donn.

[12]-[consulte le 25-02-2022]. https://www.oracle.com/fr/big-data/guide/what-is-big-data.html

[13]-[consulte le26-02-2022].https://www.piloter.org/business-intelligence/technologie bigdata.html.

[14] -[consulte le 14-2-2022]. https://superdatacamp.com

[15]- [consulté le 15-2-2022]. http s ://www.mba-esg.com/actus/enjeux-big-data.

[16] -[consulte le 24-02-2022]https://datascientest.com/hadoop

[17] -[consulte le 24-02-2022]https://www.piloter.org/business-intelligence/hadoop.htm .

[18] -[consulte le 24-02-2022]https://fr.wikipedia.org/wiki/MapReduce.

[19]-[consulte le 24-02-2022]https://www.data-transitionnumerique.com/hadoop-mapreduce.

[20]-[consulte le 24-02-2022] https://actualiteinformatique.fr/data/big-data/hadoop-et-big-data