République Algérienne Démocratique et Populaire

Ministère de l'Enseignement Supérieur et de la Recherche Scientifique

UNIVERSITE DE SAAD DAHLED DE BLIDA

Faculté des sciences

Département informatique

Mémoire présenté par :

Melle. ZEGAOUI Nadjet

Mr. CHAIBI El Mahdi

En vue d'obtention le diplôme de Master

Domaine : Informatique

Filière : Informatique

Spécialité : Informatique General

Option : Ingénierie du logiciel

Sujet : Conception et implémentation d'une architecture Big data « Hadoop » pour l'analyse et les prévisions des départs

Promoteur : Mr. KAMECH Abdallah Hichem

Encadreur : Mr. MEKIDACH Mounir

Soutenue le 22 juin 2016, devant le jury composé de :

Mr. H.DERRAR	USDB	Président
Mr. M.HAMOUDA	USDB	Examinateur

- 2015/2016 -

Dédicace

*C'est grâce à **Dieu**, le clément, le miséricordieux, que je termine mon projet de fin d'études.*

C'est avec une grande joie que je dédie cet humble mémoire, fruit de mes études en exprimant ma profonde reconnaissance

Aux personnelles plus chères :

*A la mémoire de **ma mère ALLAH YARHAMHA***

*A **mon père** qui par sa sagesse m'appris le sens des valeurs et le respect envers les autres.*

A ma cher mère Malika

*Pour **mes frères**, auxquels je souhaite autant de réussite, de bonheur et de prospère dans la vie.*

*A mes cousines **A**mel et **M**anal.*

A mon cher Ami Abdelrahim BENACHOUR

*A mes meilleurs amis, **Soumia**, **Imene , Nadjia, Yacine ,Sanna et Radia**.*

Et à ceux ou celles qui sont proches à mon cœur qui m'encouragent à donner le meilleur de moi-même.

Nadjet

Dédicace

Dédicace Ce travail, et bien au-delà, je le dois à mes très chers parents qui ont toujours été là pour moi, et qui m'ont donné un magnifique modèle de labeur et de persévérance. J'espère qu'ils trouveront dans ce travail toute ma reconnaissance et tout mon amour.

A mes très chèrs frères et sœurs, présent dans tous mes moments de travail par leur soutien moral et ses belles blagues. Je vous souhaite un avenir plein de joie, de bonheur, de réussite et de sérénité.

A ma chérie Sara que Dieu tu gardes pour moi.

A ma binôme, A mes amies : Abderrahmane, Abdel Hadi, Yasmine, Moufida, Dalel, Malik, Diaa, Lotfi, ma cousine Nadjet, Rafik, Bilal et a tous mes amis avec lesquels j'ai passée des merveilleuses années universitaires. A toutes les personnes qui ont participé de loin ou de près à l'élaboration de ce travail. A toutes les personnes que j'ai oubliées de citer.

Chaibi El Mahdi

Remerciements

Nous tenons à remercier en premier lieu et exprimer notre gratitude à notre promoteur ***A. KAMECH*** *qui a été un exemple pour nous par son sérieux, ses conseils. Son approche scientifique nous a permis d'accomplir notre travail d'une façon facile et fluide.*

Nos vifs remerciements vont aussi à notre encadreur ***M. MEKIDECH*** *pour avoir mis à notre disposition tous les moyens nécessaires pour notre travail, et aussi pour ses conseils et orientations fructueuses.*

Nous adressons également nos remerciements aux membres de la commission de jurys.

Nous tenons à remercier aussi Mr.Cherahabil et Mr.Amir pour leurs précieuses aide durant notre stage

Nous ne pourrions terminer ces remerciements sans une pensée à l'ensemble de nos enseignants ainsi qu'à nos collègues avec qui nous avons partagé d'agréables moments durant cette formation Master à Université Saad Dahlab Blida.

Sommaire

Liste de figures

Liste de tableaux

ملخص

أثبت تحليل البيانات أهميتها في معرفة الاكتشاف والتنبؤ وفي دعم اتخاذ القرار. في عصر البيانات الكبيرة, السؤال الذي يطرح نفسه في كثير من الأحيان ما هي التكنولوجيات والأبنية الأنسب لدعم العمليات التحليلية على نطاق واسع.

بسبب حجم البيانات الكبيرة، يصبح من الصعب جدا إجراء تحليل الفعال باستخدام التقنيات والبنى التقليدية. تحقيقا لهذه الغاية، كان هناك ظهور تطبيقات تحليل البيانات أو تطبيقات البيانات الكبيرة كنوع جديد من التطبيقات البرمجية كنوع جديد من التطبيقات البرمجية التي تحليل كميات كبيرة من البيانات باستخدام الإطار المعالجة المتوازية

هذا العمل هو سهم من رؤية طويلة الأجل التي تبدأ مع نهج خفيفة، التي يشمل أولا تطوير تطبيقات للتنبؤ و التحليل انطلاقا من البيانات الكبيرة بمختاف انواعها, و الذي سيتم باستعمال كلاودرا هادوب.

والهدف من هذا العمل هو تصميم بنية البيانات الكبيرة وتنفيذ ذلك، وهذا من البيانات التي تم جمعها في إطار المنظمة المضيفة لأي سجل على كتلة التخزين .سيتم حفظ هذه البيانات، سابقة التجهيز وتحليلها لإنشاء تقارير والاتجاهات في وقت لاحق.

RESUME

L'analyse des données a prouvé son importance dans la découverte des connaissances, les prévisions et dans l'aide à la décision. À l'ère du Big Data, la question se pose souvent de savoir quelles sont les technologies et les architectures les mieux adaptées pour soutenir des processus analytiques à grande échelle. En raison de cette grande taille des données, il devient très difficile d'effectuer une analyse efficace en utilisant les techniques et les architectures traditionnelles. A cet effet, il ya eu, l'apparition des applications d'analyse de données ou Big Data Analytics Applications (BDA Apps) qui constitue un nouveau type d'applications logicielles qui analysent de grandes quantités de données à l'aide de Framework de traitement parallèle (par exemple, Hadoop).

Ce travail s'insère dans une vision à long terme qui débute par une approche légère, qui consiste tout d'abord par le développement d'une application pour la prévision et l'analyse des départs, à partir de données de type Big data (structuré et semi structuré) exploitant de grande quantité de données et qui s'effectuera dans une Distribution Hadoop Cloudera(CDH)

L'objectif de ce travail consiste à concevoir une architecture Big data et l'implémenter, qui à partir des données collectées au sein de l'organisme d'accueil pour tout enregistrer sur un cluster de stockage. Ces données seront sauvegardées, prétraitées et analysées afin par la suite de créer des rapports et des tendances.

Mots clés:

Applications d'analyse Big data, analyse, prevision, stockage, Hadoop, Distribution Hadoop Cloudera.

Abstract

Data analysis has proved its importance in knowledge discovery, prevision and the decision support. In the era of Big Data, the question often arises as to what technologies and architectures that are best suited to support analytical processes on a large scale.

Because of this large data, it becomes very difficult to make an effective analysis using traditional techniques and architectures. For this purpose, there is an emergence of data analysis or Big Data Analytics Applications (Apps BDA), which is a new type of software applications that analyse large amounts of data in applications using parallel Processing Framework (Eg Hadoop).

This work is part of a long-term vision that begins with a light approach, which involves first the development of an application for the prediction and analysis of departures from Big data type data (structured and semi-structured), carrying large amounts of data, and that will be all done in a Cloudera distribution Hadoop.

The objective of this work is to design a Big data architecture and implement it, that from the data collected within the host organization for any record on a storage cluster. These data will be saved, pre-treated and analysed to later create reports and trends.

Keywords:

Big data analysis applications, analysis, prevision, storage, Hadoop, Cloudera Hadoop Distribution

INTROUCTION GENERALE

Chaque jour, 2,5 de téraoctets de données sont générées dans le monde. D'ici 2020, il est prévu que la taille des données se multiplie de 50 fois (Miranda, 2013) **[1]**. Depuis le 21ème siècle, l'expansion de la donnée est en train d'atteindre toutes les entreprises, elle est le symbole d'une transformation économique et sociale globale.

Ces volumes donnés sont liés aux caractéristiques de l'univers numérique actuel, c'est-à-dire cette masse de données provient des services désormais inévitables dans notre quotidien : des capteurs de géolocalisation, des Smartphones, tablettes, vidéos, photos, satellites, applications mobiles et plates-formes de partage...etc. Ce sont des technologies « Machine to machine » qui font dialoguer les objets entre eux, inter opérer et transmettre ainsi, parfois à notre insu, des données nous concernant mais dont nous n'avons pas connaissance ni conscience.

Par conséquent, le monde est devenu de plus en plus complexe, et les domaines d'affaires commencent à être étroitement liés, ce qui redéfinit les relations entre producteur, consommateur et bien et service. Face à cette complexité, les organisations et leurs leaders trouvent des difficultés à prendre les décisions en comptant seulement sur l'expérience, d'autant plus qu'ils sont en temps de crise où il faut trouver les vecteurs de performances. Ils savent très bien qu'ils manquent de bons services de données pour leurs décisions.

Ces organisations reçoivent quotidiennement un flot croissant d'informations hétérogènes et souvent non structurées issues de sources multiples. Sauf que, dû à la complexité de leurs traitements, elles sont abondantes par les responsables, surtout que les méthodes traditionnelles de traitement d'information étaient jusqu'à présent incapables de donner un sens à ces volumes d'informations « dormantes ».

L'augmentation en taille de base de données et les nouveaux besoins d'analyse dans l'entreprise qui évolue dans un environnement complexe caractérisé par l'interaction de plusieurs systèmes, ont engendrés l'essor du phénomène : « Big Data », « Hadoop » étant son principal Framework.

L'exploitation et l'intégration de ces données dans les processus de l'organisation peut permettre à l'entreprise d'améliorer les processus de prise de décision pour adapter en temps réel son approche Marketing et sa relation client, valoriser son image sur les

réseaux sociaux, optimiser ses processus de gestion logistique et concevoir de nouveaux produits et services **[1].**

Toutes les techniques utilisées jusqu' ici ne font que l'analyse sur des masses de données numériques et quantitatives. Nous avons une autre tendance à aboutir, de plus en plus pressante, qui est la prédiction automatique du comportement dans différentes activités de l'entreprise en se basant sur ses données massives existantes.

Les données numériques de Big data sont tellement considérables, qu'elles ont imposé une redéfinition des modes de traitements et d'approches des informations stockées. Avec la popularité de ce nouveau phénomène, un certain nombre de questions se posent :

- Qu'est-ce que le concept du Big data ? qu'elles sont ses origines ? et comment s'est-t-il apparu ?
- Pourquoi les méthodes de traitement traditionnelles ont échoué à sa manipulation ? et quelles sont les technologies alternatives ?
- Quels sont les outils pouvant extraire un intérêt mesurable du Big data ?
- Avec ces masses de données, qu'elles sont les technologies de stockages existantes pouvant héberger de telles données gigantesques ?

Pour cela, notre stage qui s'est déroulé à la Division Exploration de Sontarach et plus précisément dans la direction Gestion du Personnel et Moyens, dans le but de se familiariser avec l'application de Big Data qui doit permettre une prédiction plus concrète des départs pour décider et agir. Notre mission dans ce mémoire est de répondre à ces questions, mais nous allons, plus précisément, essayer d'atteindre les objectifs suivants :

- Accéder aux différentes sources de données de type Big data, et collecter celle qui a le plus de potentiel pour faire produire nos prédictions
- Choisir la plateforme la plus adéquate pour leur stockage, selon leurs types, leurs formats, et leurs intentions d'exploitation.
- Après avoir étudier les solutions possibles pour leurs traitements et leurs analyses, effectuer les différentes opérations de traitements conformes à leurs motifs d'utilisation.

- Elaborer une solution sous forme d'un tableau de bord adaptable à tout domaine pouvant, à travers les informations résultantes de l'analyse des Big data, générer des rapports et des tendances qui décrivent la situation du départ des employés.

Pour la présentation de notre travail, nous avons opté pour la démarche suivante : traitée le sujet en quatre chapitres :

Dans le chapitre préliminaire de notre travail, nous présentons l'étude de l'existant et la problématique.

Le **chapitre II** Concept Big Data, Hadoop qui est son principale Framework.

Le **chapitre III** Plateformes de stockage du Big data et ses technologies

Le **chapitre III** Différentes solutions relatives aux applications de traitement et d'analyse du Big data (BDA Apps)

Et pour finir dans le dernier chapitre, nous aborderons la conception de l'architecture et son implémentation technique.

CHAPITRE I : ETUDE DE L'EXISTANT

I.1. Introduction

Etant en fin de formation du cycle Master en ingénierie logiciel, un stage s'avère nécessaire pour rendre complet notre cursus. Ce stage a pour but de nous permettre de joindre la pratique à la théorie mais aussi de faire face aux réalités du monde professionnel. C'est donc dans cette optique que la Division Exploration Sontarach nous a accueilli au sein du **Développement Ressources Humaines** pour un stage au cours duquel nous avons pu apprendre beaucoup de choses.

Dans ce chapitre, nous allons présenter l'organisme d'accueil et les règles de départs, ainsi nous allons cerner la problématique et les besoins de l'entreprise.

I.2. Présentation de l'organisme d'accueil

SONATRACH est une compagnie étatique algérienne et un acteur international majeur dans l'industrie des hydrocarbures.

I.2.1 Missions de la division Exploration

La direction Exploration a été élevée au rang de division le 04 Avril 1987 et elle a été créée le 18 juillet 1988.

Les émissions de la division exploration sont :

- La conduite et le développement des activités de prospection et de recherche des Hydrocarbures.
- La participation avec les autres divisions aux appels d'offres d'exploration en Algérie et à l'étranger.
- La participation à l'évaluation des offres de partenariats que des projets d'exploration en Algérie et à l'étranger.
- La mise en œuvre de la stratégie de la société en matière d'exploration.
- La préparation, l'établissement, et la recommandation des programmes techniques d'exploration et leurs suivis.
- Le développement et la conduite des travaux d'analyse en matière de géologie et géophysique.
- La constitution en un centre d'excellence et d'expertise technique et scientifique dans le domaine de l'exploration, et l'appui aux activités de la société en la matière.

La **figure I.1** représente l'organigramme de la division Exploration/Sonatrach :

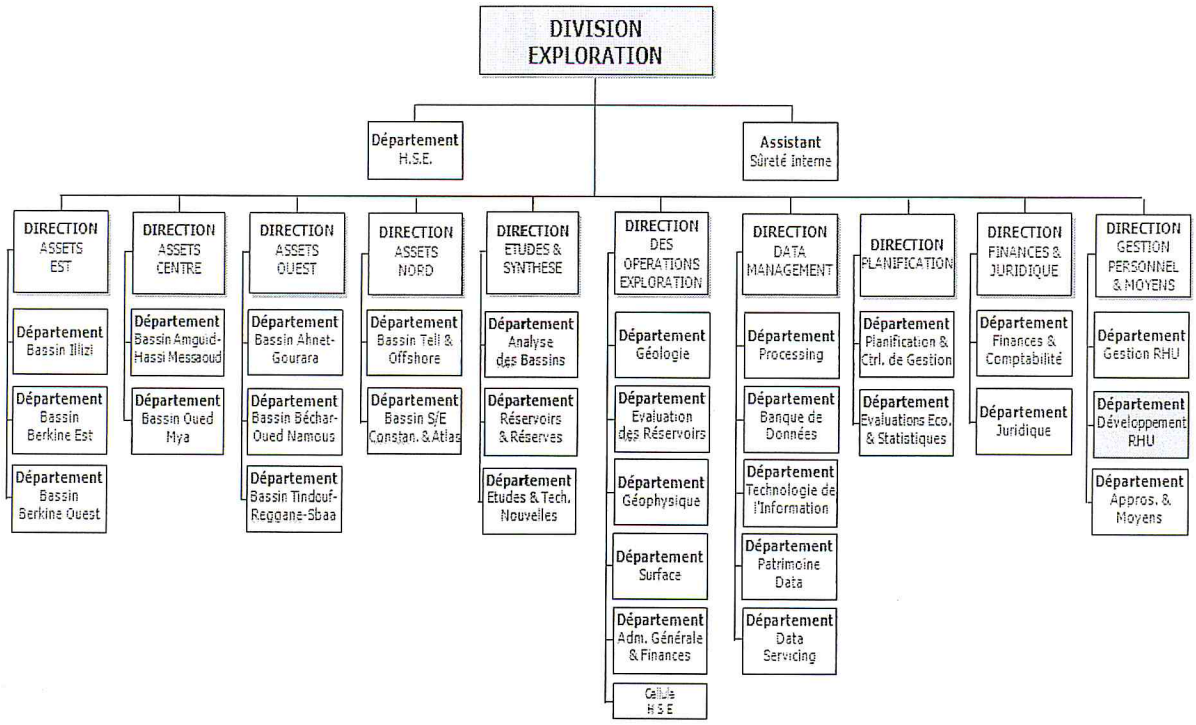


Figure I.1 : Organigramme de la division Exploration /SONATRACH.

I.2.2 Missions de La direction Gestion du Personnel et Moyens

La direction **Gestion du Personnel et Moyens** a pour principales missions :

- La mise en œuvre des politiques, règles et procédures arrêtées par la société en matière de gestion des ressources humaines.
- L'élaboration des plans annuels et pluriannuels de la Division en matière d'emploi, de formation et salaire et leur transmission à la Direction Ressources Humaines de l'activité Amont (DRH/Amont).
- L'élaboration et la transmission à la DRH/Amont du tableau de bord Ressources Humaine de la division.
- La réalisation et le suivi des actions de formation de courte durée dans le cadre des plans et budgets approuvé.
- L'élaboration des bilans annuels d'exécution des plans.
- La gestion et le suivi des organigrammes des effectifs de la division.
- La gestion administratives paie et social du personnel (hormis des chefs de services et plus) du siège de la division.
- La prévention et le règlement, au niveau de la division, des conflits socioprofessionnels individuels et collectifs.

I.2.3 Missions de département Développement Ressources Humaines

Le Département Développement Ressources Humaines a pour missions essentielles :

- L'application des politiques de l'entreprise en matière de sélection, induction, gestion des carrières et formation
- L'élaboration et la mise à jour des manuels d'organisation et de procédures spécifiques à la Division
- La satisfaction des besoins de structures en personnel compétent et qualifié.
- L'élaboration la mise en œuvre et le suivi de développement ressources humaines (plan de mobilité plans de carrières, plans de succession...).
- La mise en place et le suivi du système de management de la performance.
- La réalisation et le suivi du tableau de bord des bilans et plans à court et moyen terme de la Division Exploration en matières d'emploi, de formation et de masse salariale.

I.2.4 Missions de service Gestion de Carrière

Le service Gestion de Carrière est chargé de réaliser les missions principales suivantes :

- Gérer en collaboration avec les autres services RH, le fichier de carrières informatise « RESHUM », de l'ensemble du personnel de la Division.
- Elaborer, consolide, analyser et commenter les prévisions des effectifs (budget emploi et salaire) annuelles et pluriannuelles, ainsi que le plan de recrutement.
- Examiner toute requêtes relative à l'évolution de carrière et donner un avis sur la base d'adéquation profils / exigences de poste.
- Organiser supervise et mettre en place les systèmes de planification des effectifs de gestion des carrières et participer aux actions de formation.
- Réaliser les analyses, études ou enquêtes spécifiques relatives aux personnels demandées par la Division ou l'Entreprise.

I.3. Problématique

L'entreprise est un système complexe dans laquelle transitent une très grosse masse de données dont on devrait faire face en prenant en compte la vitesse avec laquelle sont déversées ainsi que leurs variétés (textes, vidéo, non structurés. . .. Etc.). Sans un

dispositif, l'entreprise ne peut pas améliorer sa performance, peut très vite être dépassée et ne peut plus fonctionner avec une qualité et quantité de service satisfaisantes car il serait difficile de continuer à traiter et à stocker ces données avec la méthode classique en utilisant les bases de données relationnelles et les datawarehouses.

L'enjeu de toute entreprise, qu'elle soit industrielle ou de services, c'est de mémoriser et traiter l'information à un temps record en faisant appel à des technologies informatiques. Pour répondre à cette problématique un nouveau paradigme est né c'est le phénomène Big Data, Hadoop étant son principal Framework. D'ailleurs c'est le cas de la Division Exploration qui n'est pas en reste de ce phénomène

La division exploration fait face au départ de compétences, le départ peut être une mise en retraite, mutation, licenciement, démission. Les prévisions des départs sont nécessaires pour toute catégorie d'emploi (cadre, maitrise et exécution) dans le but est d'atteindre un certain nombre d'employés de qualité dans l'entreprise,

Notre défi sera de concevoir une plateforme Big data (hadoop, HDFS et MapReduce) afin de collecter et d'analyser toutes les données qui peuvent aider à l'élaboration d'un plan de prévision de départ potentiel et de départ en retraite.

I.4. Départs

I.4.1 Définition de départ

Le départ de salarié fait suite à la rupture du contrat de travail, autrement dit le salarié marque la fin de son parcours professionnel au sein de l'organisation. Quelles que soient les raisons, il implique l'accomplissement d'un certain nombre de formalités.

I.4.2 Etude des cas de départ

Le départ est regroupé en quatre types : le départ en retraite, la mutation, le licenciement, la démission et le décès qui est un imprévu.

I.4.2.1 Départ à la retraite

Le départ en retraite est la cessation du contrat de travail résultant de la décision du salarié qui, en considérant son âge et sa situation vis-à-vis des organismes de retraite (Sécurité sociale et caisses de retraite complémentaire), décide de faire valoir ses droits à la retraite.

Le départ en retraite du salarié s'effectue sous réserve du respect d'un préavis, et donne lieu au paiement par l'employeur d'une indemnité de départ en retraite, tous deux

fixés par voie légale ou conventionnelle. On distingue 4 types de retraite qui sont résumé comme suivant :

- **Retraite à âge légal**

- L'âge légal de départ à la retraite est fixé à soixante (60) ans. Les femmes ont la possibilité de partir à la retraite à cinquante-cinq (55) ans.

- Cette condition est facultative, elles ne peuvent en bénéficier que si elles le demandent expressément.

- De plus, elles peuvent, si elles ont élevé un ou plusieurs enfants, pendant au moins neuf ans, bénéficier d'une réduction d'âge d'une année par enfant dans la limité de trois (03) années (art. 8 loi 83-12).

- **Retraite proportionnelle**

Cette retraite est valable à partir de l'âge de 50 ans et si l'employé totalise un minimum de 20 ans de travail effectif ayant donné lieu à un versement de cotisations de sécurité sociale.

- L'âge et la durée d'activité sont réduits de 05 ans pour les femmes salariées.

- La législation ne fixe pas de montant minimum dans les cas de la retraite proportionnelle.

- **Retraite sans condition d'âge**

Effective si l'employé réunit 32 années au moins de travail effectif, avec un versement des cotisations de sécurité sociale.

- **Retraite anticipée**

La retraite anticipée à lieur selon le décret n°94-10 du 26 mai 1994. Les dispositions du présent décret sont applicables à tous les salariés du secteur économique susceptibles de perdre leur emploi de façon involontaire, pour raison économique et dans le cadre soit d'une compression d'effectif, soit d'une cessation légale de l'activité de l'employeur.

Pour bénéficier de la mise en retraite anticipée, le salarié doit justifier des conditions ci-après :

- Etre âgé d'au moins 50 ans s'il est de sexe masculin et de45 ans s'il est de sexe féminin.

- Réunir un nombre d'années de travail ou assimilées validées au titre de la retraite égale à 20 années au moins et avoir cotisé à la sécurité sociale pendant au moins 10 ans de façon plein dont cinq 5 années précédant la fin de la relation de travail qui justifie et ouvre droit à une retraite anticipée.

- Figurer sur la liste des travailleurs devant faire l'objet d'une compression d'effectif ou sur celle identifiant les salariés d'un employeur en cessation d'activité.

- Ne pas bénéficier d'un revenu procuré par une activité professionnelle quelconque.

- **Dispositions particulières concernant les moudjahidines**

Les travailleurs justifiant de la qualité de moudjahidine ou moussabiline bénéficiant s'ils le souhaitent des dispositions suivantes :

- L'âge de départ en retraite est fixé à 55 ans.

- En cas d'invalidité due à la participation à la lutte de libération, la condition d'âge précitée est encore réduite d'une (01) année pour chaque tranche de 10/100.

- La tranche d'invalidité de 05/100 donne lieu à une réduction de six (06) mois.

- La période reconnue au titre de la participation à la lutte de libération nationale, ainsi que la période reconnue au titre de l'invalidité citée ci-dessus sont validées en double par la caisse de retraite.

- Les cotisations de sécurité sociale dues (parts employeur et employé) sont à la charge de l'entreprise (cf. article 29 loi 83-12).

Il y a un cas particulier dans la retraite c'est dans le cas où l'âge du salarié est supérieur à 60 ans et sa cotisation est moins de 15 ans, dans ce cas travailleur à le doit de poursuivre suivre sa carrière dans l'entreprise comme maximum 5 ans.

I.4.2.2 Démission

La démission est une rupture volontaire de la relation du travail. La démission est un droit reconnu au travailleur. Il est tenu de signifier, par écrit" sa volonté non équivoque de mettre fin à la relation du travail, au terme d'un préavis de démission obligatoire.

Le préavis obligatoire est de :

- Un (01) mois pour les agents d'exécution.

- Deux (02) mois pour les agents de maîtrise,

- Six (06) mois pour les cadres.

- Six (06) à douze (12) mois pour les cadres supérieurs.

Le préavis peut être réduit par accord express des deux (2) parties.

I.4.2.3 Mutation

La modification du poste de travail et le transfert du salarié dans une autre unité de l'entreprise décidés par l'employeur se dénomme mutation. Le changement de poste de travail à l'intérieur d'un même établissement lorsqu'il préserve la qualification et le niveau salarial du travailleur ne relève que du pouvoir de gestion du chef d'entreprise. Et la Chambre sociale de juger que constitue une mutation dans un même secteur géographique celle d'un salarié, dont le lieu de travail se trouvait dans un établissement situé au chef-lieu du département, et qui, en raison de sa fermeture, est muté dans un autre établissement de la couronne urbaine de ce même chef-lieu.

Seulement les fonctionnaires titulaires qui sont mutés, soit une mutation interne ou externe. La mutation interne est un changement d'affectation au sein d'un même ministère.

Elle a lieu à la demande du fonctionnaire ou à l'initiative de l'administration par contre la mutation externe est un changement d'employeur, de CT ou d'établissement de santé.

Elle est prononcée sur demande du fonctionnaire.

• Procédure de la mutation :

- Etablissement d'une demande manuscrite de la part du fonctionnaire l'administration.

- Etablissement d'un avis favorable de la hiérarchie de la part de l'organisation.
- Validation de la demande.
- Envoie de l'imprimé à la structure par un agent spécifique.
- Validation et retour de l'imprimé de la part de la structure d'accueil.
- Prendre la décision.

I.4.2.4 Licenciement

Le licenciement est la mesure par laquelle, agissant d'une manière unilatérale, un employeur met fin au contrat de travail qui le lie à un salarié. Constitue une promesse d'embauche valant contrat de travail, l'écrit qui précise l'emploi proposé et la date d'entrée en fonction. La rupture de cet engagement par l'employeur s'analyse en un licenciement sans cause réelle et sérieuse. C'est la lettre de licenciement qui fixe les termes et les limites du litige, de sorte qu'aucune clause du contrat ne peut valablement

décider qu'une circonstance quelconque constituera en elle-même une cause de licenciement. Le Code du travail, subordonne la régularité matérielle du licenciement à l'utilisation d'une procédure à l'usage de laquelle l'employeur doit se conformer. Notamment, préalablement à l'envoi de la lettre de licenciement, il doit convoquer le salarié à un entretien. Cependant, l'absence d'entretien préalable n'a pas pour effet de priver la cause du licenciement de son caractère réel et sérieux.

Les causes d'un licenciement se résument en ses situations irrégulières :

• Absence de 7 jours successive : dans les 48 heures première une mise en demeure est envoyée au salarié, dans le cas où il n'y a pas une réponse de la première mise de demeure et une absence d'une autre 48 heure un avertissement est envoyer au salarié, si les 7 jours écoulé sans réponse le dossier du salarié passe en commission pour décider de son licenciement après attente de justification totale de 15 jours.

• Violation de secret professionnel.

• Détournement ou dissimulation des documents.

• Atteinte à l'autorité de la hiérarchie.

• Sabotage.

• Vol de toute nature.

• Corruption.

• Introduction et consommation des produits illégaux (alcool, la drogue...).

I.5. Conclusion

Ce chapitre a été consacré essentiellement à la présentation de l'organisme d'accueil dans lequel notre projet de fin d'études a été effectué. Aussi mis l'accent sur la présentation du problématique de notre projet.

CHAPITRE II : LE BIG DATA : CONCEPT, DEFIS, ARCHITECTURE, LES TECHNOLOGIES, ET L'AIDE A LA DECISION

II.1 Introduction

Le stockage des grandes masses de données dans les bases de données traditionnelles devient de plus en plus difficile et leur interrogation ou analyse avec les outils classiques devient aussi si lente. D'où la nécessité d'augmenter la capacité de stockage et la vitesse de traitement dans ce type de SGBD, c'est l'une des limites que ce système a atteintes. Et dans le but de passer l'obstacle de vitesse et la performance, de nouvelles technologies, de nouveaux concepts de stockage et d'interrogation de données sont nécessaire à mettre en place.

Dans ce chapitre nous présentons le concept de Big data inclus ses dimensions, caractéristiques et ses défis, ainsi nous présentons le concept de scalabilité et une vision globale sur son architecture, son moyen d'intégration de données (ETL), et à la fin la place de Big Data dans l'informatique décisionnelle.

II.2 Présentation de Big data

Big data est le buzz-word dans les entreprises. Dans nos jours tout le monde le veut et veut sa technologie Sans le Big data, des domaines aussi variées tel que marketing, santé et politique ne garde plus leurs avantages **[1]**.

II.2.1 Définitions

Nombreuses sont les définitions du Big data, elles dépendent de chacun des acteurs du domaine, tel que chacun lui associe une définition qui reflète son point de vue et ses centres d'intérêt envers le phénomène émergent.

Big data est un terme qui décrit l'évolution de tout montant exponentiel et la disponibilité de données structurées, les données semi-structurées et non structurées qui ont le potentiel pour être exploité afin d'avoir des informations spécifiques **[2]**.

Selon M. Lessard, « Big Data » est une expression qui circule depuis quelque temps dans la niche hi-tech de l'informatique dématérialisée (computer in the Cloud) et qui fait

référence aux outils, processus et procédures permettant à une entreprise de créer, manipuler et gérer de très larges quantités de données **[3]**.

Selon O'Reilly2, « Big Data sont des données qui dépassent la capacité de traitement des systèmes de bases de données classiques. Les données sont trop grosses, se déplacent trop rapidement, ou ne correspondent pas au rétrécissement de vos architectures de bases de données. Pour gagner de la valeur à partir de ces données, vous devez choisir une autre façon de traitement. » **[4]**.

David Kellogg, quant à lui, définit simplement les Big Data comme étant « trop grandes pour être raisonnablement traitées par les technologies traditionnelles. » **[5]**.

En résumé de ce qui a été dit concernant les Big Data, « c'est une nouvelle technologie qui est apparue et s'est développée au cours de la dernière décennie exprimant la grande évolution dans le monde Hi-Tech et qui a poussé les chercheurs à mettre à niveau les anciens outils de traitement ». La **figure II.1** nous montre le concept Big data selon IDC.

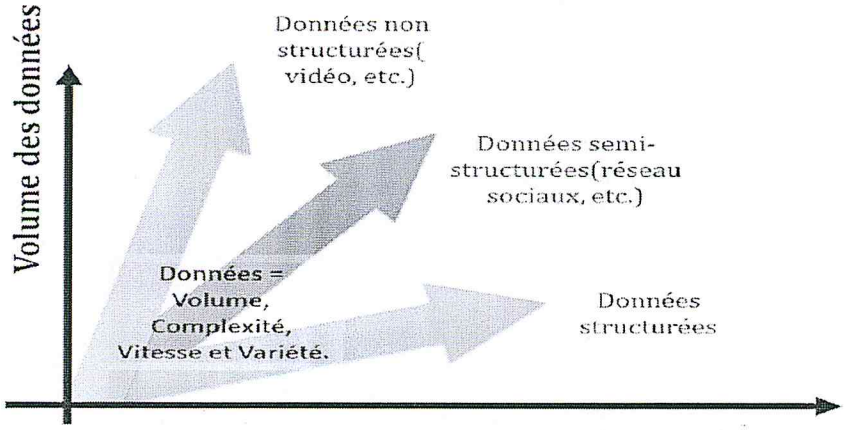


Figure II.1 : Définition du concept « Big data » **[6]**

II.2.2 Dimensions de Big Data

La première définition de Big data a été développé par le Meta group (maintenant partie de Gartner) pour décrire ses trois caractéristiques nommées 3V (Volume, Vitesse, Variété), en se basant sur la qualité, IBM a ajouté une quatrième caractéristique V appelée : Valeur **[7]**.

La convergence de ces quatre dimensions permet à la fois de définir et distinguer les Big Data :

II.2.2.1 Volume

Le Big data est associé à un volume de données vertigineux, se situant actuellement entre quelques dizaines de téraoctets et plusieurs zétaoctets en un seul jeu de données. Les entreprises tous secteurs d'activités confondus, devront trouver des moyens pour gérer l'augmentation du volume de données crée quotidiennement **[8]**.

II.2.2.2 Vitesse

La vitesse décrit la fréquence à laquelle les données sont générées, capturés et partagées. Les entreprises doivent appréhender la vitesse non seulement en termes de création de données, mais aussi sur le plan de leur analyse et de leur restitution à l'utilisateur en respectant les exigences des applications en temps réel **[8]**.

II.2.2.3 Variété

La croissance de la variété des données est très largement la séquence des nouvelles données multi-structurelles et de l'expansion des types de donnée y compris les données structurées, semi-structurées et non structurées **[8]**.

- **Données structurées**

Ce sont les données associées à des bases de données classiques, tels que les transactions relationnelles où l'information est organisée en lignes et en colonnes dans les tables. Presque tous les systèmes de gestion de base de données compris (SGBD) sont conçus pour des données structurelles **[9]**.

- **Données semi-structurées**

Les données semi-structurées sont organisées en entités sémantiques, ce qui les caractérise c'est que les entités similaires sont regroupées, les entités du même groupe n'ont pas forcément les même attributs, l'ordre des attributs n'est pas nécessairement important et les attributs ne sont pas forcément tous nécessaires, de plus la taille ou le type du même attribut du groupe peuvent différer.

Pour être organisé et recherché, les données semi-structurées doivent être fournies par voie électronique à partir des systèmes de bases de données, systèmes de fichiers

(par exemple, les données bibliographiques, des données sur le Web) ou via des formats d'échange de données (par exemple, l'EDI, les données scientifiques, XML) **[9]**.

- **Données non-structurées**

Ce sont les données qui ne suivent aucun format défini, elles peuvent être de tout type, elles ne suivent pas les règles, ne sont pas prévisibles, et peuvent généralement être décrites comme 'Forme libre'. Les données non-structurées sont comme par exemple : (texte, images, vidéo ou son).

Généralement les moteurs de recherches qui récupèrent les données non-structurées **[9]**.

Les organisations nécessitent d'intégrer et d'analyser des données à partir d'un ensemble complexe de deux sources d'information traditionnelles et non traditionnelles, de l'intérieur et de l'extérieur de l'entreprise. Avec l'explosion des capteurs, appareils intelligents et les technologies de collaboration sociale, les données sont générées d'innombrables formes, y compris : texte, données web, tweets, les données des capteurs, audio, vidéo, les fichiers journaux et d'autres.

II.2.2.4 Valeur

Expriment le besoin de la disposition de données pertinentes et significatives, pour donner suffisamment de sens et d'intérêt économique des analyses menées. La valeur recouvre en effet plusieurs spectres nécessitant chacun une analyse spécifique. On parlera ainsi de valeur d'impact sur un contexte, de valeur de modélisation, de valeur de prédiction, de valeur de management, de valeur économique ou de revente...etc **[10]**. Les caractéristiques principales du Big data sont représentée par la **figure I.2** :

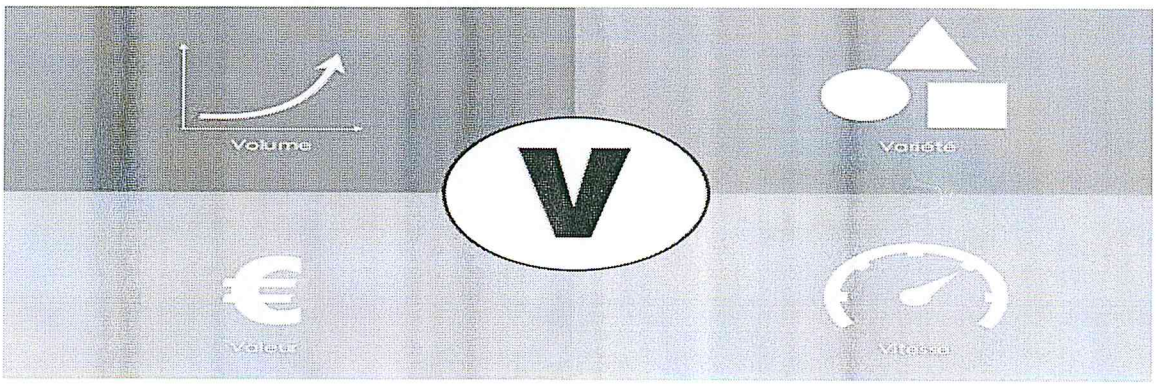


Figure II.2 : Caractéristiques principales du Big data **[10]**.

II.2.3 Caractéristiques de Big Data

Les plateformes des Big data – et pour satisfaire les besoins des entreprises – doivent leur proposer des solutions qui leur sont spécialement conçues, et donc réaliser un certain nombre de caractéristiques. Elles sont les suivantes **[11]** :

- Compréhensive : en se basant sur les trois dimensions, volume, vélocité et variété, elle doit offrir une plateforme large et aborder les défis du Big data.
- Prêt pour l'entreprise (Entreprise-ready) : toute performance, sécurité, facilité d'utilisation et fiabilité doivent être inclus.
- Intégré : elle doit simplifier et accélérer l'intégration des technologies Big data à l'entreprise.
- Faible latence.
- Robuste et tolérant aux pannes.
- Evolutivité.
- Extensible.
- Entretien minimal **[11]**.

II.2.4 Cas d'utilisations de Big Data

L'avantage majeur des technologies Big data est qu'elles sont utilisées dans plusieurs domaines différents, c'est l'une des raisons qui ont amené les entreprises à opter pour cette nouvelle tendance pour satisfaire quelques besoins, et résoudre quelques problèmes dont ce n'était pas facile de le faire avant. Les exemples sont infinis mais nous allons citer quelques-uns, parmi les plus fréquents et les plus demandés :

- **Comprendre le client et personnaliser les services**

Le premier objectif des entreprises est de satisfaire les besoins des clients le maximum possible et en temps réel. C'est parmi les applications du Big data les plus évidentes car en captant et analysant les flux de données sur les clients, les entreprises peuvent améliorer leurs services en fonction de ces besoins, et ceci en intégrant les données classiques « structurées » aux données non-structurées, permettant ainsi et avec la quantité infinie de données collectées d'ouvrir aux entreprises l'opportunité de prévoir les souhaits des clients pour être les premiers à les satisfaire sur le marché **[12].**

- **Optimiser les processus business**

Le Big data peut aussi impacter les processus business complexes tel que le Supply Chain management (SCM), ils seront optimisés en fonction des prévisions issues de l'analyse de données collectées via différents moyens **[12]**.

- **Améliorer la santé et optimiser les performances**

Le Big data permet, par la capture et l'analyse des données concernant le corps humain ; sa santé ou ses activités. Des avancées considérables dans ce domaine tel que le décodage de l'ADN ou la prédiction des épidémies ou la lutte contre les maladies incurables. Il peut également améliorer le domaine du sport et augmenter les chances de gains grâce aux captures vidéo et analyse de données liées aux adversaires du jeu **[12]**.

- **Rendre les machines intelligentes**

Le Big data va permettre aux objets les plus divers d'être plus autonomes donc plus intelligents grâce à la multiplication des capteurs qu'ils intègrent. C'est ce qu'on appelle l'internet des objets (IoT). Parmi les exemples, on trouve les voitures intelligentes qui ont vu naissance grâce aux quantités de données analysées pour optimiser l'expérience de conduite **[12]**.

II.2.5 Domaines d'application

Depuis plusieurs années, les données sont utilisées pour des fins de traitements et d'analyse. Ce qui est nouveau avec le Big data, c'est la quantité de données évolutive exponentiellement, les sources variées de ces données ainsi que le temps réel de traitement. Ceci n'a rien changé dans la nature des résultats trouvés mais a donné l'accès à d'autres domaines d'utilisation de données.

II.2.5.1 Marketing

Dans ce secteur, l'objectif majeur pour les entreprises est la satisfaction de leurs clients en connaissant leurs centres d'intérêts, leurs comportements envers les produits, et encore même leurs achats en magasins. En profitant de ces dernières en tant que clés primaire, les entreprises réussirent à s'approcher aux mieux des clients. Big data a prouvé son succès par rendre le marketing de plus en plus prévisionnel **[12]**.

II.2.5.2 Pilotage d'entreprise

L'innovation qui a porté le Big data à ce domaine a rendu possible l'optimisation complète des ressources et des métiers en assurant la circulation de l'information sur l'activité, ainsi la réduction du temps de réactions face aux erreurs et des couts, et l'ajustement les équilibres offre-demande et temps-ressource. Ce qui n'était pas réalisable avec les indicateurs traditionnels **[12]**.

II.2.5.3 La recherche scientifique

Le Big data a permis des avancées importantes dans ce domaine. Grace au traitement de la masse de données issue de l'exploration de l'infiniment petit, du croisement des données complexes et de nombreuses simulations **[12]**.

II.2.5.4 Le domaine de l'information

Le Big data, en se servant d'une multiplicité de sources d'informations, a réussi la satisfaction de requêtes complexes. Les réseaux sociaux étant la source principale d'informations et de renseignements, vu qu'ils permettent de comprendre les comportements et goûts d'une quelconque population ciblée. Leur second avantage également important, est le rôle qu'ils jouent faire passer des messages au public et mettre ainsi les événements complexes en lumière **[12]**.

II.2.5.5 Les données mobiles

Le Big data doit bénéficier d'un traitement en temps réel pour ce nouveau type de données. Connaitre la géo-localisation des utilisateurs est essentiel, ce qui a ouvert des opportunités de valeurs à plus d'un domaine. Les annonceurs l'ont utilisé pour placer leurs publicités au bon endroit à proximité des clients, les opérateurs télécom ont pu comprendre les flux de population, même le secteur de tourisme a tiré bénéfice de ces données et les a utilisés pour mettre en place ses services dans le bon endroit au bon moment **[12]**.

II.3 Les Défis de Big data

Issues de problèmes de stockages et de bases de données que les experts se sont confronté pendant plusieurs années, les défis les plus présents dans la communauté Big data sont :

II.3.1 Accès rapide

La croissance dans le volume, la variété et la vélocité des Big data a été associé aux besoins de haute performance et d'accès rapide aux données. Plusieurs technologies ont été introduites pour faire face à ce défi, tel que les mémoires flash et les SSD. Egalement, au niveau software de nouvelles architectures de réplication et de redondance. Egalement, au niveau software de nouvelles architectures de réplication et de redondance **[13]**.

II.3.2 Disponibilité de données

La donnée est dite disponible, si pour plusieurs demandes d'accès, la plupart ou toutes les réponses sont réussies. C'est un critère majeur pour les applications Big data où certains services demandent un haut degré de disponibilité pour que les clients aient toujours une réponse satisfaisante à leurs demandes, évitant ainsi la perte financière. Des architectures garantissant la disponibilité de données ont été introduites **[13]**.

II.3.3 Tolérances aux pannes

Souvent les composants d'une infrastructure hardware tombent en panne, le défi du Big data ici est donc assurer la continuité opérationnelle des infrastructures même en présence de composants endommagés. La redondance et la réplication sont les solutions les plus connus qui ont été utilisées pour relever ce défi, cependant ces technologies souffrent d'un problème de scalabilité (évolutivité).

Aujourd'hui les données sont répliquées et réparties entre différentes zones géographiques à l'aide de solutions Cloud, ce qui a renforcé la capacité de récupération plus facilement et à moindre coût **[13]**.

II.4 Le concept de Scalabilité

La « scalabilité » est le terme utilisé pour définir l'aptitude d'un système à maintenir un même niveau de performance face à l'augmentation de charge ou de volumétrie de données, par augmentation des ressources matérielles.

Il y a deux façons de rendre un système extensible : la « scalabilité » horizontale (externe) ainsi que la « scalabilité » verticale (interne). La **figure II.3** présente la « scalabilité » horizontale ainsi que la « scalabilité verticale »

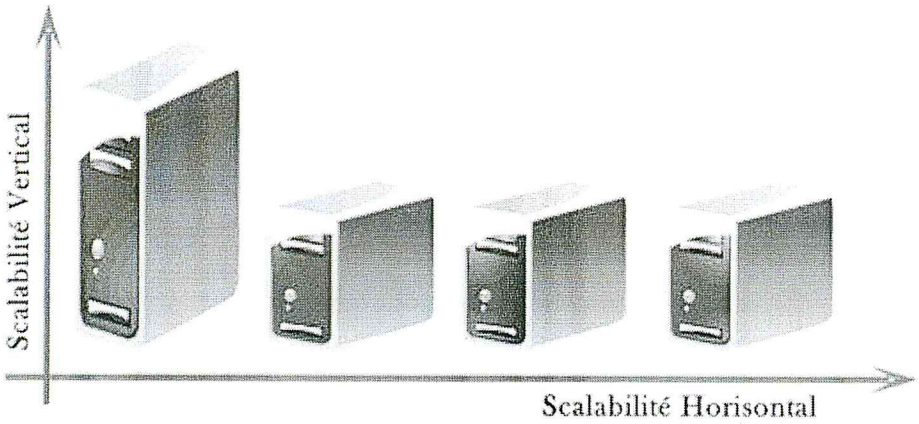


Figure II.3 : « Scalabilité » horizontale (externe) ainsi que la « scalabilité verticale »

II.4.1 Scalabilité Horizontale

Le principe de la « scalabilité » horizontale consiste à simplement rajouter des serveurs en parallèle, c'est la raison pour laquelle on parle de croissance externe. On commence par un serveur basique et on lui rajoute de nouveaux serveurs identiques afin de répondre à l'augmentation de la charge.

Les avantages :

- Achat de serveurs quand le besoin s'en fait sentir. Une panne de serveur ne pénalise pas le système.
- Flexibilité du système.
- Possibilité de mises à jour sans interruption de service.

Les inconvénients :

- Achat de licences pour chaque serveur.
- Difficulté de trouver des serveurs identiques sur le long terme

II.4.2 Scalabilité Verticale

Le principe de la « scalabilité » verticale consiste à modifier les ressources d'un seul serveur, par exemple le remplacement du CPU (Central Process Unit) par un modèle plus puissant ou par l'augmentation de la mémoire RAM (Random Access Memory). C'est ce qu'on appelle la croissance interne. On a une machine et on l'a fait évoluer au cours du temps.

Les avantages :

- Achat d'une seule licence.
- Administration simplifiée du serveur.

Les inconvénients :

- Aucune tolérance à la panne.
- Achat d'une machine coûteuse.
- Le système est limité dans la flexibilité.
- Impossibilité de faire des mises à jour sans interrompre le système.

II.5 Architecture de Big Data

Dans l'environnement du Big data les sources de données sont riches et diverses, elles doivent être stockées et analysées avec une haute qualité de présentation.

II.5.1 L'architecture en couches logique

On trouve dans toutes les architectures Big data les mêmes composantes logiques, La **figure II.4** présente l'architecture physique de Big data

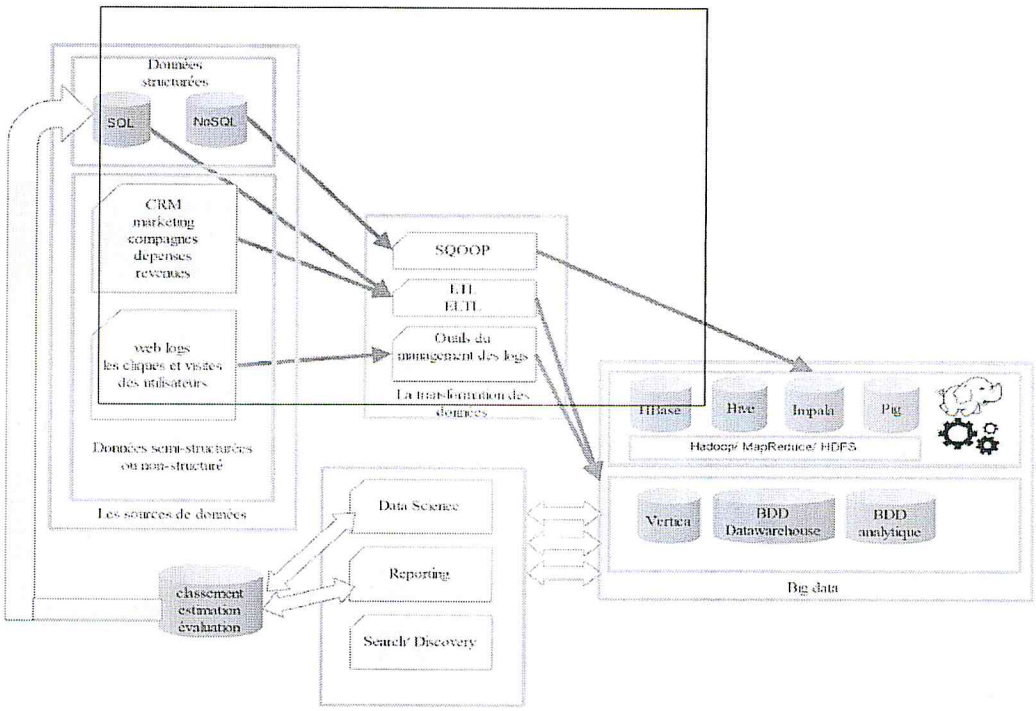


Figure II.4 : Architecture logique de Big data **[15]**.

II.5.2 La source de données

Les données peuvent être structurées, semi-structurées ou non structurées et elles sont récoltées de plusieurs sources. Pour les données structurés, ils sont récupérées depuis SQL, NoSQL, MySQL… etc, les données semi-structurées ou non structurées nous provient des statistique, fichier EXEL et les fichiers log que ce soit les Web log ou les fichiers log local provenant des cliques, des interactions des clients **[14]**.

II.5.3 Transformation des données

La transformation des données c'est le processus de conversion des données d'un format à un autre, (Fichier de base de données, document XML, ou une feuille Excel). Cette étape est nécessaire car les données résident souvent dans des endroits et sous des formats différents, et ont besoin d'être éligibles à d'autres applications et bases de données. Le but donc la transformation de données est de regrouper toutes les données de différentes sources dans un même flux de données à traiter. Dans un scénario typique où l'information doit être partagée, les données sont extraites à partir de l'application source ou d'un entrepôt de données, transformés en un autre format, et ensuite chargé dans l'emplacement cible. Extraction, transformation et chargement sont les processus centraux de l'intégration de données (appelé ETL). SQOOP et les outils de management des logs sont aussi parmi les outils de transformation de données **[14]**.

II.5.4 Intégration des données

Le rôle de l'intégration des données est de concevoir des combinaisons de données structurées avec d'autres non-structurées afin de générer des données utilisables prêtes à servir dans les composantes de consommation de données. Parmi les outils d'intégration des données on retrouve :

- L'écosystème de Hadoop (Hdfs, Mapreduce, Hbase, Pig, Hive ...)
- In-memory analytics : comme Druid et Kognitio, ce sont de nouvelles solutions qui proposent une analyse rapide, leur avantage est qu'ils lisent les données directement de puis le système de fichier HDFS après intégrations dans Hadoop **[15]**.

II.5.5 La consommation de données

Elle permet le traitement et la présentation des données sous formes utilisables par l'utilisateur final en utilisant les outils du Reporting, les outils du search/Discovery ou

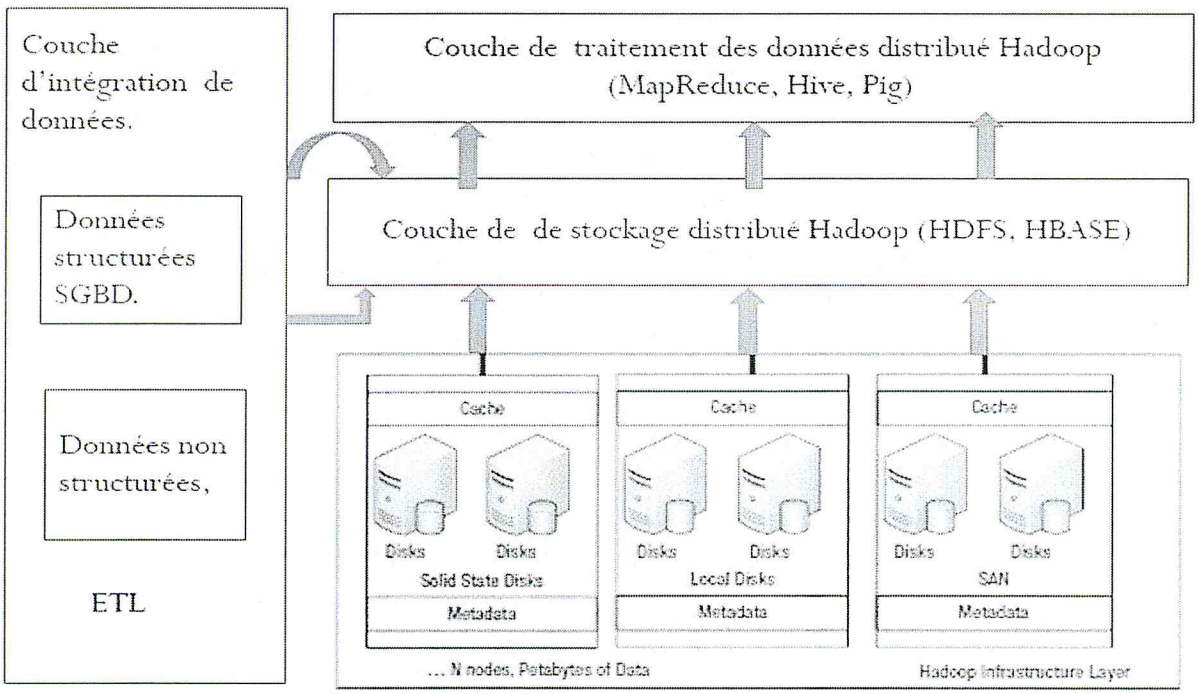
les outils du Data Science (pour le Mining et l'analyse dans des buts de d'estimations et prediction…etc) **[15]**.

II.5.6 L'architecture en couches physiques

La ci-dessous décrit les composants de l'architecture physique qui devraient faire partie de toute solution Big data. Vous pouvez choisir dans chaque couche des Frameworks open source ou des produits packagés. On distingue principalement les couches suivantes **[16]** :

- **Couche matériel** (infrastructure Layer) : peut-être des serveurs virtuels VMware, ou des serveurs lame blade.
- **Couche stockage** (Storage layer) : les données seront stockées soit dans une base NoSQL, ou bien directement dans le système de fichier distribué ou les Datawarehouse
- **Couche management et traitement** : on trouve dans cette couche les outils de traitement et analyse des données comme MapReduce ou Pig.
- **Couche visualisation** : pour la visualisation du résultat du traitement.

La **figue II.5** présente l'architecture physique de Big data :



Figue II.5 : Architecture physique de Big data **[16]**.

II.6 Big Data et les outils ETL (extraction, transformation et chargement)

Certains outils ETL traditionnels comme Talend commencent à s'adapter avec le monde du Big Data.

Les outils ETL sont utilisés pour transformer les données dans le format requis par l'entrepôt de données (datawarehouse). La transformation est effectivement faite dans un endroit intermédiaire avant que les données ne seront chargées dans l'entrepôt de données.

Pour le Big Data, des outils ETL comme Informatica ont été utilisés pour permettre une solution d'ingestion rapide et flexible des données non structurées (supérieure à 150 Go / jour). Informatica PowerCenter peut être utilisée comme un moteur d'ingestion de données brutes primaire **[17]**.

La **Figure II.6** représente un scénario dans lequel un outil ETL traditionnel a été utilisé à ingérer les données dans HDFS (fichier distribuer de hadoop).

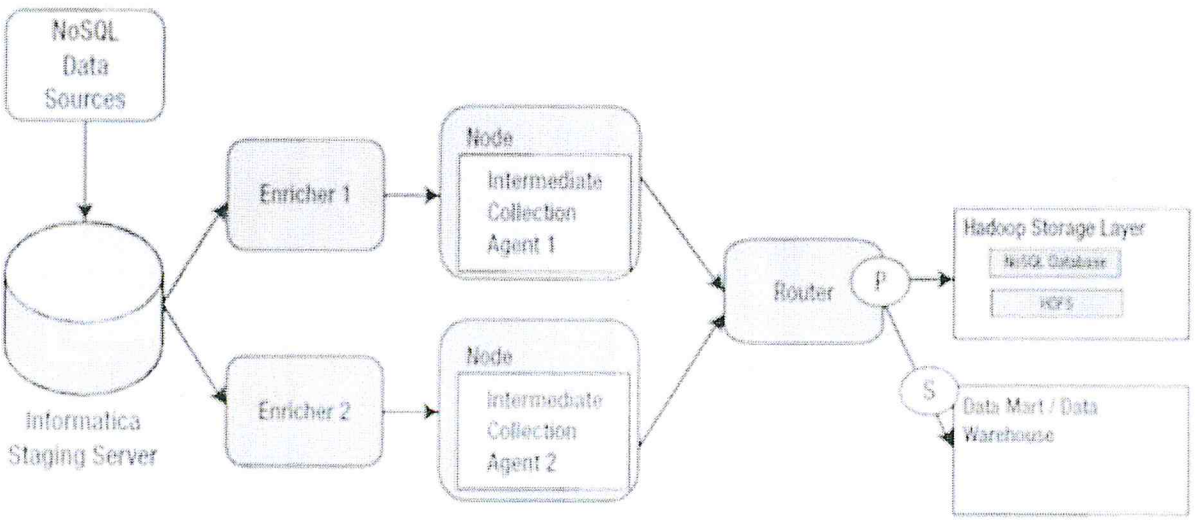


Figure II.6 : Utilisation d'ETL informatica pour Big Data **[17]**.

II.7 Place du Big data dans l'informatique décisionnelle

La **Figure II.7** illustre la place de la technologie Big Data dans le monde de l'informatique décisionnelle.

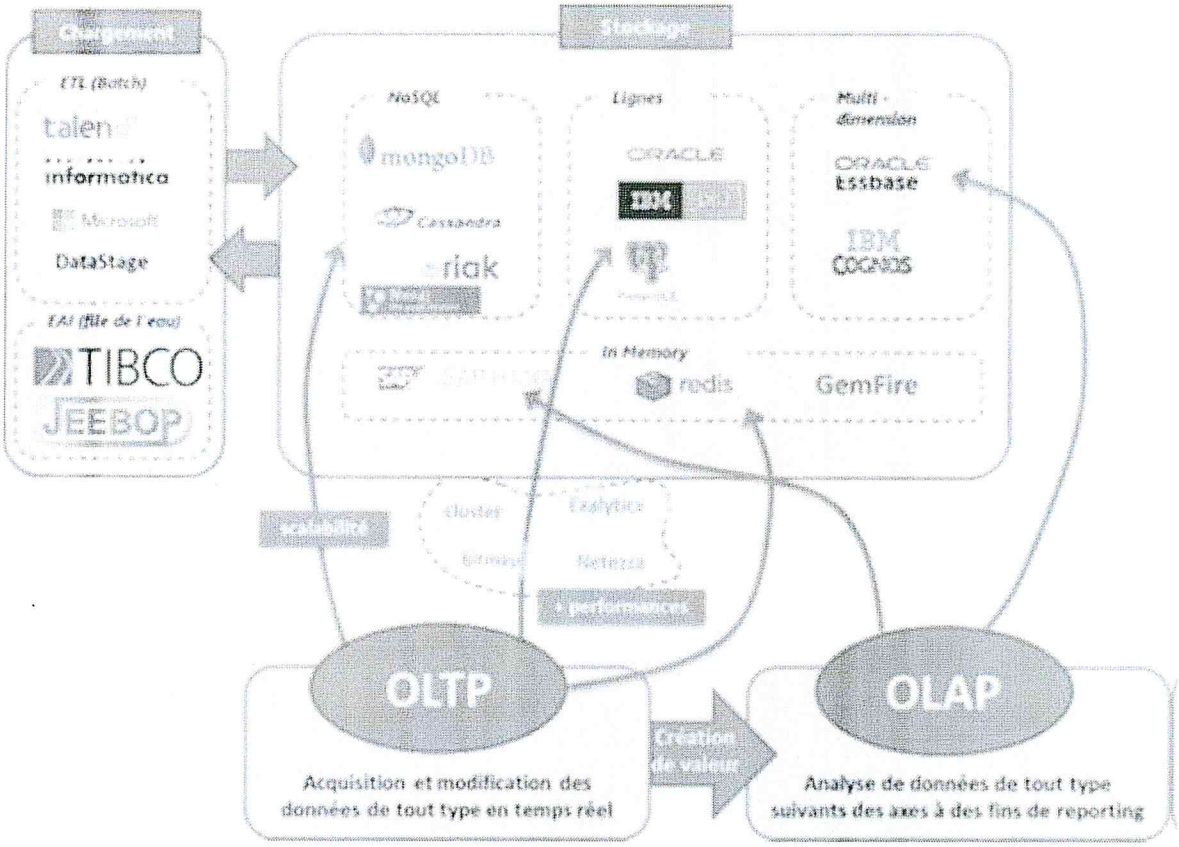


Figure II.7 : place de la technologie Big Data dans le monde de l'informatique décisionnelle.

Comme la **Figure II.7** montre les principaux outils de l'informatique décisionnelle sont :

1. Alimenter la base cible les ETL

En mode batch, on dispose de vrais ateliers de conception permettant de définir graphiquement les flux d'alimentation de la base. A l'image de Talend, ces derniers sont désormais des produits matures et sur les nouvelles applications, il est de plus en plus rare d'écrire des scripts d'alimentation de la base **[17]**.

En mode _temps réel _, la solution à base de web services est devenu un standard de fait. De nombreuses équipes développent des services de ce type afin d'interconnecter les applicatifs entre eux. Cette solution propriétaire peut être normalisée et industrialisée au sein d'un EAI (Jeebop) ou d'un ESB (Tibco, solution IBM). De plus, ces middlewares nouveaux génération ne proposent tout type de

connecteur afin de relier les anciens applicatifs aux nouvelles architectures. Habituellement, les données sont stockées initialement dans une base relationnelle pour des raisons historiques mais aussi car ce modèle se prête bien à l'acquisition des données (respect des critères ACID, bonne montée en charge pour les mises à jour) **[16].**

Le NoSQL est une solution de stockage de données indexées qui contrairement à ses prédécesseurs relationnels de type Oracle, Sybase, MySQL etc., ne répond pas à une définition précise. Il fait clairement table rase des anciens standards définis par les bases de données relationnelles **[17]**.

2. OLTP (Online Transaction Processing)

En informatique et plus particulièrement dans le domaine des bases de données, le traitement transactionnel en ligne (en anglais online transaction processing, abrégé en OLTP) est un type d'application informatique qui sert à exécuter des modifications d'informations en temps réel. Ce type d'application est utilisé dans des activités opérationnelles, typiquement des transactions commerciales (opérations bancaires, achats de biens, billets, réservations). Ce type d'application se connecte à des bases de données en lecture et écriture. Ce type d'application est typiquement opposé au traitement OLAP (pour online analytical processing ou traitement analytique en ligne) **[16]**.

3. OLAP (Online Analytical Processing) : « Refroidir » la donnée

Considérons un ensemble de données décrivant une opération à un instant t (opération bancaire, réservation d'un billet, achat d'un objet). Une fois l'opération validée, les données sont insérées immédiatement dans un ensemble de tables normalisées au préalable lors de la phase de conception (formes NF). Si ces données sont très volumineuses, de l'ordre de quelques dizaines de GB dans un intervalle de temps donné, un partitionnement est défini afin d'optimiser les performances en lecture. Ce type de donnée dit « hot » au début, puis « cold » par la suite est transmis par le système OLTP en partie au système OLAP. Ce modèle ne traite pas les données non structurées, il montre ses limites au-delà d'une certaine volumétrie et surtout, il n'offre aucune souplesse au niveau de son système d'interrogation. En effet, il va falloir écrire de nouvelles requêtes complexes pour chaque nouveau besoin exprimé par le client **[17]**.

3. Big Data (analyser les données)

Vu les limites des modèles traditionnel, le monde du Big data est susceptible de fournir des améliorations afin d'analyser les données à des échelles plus grandes. Le premier apport est sa capacité à analyser des données non structurées grâce à son mécanisme Map/Reduce (Hadoop). Ce dernier reposant son calcul sur un très grand nombre de nœuds (probablement plusieurs dizaines de milliers chez Google), il distribue la puissance nécessaire à l'exploitation de la donnée libre. Cela permet d'injecter de nouvelles informations non traitées dans les systèmes actuels **[17].**

II.8 Conclusion

Après avoir étudié dans le **chapitre II** le nouveau concept Big Data qui apporte une architecture distribuée, et dont le but est d'améliorer la performance, la vitesse d'exécution et de traitement des requêtes et le stockage de données structurées ou non structurées nous abordons dans l'annexe les aspects théoriques du Framework Hadoop et ses composants présentant le principal outil de Big Data.

CHAPITRE III : LES DIFFERENTES SOLUTIONS RELATIVES AUX APPLICATIONS DE TRAITEMENT ET D'ANALYSE DU BIG DATA (BDA APPS)

III.1. Introduction

Comme c'est déjà mentionner dans le chapitre précédent, la quantité de données dans le monde augmente de façon exponentielle au fil de temps, ce qui oblige les entreprises à dépenser beaucoup d'efforts et d'argents sur la collecte et le stockage de données dont la majorité sont mal traité et analysé. L'analyse de donnes est en plein accroissement, elle réinvente le monde, elle permet de prévoir le futur et produire aux mieux les connaissances exploitables et actionnables.

Dans la majorité des entreprises, les dirigeants se demandent s'ils accèdent à la valeur totale d'informations issues des quantités massives de données qu'ils collectent dans leurs organisations en utilisant les nouvelles technologies de collecte et de sauvegarde. Ces organisations sont toujours à la recherche de nouvelles méthodes et technologies pour analyser leurs données et en tirer le maximum de bénéfices **[41].**

Toutefois, l'extraction de connaissances utiles à partir d'énormes ensembles de données numériques nécessite des services intelligents et évolutifs de traitement et d'analyse, des outils de programmation et des applications. Ces moyens sont connus sous l'appellation « analyse du Big data » ou 'Big data analytics' **[42]**.

L'analyse du Big data est en plein accroissement, elle réinvente le monde, elle le gère, le dirige et le fait fonctionner plus efficacement, elle permet de découvrir les tendances cachées, les corrélations inconnues et d'autres informations utiles à partir de grande quantité de données de types variés. D'où elle nécessite une infrastructure parallèle et hautement évolutive ainsi que des techniques avancées avec leurs outils associés **[43].**

Ce chapitre vise à décrire le paradigme de l'analyse du Big data, ses défis, ainsi à présenter les phases de cette analyse et à étudier les différentes applications et outils aident à extraire les informations de ces données numériques massives, et à la fin il vise découvrir les différentes méthodes et techniques utilisées dans le traitement et l'analyse du Big data

III.2. Aspect de traitement

Dans le cycle de traitement de l'information, où les données sont acquises, saisies, validées, traitées, stockées et sorties, le traitement de données n'est qu'une étape distincte qui organise l'information de façon à former la sortie désirée.

Le traitement des données relatives au Big data, consiste à diviser un problème en plusieurs plus petites opérations, de les résoudre et de les combiner en un seul résultat.
Même avec les progrès technologiques importants dans les systèmes traditionnels, les technologies de traitement de Big data ont changé les règles du jeu de toute forme et taille **[44]**. Ces nombreuses méthodes de traitement et d'analyse partagent les mêmes caractéristiques et bénéficient d'un matériel qui leur permet de garantir le Scale-out (montée en puissance parallèle), d'employer les capacités de stockage non-relationnelles pour manipuler les données non et semi-structurées **[45].**

Jusqu'à présent la solution la plus notable pour traiter et gérer les Big data est le MapReduce Framework. Depuis sa création, le MapReduce garantit le traitement d'un grand volume de données par lots, il a rendu simple et efficace le traitement de données complexes et à grand échelle. Cependant, il n'est pas adapté aux nouvelles nécessités de traitement en temps réel et le traitement en ligne. Il offre trois caractéristiques principales à savoir : modèle de programmation simple et efficace, une évolutivité automatique et linéaire et une tolérance aux pannes intégrée **[46]**.

III.2.1. Traitement des Big data

Dans le traitement de données en temps réel, les données sont chargées, traitées et ensuite sorties de façon continue. Ce type de traitement doit être réalisé en petites périodes de temps précis (en temps réel).

Les solutions sont classifiées en deux catégories : une première qui vise l'exécution des tâches MapReduce en quelques secondes ou moins, connue sous l'appellation 'calcul en-mémoire' en anglais 'in-memory computing'. Une seconde, qui s'appuie sur de nouvelles approches optimisées pour l'exécution de requêtes en temps réel sur les Big data structurées et non-structurées, appelée 'requêtes en temps réel sur les Big data'.

III.2.1.1 Calcul en-mémoire

Les deux principales raisons de la lenteur d'Hadoop sont dues d'une part, au fait qu'initialement Hadoop n'a pas été conçu pour l'exécution rapide des tâches. D'autre part, HDFS (son système de fichiers distribués) est seulement optimisé pour les données à haut débit d'E/S.

Afin de résoudre ces problèmes, une des solutions proposées pour le premier problème suggère la refonte des modules de démarrage et d'exécution des tâches. Cependant, pour le second problème c'est moins évident car celui-ci est causé par le matériel.

Pour toutes ces raisons, le calcul en-mémoire a été nommé la solution la plus élégante à ces problèmes. Il se base sur le stockage et le traitement du Big data en temps réel et sur l'utilisation d'un système de mémoire principale distribué.

Cette solution de calcul se caractérise par des performances supérieures en matière de bande passante élevée et de latence d'accès, ce qui en fait une alternative prometteuse par rapport à la méthode d'analyse de Big data basée sur disque. Les solutions disponibles de calcul en-mémoire consiste en : Spark d'Apache seule solution gratuite et Open source, GridGain, et XAP **[46]**.

III.2.1.2 Requêtes en temps réel sur les Big data

Les solutions qui tentent de permettre les requêtes en temps réel sur les Big data sont nombreuses, et chacune d'entre elles possède ses propres caractéristiques. La première qui a été lancée dans ce domaine est le fruit des travaux du

Un géant du Web 'Dremel'. Il utilise pour le traitement des requêtes complexes en temps réel, deux techniques majeures : un nouveau type de stockage des structures imbriquées sous forme de colonne, et un algorithme d'agrégation évolutif pour le calcul des résultats des requêtes en parallèle.

En essayant de concevoir une implémentation Open source de 'Dremel', 'Cloudera' a fait sortir son produit 'Impala' et a développé 'Paquet' un stockage binaire en colonne efficace pour Hadoop utilisant les techniques des SGBDs parallèles pour les requêtes en temps réel.

Apache a aussi essayé de reproduire le travail de Google et a lancé sa solution Apache Drill qui n'est pas seulement adaptée à Hadoop mais fournit les requêtes en temps réel à d'autres systèmes de stockage tel que Cassandra.

Une autre solution a été développée pour être compatible avec Hive et pour exécuter ses requêtes, Shark qui est basés sur Spark en utilisant les capacités de calcul en-mémoire et les moteurs de recherche rapide de Spark, qui revendique un temps de réponse jusqu'à 100 fois plus rapide par rapport à Hive **[46]**.

III.2.2. Traitement des flux de Big data

Les applications d'aujourd'hui nécessitent beaucoup plus de demandes en flux dans lesquelles les données d'entrées arrivent constamment, ceci a été appelé 'le traitement des flux'. Malheureusement ce type de traitement n'est pas supporté par le Framework MapReduce et ses implémentations comme Hadoop. Afin de remédier à cela, une nouvelle technique a été adoptée par MapReduce pour gérer les flux partiellement. L'idée consiste à

traiter le flux comme une suite de petits morceaux de lots de données. Cependant cette technique n'est pas adaptée pour les besoins d'un véritable système de flux.

Récemment des Framework conçu spécialement pour ce type de traitement, ont vu le jour pour traiter le flux des Big data. Nous allons présenter les plus populaires entre eux et qui sont : Storm de Twitter et S4 de Yahoo. Ces derniers partagent quelques points en commun ; par exemple ils s'exécutent tous les deux sur la machine virtuelle de JAVA (JVM). Ils différent dans leur modèle de programmation ; dans S4 un programme est défini en fonction d'un graphe d'éléments de traitement PE (Processing Elements), il crée une occurrence PE pour chaque clé. Quand à Storm son programme est défini par deux abstractions : Spouts et Bolts. Un Spout est la source de flux, il peut lire les données à partir d'une file d'attente d'entrée ou même les générer lui-même. Un Bolt traite un ou plusieurs flux d'entrée et génère un nombre de flux en sortie, ce graphe de Spouts et Bolts est appelé 'Topologie'.

Dans n'importe quel niveau de traitement dans S4, si le buffer d'entrée des PE atteint ses limites de chargement, les messages qui arrivent après seront supprimés. Afin de garantir sa tolérance aux pannes, S4 utilise une stratégie de points de reprise ; si un noeud se bloque, ses PE seront renouvelés dans un autre noeud à partir de leur dernier état sauvegardé, d'où les traitements effectués après le dernier état seront perdus.

Néanmoins, Storm garantit le traitement de chaque tuple entrant, et si le tuple n'arrive pas à traverser la topologie de Storm dans une période prédéfinie, le Spout qui a généré ce tuple le réinitialise. Ceci donne à Storm un point fort face à S4 au niveau de la tolérance aux pannes.

Bien que tous les deux ont des points forts et des points faibles, Storm est noté le meilleur et possède une plus large communauté d'utilisateurs. En résumé, vue les demandes et les applications, le traitement de flux des Big data va certainement évoluer dans le futur **[46]**.

En conclusion, nous dirons que les solutions de traitement de données par lots (Batch processing), tel que Hadoop, ont atteint un niveau de maturité acceptable, sauf que malheureusement elles ne satisfont pas les exigences du traitement non-Batch. Les flux des Big data et les demandes des requêtes interactives ont incité à développer de nouvelles solutions, comme le calcul en-mémoire qui est considéré comme une solution intéressante pour gérer à la fois les exigences du temps réel et des flux. Parmi les Framework utilisés pour le calcul en-mémoire, Spark permet les requêtes interactives et en temps réel en utilisant Shark et le traitement des flux en utilisant le micro-dosage rapide, or l'avenir nous dira quelle approche sera populaire dans le futur.

III.3. Aspect d'analyse

III.3.1. Définition de l'analyse de Big data

Nombreuses sont les définitions de l'analyse du Big data, elles dépendent de chacun des acteurs du domaine, tel que chacun lui associe une définition qui reflété son point de vue et ses centres d'intérêt envers le phénomène émergent.

Selon un rapport TDWI, L'analyse du Big est en plein accroissement, elle consiste en un ensemble de techniques analytiques avancées qui opèrent sur de grands volumes de données structurés, semi-structurées, et non structurées issues de différentes sources **[49].**

L'analyse du Big data est lié aux processus d'utilisation des techniques et outils d'analyse avancés relatifs aux grands ensembles de données dont le type est différent. Il distingue les formats structurés / non-structurés, le streaming/lot, et dont la taille varie entre Terabytes et zettabytes **[50]**.

L'analyse de gros volumes de données permet aux analystes, les chercheurs et les utilisateurs professionnels de prendre des meilleures décisions et plus rapides à l'aide de données qui étaient auparavant inaccessibles ou inutilisables. En utilisant des techniques d'analyse avancées, telles que l'analyse de texte, l'apprentissage automatique, l'analyse prédictive, l'exploration de données, les statistiques et le traitement du langage naturel, les entreprises peuvent analyser les sources de données jusque-là inexploitées indépendante ou conjointement avec leurs données d'entreprise existantes pour acquérir de nouvelles connaissances résultant en une meilleure et plus rapide les décisions **[50]**.

L'analyse du Big data en temps réel, est la possibilité de prendre les meilleures décisions et les mesures significatives, au bon moment. Comme la détection de la fraude alors que quelqu'un est en train de glisser une carte de crédit, ou le déclenchement d'une offre alors qu'un client se tient debout sur une ligne de commande. Il s'agit de combiner et analyser les données afin que vous puissiez prendre les bonnes décisions au bon moment et au bon endroit **[51].**

Big analyse de données est le processus d'examen de grands ensembles de données contenant une variété de types de données - à savoir, Big Data - pour découvrir des tendances cachées, des corrélations inconnues, les tendances du marché, les préférences des clients et d'autres informations commerciales utiles. Les résultats d'analyse peuvent conduire à une commercialisation plus efficace, de nouvelles opportunités de revenus, un meilleur service à la clientèle, l'amélioration de l'efficacité opérationnelle, des avantages concurrentiels sur les organisations rivales et d'autres avantages commerciaux **[56].**

De notre point de vue, l'analyse des Big data consiste en l'analyse des données massives non-structurées en utilisant des moyens et systèmes plus avancés pour leur traitement, face à l'échec d'outils et techniques traditionnelles. L'objectif étant d'améliorer la prise de décisions dans les domaines divers et variés, technologiques, économiques et scientifiques.

III.3.2. Défis de l'analyse des Big Data

Les défis du Big data qui sous-tendent les différentes phases de son analyse, sont identifiés dans ce qui suit :

III.3.2.1. Hétérogénéité et incomplétude

Lorsque la consommation des informations est réalisée par les êtres-humains, une grande hétérogénéité est tolérée. Car la nuance et la richesse du langage naturel nécessite une certaine profondeur utile. Toute fois les algorithmes d'analyse des machines s'attendent à des données homogènes et ne peuvent pas comprendre la nuance. Par conséquent, les données, avant qu'elles ne soient analysées, elles doivent être soigneusement structurées et organisées.

Cependant, les modèles les moins structurés sont les plus riches en informations, d'un autre côté, les machines stockent les données identiques en taille et en structure, d'une manière plus efficace. En conséquence la représentation, l'accès et l'analyse des données semi-structurés nécessitent un travail plus profond.

Durant la phase de l'analyse, et même après le nettoyage et la corrélation des données, certaines incomplétudes et erreurs peuvent se produire. L'un des défis de l'analyse des Big data est de gérer ces erreurs et incomplétudes correctement.

III.3.2.2. Évolutivité

Gérer des données volumineuses de croissance rapide a été un défi pour les chercheurs durant de nombreuses années. Auparavant, ce défi a été atténué par la rapidité des processeurs qui fournissent les ressources nécessaires pour la gestion de ce grand volume de données, mais le problème qui persiste actuellement est que le volume de données est de croissance rapide, alors que les ressources matérielles et la vitesse du CPU sont statiques. Ces changements ont obligé les chercheurs à repenser la façon de concevoir, de construire et d'exploiter les composants qui interviennent dans le traitement des données.

III.3.2.3. Opportunité

Dans le monde technologique, la rapidité est à l'opposé du volume. Il existe plusieurs cas dans lesquels les résultats d'analyse sont requis immédiatement, tel que pour la détection de la fraude ou l'analyse est nécessaire avant la fin de la transaction. Evidemment pour faire ce

type d'opération, une analyse complète est souhaitée contenant l'historique des transactions réalisées par le client, ce n'est pas faisable surtout quand il s'agit d'une analyse en temps réel.

Généralement, pour trouver des réponses à des situations critiques, il est nécessaire d'analyser un grand nombre de données. En revanche balayer la totalité des données à chaque fois pour trouver la réponse adéquate à une requête n'est évidemment pas pratique. La solution est d'organiser ces données sous forme de structure d'index, néanmoins chaque structure supporte un certain nombre de classe de critères spécifié. Avec l'analyse désirée du Big data de nouvelles classes de critères sont apparues et nécessitent de nouvelles structures pouvant les supporter. La conception de ces structures devient un défi, lorsque le volume de données augmente rapidement et les réponses aux requêtes sont limitées dans le temps.

III.3.2.4. Confidentialité

La confidentialité des données est une préoccupation énorme, et qui augmente avec le Big data. Malgré les règles strictes mises en place, des questions restent posées concernant l'utilisation inappropriée des informations personnelles prévenant des différentes sources. La gestion de la confidentialité persiste comme problème prioritaire à l'échelle technique et sociologique et nécessite plus d'attention pour réaliser les promesses du Big data.

III.3.3 Les phases d'analyse du Big data

L'acquisition et l'enregistrement des données, l'extraction et le nettoyage des informations, l'intégration, l'agrégation et la représentation des données, le traitement des requêtes modélisation et analyse de données, et l'interprétation sont les cinq phases de l'analyse :

III.3.3.1. Acquisition et enregistrement des données

Les données liées au Big data sont générées et enregistrées à partir de plusieurs sources avec un nombre croissant pouvant arriver jusqu'à 1 million de téraoctets de données brutes par jour. Une grande partie de ces données peut ne pas être utile, pour cela, les données doivent être filtrées et comprimées par des ordres de grandeur. Il reste important de définir les filtres de données d'une manière à ne pas rejeter les informations utiles, pour cela il est nécessaire de faire appel à la recherche scientifique pour la réduction et le traitement des données brutes intelligemment. En outre, les techniques d'analyse en ligne s'imposent vu que les gros volumes de données ne peuvent pas être stockés pour être réduits par la suite

III.3.3.2. Extraction et analyse des informations

La majorité des données collectées aujourd'hui ne sont pas sous un format prêt à être analyser, pour cette raison, il faut un procédé d'extraction d'information permettant de sortir l'information des sources et de l'exprimer sous forme structurée et adaptée à l'analyse. D'autant plus que les données récentes sont dans la majorité du temps sous forme d'images ou même de vidéos el leur extraction dépend des applications.

III.3.3.3. Intégration, agrégation et représentation de donnée

L'analyse des données ne se résume pas dans le fait de localiser, d'identifier, de comprendre et de citer les données. Pour qu'elles soient efficaces et à grand échelle, toutes ces étapes doivent être automatisées, nécessitant ainsi des structures de données compréhensible par les machines et solvables par les robots.

III.3.3.4 Traitement des requêtes, modélisation et analyse de donnée

L'extraction de données exige des données intégrées, nettoyées, fiables, et efficacement accessibles, des interfaces de requêtes déclaratives, des algorithmes d'exploration évolutifs et des environnements de calcul des Big data.

L'actuelle problème de l'analyse de Big data réside dans le manque de coordination entre les systèmes de bases de données qui accueillent les données et fourni les interrogations SQL, et les paquets d'analyse qui exécutent diverses formes de traitement NoSQL. Les analystes d'aujourd'hui trouvent des difficultés avec cela et le considèrent comme un obstacle à la réalisation interactive de la première génération des systèmes OLAP dans les types d'analyse datamining qui est en demande croissante.

Dans l'avenir, les requêtes vers le Big data seront automatiquement générées pour la création du contenu des sites web, pour remplir les recommandations et pour fournir une analyse ad hoc sur un ensemble de données, de même le temps de réponse interactif qui est considéré comme un problème majeur aujourd'hui sera permis

III.3.3.5 Interprétation

L'interprétation des résultats de l'analyse inclut l'examen de toutes les hypothèses formulées et le retracement de l'analyse. En bref, il est rarement suffisant de fournir seulement des résultats. Au contraire, il faut fournir des informations supplémentaires qui expliquent comment un résultat a été dérivé, précisément basé sur les entrées. L'utilisateur devra être capable de naviguer dans chaque partie de données qu'il voit et de comprendre sa provenance, il doit aussi être en mesure de voir non seulement les résultats, mais aussi de

comprendre pourquoi il voit ces résultats. Accomplir cela nécessite que le système fournisse des outils pratiques pour l'utilisateur pour spécifier des analyses. La spécification déclarative est un composant d'un tel système.

III.3.4. Exemples d'outils

Ci-dessous quelques outils qui ont été utilisé avec l'analyse des grands volumes de données :

III.3.4.1. Apache Mahout

Machine Learning et datamining plateforme pour la classification, le regroupement et la recommandation. Mahout, mot indien qui fait référence à un pilote d'éléphant, a vu le jour en 2008 comme étant un sous projet du Licence d'Apache, qui fournit le moteur de recherche open-source connu sous le même nom **[52]**.

Sa mise en œuvre visait à priori l'expansion des algorithmes d'analyse de données. Non seulement sur Apache Hadoop, mais il peut être utilisé dans une variété d'environnements. Il propose également des forfaits efficaces en termes de stockage de la bibliothèque mathématique et de la collection java **[47].**

III.3.4.2. GNU R

C'est un environnement logiciel comprenant des langages de programmation spécialisés pour l'analyse des statistiques, des graphiques (visualisation) et des packages. Il assure un bon déroulement du processus de données vectorielles et matricielles afin d'être optimisé pour les calculs statistiques en matière de langue.

III.3.4.3. Apache Pig

C'est une plateforme d'analyse de grands ensembles de données, qui se compose d'un langage de haut niveau pour l'expression de programmes d'analyse de données. (Apache, 2013). Il supporte un langage appelé Pig Latin, et présente les caractéristiques suivantes : **[47].**

✓ **Facilité de programmation :** Les tâches complexes composées de plusieurs transformations de données connexes, sont explicitement codées comme des séquences de flux de données, ce qui les rend faciles à écrire, à comprendre et à maintenir.

✓ **Possibilités d'optimisation :** La manière dont les tâches sont codées, permet au système d'optimiser l'exécution automatiquement, ce qui permet à l'utilisateur de se concentrer sur la sémantique plutôt que l'efficacité.

✓ **Extensibilité :** Les utilisateurs peuvent créer leurs propres fonctions pour faire du traitement à des fins spéciales **[53].**

III.3.4.4. Apache Hive

Une solution open-source d'entreposage de données construite sur Hadoop. Apache Hive permet d'analyser les grands volumes de données stockées dans des environnements distribués, en utilisant un langage déclaratif de type SQL. Il s'agit de HiveQL, ce langage supporte les scripts Map-Reduce personnalisés pour être branchés dans des requêtes **[54]**.

Hive consiste en deux composantes nommés Hcatalog et WebHcat. Hcatalog est une table qui permet aux utilisateurs d'écrire les données sur la grille de manière plus facile. Quant à WebHcat fournit un service d'exécution de MapReduce d'Hadoop, Pig, Hive en utilisant une interface web **[55].**

III.4. Conclusion

Les Big Data permettent d'accéder à de nouvelles opportunités d'affaire et de mieux contrôler les risques, mais engendrent en contrepartie une évolution considérable des modèles technologiques de l'entreprise. Les technologies qui permettent de traiter analyser et archiver ces données. Posant ainsi des défis considérables concernant leur manipulation. Ces derniers consistent en les nouvelles technologies complexes à appréhender.

Dans ce chapitre, nous avons présenté les méthodes d'analyse les plus connues et les plus utilisées par les organisations.

CHAPITRE IV : CONCEPTION ET IMPLEMENTATION TECHNIQUE DE L'ARCHITECTURE

IV.1. Introduction

Dans ce chapitre, nous allons aborder la partie pratique de notre projet qui consiste à implémenter l'architecture proposée, décrire la conception technique et la mise en œuvre complète de l'environnement distribué Hadoop. Nous définirons les différentes étapes d'installation et de configuration de notre cluster Hadoop ainsi que les différents tests effectués sur des tâches MapReduce

IV.2. L'architecture utilisée

Notre travail s'insère dans une vision à long terme et débute par une approche légère qui consiste à mettre en place un outil de base de données qui prétraite des faibles quantités des données. A l'aide de Talend, un outil ETL « Extraction, Transformation, Load », les données semi-structurées prétraitées sont intégrées dans une plateforme hadoop. Par contre les données semi-structurés (fichier proxy) qui ne peuvent pas être traitées, sont incorporées directement dans la plateforme, Hdfs étant son système de stockage distribué sur plusieurs machines (ce qui forme un cluster). Après que les données soient stockées, des jobs mapreduce sont lancés sur ces volumes de données pour avoir des analyses qui seront interprétées. La **figure IV.1** présente l'architecture utilisée :

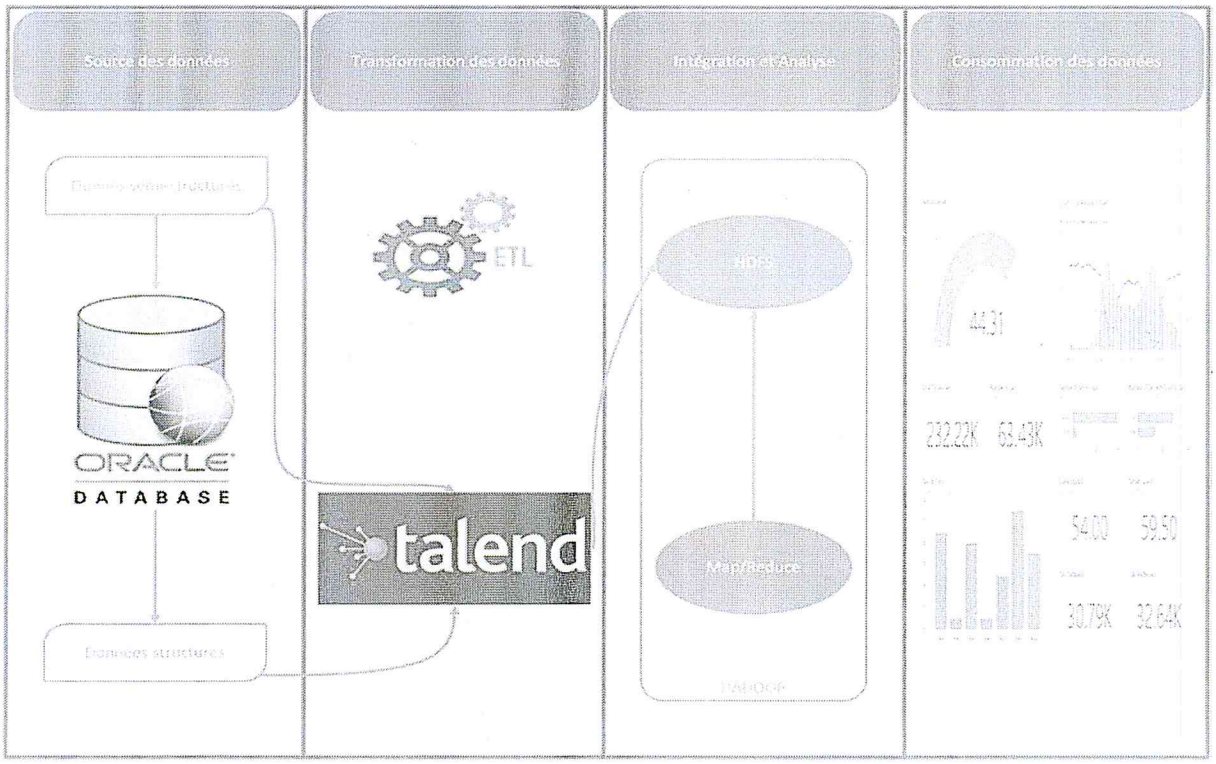


Figure IV.1 : Architecture utilisé

Pour l'implémentation, nous allons utiliser une architecture distribuée à plusieurs nœuds, autrement dit un cluster de deux machines :

Le **Master** qui est responsable de la distribution, de la réplication des blocs et l'Assignement des tâches aux Slaves, et le **Slave** qui Stocke des blocs de données et exécute les tâches données par le Master. La **figure IV.2** présente l'architecture distribuée à plusieurs nœuds :

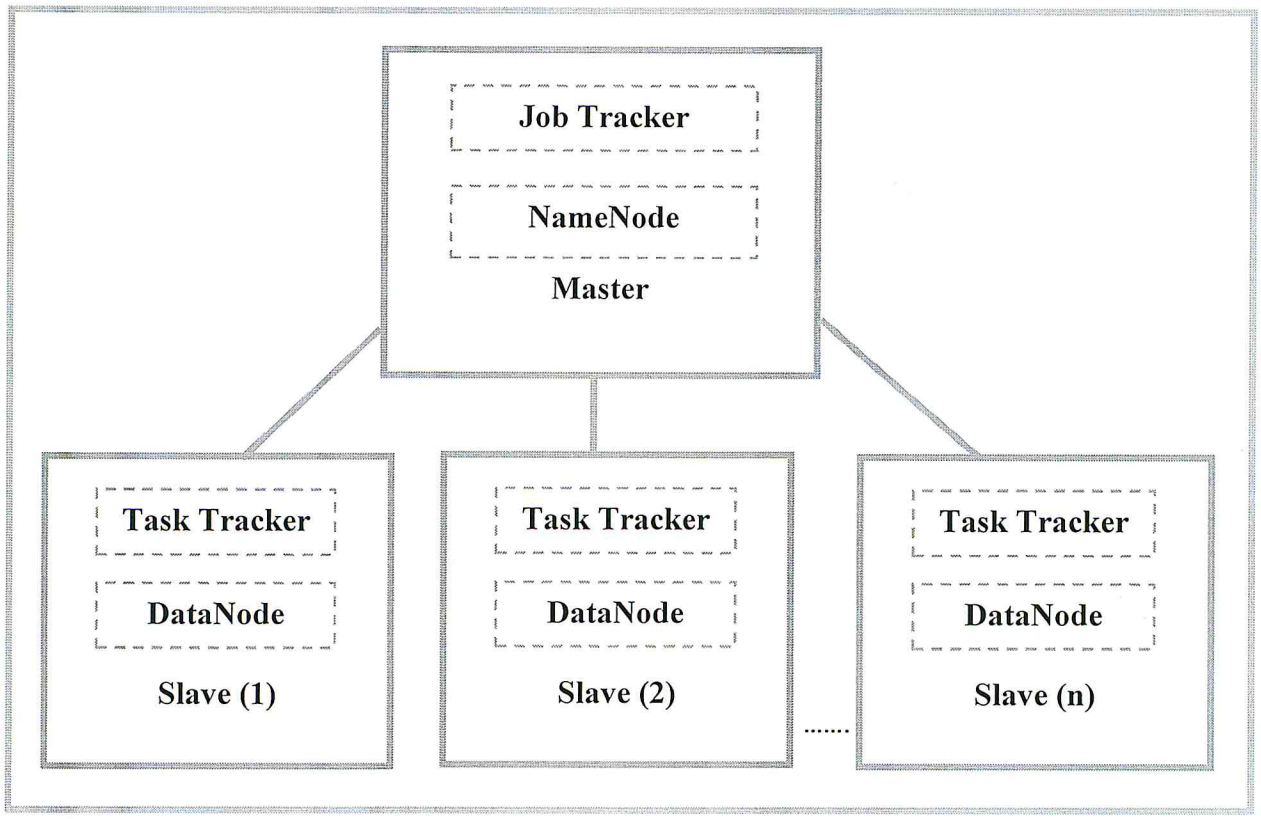


Figure IV.2 : Architecture distribuée à plusieurs nœuds

IV.3. Description de la solution

Pour tester notre système Hadoop, on a proposé comme solution d'exécuter des jobs en s'appuyant sur le modèle MapReduce écrit en java :

- Les jobs de départ en retraite.
- Les jobs de départ potentiel.

Le stockage des données est effectué dans des fichiers distribués d'Hadoop HDFS.

Pour la réalisation, nous avons procédé à :

✓ L'utilisation de deux machines virtuelles (VM) avec linux Centos 6.6 comme un système d'exploitation.

✓ L'installation et la configuration de Hadoop à partir de la distribution CDH (Cloudera's Distribution Hadoop). Cette distribution contient les principaux composants de Hadoop (HDFS, MapReduce)

✓ La création et la configuration un job Talend pour transformer et charger les données semi-structurées dans des tables Oracle.

✓ La création d'un volume HDFS et la redirection nos fichier proxy et table de base données crée via Talend vers le volume HDFS.

✓ L'implémentation de l'algorithme de MapReduce en java avec Eclipse qui contient les API d'Apache Hadoop.

IV.4. L'environnement de travail

Dans la section suivante, on va décrire les outils et systèmes composant notre environnement de travail.

IV.4.1. VMware Workstation 11

VMware est un logiciel qui permet la création d'une ou plusieurs machines virtuelles simulant plusieurs systèmes d'exploitation x86 au sein d'un même système d'exploitation. Celles-ci, pouvant être reliées au réseau local avec des adresses IP différentes, tout en étant sur la même machine physique qui existe réellement. Il est possible de faire fonctionner plusieurs machines virtuelles en même temps. Workstation maximise les ressources de l'ordinateur de façon à permettre l'exécution des applications les plus gourmandes dans un environnement virtuel. Il permet aussi de construire des machines virtuelles complexes pour l'exécution des applications Big Data sous la forme d'un cluster Apache Hadoop.

La **figure IV.3** présente VMware Workstation 11 :

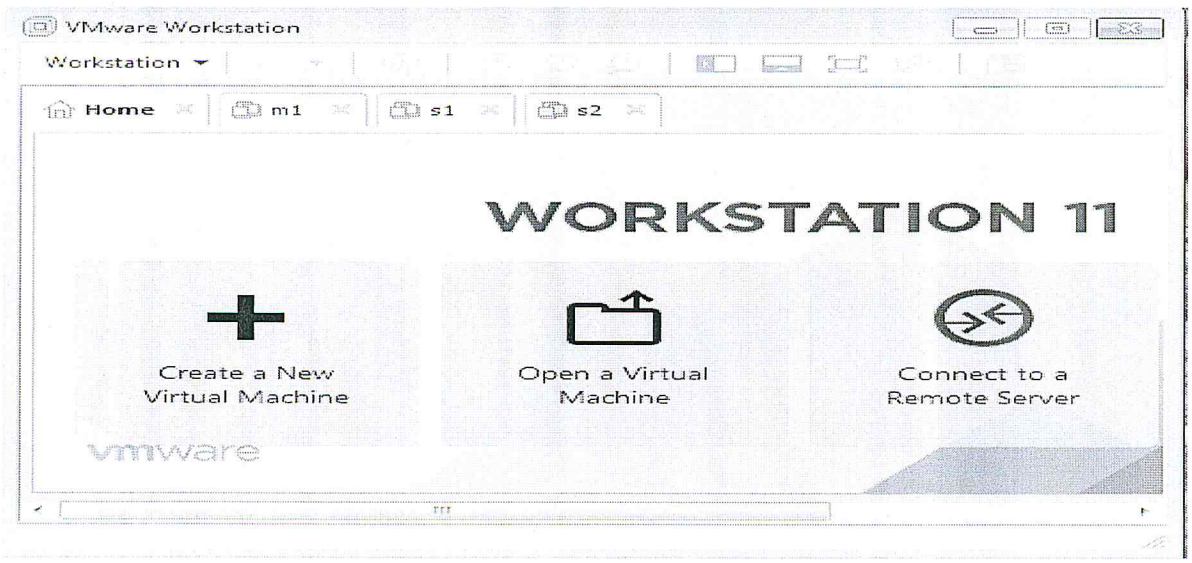


Figure IV.3 : VMware Workstation 11

IV.4.2. Linux CentOs-6.6

CentOS (Community entreprise Operating System) est une distribution GNU/Linux principalement destinées aux serveurs. Tous ses paquets, à l'exception du logo, sont des paquets compilés à partir des sources de la distribution RHEL (Red Hat Entreprise Linux), éditée par la société Hat. On peut télécharger CentOs sous la forme de DVD OU CD **[64]**.

Figure IV.4 : Système linux CentOS 6.6

IV.4.3. Cloudera CDH

CDH est la distribution open source la plus complète et la plus populaire dans le monde. Cloudera est une société de logiciels américaine cofondée en 2008 par le mathématicien Jeff Hammerbach, un ancien de Facebook. Les autres cofondateurs sont Christophe Bisciglia, ex-employé de Google, Amr Awadallah, ex-employé de Yahoo, Mike Olson, PDG de Cloudera. En mars 2009, suivie de Diane Greene, la cofondatrice de VMware, de MartenMickos, l'ex-PDG de MySQL, et de GideonYu, le responsable des finances de Facebook.

La firme Cloudera se consacre au développement de logiciels fondés sur Apache Hadoop, permettant l'exploitation de Big Data, à savoir des bases de données accumulant plusieurs pétaoctets.

CDH contient les principales composantes d'Apache Hadoop pour le stockage évolutif et des calculs distribués. La **figure IV.5** présente les composants de la distribution Hadoop de Cloudera.

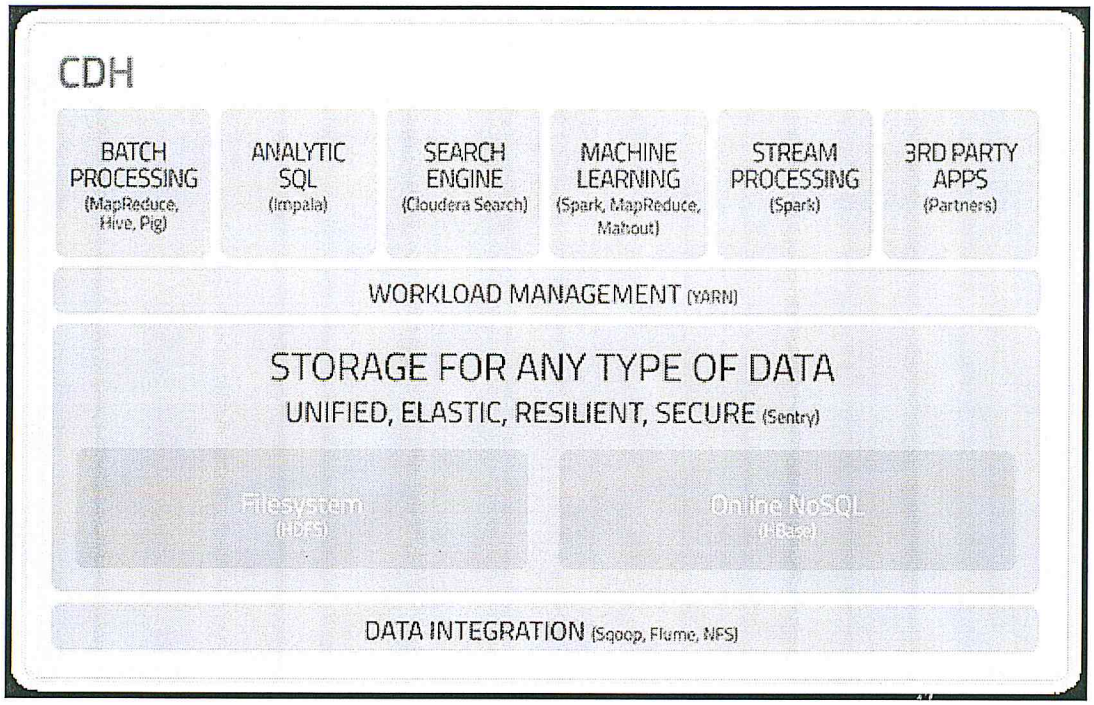


Figure IV.5 : Composants de la distribution Hadoop de Cloudera **[58].**

IV.5. Pourquoi Hadoop ?

Nous avons opté pour le choix de la plateforme Hadoop pour les raisons suivantes :

✓ **Economique** : c'est une solution à moindre coût qui permet aux entreprises de libérer toute la valeur de leurs données en utilisant des serveurs peu onéreux.

✓ **Flexible** : l'utilisation de Hadoop-HDFS permet de stocker d'une manière extensible tout type de donnée structuré (base de données de départ) et semi-structuré (fichier proxy). Ces données sont stockées dans plusieurs machines afin de les traiter de façon distribuée.

✓ **Analyse performante** : l'utilisation de Hadoop-MapReduce permet d'analyser rapidement les différents types de données et de traiter parallèlement de multiples calculs en distribuant une opération sur plusieurs serveurs.

✓ **Architecture Scale-out** : cette architecture nous permet d'ajouter des ressources (stockage et puissance de traitement) à la demande et aux besoins.

✓ **Sécurité optimale de données** : Hadoop est également tolérant aux pannes puisqu'il réplique chaque donnée dans plusieurs serveurs afin de garantir qu'aucune donnée ne soit perdue. Aussi il peut détecter les pannes des serveurs est capable de poursuivees tâches en sollicitant de nouveaux serveurs.

✓ Peu de test sont nécessaires, les librairies mapreduce ont été déjà testées et fonctionnent correctement.

✓ Cette plateforme fournit une abstraction du la programmation et les mécanismes de parallélisassions.

IV.6. Talend open studio

Dans cette section, nous représenté l'outil Talend :

IV.6.1. Définition

Talend Open Studio est un ETL open source apparu en 2005, développé par la société Talend. C'est un ETL de type « générateur de code », c'est-à-dire qu'il permet de créer graphiquement des processus de manipulation et de transformation de données puis de générer l'exécutable correspondant sous forme de programme Java ou Perl **[61].**

Talend Open Studio supporte les composants tiers (Alfesco, SAP, Vtiger CRM…etc), les connectivités aux bases de données (Oracle, SAS, Vertica…etc), les système d'exploitation (Centos Linux, Redhat Entreprise, Microsoft Windows…etc) et les différentes distributions d'Hadoop (ApacheHadoop, Cloudera…etc) **[62].**

IV.6.2. Talend Open studio pour Big data

Talend offre un produit Big Data Open source puissant et flexible simplifiant le fonctionnement des technologies Big Data et permettant de contrôler et d'améliorer les performances de notre application, sans avoir besoin de connaissances ou de ressources spécialisées **[62].**

IV.6.3. Fonctionnalité de Talend Open Studio pour le Big data

Le taux de disponibilité des ressources sur le Big data s'en trouve restreint. La faible quantité de données et le processus d'adoption de Big data en est à ses premières phases et risque d'avoir un impact important sur l'efficacité de sa gestion. Pour relever les défis décrits précédemment, il faut garde à l'esprit quatre éléments déterminants :

- **Intégration des Big Data :** Talend propose un espace de travail ainsi qu'un ensemble intuitif d'outils graphiques permettant une interaction avec une source ou une cible Big Data, sans besoin d'apprendre ni d'écrire du code complexe. Une fois la connexion Big Data configurée et représentée graphiquement, le code sous-jacent est généré de façon automatique
- **Manipulation des Big Data :** Talend résume ces fonctions en une gamme d'outils à partir desquels les scripts sont définis dans un environnement graphique en tant que partie du flux de données, et peuvent être développées rapidement sans aucune connaissance préalable du langage sous-jacent.

- **Qualité et Big Data :** Talend offre des fonctionnalités de qualité des données qui bénéficient de l'environnement massivement parallèle de Hadoop et mettent à disposition tâches et fonctionnalités explicites pour établir le profiling des doublons et les identifier en quelques instants (au lieu de quelques jours) parmi les immenses quantités d'informations stockées. Ceci s'impose comme le complément logique des solutions de qualité et d'intégration des données de l'entreprise, ainsi que de ses bonnes pratiques.

Gestion et gouvernance de projets Big Data : Talend propose une gamme de fonctionnalités pour la gestion de projets Big Data au moyen de laquelle les entreprises peuvent programmer, surveiller et déployer toute sorte de job, tout en utilisant un référentiel commun où les développeurs échangent et partagent métadonnées et artefacts. Par ailleurs, Talend simplifie la génération de codes tels que Hcatalog et Oozie **[62].**

IV.7. Implémentation de l'architecture :

Pour l'implémentation de l'architecture nous avons utilisé un cluster de deux machines virtuels de 4 GO de RAM, tout en sachant que les pré-requis minimale pour la distribution Hadoop de cloudera est de 4 Go de RAM.

Après avoir formé notre cluster comme indiqué dans la section V.5 et V.7 de ce chapitre ;

Après l'installation de Talend Open Studio, son ouverture et l'importation des packages apaches hadoop dans Eclipse comme nous l'avons expliqué dans la **section V.8** de ce chapitre, nous allons expliquer l'implémentation des jobs MapReduce développés en Java Eclipse.

On rappelle que le but de ces jobs mapreduce est d'améliorer l'analyse et les prévisions des départs en retraite et le départ potentiel.

IV.7.1.Retraite

IV.7.1.1. Source de données

Le fichier csv fournit par le Service Gestion de Carrière comporte des données de type structurées pour chaque employé, la **figure IV.26** illustre un extrait de ce fichier.

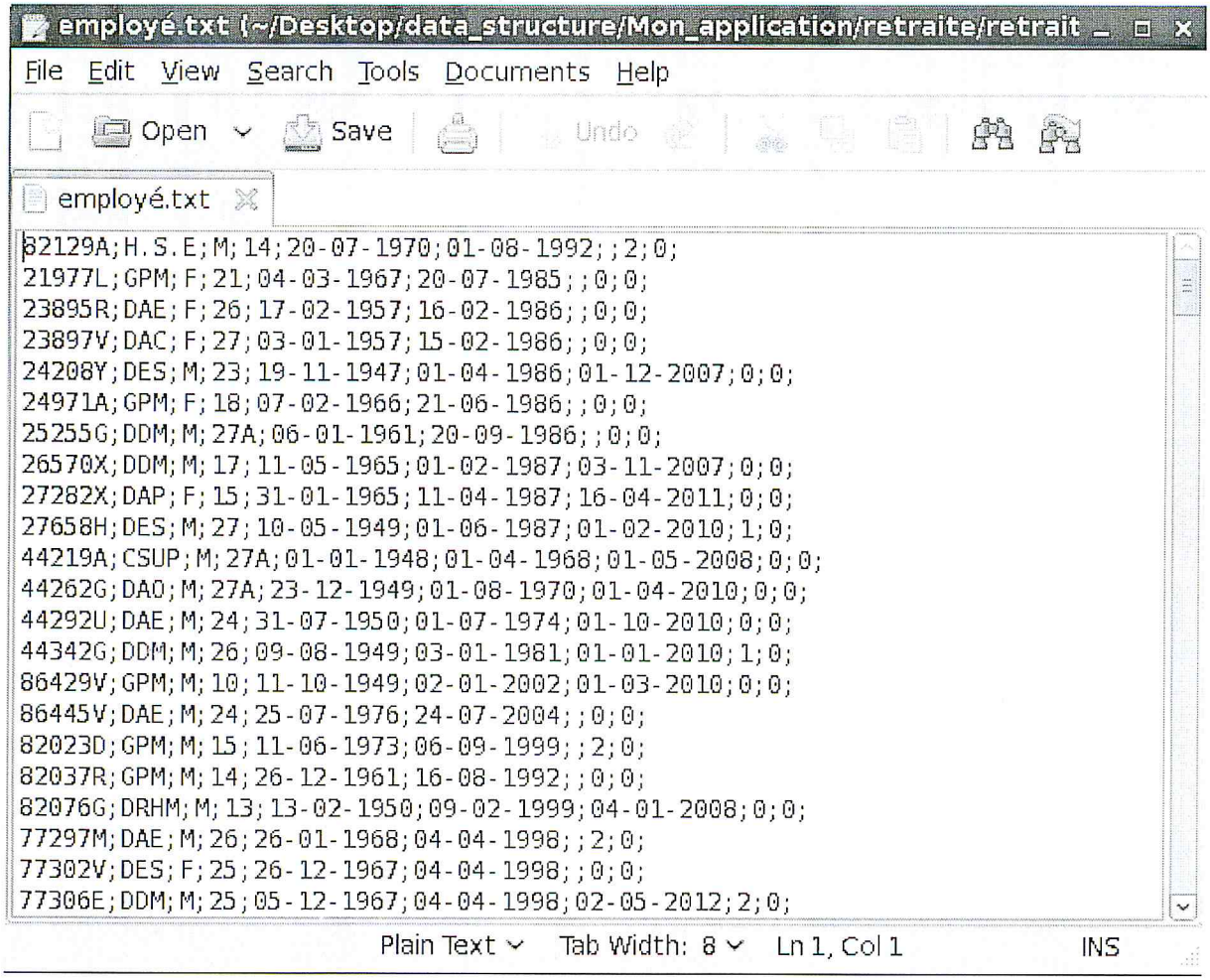


Figure IV.26 : Extrait de fichier **Employé.txt**

Colonne 1 : le matricule de l'employé.
Colonne 2 : la direction de l'employé.
Colonne 3 : sexe de l'employé.
Colonne 4 : Echelle de l'employé.
Colonne 5 : la date de naissance de l'employé.
Colonne 6 : la date de recrutement de l'employé.
Colonne 7 : la date de fin contrat de l'employé.
Colonne 8 : le nombre d'enfant de l'employé.
Colonne 9 : enfant de Moudjahid.

Ce fichier contient des lignes ou la **colonne 7** est non nulle, cela signifie que l'employé est sans poste soit il a démissionné, soit il a retraité où licencie. Dans notre cas, nous s'intéressons aux employés qui sont en production. Pour illuminer ce type de lignes on a chargé le fichier en entrée dans une table oracle, c'est le prejmier prétraitement. Le fait que les données soient chargées, leur type est devenu structuré.

Le deuxième prétraitement revient à ajouter 3 colonnes à la table précédente qui sont prédéfinies par default 'N', ces colonnes sont retraites sans condition d'âge, retraite proportionnel et retraite âge légal. La **figure IV.27** présente la nouvelle table Oracle convertie en fichier texte :

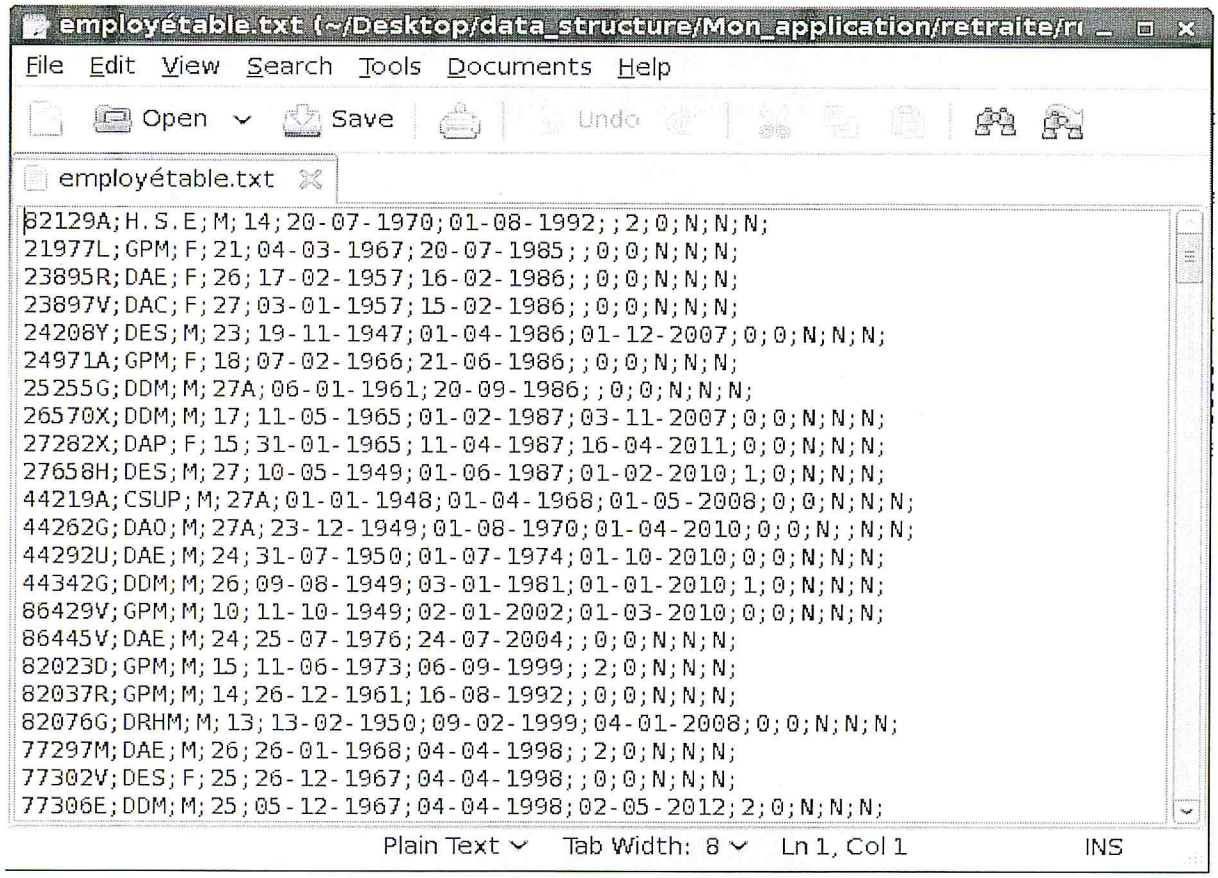


Figure IV.27 : L'extrait table Oracle en fichier texte

Après cet ajout, on applique 3 curseurs sur les nouvelles colonnes, le rôle de ses curseurs est d'identifier les agents qui sont aptes de partir en retraite. **L'ANNEXE1, 2** et **3** sont les codes des trois curseurs cités précédemment.

Le curseur cité en l'ANNEXE1 est utilisé pour prévoir si un employé est concerné par une retraite âge l'égale. Pour établir cette prévision, le curseur a comme entrée la date de naissance de l'agent, son sexe. Pour dire que l'employé X est mis à ce type de retraite il faut que la différence entre la date de prévision et sa date de naissance soit 60 ans pour l'homme et 55 ans pour les femmes. Si la condition précédente est vérifié les colonnes RE_AL, RE_PR, RE_SCA sont mise à Y au lieu N. En fonction de la différence entre la date de prévision et la date de recrutement, on peut déterminer si l'employé sera mis en retraite sans condition d'âge. Cela est programmer dans un curseur cité en ANNEXE2. Le troisième curseur est utilisé afin de déterminer si l'employé est concerné par la retraite proportionnel, les conditions usitées pour vérifier cette condition sont : L'âge est supérieur ou égale à 50 ans et un minimum de 20

ans de versement de cotisation pour les hommes. L'âge est supérieur ou égale à 45 ans et un minimum de 15 ans de versement de cotisation pour les hommes.

La **figure IV.28** suivante présente la nouvelle table en format plat

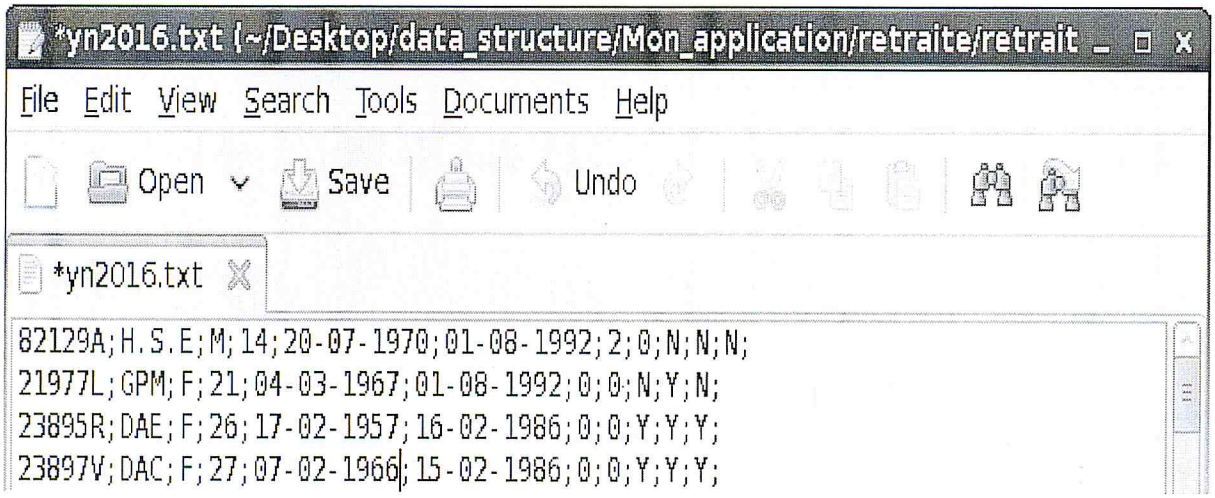


Figure IV.28 : Table Oracle après l'exécution des curseurs

IV.7.1.2. Transformation et intégration des données

Après le prétraitement des données, on doit charger notre table Oracle dans un volume Hdfs, il suffit donc d'utiliser Talend Open Studio comme un moyen de transformation et d'intégration vers Hdfs. Pour établir ce chargement on est passé par les étapes suivantes :

- Créer un projet Talend Open Studio nommé « Sonatrach » comme le commentaire de la **figure IV.29** montre

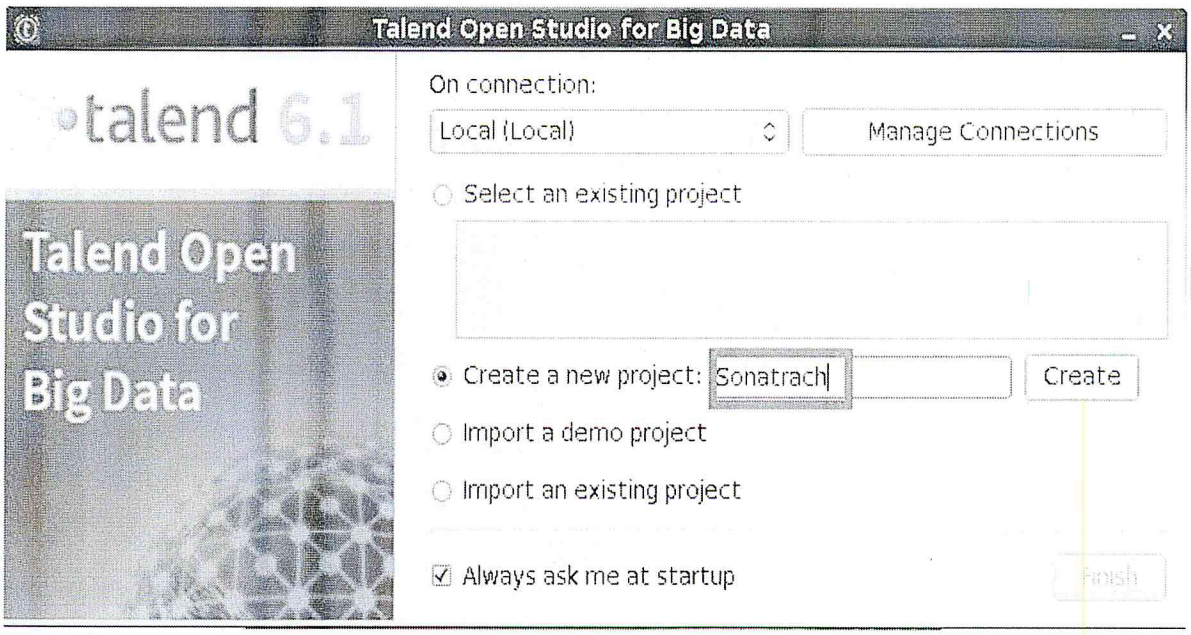


Figure IV.29 : Création d'un projet Talend Open Studio

- Créer les quatre composants de notre job Talend comme la **figure IV.30** montre

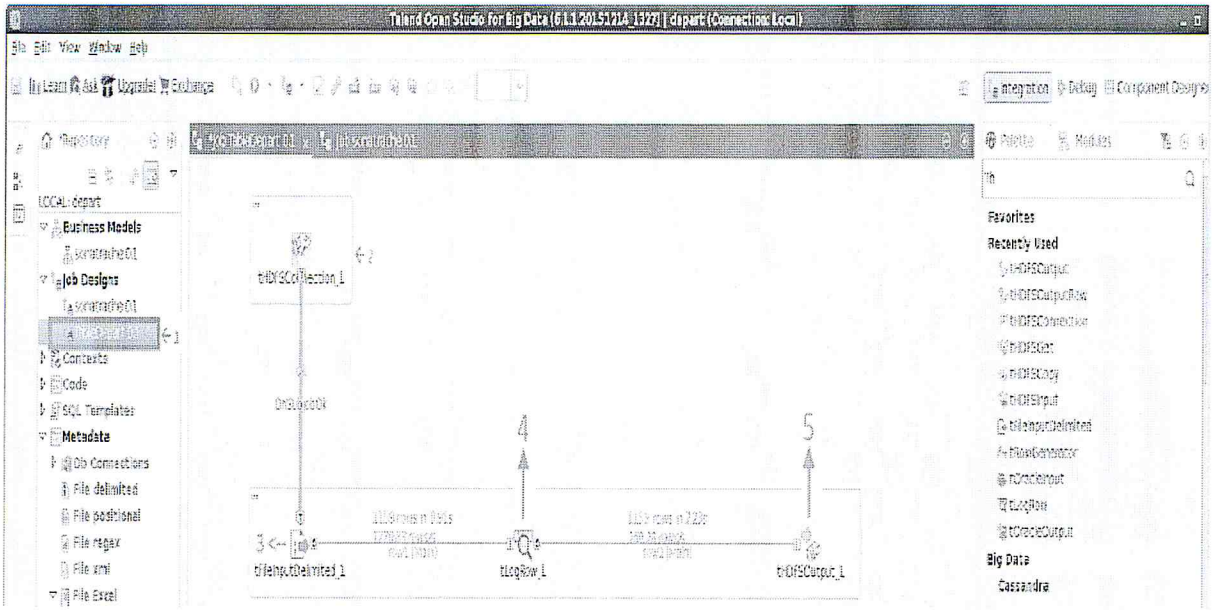


Figure IV.30 : Schéma de quatre composants du job Talend crée

Les quatre composants créés sont :

1. tHDFSConnection Permet d'établir la connexion HDFS.
2. tFileInputDelimited Prend en paramètre le ficher texte à charger.7
3. tLogRow Générer les tuples à partir du ficher texte.
4. tHDFSOutput prend en paramètre le ficher texte charger dans le HDFS.

Après cette étape on doit les configurés comme le montre les **figures IV.31**, **IV.32**, **IV.33**, **IV.34** :

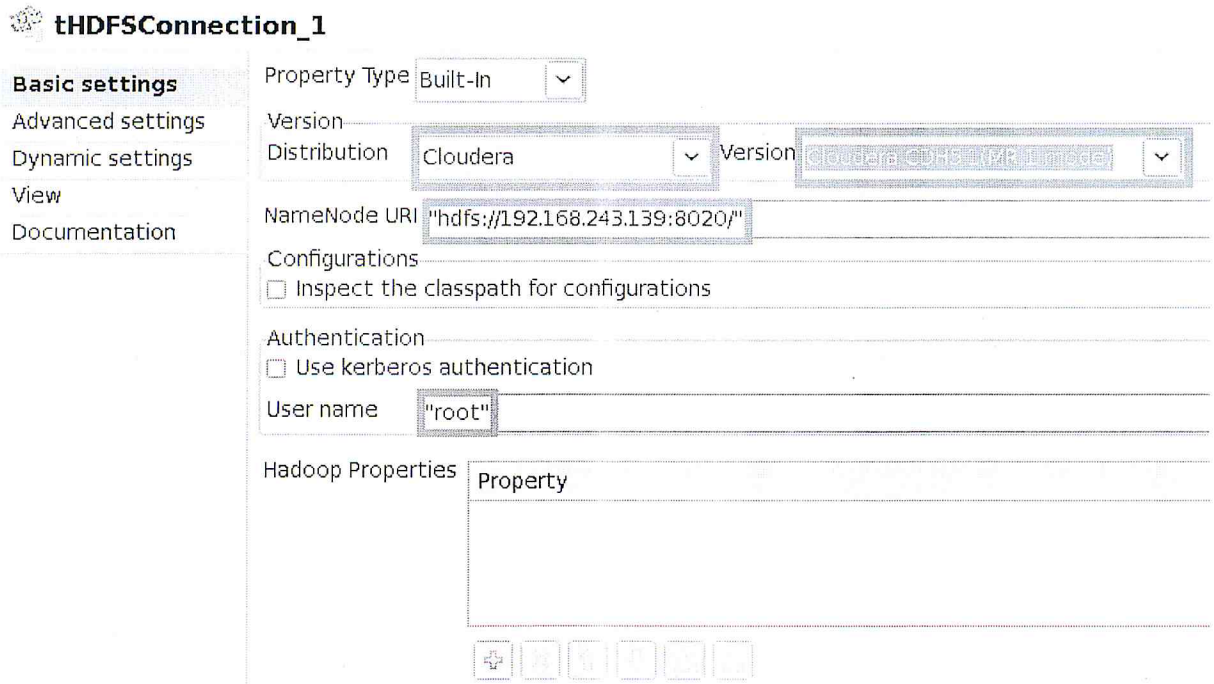


Figure IV.31 : Configuration du fichier tHDFSConnection.

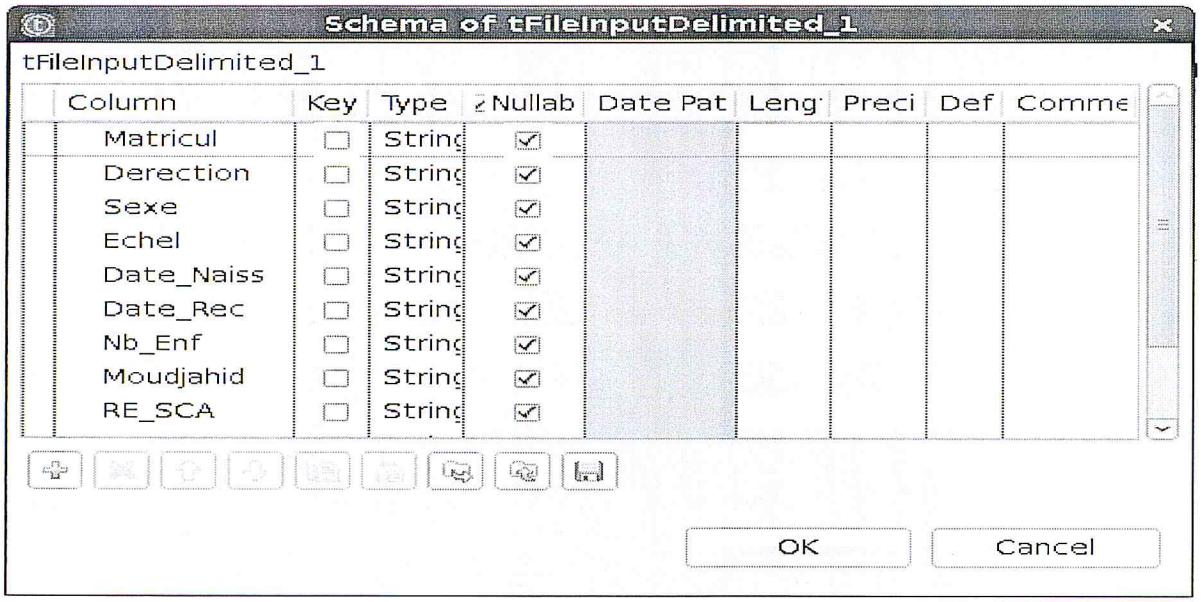


Figure IV.32 : Configuration du fichier tFileInputDelimited.

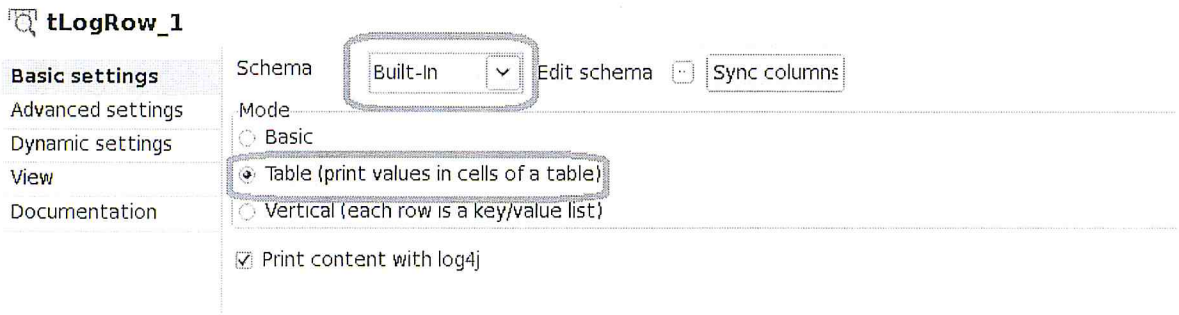


Figure IV.33 : Configuration du fichier tLogRow.

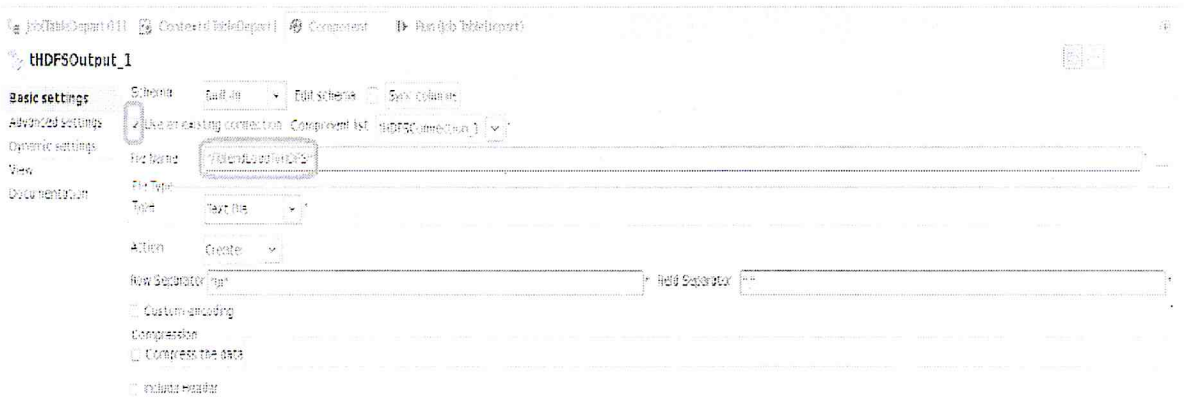


Figure IV.34 : Configuration du fichier tHDFSOutPut.

- Exécuter le job Talend dès qu'il est créé La **figure IV.35** présente l'exécution de job Talend

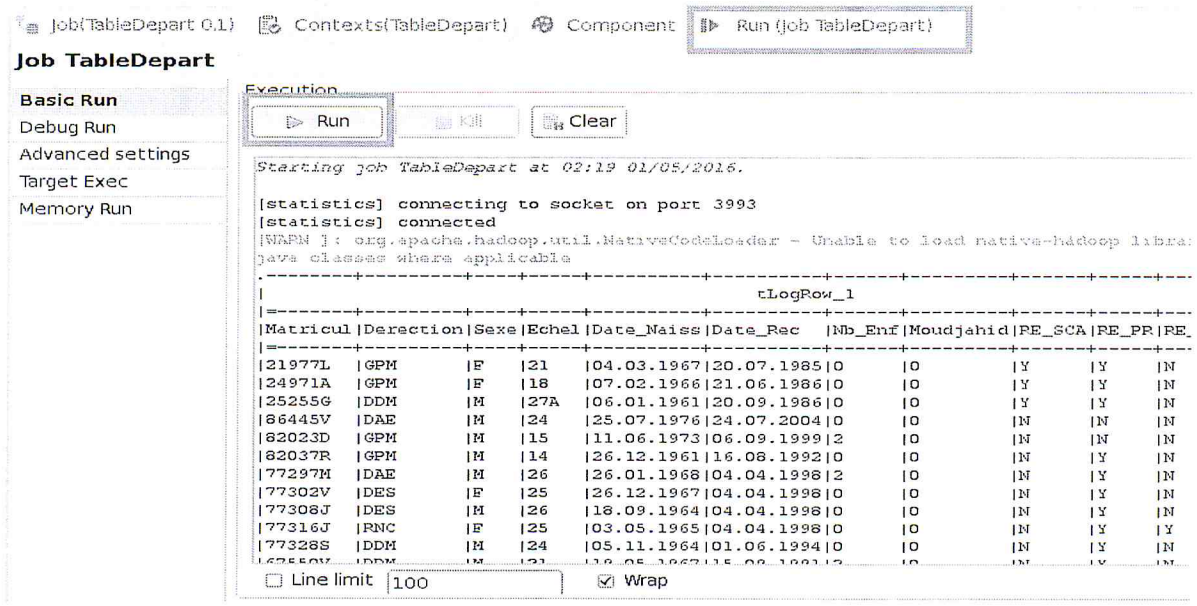


Figure IV.35 : L'exécution de Job Talend crée

Jusqu'à ici la table Oracle est stockée dans un volume HDFS, la **figure IV.36** présente la consultation de volume HDFS crée :

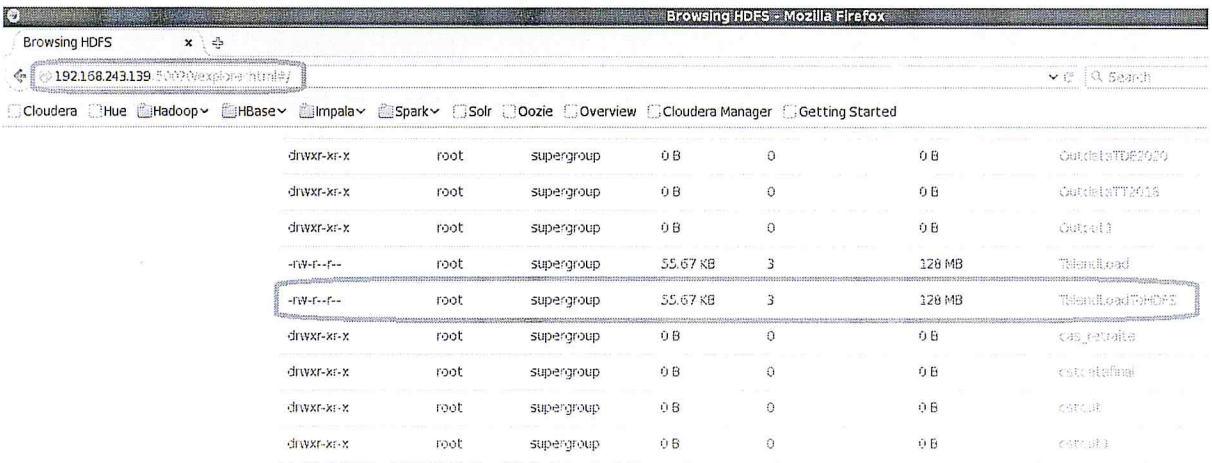


Figure IV.36 : consultation de volume HDFS crée

IV.7.1.3. L'analyse de données

Dès que Les données sont stockées dans un volume Hdfs, on exécute des jobs MapReduce pour l'analyse. Ils sont développé en Java Eclipse grâce aux package Apache Hadoop. Pour la retraite on a développé 3 jobs, le premier est un job d'analyse par employé, le deuxième est un job d'analyse par direction et le 3 est développé pour l'analyse par direction et échelle

IV.7.1.4. L'analyse par employé

- Exécution de job **TNY.jar :**

Pour l'analyse par employé, l'agrégation se fera par rapport au matricule. Cela nous permettra de mieux visualiser et compter le nombre des Y par employé. **L'ANNEXE4** est le code source du job **TNY.jar** commenté. Pour exécuter le job, il suffit d'exécuter le programme jar dans le cluster Hadoop. Il faut donner le paramètre d'entrée **TalendLoadToHDFS** et le paramètre de sortie **Out_traitement** à l'aide de la ligne de commande indiquée dans le commentaire 1 :

Les **figures IV.37** et **IV.38** montrent le début d'exécution et la fin d'exécution du job **TNY.jar** :

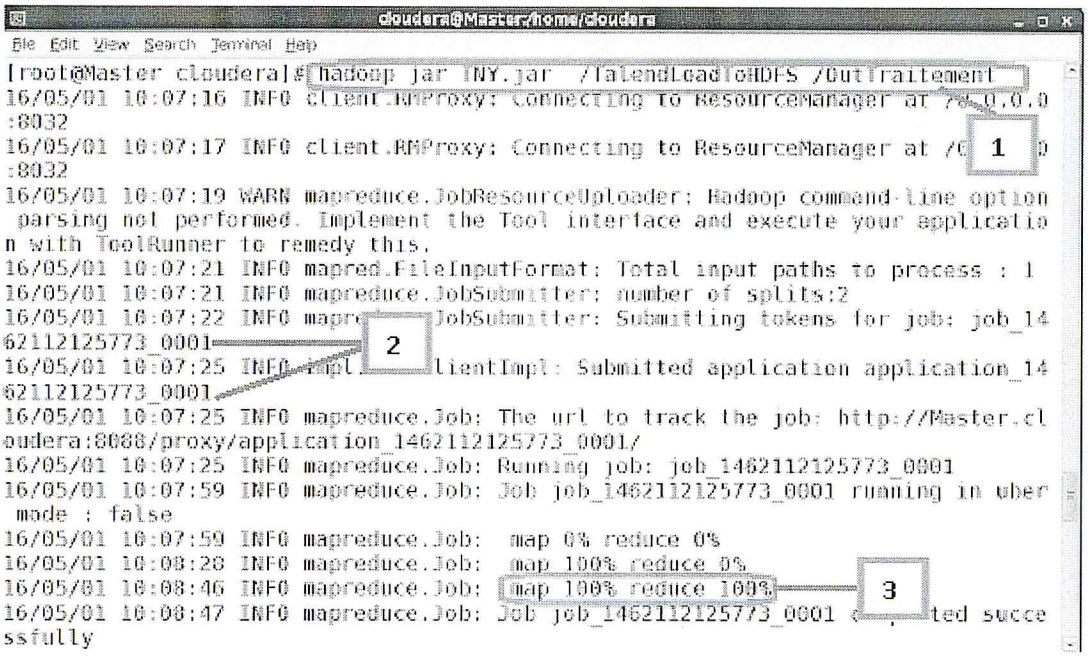


Figure IV.37 : Début d'exécution du **TNY.jar**

Dans la **figure IV.38,** le commentaire N°2 indique l'ID (identifiant) du job mapreduce crée par le JobTracker. Le commentaire N°3 affiche le sucées d'exécution des opérations Map et Reduce sur le volume Hdfs. Le commentaire N°4 de la **figure IV.39** affiche les informations sur l'opération Map et Reduce, on peut lire le Reduce a eu en entré 1073 ligne et en sortie 1073 lignes

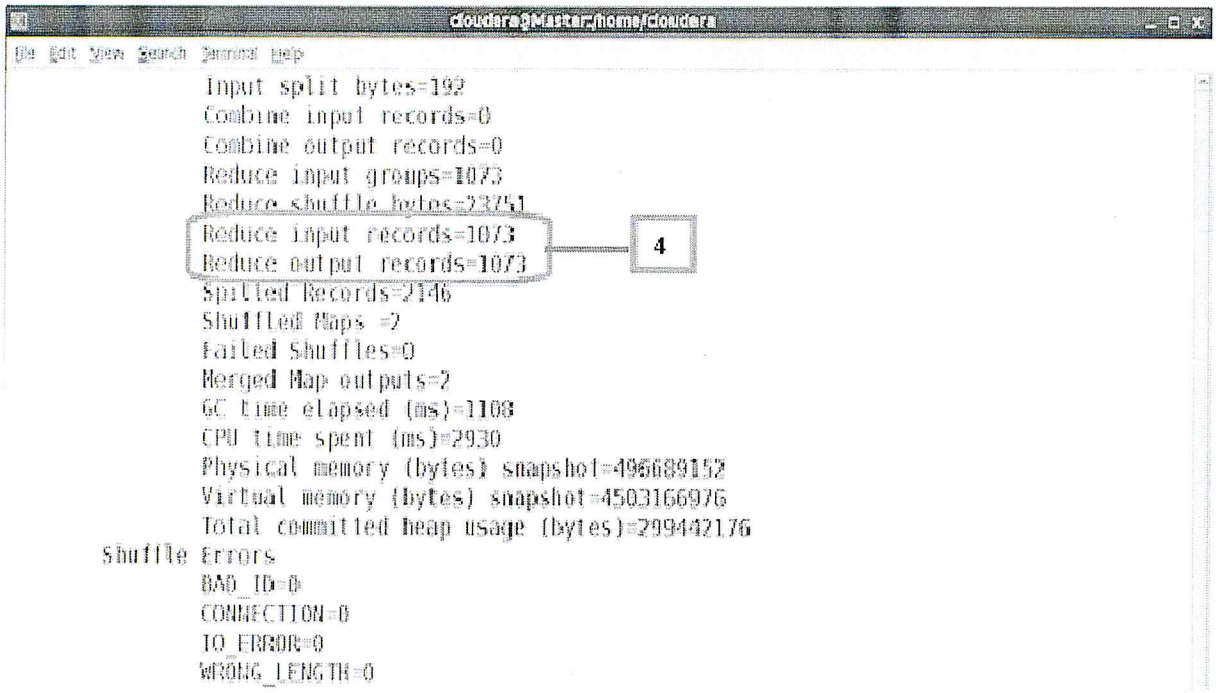


Figure IV.38 : Fin d'exécution du **TNY.jar**

Pour afficher le résultat du job : **TNY. jar** il suffit de lancer la commande suivante

```
hdfs dfs -cat /user/hdfs/output1/part-r-00000
```

Le résultat est présenté dans la **figure IV.39 :**

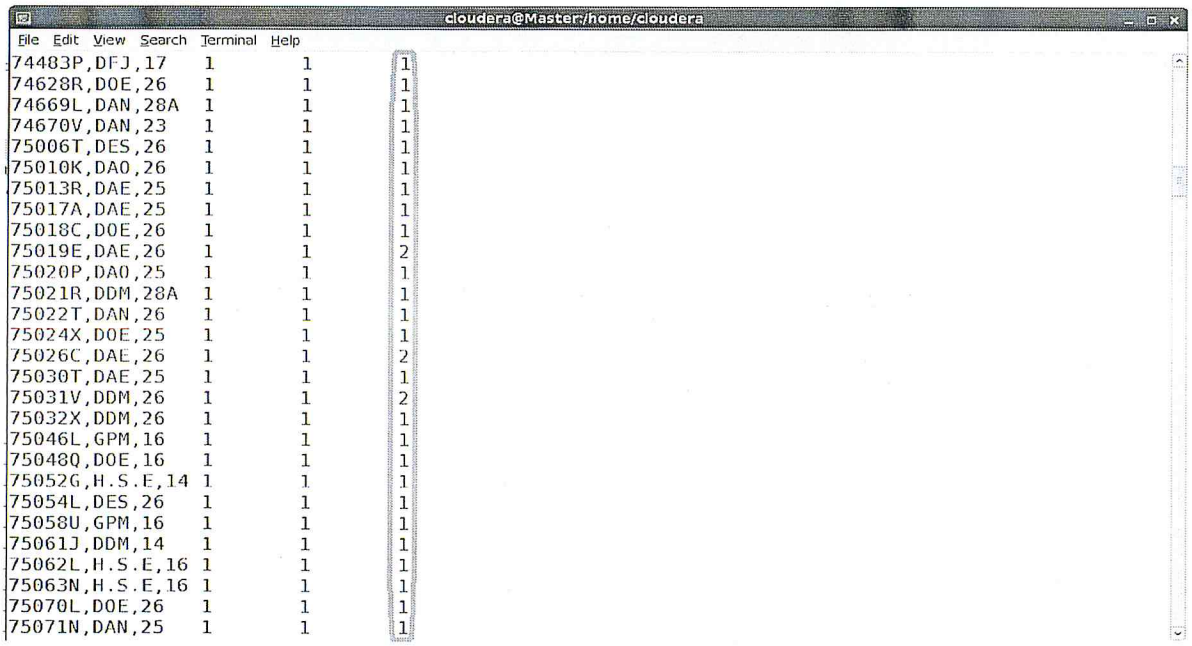


Figure IV.39 : Résultat d'exécution du **TNY.jar**

La colonne encerclée en rouge représente le nombre de Y pour chaque employé, si ce nombre égale à 3, l'employé part en retraite.

IV.7.1.4.1. L'analyse par Direction

- Exécution de job **TD.jar :**

En **ANNEXE5** le code source du job **TD.jar**, qui permet de compter le nombre des retraités par direction. Pour l'exécution du job **TD.jar** on exécute la requête indiquée par le commentaire1 de la Figure **IV.40**

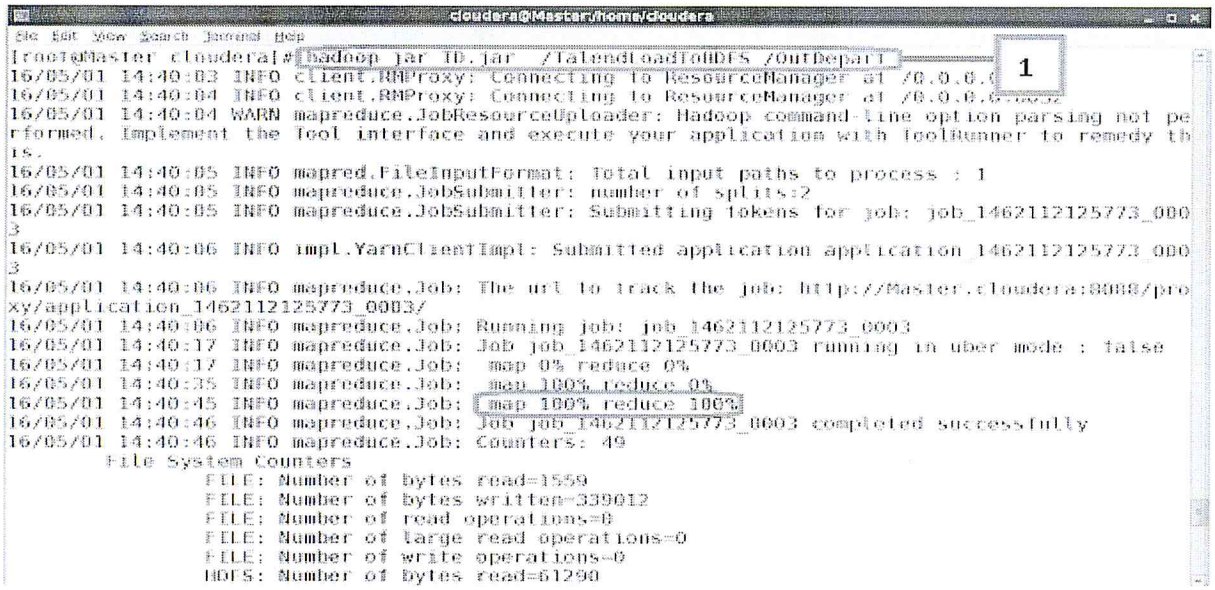


Figure IV.40 : Début d'exécution **de TD.jar.**

Le commentaire N°1 de la **Figure IV.40** illustre la requête d'exécution du job Mapreduce **TD.jar**, le fichier résultat a été placé dans le chemin ou volume HDFS :**/OutDepart**.

Le commentaire N°2 dans la **Figure IV.41** suivante indique que la sortie après l'exécution du job est de 96 lignes au lieu de 1073 lignes.

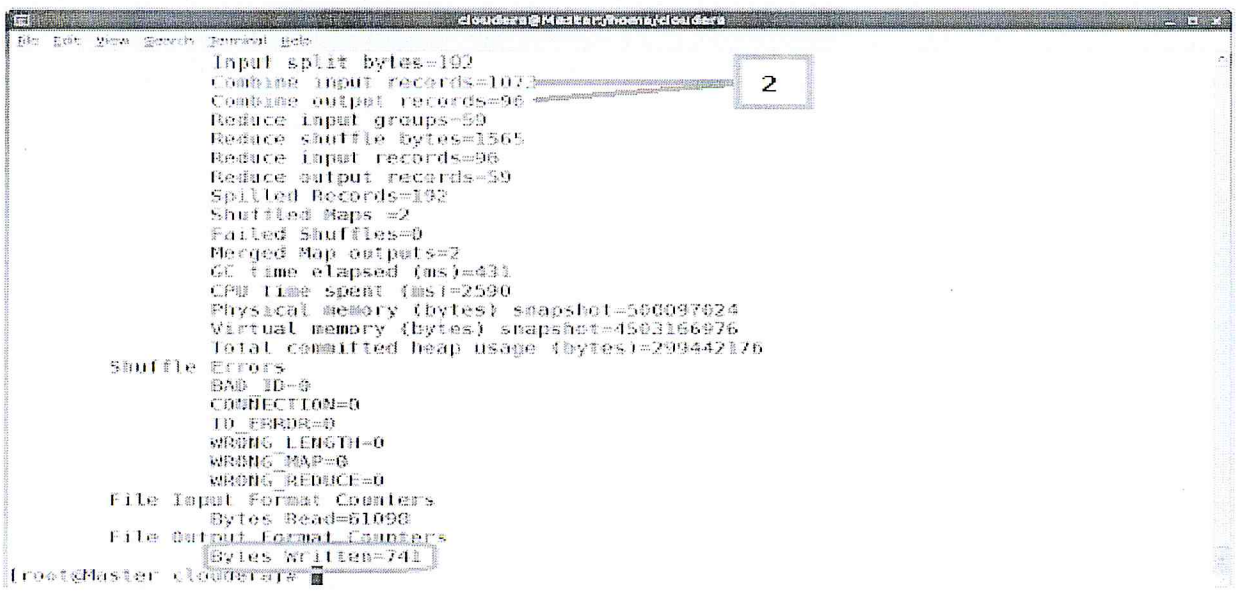


Figure IV.41 : Fin d'exécution du **TD.jar.**

Pour l'affichage du résultat de l'exécution du job mapreduce **TD.jar** il suffit d'exécute la commande :

```
hadoop dfs -cat /OutDepart/part-*
```

La **Figure IV.42** illustre un extrait du résultat du job **TD.jar**. Le commentaire 1 présente les départements de l'organisme d'accueil et le commentaire 2 présente le nombre de retraité par département.

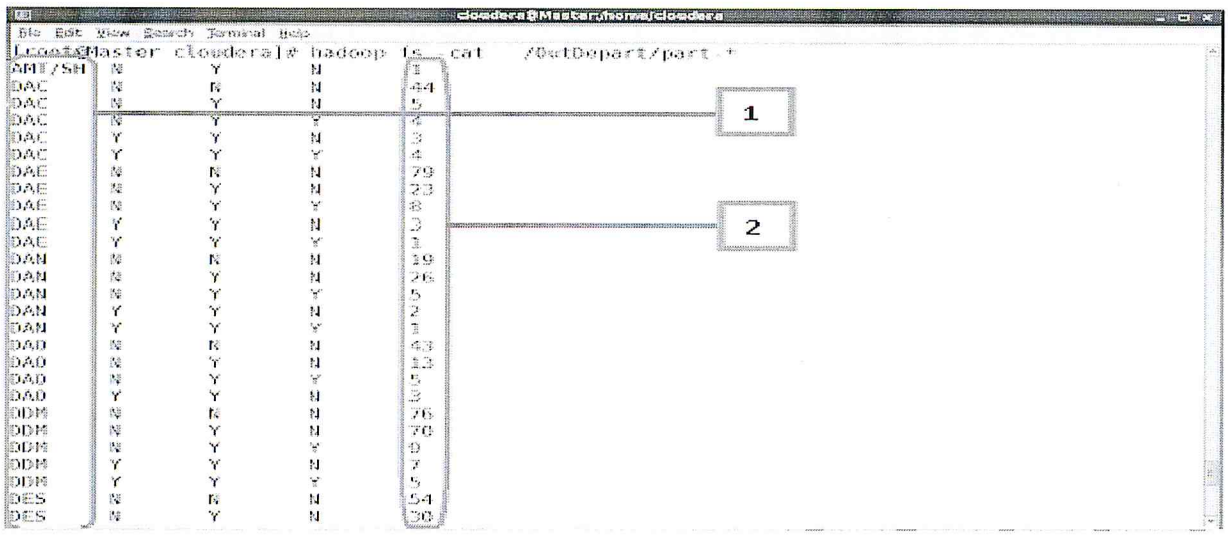


Figure IV.42 : Résultat de job **TD.jar**

IV.7.1.4.2. L'analyse par département et échelle

- **Exécution de job TDE.jar :**

L'ANNEXE6 présente le code source du job **TDE.jar**. Il sert à compter le nombre des retraités par département et échelle. Le commentaire 1 de la **figure IV.43** illustre la commande d'exécution de ce job.

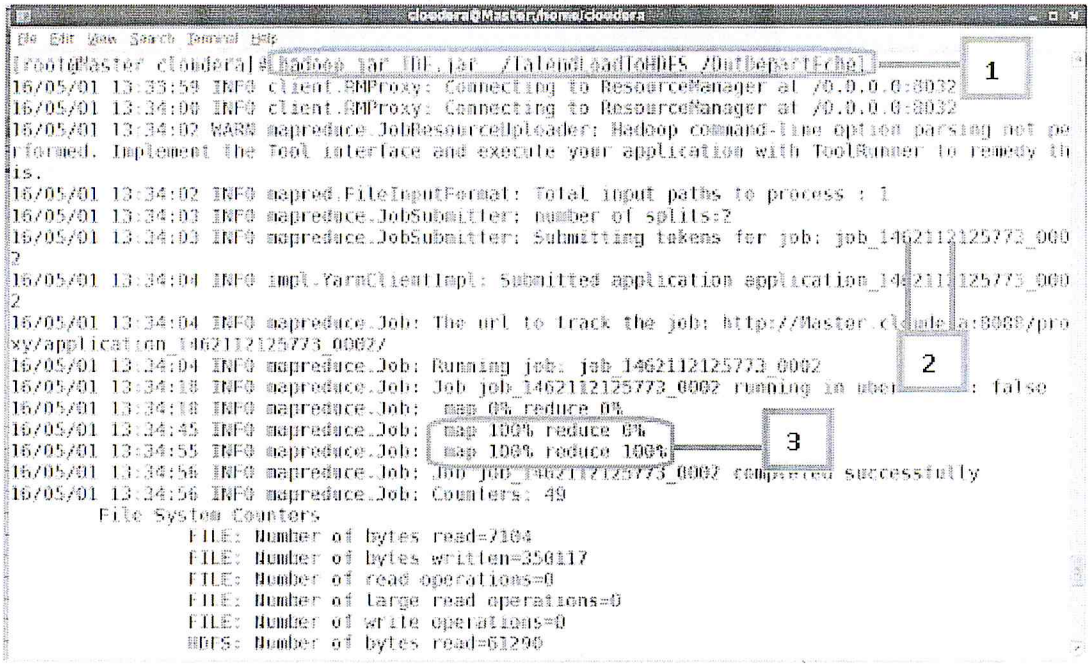


Figure IV.43 : Début d'exécution du job **TDE.jar**

Les 3 commentaires précédents de la **figure IV.43** montre que les opérations Map et Reduce de job identifiés dans le commentaire 2, ont été exécuté avec succès.

A la fin d'exécution du job **TDE.jar**, il a généré en sortie 369 tuples au lieu de 1073 tuples en entrée comme l'indique le commentaire 4 de la **figure IV.44**

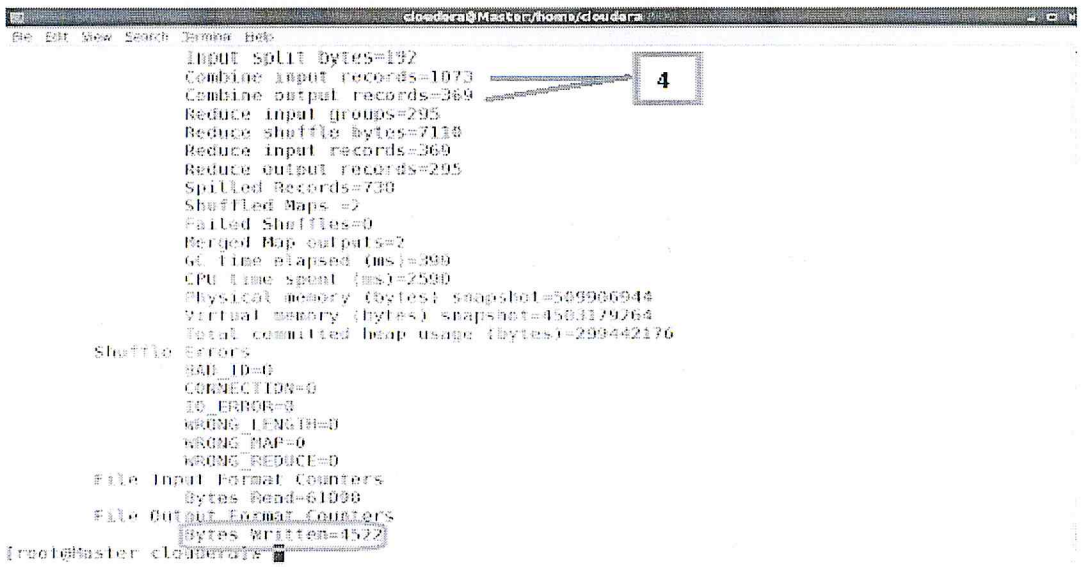


Figure IV.44 : Fin d'exécution du job **TDE.jar.**

La **Figure IV.45** illustre un extrait du résultat du job **TDE.jar**. Le commentaire 1 présente les départements de l'organisme d'accueil et l'échelle pour les retraités pour chaque département le commentaire 2 présente le nombre de retraité par département et échelle

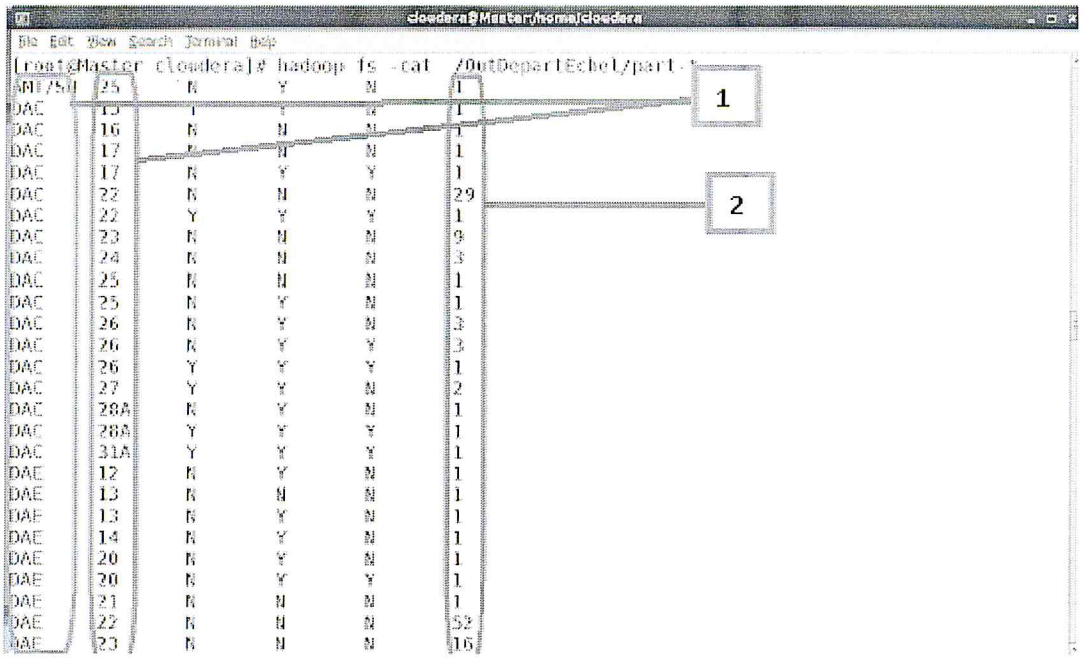


Figure IV.45 : Résultat du job **TDE.jar**

IV.7.2. Le départ potentiel

Le départ potentiel comme son nom l'indique, est la probabilité de la démission d'un employé

L'objectif est de déterminer les employées qui vont probablement partir. Pour l'atteindre on passe par les étapes suivantes :

IV.7.2.1. Source de données

On a reçu du part de service Gestion de carrière un fichier proxy. Les données de ce fichier sont de type semi-structurée. La **figure IV.46** illustre un extrait de ce fichier

Figure IV.46 : Extrait de fichier proxy

Où :

La colonne 1 : les matricules d'employé.

La colonne 2 : les sites consultés par l'employé.

Les données de ce fichier sont déjà, d'où on passe directement à l'étape de l'intégration.

IV.7.2.2. Intégration et analyse des données :

D'après l'architecture proposée dans le chapitre précédent. On a chargé les données semi-structurés directement dans le volume HDFS, les commandes qui permettent ce chargement sont citées au-dessous :

```
Hadoop dfs  -put /DepartPotentiel/DepartPotentiel.csv  /inDepartPotentiel
```

Le chemin /inDapartPotentiel indique le chemin HDFS crée par la commande

```
Hadoop dfs  -mkdir /inDepartPotentiel
```

Exécution de job **PotentielDepart.jar :**

En **ANNEXE7** se trouve le job **PotentielDepart.jar**, il est utilisé pour compter le nombre de consultation d'un site par employé. Le fichier proxy en entrée (inDepartPotentiel) contient des adresses IP qui sont converties en Sites web à travers ce job pour une meilleure observation. La **figure IV.47** montre le début d'exécution de **PotentielDepart.jar.**

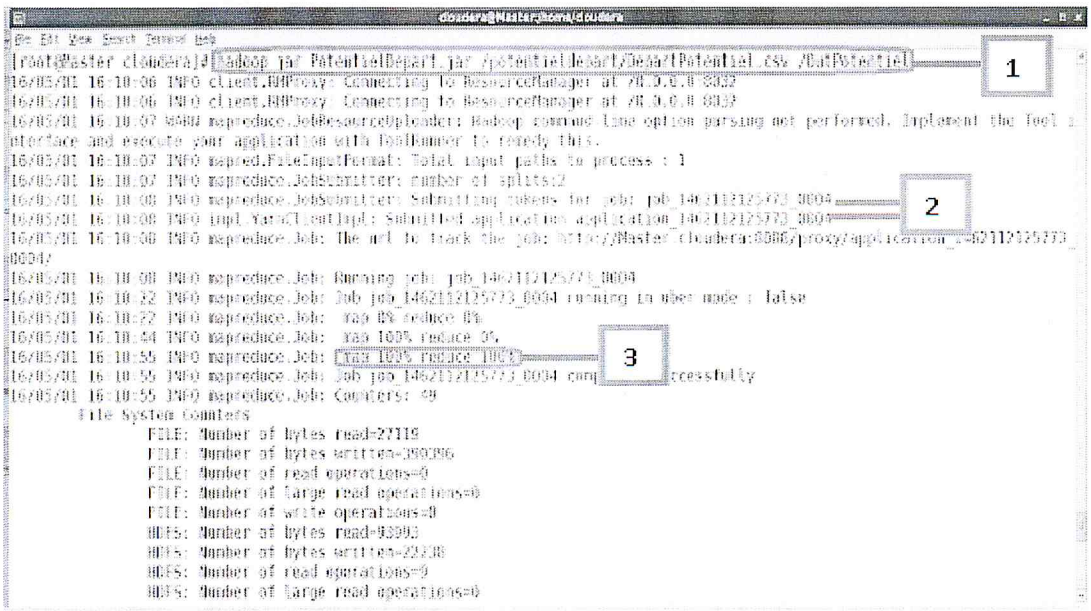


Figure IV.47 : Début d'exécution de **PotentielDepart.jar.**

Le commentaire 1 de la figure précédente représente la commande d'exécution de **DepartPotentiel.jar.** Le commentaire 2 indique l'identifiant du job et le commentaire3 montre le succès d'exécution des opérations Map et Reduce.

La **figure IV.48** indique le résultat d'exécution du job **DepartPotentiel.jar**. La colonne 1 représente le matricule d'employé, la colonne 2 identifie l'url de site consulté par employé et la colonne 3 présente le nombre de consultations de chaque site. Les différents sites qu'on trouve dans notre résultat sont : https://www.linkedin.com , https://dz.viadeo.com/fr , https://plus.google.com , et https://twitter.com .

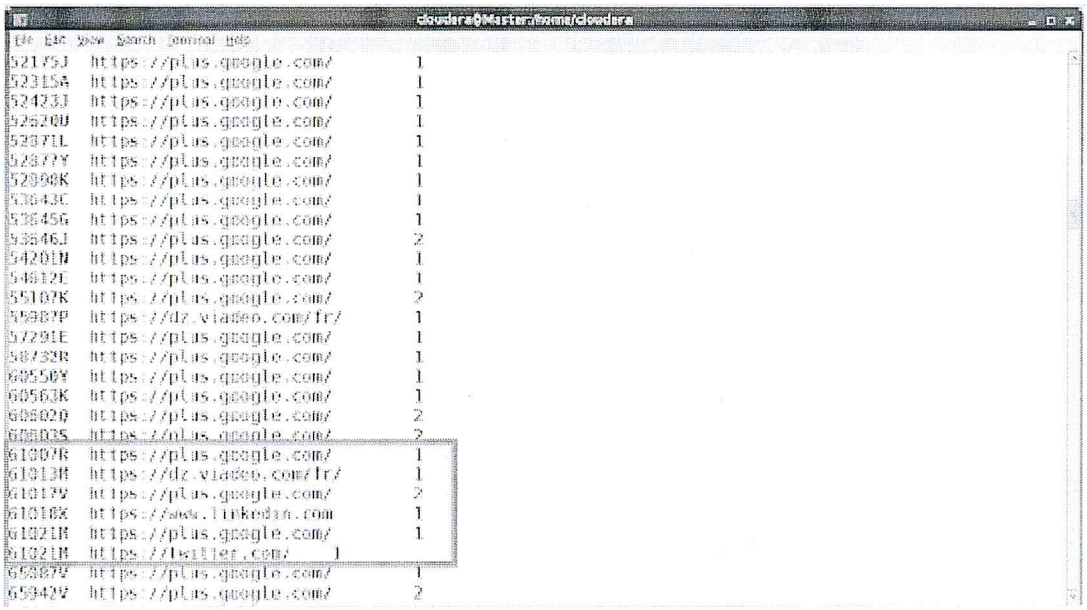


Figure IV.48 : Résultat d'exécution du job **DepartPotentiel.jar**

IV.7.3. Consommation des résultats

Pour mieux visualiser l'importance des résultats générés par les jobs exécutés au paravent, nous allons les interprétés à l'aide des tableaux et des graphes pour souligner l'aide immense qu'apporte l'analyse des fichiers employés à l'aide de hadoop. Nous allons organiser notre consommation par l'interprétation des résultats des jobs de retraite et après l'interprétation celles de départ potentiel :

IV.7.3.1. Interprétation retraite

Au départ, le nombre d'employés pour l'année 2016 est de 1160 agents. La **figure IV.49** est une capture d'écran d'un extrait d'employés depuis la table de bord de notre application :

1	82129A	H.S.E	M	14	20.07.1970	01.08.1992	2	Non	✗	✗	✗
2	21977L	GPM	F	21	04.03.1967	20.07.1985	0	Non	✗	✓	✗
3	23895R	DAE	F	25	17.02.1957	16.02.1985	0	Non	✗	✓	✓
4	23897V	DAC	F	27	03.01.1957	15.02.1985	0	Non	✗	✓	✓
5	24571A	GPM	F	18	07.02.1966	21.06.1986	0	Non	✗	✓	✗
6	25255G	DDM	M	17A	06.01.1961	20.09.1985	0	Non	✗	✓	✗
7	86445V	DAE	M	24	25.07.1976	24.07.2004	0	Non	✗	✗	✗
8	82023D	GPM	M	15	11.06.1973	06.09.1999	2	Non	✗	✗	✗
9	82037R	GPM	M	14	26.12.1961	16.09.1992	0	Non	✗	✓	✗
10	77297M	DAE	M	26	26.01.1968	04.04.1998	2	Non	✗	✗	✗
11	77302V	DES	F	25	26.12.1967	04.04.1998	0	Non	✗	✓	✗
1155	21462P	DES	F	22	25.01.1984	27.09.2015	0	Non	✗	✗	✗
1156	21464G	DAE	M	22	23.11.1989	27.09.2015	0	Non	✗	✗	✗
1157	21477S	DAO	M	22	06.06.1987	02.12.2015	0	Non	✗	✗	✗
1158	15419F	DES	F	22	31.01.1981	15.02.2016	0	Non	✗	✗	✗
1159	15409B	DDM	M	22	21.07.1983	01.02.2015	0	Non	✗	✗	✗
1160	15418D	DAE	M	22	03.06.1990	21.02.2016	0	Non	✗	✗	✗

Figure IV.49 : Extrait d'employé pour l'année 2016.

Comme la figure précédente l'illustre, les 3 dernières colonnes présentent les trois types de retraite : retraite âge légale, retraite proportionnelle et retraite sans condition d'âge. Si une de ses cases est à vrai, donc l'employé a la possibilité de mettre fin à son parcours, et si les 3 cases sont toutes vrai, on est sûr que l'employé fait son départ en retraite.

Après l'exécution du premier job **TNY.jar,** on constate que le nombre d'employés sortants en retraite pour l'année 2016 est de 20 agents, donc il reste 1140 employés pour l'année 2017. La **figure IV.50** présente l'extrait des employés sortants pour l'année 2016 :

N°	MATRICULE	DEPARTEMENT	SEXE	ECHELON	DATE NAIS	DATE EMB	ENFANT	MOTIF
1	01312C	DOE	M	25	07-10-1956	15-08-1979	0	Non
2	02749H	DOE	M	26	08-10-1956	16-08-1982	0	Non
3	02752W	DOE	M	31A	21-11-1955	21-08-1982	1	Non
4	03498T	DDM	M	28A	31-12-1956	30-09-1981	0	Non
5	03505G	DAN	M	27	08-02-1956	01-12-1981	1	Non
6	04831E	DOE	F	26	10-08-1959	01-10-1984	0	Non
7	11082R	DXP	M	30A	29-06-1950	17-08-1970	0	Non
8	17811H	DDM	M	27A	09-07-1956	16-08-1982	1	Non
9	20180N	DXP	M	28A	15-10-1956	30-03-1985	0	Non
10	22626D	DES	M	24	19-03-1956	13-09-1976	0	Non
11	23899A	DOE	M	27	31-12-1956	25-02-1986	0	Non
12	24966G	GPM	M	21	04-04-1956	21-06-1986	1	Non
13	49136F	DES	M	28	10-04-1956	01-10-1977	0	Non
14	49235M	INC	M	26	08-12-1952	01-07-1974	0	Non
15	65087V	DDM	F	24	14-05-1960	01-02-1983	0	Non
16	65386U	RNC	M	28A	01-01-1956	01-10-1977	0	Non
17	73004E	DOE	M	31A	22-08-1952	10-09-1978	0	Non
18	73588T	DOE	M	25	31-12-1955	12-07-1976	0	Non
19	89252D	SIE	M	26B	13-06-1956	24-02-1996	3	Non
20	97205N	DAE	M	31A	23-08-1956	16-08-1982	0	Non

Figure IV.50 : Employés sortants pour l'année 2016

La **figure IV.51** présente un histogramme qui rassemble l'ensemble d'employé par rapport aux départements pour l'année 2016. Comme on le voit, l'organisme d'accueil gère 17 départements qui sont **H.S.E**, **GPM**, **DAE**, **DAC**, **DDM**, **DES**, **RNC**,**DFJ**, **AMT/SH**, **DXP**, **DOE**, **PLF**, **DAN**, **DAO**, **SIE**, **INC** et **DRN**. Ces départements disposent respectivement de : 46, 165, 117, 64, 188, 97, 5, 55, 1, 10, 258, 21,56, 65, 7, 4 et 1 agents.

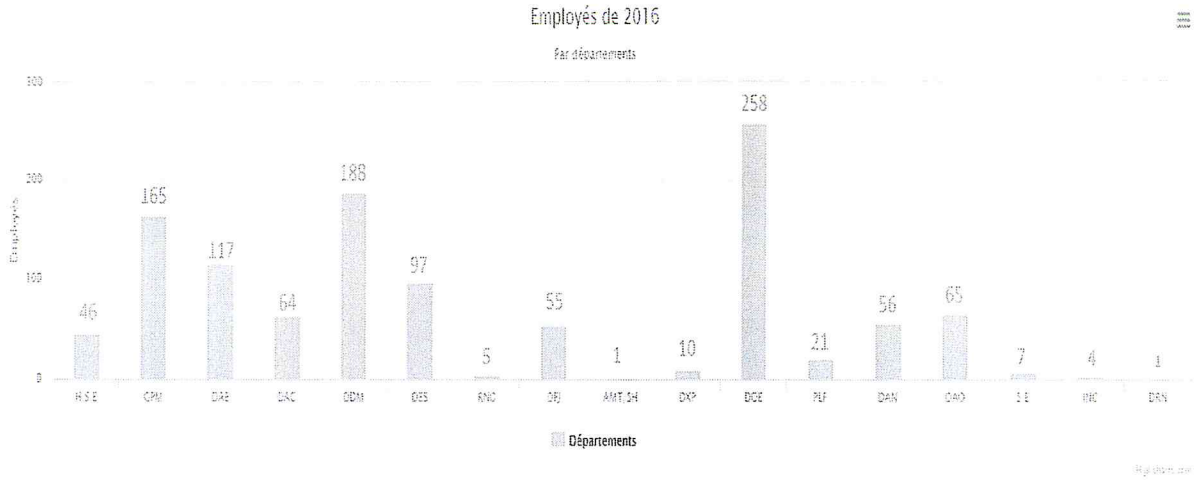


Figure IV.51 : Représentation graphique du nombre employé par département pour l'année 2016

L'exécution du job **TD.jar** nous a permet d'avoir le nombre d'employés sortants par département. Comme la **figure IV.52** l'illustre, certain département n'ont connu aucun départ en retraite comme **H.S.E**, **DFJ, AMT/SH**, **PLF**, **DAO** et **DRN**, par contre les départements **DOE**, **DDM**, **DAN**, **DXP**, **DES**, **GPM**, **INC**, **RNC**, **SIE** et DAE ont perdu successivement 7, 3, 1, 2, 2, 1, 1,1, 1 et 1 agents

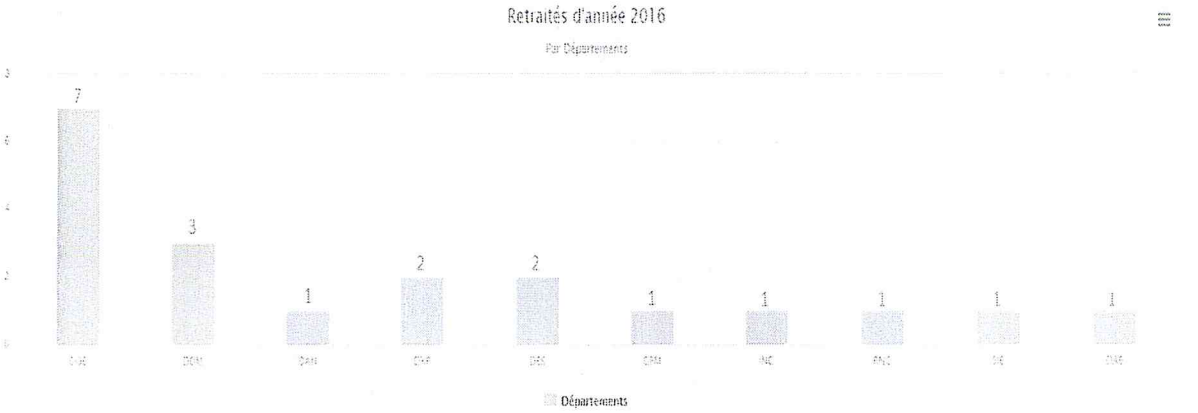


Figure IV.52 : Représentation graphique de nombre employés retraités par département pour l'année 2016.

La **Figure IV.53** est une représentation de tous les employés par échelle pour l'année 2016 sous une forme d'une table.

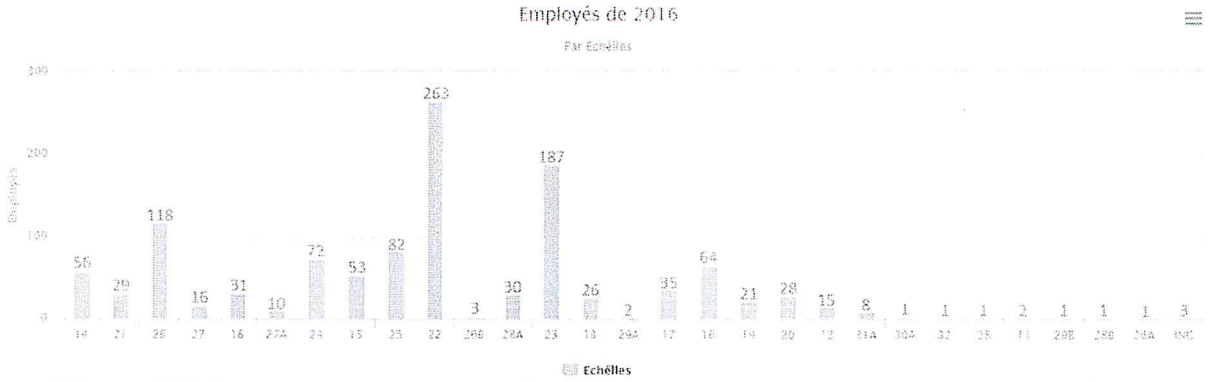


Figure IV.53 : Représentation graphique des employés par échelle pour l'année 2016

La Figure IV.54 est une représentation des employés retraités par échelle pour l'année 2016 sous une forme d'une table.

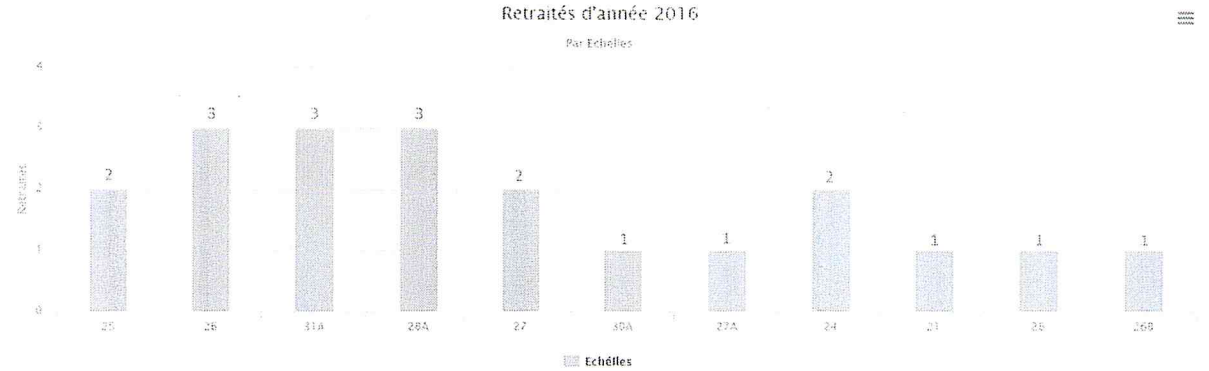


Figure IV.54 : Représentation graphique des employés retraités par échelle pour l'année 2016

On a élaboré la même démarche pour avoir un reporting sur la retraite de 2017. La **figure IV.55** est un aperçu de tous les employés pour l'année 2017.

1	82139A	H.S.E	M	14	20.07.1970	01.08.1992	2	Non	✗	✗	✗
2	21977L	GPM	F	21	04.01.1967	20.07.1987	0	Non	✓	✓	✗
3	23895R	DAE	F	26	17.02.1957	16.02.1986	0	Non	✓	✓	✓
4	23897V	DAC	F	27	05.01.1957	15.02.1986	0	Non	✓	✓	✓
5	24971A	GPM	F	18	07.02.1966	21.06.1986	0	Non	✗	✓	✗
6	25255G	DDM	M	27A	06.01.1961	20.09.1986	0	Non	✗	✓	✗
7	86445V	DAE	M	24	25.07.1976	24.07.2004	0	Non	✗	✗	✗
8	82023D	GPM	M	15	11.06.1973	06.09.1999	2	Non	✗	✗	✗
9	82037R	GPM	M	14	26.12.1961	16.06.1992	0	Non	✗	✓	✗
10	77297M	DAE	M	26	26.01.1968	04.04.1998	2	Non	✗	✗	✗
11	77301Y	DES	F	25	26.12.1967	04.04.1998	0	Non	✓	✓	✓
1133	21443A	DAO	F	22	01.06.1971	27.09.2015	0	Non	✗	✗	✗
1134	21458L	DAE	M	22	16.10.1968	27.09.2015	0	Non	✗	✗	✗
1135	21482P	DES	F	22	25.01.1984	27.09.2015	0	Non	✗	✗	✗
1136	21464G	DAE	M	22	23.11.1989	27.09.2015	0	Non	✗	✗	✗
1137	21477S	DAO	M	22	06.06.1987	02.12.2015	0	Non	✗	✗	✗
1138	15419F	DES	F	22	31.01.1981	15.02.2016	0	Non	✗	✗	✗
1139	15409B	DDM	M	22	21.07.1983	01.02.2015	0	Non	✗	✗	✗

Figure IV.55 : Aperçu de tous les employés de l'année 2017

En exécutant le premier job qui est **TNY.jar,** on a eu1139 retraité pour l'année 2017 et un reste de 1120 employés pour l'année 2018. La **figure IV.56** est un aperçu des employés retraitées pour l'année 2017.

N°	Matricule	Service	Sexe	Position	Naissance	Retraite	Enfant	Marié
1	23895R	DAE	F	26	17.02.1957	16.02.1986	0	Non
2	23897V	DAC	F	27	03.01.1957	15.02.1986	0	Non
3	050388	PLF	M	29A	06.03.1957	04.09.1982	0	Non
4	75069W	GPM	M	14	25.06.1957	23.08.1994	2	Non
5	27289T	DDM	M	27A	22.09.1957	18.04.1987	0	Non
6	90019M	DDM	M	31A	24.11.1957	01.10.1981	1	Non
7	86394G	DDM	F	15	07.09.1957	11.12.1999	0	Non
8	74629T	DDM	M	26	08.12.1957	03.08.1993	0	Non
9	02756F	DXP	M	32	03.01.1957	16.08.1982	0	Non
10	76218M	DES	M	31A	27.04.1957	02.05.1981	0	Non
11	95583K	DXP	M	28A	26.04.1957	16.08.1982	0	Non
12	95641N	DXP	M	27	26.04.1957	16.08.1982	1	Non
13	77354W	DDM	M	28A	01.09.1957	01.08.1999	0	Non
14	01307J	DOE	M	26	28.02.1957	19.11.1979	0	Non
15	95576M	DOE	M	28A	19.12.1957	02.10.1982	0	Non
16	78760U	DOE	M	24	12.12.1957	01.02.1999	0	Non
17	84412R	DOE	M	26	04.11.1957	23.11.1999	0	Non
18	91754W	DOE	M	18	02.09.1957	22.08.1977	0	Non
19	74043M	GPM	M	14	31.12.1957	30.01.1994	0	Non

Figure IV.56 : Employés sortants pour l'année 2017

La **figure IV.57** est une représentation graphique des employés par département pour l'année 2017, et la **figure IV.58** est aussi une représentation graphique des employés retraités, mais pour l'année 2018.

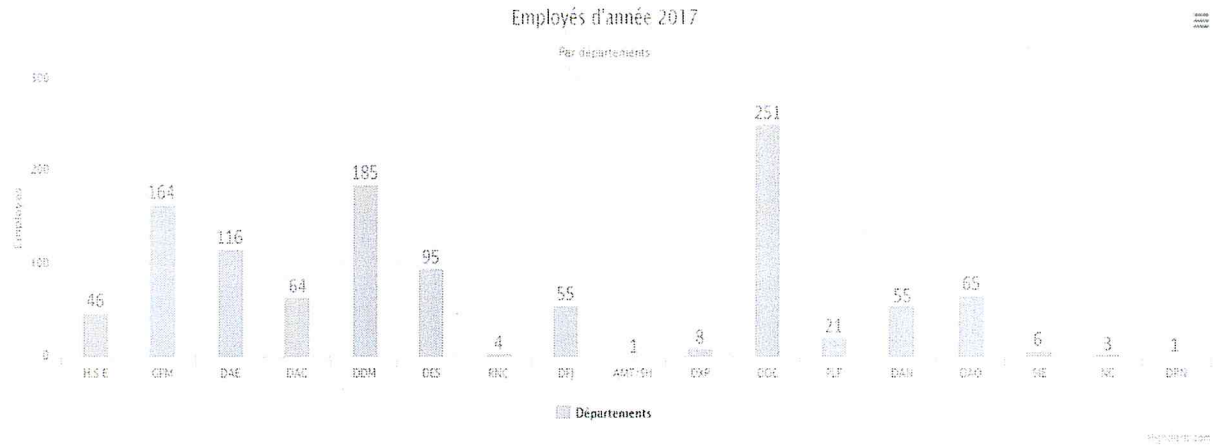


Figure IV.57 : Représentation graphique des employés par département pour l'année2017

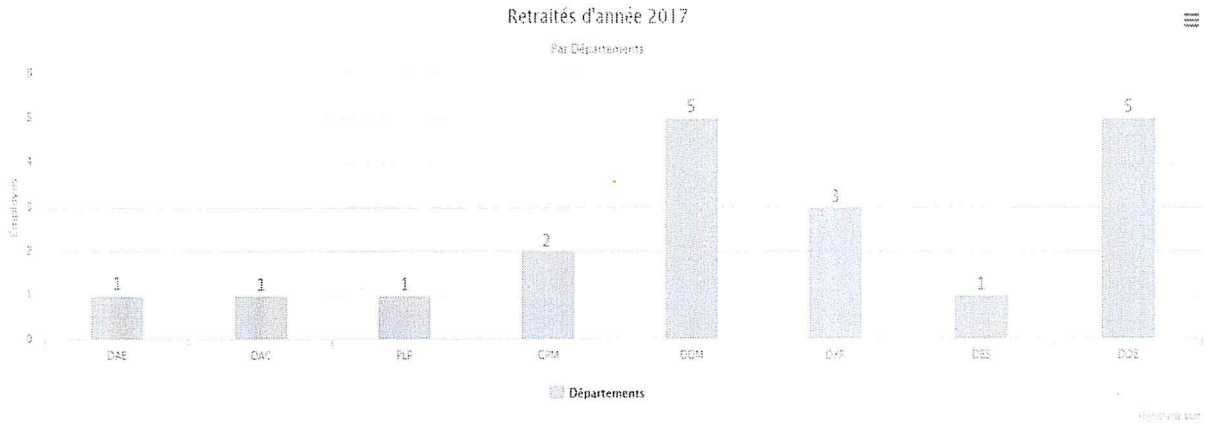


Figure IV.58 : Représentation graphique des employés retraités par département pour l'année2017

Comme on remarque, les départements touchés par un nombre de retraite élevé sont **DDM**, **DRP** et **DOE**

La **Figure IV.59** est une représentation de tous les employés par échelle pour l'année 2017 sous une forme d'une table.

Figure IV.59 : Représentation graphique des employés par échelle pour l'année 2017

La **Figure IV.60** est une représentation des employés retraités par échelle pour l'année 2017

Figure IV.60 : Représentation graphique des employés retraités par échelle pour l'année 2017.

IV.7.3.2. Interprétation départ potentiel :

Pour ne pas oublier, le job **DepartPotentiel.jar** a pour rôle l'identification des employés qui consulte fréquemment les sites de recherche travail. Après l'exécution de ce job, le résultat est présenté dans la **figure IV.61**

La **Figure IV.61** nous permet de voir le nombre total des employés qui potentiellement veulent quitter leur travail selon le site web visité.

Figure IV.61 : Représentation graphique du nombre employé par site visité

IV.8. Conclusion

Dans ce chapitre, nous avons implémenté l'architecture du Big Data avec toutes sec couches, nous avons choisi Hadoop comme notre principale Framework, le HDFS de Hadoop pour le stockage et la réplication des données, Talend pour la préparation et l'intégration des données dans le HDFS, le Mapreduce pour le traitement de ces dernières et pour la visualisation du résultat du traitement. Nous avons opté pour la réalisation d'un tableau de bord pour le suivi des serveurs.

CONCLUSION GENERALE

L'informatique à la base était le traitement de grande quantité de données, mais ce qui a changé avec la venue du Big Data, c'est la formalisation de l'évolution des volumes, de la vitesse et de la variété des données, qui crée de la valeur ajoutée.

Le Big Data a déjà marqué de son empreinte le marché IT (Information Technologie) et commence à s'imposer comme un standard en matière de traitement de données volumineuses. Toutefois, dans ce contexte, il est difficile de savoir quel fournisseur propose l'implémentation la plus complète d'Apache Hadoop dans un Cloud public.

Le Big Data est avant tout une démarche stratégique, il faut se poser la question : Comment est-ce qu'à partir des données qu'on a ou qu'on peut collecter dans l'entreprise, on peut créer de la valeur ? Autrement dit, qu'est-ce qu'on peut créer à partir de nos données. C'est ça la stratégie, donner de la valeur aux données en les transformant en valeur ajoutée. C'est grâce à cette stratégie que le Big Data fonctionnera à l'intérieur d'une entreprise.

Le bilan de ce stage est dans l'ensemble positif, les principaux buts du projet étant accomplis. La plus grande partie du travail qui nous a été demandée a été réalisée.

Pour conclure, on dirait que ce stage nous a été d'un apport indéniable en matière de connaissances acquises sur les technologies du Big Data. C'est une expérience enrichissante qui nous a permis de comprendre les enjeux d'un projet car nous ne possédons pas de connaissance concernant les technologies du Big Data. A l'issu de notre stage, la montée en compétence était bien perceptible au travers du travail réalisé.

Nous avons réalisé l'objectif attendu qui était la compréhension du nouveau paradigme qui est le Big data et implémenté le plus répondu de ses outils qui est Hadoop. Le plus important et que l'application de hadoop nous a permis de nous familiariser avec le concept du Big data et surtout d'élaborer une architecture de stockage et d'analyse de données scalable et élastique, c'est-à-dire qu'il suffit d'ajouter un serveur hadoop pour augmenter les performances de stockage et de traitement.

Nous avons pu atteindre nos objectifs préalablement fixé dans ce projet, et pour la vision à long terme dans laquelle des améliorations peuvent avoir lieu comme : Étaler l'ensembles des sources de données sur d'autres nouvelles sources, Sauvegarder l'ensembles des données collectées sur des périphériques de stockages distribués, appliquer la prévision et l'analyse avec les technologies de traitement et le stockage traditionnelles et faire la synthèse comparative entre les deux technologies de stockage.

BIBLIOGRAPHIE

[1] M.Karoui, G.Devauchelle, A.Dudezert, Système d'Information et prise de décision à l'ère du " Big Data " : Le cas d'une entreprise française du bâtiment. 18 Conférence Internationale de l'Association Information et Management, Lyon, France, (2013), pp 17.

[2] C.Schoch. Big? Smart? Clean? Messy? Data in the Humanities. Journal of Digital Humanities, 2013, 2 (3), pp.2-13.

[3] M. Lessard Qu'est-ce que le Big Data: http://zeroseconde.blogspot.com/2010/12/qu-que-le-big-data.html?m=0 , 2010.

[4] datastaxbig data: beyond the hype, what big data matters to you. san mateo, http://www.datastax.com/wp-content/uploads/2011/10/WP-DataStax-BigData.pdf (mars ,2012).

[5] datastaxbig data: beyond the hype, what big data matters to you. san mateo, http://www.datastax.com/wp-content/ , 2012.

[6] Big Data concept. http://www.IDC.com.

[7] B.Teboul,T.Berthier Valeur et Véracité de la donnée :Enjeux pour l'entreprise et défis pour le Data Scientist, École Militaire , France, 2015 .

[8] D. Hawley, Big Data and Digital Blog. Http://blog.businessdecision.com/bigdata/2015/01/3vopportunites-big-data , 2015.

[9] C. Scyphers, Big Data: An Overview, sur http://fr.slideshare.net/cscyphers/bd-101-slideshare , 2013.

[10] G.Jerome big-data-acces-et-traitement-des-donnees-dobservation-de-laterre http://fr.slideshare.net/GasperiJerome/big-data-acces-et-traitement-des-donnees-dobservation-de-laterre

[11] S.Singh, Big Data Analytics. International Conference on Communication, Information & Computing Technology (ICCICT) (p. 4). Mumbain, india: IEEE, 2012.

[12] IBM, guide du big data. IBM Software, sur http://www.data-business.fr/wp-content/Downloads/LivresBlancs/Guide_du_Big_Data_2013_2014.pdf (2013/2014).

[13] H. E. Chihoub, Managing Consistency for big data applications on clouds. Bretagne Atlantique: École doctorale MATISSE, 2013.

[14] Data Transformation. (s.d), sur http://www.mulesoft.com/: http://www.mulesoft.com/resources/esb/data-transformation

[15] A.Kouicem, N, Les technologies Big data, 2013.

[16] H. Shah,N. Swant, Big data application and architecture_.Edition.157, 2012.

[17] A. Borg .tutorials point. Http : //www.tutorialspoint.com/computer_whoiswho.htm/

[18] C. Scyphers, Big data platforms: an overview http://fr.slideshare.net/cscyphers/big-data-platforms-an-overview , 2012.

[19] M. Maier, Towards a Big Data Reference Architecture. Eindhoven University of Technology http://www.win.tue.nl/~gfletche/Maier_MSc_thesis.pdf , 2013.

[20] R.Chaabane. Le modèle de données relationnel http://www.ai.univ-paris8.fr/~lysop/bd/seance4-ModeleRel.pdf

[21] Meyer Léonard. Quel avenir pour le nosql ? In L'AVENIR DU NoSQL, 2014

[22] L.HEINRICH, Architecture NoSQL et réponse au Théorème CAP TGID-74 https://doc.rero.ch/record/31235/files/TDIG_74.pdf , 2012.

[23] E.KOUEDI, Approche de migration d'une base de données relationnel vers une base de données NoSQL orientée colonne https://upload.wikimedia.org/wikipedia/commons/5/59/Approche_de_migration_d'une_base_de_donn%C3%A9es_relationnelles_vers_une_base_NoSQL.pdf , 2012.

[24] NoSQL. https://www.techopedia.com/definition/27689/nosql-database

[25] C. Soullard SQLI ENTREPRISE. http : //www.technologiesebusiness.com/langages/les-bases-no-sql

[26] M. Domenjoud, Bases de données graphes : un tour d'horizon http://blog.octo.com/bases-de-donnees-graphes-un-tour-dhorizon/ , 2012.

[27] C. Soullard .SQLI ENTREPRISE. http ://www.technologiesebusiness. com/langages/les-bases-no-sql.

[28] S. Bejaoui, Le NoSQL (Not Only SQL) http://bigdataformation.com/le-nosql-not-only-sql-12 , 2015.

[29] M.TOURE, HBase : Un Système de Base de Données Orientée Colonne http://blog.mahamadoutoure.com/2015/01/hbase-un-systeme-de-base-de-donnees.html , 2015.

[30] Bright Computing, Hadoop deployment manual, 2016.

[31] V.K.V.A.MURTHY with J.Markham and D.Eadline, Sapache hadoop yarn. Moving beyond MapReduce and Batch Processing with Apache Hadoop, 2014.

[32] D.Sullivan, Getting Started with Hadoop 2.0. http://www.tomsitpro.com/articles/hadoop-2-vs-1.2-718.html , 2014.

[33] S.Alike, L'ecosysteme Hadoop, page 24 pages. LIVRE BLANC, Paris Nantes Bordeaux Washington richmond New york, 2015.

[34] J Lejeune, "une plate-forme d'exécution de programmes Map-Reduce", Cours de L'école des Mines de Nantes, 2015.

[35] L.Houdayer._Présentation Hadoop Distributed File System, https://blog.groupesii.com/aprèsentation-hadoop-distributed-le-system

[36] Démarrer avec le big data. In Guide de planification, 2013.

[37] G.D.Morales, Big Data and theWeb: Algorithms for Data Intensive Scalable Computing. Lucca, Italy: IMT Institute for Advanced Studies. http://melmeric.files.wordaprèss.com/2012/03/big-data-and-the-web.pdf , 2012.

[38] B.Renaut, Introduction à hadoop et mapreduce, (2014-2015).

[39] T. White, 'Hadoop: The De_nitive Guide', 3ieme Edition. O'Reilly Media, 2012.

[40] S.Antipolis. Introduction à Hadoop & « MapReduce » Cour de l'université nice, 2014.

[41] S.LaValle et AL. Big Data, Analytics and the path from insights to value. North Hollywood, http://www.peerevaluation.org/data/8065d07da4a77621450aa84fee5656d9/PE_doc_30147.pdf , 2011.

[42] D. Talia, Clouds for scalable Big data analytics, IEEE. http://ieeexplore.ieee.org.www.sndl1.arn.dz/stamp/stamp.jsp?tp=&arnumber=6515548 , 2013, (pp. 98-101).

[43] A. Mukherjee & AL. Shared Disk Big Data Analytics with Apache Hadoop. High Performance Computing (HiPC), 2012 19th International Conference on (p. 6). India: IEEE. ttp://ieeexplore.ieee.org.www.sndl1.arn.dz/stamp/stamp.jsp?tp=&arnumber=6507520 , 2012.

[44] T. DULL, Processing and Big Data: Take Advantage of Big Data Technologies.sur www.sas.com/: http://www.sas.com/knowledge-exchange/customer-intelligence/find-opportunities/processing- , 2013.

[45] J. Kelly, Big Data: Hadoop, Business Analytics and Beyond. wikibon.org/: http://wikibon.org/wiki/v/Big_Data:_Hadoop,_Business_Analytics_and_Beyond#Data_Processing_and_Analytics:_The_Old_Way , 2014.

[46] S. Shahrivari, Beyond Batch Processing: Towards Real-Time and Streaming Big Data. arxiv.org/: http://arxiv.org/abs/1403.3375 , 2014.

[47] L. K. Jae, Platforms for Big Data. www.cubrid.org: http://www.cubrid.org/blog/dev-platform/platforms-for-big-data/ , 2012.

[48] X. Liu, *Understanding Big Data Processing and Analytics*. developer.com/: http://www.developer.com/db/understanding-big-data-processing-and-analytics.html , 2013.

[49] P. Russom, *BIG DATA ANALYTICS.* TDWI, 2011.

[50] D. Rangarao, *What is Big Data Analytics?* http://www-01.ibm.com/: http://www-01.ibm.com/software/data/infosphere/hadoop/what-is-big-data-analytics.html , 2013.

[51] Barlow, M, *Real-Time Big Data Analytics: Emerging Architecture.* united states: O'ReillyMedia.http://books.google.fr/books?id=64Uba0n38R4C&printsec=frontcover&dq=big+data+analytics&hl=fr&sa=X&ei=QevrUqHSOPLH7AbR_YHoAw&ved=0CIsBEOgBMAk#v=onepage&q=big%20data%20analytics&f=false , 2013.

[52] S.Owen et AL, Mahout in action. (MEAP, Éd.) Manning. http://www.manning.com/owen/Mahout_MEAP_CH01.pdf , 2011.

[53] Apache, *Apache Pig*. Consulté le 02 22, 2014, sur apache.org: http://pig.apache.org/ 2013.

[54] A.Thusoo et AL. (s.d.). Hive - A Warehousing Solution Over a Map-Reduce Framework. US. http://cs.brown.edu/~debrabant/cis570-website/papers/hive.pdf

[55] M.Mayilvaganan et AL, A cloud-based architecture for Big-Data Analytics in Smart Grid: A Proposal. *IEEE International Conference on Computational Intelligence and Computing Research* (p. 4). India: IEEE, 2013.

[56] big data analytics, http://searchbusinessanalytics.techtarget.com/definition/big-data-analytics , 2014.

[57] dictionnaire-juridique. http://www.dictionnaire-juridique.com/definition/ mutation.php.

[58] C. PARAGEAUD (2014). La jungle des di_érentes distributions open source Hadoop. http : //blog.ippon.fr/2013/05/14/big-data-la-jungle-des-di_erentes-distributionsopen-source-hadoop/

[59] Fondation logicielleApacheHdoop /http ://hadoop.apache.org/docs /r1.2.1/cluster _setup.html , 2013.

[60]Fondation logicielleApacheHdoop, /https ://hadoop.apache.org/docs/current/api /overviewtree.html/ , 2013.

[61] Talend, http://www.open-source-guide.com/Solutions/Developpement-et-couches-intermediaires/Etl/Talend , 2016.

[62] Simplifier le Big Data, http://www.talend.com/sites/default/files/partners/marketing-collateral/product/Talend-Big-Data.pdf , 2015.

[63] Les quatre piliers clés d'une solution de gestion des Big Data, http://info.talend.com/rs/talend/images/WP_FR_BD_Talend_4Pillars_BigDataManagement 2013.

[64] Page web, [En ligne]. adresses URL: http://pvbookmarks.readthedocs.io/en/master/devel/OS/linux/linux_based/centos/index.html 2016.

ANNEXES

LES PLATEFORMES DE STOCKAGE DU BIG DATA ET SES TECHNOLOGIES

1- Introduction

Il s'agit de n'importe quelle plateforme qui supporte tout type de grandes données **[18]** elle permet aux utilisateurs d'accéder, de traiter, d'analyser et de construire des applications sur de grands ensembles de données. Ces plateformes dépendent des différentes opérations nécessaires au stockage, au traitement ainsi qu'aux méthodes d'analyse des Big data.

Nous allons tout d'abord présenter quelques technologies de stockage qui ont été qualifié pour être les mieux adaptées pour ce type de données et puis les technologies de Big data.

2- Système de stockage

Les infrastructures informatiques relatives au Big data doivent être capable de stocker des volumes plus élevés et de nombreux types de données, la vitesse de renouvellement doit également être prise en considération.

Diverses solutions ont été testées pour le stockage du Big data, certaines étaient parues propices, mais après un certain temps et vu le développement et l'évolution des données, ces solutions se sont confrontées à quelques problèmes. D'autres ont été utilisées et améliorées au fur et à mesure de l'évolution des données, ce qui en a fait les meilleures solutions de stockage.

i. Les technologies de stockage traditionnelles

Dans cette section nous allons parler de technologie de stockage traditionnelle.

1. Les bases de données relationnelles

Parmi toutes les technologies de stockage traditionnelles, celles qui suivent le modèle relationnel sont les plus largement utilisées. Dans ce type de modèle, les données sont représentées en tables et chaque table représente un type d'objet. L'accès aux données dans le modèle relationnel se fait en utilisant un langage de requêtes relationnel, généralement le SQL.

Avant d'être chargées dans les bases de données relationnelles, les données doivent être modélisées et analysées dans le modèle relationnel. Cette contrainte réduit considérablement la flexibilité de données et nécessite que le schéma de données soit reconnu à l'avance.

La nécessité d'analyser les données avant leur chargement et la vérification de contraintes conduit à un temps de chargement accru. De plus, le langage déclaratif SQL est un langage de haut niveau, plus facile à lire et apprendre et requiert moins de ligne de code, c'est le plus important langage et de nombreux analystes l'utilisent pour interagir avec les magasins de données **[19].**

On distingue deux manières de stockage dans les bases de données relationnelles qui se résument en :

a. Stockage en colonne

L'idée du stockage en colonne consiste à stocker les données colonne par colonne, contrairement aux bases de données qui stockent les données ligne par ligne (un tuple complet après l'autre).

Il est clair que les deux systèmes de stockage ont des avantages et des inconvénients, mais le stockage en colonne se considère meilleur dans le fait que d'une part il atteint des taux de compression plus élevés. Cette compression a comme avantage que plus de données peuvent être stockées dans la mémoire principal assurant ainsi moins de copie entre le disque et la mémoire, plus une meilleure utilisation cache CPU. D'autre part, le stockage en colonne est également conseillé pour les applications d'entrepôts de données pour la rapidité de requêtes. En effet le stockage en colonne est plus flexible quand il s'agit de modèles de données nécessitant des changements en ajoutant des colonnes à une table.

Parmi les bases de données basées sur le stockage en colonne on trouve : C-Store, MonetDB, SybaseIQ, Vertica, Infobright, et Paraccel **[19].**

b. Les bases de données en mémoire

C'est une technologie récente nettement plus rapide, comparées aux bases de données sur disque. Suite au stockage de données en mémoire, ces bases de données n'ont pas besoin de gérer la mémoire tampon ou le verrouillage. Cependant les données sont de temps en temps écrites sur le disque pour des raisons de disponibilité.

Malgré que les bases de données en mémoire soient performantes pour l'écriture et la lecture en même temps, elles restent coûteuses même si les mémoires principales deviennent de moins en moins chères car le disque reste toujours plus couteux.

La technologie de stockage de données en mémoire est beaucoup plus utilisée par les zones de stockages qui bénéficient de l'élimination du temps d'attente des E/S pour le chargement de données à partir du disque. Elle pourrait également permettre d'exécuter des

applications qui ne s'exécutaient que dans le mode de traitement par lots. Cette technique n'est bien sûr pas limitée aux systèmes de base de données relationnelle seulement, mais elle est de plus en plus appliquée **[19].**

2. Les systèmes de gestions de base de données relationnelle

Dans ce modèle, les données sont représentées par des tables, sans préjuger de la façon dont les informations sont stockées dans la machine. Les tables constituent donc la structure logique du modèle relationnel. Au niveau physique, le système est libre d'utiliser n'importe quelle technique de stockage dès lors qu'il est possible de relier ces structures à des tables au niveau logique. Les tables ne représentent donc qu'une abstraction de l'enregistrement physique des données en mémoire [**20**].

ii. Limitation des technologies de stockage traditionnelles

Le théorème CAP est une théorie soumise par Brewer en 2000. Elle a été reprise en 2003 par Seth Gilbert et Nancy Lynch du MIT qui l'ont redéfinie et elle prit le nom de théorème CAP. Cette dernière théorie stipule que dans le cadre d'un système distribué, plus spécifiquement dans l'utilisation d'une application web, une base de données ne peut pas garantir ces trois attributs en même temps **[21].**

- Consistery : *ou* consistance : tous les nœuds du système voient exactement les mêmes données au même moment **[21].**
- Availibility : ou Disponibilité : la perte de nœuds n'empêche pas les survivants de continuer à fonctionner correctement, les données restent accessibles **[22].**
- Partition tolérance : le système étant partitionné, aucune panne moins importante qu'une coupure totale du réseau ne doit l'empêcher de répondre correctement (en cas de partitionnement en sous réseaux, chacun doit pouvoir fonctionner de manière autonome) **[22].**

iii. Les nouvelles technologies de stockage

Avant de présenter le concept NoSQL comme une nouvelle approche de stockage et de manipulation de données, commençons par dire ce que le NoSQL, c'est contrairement à ses prédécesseurs relationnels de type Oracle, Sybase, MYSQL…etc **[23].**

1. Définition de NoSQL

Le NoSQL désigne une classe de système de gestion de base de données (SGBD) apparue en 2009 qui se différencie de model relationnel que l'on trouve dans des bases de données connues, il comprend une multitude de bases de données de type et d'utilité différentes. Ils sont généralement utilisés dans de très grandes bases de données qui sont particulièrement

exposées à des problèmes de performance causés par les limites de SQL et les modèles relationnels de données **[24]**. Leur avantage primaire est que, contrairement aux bases de données relationnelles, ils traitent les données non-structurées telles que les documents, e-mails, multimédias et les médias sociaux de manière efficace

2. Avantages de NoSQL

Les principaux avantages des bases de données NoSQL peuvent être résumées comme suit : une grande évolutivité (scalability) et fiabilité, un modèle de données très simple, un langage de requête très simple (primitive), et presque pas de soutien à la sécurité au niveau de la base de données.D'autres avantages sont associés à ces systèmes de données :

- Leur performance ne s'écroule jamais quel que soit le volume traité.
- Elles se migrent facilement.
- Elles s'intègrent facilement dans les Clouds du marché
- Elles possèdent un modèle extensible
- Des solutions open source et gratuite **[25].**

3. Les familles de base de données de NoSQL

Le NoSQL regroupe 4 grandes familles de base de donnée qui permettent d'offrir une représentation différente de donnée chacune dispose d'avantages et d'inconvénients en fonction e contexte qu'on souhaite l'utiliser. Ces grandes 4 familles sont : base de donne clé-valeur, orienté colonne, orienté document base de donnée orienté graphe **[25].** La **figure 1** reflète les 4 familles de NoSQL :

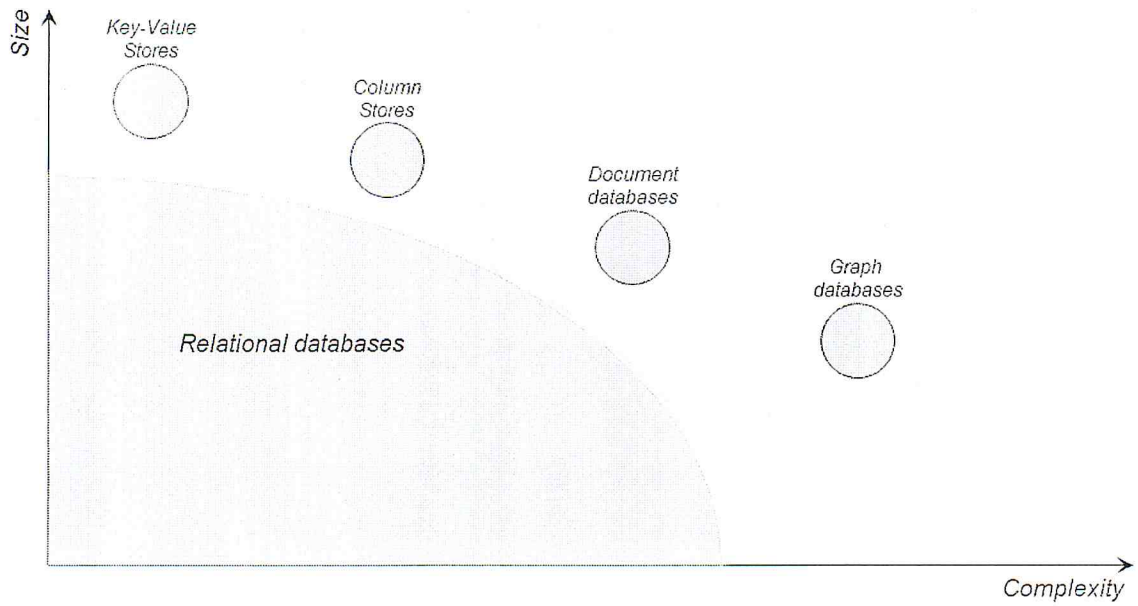


Figure 1 : Les 4 types de base de données NoSQL **[26].**

a. Les bases de données clé-valeur

C'est la base de données la plus simple elle se réduit à une simple table de hachage persistance au moyen de laquelle on pourra enregistrer ou récupérer n'importe quoi au moyen de clé **[27]**.

La **figure 2** suivante représente les bases de donnes clé-valeur :

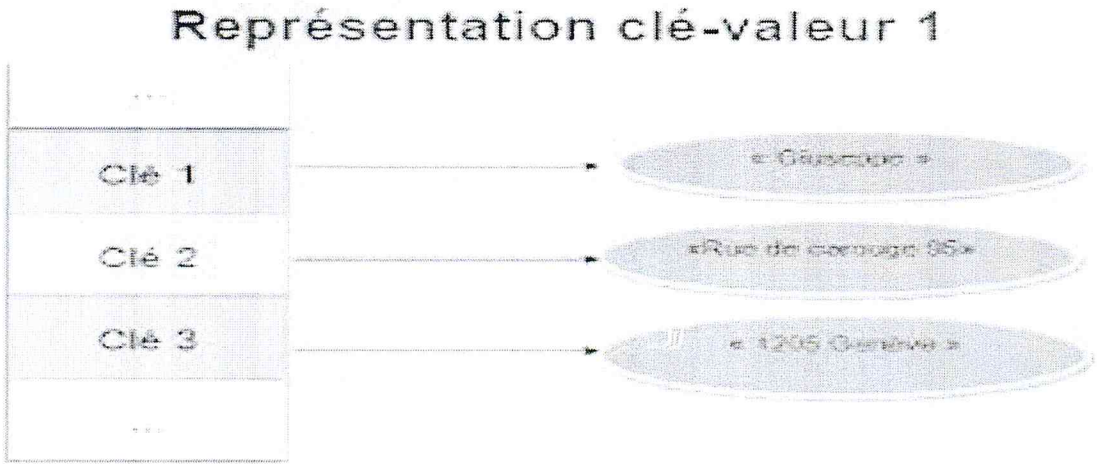
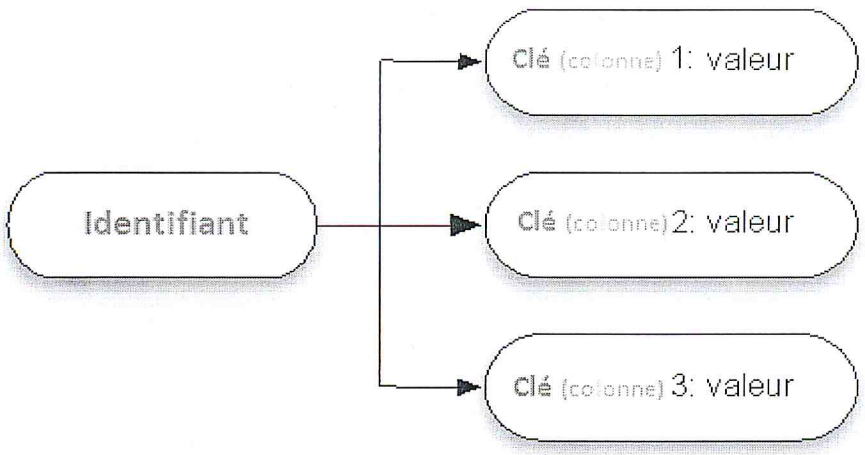


Figure 2 : Bases de donnes clé-valeur **[28]**.

b. Les bases de données orienté colonne

Elle est structurée en ligne et en colonne comme les SGBDR, les enregistrements sont toutes fois rassemblés en groupe de colonnes et les opérations d'agrégations sont très performantes Ce qui permet de bien comprendre comment les données sont organisé **[27].**

La figure suivante présente les concepts orientés colonne :



Concepts orientés colonne **[29]**.

La **figure 3** suivante illustre une comparaison entre une base de donnée relationnelle et une autre orienté colonne :

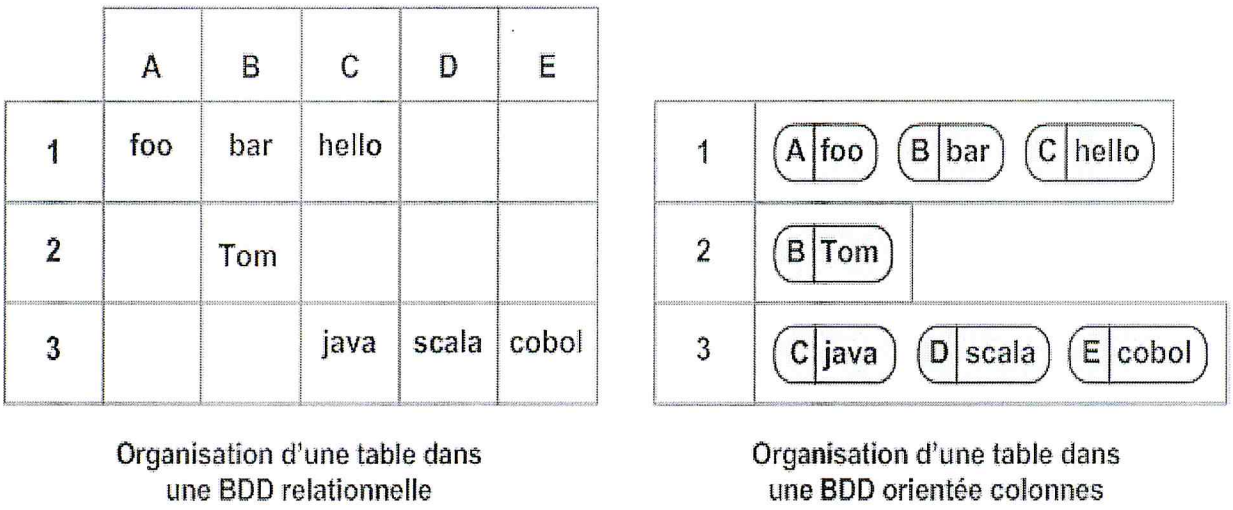
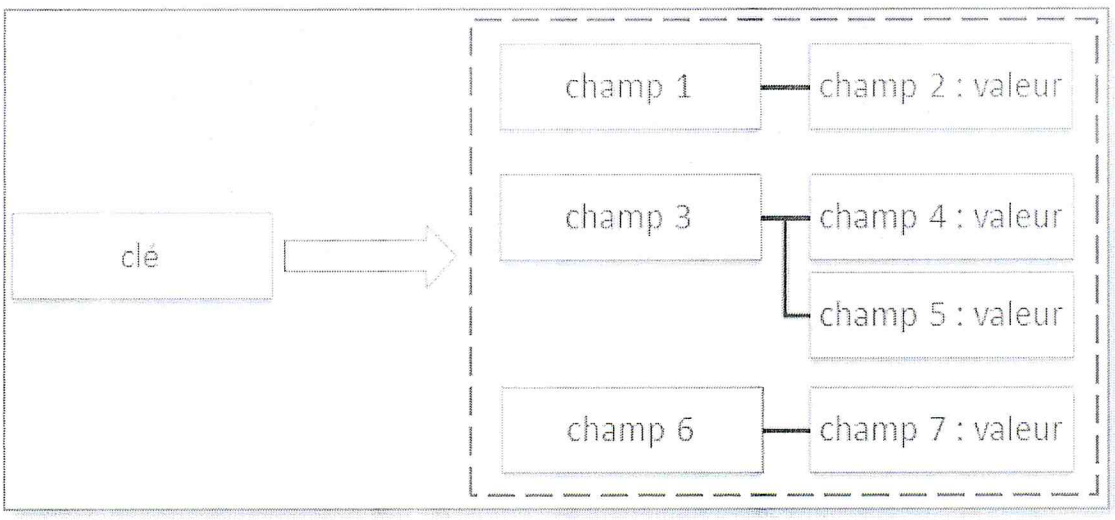


Figure 3 : Comparaison entre base de donnée relationnel et une base de données orienté colonne **[29].**

c. Les bases de données orientées document

La représentation orientée document est plus adaptée au monde de l'internet. Cette représentation est très proche de la représentation clé-valeur à l'exception faite que la valeur est représentée sous la forme d'un document, on peut trouver dans ce document les données organisées de manière hiérarchique **[27].**

La figure suivante représente une base de données orientée document



Base de données orienté document **[29].**

d. Les bases de données orientées graphe

La représentation orientée graphe est un palier à des problèmes impossibles à résoudre avec la BD relationnelle. Le cas d'utilisation le mieux connu est les réseaux sociaux où l'aspect graphe prend tout son sens **[27].** La **figure 4** suivante représente le concept base de données orienté graphe :

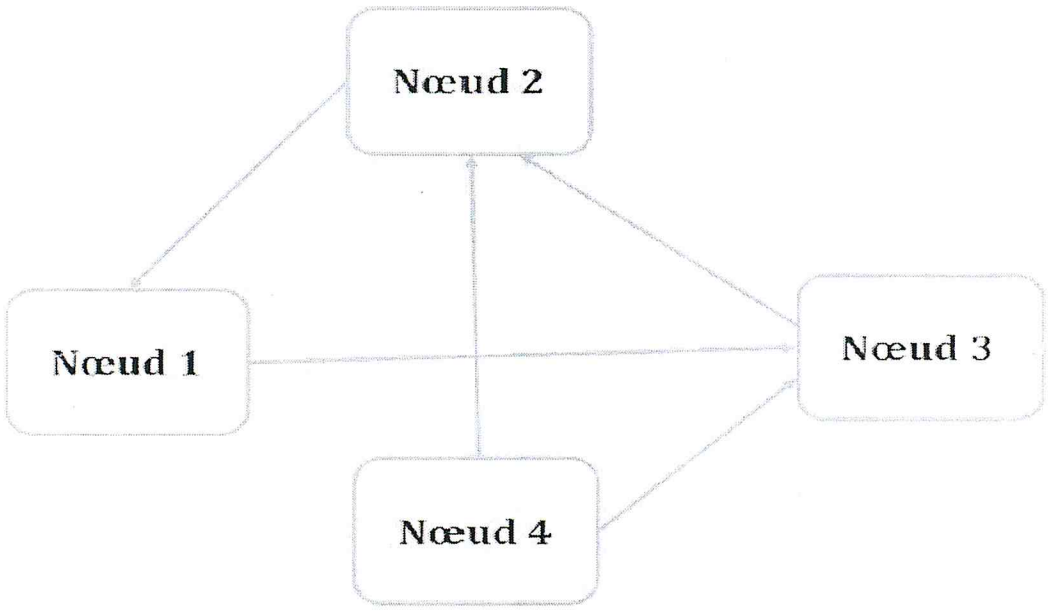


Figure 4 : Concept base de données orienté graphe **[29].**

3- Technologies de Big data

Pour avoir le meilleur bénéfice de l'utilisation des SGBD et des données stockées dans les machines, la solution est d'utiliser une interface conviviale, et pour supporter ces dernières, il faut avoir une architecture physique et un système de fichier capable de les gérer. Dans le cas du Big data on peut trouver plusieurs technologies et architectures matérielles pour supporter la gestion et le traitement d'une grosse masse de données quel que soit son type (texte, image, vidéo, base de données, etc.). Dans cette partie on va décrire les technologies les plus utilisées de nos jours pour implémenter le Big data ainsi que l'architecture matérielle supportée pour chaque technologie.

i. Le Framework Hadoop

Hadoop est d'abord un Framework Java open source d'Apache, il est la mise en œuvre de base d'une technologie de traitement de données distribuées afin d'analyser un très grand volume de données (non structurées ou structurées) stockées en Hadoop Distributed File

System (HDFS), la taille de l'ensemble de données varie généralement de plusieurs téraoctets à pétaoctets **[30].**

La **figure 5** illustre l'architecture Hadoop :

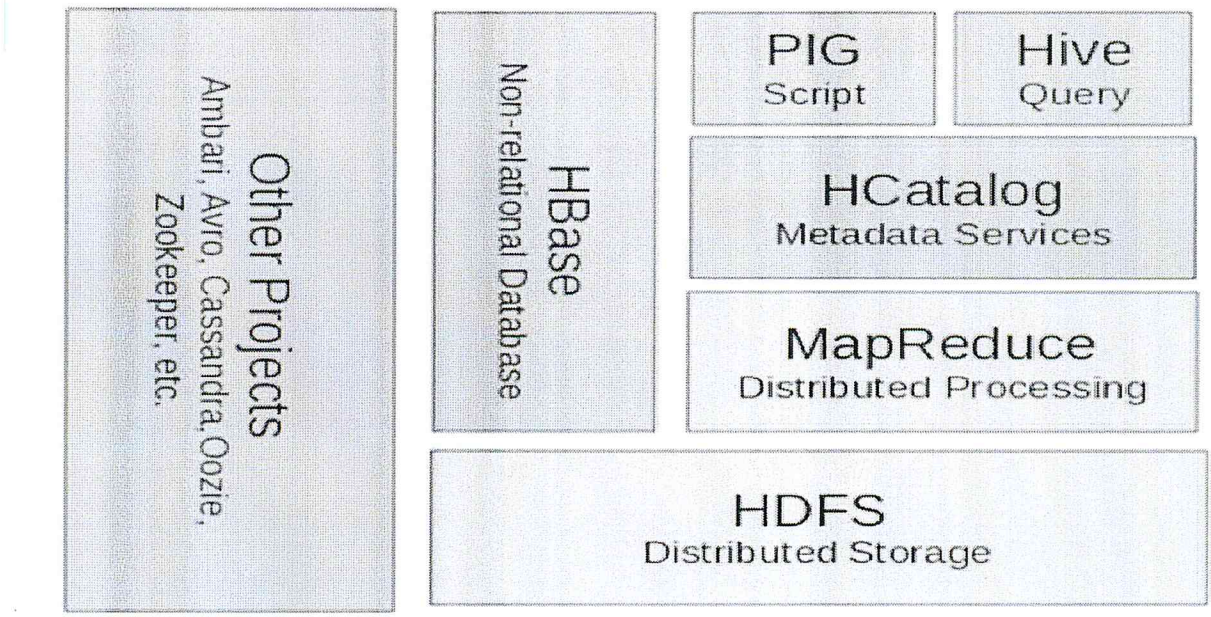
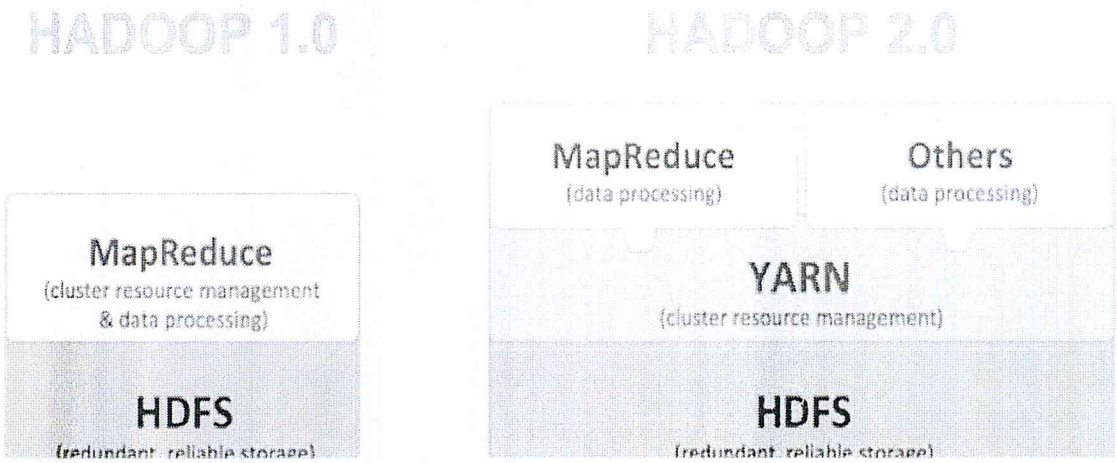


Figure 5 : Architecture Hadoop 1.0 **[31].**

Dans Hadoop 1.0 il n'y avait pas d'autres modèles (autres que MapReduce) de traitement de données, les développeurs Hadoop 1.0 réécrivirent les principaux composants du système de fichiers pour produire Hadoop 2.0. Hadoop 2.0 fournit par API YARN, permet d'exécuter les applications non-MapReduce (R, Spark, MPI, Giraph). La figure illustre les différents modules ajoutés à Hadoop 2.0.



Différents modules ajouter à Hadoop 2.0 **[32]**.

1. Hadoop Distributed File System HDFS

HDFS est un système de fichier utilisé pour stocker de très gros volumes de données (structurées ou non structurées) sur un grand nombre de machines équipées et distribuées, l'ensemble de ces machines forme ce que on appelle un cluster. C'est un système distribué, extensible et portable développé par le créateur d'Hadoop à partir du système développé par Google (GoogleFS) **[33].**

HDFS reprend de nombreux concepts proposés par des systèmes de fichiers classiques comme ext2 pour Linux ou FAT pour Windows, HDFS est le système de fichier qu'utilise Hadoop pour sa base de données distribuées, inspiré de Google File System (GFS). Nous retrouvons donc la notion de blocs (la plus petite unité que l'unité de stockage peut gérer), les métadonnées qui permettent de retrouver les blocs à partir d'un nom de fichier, les droits ou encore l'arborescence des répertoires. Il est composé d'un Namenode qui est le serveur maître et d'un certain nombre de DataNodes qui sont des serveurs contenant les données. Une même donnée est répliquée un certain nombre de fois sur différents DataNodes afin d'éviter les pertes d'information.

La **figure 6** présente la structure en nœuds du HDFS :

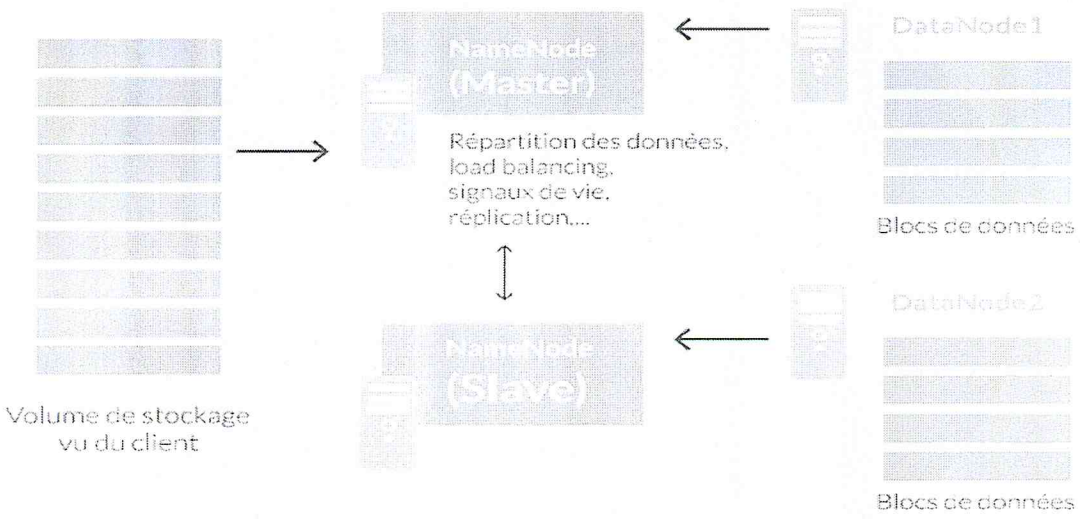


Figure 6 : La structure en noeuds du HDFS **[33]**.

2. Le paradigme Mapreduce

MapReduce est une plate-forme de programmation de logiciels de la pile Hadoop qui simplifie le traitement de grands ensembles de données et offre aux programmeurs une méthode commune pour définir et orchestrer les tâches de traitement complexes sur un système distribué **[36],** MapReduce implémente les fonctionnalités suivantes :

- Parallélisassions automatique des programmes Hadoop.
- Gestion transparente du mode distribué.
- Tolérance aux pannes.

MapReduce agit en décomposant le traitement en deux phase : une phase 'Map' et une phase 'Reduce'. Chaque phase possède des pairs clé/valeur comme entrée et sortie. Les types peuvent être choisis par le programmeur qui spécifie également deux fonctions : la fonction 'Map' et la fonction 'Reduce'. La fonction 'Map' lit l'entrée comme une liste de paires clé/valeur et applique une UDF pour chaque paire. Le résultat est une seconde liste de paires clé/valeur intermédiaires. Cette liste est triée et regroupée par clé dans la phase 'Shuffle' (brassage), et utilisée comme entrée pour la fonction 'Reduce'. La fonction 'Reduce' applique une deuxième UDF à chaque clé intermédiaire avec toutes ses valeurs associées pour produire le résultat final **[37].**

a. Architecture de Framework Mapreduce

Il est basé sur une architecture maître-esclave, représentée comme suit :

- **Le maitre Mapreduce** : Il contient le service Jobtracker qui est unique sur un Cluster, son rôle est de recevoir les tâches Mapreduce à exécuter (sous la forme d'une archive Java .jar) et organise leur exécution sur un cluster **[34].**

- **L'esclave Mapreduce** : Il contient le service TaskTracker, chaque nœud du cluster l'intègre pour un bon fonctionnement du Mapreduce **[34]**.

1. Le JobTracker

- Gère l'ensemble des ressources du système ;
- Reçoit les jobs des clients ;
- Ordonnance les différentes tâches des jobs soumis ;
- Assigne les tâches aux TaskTrackers ;
- Réaffecte les tâches défaillantes ;
- Maintien des informations sur l'état d'avancement des jobs **[34]**.

2. Un TaskTracker

- Exécute les tâches données par le Jobtracker ;
- Exécution des tâches dans une autre JVM (Child) ;
- A une capacité en termes de nombres de tâches qu'il peut exécuter ;
- Heartbeat avec le JobTracker **[34]**.

L'architecture du Framework Mapreduce est représentée par la **figure 7** :

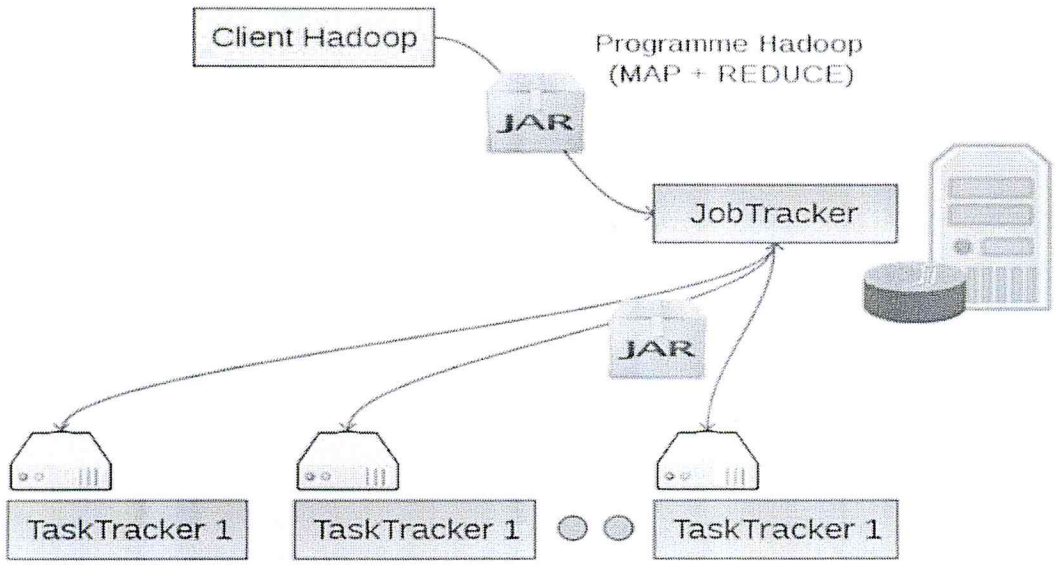


Figure 7 : L'architecture du Framework Mapreduce **[38].**

b. Fonctionnement du Framework MapReduce

L'exécution d'un job MapReduce, concerne quatre entités indépendantes **[39]** :

- Le client, qui soumet le job MapReduce.
- Le JobTracker, qui coordonne l'exécution et le partage du job en sous taches. Le

JobTracker est une application Java dont la classe principale est JobTracker.

- Les TaskTrackers, qui gèrent les tâches que le JobTracker lui a confiées. Les TaskTrackers sont des applications Java dont la classe principale est TaskTracker.
- Le système de fichiers distribué, qui est utilisée pour le partage de fichiers de travail entre les autres entités (TaskTrackers).

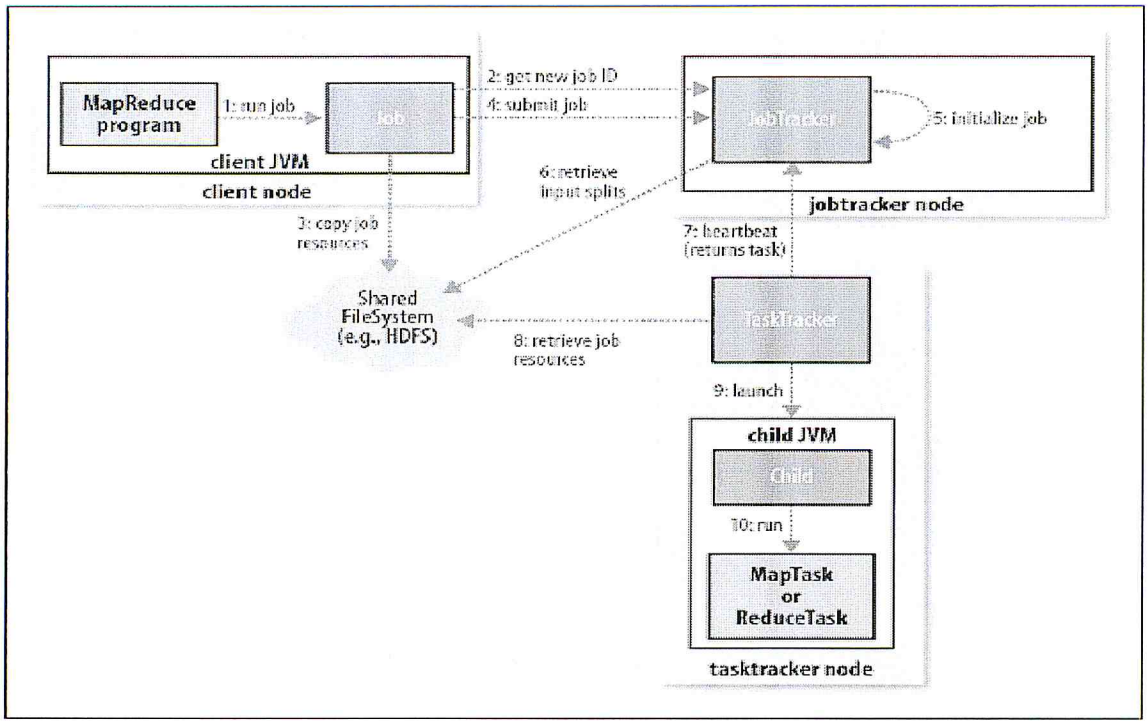


Figure 8 : Schéma de fonctionnement de MapReduce **[39].**

L'exécution de MapReduce, suit les étapes suivantes :

1. **Lancement du job**

Le processus de lancement du job est établi par JobSummitter qui effectue les Opérations suivantes (Figure 3.6.2) **[39]** :

- Demande aux JobTrackers l'ouverture d'un nouvel job ID (en appelant getNew-JobId () sur JobTracker) (étape 2 de la figure 3.6.2).
- Vérifie si les fichiers en input et output existent bien, si ce n'est pas le cas il retourne une erreur.
- Etabli la liste des sous taches qui seront données aux TaskTrackers, et renvoi une erreur en cas de problème (exemple : fichier de l'input introuvable).
- Copie les ressources nécessaires pour exécuter le travail, y compris le fichier JAR (programme mapreduce) du job, les paramètres de configuration hadoop (hdfs.xml et core-site.xml), les sous taches , la copie ce fait dans le système de fichiers du JobTracker dans un répertoire nommé d'après l'ID du job .
- Le JAR du job est copié avec une réplication haute (contrôlé par le paramètre mapred.submit.replication dans le fichier mapred-site.xml, qui est par défaut 10), de sorte qu'il y'est beaucoup de copies à travers le cluster pour les TaskTrackers (étape 3 de la figure 3.6.2).

- Indique au JobTracker que le job est prêt pour l'exécution en appelant submit-Job () sur JobTracker (étape 4 de la figure3.6.2).

2. Initialisation du job

Une fois que le JobTracker exécute le submitjob () les actions suivantes sont déclenchées **[39] :**

- Création d'un objet pour représenter le job ou la tâche à exécuter et lance un processus pour garder une trace de l'état d'avancement du job (étape 5).
- Récupération de la liste des sous taches (input split compute) crée précédemment lors de l'étape3, récupération du nombre maximum de Reduce tasks a créé, il est déterminé par la variable mapred.reduce.tasks dans {$HADOOP_HOME}/conf/mapredsite.xml. (Etape 6)

3. Affectation des tâches

- Les TaskTrackers envoient régulièrement des hearthbeat au JobTracker pour lui signifier leurs disponibilités à exécuter des jobs, si le JobTracker possède des jobs en file d'attente il le confiera au TaskTracker. (Etape 7) **[39].**
- Maintenant que le job a été attribué au TaskTracker, Tout d'abord, il localise le fichier JAR (copier lors de l'étape 3) en le copiant depuis le système de fichiers partagé, Il copie également tous les fichiers nécessaires à partir du cache distribué par l'application sur le disque local (étape 8) **[39]**.

4. Exécution du job

- _TaskRunner lance une nouvelle machine virtuelle Java (JVM, étape 9) pour exécuter chaque tâche (étape 10) **[39].**

3. La relation entre HDFS et MapReduce

Le serveur JobTracker est en communication avec HDFS ; il sait où sont les données d'entrée du programme Map/Reduce et où doivent être stockées les données de sortie. Il peut ainsi optimiser la distribution des tâches selon les données associées **[40]**.

Pour exécuter un programme Map/Reduce, on devra donc :

- Écrire les données d'entrée sur HDFS.
- Soumettre le programme au JobTracker du cluster.
- Récupérer les données de sortie depuis HDFS.

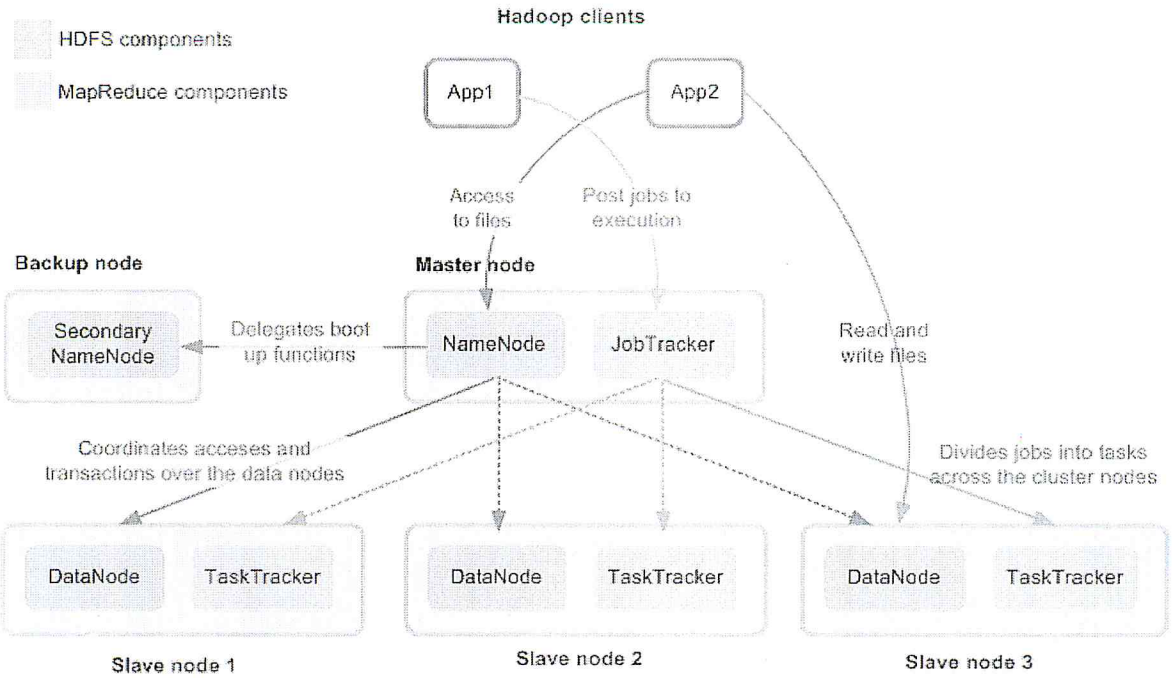


Figure 9 : Relation entre HDFS et MapReduce **[40]**

4- Conclusion

Les Big Data permettent d'accéder à de nouvelles opportunités d'affaire et de mieux contrôler les risques, mais engendrent en contrepartie une évolution considérable des modèles technologiques de l'entreprise. Les technologies qui permettent de traiter analyser et archiver ces données. Posant ainsi des défis considérables concernant leur manipulation. Ces derniers consistent en les nouvelles technologies complexes à appréhender. Comme les Big data, avant qu'elles soient analysé nécessitent d'être stockées et traitées, nous avons discuté les différentes solutions de stockage et de traitement possible pour ce type de données. Dans le chapitre suivant, nous allons présenter les méthodes de traitement relatives au Big data.

ANNEXE1: Curseur retraite age l'égale

```
DECLARE
CURSOR c_agent_cursor_NESS IS SELECT EXTRACT(year FROM
TO_DATE(AGT_DNAISS ,'DD-MM-YY')) from h_agent_r FOR UPDATE OF RE_AL;
CURSOR c_agent_cursor_SEXE IS SELECT AGT_SEXE from h_agent_r FOR UPDATE
OF RE_AL;
CURSOR c_agent_cursor_REC IS SELECT EXTRACT(year FROM
TO_DATE(AGT_DRECRUT,'DD-MM-YY')) from h_agent_r FOR UPDATE OF RE_AL;
CURSOR c_agent_cursor_ACT IS SELECT EXTRACT(year FROM
TO_DATE(SYSDATE,'DD-MM-YY')) from dual ;
v_agent_ness number ;
v_agent_sexe char ;
v_date_act number ;
v_agent_nbef number ;
v_agent_rec number ;
BEGIN
OPEN c_agent_cursor_NESS ;
OPEN c_agent_cursor_NBE ;
OPEN c_agent_cursor_SEXE ;
OPEN c_agent_cursor_ACT;
OPEN c_agent_cursor_REC ;
LOOP
FETCH c_agent_cursor_NESS INTO v_agent_ness ;
FETCH c_agent_cursor_SEXE INTO v_agent_sexe ;
FETCH c_agent_cursor_NBE INTO v_agent_nbef ;
FETCH c_agent_cursor_REC INTO v_agent_rec ;
FETCH c_agent_cursor_ACT INTO v_date_act ;
EXIT WHEN c_agent_cursor_NESS%NOTFOUND;
IF v_agent_sexe ='M' THEN
IF v_date_act -v_agent_ness >= 60 THEN
```

```
UPDATE h_agent_r SET RE_AL= 'Y' WHERE CURRENT OF
c_agent_cursor_NESS ;
UPDATE h_agent_r SET RE_SCA= 'Y' WHERE CURRENT OF
c_agent_cursor_NESS ;
UPDATE h_agent_r SET RE_PR = 'Y' WHERE CURRENT OF
c_agent_cursor_NESS ;
END IF ;
ELSE
IF v_agent_nbef > 3 THEN
v_agent_nbef :=3 ;
END IF ;
IF v_date_act -v_agent_ness + v_agent_nbef >= 55 THEN
UPDATE h_agent_r SET RE_AL = 'Y' WHERE CURRENT OF
c_agent_cursor_NESS ;
END IF ;
IF v_date_act -v_agent_ness >= 60 THEN
UPDATE h_agent_r SET RE_AL= 'Y' WHERE CURRENT OF
c_agent_cursor_NESS ;
UPDATE h_agent_r SET RE_SCA= 'Y' WHERE CURRENT OF
c_agent_cursor_NESS ;
UPDATE h_agent_r SET RE_PR = 'Y' WHERE CURRENT OF
c_agent_cursor_NESS ;
END IF ;
END IF ;
END LOOP;
END;
/
```

ANNEXE2 : Curseur retraite Sans condition d'âge

```
DECLARE
```

```
CURSOR c_agent_cursor IS SELECT EXTRACT(year FROM
TO_DATE(AGT_DRECRUT,'DD-MM-YY')) from h_agent_r FOR UPDATE OF
RE_SCA;
CURSOR c_agent_cursor1 IS SELECT EXTRACT(year FROM
TO_DATE(SYSDATE,'DD-MM-YY')) from dual ;
v_agent number ;
v_agent1 number ;
BEGIN
OPEN c_agent_cursor ;
OPEN c_agent_cursor1 ;
LOOP
FETCH c_agent_cursor INTO v_agent ;
FETCH c_agent_cursor1 INTO v_agent1 ;
EXIT WHEN c_agent_cursor%NOTFOUND;
IF v_agent1 -v_agent >= 32 THEN
UPDATE h_agent_r SET RE_SCA = 'Y' WHERE CURRENT OF c_agent_cursor ;
END IF ;
END LOOP;
END;
/
```

ANNEXE3

```
DECLARE
CURSOR c_agent_cursor_REC IS SELECT EXTRACT(year FROM
TO_DATE(AGT_DRECRUT,'DD-MM-YY')) from h_agent_r FOR UPDATE OF RE_PR;
CURSOR c_agent_cursor_NESS IS SELECT EXTRACT(year FROM
TO_DATE(AGT_DNAISS ,'DD-MM-YY')) from h_agent_r FOR UPDATE OF RE_PR;
CURSOR c_agent_cursor_SEXE IS SELECT AGT_SEXE from h_agent_r FOR UPDATE
OF RE_PR;
```

```
CURSOR c_agent_cursor_ACT IS SELECT EXTRACT(year FROM
TO_DATE(SYSDATE,'DD-MM-YY')) from dual ;
v_agent_ness number ;
v_agent_sexe char ;
v_date_act number ;
v_agent_rec number ;
BEGIN
OPEN c_agent_cursor_NESS ;
OPEN c_agent_cursor_REC ;
OPEN c_agent_cursor_SEXE ;
OPEN c_agent_cursor_ACT;
LOOP FETCH c_agent_cursor_NESS INTO v_agent_ness ;
FETCH c_agent_cursor_SEXE INTO v_agent_sexe ;
FETCH c_agent_cursor_REC INTO v_agent_rec ;
FETCH c_agent_cursor_ACT INTO v_date_act ;
EXIT WHEN c_agent_cursor_NESS%NOTFOUND;
IF v_agent_sexe ='M' THEN
IF v_date_act -v_agent_ness >= 50 AND v_date_act - v_agent_rec >= 20 THEN
UPDATE h_agent_r SET RE_PR = 'Y' WHERE CURRENT OF
c_agent_cursor_NESS ;
END IF ;
ELSE
IF v_date_act -v_agent_ness >= 45 AND v_date_act - v_agent_rec >= 15 THEN
UPDATE h_agent_r SET RE_PR = 'Y' WHERE CURRENT OF
c_agent_cursor_NESS ;
END IF ;
END IF ;
END LOOP;
```

END;

/

ANNEXE4

```
import java.io.IOException ; import java.util.* ;

import org.apache.hadoop.fs.Path ;

import org.apache.hadoop.conf.* ; ⟹ Package d'apache hadoop utilisé dans le développement

import org.apache.hadoop.io.* ;

import org.apache.hadoop.mapred.* ;

import org.apache.hadoop.util.* ;

public class RetraiteTraitement implements Tool{

 public static class MapClass  implements Mapper<LongWritable, Text, Text, Text>{

 private Text loc = new Text();

 private Text rating = new Text();

 public void map(LongWritable key, Text value, OutputCollector<Text, Text> output,
Reporter reporter) throws IOException{          La phase Map ⟹

 String[] rows = value.toString().split(";"); ⟹ Le split (séparateur) est fait par « ; ».

 StringTokenizer tokenizer = new StringTokenizer(rows[0]);

 if (rows.length >= 12){ ⟹ Pour ne pas dépasser la colonne RE_AL

 String matricule = rows[0]; ⟹ Représente l'identifiant de Matricule de l'employé

 String direction = rows[1]; ⟹ Représente l'identifiant du Direction de l'employé

String statu = rows[4]; ⟹ Représente l'identifiant d'Echelle de l'employé

 String RE_PR= rows[8]; ⟹ Représente l'identifiant du Retraite prportionnel

 String RE_SCA= rows[9]; ⟹ Représente l'identifiant du Retraite sans condition d'âge

String RE_AL = rows[10]; ⟹ Représente l'identifiant du Retraite d'âge légale.


int count = 0; ⟹ Utilisé pour compter le nombre de Y
```

int rated = 0; ⟹ **Utilisé pour indiquer si le nombre De Y dans une ligne est supérieure à 0**

for (int col= 9;col<=11; col++) ⟹ **Colonnes 9 jusqu'a 11 représente RE_SCA, RE_PR et RE_AL respectivement, les valeur de ses colonnes peuvent être Y ou N**

{

If (rows[col].equals("Y")) ⟹ **Si la valeur de la colonne est Y alors on incrémente count**

count++;

}

If (count > 0) {

rated = 1;

}

loc.set(matricule+';'+direction+";'+echelle+'); ⟹ **Modifier la variable loc avec la concaténation des trois valeurs cité entre parenthèses.**

rating.set(1 + '\t' + rated + "\t" + count); ⟹ **numTotal, numRated, rating**

output.collect(loc, rating); ⟹ **Sortie du Map avec les fréquences.**

}

}

public static class Reduce extends MapReduceBase implements Reducer<Text, Text, Text, Text>

{

public void reduce(Text key, Iterator<Text> values, OutputCollector<Text, Text> output,

Reporter reporter) throws IOException{ ⟹ **la phase Reduce**

int rating = 0; ⟹ **Avoir le nombre De Y**

int numRated = 0; ⟹ **Utilisé pour Définir si la ligne contient des Y**

int numTotal = 0; ⟹ **Utilisé pour vérifier que les lignes sont unique**

while(values.hasNext()){

String tokens[] = (values.next().toString()).split("\t"); ⟹ **le split es fait par la tabulation.**

int tot = Integer.parseInt(tokens[0]);

```
int num = Integer.parseInt(tokens[1]);
int val = Integer.parseInt(tokens[2]);
if(val > 0) //filters out markets with no data
 {
 rating = (rating*numRated + val*num)/(numRated+num);
 numRated = numRated + num;
}
numTotal = numTotal + tot;
}
If (rating>0)
output.collect(key, new Text(numTotal + "\t" + numRated + "/t"+ rating));
}
}
public static void main(String[] args) throws IOException{
JobConf conf = new JobConf(RetraiteTraitement.class);
conf.setJobName(" RetraiteTraitement");
conf.setOutputKeyClass(Text.class);
conf.setOutputValueClass(Text.class);
conf.setMapperClass(MapClass.class);
FileInputFormat.setInputPaths(conf, new Path( args[0]));
FileOutputFormat.setOutputPath(conf, new Path(args[1]));
conf.setInputFormat(TextInputFormat.class);
```

⟹ **Affecter la valeur de rated à num**

⟹ **Affecter le nombre de Y à val .**

⟹ **Affichage du résultat du job**

⟹ **Classe de org.apache.hadoop.mapred.JobConf**

⟹ **Nom du job**

⟹ **Déterminer le type de clés clé géré par le mapper**

⟹ **Déterminer le type de valeur géré par le mapper**

⟹ **Définir la class que le mapper qu'on va utiliser**

⟹ **Définir le fichier HDFS d'entré du job et le chemin des données à traiter**

⟹ **Définir le fichier HDFS de sortie du job, ou le programme affiche ses résultats**

⟹ **Déterminer le format d'entrer, ce format sera transmis au mapper**

conf.setOutputFormat(TextOutputFormat.class); ⟹ **Déterminer le format de sortir du programme**

JobClient.runJob(conf);

}

ANNEXE5

import java.io.IOException ; import java.util.* ;

import org.apache.hadoop.fs.Path ;

import org.apache.hadoop.conf.* ; ⟹ **Package d'apache hadoop utilisé dans le développement**

import org.apache.hadoop.io.* ;

import org.apache.hadoop.mapred.* ;

import org.apache.hadoop.util.* ;

public class MPWR implements TOOL{

public static class Map extends MapReduceBase implements Mapper
<LongWritible,text,text,IntWritble>{

Private final static IntWritable one = new IntWritable(1);

Private TexT map = new Text();

private String line;

public void map(LongWritable key,TEXT value, OutputCollectorw <Text,IntWritable>
output,Reparter reparter) throws IOException{ ⟹ **La phase Map**

String line = value.toString();

String[] rows = value.toString().split(";"); ⟹ **Le split (séparateur) est fait par « ; ».**

StringTokenizer tokikenizer = new StringTokenizer(rows[1]);

int count=0;

String departement= rows [1]; ⟹ **Représente l'identifiant du Direction de l'employé**

String RE_PR= rows[8]; ⟹ **Représente l'identifiant du Retraite prportionnel**

String RE_SCA= rows[9]; ⟹ **Représente l'identifiant du Retraite sans condition d'âge**

String RE_AL = rows[10]; ⟹ **Représente l'identifiant du Retraite d'âge légale.**

While(tokikenizer.hasMoreTokens()){

```
map.set(tokikenizer.nextToken());
map.set(departement+ "\t"+RE_SCA+"RE_AL); ⟹ Sortie du Map avec les fréquences.
output.collect(map,one); ⟹ Contage du nombre d'évènements.
}
}

public static class Reduce extends MapReduceBase implements Reducer<Text, Text, Text,
Text>
{
public void reduce(Text key, Iterator<Text> values, OutputCollector<Text, Text> output,
Reporter reporter) throws IOException{ ⟹ la phase Reduce
int rating = 0; ⟹ Avoir le nombre De Y
int numRated = 0; ⟹ Utilisé pour Définir si la ligne contient des Y
int numTotal = 0; ⟹ Utilisé pour vérifier que les lignes sont unique
while(values.hasNext()){
String tokens[] = (values.next().toString()).split("\t"); ⟹ le split es fait par la
tabulation.
 int tot = Integer.parseInt(tokens[0]);
 int num = Integer.parseInt(tokens[1]); ⟹ Affecter la valeur de rated à num
 int val = Integer.parseInt(tokens[2]); ⟹ Affecter le nombre de Y à val .
 if(val > 0) //filters out markets with no data
  {
  rating = (rating*numRated + val*num)/(numRated+num);
  numRated = numRated + num;
 }
 numTotal = numTotal + tot;
 }
If (rating>0)
```

output.collect(key, new Text(numTotal + "\t" + numRated + "/t"+ rating)); ⟹ **Affichage du résultat du job**

}

}

public static void main(String[] args) throws IOException{

JobConf conf = new JobConf(RetraiteTraitement.class); ⟹ **Classe de org.apache.hadoop.mapred.JobConf**

conf.setJobName(" RetraiteTraitement"); ⟹ **Nom du job**

conf.setOutputKeyClass(Text.class); ⟹ **Déterminer le type de clés clé géré par le mapper**

conf.setOutputValueClass(Text.class); ⟹ **Déterminer le type de valeur géré par le mapper**

conf.setMapperClass(MapClass.class); ⟹ **Définir la class que le maper qu'on va utiliser**

FileInputFormat.setInputPaths(conf, new Path(args[0])); ⟹ **Définir le fichier HDFS d'entré du job et le chemin des données à traiter**

FileOutputFormat.setOutputPath(conf, new Path(args[1])); ⟹ **Définir le fichier HDFS de sortie du job, ou le programme affiche ses résultats**

conf.setInputFormat(TextInputFormat.class); ⟹ **Déterminer le format d'entrer, ce format sera transmis au mapper**

conf.setOutputFormat(TextOutputFormat.class); ⟹ **Déterminer le format de sortir du programme**

JobClient.runJob(conf);

}

ANNEXE6

package wcr;

import java.io.IOException ; import java.util.* ;

import org.apache.hadoop.fs.Path ;

import org.apache.hadoop.conf.* ; ⟹ **Package d'apache hadoop utilisé dans le développement**

import org.apache.hadoop.io.* ;

import org.apache.hadoop.mapred.* ;

```
import org.apache.hadoop.util.* ;

public class MPWR implements TOOL{

public static class Map extends MapReduceBase implements Mapper
<LongWritible,text,text,IntWritble>{

private final  static IntWritable one = new IntWritable(1);

private TexT map = new Text();

private String line;

public void map(LongWritable key,TEXT value, OutputCollectorw <Text,IntWritable>
output,Reparter reparter) throws IOException{

String line = value.toString();

String[] rows = value.toString().split(";");

StringTokenizer tokikenizer = new  StringTokenizer(rows[1]);

int count=0;

String departement= rows [1];⟹ Représente l'identifiant du Direction de l'employé

String Echelle = rows[3] ;⟹ Représente l'identifiant du Direction de l'employé

String RE_PR= rows[8]; ⟹ Représente l'identifiant du Retraite Proportionnel

String RE_SCA= rows[9];⟹ Représente l'identifiant du Retraite Sans Condition
d'AgeS

String RE_AL = rows[10]; ⟹ Représente l'identifiant du Retraite Age Légale

While(tokikenizer.hasMoreTokens()){

map.set(tokikenizer.nextToken());

map.set(departement+ "\t"+Echelle+"\t"+RE_PR+"\t"+RE_SCA+"\t"RE_AL);⟹ Sortie
du Map avec les fréquences.

output.collect(map,one); ⟹ Contage du nombre d'évènements.

}

}

public static class Reduce extends MapReduceBase implements Reducer<Text, Text, Text,
Text>

{

public void reduce(Text key, Iterator<Text> values, OutputCollector<Text, Text> output,
```

```
Reporter reporter) throws IOException{ ⟹ la phase Reduce
int rating = 0; ⟹ Avoir le nombre De Y
int numRated = 0; ⟹ Utilisé pour Définir si la ligne contient des Y
int numTotal = 0; ⟹ Utilisé pour vérifier que les lignes sont unique
while(values.hasNext()){
String tokens[] = (values.next().toString()).split("\t"); ⟹ le split es fait par la tabulation.
 int tot = Integer.parseInt(tokens[0]);
 int num = Integer.parseInt(tokens[1]); ⟹ Affecter la valeur de rated à num
 int val = Integer.parseInt(tokens[2]); ⟹ Affecter le nombre de Y à val .
 if(val > 0) //filters out markets with no data
  {
  rating = (rating*numRated + val*num)/(numRated+num);
  numRated = numRated + num;
 }
 numTotal = numTotal + tot;
 }
If (rating>0)
output.collect(key, new Text(numTotal + "\t" + numRated + "/t"+ rating)); ⟹ Affichage du résultat du job
}
}
public static void main(String[] args) throws IOException{
JobConf conf = new JobConf(RetraiteTraitement.class); ⟹ Classe de org.apache.hadoop.mapred.JobConf
conf.setJobName(" RetraiteTraitement"); ⟹ Nom du job
conf.setOutputKeyClass(Text.class); ⟹ Déterminer le type de clés clé géré par le mapper
conf.setOutputValueClass(Text.class); ⟹ Déterminer le type de valeur géré par le mapper
```

conf.setMapperClass(MapClass.class); ⟹ **Définir la class que le maper qu'on va utiliser**

FileInputFormat.setInputPaths(conf, new Path(args[0])); ⟹ **Définir le fichier HDFS d'entré du job et le chemin des données à traiter**

FileOutputFormat.setOutputPath(conf, new Path(args[1])); ⟹ **Définir le fichier HDFS de sortie du job, ou le programme affiche ses résultats**

conf.setInputFormat(TextInputFormat.class); ⟹ **Déterminer le format d'entrer, ce format sera transmis au mapper**

conf.setOutputFormat(TextOutputFormat.class); ⟹ **Déterminer le format de sortir du programme**

JobClient.runJob(conf);

}

}

ANNEXE7

package wcr;

import java.io.IOException ; import java.util.* ;

import org.apache.hadoop.fs.Path ;

import org.apache.hadoop.conf.* ; ⟹ **Package d'apache hadoop utilisé dans le développement**

import org.apache.hadoop.io.* ;

import org.apache.hadoop.mapred.* ;

import org.apache.hadoop.util.* ;

public class Potentionl Total implements Tool {

public static class Map extends MapReduceBase implements Mapper <LongWritable, Text, Text, IntWritable> {

private final static IntWritable one = new IntWritable (1) ;

private Text map = new Text() ;

private String line ;

public Void map(LongWritable key, Text value, Outoutcollector<Text, Intwritable> output, Reporter reporter) Throws IOException

String line = value. toString() ;

```
String[] rows = value. toString (). split(" ;");
```
⟹ le split est fait par « ; »

```
StringTokenizer tokenizer = new StringTokenizer(rows[2]) ;
int count = 0 ;
String Matricule = rows [0];
```
⟹ **Représente l'identifiant du matricule de l'employé**

```
String Destination = rows [1];
```
⟹ **Représente l'identifiant du site visite.**

```
while (tokenizer.hasMoreTokens()_ {
map.set(tokenizer.nextToken()) ;
if(rows[3].equals("185.63.147.10") || rows[3]. equals ("104.83.17.25")
  || rows[3].equals("210.58.211.78") || rows[3]. equals("199.10.156.70") ){
  if(rows[3].equals(185.63.147.10"))
        Destination="https://www.linkedin.com";
  if(rows[3].equals(104.83.17.25"))
        Destination="https://dz.viadeo.com/fr/";
   if(rows[3].equals(216.58.211.78"))
        Destination="https://plus.google.com";
   if(rows[3].equals(199.16.156.70"))
        Destination="https://twitter.com";
map.set(Matericul+ '.'+Destination) ;
Output.collect(map, one);
}
}
public static class Reduce extends MapReduceBase implements Reducer<Text, IntWritable,
Text, IntWritable> { / phase reduce public void reduce(Text key, Iterator<IntWritable>
values, OutputCollector<Text, IntWritable> output, Reporter reporter) throws IOException {
int sum = 0 ;
while (values.hasNext()) {
sum += values.next().get() ; /agrégation ou sommation
}
output.collect(key, new IntWritable(sum)) ; / affichage du résultat.
}
```

```
}
public static

public static void main(String[] args) throws IOException{

JobConf conf = new JobConf(RetraiteTraitement.class);
```
⟹ **Classe de org.apache.hadoop.mapred.JobConf**

```
conf.setJobName(" RetraiteTraitement");
```
⟹ **Nom du job**

```
conf.setOutputKeyClass(Text.class);
```
⟹ **Déterminer le type de clés clé géré par le mapper**

```
conf.setOutputValueClass(Text.class);
```
⟹ **Déterminer le type de valeur géré par le mapper**

```
conf.setMapperClass(MapClass.class);
```
⟹ **Définir la class que le maper qu'on va utiliser**

```
FileInputFormat.setInputPaths(conf, new Path( args[0]));
```
⟹ **Définir le fichier HDFS d'entré du job et le chemin des données à traiter**

```
FileOutputFormat.setOutputPath(conf, new Path(args[1]));
```
⟹ **Définir le fichier HDFS de sortie du job, ou le programme affiche ses résultats**

```
conf.setInputFormat(TextInputFormat.class);
```
⟹ **Déterminer le format d'entrer, ce format sera transmis au mapper**

```
conf.setOutputFormat(TextOutputFormat.class);
```
⟹ **Déterminer le format de sortir du programme**

```
JobClient.runJob(conf);

}

}
```

Annexe8

Installation de Hadoop :

Pour installer Hadoop en mode Single Node, on passe par les étapes suivantes :

- **Création d'une machine virtuelle**

Pour créer notre première machine virtuelle **Master** (adresse IP= 192.168.78.111), on clique sur New Virtuel Machine sous **VMware** puis on spécifie le système utilisé par l'image du système **Centos** qu'on a téléchargé « CentOS-6.4-x86_64-bin-DVD1.iso », ensuite on affecte à notre machine 4 GO de la RAM et 20 GO de disque dur. Pour le mode Network on

utilise le **NAT** car il permet de connecter plusieurs machines virtuelles entre elle sur le même poste physique et d'avoir accès à internet. La **figure 10** présente la configuration de la machine virtuelle **Master.**

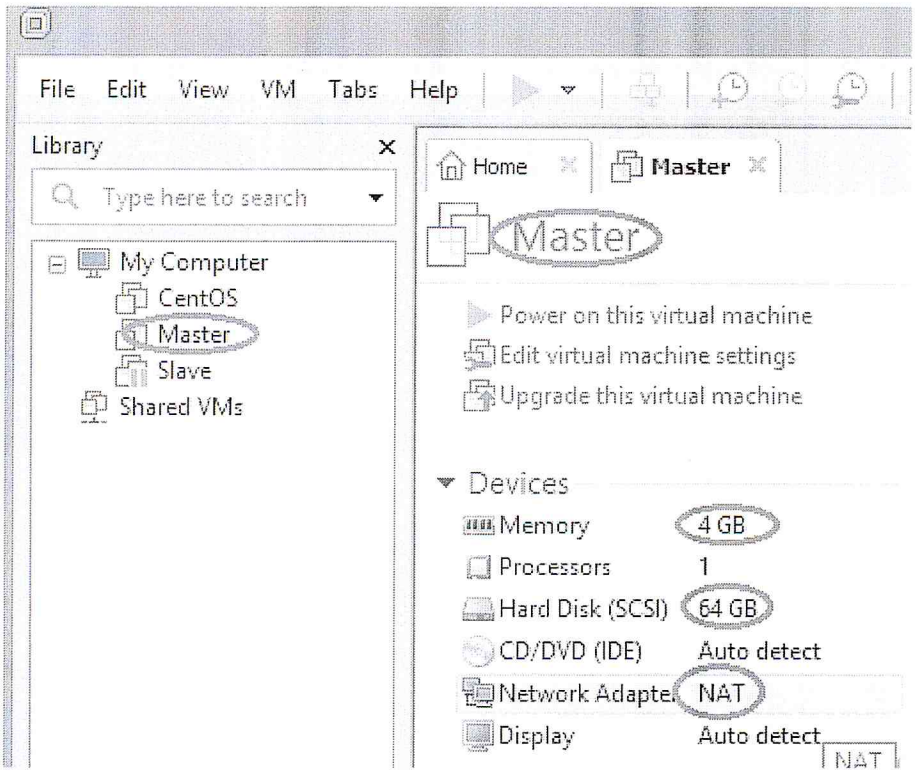


Figure 10 : Configuration de la machine virtuelle **Master**

Avant toute installation de nouveaux paquets, on a mis à jour le cache des paquets sur notre machine. La commande suivante sert à télécharger la nouvelle liste des paquets proposés par le dépôt

```
yum – update.
```

A la fin, on va modifier le nom de l'host (hostname) de la machine dans le fichier /etc/sysconfig/network.

- **Installation de JAVA**

Le composant java est un outil indispensable pour l'exécution des jobs Hadoop. Dans notre cas on a installé la version de java qui est compatible avec la version 4.7.1 du Cloudera CDH. La **figure 11** présente la version de Java installée

```
[root@Master cloudera]# java -version
java version "1.7.0_67"
Java(TM) SE Runtime Environment (build 1.7.0_67-b01)
Java HotSpot(TM) 64-Bit Server VM (build 24.65-b04, mixed mode)
[root@Master cloudera]# By ZEGAOUI Nadjet et CHAIBI Elmahdi
```

Figure 11 : Version du Java installée

Après l'installation, on va informer Linux qu'Oracle Java 1.7.0_67 sera désormais l'environnement Java par défaut en utilisant la commande suivante :

```
Sudo update-alternatives –install "/usr/bin/java" "java" "/usr/local/java/jdk 1.7.0_67/bin/java" 1
```

A la fin, on ajoute dans le fichier descripteur des variables du système./etc/profile, les lignes prescrites dans la figure suivante, pour définir les variables d'environnement JAVA_HOME. **La figure 12** présente le contenu de fichier./etc/profile.

```
 GNU nano 2.0.9              File: /etc/profile                    Modified

        fi
    fi
done

unset i
unset -f pathmunge
export JAVA_HOME=/usr/java/jdk1.7.0_67-cloudera
export ANT_HOME=/usr/local/apache-ant/apache-ant-1.9.2
export M2_HOME=/usr/local/apache-maven/apache-maven-3.0.4
export PATH=/usr/local/firefox:/sbin:$JAVA_HOME/bin:$ANT_HOME/bin:$M2_HOME/bin:$
By ZEGAOUI Nadjet et CHAIBI Elmahdi
```

figure 12 Fichier. /etc/profile

Installation de Cloudera Quickstart

Le Quickstart VM contient un cluster d'un nœud unique Hadoop, avec des exemples de données, des requêtes, des scripts et Cloudera manager pour gérer le cluster. Cette machine est disponible pour VMware, Virtual Box, et KVM. Elle offre la possibilité d'avoir et d'utiliser tous les services d'Hadoop. Au début on a installé et testé la version cloudera-quickstart-vm-4.7.0-0-vmware sous VMware. Elle nous a été très utile pour comprendre initialement le fonctionnement de base d'Hadoop.

On a remarqué que tous les services d'Hadoop ont démarré automatiquement dès qu'on lance la machine VM :

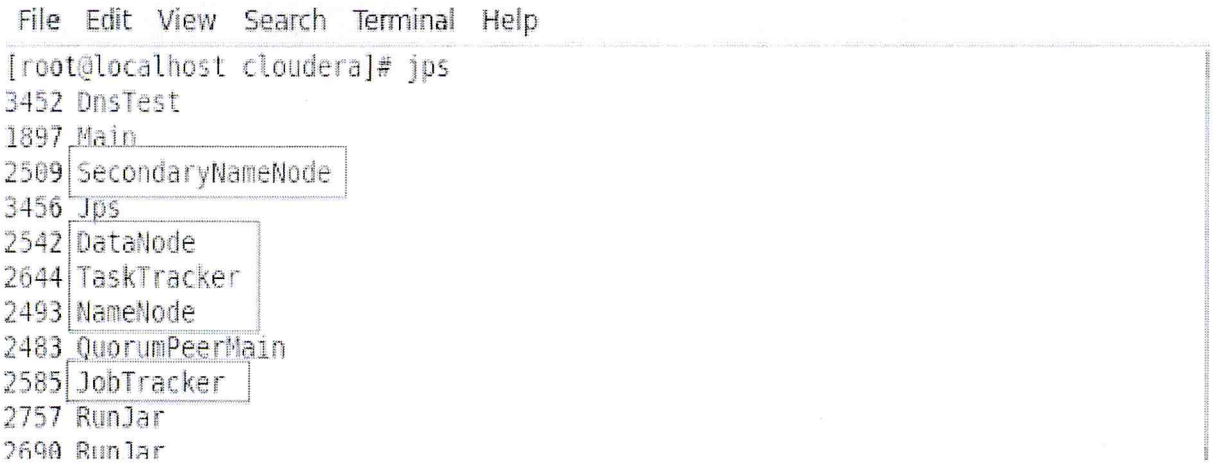


Figure 13 : Processus java d'Hadoop

Le problème rencontré dans cette méthode d'installation est lié au fait que Cloudera Quickstart ne fonctionne qu'avec un seul nœud et toutes les modifications effectuées doivent être réinitialisées après chaque redémarrage de la machine.

➢ **Installation manuelle du CDH repository**

La distribution Cloudera fournit une installation modulaire par package assez souple. Il est donc possible d'ajouter les différents composants de la distribution Hadoop via le gestionnaire de packages.

Après ces tentatives, on s'est fixé sur cette troisième option d'installation manuelle de CDH 4, en passant par les étapes suivantes :

- Télécharger le fichier de configuration de l'entrepôt de Cloudera à partir du site (Http://archive.cloudera.com/cdh4/one-click-install/redhat/6/x86_64/cloudera-cdh-4-0.x86_64.rpm) ou avec la ligne de commande :

```
wget archive.cloudera.com/cdh4/one-click-install/redhat/6/x86_64/Cloudera-cdh-4-0.x86_64.rpm
```

- Exécuter la commande pour installer le fichier téléchargé :

```
yum –nogpgcheck localinstall Cloudera-cdh4-0.x86_64.rpm
```

La **figure 14** montre la version de Cloudera CDH installé avec la version d'Hadoop utilisée.

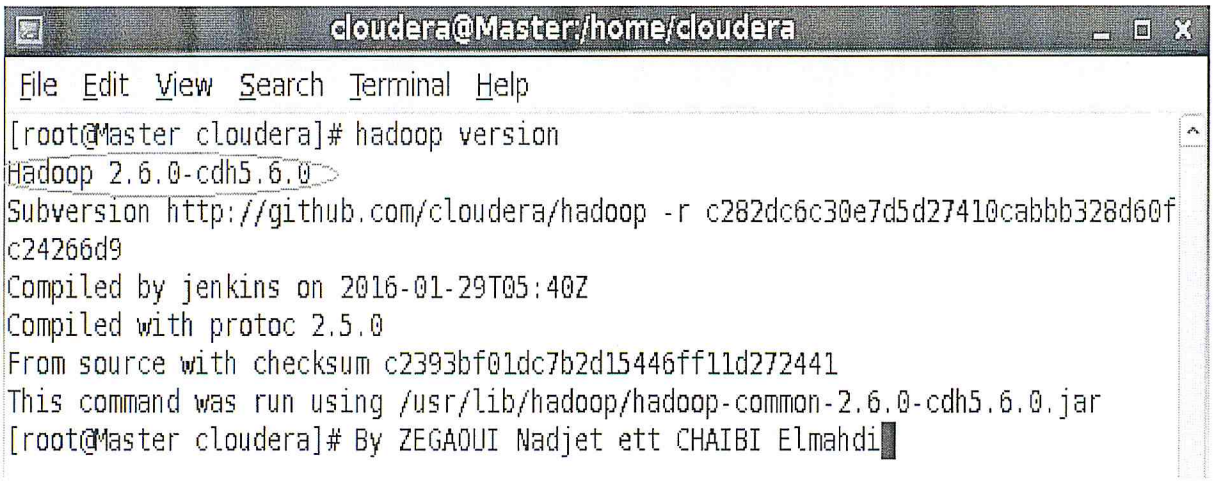


Figure 14 : Version d'Hadoop utilisée

Annexe9

Variables ou paramètres de configuration de hadoop :

Tous les fichiers de configuration d'Hadoop sont disponibles dans le répertoire /etc/hadoop/conf **[59].**

Les fichiers de configuration d'Hadoop fonctionnent sur le principe de clé/valeur : la clé correspondant au nom du paramètre et valeur à la valeur assignée à ce paramètre. Ces fichiers de configuration utilisent le format XML.

Les principaux fichiers de configuration de l'environnement hadoop sont :

- core-site.xml : contient Les paramètres de configuration de base pour Hadoop, tels que les paramètres d'E/S qui sont commun à HDFS et MapReduce **[59].**

- hdfs-site.xml : contient les paramètres de configuration pour les process HDFS : le nom du NameNode, le secondaire NameNode, et les noms des DataNodes, aussi le nombre de réplication d'un bloc qui détermine généralement le nombre de DataNodes. Beaucoup d'autres paramètres y figurent **[59].**

- mapred-site.xml : contient les paramètres de configuration pour les jobs MapReduce, l'adresse du JobTracker, et l'adresse des TaskTrackers

➢ Paramètres du fichier core-site.xml **[59].**

Tableau 1 : Paramètres du fichier core-site.xml **[59]**

Paramètre	Valeur	Exemple
fs.defaultFS	URI of NameNode	192.168.78.111

➢ Paramètre du fichier Hdfs-site.xml **[59].**

Tableau 2 : Paramètre du fichier Hdfs-site.xml **[59].**

Paramètre	Valeur	Exemple
dfs.name.dir	Chemin sur lequel les journaux de transactions sont stockés.	Liste de répertoire qui peut être séparée par des virgules pour la redondance. Rep1, Rep2, Rep3.
Dfs.replication	Nombre de réplications d'un bloc sur les DataNodes	
dfs.data.dir	Chemin local sur lequel des DataNodes stocke leur block	Liste de répertoire qui peut être séparée par des virgules pour la redondance. Rep1, rep2, rep3

➢ Paramètres du fichier mapred-site.xml **[59]**

Tableau 3 : Paramètres du fichier mapred-site.xml **[59].**

Paramètre	Valeur	Exemple
mapred.job.tracker	Host ou IP et le port du JobTracker	192.168.78.111:8021
mapred.system.dir	Chemin sur le HDFS où doit Mapreduce stocke fichiers	doit être accessible depuis le serveur et les machines

	system.	clientes.
mapred.local.dir mapred.tasktracker. {map|reduce}.tasks.maximum	Liste de chemins séparés par des virgules, sur le système de fichiers local où les données de MapReduce seront temporairement écrites. Le nombre maximum de tâches MapReduce, qui sont exécutées simultanément sur un TaskTracker donné.	Par défaut, 2 (2 cartes et 2 réduit), mais il varie en fonction de votre matériel.
dfs.hosts/dfs.hosts.exclude	Liste des datanode admis et exclu.	
mapred.hosts/mapred.hosts.exclude	Liste des tasktrackers admis et exclu.	

Annexe10

Configuration d'un cluster Hadoop :

Après avoir réalisé les étapes de la section précédente, on passe à la configuration de notre cluster Hadoop multi nœuds, on va utiliser un cluster de deux machines virtuelles, un master **Master** (Namenode, JobTracker) et un slaves : **Slave** (Datanode, TaskTracker).

Pour y arriver il faut suivre les étapes suivantes :

1. Grâce à la virtualisation, on a cloné une machine virtuelle **Slave** (adresse IP= 192.168.78.112) à partir de la machine Master (Slave est une copie du Master), après on ne change que leur noms hosts de nos machines à l'aide de la commande linux :

```
nano /etc/init.d/cloudera-quickstart-init.
```

Comme la **figure 15** nous l'indique.

```
    fi
    if [ "${DOCKER}" != 'true' ]; then
        if [ -f /sys/kernel/mm/redhat_transparent_hugepage/defrag ]; then
            echo never > /sys/kernel/mm/redhat_transparent_hugepage/defrag
        fi

        cloudera-quickstart-ip
        HOSTNAME=Master.cloudera
        hostname ${HOSTNAME}
        sed -i -e "s/HOSTNAME=.*/HOSTNAME=${HOSTNAME}/" /etc/sysconfig/network
    fi

    (
        cd /var/lib/cloudera-quickstart/tutorial;
        nohup python -m SimpleHTTPServer 80 &
    )
fi
by: ZEGAOUI Nadjet, CHAIBI Elmahdi
-- INSERT --
```

Figure 15 : Fichier etc/init.d/cloudera-quickstart-init

2. Configurer un réseau entre les deux serveurs de cluster en ajoutant les deux hôtes et leurs adresses dans le fichier /etc/hosts. Comme la **figure 16** nous l'indique :

```
if [ "${IP}" == '127.0.0.1' ]; then
    cat >> /etc/hosts <<EOF
192.168.78.111  Master.cloudera          Master          localhost       localho$
192.168.78.112          Slave.cloudera          Slave       localhost
localhost.domain
EOF
Else
    cat >> /etc/hosts <<EOF
127.0.0.1       localhost       localhost.domain
${IP}   quickstart.cloudera     quickstart

EOF
fi

by: ZEGAOUI Nadjet , CHAIBI Elmahdi
```

Figure 16 : Fichier /etc/hosts. Des deux serveurs

Cette configuration est temporaire, il faut la garder d'une manière permanente en réécrivant le fichier /usr/bin/**cloudera-quickstart-ip,** comme illustre la **figure 17** :

```
# This file is generated by /usr/bin/cloudera-quickstart-ip, which is invoked
# by /etc/init.d/cloudera-quickstart-init. If you wish to change the way that
# /etc/hosts is generated, you may edit /etc/init.d/cloudera-quickstart-init
# and hard-code a different IP address as a parameter to
# /usr/bin/cloudera-quickstart-ip, or you may comment out that line and manage
# /etc/hosts yourself.

192.168.78.111  Master.cloudera          Master          localhost       localho$
192.168.78.112          Slave.cloudera          Slave      localhost
localhost.domain
127.0.0.1       localhost       localhost.domain
127.0.0.1       quickstart.cloudera     quickstart

by: ZEGAOUI Nadjet, CHAIBI Elmahdi
```

Figure 17 : Fichier /usr/bin/**cloudera-quickstart-ip**

Configurer le SSH. Après l'établissement de la connexion entre les deux serveurs, il faut sécuriser cette connexion par la configuration du ssh. Les deux serveurs doivent être accessibles par le protocole ssh sans mot de passe (par échange de clé public et privé uniquement). Pour ce faire, il suffit de générer la clé publique RSA sur les deux serveurs :

```
cloudera@Master:/home/cloudera
File  Edit  View  Search  Terminal  Help
[root@Master cloudera]# ssh-keygen
Generating public/private rsa key pair.
Enter file in which to save the key (/root/.ssh/id_rsa):
/root/.ssh/id_rsa already exists.
Overwrite (y/n)? y
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /root/.ssh/id_rsa.
Your public key has been saved in /root/.ssh/id_rsa.pub.
The key fingerprint is:
55:32:90:0a:c8:e7:50:7c:cd:60:45:20:07:56:91:eb root@Master.cloudera
The key's randomart image is:
+--[ RSA 2048]----+
|  . +=oBX+oo .   |
|   +.++o +  +    |
|    + o o  .     |
|     . o  .      |
|      .  S       |
|       E         |
|                 |
|                 |
|                 |
+-----------------+
[root@Master cloudera]# BY ZEGAOUI Nadjet, et CHAIBI Elmahdi
```

Figure 18 : Génération d'une clé publique RSA depuis **Master**

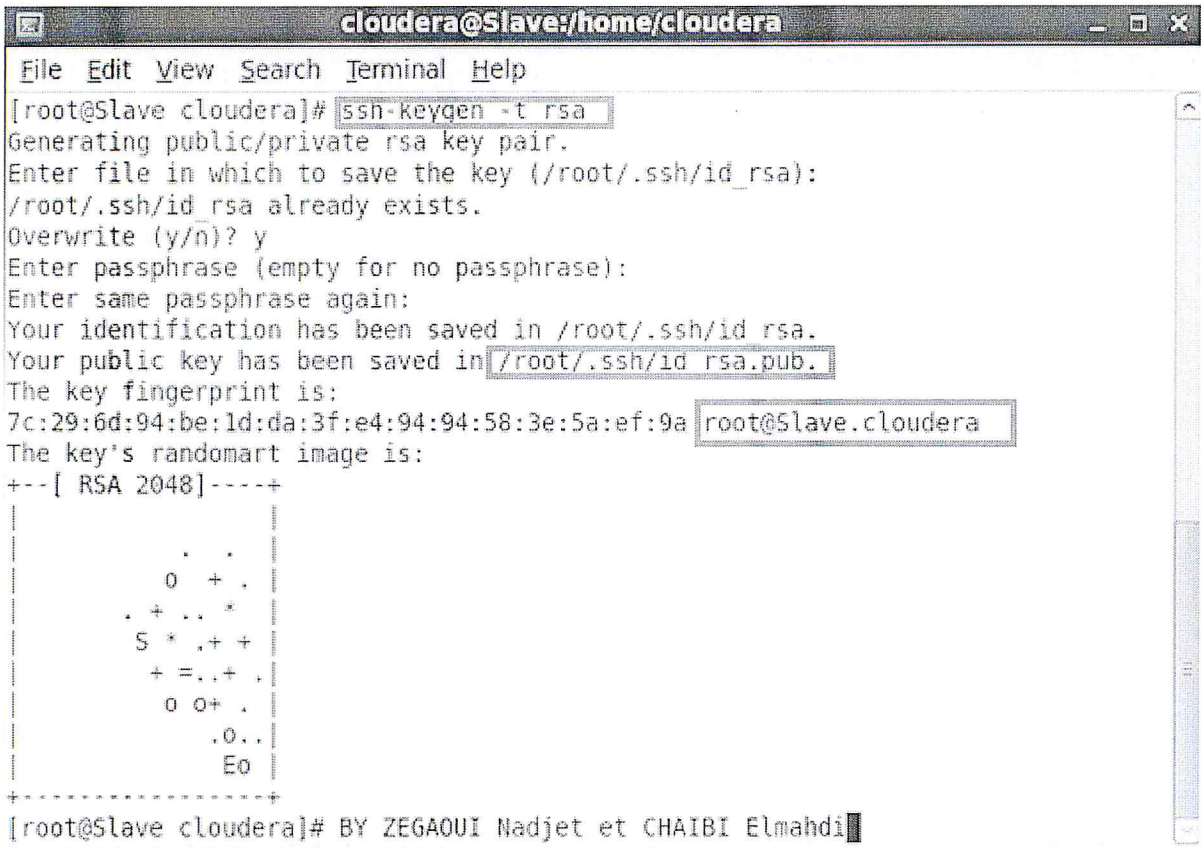


Figure 19 : Génération d'une clé publique RSA depuis **Slave**

Après cette étape, les clés doivent être échangées entre les deux serveurs :

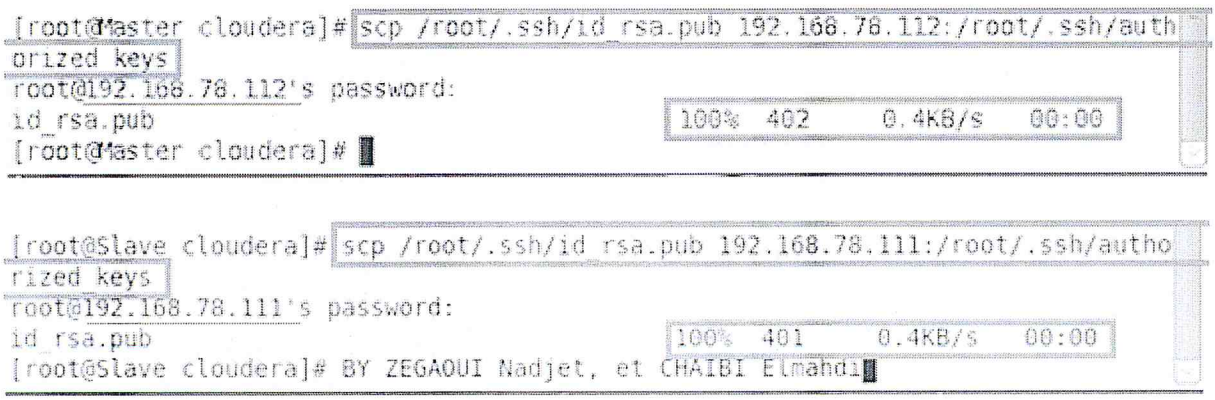


Figure 20 : Copie de la clé publique sur les nœuds **Master** et **Slave**

3. Installer les composants de la distribution Hadoop

Après avoir configuré notre cluster, il faut ajouter les différents composants requis via le gestionnaire de packages :

- Pour le **Master**, on ajoute deux packages NameNode et JobTracker avec les lignes de commandes suivantes :

```
yum install Hadoop-0.20-mapreduce-jobtracker
```

```
yum install Hadoop-hdfs-namenode
```

- Pour le **Slave**, on ajoute les packages DataNode et TaskTracker en exécutons la commande suivante :

```
yum install Hadoop-0.20-mapreduce-tasktracker Hadoop-hdfs-datanode
```

4. Configurer les fichiers d'environnement d'Hadoop

Tous les fichiers de configuration d'Hadoop sont disponibles dans le répertoire /etc/Hadoop/conf. Ces paramètres de configuration sont lus par le Framework Java d'HDFS et de MapReduce.

- **Configuration du fichier /etc/Hadoop/conf/core-site.xml (**Voir **figure 20)**

On doit spécifier l'URI du NameNode, qui devra être connue par tout le système ; pour notre cas c'est la machine **Master**.

```
<configuration>
  <property>
    <name>fs.defaultFS</name>
    <value>hdfs://192.168.78.111:8020</value>
  </property>


  <!-- OOZIE proxy user setting -->
BY ZEGAOUI Nadjet , et CHAIBI Elmahdi

  <property>
```

Figure 20 : Configuration du fichier **/etc/Hadoop/conf/core-site.xml**

- **Configuration de fichier /etc/Hadoop/conf/mapred-site.xml** (voir **figure 21** et **figure 22)**

Dans ce fichier on doit définir l'host et le numéro de port du JobTracker qui est la machine **Master**.

```
cloudera@Master:/home/cloudera
File  Edit  View  Search  Terminal  Help
 GNU nano 2.0.9      File: /etc/hadoop/conf/mapred-site.xml


<configuration>
  <property>
    <name>mapred.job.tracker</name>
    <value>192.168.78.111:8021</value>
  </property>
```

Figure 21 : Configuration de fichier **/etc/Hadoop/conf/mapred-site.xml** du **Master**

Les mêmes modifications du fichier mapred-site.xml, sont refaites sur le nœud **Slave** en ajoutant les répertoires ou les TaskTracker enregistrant les données.

```
<configuration>
  <property>
    <name>mapred.job.tracker</name>
    <value>192.168.78.111:8021</value>
  </property>

  <property>
    <name>mapreduce.framework.name</name>
    <value>yarn</value>
```

Figure 22 : Configuration de fichier **/etc/Hadoop/conf/mapred-site.xml** du **Slave**

- **Configuration de fichier /etc/Hadoop/conf/hdfs-site.xml** (voir **figure 23)**

Ce fichier doit être configuré dans les deux nœuds, il contient les paramètres de configuration pour le processus **HDFS**.

```
<configuration>
  <property>
    <name>dfs.replication</name>
    <value>2</value>
  </property>

BY ZEGAOUI Nadjet, et CHAIBI el mahdi
```

Figure 23 : Configuration de fichier **/etc/Hadoop/conf/hdfs-site.xml**

5. Démarrer les services d'Hadoop en exécutant les commandes suivantes :

- Avant de démarrer le serveur Hadoop dans le nœud master (NameNode), on doit formater le système de fichiers en HDFS.

```
sudo –u hdfs Hadoop namenode –format
```

- Pour démarrer les services HDFS ainsi que MapReduce dans le noeud master NameNode/JobTracker, on exécute :

```
/etc/init.d/Hadoop-0.20-mapreduce-jobtrackerstart
```

```
/etc/init.d/Hadoop-0.20-mapreduce-jobtrackerstart
```

- Pour démarrer les services DataNode et TaskTracker dans le noud Slave, on exécute :

```
/etc/init.d/Hadoop-hdfs-datanode start
```

```
/etc/init.d/Hadoop-0.20-mapreduce-tasktracker start
```

6. Vérification de l'état des services d'Hadoop :

Pour vérifier l'exécution des processus Java, il faut lancer la commande **jps** dans le terminal linux pour le master (voir la **figure 24**), et pour le Slave (voir la **figure 25)**

```
[root@Master cloudera]# jps
8399 Jps
2810 Bootstrap
4235 HRegionServer
5436 Bootstrap
2368 JobTracker
3223 HMaster
3466 RunJar
4012 HistoryServer
2857 JobHistoryServer
5184 Bootstrap
3126 ResourceManager
2655 SecondaryNameNode
5490 -- process information unavailable
2199 QuorumPeerMain
3393 ThriftServer
5457
2417 DataNode
2493 JournalNode
3996 Bootstrap
2938 NodeManager
2571 NameNode
3304 RESTServer
[root@Master cloudera]# BY ZEGAOUI Nadjet et CHAIBI Elmahdi
```

Figure 24 : Processus java d'Hadoop sur le nœud **Master**

```
[root@Slave init.d]# jps
5083 -- process information unavailable
3164 RunJar
2540 Bootstrap
3017 RESTServer
3704 Master
2004 QuorumPeerMain
6166 DataNode
2226 JournalNode
3607 Bootstrap
2379 SecondaryNameNode
2685 NodeManager
3909 Worker
3631 HistoryServer
2114 TaskTracker
6210 Jps
5035 Bootstrap
5059
3105 ThriftServer
2843 ResourceManager
[root@Slave init.d]# BY ZEGAOUI Nadjet et CHAIBI Elmahdi
```

Figure 25 : Processus java d'Hadoop sur le nœud **Slave**

- Il est possible de vérifier le bon fonctionnement du notre cluster à l'aide des interfaces WEB. Par défaut, elles sont disponibles sur le Web UI pour le NameNode : http://192.168.78.111:50070 , la **figure 26** présente l'état de notre cluster :

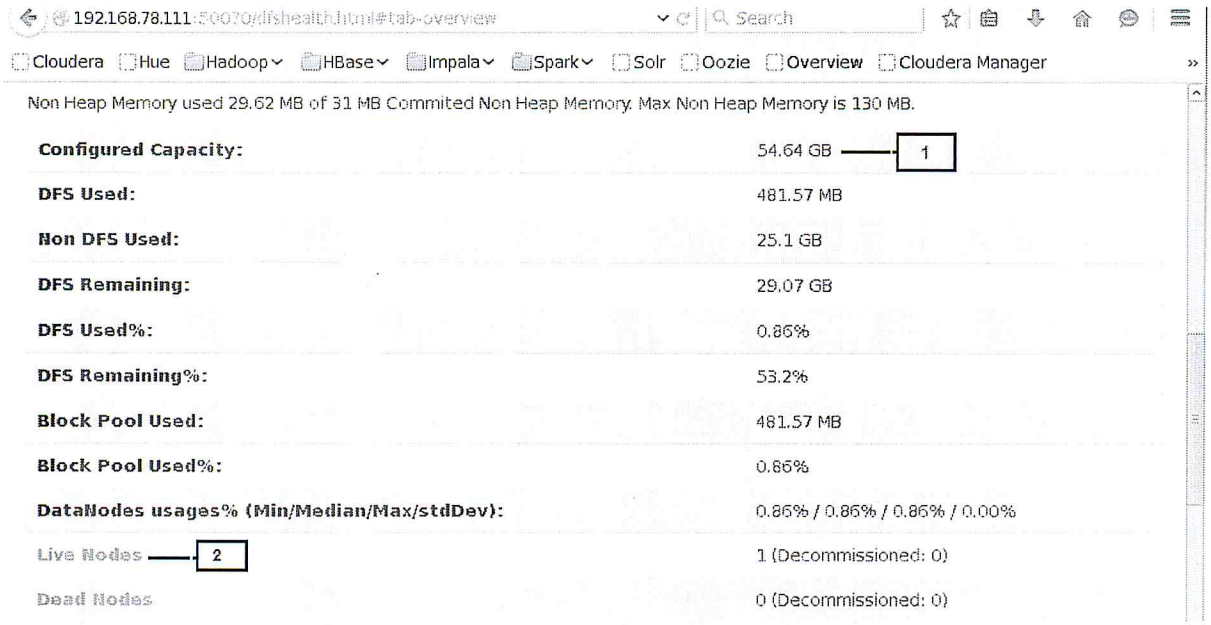


Figure 26 : Etat de notre cluster

Le commentaire N°1 de la **figure 26** indique une capacité totale de 54.64 GO, aussi le commentaire N°2 indique l'existence d'un seul DataNode en cliquant sur le « live Nodes »

La Figure 27 presente l'etat du **Slave**

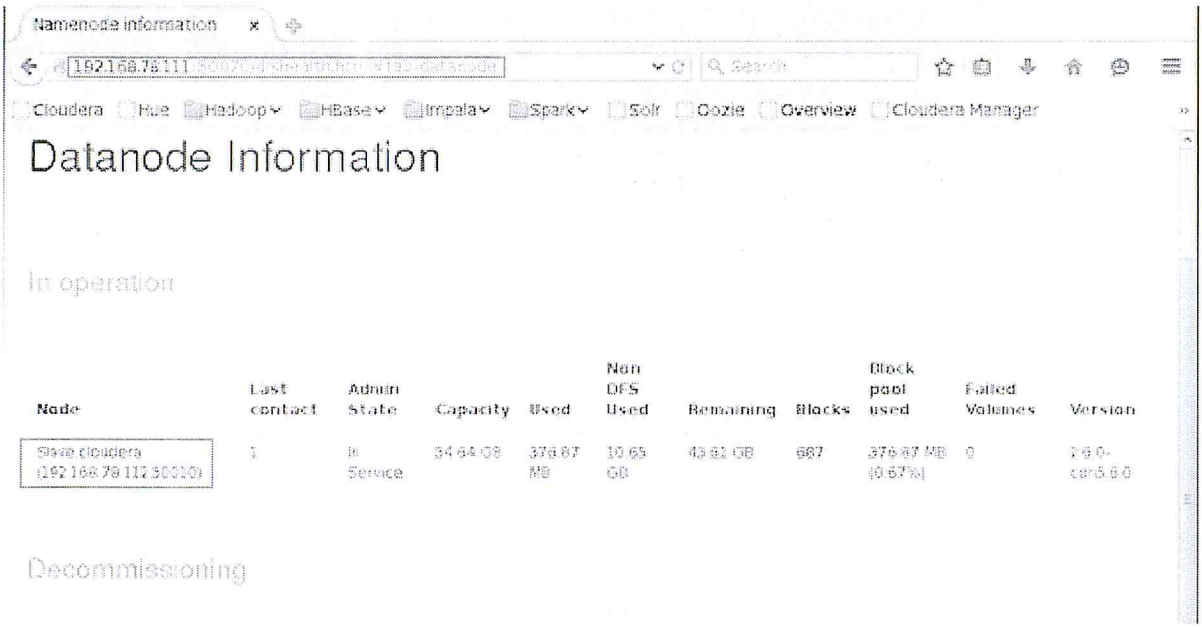


Figure 27 : Etat du **Slave**

L'architecture du cluster Hadoop est prête pour l'exécution des Jobs MapReduce, il reste bien sûr à implémenter le job MapReduce en Java.

Annexe11

Environnement de développement des programmes MapReduce :

Comme outil de programmation nous allons travailler avec Eclipse Java, pour cela nous allons commencer par importer les packages Hadoop.

La **figure 28** illustre l'environnement de développement des jobs mapreduce, en rouge le nom des principaux packages hadoop que nous allons utiliser :

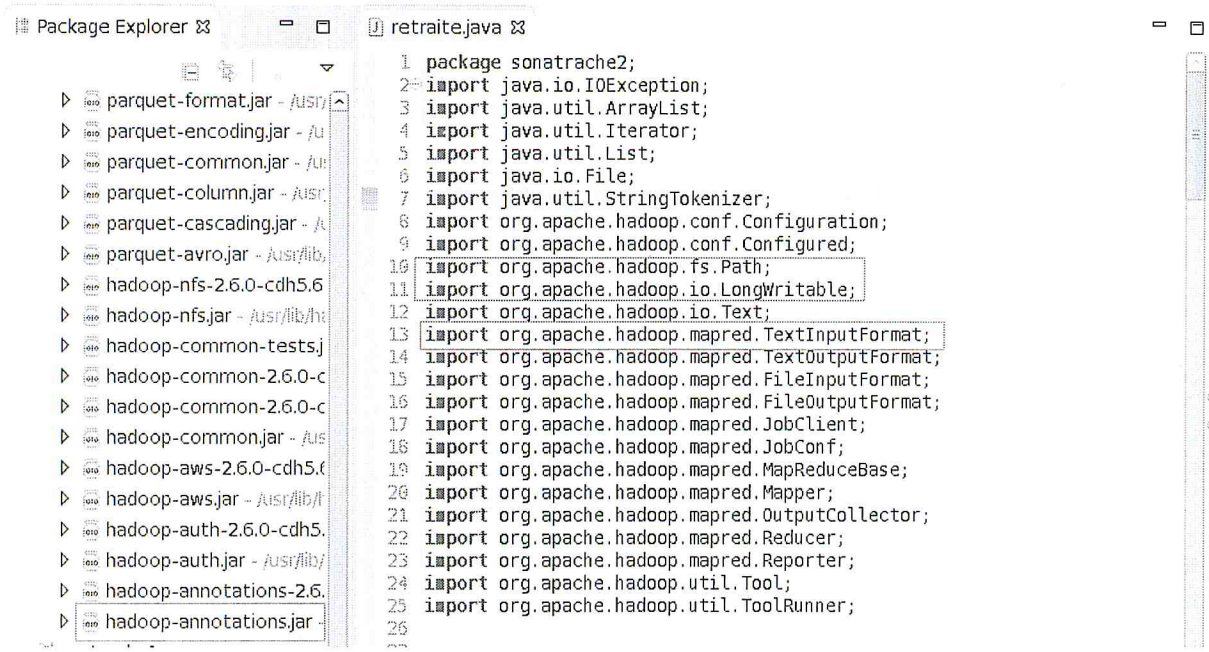


Figure 28 : L'environnement de développement des jobs mapreduce

Nous allons notamment utiliser les packages org.apache.hadoop.io, org.apache.hadoop.fs et org.apache.hadoop.mapred, pour cela nous allons faire un bref topo sur les principales classes de ses packages.

- **Package org.apache.hadoop.fs [60]:**

Ce package gère les systèmes de fichier (notamment HDFS). La **table 4** présente les principales classes du package HDFS :

Tableau 4 : Les principales class du package HDFS **[60].**

Classe	Usage
Path	Spécifie les noms des fichiers ou de répertoires dans un système des fichiers.

FSDataInputStream	Utilis2E en générale pour la lecture, il encapsule les données, et fait le lien entre le NameNode et les DataNodes.
FSDataOutputStream.	Utiliser lors de l'écriture d'un fichier, il se charge de la coordination entre le NameNode et les DataNodes lors de l'écriture

- **Package org.apache.mapred [60] :**

Utilisé pour l'implémentation du mapreduce, La **table 5** présente les principales classes du package HDFS :

Table 5 : Les principales class du package de l'implémentation de mapreduce **[60].**

Classe	Usage
InputFormat<K, V>	InputFormat décrit l'entrée d'un job MapReduce (keys, value)
Mapper<K1, V1, K2, V2>	Implémente la fonction Map, reçois une liste de key, value en input et génère une autre liste de Key, value.
Reducer<K2, V2, K3, V3>	Implémente la fonction Reduce , reçois une liste de Key, value en input (k2,v2) et génère une liste réduite (k3,v3).

- **Package org.apache.hadoop.io: [60]**

Ce package est utilisé lors de la lecture et de l'écriture des données sur le réseau, bases de données, et les fichiers, la **table 6** présente les principales class de ce package sont :

Table 6 : Les principales class du package pour la lecture et l'écriture des données **[60].**

Classe	Usage
IntWritable	Pour la lecture et écriture d'un entier
DoubleWritable	Pour la lecture et écriture d'un double

DataOutputOutputStream	Mise en œuvre Output Stream qui encapsule un Data Output.