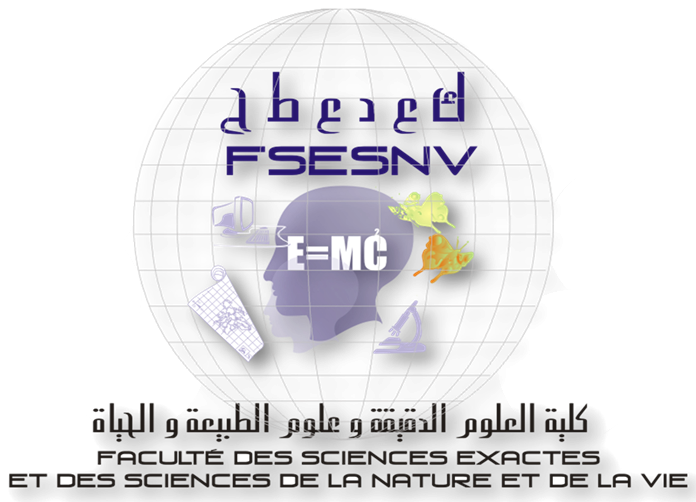
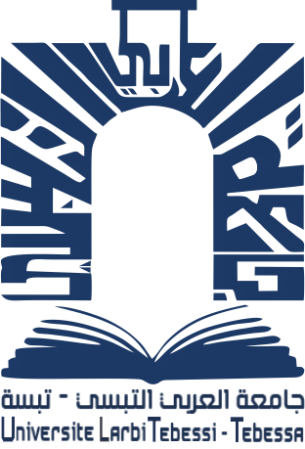
République Algérienne Démocratique et Populaire



Ministère de l’Enseignement Supérieur et de la Recherche Scientifique

Université de Larbi Tébessi –Tébessa-

Faculté des Sciences Exactes et des Sciences de la Nature et de la Vie

Département : Mathématique et Informatique

MEMOIRE DE MASTER

Domaine: Math et Informatique

Filière: Informatique

Option: Système Information

Thème:

Modélisation du processus ETL au niveau conceptuel, logique et physique

Présenté par:

LAIFA Hassiba TABIOU Imane

Devant le jury:

M.Derdour M.C.B Université de Tébessa Président

L.BRADJI M.C.A Université de Tébessa Encadreur

S.Khediri M.A.A Université de Tébessa Examinateur

Date de soutenance: 29/05/2016

Note :

Mention :

**Promotion : 2015 – 2016**

*DEDICACE*

*Je dédie ce présent travail à,*

Ma mère, qui depuis ma naissance, n’a cessé de me bercer avec des bons conseils et sans doute celle qui me réconforte et m’encourage dans les moments les plus difficiles de ma scolarité.

Mon père, celui qui ne se lassait jamais pour aménager tous ses efforts pour que je sois quelqu’un dans la vie.

Quoi que je fasse, je ne pourrais jamais vous récompenser pour les grands sacrifices que vous avez faits et continuez de faire pour moi. Aucune dédicace ne saurait exprimer mes grandes admirations, mes considérations et mes sincères affections pour vous.

Mes frères, Messaoud et sa femme Nadia, Sofiane, Abderahim et Riadh.

Mes sœurs, Aouatef et son mari Sedik, Souhaila et son mari Mohamed et Hadjer.

Les poussins de la famille, Louai, Alaa, Israa ,Tasnim rt Mohamed Wassim.

Mes oncles, tantes paternelle et maternelle, tous cousins et cousines.

Mes proches

Mes amies

*Mes professeurs.*

*Ma binôme, TABIOU Iman*

*Mes camarades de la promotion.*

*Tous ceux qui de près ou de loin m’ont aidé à réaliser ce mémoire.*

LAIFA Hassiba

|  |
| --- |
| TABIOU Imane |

|  |
| --- |
| A tous ceux qui me sont chers … |

*DEDICACE*

Remerciment

Ce rapport est le résultat d'une période d'immersion dans le milieu professionnel. En préambule, je souhaite adresser mes remerciements aux personnes que j'ai côtoyées durant cette période et qui ont ainsi contribué à l'élaboration de ce projet.

Tout d'abord de grands remerciements à Monsieur L.BRADJI , notre encadreur pour sa disponibilité, ses conseils et ses remarques avisées et l'encadrement apporté durant ce travail ;

Je remercie Y .BOUAALLEG, pour son aide et leur accompagnement malgré leur emploi de temps chargé.

Tous ceux que nous avons omis de citer ici, et qui de près ou de loin ont contribué au bon déroulement de ma formation.

Résumé

La notion de ***big data*** est un concept s’étant popularisé en 2012 pour traduire le fait que les entreprises sont confrontées à des volumes de données (data) à traiter de plus en plus considérables et présentant un fort enjeu commercial et marketing[[1]](#footnote-1),de nouvelles techniques ont été proposées pour améliorer le stockage et le traitement de ces données massives, comme le projet ***Hadoop***, Ces données ne peuvent être traiter avec gestion de base de données classique c’est pour ça que sont nées les bases de données ***NoSQL*** (Not Only SQL)

Le **Data Warehouse**, ou entrepôt de données, est une base de données dédiée au stockage de l'ensemble des données issues depuis les bases de production. Il est alimenté en données grâce notamment aux outils d »extraction, transformation et chargement connu sous le terme processus ***ETL*** ( Extract Transform Load ).

Nous avons travaillé sur la conception d’une architecture modélise le processus ***ETL*** basé ***MapReduce***, ainsi, nous avons présenté une implémentation du notre architecture.

Mots clé : big data , hadoop , nosql , entrepot de données , ETL , MapReduce

Abstract

The concept of ***big dated*** is a concept being popularized to translate the fact in 2012 that the companies are confronted with volumes of data (dated) to treat increasingly considerable and presenting a strong commercial challenge and marketing, novel methods were proposed to improve storage and the treatment of these massive data, like the ***Hadoop*** project, These data cannot be to treat with data base management classic it is for that were born the databases ***NoSQL*** (Not Only SQL)

***Data warehouse***, is a database dedicated to the storage of the whole of the data resulting since the bases from production. It is fed in data grace in particular to the tools D “extraction, transformation and loading known under the term process ***ETL*** ( Extract Transform Load )

We worked on the design of an architecture models process ***ELT*** based ***MapReduce***, thus, we presented an implementation of our architecture.

Keywords: Big Data,hadoop,nosql, ETL, Extraction, Transformation, Loading, datawarehouse, mapreduce

ملخص

كلمة ***Big data*** هو مفهوم شاع سنة 2012 ليعكس حقيقة أن الشركات تواجه كم هائل من البيانات على نحو متزايد وقد اقترحت تقنيات جديدة لتحسين تخزين ومعالجة هذه البيانات الضخمة مثل مشروع ***Hadoop*** لا يمكن أن تعامل هذه البيانات مع إدارة قواعد البيانات التقليدية لهذا خلقت قواعد البيانات )***NoSQL*** ليس فقط (***SQL***

مستودع البيانات، هو قاعدة بيانات مخصصة لتخزين كل البيانات القادمة من قواعد الانتاج. هو مدعوم بفضل البيانات إلى أدوات معروفة باسم ***ETL*** "استخراج، تحويل وتحميل" المدى (استخراج، تحويل، وتحيل)  
لقد عملنا على تصميم نموذج لهذه العملية عملية **ETL** أساس *MapReduce* .

|  |
| --- |
| **Table de matières** |

**Introduction générale**

Introduction générale  1

**Chapitre 1 : Big Data**

1. Définition Du Big Data 2

2. Caractéristique Du Big Data 3

2.1. Volume 3

2.2. Vélocité 4

2.3. Variété 4

2.4. Véracité 4

3. Technologie Big Data 4

3.1. Mapreduce 4

3.1.1. Définition  4

3.2. Apache Hadoop 6

3.2.1 Définition 7

3.3. Hadoop Distributed File System (Hdfs) 7

3.3.1. Namenode 8

3.3.2. Secondary Namenode 9

3.3.3. Datanode 9

3.4. Architecture 10

3.4.1. Définition 10

3.4.2. Exécution D'une Tâche 11

4. Les Bases De Données Nosql 11

4.1. Base De Données Orientée Clé- Valeur 11

4.2. Base De Données Orientée Document 12

4.3. Base De Données Orientée Colonne 12

4.4. Base De Données Orientée Graphe 13

Conclusion  13

**Chapitre 2 : Entrepôt de données**

Introduction 15

1. Différence Entre L'informatique De Production Et L'informatique De

Décision 15

1.1. L'informatique De Production 16

1.2. L'informatique Décisionnelle 17

1.2.1. Le Système Décisionnel Dans L’organisation 17

1.2.2.Traitement Décisionnelle 18

1.2.3. Base De Données Décisionnelle 18

2. Entrepôt De Données 19

2.1. Définition 19

3. Conception D'un Entrepôt De Données 20

4. Modélisation D’un entrepôts De Données 21

4.1. Niveau Conceptuel 21

4.1.1. Le Schéma En Etoile 22

4.1.2. Le Schéma En Flocon 22

4.1.3. Le Schéma En Constellation 23

4.2. Niveau Logique 24

4.3. Niveau Physique 25

6. Restitution Et Analyse OLAP 25

Conclusion 26

**Chapitre 3 : Processus ETL**

Introduction 27

1. Processus ETL 27

1.1. La Phase D'extraction 27

1.2. La Phase De Transformation 28

1.3. La Phase De Chargement 28

2. Modélisation De Processus ETL 29

2.1. Modélisation Conceptuelle 30

2.2. Modélisation Logique  31

2.3. Modélisation Physique  31

3. Travaux Antérieurs 32

4. Etude Comparative 35

4.1. Critère De Comparaison 35

5. Classification Des Travaux Sur Le Processus ETL 38

5.1. Processus ETL Classique 38

5.2. Processus ETL Base Sur Le Modèle Mapreduce 39

5.3. Partitionnement des données 40

Conclusion 4 0

**Chapitre 4 : Contribution Et Implémentation**

Introduction 42

1. Présentation Des Environnements 42

1.1. Cloudera Cdh 42

1.2. Machine Virtuelle 42

1.3. Hive 43

2. Les Sources De Données 43

2.1. Fichier Csv 43

2.1.1. Définition 43

3. Contribution 43

3.1. L’Architecture 44

3.2. Implémentation 47

3.2.1. Importation Des Données 47

3.2.2. Extraction 48

3.2.3. Transformation 49

3.2.4. Chargement 50

Conclusion 53

Conclusion générale 54

Bibliographie 55

**TABLE DES FIGURES**

**Chapitre 1. Big Data**

|  |  |  |
| --- | --- | --- |
| 6 | Exemple d’un programme MapReduce (WordCount)……………….. | **Figure 1.1** |
| 9 | Présentation du namenode, secodary namenode et data node……….. | **Figure 1.2** |
| 10 | Architecture de Hadoop……………………………………………… | **Figure 1.3** |
| 12 | Modèle de données clé-valeur……………………………………….. | **Figure 1.4** |
| 12 | Modèle de données orienté documents………………………………. | **Figure 1.5** |
| 13 | Modèle de données orienté colonnes………………………………… | **Figure 1.6** |
| 13 | Modèle de données en graphes………………………………………. | **Figure** **1.7** |

**Chapitre 2. Entrepôt de données**

|  |  |  |
| --- | --- | --- |
| **Figure 2.1** | Positionnement d’un système décisionnel dans l’organisation……… | 18 |
| **Figure 2.2:** | le système d’aide à la décision………………………………………. | 19 |
| **Figure 2.3** | Cycle de vie des entrepôts de données………………………………. | 21 |
| **Figure 2.4** | Exemple de modélisation en étoile………………………………….. | 22 |
| **Figure 2.5** | Exemple de modélisation en flocon…………………………………. | 23 |
| **Figure 2.6** | Exemple de modélisation en constellation…………………………... | 24 |
| **Figure 2.7** | Exemple d’un cube représentant les ventes de matériels informatiques………………………………………………………… | 26 |

**Chapitre 3. Processus ETL**

|  |  |  |
| --- | --- | --- |
| **Figure 3.1:** | processus d’entreposage des données…………………………. | 29 |
| **Figure 3.1** | Processus ETL Classique……………………………………… | 38 |
| **Figure 3.2** | Processus ETL basé sur le modèle MR……………………….. | 39 |

**Chapitre 4. Contribution Et Implémentation**

|  |  |  |
| --- | --- | --- |
| 45 | architecture proposé modélise le processus ***ETL*** basé sur MAPREDUCE ……………………………………………………….. | **Figure 4.1** |
| 48 | Création de la structure des données………………………………….. | **Figure 4.2** |
| 48 | Chargement des données dans hadoop………………………………... | **Figure 4.3** |
| 49 | Extraction des données depuis les les fichiers .CSV et .TXT…………. | **Figure 4.4** |
| 50 | Transformation des données ………………………………………….. | **Figure 4.5** |
| 51 | Construction des du base de donnée …………………………………. | **Figure 4.6** |
| 51 | Chargement des données……………………………………………… | **Figure 4.7** |
| 52 | Suivi d’exécution d’une requête………………………………………. | **Figure 4.8** |

**LISTE DES TABLEAUX**

|  |  |  |
| --- | --- | --- |
| **Tableau 3.1** | Formalisme de Vassiliadis pour la modélisation au niveau conceptuel …………………………………………………… | 30 |
| **Tableau 3.2** | Comparaison des travaux ……………………………………. | 37 |

|  |  |
| --- | --- |
| **HDFS** | Hadoop Distributed File System |
| **IBM** | International Business Machines |
| **NoSQL** | Not Only Sql |
| **SGBD** | Systèmes De Gestion De Base De Données |
| **ED** | Entrepôt De Données |
| **DW** | Data Warehouses |
| **OLAP** | On-Line Analytical Processing |
| **OLTP** | On-Line Transaction Processing |
| **ETL** | Extract – Transform – Load |
| **ROLAP** | Relational Online Analytical Processing |
| **ZPD** | Zone De Préparation Des Données |
| **MD** | Magasin De Données |

**LISTE DES ABREVIATION**

**INTRODUCTION GENERALE**

Les systèmes d’information deviennent de plus en plus complexes et diversifies en raison notamment de l’émergence de nouvelles technologies. L’accroissement continu de la volumétrie des données numériques ainsi que la multiplicité des sources de données de plus en plus hétérogènes, conjugues aux besoins pressants des entreprises a exploiter ces données dans un processus d’aide a la prise de décisions. [43]

Pour en concrétiser le potentiel, l’intégration de données, et plus précisément l’***ETL*** (Extract-Transform-Load), s’est imposé comme la solution. L’***ETL*** est le processus qui consiste à extraire les données brutes des systèmes sources, à les transformer dans un format exploitable, puis à les charger dans un système cible à des fins d’analyses avancées et de reporting. Pendant des années, les entreprises ont peiné à faire évoluer leurs architectures ***ETL*** traditionnelles pour tenir la cadence imposée par les trois « V » du phénomène ***Big Data*** : Volume, Vitesse et Variété des données.

Et pendant ce temps, les volumes de données, eux, continuent de gonfler. Sachant que la courbe n’est pas prête de s’infléchir, les entreprises ont commencé à se tourner vers ***Hadoop*** pour collecter, traiter et distribuer des données toujours plus volumineuses. En transférant les opérations ***ETL*** vers ***Hadoop***, elles épargnent à leur ***data warehouse*** des tâches de transformation lourdes. Ce faisant, elles parviennent à réduire considérablement leurs coûts et à libérer des capacités de leur base de données pour accélérer leurs analyses et raccourcir les temps de réponse aux requêtes utilisateurs.

Notre travail consiste a :

* réaliser un état de l'art. Cette étude s'intéresse aux systèmes décisionnels et précisément le principe de l'entreposage des données est de rassembler de multiples données sources qui souvent sont hétérogènes en les rendant homogènesafin de les analyser.
* la réalisation d’une nouvelle architecture d’***ETL*** basé sur le mécanisme ***Mapreduce*** dans les trois phases du processus et aussi élargir la contribution de MAHFOUD bala au niveau physique par une proposition d’une implémentation orienté ***MapReduce***.

|  |
| --- |
| CHAPITRE 1 |
| **BIG DATA** |

|  |
| --- |
| **PLAN DU CHAPITRE** |

**Introduction**

**1. Définition Du BigData**

**2. Caractéristique Du BigData**

2.1.Volume

2.2. Vélocité

2.3. Variété

2.4. Véracité

**3. Technologie BigData**

3.1. Mapreduce

3.1.1. Définition

3.2. Apache Hadoop

3.2.1 Définition

**3.3. Hadoop Distributed File System (Hdfs)**

3.3.1. Namenode

3.3.2. Secondary Namenode

3.3.3. Datanode

**3.4. Architecture**

3.4.1. Définition

3.4.2. Exécution D'une Tâche

**4. Les Bases De Données Nosql**

4.1. Base De Données Orientée Clé- Valeur

4.2. Base De Données Orientée Document

4.3. Base De Données Orientée Colonne

4.4. Base De Données Orientée Graphe

**Conclusion**

**INTRODUCTION**

L’évolution du ***SI*** amène les entreprises à traiter de plus en plus de données issues de sources toujours plus variées.

Selon ***IBM*** (Jacobs et Dinsmore (2013)) , chaque heure sont générés 2,5 trillions d'octets de données, soit 2,5 Po. Il est prévu également que pendant l'année 2020 seront générés 35 Zetta-octets, alors que seulement 1 Zetta-octets de données numériques ont été générés par l'humanité entre le début de l'informatique.

Un problème se pose alors quant au stockage et à l'analyse des données. La capacité de stockage des disques durs augmente mais le temps de lecture croît également. Les technologies associées, comme ***Hadoop***, ne constituent pas un monde à part dans les systèmes d’information. L’adoption de ces solutions encore jeunes représente un certain nombre d’enjeux, tant pour les responsables des entreprises que pour les utilisateurs métier.

Dans ce chapitre, nous allons essayer de survoler les concepts de base liées au phénomène du ***BIG DATA*** .Une présentation du technologie ***HADOOP*** et de son système de fichiers distribué ***HDFS*** va prendre sa part d'attention; etant donné que les informations massives deviennent difficiles à gérer avec les outils classiques de gestion de base de données, nous tenons à présenter une certaine catégorie de système de gestion des base de données volumineuse. Il s'agit du célèbre ***NOSQL***.

**1. DEFINITION DU BIG DATA**

Tout d’abord, il est important de spécifier que le ***Big Data*** n’a pas une définition arrêtée. De nombreuses variantes de descriptions sont retrouvées mais, bien sûr, elles sont similaires et renferment les mêmes concepts. Pour expliquer cette variation , nous pouvons dire que ce terme est relatif à l’entreprise concernée.[1]

Dans son article Qu’est-ce que le ***Big Data***, M. Lessard de ZeroSeconde le définit comme étant

**« 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 quantité de données ».** [2]

Tandis que, dans la définition ***big data*** , publiée sur le site de TechTarget, le terme est défini comme un terme général utilisé pour décrire les quantités volumineuses de données, structurées ou non, créées par une entreprise. [1]

### “would take too much time and cost too much money to load into a relational database for analysis. Although Big data doesn’t refer to any specific quantity, the term is often used when speaking about petabytes and exabytes of data”. [3]

En 2001, l’analyste du cabinet Meta Group (devenu Gartner) [Doug Laney](http://blogs.gartner.com/doug-laney/files/2012/01/ad949-3D-Data-Management-Controlling-Data-Volume-Velocity-and-Variety.pdf) décrivait les ***big data*** d'après le principe des « trois V » [4]

* le Volume de données de plus en massif ;
* la Variété de ces données qui peuvent être brutes, non structurées ou semi-structurées ;
* la Vélocité qui désigne le fait que ces données sont produites, récoltées et analysées en temps réel.

Certaines entreprises ajoutent un quatrième « V » à cette définition pour la Véracité qui évoque la nécessité de vérifier la crédibilité de la source et la qualité du contenu afin de pouvoir exploiter ces données.[4]

**2. CARACTERISTIQUE DU BIG DATA**

Les 4 principaux caractéristique du ***big data*** sont :

**2.1. VOLUME**

Le ***big data*** est associé à un volume de données vertigineux, se situant actuellement entre quelques dizaines de téra-octets et plusieurs péta-octets en un seul jeu de données. Les entreprises et tous les secteurs d’activités confondus, devront trouver des moyens pour gérer le volume de données en constante augmentation qui est créé quotidiennement. [5]

Voici quelques chiffres pour illustrer ce phénomène :

* + - * 90% des données actuelles ont été créées dans les deux dernières années seulement;
* Twitter comme exemple, génère 7 To de données chaque jour. [6]

**2.2. VELOCITE**

La vitesse décrit la fréquence à laquelle les données sont générées, capturées et partagées. Les entreprises doivent appréhender la vitesse non seulement en termes de création de données, mais aussi sur le plan de leur traitement, de leur analyse et de leur restitution à l'utilisateur en respectant les exigences des applications en temps réel. [7]

**2.3. VARIETE**

La croissance de la variété des données est la conséquence des nouvelles données multi structurelles et de l'expansion des types de données provenant de différentes sources hétérogènes. Aujourd’hui, on trouve des capteurs d'informations aussi bien dans les appareils électroménagers, les trains, les automobiles ou les avions, qui produisent des informations très variées. [8]

**2.4. VERACITE**

1 décideur sur 3 ne fait pas confiance aux données sur lesquelles il se base pour prendre ses décisions. Comment pouvez-vous vous appuyer sur l'information si vous n'avez pas confiance en elle ? Etablir la confiance dans les ***Big Data*** représente un défi d'autant plus important que la variété et le nombre de sources augmentent. [9]

**3. TECHNOLOGIE BIGDATA**

**3.1. MAPREDUCE**

**3.1.1. DEFINITION**

Le modèle ***MapReduce*** est un paradigme de programmation qui permet une scalabilité massive à travers une distribution de données et un traitement parallèle.[10]

Il cible la résolution des problèmes liés à la manipulation des données extrêmement volumineuses. Ce modèle a été introduit par ***Google*** en 2004 surtout pour construire et gérer son index WEB. Après avoir prouvé son efficacité, il est utilisé par d’autres sociétés telles que ***Yahoo*** et ***Facebook***. Son principe de base réside dans le fait de répartir les tâches de traitement (calcul) sur un nombre important de nœuds (machine) selon un modèle de programmation parallèle. Le nombre de nœuds peut atteindre facilement quelques milliers.[10]

Parmi les avantages notables de ce modèle :

* La scalabilité
* La tolérance aux panne
* La simplicité d’utilisation
* La réduction du coût. [10]

Un programme ***MapReduce*** peut se résumer à deux fonctions Map () et Reduce () .

**a. MAP**

Va transformer les données d'entrée en une série de couples clef /valeur. Elle va regrouper les données en les associant à des clefs, choisies de telle sorte que les couples clef/valeur aient un sens par rapport au problème à résoudre. Par ailleurs, cette opération doit être parallélisable: on doit pouvoir découper les données d'entrée en plusieurs fragments, et faire exécuter l'opération ***Map*** à chaque machine du cluster sur un fragment distinct. La fonction ***Map*** s'écrit de la manière suivante :

**Map (clé1, valeur1) → List (clé2, valeur2).** [11]

**b. REDUCE**

Va appliquer un traitement à toutes les valeurs de chacune des clefs distinctes produites par l'opération ***MAP***. Au terme de l'opération ***REDUCE***, on aura un résultat pour chacune des clefs distinctes. Ici, on attribuera à chacune des machines du cluster une des clefs uniques produites par **MAP**, en lui donnant la liste des valeurs associées à la clef. Chacune des machines effectuera alors l'opération ***REDUCE*** pour cette clef. La fonction ***Reduce*** s'écrit de la manière suivante :

**Reduce** **(clé2, List (valeur2)) → List (valeur2).** [11]

L’exemple classique est celui du **WordCount** qui permet de compter le nombre d’occurrences d’un mot dans un fichier. En entrée l'algorithme reçoit un fichier texte qui contient les mots suivants **voiture la le elle de elle la se la maison voiture**

Dans notre exemple, la clé d'entrée correspond au numéro de ligne dans le fichier et tous les mots sont comptabilisés à l'exception du mot « se ».

Le résultat de la fonction ***Map*** est donné ci-dessous.

**(voiture, 1) /(la,1) /(le,1) / (elle,1) / (de,1) / (elle,1) / (la,1) / (la,1) / (maison,1) /(voiture,1)**

Avant de présenter la fonction **Reduce**, deux opérations intermédiaires doivent être exécutées pour préparer la valeur de son paramètre d'entrée. La première opération appelée **shuffle** permet de grouper les valeurs dont la clé est commune. La seconde opération appelée ***sort*** permet de trier par clé. A la différence des fonctions ***Map*** et ***Reduce***, ***shuffle*** et ***sort*** sont des fonctions fournies par le Framework ***Hadoop***, donc, il n’a pas à les implémenter.[12]

Ainsi, après l'exécution des fonctions ***shuffle*** et sort le résultat de l'exemple est le suivant :

**(de, [1]) / (elle, [1,1]) / (la, [1, 1,1]) / (le, [1]) / (maison, [1]) / (voiture, [1,1]**

Suite à l'appel de la fonction ***Reduce***, le résultat de l'exemple est le suivant :

**(de, 1) / (elle, 2) / (la, 3) / (le, 1) / (maison, 1) / (voiture, 2)**

**voiture la le elle de elle la se la maison voiture**

**voiture la le elle de elle la se la maison voiture**

**voiture la le elle de elle la se la maison voiture**

**voiture la le elle de elle la se la maison voiture**

**voiture 1**

**la 1**

**le 1**

**elle 1**

**…**

**de [1]**

**elle [1 ,1]**

**la [1 ,1 ,1]**

**le [1]**

**…**

**de 1**

**elle 2**

**la 3**

**le 1**

…

**Mapper**

**Reduce**

**Shuffle & Sort**

**Figure 1.1:** Exemple d’un programme ***MapReduce*** (WordCount) **[12]**

**3.2. APACHE HADOOP**

Un problème se pose alors quant au stockage et à l'analyse des données massive. La capacité de stockage des disques durs augmente mais le temps de lecture croît également. Il devient alors nécessaire de paralléliser les traitements en stockant sur plusieurs unités de disques durs. Toutefois, cela soulève forcément le problème de fiabilité des disques durs qui engendre la panne matérielle. La solution envisagée est la duplication des données comme le ferait un système RAID[[2]](#footnote-2). [12]

**3.2.1 DEFINITION**

Il s'agit d'un framework **Open Source** conçu pour réaliser des traitements sur des volumes de données massifs, de l'ordre de plusieurs peta-octets (soit plusieurs milliers de To). Il s'inscrit donc typiquement sur le terrain du ***Big Data***. Géré sous l'égide de la fondation ***Apache***, ***Hadoop*** est écrit en Java. ***Hadoop*** a été conçu par Doug Cutting en 2004 . Doug Cutting cherchait une solution pour accroître la taille de l'index de son moteur. Il eut l'idée de créer un framework de gestion de fichiers distribués. ***Hadoop*** était né.[13].

D'une part, il propose un système de stockage distribué via son système de fichier ***HDFS*** (Hadoop Distributed File System) et ce dernier offre la possibilité de stocker la donnée en la dupliquant, un cluster ***Hadoop*** n'a donc pas besoin d'être configuré avec un système ***RAID*** qui devient inutile. D'autre part, ***Hadoop*** fournit un système d'analyse des données appelé ***MapReduce***. Ce dernier officie sur le système de fichiers ***HDFS*** pour réaliser des traitements sur des gros volumes de données.

Trois principales distributions ***Hadoop*** sont aujourd’hui disponibles : ***Cloudera,***  ***Hortonworks*** , ***MapR***. [14]

**3.3. HADOOP DISTRIBUTED FILE SYSTEM (HDFS)**

***HDFS*** est un système de fichiers distribué qui donne un accès haute-performance aux données réparties dans des clusters ***Hadoop***. Comme d’autres technologies liées à ***Hadoop***, ***HDFS*** est devenu un outil clé pour gérer des pools de ***Big Data*** [15]

Toutefois, ***HDFS*** se démarque d'un système de fichiers classique pour les principales raisons suivantes.

* ***HDFS*** n'est pas solidaire du noyau du système d'exploitation. Il assure une portabilité et peut être déployé sur différents systèmes d'exploitation. Un des inconvénients est de devoir solliciter une application externe pour monter une unité de disque ***HDFS***. [16]
* ***HDFS*** est un système distribué. Sur un système classique, la taille du disque est généralement considérée comme la limite globale d'utilisation. Dans un système distribué comme ***HDFS***, chaque nœud d'un cluster correspond à un sous-ensemble du volume global de données du cluster. Pour augmenter ce volume global, il suffira d'ajouter de nouveaux nœuds. On retrouvera également dans ***HDFS***, un service central appelé ***Namenode*** qui aura la tâche de gérer les métadonnées. [16]
* ***HDFS*** utilise des tailles de blocs largement supérieures à ceux des systèmes classiques. Par défaut, la taille est fixée à 64 Mo. Il est toutefois possible de monter à 128 Mo, 256 Mo, 512 Mo voire 1 Go. Alors que sur des systèmes classiques, la taille est généralement de 4 Ko, l'intérêt de fournir des tailles plus grandes permet de réduire le temps d'accès à un bloc. Notez que si la taille du fichier est inférieure à la taille d'un bloc, le fichier n'occupera pas la taille totale de ce bloc. [16]
* ***HDFS*** fournit un système de réplication des blocs dont le nombre de réplications est configurable. Pendant la phase d'écriture, chaque bloc correspondant au fichier est répliqué sur plusieurs nœuds. Pour la phase de lecture, si un bloc est indisponible sur un nœud, des copies de ce bloc seront disponibles sur d'autres nœuds. [16]

**3.3.1. NAMENODE**

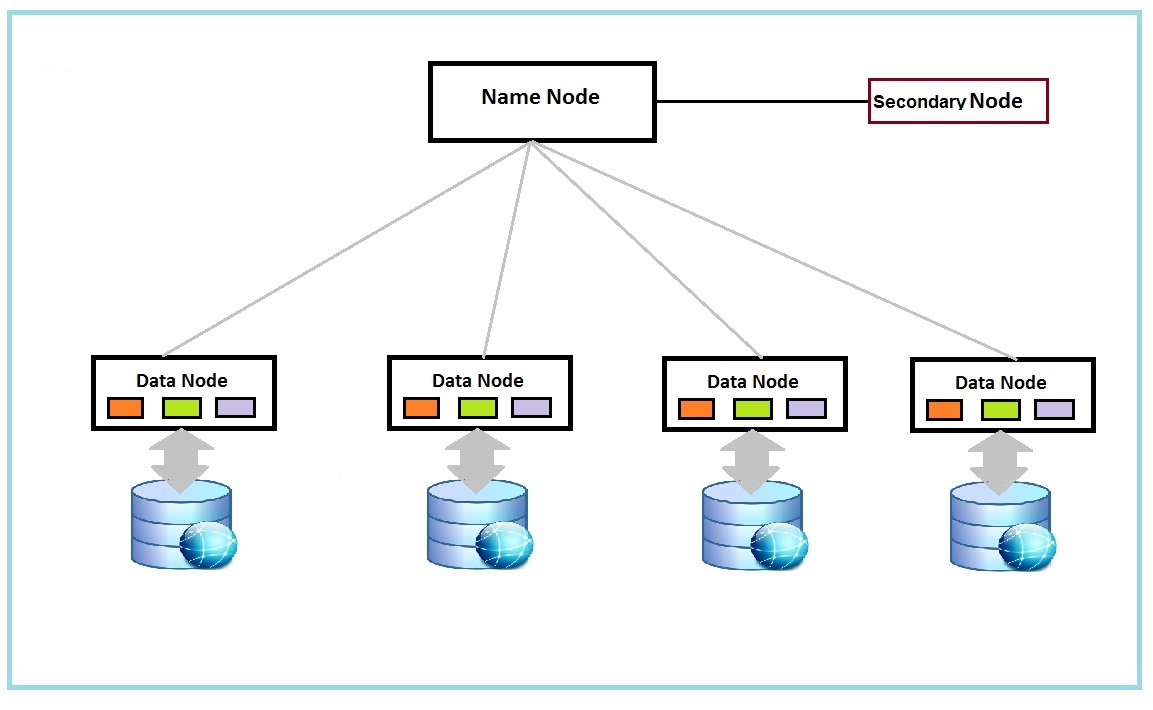
Un ***Namenode*** est un service central (généralement appelé aussi maître) qui s'occupe de gérer l'état du système de fichiers. Il maintient l'arborescence du système de fichiers et les métadonnées de l'ensemble des fichiers et répertoires d'un système ***Hadoop***. Le ***Namenode*** a une connaissance des ***Datanodes*** (étudiés juste après) dans lesquels les blocs sont stockés. Ainsi, quand un client sollicite ***Hadoop*** pour récupérer un fichier, c'est via le ***Namenode*** que l'information est extraite. Ce ***Namenode*** va indiquer au client quels sont les ***Datanodes*** qui contiennent les blocs. Il ne reste plus au client qu'à récupérer les blocs souhaités. [17]

**3.3.2. SECONDARY NAMENODE**

Le ***Namenode*** dans l'architecture ***Hadoop*** est un point unique de défaillance (Single Point of Failure en anglais). Si ce service est arrêté, il n'y a pas moyen de pouvoir extraire les blocs d'un fichier donné. Pour répondre à cette problématique, un ***Namenode*** secondaire appelé ***Secondary Namenode*** a été mis en place dans l'architecture ***Hadoop***. Son fonctionnement est relativement simple puisque le ***Namenode*** secondaire vérifie périodiquement l'état du ***Namenode***. S'il est indisponible, le ***secondary Node*** prend sa place . [17]

**3.3.3. DATANODE**

Précédemment, nous avons vu qu'un ***Datanode*** contient les blocs de données. Les ***Datanodes*** sont sous les ordres du ***Namenode*** et sont surnommés les ***Workers***. Ils sont donc sollicités par les ***Namenodes*** lors des opérations de lecture et d'écriture. En lecture, les ***Datanodes*** vont transmettre au client les blocs correspondant au fichier à transmettre. En écriture, les ***Datanodes*** vont retourner l'emplacement des blocs fraîchement créés. Les ***Datanodes*** sont également sollicités lors de l'initialisation du ***Namenode*** et aussi de manière périodique, afin de retourner la liste des blocs stockés . [18]



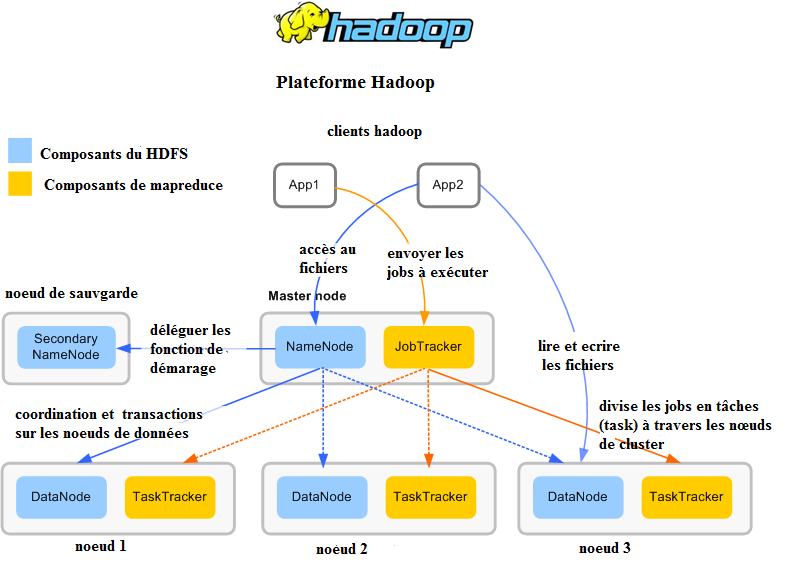
**Figure 1.2** : Présentation du namenode, secodary namenode et data node [19]

**3.4. ARCHITECTURE**

**3.4.1. DEFINITION**

Repose sur deux serveurs:

* Le ***JobTracker***, unique sur le *cluster*. Reçoit les tâches map/reduce à exécuter (sous la forme d'une archive Java.jar), organise leur exécution sur le *cluster*. [11]
* Le ***TaskTracker***, plusieurs par *cluster*. Exécute le travail map/reduce lui-même (sous la forme de tâches map et reduce ponctuelles avec les données d'entrée associées). [11]
* Chacun des ***TaskTrackers*** constitue une unité de calcul du *Cluster.* [11]
* 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. [11]
* 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***.[11]



**Figure 1.3** : architecture de ***Hadoop*** [20]

**3.4.2. Exécution d'une tâche**

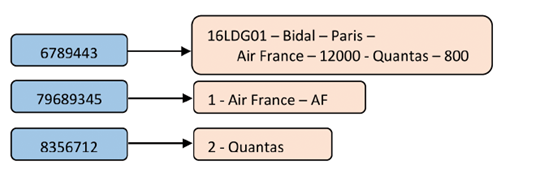
* Tous les ***TaskTrackers*** signalent leur statut continuellement par le biais de paquets *heartbeat*.
* En cas de défaillance d'un ***TaskTracker*** (heartbeat manquant ou tâche échouée), le ***JobTracker*** avise en conséquence: redistribution de la tâche à un autre noeud, etc.
* Au cours de l'exécution d'une tâche, on peut obtenir des statistiques détaillées sur son évolution (étape actuelle, avancement, temps estimé avant completion, etc.), toujours par le biais du client console ***hadoop***.[19]

**4. LES BASES DE DONNEES NOSQL**

Le terme ***NoSQL*** est pour la première fois sorti de la bouche de *Carlo Strozzi* en 1998 [21], il désigne une catégorie de systèmes de gestion de base de données destinés à manipuler des bases de données volumineuses pour des sites de grande audience. ***NoSQL*** (***Not Only SQL***) est un terme générique pour désigner les bases de données ne suivant pas les principes classiques des ***SGBD*** relationnels. L’idée sous-jacente du mouvement ***NoSQL*** est d’avoir des bases plus adaptées au besoin réel de l’application plutôt que d’utiliser systématiquement une base relationnelle. [5] La famille des ***SGBD*** ***NoSQL*** se compose de plusieurs catégories, les ***SGBD*** orienté colonne, orienté document, orienté graphe et clé/valeur.[22]

**4.1. BASE DE DONNEES ORIENTEE CLE- VALEUR**

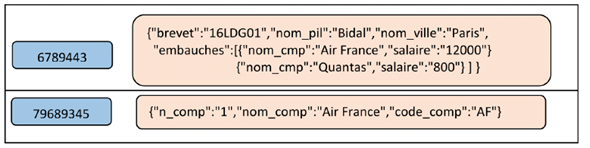
Ce modèle peut être assimilé à une hashmap[[3]](#footnote-3) distribuée. Les données sont, donc, simplement représentées par un couple clé/valeur. La valeur peut être une simple chaîne de caractères, un objet sérialisé… Cette absence de structure ou de typage ont un impact important sur le requêtage. En effet, toute l’intelligence portée auparavant par les requêtes ***SQL*** devra être portée par l’applicatif qui interroge la BD. Néanmoins, la communication avec la BD se résumera aux opérations PUT, GET et DELETE. Les solutions les plus connues sont Redis, Riak et Voldemort créé par LinkedIn. [23]



**Figure 1.4** : modèle de données clé-valeur [24]

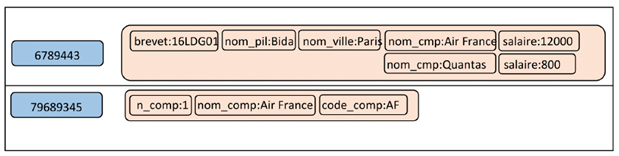
**4.2. BASE DE DONNEES ORIENTEE DOCUMENT**

Ce modèle se base sur le paradigme clé valeur. La valeur, dans ce cas, est un document de type ***JSON*** ou ***XML***. L’avantage est de pouvoir récupérer, via une seule clé, un ensemble d’informations structurées de manière hiérarchique. La même opération dans le monde relationnel impliquerait plusieurs jointures. Pour ce modèle, les implémentations les plus populaires sont ***CouchDB*** d’Apache, ***RavenDB*** (destiné aux plateformes .NET/Windows avec la possibilité d’interrogation via LINQ) et ***MongoDB***.[23]



**Figure 1.5:** modèle de données orienté documents [24]

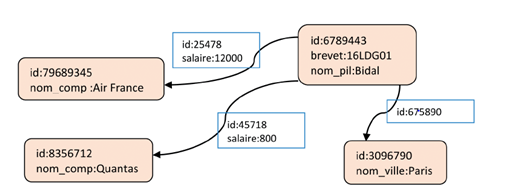
**4.3. BASE DE DONNEES ORIENTEE COLONNE**

Ce modèle ressemble à première vue à une table dans un ***SGBDR*** à la différence qu’avec une BD ***NoSQL*** orientée colonne, le nombre de colonnes est dynamique. En effet, dans une table relationnelle, le nombre de colonnes est fixé dés la création du schéma de la table et ce nombre reste le même pour tous les enregistrements dans cette table. Par contre, avec ce modèle, le nombre de colonnes peut varier d’un enregistrement à un autre ce qui évite de retrouver des colonnes ayant des valeurs NULL. Comme solutions, on retrouve principalement ***HBase*** (implémentation Open Source du modèle ***BigTable*** publié par Google) ainsi que Cassandra (projet Apache qui respecte l’architecture distribuée de Dynamo d’Amazon et le modèle ***BigTable*** de Google).[23]

**Figure 1.6:** modèle de données orienté colonnes [24]

**4.4. BASE DE DONNEES ORIENTEE GRAPHE**

Ce modèle de représentation des données se base sur la théorie des graphes. Il s’appuie sur la notion de nœuds, de relations et de propriétés qui leur sont rattachées. Ce modèle facilite la représentation du monde réel, ce qui le rend adapté au traitement des données des réseaux sociaux. La principale solution est Neo4J. [23]



**Figure** **1.7** **:** modèle de données en graphes[24]

**CONCLUSION**

Ce 1er chapitre a été consacré essentiellement à la présentation de l’environnement dans lequel mon projet de fin d’études a été effectué.

D’abord nous avons défini les principaux concepts liées au terme ***BIG DATA*** , ensuite nous avons expliqué la technologie ***HADOOP*** et ses différents constituants, ainsi nous décrivons en détail le fonctionnement du mécanisme ***Mapreduce*** , Nous terminerons ce chapitre par une présentation du différents types des bases des données ***NOSQL*** .

Dans le 2eme chapitre nous continuerons a présenter l’état de l’art  , nous décrivons le système décisionnel ainsi qu’une présentation globale sur l***’entrepôt de données*** .

|  |
| --- |
| CHAPITRE 2 |
| **ENTREPOT DE DONNEES** |

|  |
| --- |
| **PLAN DU CHAPITRE** |

**Introduction**

**1. Différence Entre L'informatique De Production Et L'informatique De Décision**

1.1. L'informatique De Production

1.2. L'informatique Décisionnelle

1.2.1. Le Système Décisionnel Dans L’organisation

1.2.2. Traitement Décisionnelle

1.2.2. Base De Données Décisionnelle

**2. Entrepôt De Données**

2.1. Définition

**3. Conception D'un Entrepôt De Données**

**4. Modélisation D'entrepôts De Données**

4.1. Niveau Conceptuel

4.1.1. Le Schéma En Flocon

4.1.2. Le Schéma En Constellation

4.2. Niveau Logique

4.3. Niveau Physique

**5. Restitution Et Analyse OLAP**

**Conclusion**

**INTRODUCTION**

Les systèmes décisionnels sont mis à disposition de l'entreprise pour permettre d'exploiter les données utiles pour la prise de décision. Ces systèmes doivent fournir un espace de stockage permettant de conserver de manière permanente un volume important de données issues des sources de production. [34]

Lors des premiers projets décisionnels, la phase de la collecte des données et de leur transformation était souvent sous-estimée. C'est peut-être là une des principales explications Des échecs de réalisations et des très nombreux dépassements de budget. [27]

En intégrant la technologie des entrepôts de données (***data warehouses***), le processus décisionnel apporte une réponse au problème de la croissance continuelle des données pouvant être de formats différents. Donc, les entrepôts de données sont considérés comme un moyen essentiel pour garantir une meilleure prise de décision. [61]

La suite de ce chapitre est organisée de la façon suivante : Nous introduisons dans une première section la différence entre l'informatique de production et l'informatique de décision, par la suite nous présentons dans la section 2 le contexte de notre chapitre en définissant ce qu'est un ***entrepôt de données*** ainsi que les processus permettant d'alimenter l'entrepôt à partir des sources. Dans la section 3 nous citons les différentes phases du cycle de vie de conception d’un entrepôt de données, suivi par les niveaux de conception d’un entrepôt de données (niveau conceptuel, logique et physique dans la section 4. La section 5 sera consacrée aux définitions du processus d’analyse en ligne ***OLAP*** (On-Line Analytical Processing - OLAP)

**1. DIFFERENCE ENTRE L'INFORMATIQUE DE PRODUCTION ET L'INFORMATIQUE DE DECISION**

Les systèmes de production sont caractérisés par une activité constante composée de modifications et d'interrogations fréquentes des bases de données par de nombreux utilisateurs : ajouter une commande, modifier une adresse de livraison, rechercher les coordonnées d'un client, etc. l'intégrité des données est nécessaire pour ce genre d'applications (il faut par exemple, interdire la modification simultanée d'une même donnée par deux utilisateurs différents). La priorité est donnée en premier lieu à l'enregistrement rapide, sûr et efficace des données.[25]

À l'inverse, nul besoin de modification ou d'enregistrement de nouvelles données dans les systèmes d'information de décision. En effet, le but principal consiste en l'interrogation du système d'information afin de permettre l'analyse des indicateurs pertinents pour faciliter les prises de décisions. Les questions posées seront de la forme : « Quelles sont les ventes du produit X pendant le trimestre A de l'année B dans la région C ». Une telle interrogation peut nécessiter des temps de calcul importants. Or, l'activité d'un serveur transactionnel ne peut être interrompue. Il faut donc prévoir une nouvelle organisation qui permette de mémoriser un grand volume de jeux de données et qui soit structurée de façon à avoir un moteur de recherche de connaissances efficace et approprié. [26]

**1.1. L'INFORMATIQUE DE PRODUCTION**

L'un des formalismes les plus utilisés pour la représentation conceptuelle des systèmes d'information est le modèle Entité Association. Dans ce genre de modèle, toute redondance est considérée comme source d'erreurs, d'incohérence, et va pénaliser les temps d'exécution et complexifier les procédures d'ajout, de suppression ou de modification.

Les systèmes transactionnels temps réel, ***OLTP[[4]](#footnote-4)*** (***On-line Transaction Processing*** ), garantissent l'intégrité des données. Toutefois, le modèle Entité Association et sa réalisation dans un schéma relationnel sont pourtant des obstacles importants pour l'accès de l'utilisateur final aux données. Généralement, ces modèles contiennent plusieurs dizaines d'entités. Les bases sont alors constituées de nombreuses tables, reliées entre elles par divers liens dont le sens n'est pas toujours explicite. Il devient clair que le but était de faciliter le travail des développeurs et d'améliorer des performances de tâches répétitives, prévues et planifiées tournées vers la production de documents standards (factures, commandes...), au détriment des besoins réels de l'utilisateur final.[27]

Enfin, La dernière caractéristique de ces bases de données est qu'elles conservent l'état instantané du système. Dans la plupart des cas, l'évolution n'est pas sauvegardée. On garde simplement une trace des versions instantanées pour la reprise en cas de panne et pour des raisons légales.[27]

**1.2. L'INFORMATIQUE DECISIONNELLE**

L'informatique décisionnelle est l'ensemble des méthodes, moyens et outils informatiques utilisés pour piloter une entreprise et aider à la prise de décision.

Les environnements d'aide à la décision s'intéressent aux tendances, aux moyennes des principaux indicateurs de bonne santé de l'entreprise, et ce, à travers les mois ou les années.

Le monde décisionnel analyse des données agrégées pour mieux apprécier l'ensemble de l'activité de l'entreprise. L'aide à la prise de décision est la responsabilité de quelques personnes dans l'entreprise (décideurs). En effet, les décideurs s'intéressent à l'ensemble de l'activité et souhaitent avoir une vue globale de la société.

Dans le monde décisionnel, on ne supprime jamais des données, on archive. Les outils de BI permettent une exploitation rationnelle et coopérative des données (contrairement à l'informatique de gestion ou l'on produit de l'information grâce aux bases de données relationnelles).[28]

* + 1. **LE SYSTEME DECISIONNEL DANS L’ORGANISATION**

Toute organisation peut être décomposée en trois systèmes :

* **Le système opérant** qui correspond à l’activité de production de l’organisation en transformant les flux primaires pour répondre aux besoins des clients. [29]
* **Le système de décision** correspondant à l’ensemble des traitements et du personnel dirigeant qui contrôle, régule, pilote et adapte l’organisation par leurs décisions.[29]
* **Le système d’information** permettant de collecter, conserver, traiter et restituer les données produites dans l’organisation ; il joue le rôle d’interfaces entre les deux systèmes précédents.[29]

La Figure 2.1 présente les trois systèmes et leurs interactions.

**Système de décision**

**Système opérant**

**Système d’information**

Flux décision

Flux d’information

Flux physique

Flux externe

Flux interne

Information de décision

Information réelles

Information réelles

Information de décision

**Figure 2.1:** Positionnement d’un système décisionnel dans l’organisation [29]

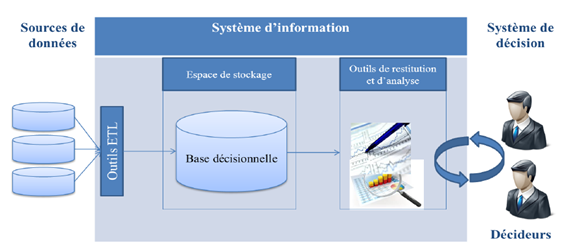
Le système d’information comporte deux types de traitements : les traitements transactionnels destinés au système opérant et les traitements décisionnels qui mettent les informations prétraitées à la disposition du système de décision pour assurer un pilotage optimal. **[29]**

**1.2.2. LES TRAITEMENTS DECISIONNELS**

Ces traitements consistent à extraire, prétraiter et stocker des données utiles à la prise de décision. Les données peuvent provenir des sources internes à l’entreprise (base de données de production dédiés aux traitements transactionnels) ou de sources de données externes (base de données de partenaires, sites web). Une base décisionnelle contient un ensemble de données destiné au système de décision et organisé selon un modèle adapté à l’usage qui en est fait. Les outils ***ETL*** (**pour Extract – Transform - Load**) permettent d’alimenter régulièrement une base décisionnelle à partir des sources. **[29]**

**1.2.3. BASE DE DONNEES DECISIONNELLE**

Une base décisionnelle est l’ensemble des outils informatiques (matériels et logiciels) permettant l’analyse des données issues du système d’information des entreprises. Ces données sont représentées dans la base décisionnelle en une vision orientée décideur puis analysées au moyen des outils de restitution et d’analyse , (Olap , Datamaining , Datamarts , ect ).[30]

**Figure 2.2:** le système d’aide à la décision [29]

**2. ENTREPOT DE DONNEE**

**2.1. DEFINITION**

Le concept ***d'entrepôt de donnée*** a pris forme au début des années 90, il est devenu depuis la clé de voute de ce que l’on appelle l’informatique décisionnelle. La définition la plus appropriée de l’***ED*** a été formalisé pour la première fois en 1990 par **Bill Inmon** : l’***ED*** est une collection de données intégrées, orientées sujet, non volatiles et historiées, organisées pour le support d’un processus d’aide à la décision. C’est une base de données d’aide a la décision qui est entretenue de manière séparée des sources a partir des quelles est formée. [31]

Les données d'un ***ED*** possèdent les caractéristiques suivantes.

* **Intégrées**: Les données proviennent de différentes sources souvent structurées et codés de façon différente. L’intégration assure une représentation uniforme, cohérente et transparente. Cela résout les problèmes d’hétérogénéité des systèmes de stockage, des modèles de données et de sémantique de données.[32]
* **Orientées sujets**: Les données s’organisent par sujet ou thème (clients, vendeurs, production, etc.), ce qui permet de rassembler toutes les informations utiles à la prise de décision.[32]
* **Non volatiles**: Les données chargées sont utilisées en mode de consultation. Elles ne peuvent pas être modifiées par l’utilisateur.[32]
* **Historisées**: L’entrepôt de données contient des données archivées afin de les utiliser pour les comparaisons, la prévision, etc.[32]
* **Organisées** : Les informations issues des sources de données doivent être agrégées et réorganisées afin de faciliter le processus de prise de décision.[32]

**3. CONCEPTION D’UN ENTREPOT DE DONNEES**

Il regroupe les phases suivantes: la planification, la conception et l’implémentation, la maintenance et la gestion de l’évolution et le test.

* **La planification**. Cette phase vise à préparer le terrain pour le développement de l’entrepôt de données. Elle consiste à:
* déterminer l’étendue du projet ainsi que les buts et objectifs de l’entrepôt à développer,
* évaluer la faisabilité technique et économique de l’entrepôt,
* identifier les futurs utilisateurs de l’entrepôt.[33]
* **La conception et l’implémentation**. Cette phase consiste à développer le schéma de l’entrepôt et à mettre en place toutes les ressources nécessaires à son implémentation et à son déploiement. [33]
* **La maintenance et la gestion de l’évolution**. Cette phase implique l’optimisation de ses performances périodiquement. L’évolution de ***l’entrepôt de données*** concerne la mise à jour de son schéma en fonction des différents changements survenant au niveau des sources ou des besoins des utilisateurs. Le résultat est une certaine forme de rétroaction, ce qui peut entraîner le retour à l’une des étapes précédentes dans la conception. [33]
* **Les tests**. Tester un ***entrepôt de données*** est une tâche cruciale, il permet de tester l’ensemble des phases de conception de l’entrepôt de données en termes de qualité, performance, sécurité, etc. [33]

Le schéma illustré par la figure 1, représente la succession des tâches nécessaires à la mise en place des entrepôts de données efficaces. Chaque rectangle indique une phase

**Planification**

**Maintenance et évaluation**

Analyse des besoins

Conception logique

ETL

**Déploiement**

Conception physique

**Conception et implémentation**

**Figure 2.3 :** Cycle de vie des entrepôts de données [33]

**4. MODÉLISATION D'ENTREPÔTS DE DONNÉES**

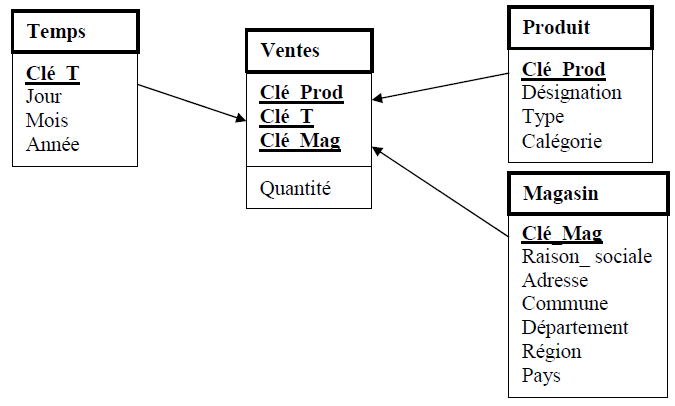
Les modèles de données d'un ***ED*** ont été conçus pour faciliter les analyses décisionnelles **[Kimball, 1996]**. Typiquement, la modélisation d'un ***ED*** repose sur trois niveaux d'abstraction : (1) le niveau conceptuel qui prend en compte les besoins des décideurs en faisant abstraction des aspects techniques, (2) le niveau logique ou l'on fait le choix d'un modèle de stockage et (3) le niveau physique ou l'***ED*** est implanté sur un Système de Gestion de Base de Données (***SGBD***) particulier. Dans ce qui suit, nous présentons les pratiques les plus courantes lors de la modélisation d'un ***ED***.[34]

**4.1. Niveau conceptuel**

Au niveau ***conceptuel***, les ***données de l'entrepôt*** sont représentées par des schémas en étoile, en constellation ou en flocon. Ces schémas organisent les données en termes de sujets d'analyse et d'axes d'analyses. Un sujet d'analyse appelée également Fait est composé par un ou plusieurs indicateurs d'analyse , ce sont les Mesures. Les axes d'analyse appelés Dimensions sont composés de Paramètres en fonction desquels les mesures sont étudiées. Les paramètres sont organisés selon des hiérarchies, de la granularité la plus faible (paramètre racine) à la granularité la plus haute. [34]

**4.1.1. Le schéma en étoile**

Dans ce schéma, il existe une relation pour les faits[[5]](#footnote-5) et plusieurs pour les différentes dimensions[[6]](#footnote-6) autour de la relation centrale. La relation de faits contient les différentes mesures et une clé étrangère pour faire référence à chacune de leurs dimensions. La figure 1.1 montre le schéma en étoile en décrivant les ventes réalisées dans les différents magasins de l’entreprise au cours d’un jour. Dans ce cas, nous avons une étoile centrale avec une table de faits appelée Ventes et autour leurs diverses dimensions : Temps, Produit et Magasin**.** [35]



**Figure 2.4 :** Exemple de modélisation en étoile[36]

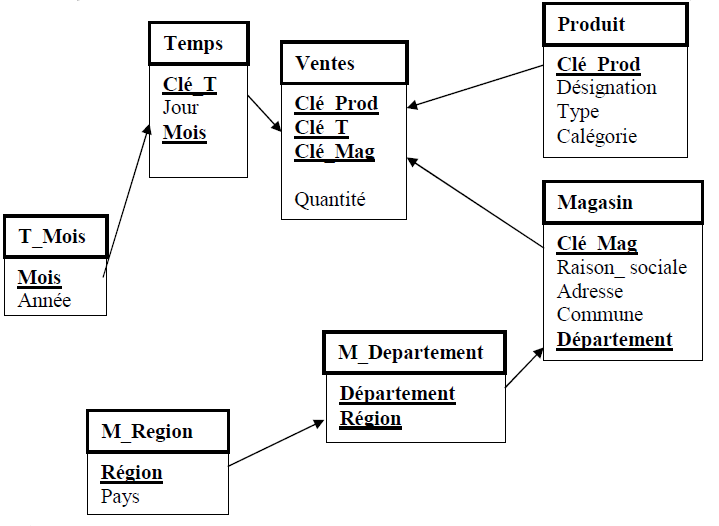
**4.1.2. Le schéma en flocon**

Il dérive du schéma précédent avec une relation centrale et autour d’elle les différentes dimensions, qui sont éclatées ou décomposées en sous hiérarchies. L’avantage du schéma en flocon de neige est de formaliser une hiérarchie au sein d’une dimension, ce qui peut faciliter l’analyse. Un autre avantage est représenté par la normalisation des dimensions, car nous réduisons leur taille.[37]

Les hiérarchies pour le schéma en flocon de neige de l’exemple de la figure 1.1 sont :

Dimension Temps = Jour Mois Année

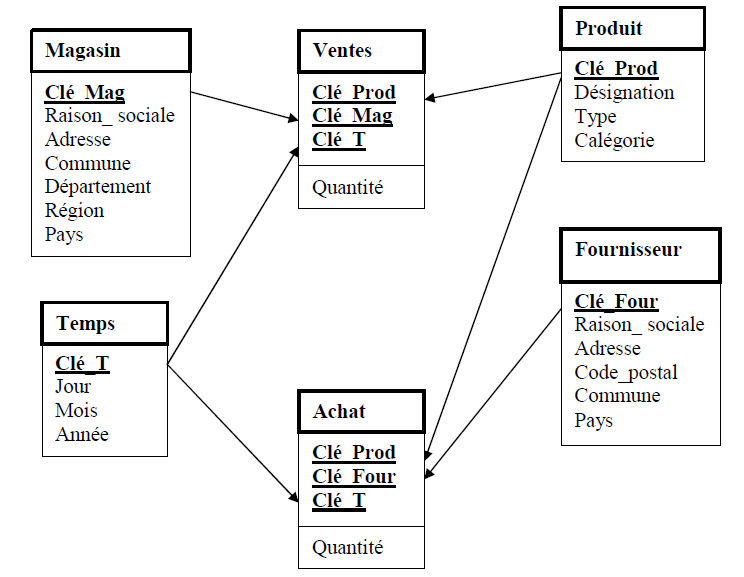
Dimension Magasin = Commune Département Région Pays [32]



**Figure 2.5 :** Exemple de modélisation en flocon[36]

**4.1.3. Le schéma en constellation**

Le schéma en constellation représente plusieurs relations de faits qui partagent des dimensions communes. Ces différentes relations de faits composent une famille qui partage les dimensions mais où chaque relation de faits a ses propres dimensions. La figure 1.3 montre le schéma en constellation qui est composé de deux relations de faits. La première s’appelle Ventes et enregistre les quantités de produits qui ont été vendus dans les différents magasins pendant un certain jour. La deuxième relation gère les différents produits achetés aux fournisseurs pendant un certain temps.[35]



**Figure 2.6 :** Exemple de modélisation en constellation[36]

La relation de faits Ventes partage leurs dimensions Temps et Produits avec la table Achats. Néanmoins, la dimension Magasin appartient seulement à Ventes. Egalement, la dimension Fournisseur est liée seulement à la relation Achats.[35]

**4.2. Niveau logique**

A ce niveau, plusieurs possibilités sont envisageables pour modéliser la structure d'un entrepôt. Le plus courant est d'utiliser un modèle relationnel dit ***ROLAP*** (***Relational*** ***OnLine*** ***Analytical*** ***Processing***). Dans les modèles ***ROLAP***, les faits et les dimensions du niveau conceptuel sont traduits en tables **[Kimball,1996]**. Il existe deux variantes principales des modèles ***ROLAP***; ce sont les modèles ***ROLAP*** dénormalisés et ***ROLAP*** normalisés. Dans un schéma ***ROLAP*** dénormalisé, chaque fait et chaque ***dimension*** du schéma conceptuel sont traduits en tables. Dans la modélisation ***ROLAP*** normalisé, les tables de dimensions sont normalisés. Conformément aux hiérarchies associées aux dimensions, chaque niveau d'agrégation est transformé en une table. L'intérêt des modèles ***ROLAP*** est la possibilité de réutiliser les principes de manipulation des bases de données relationnelles. [34]

**4.3. Niveau physique**

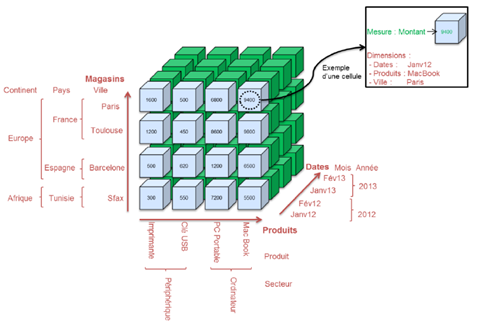
Suite à une modélisation ***ROLAP***, la modélisation physique consiste à choisir un mode d'implantation particulier (notamment le ***SGBD***) et à définir des scripts ***SQL*** pour la création et l'alimentation de l'entrepôt. **[34]**

**5. RESTITUTION ET ANALYSE OLAP**

Un système ***OLAP*** est un système d’information décisionnel qui organise les données dans un espace dimensionnel. Il regroupe un ensemble d’outils en interaction qui réalisent la synthèse dynamique, l’analyse interactive et l’agrégation d’un grand volume de données afin d’améliorer le processus de prise de décision. [38]

Selon l’architecture à deux niveaux, un ***entrepôt de données*** est structuré suivant une modélisation multidimensionnelle. Ceci permet de représenter l’extension d’un entrepôt sous la forme de points dans un espace à plusieurs dimensions avec la métaphore du ***cube[[7]](#footnote-7)*** ou de ***l’hyper-cube*** de données.

La Figure 2.7 présente un exemple de cube qui permet l’analyse des ventes de matériels informatiques. L’analyse des montants de ventes s’effectue en fonction de trois dimensions : les magasins où ont été effectuées les ventes, les dates de ventes et les produits vendus. Chacune de ces dimensions est associée à des paramètres de granularité différente (pour la dimension Magasin: ville, pays et continent). Ces niveaux hiérarchiques permettent d’obtenir des visions plus ou moins synthétiques lors des analyses ***OLAP***. **[29]**



**Figure 2.7 :** Exemple d’un cube représentant les ventes de matériels informatiques [29]

**CONCLUSION**

Le but principal de notre travail est l’alimentation d’une base de donnée spéciale appelé ***DW*** pour la prise de décision.

Dans ce chapitre nous avons traité le sujet des ***entrepôts de données*** qui étendent les concepts et la technologie traditionnelle des bases de données pour créer des systèmes d'aide à la décision . Nous avons montré la différence entre les deux types d’informatique , après expliqué le terme ***DW***. ainsi décrit les différents modèles multidimensionnels pour la construction d'un entrepôt de données . Nous avons étudié la modélisation ***DW*** au niveau conceptuelle , logique, et physique , finalement nous avons présenter la restitution et analyse ***OLAP*** .

Dans le chapitre qui suit , nous avons détaillons sur le processus ***ETL*** ainsi leur modélisation aux niveau conceptuel et logique .

|  |
| --- |
| CHAPITRE 3 |
| **PROCESSUS ETL** |

|  |
| --- |
| **PLAN DU CHAPITRE** |

**Introduction**

**1. Processus ETL**

1.1. La Phase D'extraction

1.2. La Phase De Transformation

1.3. La Phase De Chargement

**2. Modélisation Du Processus ETL**

2.1. Modélisation Conceptuelle

2.2. Modélisation Logique

2.3. Modélisation Physique

**3. Travaux Antérieurs**

**4. Etude Comparative**

4.1. Critère De Comparaison

**5. Classification Des Travaux Sur Le Processus ETL**

5.1. Processus ETL Classique

5.2. Processus ETL Base Sur Le Modèle Mapreduce

5.3. Partitionnement Des Données

**Conclusion**

**INTRODUCTION**

***ETL***, acronyme de *Extraction, Transformation, Loading*, est un système de chargement de données depuis les différentes sources d'information de l'entreprise (hétérogènes) jusqu'à l'entrepôt de données (modèles multidimensionnels). Ce système ne se contente pas de charger les données, il doit les faire passer par un tas de moulinettes pour les dé-normaliser, les nettoyer, les contextualiser, puis de les charger de la façon adéquate. Nous verrons par la suite ce que chaque mot veut dire. [62]

Ce chapitre est consacré à l’étude de processus ***ETL*** . Cette étude va nous nous permettons par la suite de mener à bien le projet. Pour cela, nous commençons tout d’abord par présenter les principaux phases d’***ETL*** et nous avons présenté une comparaisons entre l’***ETL*** classique et l’***ETL*** ***Mapreduce*** . nous avons classifier les Travaux Antérieurs du processus ***ETL*** . nous présentons ensuite la modélisation ***ETL*** selon l’approche de vassiliadiss . après nous décrivons le partitionnement des données ***Mapreduce*** . nous terminons en présentant l’approche de modélisation dans un environnement ***Mapreduce***.

**1. PROCESSUS ETL**

Les données pertinentes pour la prise de décision sont collectées à partir des sources et intégrées dans l'entrepôt. L'entreposage de ces données se fait par le biais des processus d'extraction de transformation et de chargement ou processus ***ETL***. [34], connecté aux différentes applications et bases de données, l'outil d'***ETL*** se charge de récupérer ces données et de les centraliser dans une base de données particulière. Pour ce faire, le processus d'***ETL*** respecte les trois étapes d'extraction, de transformation et de chargement. [39]

**1.1. La phase d'extraction**

Consiste en l'identification et l'épuration des données, seules les données destinées à l'exploitation pour analyser un sujet bien précis seront gardées. Pour extraire les données utiles, l'outil d'***ETL*** doit pouvoir se connecter aux différentes sources à disposition, qu'il s'agisse d'applications ou des bases en production. [39]

En conséquence, les ***ETL*** utilisent des moteurs d'extraction ou des programmes ***ad hoc***[[8]](#footnote-8) générés par des outils dédiés. En ce sens, ils jouent un rôle d'intégration au niveau des données. [39]

**1.2. La phase de transformation**

Regroupe les opérations de mise au format nécessaire des données, de calcul des données secondaires et de fusion ou d'éclatement des informations composites (par exemple le produit d'une quantité vendue et du prix peut devenir un total de vente). Cette étape de transformation comprend aussi une phase d'agrégation des données. Le niveau d'agrégation est choisi au moment de la construction de l'entrepôt et les données initiales seront perdues. C'est effectivement ce qui permet d'avoir des temps de réponse très courts. [39]

**1.3. La phase de chargement**

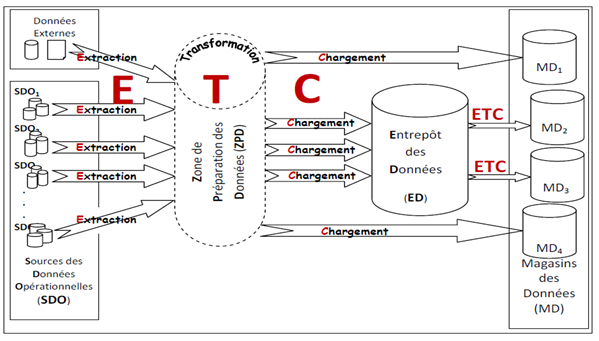
A pour rôle de stocker les informations dans les ***entrepôts de données***. Ce stockage dépend de la manière dont est administré ***l'entrepôt de données***, le chargement de nouvelles données peut bien parfois écraser les données déjà existantes.

L'utilisation des ***ETL*** permet de filtrer et de nettoyer les données qui sont issues souvent de sources variées. La plupart des bases de données volumineuses soufrent de l'existence de données "bruitées". Ce "bruit" est dû à l'hétérogénéité et l'incohérence des données ainsi qu'à la présence de données manquantes ou encore à la duplication de certaines données. D'autres approches se sont intéressées à la réduction de ce "bruit"' existant dans un système d'information.[39]

La plupart des processus ***ETC*** stockent les données avant leur chargement dans l’***ED*** dans une base temporaire dite Zone de Préparation des Données (***ZPD***). La ***ZPD*** est une structure intermédiaire (stockage tampon) qui stocke les données issues des ***SDOs*** avant leur transformation et leur intégration dans l’***ED***. Bien souvent le modèle de données de la ***ZPD*** est un modèle relationnel classique assez proche des modèles des ***SDOs***. [40] [41]

Le ***Magasin de*** ***Données*** (***MD***) est un extrait de l'***ED***. Ces données extraites sont adaptées à une classe de décideurs ou à un usage particulier  [42]. Les ***MDs*** peuvent être dépendants ou indépendants de l’***ED***. Les ***MDs*** dépendants sont construits à partir de l’***ED*** grâce aux outils ***ETC***.[43]

Cependant les ***MDs*** indépendants sont construits directement a partir des sources des données Opérationnelles. Dans la littérature des ***EDs***, nous avons constaté des différentes architectures d'entreposage des données. Dans la figure 2.3, nous décrivons une schématisation de l’architecture générique d’un ***ED*** qui intègre tous ces composants de base. [43]



**Figure 3.1:** processus d’entreposage des données [44].

**2. MODELISATION DE PROCESSUS ETL**

Un processus ***ETL*** regroupe les éléments suivants :

* Sources de données candidates au processus d’entreposage
* L’entrepôt de données, destination finale des données à préparer
* Les tâches d’extraction, de nettoyage, conversion, filtrage, d’agrégation et de chargement

La modélisation devra formaliser tous ces éléments de manière à comprendre le circuit des données depuis les systèmes sources jusqu’à l’entrepôt en passant par les différentes tâches de transformation. Le mappage des données à différents niveaux est un aspect très important pour comprendre le cheminement des données.[45]

**2.1. Modélisation conceptuelle**

Le niveau conceptuel consiste à donner une idée très générale sur l'environnement du projet décisionnel à savoir : les besoins des utilisateurs en termes d'analyses (mesures et dimensions), les sources disponibles qui répondent à ces besoins, principales transformations à faire subir aux données sources avant leur chargement dans l'entrepôt. Ce niveau permet aux concepteurs d'aborder les premières réunions pour discuter de la pertinence des données disponibles, leur qualité et sous quels formats existent-elles. Il s'agit aussi de vérifier si tous les besoins peuvent être satisfaits à partir des sources disponibles et quelles sont les transformations nécessaires pour répondre aux exigences de qualité pour ***l’entrepôt de données***. [45]

Voici les notations graphiques retenues pour la représentation du processus au niveau conceptuel :

|  |  |
| --- | --- |
| Symbol | Désignation |
| Concept | Représente une entité dans la source de données, dans le DSA ou dans l’ED |
| Attribute | Comme dans une modélisation E/R, ils permettent de définir les concepts |
| part of | Cette association permet de relier le concept à ses attributs |
| tronsformation | Abstraction d’un bout ou d’un module complet de code exécutant une tâche  ETL : (1) nettoyage/ filtrage de données (comme violation de la contrainte  PK/FK, (2) transformations de données (comme une agrégation). |
| ETL\_constraint | Permet d’exprimer certaines contraintes sur le contenu de l’ED à travers les attributs de celui-ci (PK, FK, NOT NULL, …) |
| Note | Permet d’expliquer des choix de conception, de préciser une sémantique ou une contrainte à vérifier en temps réel (temps d’exécution, événements, erreurs,…) |
| provider 1:1  Provider N :M | Représentent le mappage entre les données sources (input) et les données de l’ED (output) via une transformation. Le cas simple(1 :1) représente le cas où une donnée source (input) donne en sortie, à travers une transformation, à une donnée de l’ED (output).  Le cas général (N:M) représente le cas où un ensemble de données sources (inputs) transformées donneront naissance à plusieurs données de l’ED (outputs) |
| Serial Coposition | Permet de modéliser le cas où dans une association « Provider » les données passent par plusieurs transformations avant de donner naissance aux données de l’ED. |
| **Active condidate**  **Condidate** **1**  **………**  XOR  **Conditate n** | Permet de préciser les sources de données candidates à l’alimentation de l’ED en mettant en relief la source active. Une des sources pourra être utilisée en même temps (XOR) |

**Tableau 3.1 :** Formalisme de Vassiliadis pour la modélisation au niveau conceptuel [45]

**2.2. Modélisation logique :**

Le passage du niveau conceptuel vers le niveau logique consiste à affiner le modèle en intégrant plus de détails dans le modèle logique : séquence d'exécution des activités, différents schémas de données relatifs à une activité (inputs, outputs, paramètres, données rejetées). En dehors du modèle logique, d’autres aspects sont prévus tels que : planification d'exécution du processus, plan de reprise et de restauration en cas de panne, plan d'administration consistant à faire de l'audit, gestion des accès et sécurité, etc. [45]

Il capture les flux de données à partir des sources vers ***l'entrepôt de données*** grâce à une suite d'activités synchronisées chargées de préparer les enregistrements de données. Il s'agit de préciser le type des données utilisées lors des traitements, les inputs/outputs et paramètres de chaque activité, la séquence des activités qui montre comment les outputs d'une activité A1 sont utilisés comme inputs dans une activité A2. [45]

**2.3. Modélisation physique :**

Le niveau physique qui décrit de manière complète le processus ***ETL*** doit préciser l'environnement sous lequel s'exécutera le processus : OS, nature des systèmes sources (***SGBD***s, ***fichiers plats***, ***Excel***, ***XML***, ...), ressources matérielles en termes de serveurs et stations nécessaires ainsi que les profils utilisateurs nécessaires pour faire aboutir le processus de bout en bout jusqu'au rafraichissement de l'entrepôt de données sans échecs. [45]

**3. TRAVAUX ANTERIEURS**

Il existe plusieurs travaux ont été proposées purement destinées à la phase de conception ***ETL*** , dans cette partie nous présentons les principaux travaux dédiées à la modélisation du processus ***ETL***.

**Kimball et al**.**1998** décrivent les principales activités devant être effectuées pour l’extraction des données des sources, leur nettoyage et transformation et leur chargement dans l’***ED***. Les schémas des sources, de l’***ED*** et les mappings sont définis au niveau physique. Les auteurs ne fournissent pas une méthode détaillée pour la conception, l’alimentation et le déploiement de l’***ED***. **[46]**

**Calvanese et al. (1999)** proposent dans une approche la définition d’un schéma global intermédiaire les schémas des sources de données et le schéma cible de l’***ED*** au niveau conceptuel en utilisant le formalisme E/A [47]

**Labio et al.2000** proposent dans un Framework permettant de décrire les activités de chargement de données (le ***L*** de ***ETL***) dans un ***ED***. Le Framework permet de décrire l’ensemble des activités : celles effectuées avec succès et celles ayant échoué. Les sources considérées sont de type relationnel. Les données sont chargées dans un ***ED*** relationnel. **[48]**

**Raman et al. (2001)** proposent une approche de transformation (le ***T*** de ***ETL***) des données issues de différentes sources pour leur chargement dans un ***ED***. Les données traitées par l’approche sont des données tabulaires (fichiers ou bases de données). L’approche est supportée par un outil nommé Potter’s Wheel. L’outil permet de spécifier les transformations graphiquement de façon itérative et interactive, dans le but de nettoyer les données. L’outil est interactif dans le sens où l’utilisateur peut annuler les transformations non validées. Le chargement des données n’est pas traité dans l’article. **[49]**

**Vassiliadis et al. (2002)** est considérée comme l'une des meilleures contributions dans ce domaine. Ils sont intéressant de visualiser les aspects statiques (données) et dynamiques (tâches ETL) aux niveaux conceptuel, logique et physique. Le formalisme proposé permet de représenter les différents objets manipulés dans le processus ETL ainsi que les associations reliant ces objets (mappage) vers ceux de l'entrepôt de données. Il prend en charge les activités fréquemment utilisées telles que le contrôle des valeurs NULL, contrôle de type clé primaire, les agrégations, etc.[45]

**Trujillo et al. (2003)** proposent dans une approche de modélisation conceptuelle des processus ETL, via un diagramme de classe ***UML***. Chaque activité ou mécanisme ***ETL*** (exemple. agregation, conversion, chargement, union) est représenté par un nouveau stéréotype dans le diagramme. Des notes ***UML*** sont utilisées pour l’explication des fonctionnalités définies et la représentation des mappings entre les attributs des sources et ceux de l’***ED*** cible. [50]

**Luján-Mora et al. (2004)** proposent dans une approche de modélisation physique d’un ***ED*** et des processus ***ETL***, en adaptant les digrammes de composant et de déploiement de la notation ***UML***. Les deux diagrammes doivent être conçus conjointement par le concepteur et l’administrateur de l’**ED**. La structure physique du processus ***ETL*** est modélisée par un diagramme nommé Integration Transportation Diagram (***ITD***). Ce dernier permet de lier les schémas physiques des sources de données avec le schéma physique de l’***ED***. Ce diagramme permet de décrire différents aspects d’implémentation physique du processus ***ETL*** comme les serveurs utilisés, les protocoles de communication, les ***SGBD*** utilisés, etc. [51]

**Simitsis et al. (2008)** proposent dans une approche semi-automatique permettant le passage d’un modèle conceptuel décrivant le processus ***ETL*** vers un modèle logique ***ETL***. L’approche est structurée autour des étapes suivantes : (1) raffiner le modèle conceptuel ***ETL***, éliminer les ambigüités existantes et identifier les sources candidates actives, (2) faire correspondre les concepts et attributs du modèle conceptuel aux structures de données et attributs du modèle logique, (3) appliquer un algorithme de mise en correspondance des transformations du modèle conceptuel avec les activités du modèle logique ***ETL*** et identifier l’ordre d’exécution des activités, (4) enrichir le modèle logique avec les contraintes ***ETL*** imposées sur les données, (5) s’assurer de la cohérence de l’ordre des activités ***ETL***, et enfin procéder à la génération du schéma représentant les séquences des activités ***ETL*** qui représente le schéma logique ***ETL***. [52]

**Skoutas et al. (2010)** proposent dans une approche de conception ***ETL*** basée sur une ontologie ***OWL*** et les annotations sémantiques, afin de spécifier formellement et explicitement la sémantique des schémas sources et cible de l’***ED***. Le but de l’approche est d’automatiser le processus ***ETL***. Les sources et l’***ED*** sont représentés par leurs schémas relationnels. Les auteurs présentent une liste d’opérateurs habituellement rencontrés dans le processus ***ETL*** (exemple : Retrieve, Aggregate, Extract, etc). Une fois les transformations identifiées, les auteurs proposent d’utiliser l’approche présentée dans ( **Simitsis et al ,2008** ) pour définir la séquence des activités ***ETL***.

Cette approche proposée considère uniquement des sources de données relationnelles. Les auteurs étendent l’approche dans (**Skoutas et al ,2007**) en considérant les données structurées (relationnelles) et semi-structurées représentées en ***XML***. [53]

**Bergamaschi et al. (2011)** proposent dans une approche ***ETL*** exploitant un thésaurus pour faciliter l’alimentation de l’***ED***. Les schémas des sources et de l’***ED*** sont préalablement définis au niveau logique relationnel. Un thésaurus décrivant les schémas des sources est défini et exploité pour générer les mappings entre les schémas des sources et le schéma de l’***ED***. Ces mappings sont basés sur des mesures de similarité sémantiques. Une fonction de transformation est définie pour gérer les conflits entre les sources et alimenter le schéma de l’***ED***. [54]

**Liu et al.** (2011) ont proposé une approche parallèle/distribuée appelée ***ETLMR*** consistant à améliorer les performances de la phase de transformation (T) et de chargement (L) de l’***ETL*** et ce en adoptant, pour chacune des deux phases, des stratégies de distribution appropriées. [55]

**Misra et al.** (2013) ont montré que le paradigme ***MapReduce*** est prometteur et que les solutions ***ETL*** basées sur des frameworks open source tel que Apache Hadoop sont plus performantes et moins couteuses par rapport aux solutions ***ETL*** commercialisées. la phase d’extraction (E) de l’***ETL*** a été traitée dans un environnement parallèle/distribué selon le paradigme ***MapReduce***.[55]

**Bala et Alimazighi** (2015) ont proposé une modélisation de l’ETL pour le *Big Data* selon le paradigme *MR* en adoptant le formalisme de Vassiliadis et *al.* (2002) enrichi par des notations graphiques pour modéliser les spécificités du modèle *MR*.[55]

**4. ETUDE COMPARATIVE**

**4.1. Critère de comparaison**

* **Automaticité :**

Le premier critère déterminer si la phase ETL suivre une approche manuelle, semi-automatique ou automatique.

MAN : manuelle

SAU : semi automatique

AUT : automatique

* **Niveau de modélisation :**

Ce critère indique le niveau de modélisation du processus ***ETL*** qui peut être : conceptuel, logique ou physique

CON : conceptuel

LOG : logique

PHY : physique

* **Phase modélisé**

Ce critère indique quelles sont les phases ***ETL*** traités : extraction, transformation ou chargement

E : extraction

T : transformation

L : chargement

* **Notation utilisé**

Ce critère indique la notation ou la langage de modélisation utilisé pour représenter le processus ***ETL*** :le modèle entité-association ,UML ou MDA

E/A : entité association

UML : Unified Modeling Language (langage de modélisation unifié)

MDA : Model Driven Architecture (architecture dirigée par les modèles)

* **Classification**

Ce dernier critère détermine si l’approche est une approche classique ou une approche basé ***mapreduce***. Ce critère sera détaillé dans la section suivante .

Le tableau suivant résume la comparaison des travaux déjà présentés ci-dessus :

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Travail | Automaticité | | | Niveau De Modélisation | | | Type Des Sources | | | Phase Modélisé | | | Notation Utilisé | | | Classification | |
| MAN | SAU | AUT | CON | LOG | PHY | REL | ONT | XML | E | T | L | E/A | UML | MDA | CSQ | MR |
| **Kimball Et Al** | **+** |  |  |  |  | **+** | **+** |  |  | **+** | **+** | **+** |  |  |  | **+** |  |
| **Calvanese Et Al** |  | **+** |  | **+** |  |  | **+** |  |  | **+** | **+** | **+** | **+** |  |  | **+** |  |
| **Labio Et Al** |  |  | **+** |  | **+** |  | **+** |  |  |  |  | **+** |  |  |  | **+** |  |
| **Raman Et Al** |  |  | **+** |  |  | **+** | **+** |  |  |  | **+** |  |  |  |  | **+** |  |
| **Vassiliadis Et Al** | **+** | **+** |  |  | **+** |  | **+** |  |  | **+** | **+** | **+** |  |  |  | **+** |  |
| **Trujillo Et Al** | **+** |  |  |  | **+** |  | **+** |  |  | **+** | **+** | **+** |  | **+** |  | **+** |  |
| **Luján-Mora Et Al** | **+** |  |  |  | **+** | **+** | **+** |  |  | **+** | **+** | **+** |  | **+** |  | **+** |  |
| **Simitsis Et Al** |  | **+** |  |  | **+** |  | **+** |  |  | **+** | **+** | **+** |  |  |  | **+** |  |
| **Skoutas Et Al** |  | **+** |  | **+** |  |  |  | **+** | **+** | **+** | **+** | **+** |  |  |  | **+** |  |
| **Bergamaschi Et Al.** |  |  | **+** | **+** |  |  | **+** |  |  | **+** | **+** | **+** |  |  |  | **+** |  |
| **Liu Et *Al*.** |  | **+** |  |  |  | **+** | **+** | **+** | **+** | **+** | **+** | **+** |  |  |  |  | **+** |
| **Misra Et *Al.*** |  | **+** |  | **+** | **+** | **+** | **+** | **+** | **+** | **+** | **+** | **+** |  |  |  |  | **+** |
| **Bala Et Alimazighi** |  | **+** |  |  |  | **+** | **+** | **+** | **+** | **+** | **+** | **+** |  |  |  |  | **+** |

**Tableau 3.2** : Comparaison des travaux

Après cette comparaison entre les travaux nous avons conclu ces trois points essentiels :

* La plupart des approches sont semi-automatiques ou le concepteur fait partie du processus
* Quelque travaux étudient une partie du processus ***ETL***, Raman et al qui traite la partie de transformation ( t ) ou Labio et al qui intéressent à traiter le chargement des données (L) ,mais la plus pats des traveaux les trois activités du processus ETL. (extraction , transformation , chargement)
* Peu de travaux de recherche proposant des contributions voir des améliorations et adaptations du processus ***ETL*** basé ***mapreduce*** pour les données distribuées.

**5. CLASSIFICATION DES TRAVAUX SUR LE PROCESSUS ETL**

Nous proposons, dans cette section, une classification des travaux sur l’***ETL*** sous forme de deux approches en se basant sur la parallélisation de celui-ci : « approche classique » et « approche basée sur le modèle ***MR***[56]

**5.1. PROCESSUS ETL CLASSIQUE**

Nous considérons que le processus ***ETL*** est classique lorsque celui-ci s’exécute sur un serveur en une seule instance (une seule exécution en même temps) et où les données sont de taille modérée. [56]

Processus ETL avec une instance en execution

F1

F2

F5

F4

F3

Fonctionnalité avec une seule instance en exécution

**Figure 3.2** : Processus ***ETL*** Classique[56]

Dans ce contexte, seules les fonctionnalités d’***ETL*** indépendantes peuvent s’exécuter en parallèle. Une fonctionnalité d’***ETL***, telles que *Changing Data Capture* (***CDC****)*, *Surrogate Key* (***SK****)*, *Slowly Changing Dimension (****SCD****)*, *Surrogate Key Pipeline (****SKP****)*, est une fonction de base qui prend en charge un aspect particulier dans un processus ***ETL***. La fonctionnalité ***CDC***, par exemple, permet dans la phase d’extraction d’un processus ***ETL*** d’identifier, dans les systèmes sources, les tuples affectés par des sources, les tuples affectés par changements afin de les capturer et les considérer dans le rafraichissement du ***DW***. Dans la figure 3.1, nous remarquons que les fonctionnalités *F1* et *F3* ou *F2* et *F3* peuvent s’exécuter en parallèle. [56]

* 1. **PROCESSUS ETL BASE SUR LE MODELE MAPREDUCE**

Le processus ***ETL***, avec un schéma classique, ne pourra pas faire face à l'intégration de données massives Le paradigme ***MR***permet de partitionner de gros volumes de données dont chaque partition sera soumise à une instance du processus ***ETL***. [56]

Processus ETL avec plusieurs instances en exécution

F3

F5

F2

F3

F4

F1

F3

F5

F2

F3

F4

F1

F1

Instance en exécution du processus ETL

F5

F2

F3

F4

F1

Une seule instance en exécution pour les fonctionnalités

**Figure 3.3 :** Processus ***ETL*** basé sur le modèle ***MR*** [56]

Comme le montre la figure 3.2, plusieurs instances du processus ***ETL*** s'exécutent en parallèle où chacune d’elles traitera les données de sa partition dans une phase appelée ***Map***. Les résultats partiels seront fusionnés et agrégés, dans une phase ***Reduce****,* pour obtenir des données prêtes à être chargées dans le ***DW.***[56]

**5.3. PARTITIONNEMENT DES DONNEES**

Dans le but de distribuer/paralléliser le processus ***ETL***, les données sources doivent être elles aussi distribuées pour permettre à plusieurs tâches de s’exécuter de façon parallèle où chacune traite sa propre partition de données. P-ETL offre trois types de partitionnement. Afin d’assurer une charge plus ou moins équitable entre les différentes tâches parallèles, le choix du type de partitionnement est important. La présence d’un taux élevé, dans une partition de données, de tuples avec des valeurs creuses implique une charge faible en termes de traitement pour la tâche. En effet, les tuples en question seront rejetés par un filtre tel que NOT NULL. [56]

* **Simple** : étant donné un volume de données source v, le type de partitionnement simple génère des partitions égales selon l’équation 1 où nb\_part étant le nombre de partitions. [56]

*taille(partition) = taille(v)/nb\_part*

* **Round Robin (RR)** : Avec la technique Round Robin, l’affectation d’un tuple depuis le volume source v vers une partition de données p est basée sur l’équation 2. rang (tuple) étant le rang du tuple dans le volume v et nb\_part étant le nombre de partitions. [56]

*p = rang(tuple) mod nb\_part*

* **Round Robin par Bloc (RRB)** : Cette technique est similaire à Round Robin. Dans le but d’accélérer le partitionnement, un bloc de tuples est affecté à la partition, plutôt qu’une affectation tuple par tuple . [56]

**CONCLUSION**

Au cours de ce chapitre, nous avons vu une étude large sur le processus ***ETL*** :

Nous avons présenter leur principaux phases , le principe de fonctionnement de chaque phases et nous avons présenter la modélisation ***ETL*** au niveau conceptuel ,logique et physique , nous citons les travaux déjà réaliser dans ce domaine , Nous présentons aussi une étude comparative du travaux .

Finalement, nous avons présenter une comparaison entre le processus ***ETL*** classique et le processus ***ETL*** basé sur ***mapreduce*** , et nous décrivons le partitionnement des données .

Dans le chapitre qui suit nous avons présenté la maquette expérimentale, de notre projet .

|  |
| --- |
| CHAPITRE **4** |
| **CONTRIBUTION**  **ET IMPLIMENTATION** |

|  |
| --- |
| **PLAN DU CHAPITRE** |

**Introduction**

**1. Présentation Des Environnements**

1.1. Cloudera Cdh

1.2. Machine Virtuelle

1.3. Hive

**2. Les Sources De Données**

2.1. Fichier Csv

2.1.1. Définition

**3. Contribution**

**3.1.L’architecture**

**3.2. Implémentation**

3.2.1. Importation Des Données

3.2.2. Extraction

3.2.3. Chargement

3.2.4 Transformation

**Conclusion**

**INTRODUCTION**

Ce chapitre présente toutes les étapes que nous avons suivis pour réalisée notre implémentation qui rende le ***Mapreduce*** intégré dans les trois phases du processus ***ETL*** , allant de la description des paramètres machines sur lesquels cette application va fonctionner, passant par l’ architecture qui nous proposons, jusqu'aux testes et résultats sur l'application finale.

Donc nous expliquons notre contributions dans ce chapitre, qui comprend une architecture et expérimentale que nous avons propose pour l'adaptation du processus d'extraction –transformation- chargement .

**1. PRESENTATION DES ENVIRONNEMENTS**

**1.1. 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***.

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.[57]

**1.2. MACHINE VIRTUELLE**

Logiciel installé dans un ordinateur et permettant de simuler le fonctionnement d'un dispositif matériel qui, normalement, ne pourrait pas fonctionner avec cet ordinateur. Grâce à une machine virtuelle, on peut, par exemple, simuler l'utilisation d'un processeur, ou d'une console de jeux sur un PC. On pourra ainsi développer et tester de nouveaux jeux pour la console sans la posséder réellement. [58]

**1.3. HIVE**

Apache ***Hive*** est un système [***d'entrepôt de données***](http://www.lemagit.fr/definition/Data-warehouse-entrepot-de-donnees)open source. Il permet d'interroger et d'analyser des ensembles de données volumineux ([***Big Data***](http://www.lemagit.fr/definition/Big-Data-Gestion)) stockés dans des [fichiers ***Hadoop***](http://www.lemagit.fr/definition/Hadoop-Distributed-File-System-HDFS). En matière de données, ***Hive*** présente trois fonctions principales : la synthèse, l’interrogation et l’analyse.[59]

Il prend en charge des requêtes rédigées en langage HiveQL. Il traduit automatiquement les requêtes de type ***SQL*** en tâches ***MapReduce*** exécutées sur ***Hadoop***. Parallèlement, ***HiveQL*** prend en charge les scripts ***MapReduce*** personnalisés qui se connectent aux requêtes. Hive autorise également la sérialisation/désérialisation des données, et accroît la flexibilité de la conception de schémas en intégrant un catalogue système appelé ***Hive-Metastore***. [59]

***Hive*** prend en charge les fichiers au format texte (appelés également fichiers plats), les fichiers SequenceFiles (fichiers plats constitués de paires clé/valeur binaires) et RCFiles (Record Columnar Files ; stockent des colonnes dans une table selon le mode d'une base de données en colonnes.) [59]

**2. LES SOURCES DE DONNEES**

**2.1. FICHIER CSV**

Le format ***CSV*** (Comma Separated Values) en français, « valeurs séparées par des virgules ». Il s'agit d'un format de fichiers ouvert qui permet de stocker les données d'un tableau. Chaque ligne du fichier correspond à une ligne du tableau ; les colonnes sont en général séparées par des virgules. On peut très bien remplacer les virgules par des tabulations ou tout autre caractère. Ce format est utilisé pour échanger de manière interopérable des données de tableurs, bases de données, annuaires, *etc.* entre logiciels différents et/ou plateformes différentes.[60]

**3. CONTRIBUTION**

On se basant à la contribution de MAHFOUD bala qui a remarqué que l'approche de vassiliadiss ne modélise pas la distribution des données et le découpage du processus pour une exécution parallèle, pour ça il intègre le partitionnement des données et les phases Map et Reduce .

Notre contribution consiste à :

* proposer une architecture qui modélise le processus ***ETL*** basé on utilisant ***Hive*** ***MapReduce*** dans les 3 phases du processus
* élargir la contribution de MAHFOUD bala au niveau physique par une proposition d’une implémentation orienté ***MapReduce*** concernant cette approche.

**3.1 L’ARCHITECTURE**

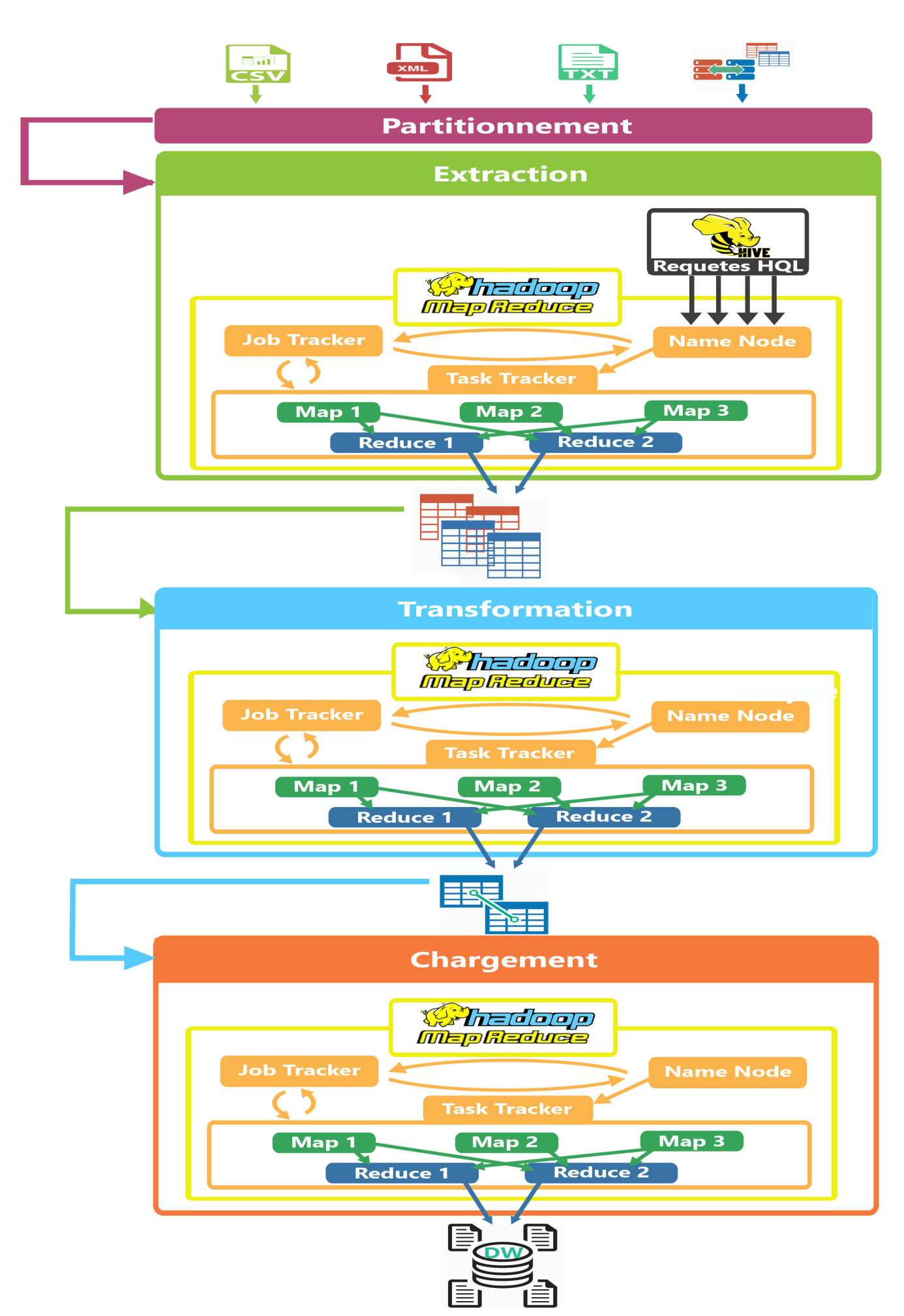
La figure 4. 1 illustre l’exécution d’un job ***MapReduce***, concerne les entités suivant :

**Hive** **:** permet de compiler, optimiser et exécuter des requêtes ***SQL*** sur un cluster ***hadoop*** en vue d’analyse et d’agréger les données.

**Jobtracker** **:** partage l’exécution de requête en sous taches et soumet les taches au ***Tasktrackers***, le ***Jobtracker*** suivis l’exécution de ses taches ce suivi est pour garantie la tolérance au panes du system.

**Name node :** partitionnerles données de ***HDFS*** des blocks de taille 64Mo (par défaut) et les distribuer aux ***Tasktrakers***.

**Tasktracker** **:** exécute les taches que le ***Jobtracker*** lui a envoyées sur les données que la Name node lui a envoyées.

****

**Figure 4.1** : architecture proposé modélise le processus ***ETL*** basé sur ***MAPREDUCE***

**FONCTIONNEMENT  :**

Après le partitionnement des données des sources avec la méthode round robine, pour minimiser le temps de recherche.

On commence par la première phase d’***ETL***

* **La** **phase d’extraction** :

Connecter aux différentes sources hétérogène et distribué pour extraire des données destinées à l'exploitation pour analyser un sujet bien précis sera gardées, le but de cette phase est le nettoyage des donnes. Cette extraction se faire à travers des requêtes ***SQL***.

* ***Hive*** communique avec le ***Jobtracker*** pour lancer le travail de ***MapReduce***.
* Le ***Jobtracker*** reçoit le job ***MapReduce*** pour partager le job et établit une liste des sous taches qui seront donnée au ***Tasktrackers***
* Dans la figure on vue des pourparlers de ***Jobtracker*** au ***Name node*** pour :
* Déterminer l’emplacement des données .
* Copie avec une réplication haute les blocs des données dans un système de fichier du ***Jobtracker*** pour les ***Tasktrackers*** .
* Le ***Tasktracker*** communique avec ***Jobtracker*** pour signifier leur disponibilité a exécuter les jobs, si le ***Jobtracker*** possède des jobs en file d’attente il affecté la tache au ***Tasktracker***
* Le **Tasktracker** copie le fichier depuis le système de fichier
* Ensuite le ***Jobtracker*** ordonnance les différentes tâches des jobs soumis et assigne les tâches aux ***Tasktrackers***.
* **la phase de transformation**

Apres l’exécution du Reduce de la phase d’extraction les résultats obtenu seront transformer et filtrées et agrégés avec des fonctions d’agrégation (SUM, CAUNT, etc…) . C'est une suite d'opérations qui a pour but de rendre les données cibles homogènes et puissent être traitées de façon cohérente.

L'ensemble des données sources, après nettoyage ou transformation d'après des règles précises ou par application de programmes (pour un contrôle de vraisemblance par des méthodes statistiques), seront restructurées et converties dans un format cible.

On Applique cette phase de transformation avec le même mécanisme appliqué dans l’extraction ***MR*** , Pour obtenir des données prêts a être chargé dans ***l’entrepôt de données*** .

* **la phase du chargement**

C’est l'opération qui consiste à charger les données nettoyées et préparées dans le ***DW*** avec le même mécanisme ***MR*** appliqué dans la phase d’extraction et de transformation.

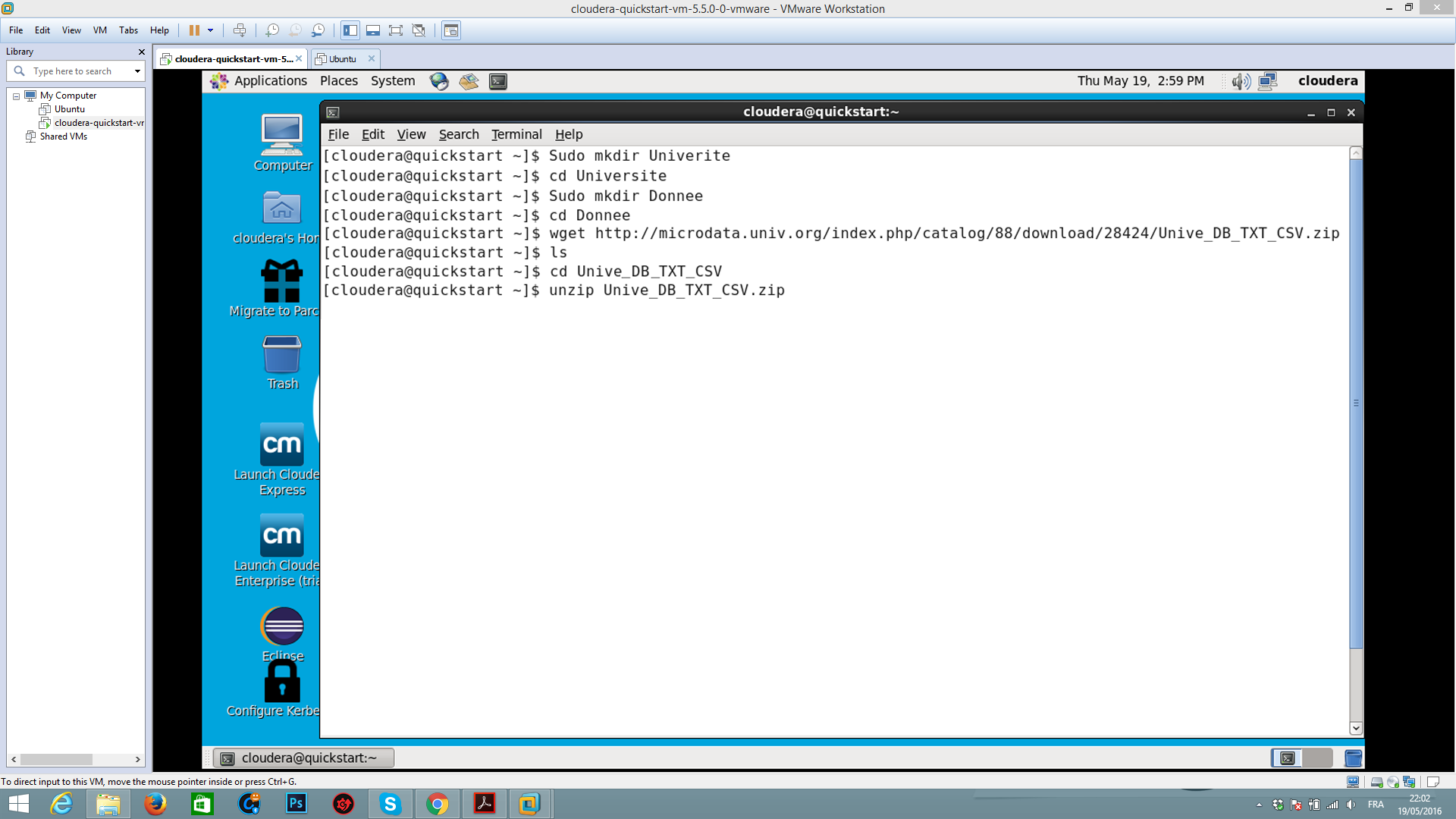
**3.2. IMPLEMENTATION**

* Dans notre implémentation nous avons appliquer les principe de notre architecture proposé
* Nous avons installé le cloudera dans une machine à les caractéristique suivant :
* ISUS ROGG75VW
* INTEL I7 3.5 GHZ ( Octa core )
* RAM 16 GO
* 750 GO HDD + 256 GO SDD
* Dans notre implémentation en a utilisé vmware workstation 12 Pro
* 4 CORES
* 12 GO RAM
* 64 GO SDD

Nous avons suivis les étapes suivants :

**3.2.1. IMPORTATION DES DONNEES**

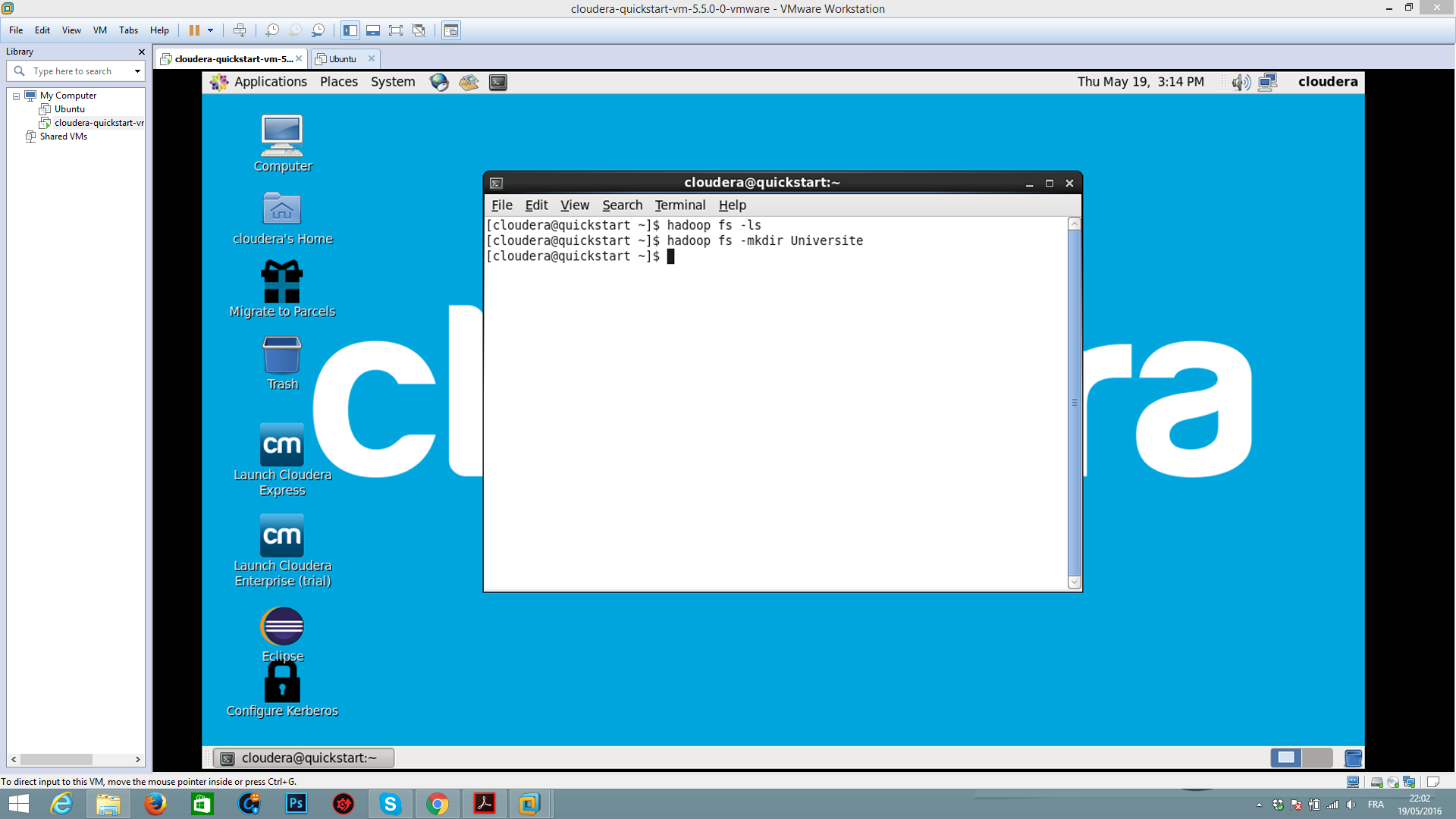
* Nous commençons par télécharger les fichiers .zip contenant des fichiers .CSV et .TXT
* Création de la structure de dossier pour le cas d'utilisation



**Figure 4.2 :** Création de la structure des données

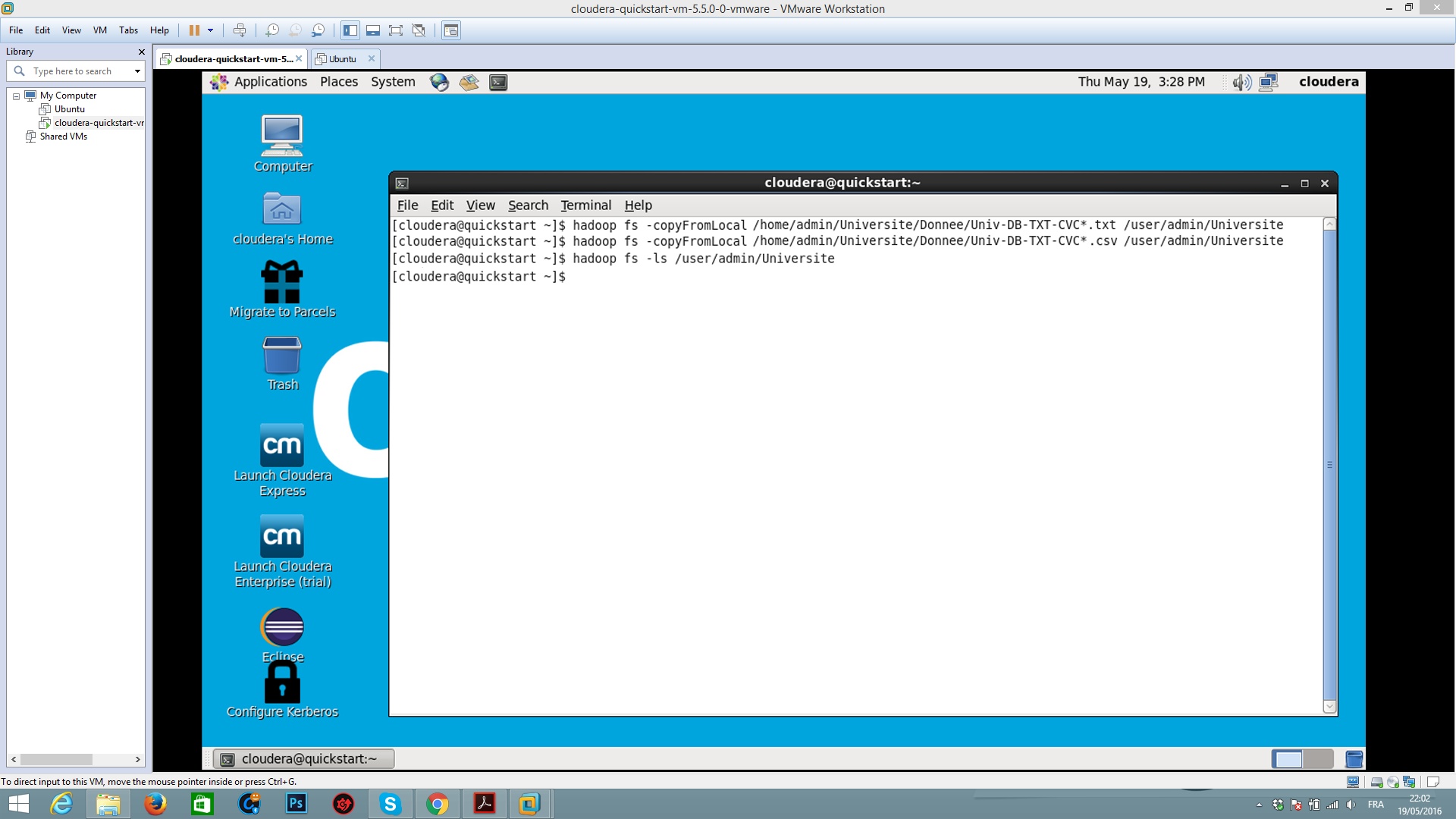
**3.2.2. EXTRACTION**

* Charger les données dans ***Hadoop*** , en utilisant la commande de terminal suivante :



**Figure 4.3 :** Chargement des données dans ***hadoop***

* Après la création du répertoire de l’université dans le HDFS, nous pouvons ajouter les fichiers .CSV et .TXT pour Hadoop, en utilisant le code suivant:

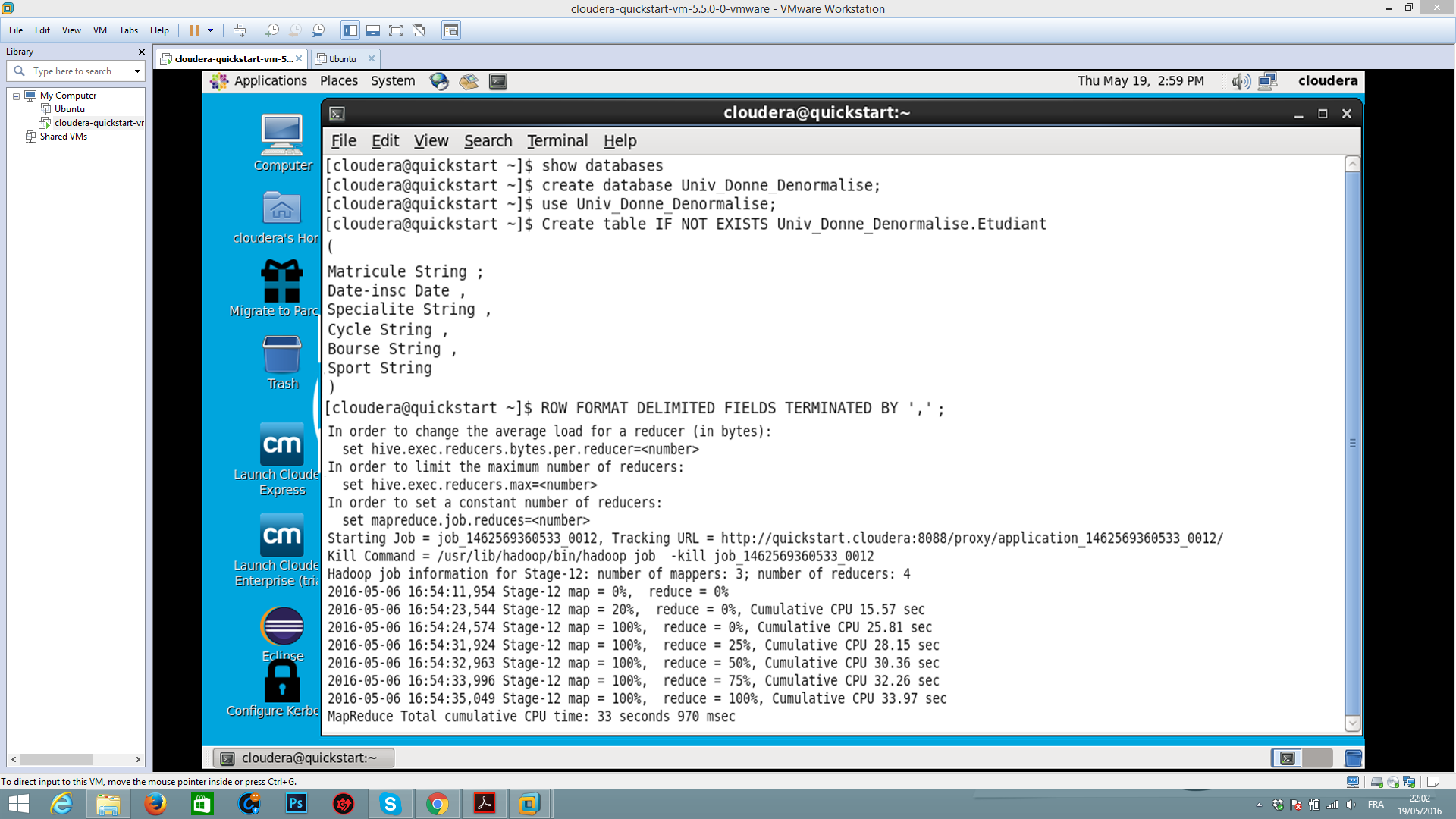


**Figure 4.4 :** Extraction des données depuis les les fichiers .CSV et .TXT

* Les fichiers .CSV et .TXT sont maintenant dans l’HDFS
* L'étape suivante consiste à transformer ces fichiers avec Hive.

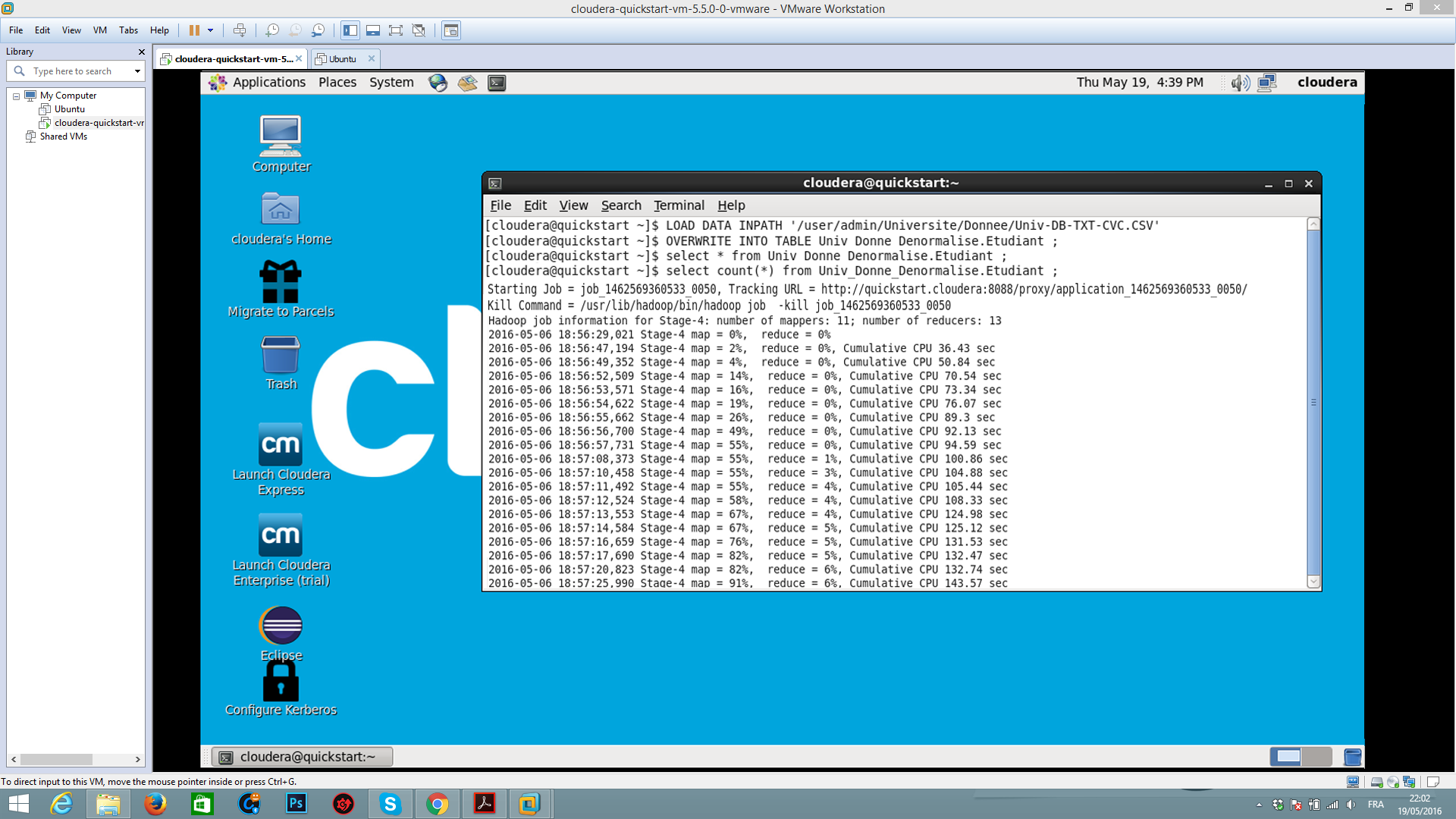
**3.2.3. TRANSFORMATION**

* Les fichiers sont de différents structure , donc , pour continuer le processus il faut normaliser et agréger les données .
* Pour déterminer le modèle de base de donnée : on a des tables de base, après avoir créé et chargez ces tables. on a crée une table maître dénormalisé ‘Univ\_Donne\_Denormalise’, Cette table intègre toutes les données des autres tables (certaines colonnes seront nulles).
* Après que la table maître a été créé et chargé, la dernière étape consiste à agréger des colonnes et construire la table Maître Transformé.
* Les colonnes suivants sont celles de l’étudiant qu’ils faut les ajouts dans notre table
* Matricule
* Date\_insc
* Cycle
* Spécialité
* Bourse
* Sport



**Figure 4.5 :** Transformation des données

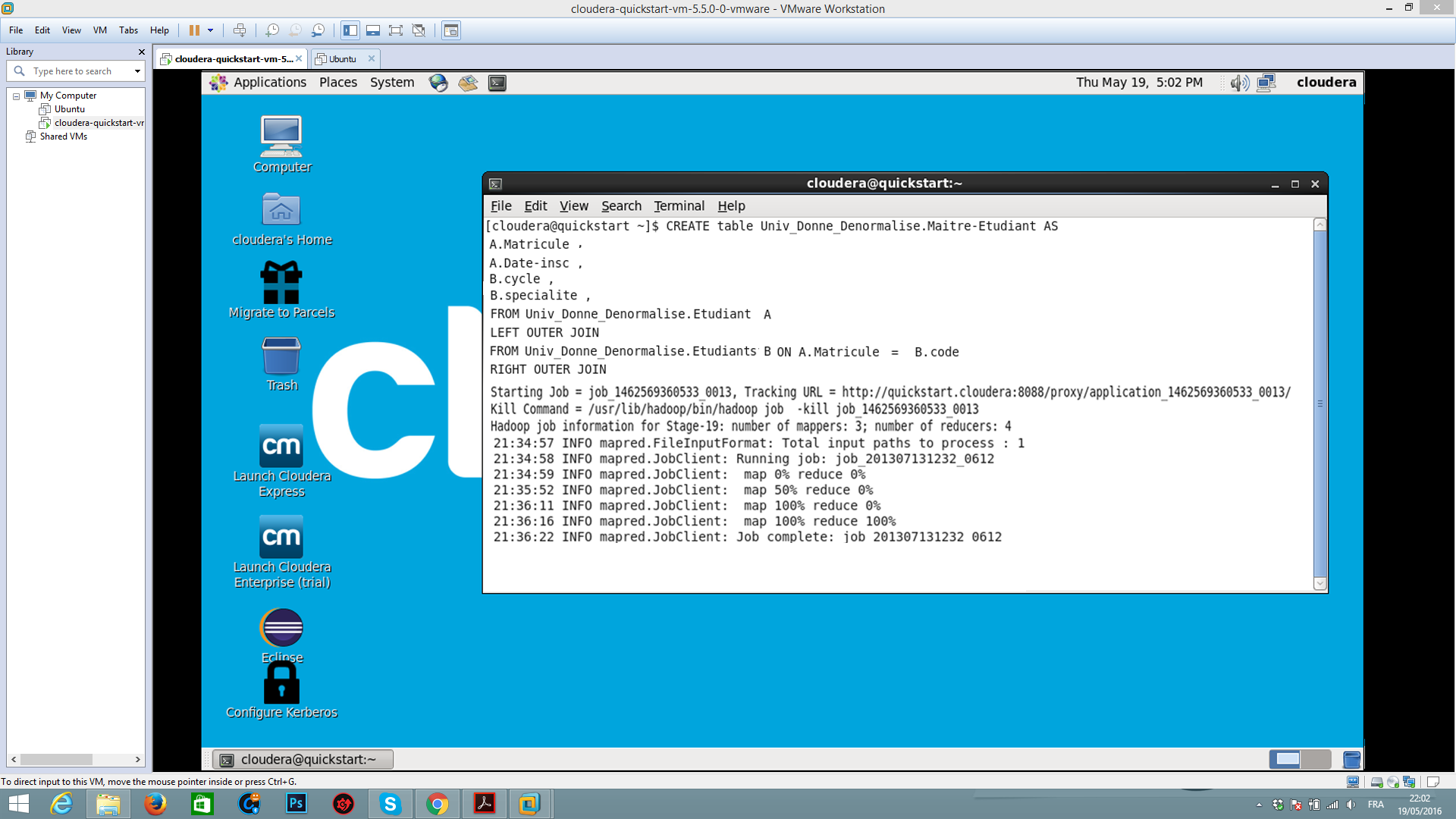
* Ensuite, nous commençons à construire la base de données
* Ouvrir le terminal de Hive pour exécuter le code suivant :



**Figure 4.6 :** Construction du base de données

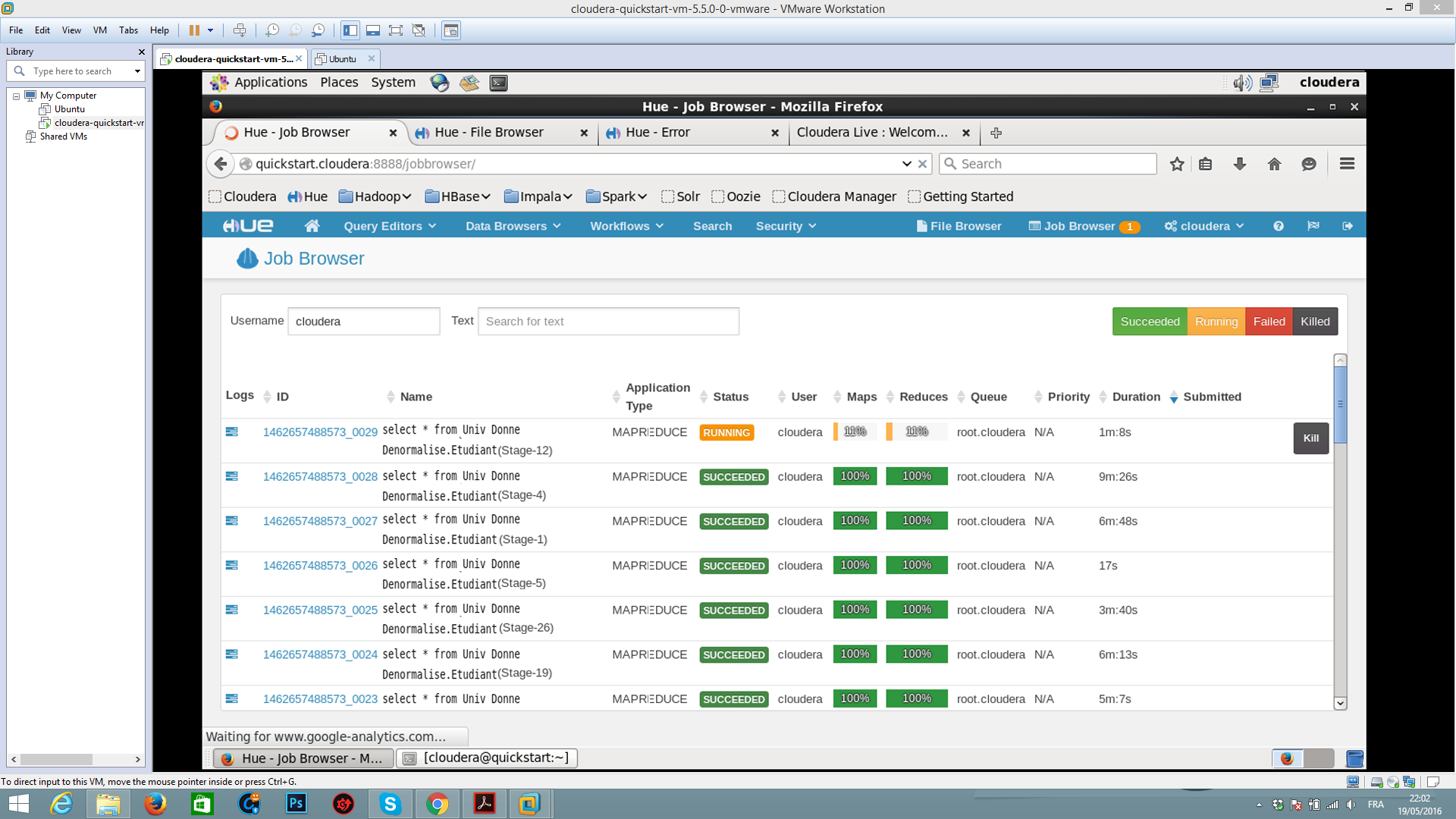
**3.2.4. CHARGEMENT**

* Après les tables ont été créés, ils sont prêts à être remplis en utilisant la commande suivante:



**Figure 4.7 :** Chargement des données

La figure suivante représente un suivi d’exécution d’une requête (select \* from Univ\_Donne\_Denormalise.Etudiant)



**Figure 4.8 :** Suivi d’exécution d’une requête

**CONCLUSION**

Dans ce chapitre La première partie présente les différents outils ainsi les différentes sources de données qui se sont utilisé dans cette implémentation.

Dans la deuxième partie nous avons expliqué notre architecture proposé , qui consiste a illustrer l’intégration du ***Mapreduce*** dans les trois phases du processus ***ETL*** .

Puis nous avons présenté notre implémentation en détaillons ses principaux phases et nous avons présenté les résultats obtenus pendant ce travail.

**CONCLUSION GENERALE**

Les systèmes d'informations décisionnels se trouvent aujourd’hui face à un défi majeur caractérisé par les ***Big Data*** et doivent donc s’adapter aux nouveaux paradigmes tels que le ***cloud computing*** et le modèle ***MR***. Le processus ***ETL*** étant le cœur d'un système d'informations décisionnel puisque toutes les données destinées à l’analyse y transitent par celui-ci. Il faudra étudier de manière profonde l’impact de ces nouvelles technologies sur l’architecture, la modélisation et les performances de l’***ETL***. Dans ce contexte, nous avons proposé une architecture pour les processus ***ETL*** dont les fonctionnalités s’exécutent selon le modèle ***MR***.[56]

Le [***Big Data***](http://www.piloter.org/business-intelligence/big-data-definition.htm) est dans une toute autre logique que le [***traditionnel Data Warehouse***](http://www.piloter.org/business-intelligence/datawarehouse.htm)***.*** Il s'agit en effet de stocker de très grandes quantités de données et de gérer et d'analyser des flux massifs d'informations en continu. L'***ETL*** est trop complexe et surtout trop coûteux et trop lent pour répondre aux besoins du ***Big data*** sans le dénaturer.

Nous avons étudie le processus extraction -transformation –chargement , cette étude englobe un état de l’art consiste à rassembler le maximum d'informations sur ce domaine et une proposition d’architecture d’***ETL*** donc nous avons intègre le ***Mapreduce*** sur les trois phases du processus ( extraction , transformation , chargement)

L’expérimentation effectuée nous a permis de confirmer les avantages et les performances offertes par le processus proposé qui sont principalement la gestion totale et continue de la qualité des données et l’amélioration du temps d’entreposage des données.

Enfin, après la discutions des résultats obtenus du travail que nous avons mené dans ce projet ouvre de nouvelles perspectives pour améliorer le processus ***ETL***, par exemple l’étude sur le nouveau processus ***ELT***.

**BIBLIOGRAPHIE**

**Référence pour une thèse :**

|  |  |
| --- | --- |
| [16] | Ange GUEDOP YIMGANG Et Kouassi Edem HOUMAVO , 2014 , « etude exploratoire du ‘‘hadoop mapreduce’’ pour le traitement d’images de télédétection » |
| [24] | KOUEDI Emmanuel, 2012, Approche de migration d’une base de données relationnelle vers une base de données NoSQL orientée colonne. |
| [26] | Abdelkader Adla,2010, « Aide à la Facilitation pour une prise de Décision Collective : Proposition d'un Modèle et d'un Outil » |
| [27] | MARHOUMI Fatima Ezzahra ,2006, « Entrepôts de données XML : Développement d'un outil Extraction Transformation Load (ETL) » |
| [29] | Fatma ABDELHÉDI , 2014 , « Conception assistée d’entrepôts de données et de documents XML pour l’analyse OLAP » |
| [31] | Cécile Favre, 12 Décembre 2008, « Évolution de schémas dans les entrepôts de données :mise à jour de hiérarchies de dimensionpour la personnalisation des analyses » |
| [32] | Serna Encinas Maria Trinidad , 30 Oct 2007, “Entrepôts de données pour l'aide à la décision médicale: conception et expérimentation » |
| [33] | Soumia BENKRID, 2014 , « Le déploiement, une phase à part entière dans le cycle de vie des entrepôts de données : application aux plateformes parallèles » |
| [34] | Faten ATIGUI , 2013 , « Approche dirigée par les modèles pour l’implantation et la réduction d’entrepôts de données » |
| [38] | Ghozzi Faiza , 2010 , « CONCEPTION ET MANIPULATION DE BASES DE DONNEES DIMENSIONNELLES à CONTRAINTES |
| [39] | Lamiaa Naoum ,2010 « Un modéle multidimensionnel pour un processus d'analyse en ligne de résumés flous », Thèse de doctorat de l’Université Université de Nantes. |
| [43] | Louardi BRADJI ,2012 , « Adaptation des techniques de l'Extraction des Connaissances à partir des Données (ECD) pour prendre en charge la qualité des données » |
| [57] | [BENAOUDA SID AHMED AMINE](http://dspace.univ-tlemcen.dz/browse?type=author&value=BENAOUDA%2C+SID+AHMED+AMINE),12 octobre 2015,  « Implantation du modèle MapReduce dans l’environnement distribué Hadoop : Distribution Cloudera », |
| [60] | CANTALOUBE J ,2013, « Manipuler les données : XML et CSV » |

**Référence pour un livre :**

|  |  |
| --- | --- |
| [11] | Benjamin Renaut, 2016 « Hadoop/ Big Data » |
| [35] | NEGRE Elsa, 2016, « Entrepôts de données » |
| [21] | Meyer Léonard, 2014*,* « L’AVENIR DU NoSQL » |
| [42] | Bernard ESPINASSE, 2013, « Introduction aux entrepôts de Données » |

**Référence pour un journal :**

|  |  |
| --- | --- |
| [5] | Mickaël CORINUS, Thomas DEREY, Jérémie MARGUERIE, William TÉCHER, Nicolas VIC, 2012, « Rapport d’étude sur le Big Data » |
| [10] | Abdelfettah Idri , Azedine Boulmakoul , 2015 , «  Aspects techniques d’un modèle de fouille de données Cloud basé sur le principe Map/Reduce de Google |
| [40] | E. Ziyati, A. El Qadi, A. Driss, 2010,"Genetic optimization to data warehouse Design" |
| [41] | P. Ponniah,2001, "Data Warehousing Fundamentals” |
| [44] | W. H. Inmon, 2005 ,"Building the Data Warehouse" |
| [45] | Mahfoud Bala , Zaia Alimazighi ,avril 2012 « ETL-XDesign : outil d'aide à la modélisation de Processus ETL ». |
| [46] | R. Kimball, L. Reeves, W. Thornthwaite, M. Ross, and W. Thornwaite. ,1998 “The Data Warehouse Lifecycle Toolkit : Expert Methods for Designing, Developing and Deploying Data Warehouses.” |
| [47] | D. Calvanese, G. De Giacomo, M. Lenzerini, D. Nardi, and R. Rosati.,1999 “A principled approach to data integration and reconciliation in data warehousing.” |
| [48] | W. J. Labio, J. L. Wiener, H. Garcia-Molina, and V. Gorelik.,2000, “Efficient resumption of interrupted warehouse loads”. |
| [49] | Raman01. Potter’s wheel ,2001, “An interactive data cleaning system”. |
| [50] | Trujillo and S. Luján-Mora.,2003, “A uml based approach for modeling etl processes in data warehouses.” |
| [51] | S. Luján-Mora and J. Trujillo.,2004, “Physical modeling of data warehouses using uml. In Proceedings” |
| [52] | A. Simitsis and P. Vassiliadis.,2008, “A method for the mapping of conceptual designs to logical blueprints for etl processes” |
| [53] | A. Simitsis, D. Skoutas, and M. Castellanos.,2010, “Representation of conceptual etl designs in natural language using semantic web technology”. |
| [54] | S. Bergamaschi, F. Guerra, M. Orsini, C. Sartori, and M. Vincini. ,2011, “A semantic approach to etl technologies”. |
| [55] | Mahfoud Bala, Oussama Mokeddem, Omar Boussaid, Zaia Alimazighi ,2015 « Une Plateforme ETL parallèle et distribuée pour l'intégration de données massives » |
| [56] | Mahfoud Bala, Omar Boussaid, Zaia Alimazighi, Fadila Bentayeb ,2015« PF-ETL : vers l'intégration de données massives dans les fonctionnalités d’ETL » |
| [61] | Cécile Favre, Fadila Bentayeb, Omar Boussaid, Jérome Darmont, Gérald Gavin, Nouria Harbi, Nadia Kabachi, Sabine Loudcher , 2016 , « Les entrepôts de données pour les nuls. . . ou pas ! » |

**Référence pour un site :**

|  |  |
| --- | --- |
| [1] | <http://www.kinaze.org/qu-est-ce-que-le-big-data-bigdata-definition/> |
| [2] | Martin Lessard de Zeroseconde.com ,2010 |
| [3] | <http://searchcloudcomputing.techtarget.com/definition/big-data-Big-Data> |
| [4] | [http://www.futura-sciences.com/magazines/high-tech/infos/dico/d/informatique-big-data- 15028/](http://www.futura-sciences.com/magazines/high-tech/infos/dico/d/informatique-big-data-%20%20%20%2015028/) |
| [6] | <http://www.redsen-consulting.com/fr/inspired/data-analyse/big-data> |
| [7] | <http://www.journaldunet.com/solutions/expert/51696/les-3-v-du-big-data---volume--vitesse-et-variete.shtml> |
| [8] | http://blog.businessdecision.com/bigdata/2015/01/3v-opportunites-big-data/ |
| [9] | <http://www.zdnet.fr/actualites/big-data-3-4-ou-5-v-39832210.htm> |
| [12] | <http://mbaron.developpez.com/tutoriels/bigdata/hadoop/introduction-hdfs-map-reduce/> |
| [13] | <http://www.journaldunet.com/solutions/systemes-reseaux/definition-d-hadoop.shtml> |
| [14] | <http://www.silicon.fr/hadoop-declin-cloudera-hortonworks-mapr-repondent-117774.html> |
| [15] | <http://www.lemagit.fr/definition/Hadoop-Distributed-File-System-HDFS> |
| [17] | <http://big-data.developpez.com/faq/hadoop/?page=HDFS> |
| [18] | <https://blog.groupe-sii.com/presentation-hadoop-distributed-file-system/> |
| [19] | <https://codemphasis.wordpress.com/tag/hadoop/> |
| [20] | <http://hadooper.blogspot.com/> |
| [22] | <http://www.lemagit.fr/conseil/Quel-SGBD-NoSQL-pour-vos-besoins-IT-Criteres-de-choix> |
| [23] | <http://blog.neoxia.com/nosql-5-minutes-pour-comprendre/> |
| [25] | <http://www.grappa.univ-lille3.fr/polys/fouille/sortie003.html> |
| [28] | <http://wwwigm.univmlv.fr/~dr/XPOSE2009/informatique_decisionnelle_olap/definition.html> |
| [30] | <http://perso.univ-lyon1.fr/haytham.elghazel/BI/presentation.html> |
| [36] | <http://www-igm.univ-mlv.fr/~dr/XPOSE2005/entrepot/datawarehouse.html> |
| [37] | <http://wwwigm.univmlv.fr/~dr/XPOSE2009/informatique_decisionnelle_olap/md.html> |
| [58] | <http://www.journaldunet.com/encyclopedie/definition/641/34/20/machine_virtuelle.shtml> |
| [59] | <http://www.lemagit.fr/definition/Hive-Apache-Hive> |
| [62] | http://grim.developpez.com/articles/concepts/etl/ |

1. Bertrand Bathelot ,2016 [↑](#footnote-ref-1)
2. Le RAID (*Redundant Array of Independent Disks* ) est un ensemble de techniques de [virtualisation du stockage](https://fr.wikipedia.org/wiki/Virtualisation_du_stockage) permettant de répartir des [données](https://fr.wikipedia.org/wiki/Donn%C3%A9e_(informatique)) sur plusieurs [disques durs](https://fr.wikipedia.org/wiki/Disque_dur) afin d'améliorer soit les performances, soit la sécurité ou la tolérance aux pannes de l'ensemble du ou des systèmes. [↑](#footnote-ref-2)
3. Il existe dans le langage Java de nombreuses classes pour stocker ses données et les parcourir facilement. La classes Hashmap permet de stocker des données sous la forme clé - valeur dans une variable. [↑](#footnote-ref-3)
4. Les applications de base de données avec traitement des transactions en ligne sont idéales pour gérer les données évolutives. Dans ces applications, il y a généralement de nombreux utilisateurs qui exécutent des transactions en même temps, ce qui modifie les données en temps réel. [↑](#footnote-ref-4)
5. Un fait représente un sujet d'analyse. Il est constitué de plusieurs mesures relatives au sujet traité. Ces mesures sont numériques et généralement valorisées de façon continue. [↑](#footnote-ref-5)
6. Les dimensions sont un concept essentiel dans les bases de données multidimensionnelles. Elles sont les critères suivant lesquels on souhaite évaluer, quantifier et qualifier le fait. [↑](#footnote-ref-6)
7. Un cube est une représentation abstraite d'informations multidimensionnelles exclusivement numérique utilisé par l'approche OLAP (acronyme de On-line Analytical Processing) [↑](#footnote-ref-7)
8. adhoc est un outil de structuration documentaire permettant de gérer de manière souple, dynamique et entièrement paramétrable une base d'objets structurés et centralisés pouvant contenir un grand nombre de notices (jusqu’à 1 million). [↑](#footnote-ref-8)