République Algérienne Démocratique et Populaire
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique

Université A/Mira de Béjaïa
Faculté des Sciences Exactes
Département d'Informatique

Mémoire de Fin de Cycle

En vue de l'obtention du diplôme Master 2 en Informatique

Option : Administration et Sécurité des Réseaux

Thème

Mise en Oeuvre d'un Entrepôt de Données sous Hadoop

Réalisé par :

M^lle CHIKH Fatima.
M^lle ZADI Warda.

Devant le jury composé de :

Présidente :	M^me . TAHAKOURT Zineb.
Examinatrice :	M^me . BOUTRID Samia.
Promoteur :	M^r . SEBAA Abderrazak.
Co-Promotrice :	M^lle . NOUICER Amina.

Promotion 2015/2016

Remerciements

**Nous remercions le bon Dieu le tout puissant de nous avoir accordées le courage et la patience
pour mener à terme le présent mémoire.**

Nous tenons, également, à exprimer notre sincère reconnaissance et notre profonde gratitude à notre encadrant Mr SEBAA AbdErrezak, pour sa disponibilité, ses orientations, ses precieux conseils et ses encouragements qui nous ont permis de mener à bien ce travail.

Nous remercions chaleureusement notre co-encadrante Mlle NOUICER Amina pour sa disponibilité, ses orientations et ses conseils.
Nous tennons à exprimer notre gratitude aux membres de jury pour avoir accepté de juger ce travail.

Un merci particulier à nos parents, pour leur amour, leur sacrifices et leurs patiences.

Un énorme merci à nos familles et amis pour leurs éternel soutient et la confiance qu'ils ont en nos capacité.

Dédicaces

Je dédie ce modeste travail à :

A mes parents,

A mes frères et soeurs,

A toute la famille,

A mes amies et collègues, et tous ceux qui m'ont aidé ;

A ma binôme Warda et sa famille.

CHIKH Fatima

Dédicaces

Je dédie ce modeste travail à :

A mes parents,

A mes frères et soeurs,

A toute la famille,

A mes amies et collègues, et tous ceux qui m'ont aidé ;

A ma binôme Fatima et sa famille.

ZADI Warda

TABLE DES MATIÈRES

LISTE DES TABLEAUX

TABLE DES FIGURES

Liste des abréviations

3D	TriDimensionnel
AmazonEMR	Amazon Elastique MapReduce
ANSI	American National Standards Institute
API	Application Programming Interface
BI	Business Intelligence
CDH	Cloudera entreprise Data Hub
CLI	Command Line Interface
DDL	Data Definition Language
ED	Entrepôt de Données
EDF	Electricité De France
ETL	Extraction, Transformation and Loading
GFS	Google File System
Go	Giga octet
Hadoop	High-availability distributed object-oriented platform
Hadoop-GIS	Hadoop Geographic Information System
HWI	Hive Web Interface
HDFS	Distributed Lifetime Maximization.
HiveQL	Hive Querry Lnaguage
HiveSP	Hive SPatial
HOLAP	Hybrid OnLine Analytical Processing
HTML	Hypertext Markup Language
IBM	International Business Machines
ISO	International Organization for Standardization
JDBC	Java DataBase Connectivity
MOLAP	Multidimensional OnLine Analytical Processing

MySQL	My Structured Query Language
NDFS	Nutch Distributed File System
NCBI	National Center for Biotechnology Information
NoSQL	No Structured Query Language
ODBC	Open DataBase Connection
OLAP	Online Analytical Processing
OLEDB	Open Linking and Embedding for Database
OLTP	OnLine Transaction Processing
OpenSSH	Open Secure SHell
PHP	Hypertext Preprocessor
RAM	Random Access Memory
ROLAP	Relational OnLine Analytical Processing
SerDe	Sérialiseur/Désérialiseur
SI	Système d'Information
SGBD	Système de Gestion de Base de Données
SGBDR	Système de Gestion de Base de Données Relationnelle
SQL	Structured Query Language
SQL/MM	Structured Query Language Multi-Media
SSH	Secure SHell
TB	TeraByte
UDTF	User Defined Tabular Function
USA	United States of America
U-SQL	User Structured Query Language
XML	eXtensible Markup Language

Introdution Générale

Le potentiel énorme associé aux données médicales a conduit le secteur de santé à s'investir dans la construction des systèmes informatiques évolutifs, qui comprennent un ensemble d'outils et de mécanismes pour charger, extraire et traiter les données médicales tout en misant sur la massive puissance de traitement parallèle pour effectuer des transformations et des analyses complexes, pour optimiser les prises de décisions.

En raison du paradigme des données massives, la conception d'un système décisionnel évolutif fait face à une série de défis techniques. Tout d'abord, en raison de la variété et du volume des sources de données disparates, il est difficile de recueillir et d'intégrer des données à partir d'emplacements distribués. Actuellement plus de 175 millions de Tweets contenant texte, image, vidéo et des relations sociales sont générées chaque jour, par des millions de comptes répartis dans le monde. Deuxièmement, les systèmes de données volumineux doivent stocker et gérer l'ensemble des données massives et hétérogènes recueillies et assurer une certaine performance en termes de récupération rapide, évolutivité et protection de la vie privée. Par exemple, Facebook a besoin de stocker, accéder et analyser plus de 30 péta-bytes de données générées par ses utilisateurs. Troisièmement, l'analyse d'un volume doit exploiter les ensembles de données à différents niveaux en temps réel comme la modélisation, la visualisation, la prédiction et l'optimisation. Ainsi ces promesses inhérentes vont jouer un rôle très important dans la prise de décisions pour tirer du sens dans la prise de décision et acquérir d'autres avantages.

Ces défis technologiques exigent un réexamen des systèmes de gestion de données actuels, allant de l'architecture aux détails d'implémentation. Car la gestion traditionnelle de données et les systèmes d'analyse basés principalement sur les systèmes de gestion de bases de données relationnelles (SGBDR), sont insuffisants par rapport aux défis précédemment mentionnés.

Plus précisément, l'inadéquation entre les SGBDR traditionnels et le paradigme émergeant du volume massif de se résume ainsi :

- Du point de vue la structure des données, le SGBDR peut seulement gérer des données structurées, et n'offrent que peu de possibilités pour les données semi-structurées ou non structurées.

- Du point de vue évolutivité, le SGBDR évolue avec du matériel coûteux et ne peut pas évoluer avec du matériel de base en parallèle, ce qui est un frein face au volume de données qui augmente continuellement.

A travers ce modeste travail, nous avons tenté de répondre et relever ces défis dans le secteur de la santé. Ainsi, nous nous sommes orientés vers une solution qui répond aux exigences d'infrastructure qu'imposent les données médicales, tels que le ratio coût-efficacité, l'élasticité, et la capacité de scalabilité. Cette solution est le Framework open-source Hadoop, il intègre le stockage et le traitement des données, la gestion du système, et un outil d'entreposage de données.

Pour réaliser notre travail, nous avons structuré notre mémoire en quatre chapitres comme suit :

- **Le chapitre 1** : donne une vision globale sur les entrepôts de données, leurs architecture et leur modélisations, et expose les limites des entrepôts de données traditionnels.

- **Le chapitre 2** : présente une plate-forme performante et évolutive, qui est le Framework Hadoop, qui contient parmi la panoplie de ces outils, un outil d'entreposage de données qui répond aux nouveaux besoins des données volumineux et qui garantit une performance de temps de réponses des requêtes.

- **Le chapitre 3** : nous avons étudié les projets réalisés basé sur ce Framework dans quelques secteurs : réseaux sociaux, spatial, entreprise et secteur de santé. Et ce pour comprendre réellement l'implémentation de la plate-forme Hadoop et de son outil Hive.

- **Le chapitre 4** : présente la mise en œuvre d'une solution de modélisation de données qui comble les lacunes d'Hive et garantie une performance des requêtes, et détaille l'implémentation de notre entrepôt de données sous la plate-forme Hadoop avec l'outil Hive.

CHAPITRE 1

Concepts de base sur les entrepôts de données

1.1 Introduction

Les entrepôts de données sont apparus en 1996, réponse au besoin de rassembler toutes les informations d'une entreprise en une base de données unique destinée aux analystes et aux gestionnaires. Celà en intègrant des informations provenant de différentes sources de données internes mais aussi externes à l'environnement de l'organisme et en offrant la possibilité de faire des analyses et des corréllations sur des agrégations créées dynamiquement à partir de plusieurs dimentions .

Les bases de données des systèmes existants de type OLTP(Online Transaction Processing) ne sont pas appropriées comme support d'analyse, vu que leur conception ne vise pas les fonctions spécifiques réalisées dans l'entreprise.D'où la nécessité de la mise en place d'un système décisionnel qui fournit une vue globale des informations de l'entreprise et aussi un moyen stratégique de prise de décision.

Avant de passer à la phase de la conception et de la mise en oeuvre de ce système décisionnel qui est l'entrepôt de données, on va d'abord présenter à travers ce chapitre quelques concepts de base fondamentaux.

1.2 Historique

L'origine des ED revient à **1960**, ou l'entreprise General Mills et l'Université Dartmouth, dans un projet conjoint, créent les termes "faits" et "dimensions".Les dates marquantes de l'histoire des entrepôts de données sont [1] :

- En **1983**, Teradata introduit dans son SGBD un système purement décisionnel.

- En **1988**, Le terme DataWarehouse est utilisé pour la premiere fois dans l'article "An architecture for business and information systems " publier par Barry Devlin et Paul Murphy dans le journal système d'IBM.
- En **1990**, Red Brick Systems construit le système " Red Brick Warehouse " dédié à la construction d'entrepôt de données.
- En **1991**, Bill Inmon publie le livre " Building the Data warehouse ".
- En **1995**, La création de l'organisation " Data Warehousing Institute " pour soutenir et promouvoir la recherche dans le domaine des ED.
- En **1996**, Ralph Kimball publie le livre " The Data Warehouse Toolkit ".
- En **1997**, Réalisation de " Oracle 8 ", avec la prise en charge des requêtes des schémas en étoiles.

1.3 Définition

Bill Inmon définit l'entrepôt de données dans son ouvrage "Building Data warehouse " [2] de la façon suivante : L'entrepôt de données est une collection de données orientées sujet, intégrées, non volatiles et historiées, organisées pour support d'un processus d'aide à la décision".
Cette définition d'ED a été conceptualisée en termes de caractéristiques du référentiel des données, qui seront détaillées dans les points suivants :

- **Orientées sujet :**Les données de l'entrepôt sont organisées par thème (autour des sujets majeurs et des métiers de l'entreprise). L'intérêt dans cette organisation est de disposer d'un ensemble d'informations utiles sur un sujet transversal aux structures fonctionnelles et organisationnelles de l'entreprise [2].

- **Intégrées :** Les données dans l'entrepôt proviennent de différentes sources éventuellement hétérogènes. L'intégration est un processus qui consiste à résoudre les problèmes d'hétérogénéité, où les données contenues dans ED sont divisées en grandes subdivisions appelées domaine [3].

- **Non volatiles :** Les données stockées au sien de l'entrepôt sont permanentes et ne peuvent être modifiées, et le rafraîchissement de l'entrepôt de données, consiste seulement à ajouter de nouvelles données sans modifier ou perdre celle qui existent. Ceci pour conserver la traçabilité des informations et des décisions prises [3].

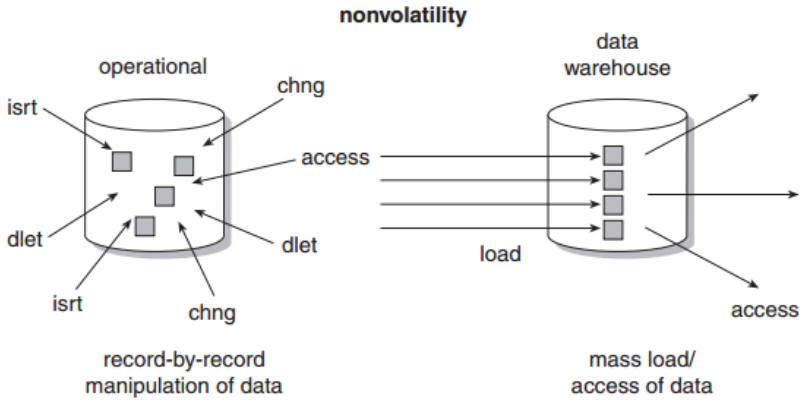

FIG. 1.1 – *Non volatilité des données de l'ED[4].*

- **Historisées :** Pour suivre dans le temps l'évolution des différentes valeurs des indicateurs à analyser, l'historisation est nécessaire. Un référentiel de temps est associé aux données, afin de permettre l'identification dans la durée de valeurs précise [2].

1.4 Objectifs d'un entrepôt de données

L'environnement d'entrepôt de données doit aligner différents ensembles de compétences, fonctionnalités et technologies. Par conséquent, il doit satisfaire les objectifs suivants [5] :

- Regrouper, organiser des informations provenant de sources diverses.
- Intégrer les informations récoltées et les stocker pour donner à l'utilisateur une vue orientée métier.
- Retrouver et analyser l'information selon plusieurs critères.
- Transformer un système d'information (SI) à vocation de production en un SI décisionnel.
- Séparer et combiner les données au moyen de toutes les mesures possibles de l'activité.
- Comporter un ensemble d'outils de requêtes, d'analyse et de présentation de l'information.

1.5 Différence entre Bases de données et Entrepôt de données

Le concept d'un entrepôt de données est apparu lors de l'existence de différence entre les systèmes transactionnels en ligne (OLTP) et les systèmes informationnels, dont certaines de ces différences fondamentales sont listées à travers le Tab1.1.Mais d'autres méthodes et techniques de conception et d'implémentation d'ED ont vu le jour, l'une de ces techniques est le modèle

dimensionnel de Kim Bail apparue en 1996 [6].

Foncionnalités	Base de données	Entrepôt de données
Caractéristiques	Basé sur le traitement optionnel	Basé sur le traitement d'information
Données	Actualisation de données stockées	Historisation de données stockées
Fonction	Les opérations quotidiennes	Les besoins d'information à long terme et aide à la décision
Utilisateur	Employés	Analystes, décideurs
Unité de travail	Court et simple transaction	Requêtes complexes
Orientation	L'orientation est sur la transaction	L'orientation est sur l'analyse
Vue	La vue des données est relationnelle plate	La vue des données est multidimentionnelle
Accès	Lecture et écriture	Lecture et rafraîchissement
Taille	Plusieurs gigaoctets	Plusieurs teraoctets
Priorité	Haute performance, haute disponibilité	Grande flexibilité, l'autonomie de l'utilisateur final
Métrique	Mesurer l'efficacité le débit , le débit transactionnel	Mesurer l'efficacité, le débit de la requête et le temps de réponse

TAB. 1.1 – *Comparaison entre les bases de données et les entrepôts de données [7].*

1.6 Source de données

L'entrepôt de données extrait les données de diverses sources souvent réparties et hétérogènes. Ces sources peuvent être des bases de données opérationnelles ou des sources de données externes à l'organisation. L'extraction de données se fait en utilisant un ETL (Extraction, Transformation and loading)[7].

1.7 Structure des données de l'entrepôt

Les données de l'entrepôt sont structurées en quatre classes. Ces dernières sont organisées selon un axe historique et un axe de synthèse [3] :

- **Les données agrégées :** Ce sont les données qui correspondent à des éléments d'analyse représentant les besoins des utilisateurs.Elle consistituent un résultat d'analyse et une synthèse de l'information contenue dans le système décisionnel, qui est facilement accessible et compréhensible.
- **Les données détaillées :** Reflétent les événements les plus récents. Les intégrations régulières des données issues des systèmes de production sont réalisées habituellement à ce niveau.
- **Les données historisées** Chaque nouvelle insertion de données provenant du système de production ne détruit pas les anciennes valeurs, mais crée une nouvelle occurrence des données.
- **Les métadonnées :** Elles constituent l'ensemble des données qui décrivent des règles ou processus attachés à d'autres données, qui reprèsente la finalité du système d'information. Ce point sera plus détaillé dans la section 1.8.2.

1.8 Architecture d'un entrepôt de données

La figure 1.2 représente la vision globale de l'architecture d'un entrepôt de données, qui se constitue de plusieurs niveaux : Source de données, Back-end tier, Data Warehouse tier, OLAP tier et Front-end tier [9]. Ces composantes seront détaillées ci-dessous.

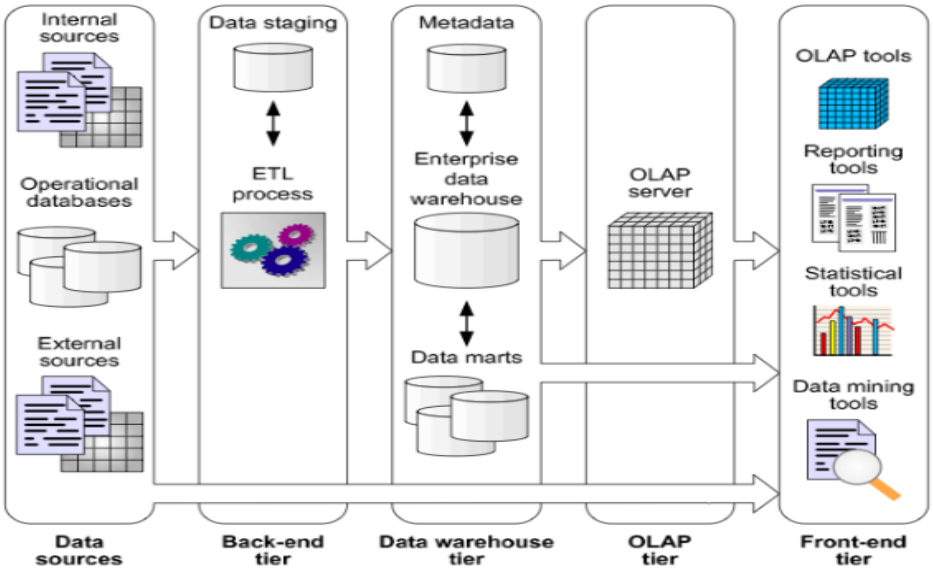

FIG. 1.2 – *Architecture d'un entrepôt de données [9].*

1.8.1 Back-end tier (niveau d'arrière-plan)

Représente la zone de construction qui contient l'ensemble d'outils et techniques utilisés lors du processus de préparation des données, avant leurs chargements au niveau de l'entrepôt, qui sont :

1.8.1.1 ETL process (Extraction, Transformation and Loading) :

utilisé pour alimenter les bases de données opérationnelles qui se situent au niveau du Data warehouse tier.Comme son nom l'indique, l'ETL est un processus à trois étapes :

- **L'extraction :** consiste à recueillir des données hétérogènes provenant de multiples sources, base de données opérationnelles, ou des fichiers de différents formats ; ils peuvent être des sources internes à l'organisation ou extérieures. Pour résoudre les problèmes d'interopérabilité, les données sont extraites chaque fois que possible en utilisant l'interface du programme d'application, telles que ODBC (Open DataBase Connection), OLEDB (Open Linking and Embedding for DataBase) et JDBC (Java DataBase Connectivity) [9].

- **La transformation :** une fois que les données sont extraites du système source, elles subissent une série de traitements destinée à les transformer en informations présentables [9].Cette procédure comporte plusieurs aspects [8] : le nettoyage, ce qui supprime les erreurs et les incohérences dans les données et les convertit en un format normalisé ; l'intégration, qui concilie les données provenant de différentes sources de données, au

niveau schéma et données ; et l'agrégation, qui résume les données obtenues à partir de sources de données en fonction du niveau de détail, ou la granularité, de l'entrepôt de données.

- **Le chargement :** alimente l'entrepôt de données avec les données transformées en respectant les contraintes du SGBD cible. Cela inclut également le rafraîchissement de l'entrepôt de données à savoir, la propagation des mises à jour dans l'entrepôt de données à partir des sources de données à une fréquence spécifiée afin de fournir des données à jour pour le processus de prise de décision [9].

  Le processus ETL nécessite généralement une zone de préparation de données (data staging), comme illustré dans la figure 1.2. Le data staging représente le chantier de l'ED. C'est là que les données sont chargées, nettoyées, combinées, archivées, puis rapidement exportées vers l'entrepôt. L'objectif de cette zone est l'obtention de données prêtes à être chargées sur un serveur de présentation (un moteur OLAP ou un SGBDR) [10].

1.8.2 Data warehouse tier (niveau entrepôt de données)

Se compose d'un entrepôt de données d'entreprise (Entreprise data warehouse) et/ou de plusieurs magasins de données (data marts), et un catalogue de métadonnées stockant des informations sur l'entrepôt de données et son contenu.
L'entrepôt de données d'entreprise est un entrepôt de données centralisé qui englobe tous les domaines fonctionnels ou départementaux dans une organisation [9]. D'autre part, un magasin de données est un entrepôt de données ciblées sur un sujet, alimenté depuis l'entrepôt de données de l'entreprise ou directement à partir de source de données. Conçu en cas de besoin spécifique de l'organisation [10].

1.8.2.1 Catalogue de métadonnées

On a décrété que les métadonnées désiganaient les données relatives aux données. Les métadonnées ont été traditionnellement classées en métadonnées techniques et business ; métadonnées business décrit la signification des données, et de l'organisation des règles, des politiques et des contraintes liées aux données ; métadonnées techniques décrit comment les données sont structurées et stockées dans un système informatique, ainsi que les applications et les processus qui manipulent les données [8].
Les termes bibliothèque d'information, référentiel, dictionnaire de métadonnées, et catalogue de métadonnées sont utilisés pour décrire l'ensemble des métadonnées (métadonnées business

et technique) présentées dans l'entrepôt. Ce catalogue représente le lieu de stockage unique des informations qui pilotent des processus dans l'ED [10].

1.8.3 OLAP tier

Se compose d'un serveur OLAP (OnLine Analytical Processing), qui prend en charge les données multidimensionnelles et les opérations. Il y a plusieurs types de serveurs OALP en fonction du modèle de mise en oeuvre sous-jacente : ROLAP, MOLAP et HOLAP.

1.8.4 Front-end tier

Est la partie publique de l'entrepôt de données. Il comporte les outils clients d'analyse et de visualisation des données (figure 1.2) [8]. Les outils typiques sont les suivants :

- **Les outils OLAP :** permettent l'exploration et la manipulation des données de l'entrepôt afin de trouver des modèles ou des tendances importantes pour l'organisation. Ils facilitent la formulation de requêtes complexes qui peuvent impliquer de grandes quantités de données. Ces requêtes sont appelées des requêtes ad-hoc.
- **Les outils de reporting :** permettent la production, la livraison et la gestion des rapports. Les rapports(figure 1.3) utilisent des requêtes prédéfinies, comme les requêtes demandant des informations dans un format spécifique qui sont effectuées sur une base régulière.
- **Les outils statistiques :** sont utilisés pour analyser les données du cube en utilisant des méthodes statistiques[8].
- **Les outils de fouille de données (data mining) :** permettent d'extraire des modèles d'une base de données historisée afin de décrire le comportement actuel et/ou de prédire le comportement futur d'un procédé. Les modèles peuvent être des modèles de calculs (des équations par exemple) ou des modèles logiques (des règles par exemple)[10].

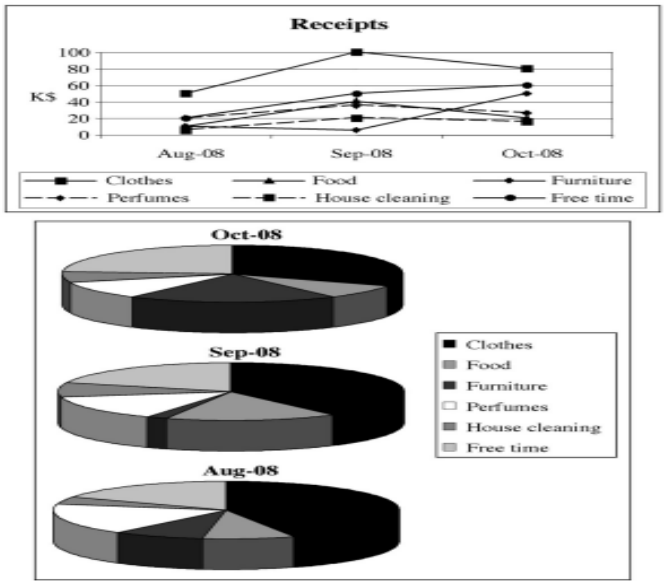

FIG. 1.3 – *Exemple de rapports : grafique, ligne et circulaire 3D [8].*

Certaines architectures d'entrepôts ne peuvents pas comporter toutes les composants illustrés dans la FIG 1.2. L'architecture de l'entrepôt peut contenir seulement un entrepôt d'entreprise sans le data mart ou, alternativement un data mart sans l'entrepôt de données d'entreprise. Dans d'autrs situations, il n'existe pas de serveur OLAP et les outils clients accèdent directement à l'entrepôt de données (FIG 1.2 : flèche reliant le Data warehouse tier au Front-end tier).
Une autre situation, un entrepôt de données virtuel, où il y a ni un entrepôt de données, ni un serveur OLAP. Il définit un ensemble de vue sur les bases de données opérationnelles qui sont matérialisées pour un accès efficace (FIG 1.2 : flèche reliant les sources de données au frond-end tier) [8].

1.9 Modélisation d'un entrepôt de données

Les données manipulées dans le contexte d'entrepôts de données sont représentées sous forme multidimensionnelle qui permet une meilleure compréhension des données à des fins d'analyse et fournit de meilleures performances pour les requêtes analytiques complexes. Cette modélisation consiste à considérer un sujet analysé comme un point dans un espace à plusieurs dimensions. Les données sont organisées de manière à mettre en évidence le sujet analysé et les différentes perspectives de l'analyse. Le modèle multidimensionnel repose sur le concept de CUBE (ou hypercube)pour représenter les données (figure 1.4).Un cube organise les données en une ou plusieurs dimensions qui déterminent une mesure d'intérêt [10]. Deux concepts fondamentaux caractérisent le modèle multidimensionnel :

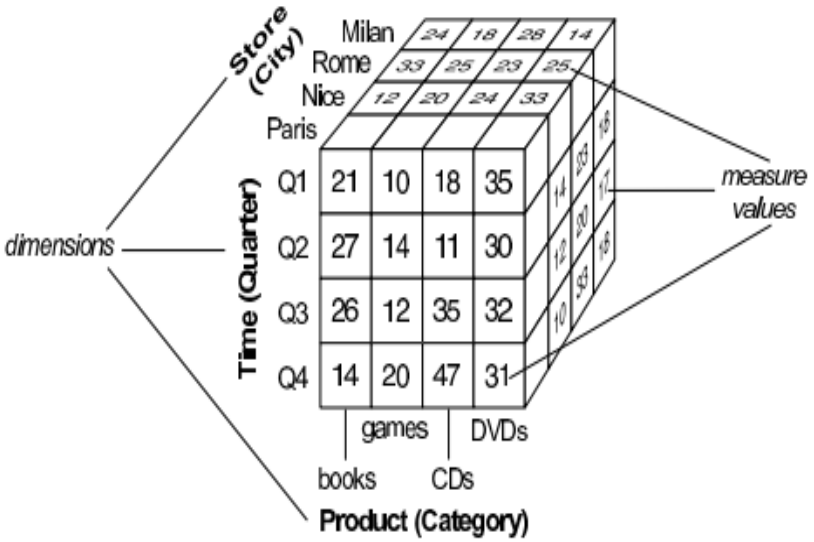

FIG. 1.4 – *Cube de données à trois dimensions : magasin, temps et produit [9].*

1.9.1 Fait

Un fait représente un sujet d'analyse. Il est constitué de plusieurs mesures relatives au sujet traité. Les mesures d'un fait sont numériques et généralement valorisées de façon continue. Elles permettent une évaluation quantitative des différents aspects du problème et effectuent une analyse à portée de main. Le cube de données de la figure 1.4 est utilisé pour analyser les chiffres de ventes.Les chiffres indiqués dans le cube de données représentent une quantité de mesure, indiquant le montant total des ventes spécifié. Un cube de données contient généralement plusieurs mesures [9].

1.9.2 Dimensions

Les dimensions sont les différentes perspectives qui sont utilisées pour analyser les données.Le cube de données de la figure 1.4 est à trois dimensions : magasin, temps, et produit . Une dimension se compose d'attributs (paramètres) qui servent à enregistrer les descriptions textuelles.L'organisation des dimensions s'effectue de façon hiérarchique selon le niveau de granularité ou de détail des attributs [9]. L'implémentation du modèle multidimensionnel sur un SGBD réel peut se faire selon deux modèles : ROLAP (Relational OnLine Analytical Processing) et MOLAP (Multidimensional OnLine Analytical Processing). Ces deux implémentations sont décrites dans la section suivante.

1.9.3 Systèmes ROLAP

Dans les systèmes de relation OLAP, les données multidimensionnelles sont mises en œuvre sous forme de tables relationnelles, tables de faits et de dimensions. La table de fait est constituée d'attributs représentant les mesures d'activité et les attributs clés étrangères de chacune des tables de dimension. Les tables de dimension contiennent les paramètres et une clé primaire permettant de réaliser des jointures avec la table de fait[9]. Ces tables sont organisées dans des structures appelées : schéma en étoile, schéma en flocon, schéma en constellation.Décrit comme suivant :

1.9.3.1 Schéma en étoile :

est un schéma relationnel dans lequel une table centrale contenant les faits anaylisés référence des tables de dimensions, chaque dimension étant décrite par une seule table dont les attributs représentent les diverses granularité [12].

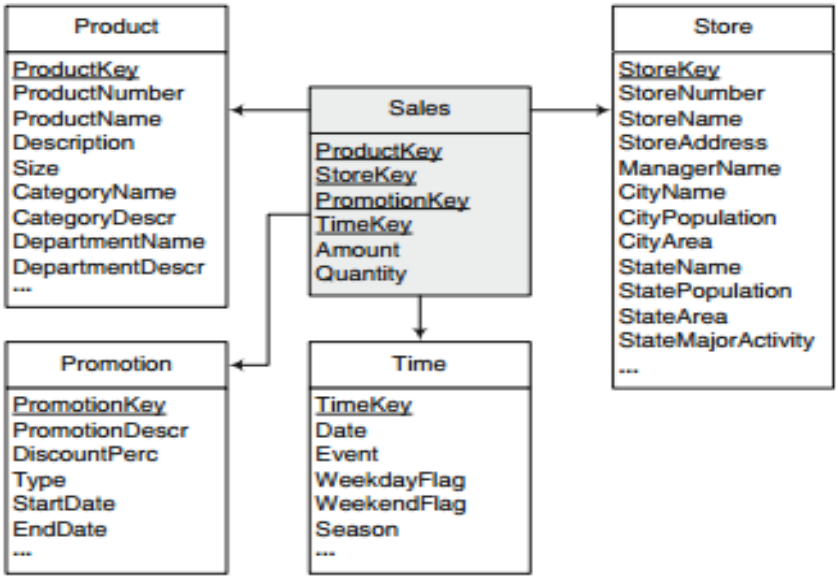

FIG. 1.5 – *Exemple de schéma en étoile [13].*

1.9.3.2 Schéma en flocon :

est un schéma relationnel dans lequel une table centrale contenant les faits analysés référence des dimensions du premier niveau, chaque dimension étant décrite par une succession de tables représentant les diverses granularités, chaînées par contraintes référentielles [12].

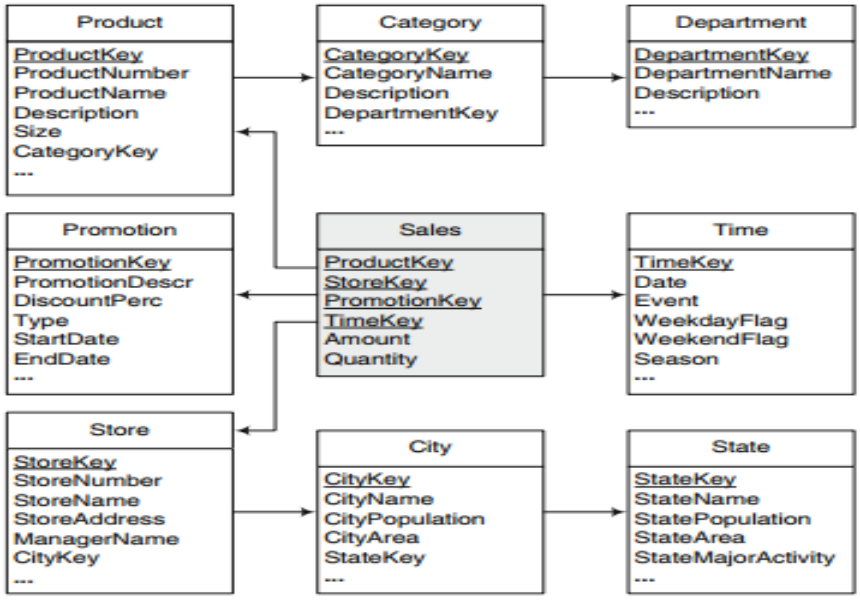

FIG. 1.6 – *Exemple de schéma en flocon [13].*

1.9.3.3 Schéma en constellation :

est un schéma de constellation à plusieurs tables de faits qui partagent des tables de dimensions. L'exemple donné dans la figure 1.6 à deux tables de faits ventes et achats partageant le temps et la dimension du produit [11].

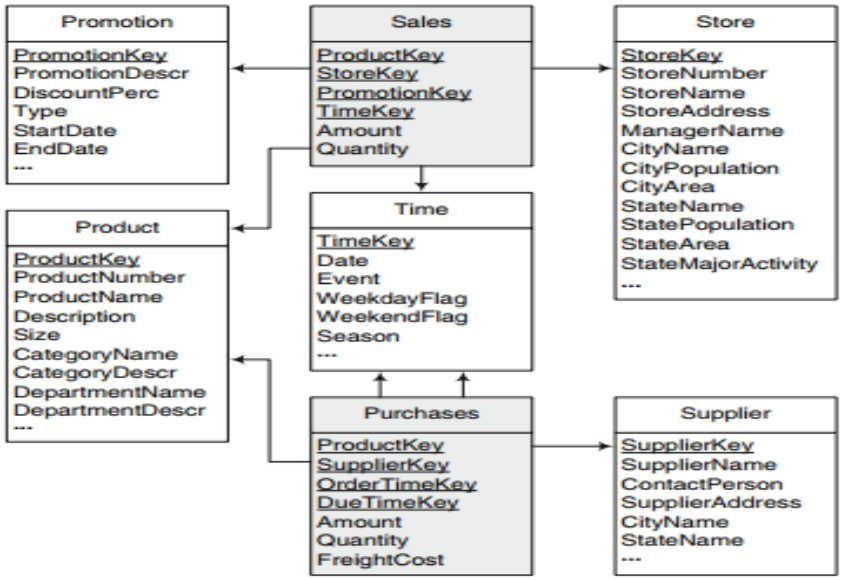

FIG. 1.7 – *Exemple de schéma en constellation [13].*

1.9.4 Systèmes MOLAP

Dans les systèmes multidimensionnels OLAP, l'implémentation classique des tables relationnelles est ignorée et propose plutôt des implémentations adaptées aux tableaux sous forme multidimensionnelle à n dimensions. Le cube de données est naturellement implémenté comme une matrice à trois dimensions du type : CUBE [1 : M, 1 : N, 1 : P] (Mesure), et la matrice est gardée en mémoire autant que possible et paginée seulement sur disque lorsqu'il n'y a plus de place en mémoire. Des techniques de compressions sont utilisées pour réduire la taille mémoire occupée.

De nombreux systèmes proposent une couche multidimensionnelle au-dessus d'un SGBD relationnel. On parle alors de MROLAP. Les données persistantes sont gérées dans les tables et montées en mémoire dans des matrices multidimensionnelles représentant les cubes [11].

1.10 Les limites des entrepôts de données traditionels

Les systèmes d'information décisionnels s'appuient sur l'entreposage de données et l'analyse en ligne qui sont deux piliers de l'informatique décisionnelle, aujourd'hui confrontés à de nouveaux défis scientifiques. De nouvelles sources d'informations hétérogènes, fortement évolutives, changeantes, dépendantes ou autonomes et distribuées apparaissent. Avec cette évolution, les entrepôts de données traditionnels sont confrontés à des limites qui sont au niveau :

- **Nature des données :** la nécessité d'intégrer le Web à l'entreprise pour compléter les données de celle-ci avec celle du monde extérieur, a engendré le besoin de comprendre de nouvelles sources de données de nature semi-structurées, comme les fichiers HTML ou XML. L'intégration de ces données, revendique la prise en compte de modèles de données riches comme l'objet, le développement d'outils interactifs d'intégration de schémas hétérogènes, la prise en compte de requête complexes, et l'optimisation de requêtes distribuées [10].

- **Disponibilité :** représente un défi pour l'entrepôt, à la fois en raison du processus de chargement et de l'infrastructure dans son ensemble ; les projections d'un ensemble croissant de données, cycles de conservation des données et le temps de réponses des requêtes associées. La non-disponibilité des données en temps opportun influence sur l'utilisation et l'adoption de la prise de décision au sein de l'entreprise.

- **Stockage des données :** le fait que les données, les index et les tables temporaires travaillent tous avec le même ensemble de disques et les contrôleurs au niveau de l'entrepôt de données, cela crée une énorme pénalité à la fois sur la disponibilité et la performance du système, et avec l'ajout de charge de travail des requêtes complexées sur l'architecture de stockage, à qui engendre un temps d'accès étendus des disques et des données, ce qui provoque un manque de performance.

- **Performance des requêtes :** C'est un autre domaine de performance qui remet en question l'entrepôt de données traditionnel. Les requêtes ad-hoc et des requêtes analytiques causent le plus d'impact sur la performance globale des requêtes, l'accès et le traitement des données, y compris le déplacer à travers le réseau en raison de leur nature non déterministe. Les requêtes peuvent nécessiter un large ensemble de données qui a besoin d'accéder à plusieurs zones de stockage ou d'un grand ensemble de données qui peut être consulté dans une zone de stockage plus petite.

- **Transport de données :** Problèmes au niveau de la surcharge lors du transport des données à partir d'une couche à l'autre et de la disponibilité des données ce qui ralentit le processus de traitement ultérieur [14].

1.11 Conclusion

De nos jours, le volume des données d'analyses atteint des tailles critiques, défiant les approches classiques d'entreposage de données, qui reposent principalement sur des bases de données relationnelles (implémentations ROLAP). Ces approches classiques sont remises en cause devant l'importance des volumes des données. Avec l'apparition des grandes plateformes Web, telles que Google, Facebook, Twitter, Amazon, des solutions pour gérer les mégadonnées ont été développées tel que la plate-forme Hadoop. Ce dernier est basé sur une approche décentralisé et a largement contribué à l'apparition de solutions de gestion de données. Hive est l'un des outils de l'écosystème d'Hadoop qui permet d'implémenter une nouvelle approche entreposage de données, que nous allons présenter dans le chapitre suivant.

CHAPITRE 2

Hadoop et son outil d'entreposage de données Hive

2.1 Introduction

Les nouvelles solutions de gestion de données doivent répondre à une triple problématique : un volume de données important à traiter, une grande variété d'informations (structurées ou non structurées), et un certain niveau de vélocité à atteindre. Pour répondre à ces besoins, il s'avère que la plate-forme Hadoop serait la solution open source par excellence.

Apache Hadoop (High-availability distributed object-oriented platform) est un système distribué qui répond à ces problématiques. Il propose un système de fichiers distribués HDFS (Hadoop Distributed File System) pour assurer le stockage et l'intégrité des données en dupliquant plusieurs copies d'un même bloc à travers des dizaines, voire des milliers de machines différentes, il fournit un système d'analyse de données appelé MapReduce pour réaliser des traitements sur des gros volumes de données grâce à sa répartition efficace du travail sur différents nœuds de calcul, et il offre une nouvelle approche d'entrepôt de données avec Hive.

Dans ce chapitre on va d'abord présenter une vue globale sur Hadoop et ensuite nous exposerons l'outil d'entreposage Hive.

2.2 Vue globale sur Hadoop

2.2.1 Historique d'Hadoop

Hadoop trouve son origine dans le projet du moteur de recherche libre Apache Lucene lancé en 2002. Les développeurs du projet Lucene rencontrent pendant le développement du projet des problèmes de volumétrie de données, que Doug Cutting résoud en adaptant une version open source des outils propriétaires développées par Google, qui deviendra le projet Hadoop.

Les dates marquantes de l'histoire du projet Hadoop sont [15] :

- En **2003**, Google publie un article qui décrit son système de gestion de fichiers distribués Google File System (GFS), ce dernier est utilisé dans le stockage des fichiers généré dans le cadre d'exploitation web et des processus d'indexation.
- En **2004**, Apache implémente une version open source du système de fichier GFS, qui est nommé Nutch distributed File system (NDFS).
- En **2004**, Google publie un article qui présente son système d'analyse des données MapReduce.
- En **2005**, Apache intègre MapReduce avec NDFS dans le projet Nutch.
- En **2006**, Doug Cutting s'inspire du doudou de son fils de cinq ans, Hadoop un éléphant jaune, pour le logo ainsi que pour le nom de ce nouveau framework Java, qui est devenu un projet indépendant d'Apache.
- En **2008**, Doug Cutting intègre Yahoo et exploite Hadoop pour améliorer l'indexation de leur moteur de recherche.

2.2.2 Présentation d'Hadoop

Hadoop est un système logiciel distribué capable de stocker et traiter de manière séquentielle, et à des coûts raisonnables, des volumes de données de plusieurs péta-octets. Il offre une grande flexibilité (possibilité d'enlever et d'ajouter des machines) et ses performances évoluent de manière quasi linéaire en fonction du nombre de machines constituant le cluster. Hadoop a été écrit en Java, qui demeure le langage de prédilection pour les développeurs de programme Hadoop [16].

FIG. 2.1 – *Le cluster Hadoop chez Yahoo [16].*

La Figure 2.1 ci-dessus, illustre l'ensemble de machines constituent le cluster Hadoop de Yahoo.

2.2.3 Objectifs d'Hadoop

Hadoop a été conçu afin de satisfaire les objectifs suivant [16] :

- **Stocker et traiter des volumes de données très importants dans des délais acceptables :** Chaque nœud du cluster offre une capacité de stockage et une puissance de calcul pouvant traiter une grande quantité de données. HDFS travaille sur des blocs de 64 Mo et de 128 Mo, ce qui réduit le temps de recherche au niveau des disques durs et par conséquent minimise le taux de transfert.

- **Tolérance aux pannes :** Hadoop a été conçu pour pouvoir fonctionner avec du matériel de milieu de gamme, dont la durée de vie ne dépasse pas cinq ans. Le risque d'une défaillance est donc très probable. Pour cela HDFS a intégré une fonction de réplication des données au niveau du cluster Hadoop. Les données sont stockées en trois exemplaires sur des nœuds différents, ce qui garantit la disponibilité des données en cas de pannes et l'efficacité d'exécution des tâches.

- **Optimisation des performances :** Hadoop est conçu selon l'architecture " share nothing ", ou les données de services sont réduites au strict minimum, ce qui permet à Hadoop d'économiser de la bande passante et du temps machine. Ce dispositif est géré directement par Hadoop d'où l'existence d'une relation linéaire entre le nombre de nœuds dans un cluster Hadoop et ces performances.

2.2.4 Les principales composantes d'Hadoop

Hadoop se constitue essentiellement de deux composantes, le système de gestion de fichiers distribué HDF et de MapReduce, qui sont détaillées dans les deux sections suivantes :

2.2.4.1 Hadoop Distributed File System (HDFS)

Il s'agit d'un système de gestion de fichiers distribué, inspiré par GFS le système de gestion de fichiers de Google. HDFS est le composant d'Hadoop en charge du stockage de données dans le cluster [17]. Il se diffère du reste des systèmes de gestion de fichier traditionnel par les principales caractéristiques suivantes [16] :

- **Portabilité :** le système de fichier HDFS peut être déployé sur différents systèmes d'exploitation. Mais le seul inconvénient se situe dans l'obligation de solliciter une application externe pour monter une unité de disque HDFS.

- **Distributivité :** HDFS est un système distribué où 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 noeuds au cluster.

- **Manipulation de blocs de grande de taille :** HDFS est optimisé pour manipuler des blocs de taille importante, couramment 64 Mo ou 128 Mo, ce qui permet de maximiser les taux de transfert des données, en limitant les temps de recherche au niveau des disques durs.

- **Mécanisme de réplication des blocs :** chaque bloc est sauvegardé en trois exemplaires dans des nœuds distincts au niveau du cluster.

Les composantes de HDFS

HDFS se compose d'un ensemble de nœuds :

- **NameNode :** est un nœud maître qui s'occupe de gérer l'état de HDFS. Le NameNode est responsable de la répartition des données sur les DataNode et de la gestion de l'espace mémoire du cluster. Il héberge les métadonnées qui représentent l'ensemble d'information sur l'emplacement des blocs de données, leurs propriétés et les autorisations.

- **SecondaryNameNode :** est aussi un nœud maître. Le SecondaryNameNode est chargé de la maintenance des nœuds du cluster, ou il garde sous contrôle l'espace disque utilisé, limite la charge processeur du NameNode et il permet la continuité de fonctionnement du cluster Hadoop en cas de panne [18].

- **DataNode :** est le nœud esclave implanté sur chaque machine du cluster qui n'est pas un nœud maître. Le DataNode héberge une partie des données du cluster Hadoop afin d'assurer une haute disponibilité des données [15].

2.2.4.2 MapReduce

MapReduce est un modèle de programmation conçu pour la lecture et le traitement d'un volume de données très important. Il implémente au sein d'Hadoop diverses fonctionnalités : la division des tâches en parallèles, leurs répartitions et la collecte des résultats [16].Un programme

MapReduce se constitue de fontion map et reduce. La fonction map fait la lecture des données stockées sur disque et les traite. La fonction reduce consolide les résultats issus de la fonction map puis les écrit sur le disque. Les données MapReduce sont lues ou écrites selon le format <clé, valeur>. L'exécution d'un programme MapReduce sur un ensemble de données désigne le job Hadoop [15].

Les composantes de MapReduce

Une tâche, au sens Hadoop du terme, est une tâche map ou une tâche reduce. Chaque tâche traite une partie seulement de l'ensemble des données du cluster. L'exécution de ces tâches est contrôlée par un ensemble de nœuds maîtres et esclaves qui sont :

- **JobTracker :** est le nœud maître responsable du lancement des tâches d'un job Hadoop et de leurs coordinations au niveau des noeuds esclaves. Il est aussi chargé de planifier l'exécution des jobs Hadoop, l'agrégation de leurs résultats et de la gestion de l'état du TaskTracker [16].
- **TaskTracker :** est le nœud esclave en charge de l'exécution des tâches map et reduce qui lui sont assignées par le JobTracker[15].

Des rapports journaliers regroupant l'ensemble d'informations sur l'exécution des tâches MapReduce est tenu par le JobTracker, ce qui garantit une facilité de réinitialisation des tâches en cas de défaillance d'un nœud TaskTracker.

L'ensemble des nœuds maîtres et esclaves des deux composantes HDFS et MapReduce constituent les composantes du cluster Hadoop [19], lesquelles sont représentées dans la FIG 2.2 ci-dessous :

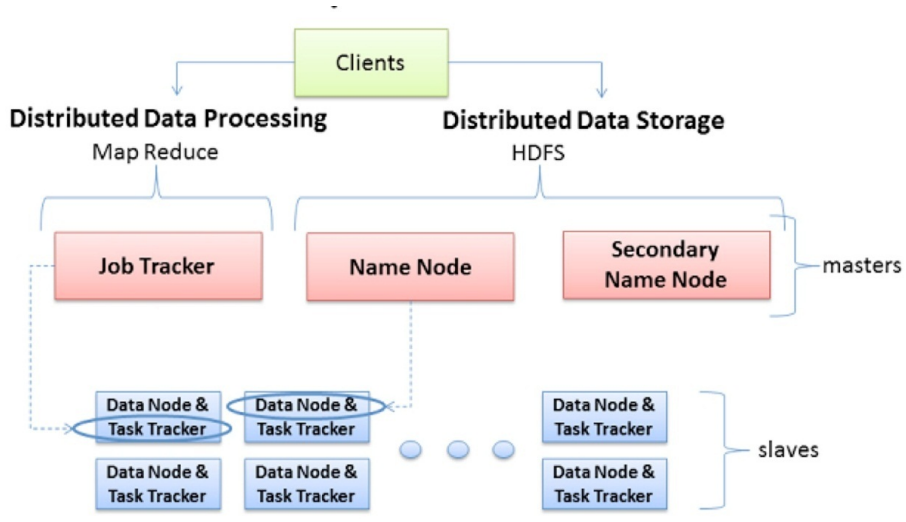

FIG. 2.2 – *Les nœuds composant un cluster Hadoop [19].*

2.2.5 Écosystème d'Hadoop

L'écosystème de Hadoop comprend de nombreux outils en outre HDFS et MapReduce, tel représenté dans la FIG 2.3. Les outils d'Hadoop peuvent être classés en trois grandes catégories :

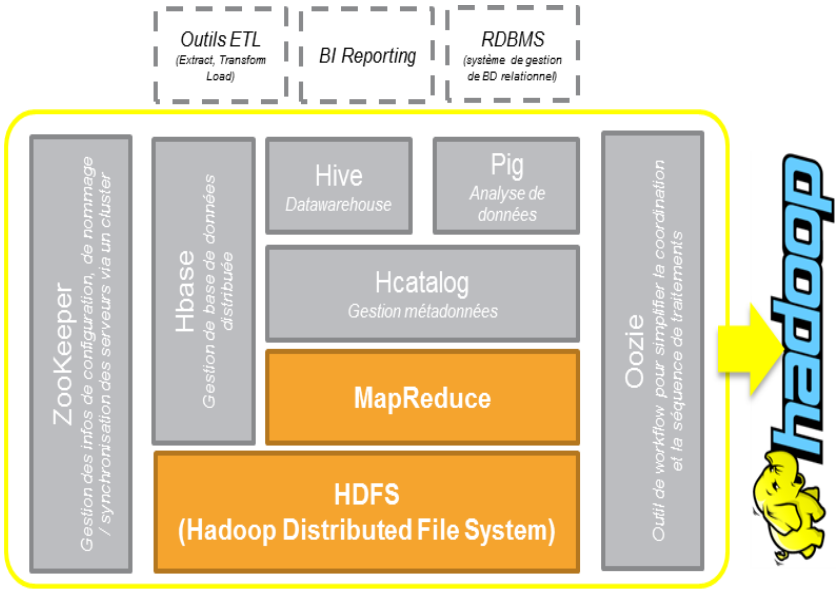

FIG. 2.3 – *Ecosystème d'Hadoop [19].*

2.2.5.1 Les outils orientés bases de données

Ils donnent aux utilisateurs d'Hadoop la possibilité d'interagir avec une base de données (Sqoop) ou les fonctionnalités d'une base de données (HBase).

- **HBase :** est une base de données non-relationnelle distribuée, open source, inspiré à partir de l'outil Bigtable de Google [20]. Le stockage de données se fait par colonne, qui est une approche totalement différente des bases de données traditionnelles [21].

- **Sqoop :** est un acronyme bâti à partir de l'expression SQl-to-Hadoop. Initialement, Sqoop était développé par Cloudera. Il s'appuie sur MapReduce et des drivers JDBC pour la manipulation de données [16]. Il est conçu pour l'exportation et l'importation des données du cluster Hadoop vers les bases de données relationnelles [22].

2.2.5.2 Les outils de Requêtage et scripting des données Hadoop

Cette catégorie définit les outils permettant de programmer les jobs Hadoop sans utiliser le langage Java.

- **Hive :** est une infrastructure d'entrepôt de données distribuées construit sur Hadoop conçu par Facebook dans le but de faciliter l'utilisation des Jobs Hadoop [23].

- **Pig :** est une plate-forme d'analyse de grands ensembles de données par le biais du langage de scripting Pig Latin. Pig est développé initialement par Yahoo et devenu par la suite un projet de la fondation logicielle Apache. Un programme Pig travaille sur des données élémentaires appelé " atom ". Un ensemble d'atoms désigne un tuple, par exemple une ligne dans un fichier. Les Tuples sont regroupés dans des bags. Et ils consistent souvent à charger des ensembles de données dans des bags, puis à créer de nouveaux bags via des opérations de tri, de filtrage, de jointure, etc[15].

- **API de streaming :** est un outil destiné au développeur disposant de compétence outre que dans le langage Java. Cet outil permet d'écrire un mapper ou reducer avec n'importe quel langage de scripting [16].

2.2.5.3 Les outils d'exploitation

Ils permettent l'automatisation de l'exécution des jobs Hadoop.

- **Zookeeper :** est un outil de coordination pour les environnements distribués, dont Hadoop. Il offre aux clusters comportant un grand nombre de nœuds plusieurs fonctionnalités telles que la gestion des groupes de nœuds, la configuration des tâches map et reduce et leur synchronisation.
  Le Zookeper fonctionne généralement dans son propre cluster, ce qui permet d'améliorer ses performances et inclut des fonctions de réplication et peut, de ce fait, rester disponible tant que la majorité de ses nœuds restent opérationnels [15].

- **Oozie :** est un outil pour l'exécution et la planification des jobs MapReduce [16].Oozie peut gérer tout type de job Hadoop, ce qui inclut les jobs Hive, Pig et Sqoop. Il est aussi capable de prendre en compte les dépendances temporelles, exécuter un job plusieurs fois de suite et informer le responsable système des problèmes survenus [15].

2.2.6 Domaines d'application

Les domaines d'application d'Hadoop englobe tous les secteurs d'activités, voici quelques exemples :

2.2.6.1 Industrie de la Télécommunication

Les grandes entreprises de télécommunications ont un certain nombre de marchés verticaux tels que le marketing, produit, ventes, ressources humaines, technologies de l'information, recherche et développement etc, qui sont dans le besoin constant d'informations. Et à travers Hadoop ces entreprises de télécommunications peuvent :

- Analyser leurs enregistrements de données avec Hadoop : pour pouvoir améliorer continuellement la qualité des appels, la satisfaction du client et les marges d'entretien.
- Réduire le taux de désabonnement des clients : en implémentant Hadoop à la fin de l'année 2013. SFR est devenu le premier opérateur mobile français à s'engager dans l'expérience Hadoop pour réduire le nombre de désabonnement de ses clients.
- Recommander suivant le produit à acheter : avec Hadoop, l'entreprise de télécommunication identifie les besoins du client et lui recommande un produit adapté à ses attentes [24].

2.2.6.2 Industrie publicitaire

La chaine de supermarchés Target (USA) a mis au point un système d'analyse des achats de ses clients, basé sur Hadoop. Afin de pouvoir envoyer à chacun de ses clients de la publicité ciblée.

2.2.6.3 Domaine de la santé

La société "Treato" a mis au point un système basé sur Hadoop et Hbase qui est capable, en analysant les échanges sur des sites sociaux, de détecter une interaction médicamenteuse. Le système permet aussi d'identifier en temps quasi réel les préoccupations des patients, leur comportement, etc.

2.2.6.4 Dans le cadre des moteurs de recherche

Orange a amélioré les résultats de son moteur de recherche en fournissant aux utilisateurs des résultats plus adéquats à leurs requêtes et cela en mettant en place un système d'indexation des pages web et un système de type PageRank, basé sur la plate-forme d'Hadoop.

2.2.6.5 Industrie des villes intelligentes

Le réseau électrique intelligent utilise l'informatique comme moyen d'optimisation de la productivité et la consommation de l'énergie. En France, EDF prévoit d'installer des compteurs électriques communicants qui permettent de récolter des informations sur la consommation en énergie des clients, ces compteurs vont générer des volumes de données considérables. Dans ces conditions, EDF utilisra Hadoop comme solutions [16].

2.2.6.6 Domaine des réseaux sociaux

L'ensemble de l'infrastructure de traitement de données dans Facebook avant 2008 a été construit autour d'un entrepôt de données utilisant un SGBDR commercial. Le volume de données de Facebook ne cessait de s'accroitre de 15TB fixé en 2007 à un ensemble de données de 700TB en 2009. L'infrastructure employée était insuffisante.

Ils avaient eu un besoin d'une plate-forme de donnée évolutif. Par conséquent ils se sont orientés vers Hadoop comme plate-forme de leur entrepôt de données Hive[25].

2.3 Présentation de l'outil Hive

Hive est un projet open source dédié à la construction d'entrepôt de données sous la plate-forme Hadoop. Hive utilise un langage de requête semblable à SQL appelé HiveQL, qui prend en charge certaines primitives de manipulation de données telles que la projection, jointure, le groupement, l'union et les sous-requêtes dans la clause FROM [25]. Le plan d'exécution d'une requête HiveQL se traduit par la création et l'exécution d'un ensemble de tâches map et de tâches reduce[26].

2.3.1 Les principales caractéristiques de Hive

Hive se distingue d'une base de données traditionnelle par les caractéristiques suivantes[16] :

- Il supporte les bases de données traditionnelles en soutenant le SQL, mais ne constitue pas une base de données complète.

- Dans un SGBDR, le schéma d'une table est appliqué au moment de la charge, si les données en cours du chargement ne sont pas conformes au schéma, elles sont rejetées, cette conception est appelée schéma en écriture. Mais Hive ne vérifie pas les données lors du chargement mais lors de la récupération, cette conception est appelée schéma en lecture.L'opération de chargement des données dans Hive est juste une copie de fichier

ou un déplacement.

- Il est basé sur la notation de Write Once Read Many. Il est donc possible d'écrire qu'une seule fois (Write Once) et de lire autant de fois souhaité (Read Many)

- Il n'autorise pas les opérations : insertions, suppressions et modification, car Hive est construit pour fonctionner sur des données HDFS utilisant MapReduce, où l'analyse et la mise à jour des données s'obtenaient en transformant les données en une nouvelle table.

- Il ne prend pas en charge les traitements de transaction en ligne OLTP (Online Transaction Processing) car Hadoop est un système orienté par lots.

2.3.2 Langage HiveQL (Hive Querry Language)

Le langage HiveQL supporte la définition de données (DDL) dans la création des tables avec des formats de sérialisation spécifiques, de partitionnement et de buckting. Et ne supporte guère la mise à jour et la suppression de lignes des tables existantes.
Il soutient l'insertion multi-table, où les utilisateurs peuvent exécuter plusieurs requêtes sur les mêmes données d'entrée en utilisant une seule déclaration Hive, ce qui optimise l'exécution des requêtes. HiveQL est un langage extensible car il permet aux utilisateurs de créer leur propre fonction avec le langage Java, également de définir les fonctions d'agrégation et les fonctions de table génératrices [27].

2.3.2.1 Types de données

Les types de données de Hive sont de deux types soit primitifs ou complexes. Les types primitifs illustrés dans FIG 2.4 correspondent aux types de données du langage Java, Mise à part le type STRING qui est l'équivalent du type VARCHAR du langage SQL. Les types Complexes (illustrés au FIG 2.5)sont de type : STRUCT qui est un enregistrement encapsulant un ensemble de champs, ainsi que des deux types ARRAY et MAP semblables à leurs homonymes en langage Java [15].

Category	Type	Description	Literal examples
Primitive	TINYINT	1-byte (8-bit) signed integer, from -128 to 127	1
	SMALLINT	2-byte (16-bit) signed integer, from -32,768 to 32,767	1
	INT	4-byte (32-bit) signed integer, from -2,147,483,648 to 2,147,483,647	1
	BIGINT	8-byte (64-bit) signed integer, from -9,223,372,036,854,775,808 to 9,223,372,036,854,775,807	1
	FLOAT	4-byte (32-bit) single-precision floating-point number	1.0
	DOUBLE	8-byte (64-bit) double-precision floating-point number	1.0
	BOOLEAN	true/false value	TRUE
	STRING	Character string	'a', "a"
	BINARY	Byte array	Not supported
	TIMESTAMP	Timestamp with nanosecond precision	1325502245000, '2012-01-02 03:04:05.123456789'

FIG. 2.4 – *Les types de données primitive de HiveQL [15].*

Category	Type	Description	Literal examples
Complex	ARRAY	An ordered collection of fields. The fields must all be of the same type.	array(1, 2)[a]
	MAP	An unordered collection of key-value pairs. Keys must be primitives; values may be any type. For a particular map, the keys must be the same type, and the values must be the same type.	map('a', 1, 'b', 2)
	STRUCT	A collection of named fields. The fields may be of different types.	struct('a', 1, 1.0)[b]

FIG. 2.5 – *Les types de données complexes de HiveQL [15].*

2.3.3 Architecture de Hive

La FIG 2.6 illustre les principales composantes de Hive et leurs interactions avec Hadoop. Ces composantes seront détaillées dans les points suivants [27] :

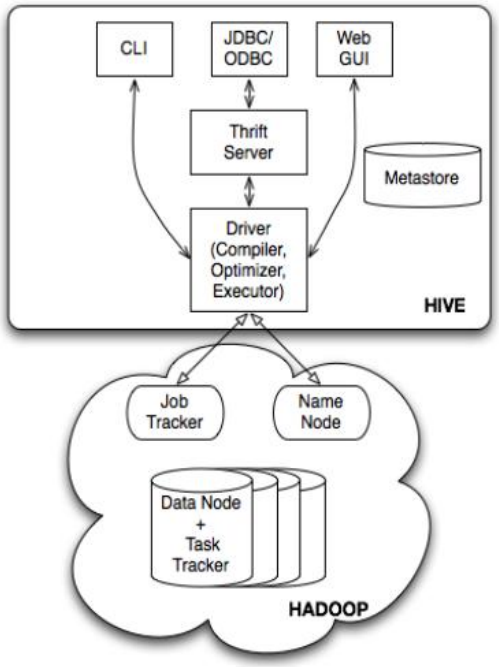

FIG. 2.6 – *Architecture de Hive [25].*

2.3.3.1 Thrift Server

Expose une API permettant l'exécution des commandes HQL. Les clients se connectent au serveur Hive en utilisant le JDBC driver, ODBC driver ou Thrift client, expliqué ci-dessous dans la section Hive clients.

2.3.3.2 Driver

Invoque le compilateur de Hive lors de la réception d'une requête HiveQL. Le compilateur traduit ensuite cette requête dans un plan qui consiste en un ensemble de tâches map et reduce.

2.3.3.3 Hive services

Désigne les nombreux services qu'un utilisateur peut exécuter sous Hive. Avec la commande "–service", un utilisateur peut spécifier le service à exécuter. Parmi ces services disponibles :

- **CLI :** l'interface de ligne de commande de Hive (le shell). C'est le service par défaut.

- **Hive server :** son exécution permet aux applications utilisant le Thrift, JDBC, et les connecteurs ODBC de communiquer avec Hive.
- **HWI :** est l'interface Web de Hive qui est une alternative au shell de Hive.
- **Jar :** représente un moyen pratique pour exécuter les applications Java qui inclut les classes Hadoop et Hive à la fois sur le classpath.

2.3.3.4 Hive clients

L'accès aux services de Hive se fait via les différents types des clients Hive cités ci-dessous. La FIG 2.7 illustre la relation entre les clients et les services de Hive.

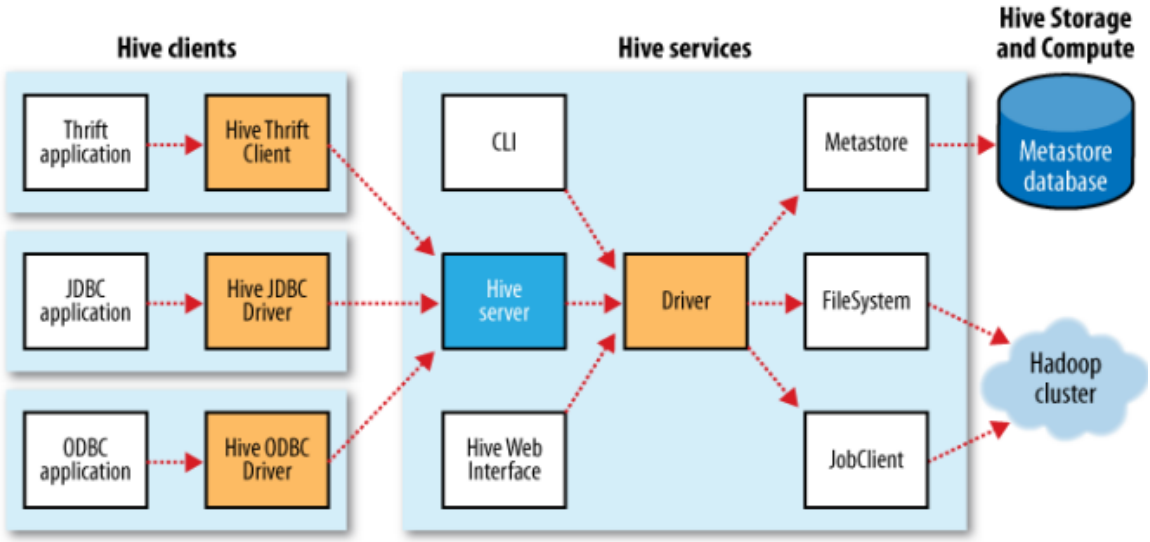

FIG. 2.7 – *Les différents types des clients de Hive[27].*

- **Hive thrift client :** Facilite l'exécution des commandes Hive à partir d'un large ensemble de langage de programmation. Le client thrift est écrit en plusieurs langages : C++, Java, PHP, Python et Ruby.

- **JDBC driver :** Hive fournit un pilote JDBC (Java DataBase Connectivity) de type 4, défini dans la classe org.apache.hadoop.hive.Jdbc.HiveDriver. , qui permet à une application Java de se connecter à un serveur Hive en cours d'exécution.

- **ODBC driver :** Le pilote ODBC de Hive permet aux applications qui prennent en charge le protocole ODBC de se connecter à Hive. Le pilote ODBC ou JDBC utilise tous les deux le Client thrift pour communiquer avec le serveur Hive.

2.3.3.5 Metastore

Contient la description des bases de données, des tables, des partitions et des colonnes .Le Metastore est le référentiel central des métadonnées de Hive. Ces derniers sont accessibles via l'API du service Metastore. Par défaut, le Metastore utilise une base de données Debry intégré dans le service Hive. Ceci est appelé la configuration en mode embarqué (Embedded metastore) tel illustré dans la FIG 2.8, ci-dessous. Le mode embarqué est seulement adapté pour les tests et le développement local, dans le cas contraire, un serveur MySQL peut être utilisé.

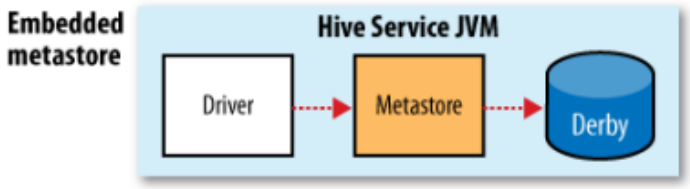

FIG. 2.8 – *Configuration de metastore de Hive en mode embarqué[27].*

2.3.4 Les bases de données sous Hive

Une base de données dans Hive se présente tel un catalogue ou un espace de tables. La création d'une base de données implique la création d'un répertoire à l'emplacement spécifié par la propriété "hive.metastore.warehouse.dir". Ce répertoire se compose d'un ensemble de sous répertoires associés à chaque table de la base. Dans le cas de la non spécification d'une base de données, Hive désigne la base de données par défaut.
La suppression d'une base de données dans Hive s'effectue seulement après la suppression de toutes les tables ou en ajoutent le mot clé CASCADE à la syntaxe de suppression de la base[15].

2.3.4.1 Les tables Hive

Une table Hive est constituée des données stockées et les métadonnées associées décrivant l'agencement des données de la table. Les données sont stockées généralement dans HDFS, comme elles peuvent résider dans un répertoire externe spécifié lors de la création des tables [15]. Lors de la création d'une table, Hive ajoute deux propriétés à celle-ci. La propriété `last_modified_by` qui contient le dernier utilisateur qui a modifié la table. Et la propriété `last_modified_time` qui contient le moment de la la dernière modification [27]. Dans Hive, il y a deux types de tables :

- **Les tables internes :** aussi appelées managed qui sont des tables dont le cycle de vie de leurs données est contrôlé par Hive. L'entrepôt Hive stocke les données des tables internes dans un sous-répertoire du répertoire défini par hive.metastore.warehouse.dir, par défaut.

- **Les tables externes :** est une table dont les données sont stockées en dehors de l'entrepôt Hive. La création et la suppression des données sont contrôlées par l'utilisateur, en spécifiant l'emplacement des données pendant la création.
  Généralement les tables externes sont utilisées pour accéder à un ensemble de données initialement stocké dans HDFS par un autre. Comme elles peuvent être utilisées pour exporter les données pour un processus tiers [15].

2.3.4.2 Format de stockage dans Hive

Il y a deux dimensions qui régissent le stockage de la table dans Hive, le format de la ligne et le format de fichier :

- **Le format de ligne :** est défini par SerDe(Sérialiseur/Désérialiseur). Un désérialiseur c'est dans le cas de l'interrogation d'une table, un SerDe sera désérialisé une ligne de données à partir des octets dans le fichier d'objets utilisés en interne par Hive. Lorsqu'il est un sérialiseur, dans le cas de l'exécution d'une instruction **INSERT** ou **CTAS**, SerDe de la table sera sérialisé représentation interne d'Hive d'une ligne de données dans les octets qui sont écrits dans le fichier de sortie [27].

- **Le format de fichier :** détermine le format de conteneur pour les champs dans une rangée. Le format le plus simple est un simple fichier texte, mais il y a des formats binaires en colonnes ligne orientée [26].

2.3.4.3 Partitionnement de table

Lors du partitionnement sous Hive,les tables sont divisées en plusieurs partitions. Chaque partition est stockée dans un sous répertoire du répertoire de la table partionnée. Le partitionnement s'effectue selon la valeur du champ spécifié par l'utilisateur avec la clause **PARTITIONED BY <valeur_champ>**, et une partition peut être sous partitionné [28].

2.3.4.4 Buckets de table

Les données sont organisées en fichiers séparés selon la valeur du hashé calculé à partir de la valeur du champ bucket définie par l'utilisateur. Les tables ou les partitions peuvent être bucketed en utilisant **CLUSTERED BY** et les données peuvent être stockées dans ses

buckets via **SORT BY** [27]. Il y a deux raisons principales pour lesquelles les tables sont organisées avec des Buckets, qui sont :

- Efficacité des requêtes : le Bucketing impose une structure supplémentaire sur la table. La jointure de deux tables est plus efficace si la jointure se fait sur une colonne " bucketée " de part et d'autre.

- Efficacité d'échantillonnage : lorsqu'on travaille avec un grand ensemble de données, il est très pratique de tester la requête sur une fraction de l'ensemble de pendant la phase de développement [15].

2.3.5 Les limites de Hive

Les limites de Hive se résument dans [29] :

- Toutes les requêtes standard ANSI SQL ne sont pas prises en charge par le langage de requête Hive (HiveQL).
- Le langage Hive ne supporte pas l'insertion au niveau des lignes, la mise à jour et la suppression.
- Après la création de la table, le type de colonne ne peut être modifié.
- Le temps de latence des requêtes Hive est plus élevé que celle d'une base de données traditionnelle.
- Hive ne fournit pas les transactions. Il ne peut être utilisé dans l'OLTP. Pour cela une base de données NoSQL doit lui être associée.

2.4 Installation et Configuration

Cette partie décrit comment installer et configurer un cluster Hadoop avec intégration de l'outil Hive.

2.4.1 Installation

Installation d'un cluster Hadoop implique généralement le désarchivage du logiciel sur toutes les machines du cluster ou de l'installer via un système de désarchivèrent en fonction du système d'exploitation. Pour la répartition des rôles dans le cluster :

- **Nœuds maitres :** Dans le cluster est désigné un nœud comme NameNode et un autr comme JobTracker qui est actuellement surnommé ResourceManager.

- **Nœuds esclaves :** Le reste des nœuds dans le cluster sont à la fois des DataNode et NodeManager.

2.4.2 Configuration du clusteur

Configuration d'Hadoop s'effectue dans deux types de fichiers de configuration importants :

- Pour une configuration par défaut, on a les fichiers suivants : *core-default.xml*, *hdfs-default.xml*, *yarn-default.xml* et *mapred-default.xml*.
- Pour des configurations plus spécifiques on a les fichiers suivants : *core-site.xml*, *hdfs-site.xml*, *yarn-site.xml* et *mapred-site.xml*

les fichiers de configuration d'Hadoop sont disponibles dans le répertoire **/usr/local/hadoop/conf**. Ils fonctionnent sur le principe de **clé/valeur** : la clé correspondant au nom du paramètre et valeur à la valeur assignée à ce paramètre. Ces fichiers de configuration utilisent le format XML. Les nouveaux paramètres sont à ajouter entre la balise <**configuration**>...</**configuration**>. Ces paramètres de configuration sont lus par le Framework Java d'HDFS et de MapReduce. Dans la configuration suivi dans le chapitre 4, nous avons choisie la la deuxième configuration, où :

- Dans le fichier **/usr/local/hadoop/etc/hadoop/core-site.xml**, on a spécifié l'URI du NameNode, qui devra être connue par tout le système.
- Dans le fichier **/usr/local/hadoop/etc/hadoop/hdfs-site.xml**, on a spécifié les paramètres spécifiques au système de fichiers HDFS.
- Dans le fichier **/usr/local/hadoop/etc/hadoop/mapred-site.xml**, on a spécifié les paramères de MapReduce.

2.4.2.1 Initialisation du système de fichier HDFS

Avant de démarrer le serveur Hadoop, le système de fichiers HDFS doit d'abord être formaté. Dans le cas de l'installation single node, seul le système de fichiers HDFS de la machine locale sera formaté.Pour formater, il faut exécuter la commande suivante en prenant soin d'employer l'utilisateur hdfs puisqu'il est le seul à avoir les droits dans le répertoire :/lib/hadoop-hdfs/cache/hdfs/dfs/namenode.Ace la commande suivante : **hdfs -format namenode**.

2.4.2.2 Démarrage et arrêt du cluster Hadoop

Ainsi, pour démarrer l'ensemble des services que installé, il faut lancer la commande suivante : **Start-all.sh**.Pour arrêter : **stop-all.sh**.

2.4.2.3 Les interfaces utilisateurs d'administrations d'Hadoop

La distribution Hadoop par Apache fournit des interfaces utilisateurs pour l'administration. Ceux-ci sont accessibles via des applications Web. La première concerne l'état du cluster et est accessible via l'adresse http ://localhost :8088. Il sera possible d'avoir une vue globale sur les nœuds du cluster et sur les jobs en cours d'exécution, la figure donnée ci-dessous :

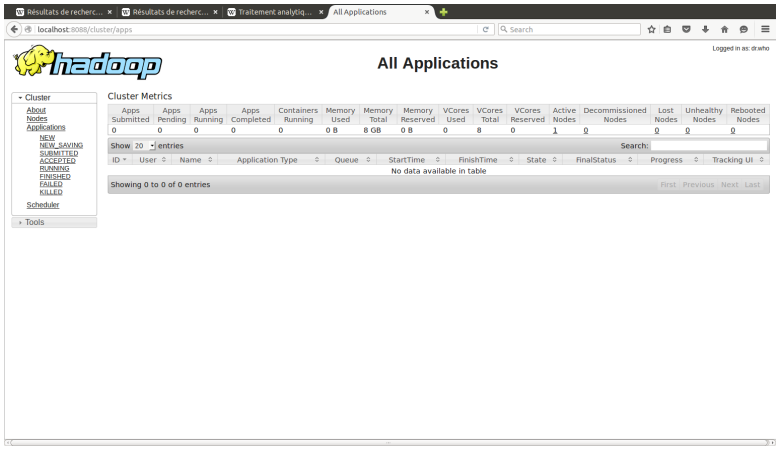

FIG. 2.9 – *Interface utilisateurs, d'accueil.*

La deuxième interface utilisateur concerne l'accès aux données contenues dans le nœud NameNode et est accessible via l'adresse http ://localhost :50070. Elle permet d'obtenir des informations sur la capacité totale et connaître l'état de disponibilité des nœuds. Elle permet également d'avoir des informations sur les fichiers et de naviguer dans le HDFS du cluster.

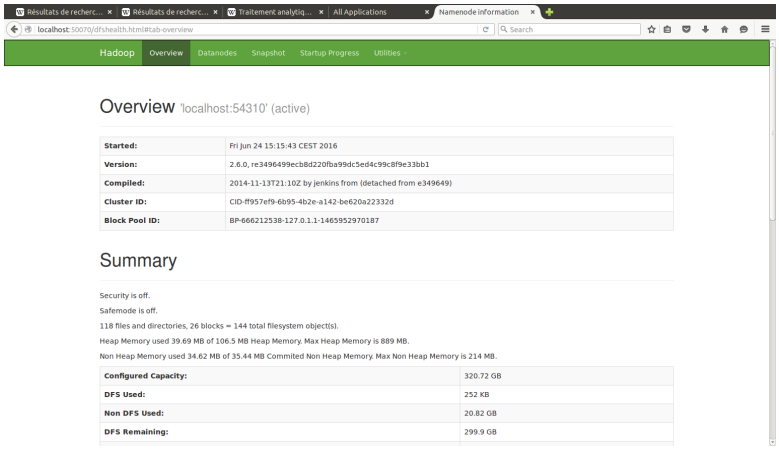

FIG. 2.10 – *Interface utilisateurs, du NameNode.*

2.4.3 Intégration d'Hive

Pour Hive, il faut installer la version correspondante à la version d'Hadoop, dans notre cas c'est la version 1.2.1, correspondante à la version 2.6.0 d'Hadoop.Pour l'intégration d'Hive avec Hadoop, il faut d'abord définir les variables d'environnement, ensuite configuré les fichiers d'Hive, comme suit :

- Définition des variables d'environnement d'Hive dans le fichier : *.bashrc* .
- Création des répertoire " warehouse " et " tmp " de Hive. Le répertoire " warehouse " est l'emplacement par défaut, pour stocker les bases ou tables données d'Hive. Et le répertoire temporaire " tmp " est l'emplacement temporaire pour stocker les résultats intermédiaire dans Hive.

Enfin,Lancer l'exécution du shell d'Hive avec la commande : **hive**.

2.5 Conclusion

En plus d'augmenter la productivité développeurs et l'efficacité des clusters, Hive est également extensible de plusieurs et différentes manières. Cette flexibilité lui permet d'être utilisé pour les transformations qui ne peuvent être facilement exprimées en langage SQL, et cela en permettant aux utilisateurs de brancher leur propre calcul à travers des fonctions définies par l'utilisateur ou en spécifiant un script personnalisé avec les jobs MapReduce.

Hive fournit un moyen pour exprimer différents types de transformations. En outre, il fournit des interfaces qui permettent aux utilisateurs de préciser leurs propres types ou formats de données qui deviennent très utiles lors du traitement de données existantes ou des types de données riches.

CHAPITRE 3

Les projets réalisés sous la plate-forme Hadoop avec Hive

3.1 Introduction

Un entrepôt de données actuellement est un référentiel de données unique de l'organisation, qui doit répondre aux nouveaux besoins : nature, volumes, qualité de données, performances et des exigences des utilisateurs. Comme indiqué dans le chapitre 1, il y a plusieurs problèmes dans l'environnement actuel des entrepôts de données, qui doivent être abordées, l'un de ces problèmes est l'infrastructure, qui ne peut pas répondre aux nouveaux besoins. L'émergence d'une nouvelle technologie tel qu'Hadoop, représente une réponse aux besoins étendus des nouvelles données et des utilisateurs. La question est de savoir comment ils sont mis en œuvre l'infrastructure d'un entrepôt de données sous Hive ? D'où l'objectif de ce chapitre, en exposant les divers projets d'entreposage de données mis en place avec Hive et Hadoop, leurs caractéristiques, composantes de leurs architectures et la circulation du flux de données.

3.2 Les caractéristiques de la nouvelle génération des entrepôts de données

La croissance continue de l'ensemble des données et la charge de travail jouent un rôle important dans les considérations de conception de l'entrepôt de données. Sur la base des limites des entrepôts de données traditionnelles citées dans le chapitre 1, nous pouvons mieux définir certaines des caractéristiques des entrepôts de données de la nouvelle génération et les plates-formes de données [30], qui sont :

- Stockage des résultats intermédiaires ; par exemple effectuer une requête et utiliser le résultat de cette requête comme une partie de l'entrée de l'autre.

- Efficacité en termes de ressources, des flux de données doivent être analysés sans être limités par la nature des données : des données complexe provenant du web, au format texte, image, géo-démographiques, ou relever de la propriété des consommateurs.

- Facilité d'administration, rentabilité, et tout en offrant un temps de réponse raisonnable.

- Performance est fondamentale, malgré la complexité des requêtes et la charge du travail, les temps de réponse doivent être en quelques secondes ou minutes.

- Soutenir un plus grand nombre d'utilisateurs et de requêtes associées avec une gamme beaucoup plus large de types de requêtes.

- Flexibilité est primordiale, ou le besoin d'utiliser des statistiques descriptives, sur des données à forte densité en information afin de mesurer des phénomènes, détecter des tendances

3.3 Architecture globale de la nouvelle génération d'entrepôt de données avec la plate-forme Hadoop

Les concept d'architecture des entrepôts de données de la nouvelle génération en tendance a compté sur une approche " polyglot ", l'utilisation d'un ensemble de bases de données différentes, les bases de données de SGBDR traditionnelles, base de données en colonnes, data store (magasin de données) non-relationnelles et des systèmes évolutifs associés aux nouvelles technologies émergentes NoSQL. Comme nous allons le voir dans la section 3.4, les différentes bases de données composant l'infrastructure des entrepôts de données de la nouvelle génération sont spécialisées pour traiter différents types de données et résoudre les différents problèmes de charge de travail. Au lieu d'utiliser un logiciel de gestion de base de données unique pour toutes les exigences de données de l'organisme [30].

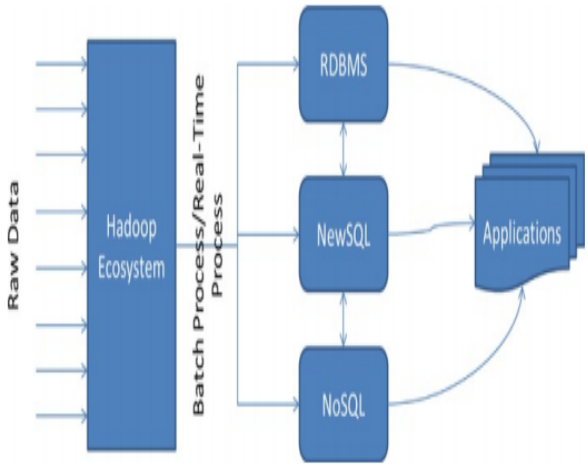

FIG. 3.1 – Illustration d'une architecture conceptuelle polyglot [30].

Pour pouvoir combiner la plate-forme Hadoop ou NoSQL et l'infrastructure de l'entrepôt de données à base de SGBDR existant, le déploiement d'un connecteur entre les deux systèmes est obligatoire. Ce connecteur sera un pont pour échanger des données entre les deux plates-formes, tel que Sqoop.La plupart des fournisseurs de SGBDR, BI, analyse et NoSQL a développé Hadoop et NoSQL connecteurs. La complexité de cette architecture est dans la dépendance à l'égard de la performance du connecteur. Les connecteurs imitent largement le comportement JDBC, et la bande passante pour le transport de données sera un goulot d'étranglement grave, compte tenu de la complexité de la requête dans le processus de découverte de données [31].

3.4 Projets d'entreposage de données mis en place Hive

Dans cette section, nous allons voir des exemples concrets de différents secteurs, de la façon dont Hive est utilisé dans l'environnement des entrepôts de données et de son intégration avec l'ensemble des composantes existantes. Et découvrir d'une certaine manière l'infrastructure des entrepôts de la nouvelle génération :

3.4.1 Secteur des réseaux sociaux

Les différents sites de réseaux sociaux produisent et fournissent quotidiennement un ensemble massif de données, d'une croissance considérable et de formats multiples. Alors que Twitter est fixée à 140 caractères, Youtube, Flicker et Facebook peuvent avoir des messages de différentes tailles en provenance d'un seul utilisateur. Ces messages peuvent être transmis, partagés et suivis par plusieurs autres utilisateurs. L'adoption d'une plate-forme capable de gérer

et analyser la quantité de données produite et de traiter chaque nouveau type d'information collectée, est devenue une nécessité pour les médias sociaux.
La solution adoptée par le géant des réseaux sociaux Facebook consiste à une plate-forme open source Hadoop pour le stockage de bases données distribuée et la sauvegarde de serveurs de données MySQL. Avec le passage du clusteur Hadoop de 21 au 30 prétaoctets stockés seulement en une année, Facebook a mis en place un outil d'entreposage de données Hive pour interagir avec le cluster Hadoop réalise des traitements analytiques.
Le cluster Hadoop de Facebook est composé d'un cluster de production utilisé pour l'exécution de tâches exigeant un temps de réponses très bref ; Et en un cluster ad-hoc pour l'exécution de lots de propriété inférieurs et l'analyse ad-hoc sur un ensemble de données historisées .

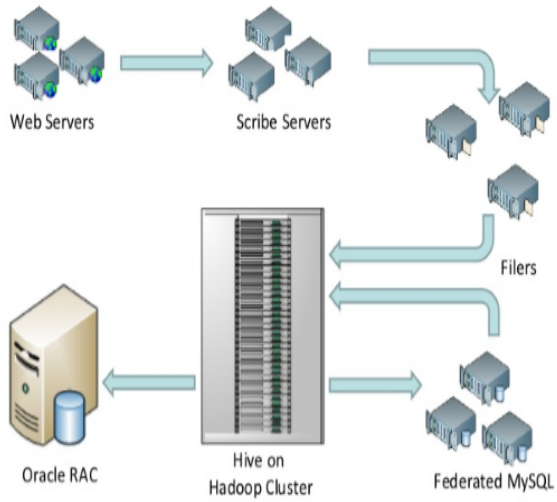

FIG. 3.2 – L'implémentation de Hive chez Facebook [32].

La Figure 3.2 illustre le flux de données au niveau de l'entrepôt de Facebook. Les serveurs Scribe regroupe les fichiers logs provenant des différents serveurs Web qui sont ensuite injectés dans les clusters Hadoop sous format des fichiers HDFS. Les données de MySQL fédérateur sont aussi chargées au niveau des clusters grâce au processus chargement scrape. Ensuite un traitement métiers est effectué via MapReduce et la publication des résultats s'effectue au niveau du serveur Oracle, ainsi que de MySQL [32].

3.4.2 Secteur des données Spatiales

La gestion et l'interrogation des données spatiales sont devenues de plus en plus complexe surtout avec l'apparition continue de technologies d'imagerie à haute résolution et d'application de géolocalisation qui a conduit à l'augmentation du volume de données spatiales et à la

diversification des ressources. Pour remédier à cela, le Centre américain d'informations biotechnologiques, National Center for Biotechnology Information (NCBI) a implémenté en 2013 Hadoop-GIS, un système d'entreposage de données spatiales scalables, à haute performance, d'interrogation spatiale pour soutenir les requêtes analytiques sur les données à grande échelle sur Apache Hadoop.

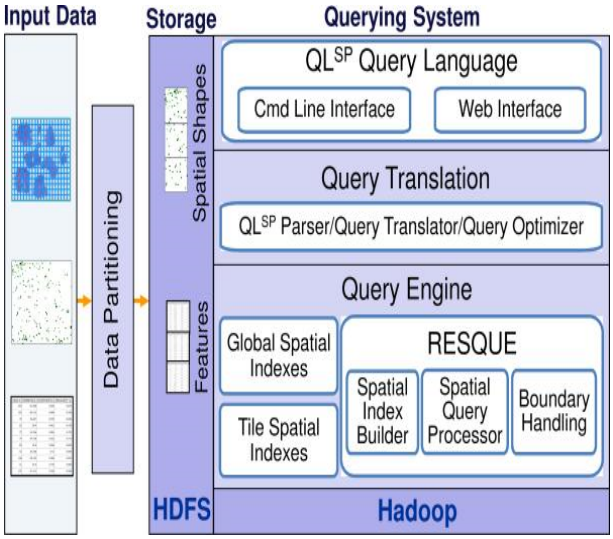

FIG. 3.3 – Présentation de l'architecture Hadoop-GIS (Hivesp)[33].

Hadoop-GIS est système performant évolutif, qui se caractérise par [33] :

- Partitionnement de données spatiales en deux dimensions et génère un ensemble de tuiles, qui est l'unité de traitement pour l'interrogation des tâches.

- Moteur de recherche autonome, qui soutient une variété de requêtes spatiales, facilement parallélisables sur des clusters, et exploite les méthodes d'indexation et d'interrogation existantes.

- Parallélisation de requête par MapReduce et utilisation du nom de la tuile comme clé. Cela réduit le temps de réponse des requêtes.

- Langage de requête spatial (Spacial Query Language) basé sur le langage HiveQL adapté aux constructions spatiales. Il permet une traduction des requêtes spatiales et une exécution automatique.

Le traitement d'une requête spatiale s'effectue en trois étapes, traduction, génération du plan logique et génération du plan physique au niveau de HiveSP, la version étendue de Hive intégrée au niveau des clusters de Hadoop-GIS. Cette version spatiale de Hive est composée d'un[33] :

- Query translator : analyse et traduit les requêtes SQL sous forme d'arbre de syntaxe.

- Query optimiser : applique un ensemble de règles d'optimisation sur l'arbre d'opérateur pris comme paramètre d'entrée.

- Query engine : prend en charge les opérations d'infrastructure comme la comparaison spatiale des relations.

  Les colonnes des tables de HiveSP définie des types de données issues de la norme ISO SQL/MM spaciale. Et une fois, les données spatiales sont chargées via l'ETL, l'écriture des requêtes spatiales peut commencer.

3.4.3 Secteur des enterprises

La plupart des entreprises ayant des problèmes de gestion de l'information à grande échelle, se lancent dans la mise en oeuvre d'entrepôt de données Hive en adoptant l'une des diverses plates-formes d'entrepôt proposée par l'une des distributions propriétaire du framework Hadoop, tels que Cloudera entreprise Data Hub (CDH) adopté par le leader mobile mondial Nokia et l'opérateur télécoms SFR comme moyen fiable de stockage de données à l'échelle pétaoctet et de traitement parallèles de haute performance pour assurer un suivi continu de leurs clients afin d'améliorer l'expérience avec leurs produits [31].

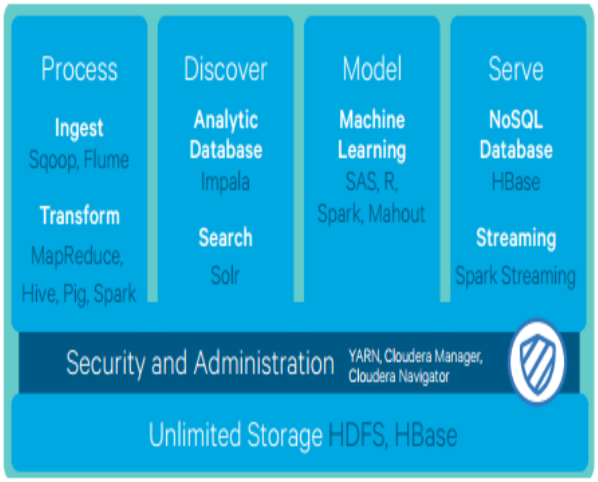

FIG. 3.4 – Illustrative des composantes de la plate-forme Cloudera entreprise Data Hub (CDH) du distributeur Cloudera d'Hadoop [34].

Hortonwork (Premier distributeur des solutions propriétaires d'Hadoop) et Amazone ont mis en place Amazone Elastique MapReduce (Amazone EMR), qui intègre le service pour l'entrepôt bâti sur Hadoop, Hive. Cette plate-forme prend en charge aussi la configuration de Hadoop, la collecte des logs, le monitoring des noeuds et les paramètres de sécurité [35].

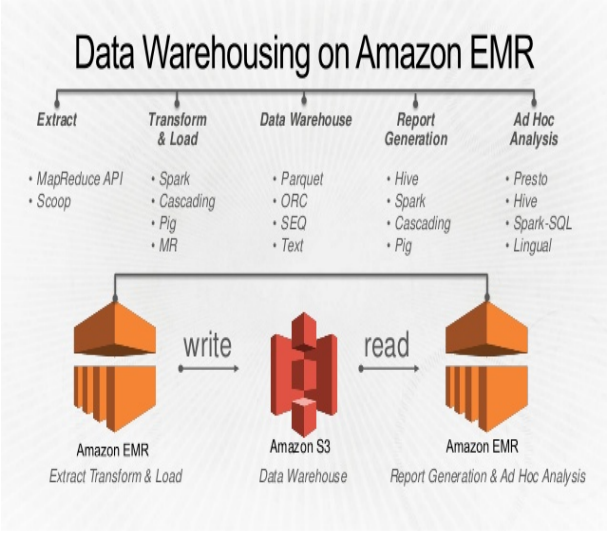

FIG. 3.5 – L'entrepôt de données avec Amazone EMR [35].

Une nouvelle plate-forme Hadoop d'entreposage de données est apparue dans le secteur des entreprises, le datalake un magasin de données qui fournit un référentiel de capture de données, quels que soit leur type, leur taille et leur vitesse. Elles peuvent être partagées à des fins de

collaboration, tels qu'Azure Data Lake de Microsofte construite sous la plate-forme Hadoop. Elle est conçue pour le traitement et l'analyse de hautes performances des données reçues en temps réel des capteurs et des boutiques en ligne, y compris la prise en charge de travail à faible latence sans restreindre la taille des fichiers ou des comptes. Azure Data Lake contient des clusters Hadoop, Spark, HBase et Storm entièrement gérés et pris en charge par Azure HDInsight, qui est un service dans le cloud basé sur Hadoop, et un ensemble d'outils et d'applications HDFS d'Apache YARN [36][37].

FIG. 3.6 – Représentation d'Azure DataLake [37].

Le Data Lake de Microsoft fournit tous les avantages d'Hadoop et d'Hive qui se constituent au niveau [36][37] :

- **Stocker et analyser des données :** le Data Lake n'impose pas de limites concernant la taille des fichiers ou des comptes et fournit un débit conséquent pour améliorer les performances des analyses, sans avoir le besoin de réécrire le code à mesure, d'augmenter ou diminuer la taille des données stockées ou le volume de calcul utilisé. Les utilisateurs peuvent réaliser des requêtes Hive afin d'analyser les données non-structurées au sein de l'interface d'Azure Data Lake et assure un monitoring des activités de leurs clients.

- **Outils :** les administrateurs de bases de données peuvent utiliser les compétences existantes, telles que U-SQL, Hive et Storm pour la visualisation des travaux, utiliser l'expérience de bloc-notes fournie par Spark ou encore Hadoop et .NET pour gagner en productivité.

- **Intégration :** Data Lake intégré avec Active Directory pour la gestion des utilisateurs et des autorisations. Il inclut également des outils web et de programmation pour la gestion et la surveillance. Il fonctionne avec Azure SQL Data Warehouse, Power BI et Data Factory pour fournir une plateforme cloud complète.

3.4.4 Secteur de la santé

Afin de simplifier la gestion des soins, une filiale de connectivité et de technologie de l'information " Heath TI " d'une société pharmaceutique américaine a été créée. L'entreprise " Health TI " possède une politique de sauvegarde des revendications d'informations de santé de sept ans, mais les systèmes de base de données avaient eu du mal à répondre à l'exigence de conservation des données lors du traitement des millions de demandes chaque jour.
Alors Health TI opta pour la solution Cloudera entreprise Data Hub (FIG 3.4), elle utilise Flume pour déplacer les données de ses systèmes sources dans le cluster CDH sur une base 24/7, des traitements des transactions en ligne Oracle à des fins de facturation. L'impact de cette plate-forme au sein de cette société de santé se représente dans[30] :

- Les fournisseurs recueillent un paiement plus rapide grâce à la messagerie accélérée avec les payeurs.
- Matériel standard de l'industrie est moins cher que les technologies alternatives tout en permettant l'analyse sur les données stockées.
- Déploiement simple de la grande plate-forme de données avec le cryptage, la maintenance continue minimale.
- Plate-forme est distribuée, pour le traitement des données, l'analyse, le stockage et le cryptage.

3.5 Conclusion

À travers les divers exemples étudiés, nous avons vu comment la plate-forme Hadoop avec l'outil d'entreposage de données Hive, peut servir d'entrepôts de données et de pont entre les données structurées, non structurées et les bases de données traditionnelles, permettant un large panel d'avantage dans des divers secteurs.L'un des secteurs d'application d'Hive est le secteur de l'information de la santé, qui représente la source de données de notre entrepôt.L'objectif du prochain chapitre est l'intégration des informations médicales.

CHAPITRE 4

Conception et implémentation de l'entrepôt de données

4.1 Introduction

Avant de démontrer le potentiel méthodologique et technologique de notre entrepôt de données, nous allons d'abord rapporter la démarche suivie dans la réalisation de notre projet.

Nous avons divisé le travail réalisé tout au long de ce chapitre en deux parties : dans la première partie, nous avons utilisé l'ensemble des concepts de conception d'ED abordés dans le chapitre 1, pour aboutir à la description de notre modèle. Dans la deuxième partie, nous avons détaillé le côté implémentation de notre entrepôt de données ou nous avons présenté notre implémentation, les configurations effectuées, chargement des données dans les tables de dimension et faits, pour terminer avec la partie reporting.

L'objectif de notre travail est de développer un système d'aide à la prise de décision, capable de garantir une distributivité optimale des unités et éléments du système sanitaire.Pour cela nous allons fournir une vision globale de l'ensemble des informations concernant les système de santé, à partir de la mise en œvre d'un entrepôt de données.

4.2 Concevoir modèle multidimensionnelle

Pour garantir une répartition optimale des moyens sanitaire, Il est nécessaire de mettre en œuvre une modélisation de données qui fournit une vision globale de l'ensemble des informations concernant le système de soin : le flux de patients, type de maladie, la disponibilité des équipements, les services proposés, la localisation des établissements de santé, etc. Pour cela, nous avons conçu un modèle de données multidimensionnelles qui répond aux exigences du système de soins d'une région. Il se compose de :

4.2.1 Les Dimensions

Les tables de dimension contiennent des descriptions textuelles sur les sujets composant le système de soins. Pour pouvoir rependre aux besoins du secteur sanitaire, nous avons proposé les tables de dimensions suivantes :

- **La table de dimension Patient** :elle contient la clé primaire IDPatient et les attributs Sexe, et Age.

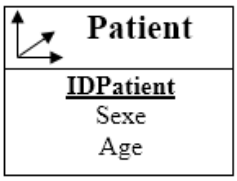

FIG. 4.1 – *Table de dimension Patient.*

- **La table de dimension Maladie** :elle contient la clé primaire IDMaladie et les attributs : FamilleMaladie et NomMaladie.

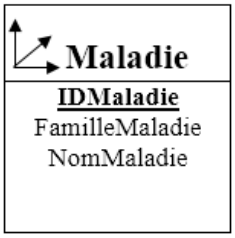

FIG. 4.2 – *Table de dimension Maladie.*

- **La table de dimension Medecin** :elle contient la clé primaire IDMedecin, l'attribut Specialite, et la clé étrangère IDEtablissement.

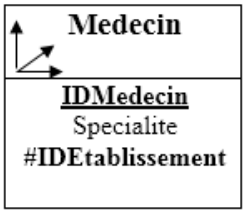

FIG. 4.3 – *Table de dimension Medecin.*

- **La table de dimension EtablissementSante** : elle contient la clé primaire IDEtablissement et les attributs : TypeEtablissement et NomEtablissemen.

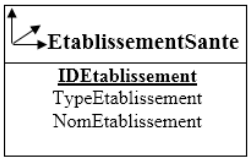

FIG. 4.4 – *Table de dimension EtablissementSante.*

- **La table de dimension Service** : elle contient la clé primaire IDService, les attributs : Pole et NomService, et la clé étrangère IDEtablissement.

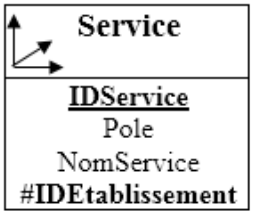

FIG. 4.5 – *Table de dimension Service.*

- **La table de dimension SalleConsultation** : elle contient la clé primaire IDSalle, l'attribut NomSalle, et la clé étrangère IDEtablissement.

FIG. 4.6 – *Table de dimension SalleConsultation.*

- **La table de dimension Equipement** :elle contient la clé primaire IDEquipement, l'attribut NomEquipement et la clé étrangère IDService.

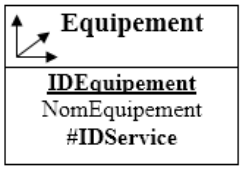

FIG. 4.7 – *Table de dimension Equipement.*

- **La table de dimension Temps** :elle contient la clé primaire IDTemps et les attributs : Mois et Annee.

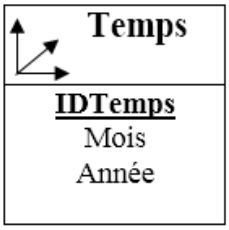

FIG. 4.8 – *Table de dimension Temps.*

- **La table de dimension Region** : elle contient la clé primaire IDRegion et les attributs : NomRegion, Departement et Ville.

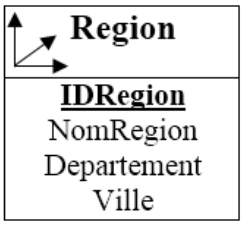

FIG. 4.9 – *Table de dimension Region.*

4.2.2 Les tables de faits

Notre conception contient trois tables de faits qui sont :

- **La table de fait Hospitalisation** :représente la gestion des séjours effectués dans l'hôpital, elle contient la clé primaire IDHospitalisation, les clés étrangères : IDPatient, IDMaladie, IDMedecin, IDService, IDEquipement et IDEtablissement.

FIG. 4.10 – *Table de fait Hospitalisation.*

- **La table de fait Consultation** : expose l'ensemble d'informations sur la consultation au sein d'un établissement de santé. Elle contient la clé primaire IDConsultation, les Clés étrangères : IDTemps, IDRegion, IDMaladie, IDPatient, IDMedecin et IDEtablissement.

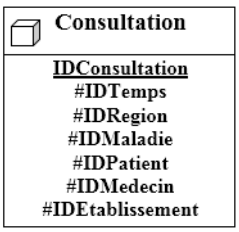

FIG. 4.11 – *Table de fait Consultation.*

4.2.3 La Modélisation

Dans notre modélisation nous avons utilisé le schéma en constellation, où les tables de faits partagent quelques tables de dimensions, tel illustré dans la FIG 4.12

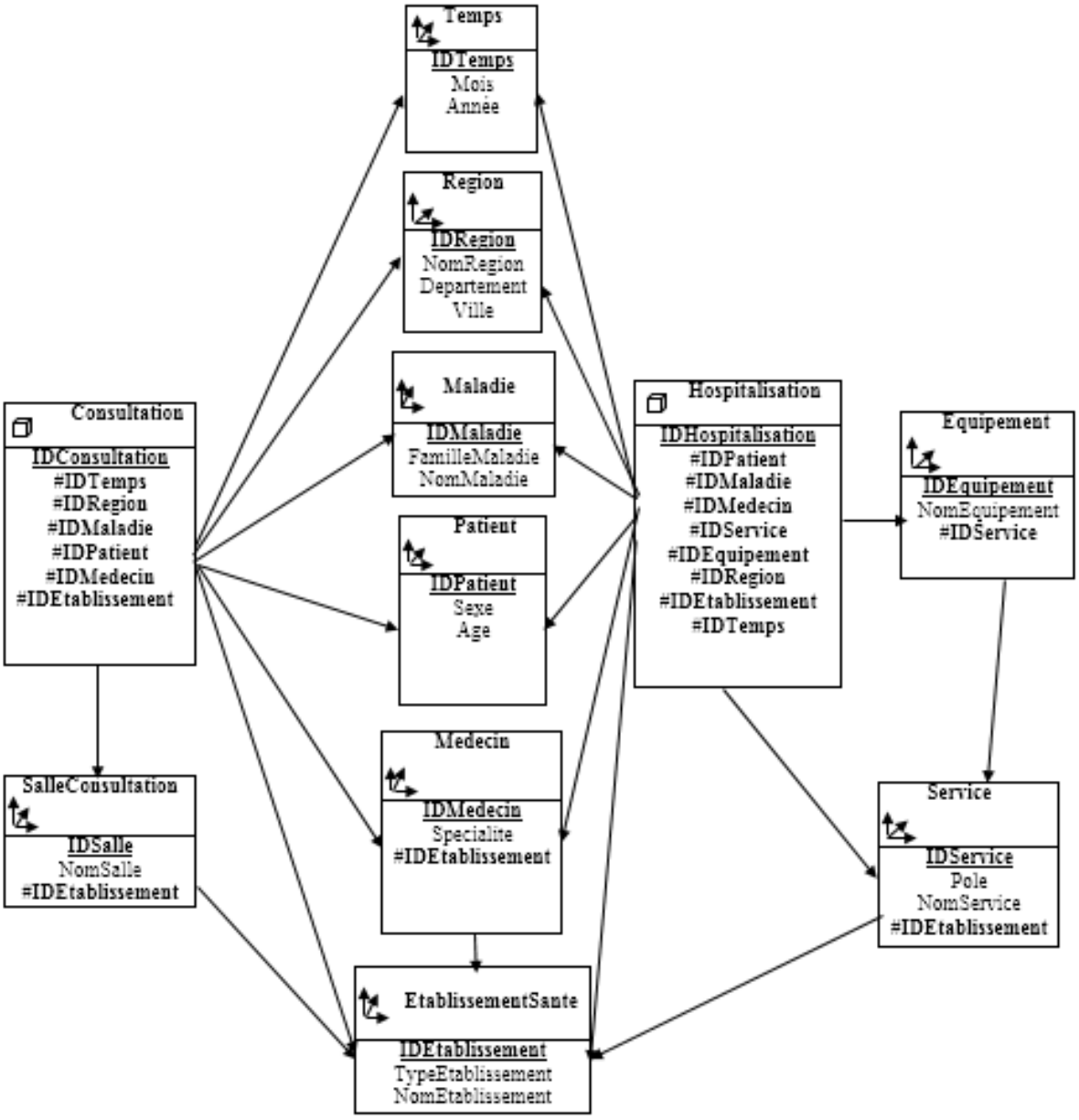

FIG. 4.12 – *Schéma en constellation.*

4.2.3.1 Palier le problème des clés de Hive

Le constat que l'on a pu faire sur la conception des données dans Hive, montre un manque au niveau d'implémentation des clés primaires et étrangères dans la gestion et la manipulation des données. Cela est liée en grande partie au fait qu'Hive, se focalisent sur les aspects de voluminosité, diversifications de la nature des données et l'aspect analytique.
Or, si l'on souhaite stocker et manipuler les données, sous plusieurs facettes et plusieurs points de vue, il est important de pouvoir identifier de manière unique chaque ligne d'une table (Clés primaires), gérer les relations entre les tables, et garantir la cohérence des données (Clés étrangères).

4.2.3.2 Solution

Afin de remédier à ce manque nous avons proposé d'exploiter les deux concepts qu'offre Hive qui sont : partitionnement de table et buckets.

a) Utilisation du partitionnement

Comme définie dans le chapitre 2, le partitionnement dans Hive est implémenté en réorganisant les données brutes en nouveaux répertoires où chaque partition a son propre répertoire, comme défini par le champ de partitionnement. Ce dernier, prend comme valeur dans notre solution de modélisation, comme illustré dans la FIG 4.13, la valeur de la clé étrangère identifiée à partir de la modélisation multidimensionnelle.

Nous avons exploité la propriété du partitionnement dans les deux tables de faits : Consultation et Hospitalisation. Nous avons appliqué un premier partitionnement selon les valeurs de la colonne de la clé étrangère IDRegion, ensuite un second partitionnement sur les valeurs de la colonne de la clé étrangère IDEtablissement. L'application du partitionnement sur les deux colonnes : **IDRegion** et **IDEtablissement**, est un moyen pour servir l'objectif majeur de notre entrepôt de données qui est de représenter le système de soins.

Nous n'avons pas appliqué de partitionnement sur les autres tables contenant un certain nombre de valeurs de la colonne de la clé étrangère, les tables : Medecin, Service, Equipement et SalleConsultation, vu le peu de valeurs que leurs colonnes contiennent, uniquement quelques valeurs, qui peuvent entrainer un partitionnement très réduit.

Le champ du partitionnement prend comme valeur la valeur de la clé étrangère d'une table, car un partitionnement sur une colonne avec une valeur unique, tel que la clé primaire de la table " Patient ", **IDPatient**, va entrainer un très grand nombre de partitions avec peu de données, qui va produire une grande nombre de sous répertoire, une surcharge inutile pour le NameNode d'HDFS.

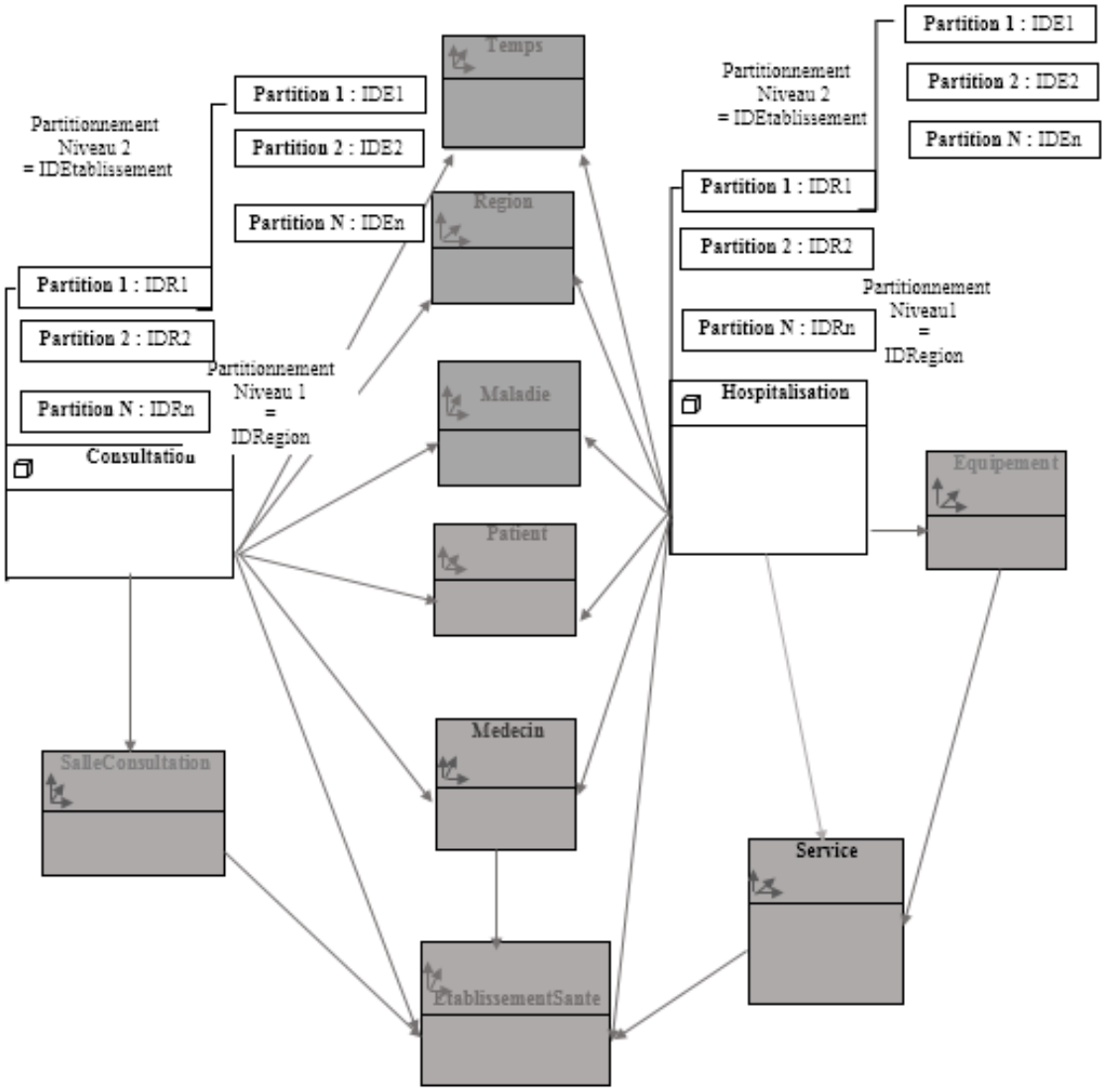

FIG. 4.13 – *Modélisation des données avec la solution du partitionnement.*

b) Utilisation de buckets

Bucketing est une autre technique pour décomposer les ensembles de données en un nombre défini de parties, qui seront stockés dans des fichiers. Cette technique présente plusieurs avantages. Le nombre de buckets est fixé de sorte qu'il ne varie pas avec les données. Pour cela nous avons choisi de faire des buckets sur les valeurs de la colonne **IDTemps** des deux tables de faits : "Consultation" et "Hospitalisation".

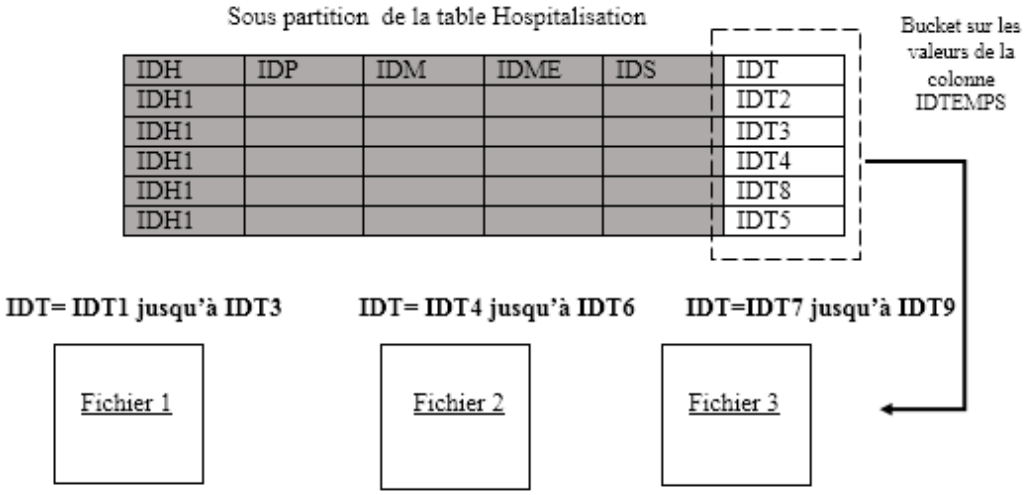

FIG. 4.14 – *Application de la solution du buckting sur une sous partition de la tables Hospitalisation.*

Voici ci-dessous, un schéma représentatif de notre modélisation de données avec les deux solutions : Partitionnement et buckets.

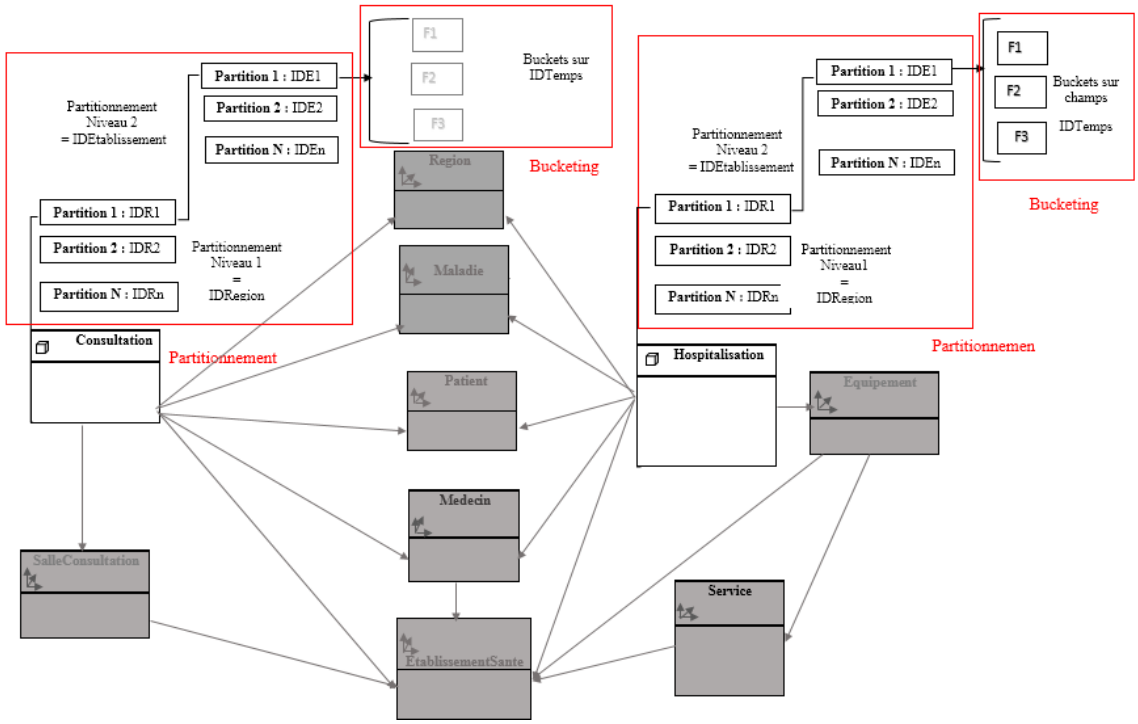

FIG. 4.15 – *Modélisation des données avec la solution du partitionnement et du buckting.*

4.3 Implémentation de l'entrepôt de données sous Hadoop avec l'outil Hive

Pour l'implémentation de notre entrepôt de données, nous avons d'abord commencé par la création de notre modèle dimensionnel, à travers la création de la base de données de l'entrepôt, ensuite la création et le chargement des tables de faits et de dimensions.

4.3.1 Environnement d'implémentation

Dans la section suivante, on va décrire les outils et systèmes composant notre environnement d'implémentation :

4.3.1.1 Matériels

Pour pouvoir installer Hadoop en single node, un micro-ordinateur de milieu de gamme (6Go à 8Go de RAM, disque dur de 500 Go ou 750Go, et quatre ou six coeurs), est recommandé.

Dans notre cas nous avons utilisé un micro-ordinateur disposant de 8Go de RAM et d'un disque dur de 500Go.

4.3.1.2 Logiciels

- **Système d'exploitation :** Hadoop a initialement été développé dans un environnement Linux, qui est actuellement le seul système d'exploitation recommandé pour exécuter Hadoop. Notre système d'exploitation est Ubuntu 14.04.

- **Installation de JAVA :** Hive nécessite Hadoop et Hadoop nécessite Java, pour l'exécution des jobs Hadoop, dans notre cas on a installé la version 7 qui est compatible avec la version 2.6.0 d'Hadoop.

- **Installation de OpenSSH :** Hadoop nécessite un accès SSH pour gérer les différents nœuds du cluster Hadoop, pour configurer l'accès vers localhost pour l'utilisateur d'Hadoop qui est " hduser ".

- **Hadoop :** Pour notre installation, nous avons utilisé la version 2.6.0, et choisi une installation single node. Dans ce mode d'installation, les nœuds d'Hadoop (NameNode, SecondaryNameNode, DataNode, JobTracker et TaskTracker) s'exécutent sur la même station de travail qu'Hadoop. Les prérequis matériels et logiciels nécessaires pour l'installation d'Hadoop.

- **Hive :** Nous avons installé la version 1.2.1, correspondante à la version 2.6.0 d'Hadoop.

4.3.2 Réalisation

Nous avons utilisé les requêtes HiveQL dans Squirrel SQL pour créer les tables et les peupler avec les données du secteur sanitaire, pour cela nous avons suivi les étapes suivantes :

4.3.2.1 Création de la base de données de l'entrepôt

Pour la création de la base de données de notre entrepôt sous Hive, nous avons utilisé les instructions suivantes :

Lorsque la base de données " datawarehouse " est créée, Hive va créer le répertoire : */user/hive/warehouse/datawarehouse.db*.Cet emplacement par défaut peut être remplacé par

```
CREATE DATABASES IF NOT EXISTS datawarehouse COMMENT 'Base Entrepot';
```

un nouveau répertoire en utilisant : **LOCATION** *<nouveaurépértoire>*.

Ici, IF NOT EXISTS est une clause facultative, qui avertit qu'une base de données avec le même nom existe déjà. **COMMENT** ajoute un commentaire descriptif à la base de données, qui sera présenté par la commande : **DESCRIBE DATABASE**. Ce dernier affiche le commentaire ajouté et également l'emplacement du répertoire de la base de données.
À tout moment, nous pouvons voir les bases de données qui existent déjà, avec la clause :

Malheureusement, il n'y a pas de commande pour nous montrer quelle base de données est

```
SHOW TABLES;
```

la base de données en cours d'utilisation. Alors, il est toujours prudent de répéter, la clause :

```
USE datawarehouse;
```

Pour désigner la base " datawarehouse ", comme base de données de travail.

4.3.2.2 Création des tables de dimensions et faits

a) Les tables de dimensions

Pour la création de la dimension 'Patient' pour notre entrepôt de données, nous avons utilisé les instructions suivantes :

```
CREATE TABLE IF NOT EXISTS patient (
IDPatient STRING COMMENT 'Identifiant du patient',
NomPatient STRING COMMENT 'Nom du patient',
PrenomPatient STRING COMMENT 'Prénom du patient',
Sexe CHAR(5) COMMENT 'F :Femme, H :Homme',
Age INT COMMENT 'Age du patient')
COMMENT 'Table de dimension'
ROW FORMAT DELIMITED FIELDS TERMINATED BY '|'
STORED AS TEXTFILE;
```

Nous pouvons associer des propriétés de clé-valeur à la table de données avec " TLPROPERTIES ", pour fournir un moyen d'ajouter des informations à la sortie de la clause DATABASE EXTENDED DATABASES.

b) Les tables de faits

Pour la création de la table des fait 'traitement' pour notre entrepôt de données, nous avons utilisé les instructions suivantes :

```
CREATE TABLE IF NOT EXISTS Hospitalisation (
IDHospitalisation STRING COMMENT 'Clé primaire',
IDPatient STRING COMMENT 'la Clé étrangère patient',
IDMaladie STRING COMMENT ' la Clé étrangère Maladie',
IDMedecin STRING COMMENT ' la Clé étrangère Medecin',
IDService STRING COMMENT ' la Clé étrangère Service',
IDEquipement STRING COMMENT ' la Clé étrangère Equipement',
IDEtablissement STRING COMMENT ' le Clé étrangère l'établissement'
IDRegion STRING COMMENT ' la Clé étrangère la Region'
IDTemps STRING COMMENT ' la Clé étrangère Temps')
COMMENT 'Table de fait'
ROW FORMAT DELIMITED FIELDS TERMINATED BY '|'
STORED AS TEXTFILE;
```

4.3.2.3 Peupler les tables de dimensions et faits

Au cours de notre projet nous avons rencontré des difficultés en ce qui concerne la récolte de données médicale dans les établissements sanitaires de la wilaya de Béjaïa, pour remédier à cela on s'est orienté vers des données de santés aléatoires issues du web.

a) Les tables de dimensions

Pour peupler la table dimension 'Patient', en charge les données à partir d'un fichier texte 'patient.tex', qui contient les données des patients. Les données peuvent être aussi chargées à partir d'HDFS. Nous avons utilisé les instructions suivantes, pour peupler la table dimension 'patient' :

```
LOAD DATA LOCAL INPATH 'patient.tex' OVERWRITE INTO TABLE patient ;
```

b) Les tables de faits

Pour peupler la table de fait 'Hospitalisation', en charge les données à partir d'un fichier texte 'Hospitalisation.tex'. Nous avons utilisé les instructions suivantes :

```
LOAD DATA LOCAL INPATH 'Hospitalisation.tex' OVERWRITE INTO TABLE Hospitalisation ;
```

Ensuite, nous avons créé les tables de faits partitionnées pour remédier au manque d'Hive et ainsi ajouter des relations entre la table de fait partitionnées et les tables de dimensions .On va d'abord créer la table de partitionnement 'HospitalisationPart' partitionner selon les valeurs du champ IDRegion, ensuite sous partitionner selon les valeurs du champ IDEtablissement, pour enfin appliqué le buckets selon le champ IDTemps :

```
CREATE TABLE IF NOT EXISTS HospitalisationPart (
IDHospitalisation STRING COMMENT 'Clé primaire',
IDPatient STRING COMMENT 'la Clé étrangère patient',
IDMaladie STRING COMMENT ' la Clé étrangère Maladie',
IDMedecin STRING COMMENT ' la Clé étrangère Medecin',
IDService STRING COMMENT ' la Clé étrangère Service',
IDEtablissement STRING COMMENT ' le Clé étrangère l'établissement'
IDTemps STRING COMMENT ' la Clé étrangère Temps')
COMMENT 'Table de fait partitionner'
PARTITIONED BY (IDRegion STRING,IDEtablissement STRING)
CLUSTERED BY (IDTemps) SORTED BY (IDHospitalisation) INTO 3 buckets
ROW FORMAT DELIMITED FIELDS TERMINATED BY '|'
STORED AS TEXTFILE;
```

4.4 Reporting

Afin de garantir une répartition optimale du système de soins, une présentation périodique d'information sur les activités et résultats du secteur sanitaire est nécessaire.

À travers le traitement analytique quand peut effectuer sur l'ensemble d'informations situé au sein de notre entrepôt de données et les performances des requêtes , ceux chargés de les superviser, les décideurs ont dans leur main un moyen stratégique pour la prise de décision.

Dans cette section nous allons présenter d'abord deux exemples de résultats trouvés à partir lesquelles un décideur peut avoir une vision globale sur les unités du secteur sanitaire. Ensuite exposer les performances des temps de réponses des requêtes issues du concept de partitionnement appliqué aux tables de données.

4.4.1 Exemple des résultats trouvés

Voici ci-dessous deux exemples de requêtes à partir desquelles on a pu fournir des présentations graphiques des résultats :

- **Premier exemple :** Pour exposer le nombre de maladies dans les hôpitaux de chaque ville, nous avons exécuté la requête suivante :

```
SELECT r.ville AS Ville, COUNT(h.idmaladie) AS nbMaladie
FROM hospitalisation h JOIN region r ON r.idregion=h.idregion GROUP BY r.ville ;
```

Résultat de la requête

ville	Nombre de maladies
Levallois-Perret	3
Saint-Médard-en-Jellas	4
Saint-Louis	5
Belfort	7
Anglet	8
Lorient	8

FIG. 4.16 – *Nombre de maladies enregistrés dans les hôpitaux par ville.*

Interprétation graphique

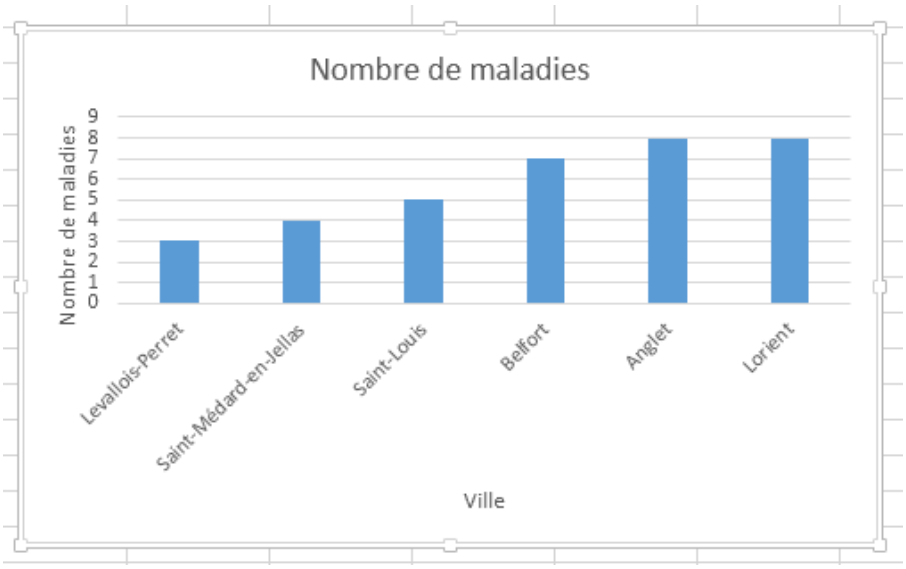

FIG. 4.17 – *Représentation graphique du nombre de maladies enregistrés dans les hôpitauxpar ville.*

- **Second exemple :** Pour exposer le nombre d'hospitalisations et de consultations dans chaque région, nous avons exécuté la requête suivante :

```
SELECT h.idregion, COUNT(idconsultation), COUNT(idhospitalisation)
FROM region r JOIN hospitalisation h ON (r.idregion=h.idregion)
JOIN consultation c ON (c.idregion=h.idregion) GROUP BY h.idregion ;
```

Résultat de la requête

Identifiant Région	Nombre Hospitalisations	Nombre Consultations
RF5	28	28
RF7	24	22
RF10	30	10
RF17	120	110

FIG. 4.18 – *Nombre d'hospitalisations et de consultations par région.*

Interprétation graphique

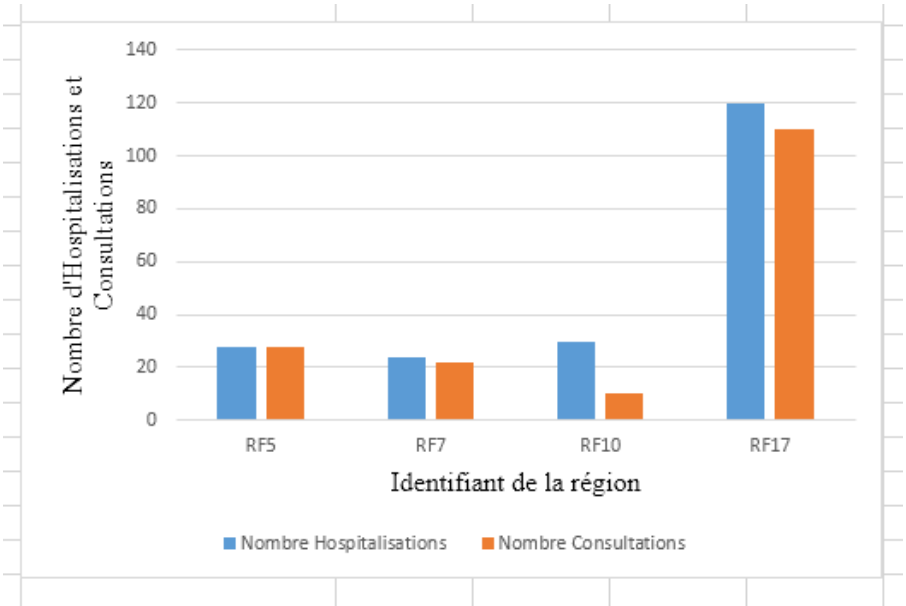

FIG. 4.19 – *Représentation graphique du nombre d'hospitalisations et de consultations par région.*

4.4.2 Performance du partitionnement

Les opérations d'E/S constituent le principal goulot d'étranglement des performances pour l'exécution de requêtes Hive. Il est possible d'améliorer les performances des requêtes en réduisant la quantité de données à lire, cela grâce au partitionnement.
Avec le partitionnement, les requêtes Hive accèdent uniquement à la quantité de données nécessaire dans les tables Hive sans avoir à lire tout l'ensemble des données.

Pour montrer cette performance que le partitionnement apporte, nous avons effectué une comparaison entre les temps de réponse d'un ensemble de requêtes exécuté sur les tables de faits partitionnées (**Hospitalisationpart** et **Consultationpart**) et les tables non partitionnées (**Hospitalisation** et **Consultation**).

4.4.2.1 Les requêtes exécutées

Voici l'ensemble de requêtes exécutées sur la table partitionnée ensuite sur la table non partitionnée :

- **Requête 1 (R1) :** SELECT idhospitalisation, idpatient, idmaladie FROM hospitalisationpart WHERE idetablissement='ESF32' ;
- **Requête 2 (R2) :** SELECT idpatient, idmaladie, idmedecin FROM hospitalisationpart WHERE idregion='FR20' OR IDRegion='FR29' ;
- **Requête 3 (R3) :** SELECT idmaladie FROM hospitalisationpart WHERE idregion='7' ;
- **Requête 4 (R4) :** SELECT * FROM hospitalisationpart WHERE idregion='RF7' AND idetablissement='ESF7' ;
- **Requête 5 (R5) :** SELECT idregion, COUNT(idpatient) FROM hospitalisation GROUP BY idregion ;
- **Requête 6 (R6) :** SELECT idetablissement, COUNT(idsalle) FROM consultation GROUP BY idetablissement ;

4.4.2.2 Les résultats obtenus

Après exécution de chaque requête nous avons récolté le temps de réponse affiché par le CLI de Hive et enregistré dans un tableau, et représenté graphiquement les résultats obtenus, illustré par les figure ci-dessous :

Les requêtes	Table partitionnée (seconds)	Table non partitionnée (Seconds)	Nombre de colonnes
R1	0,6	0,61	5
R2	0,67	0,74	7
R3	0,53	0,58	8
R4	0,73	0,76	8
R5	1,23	1,25	7
R6	1,24	1,26	7

FIG. 4.20 – *Les temps de réponse des requêtes.*

4.4.2.3 Interprétation graphique

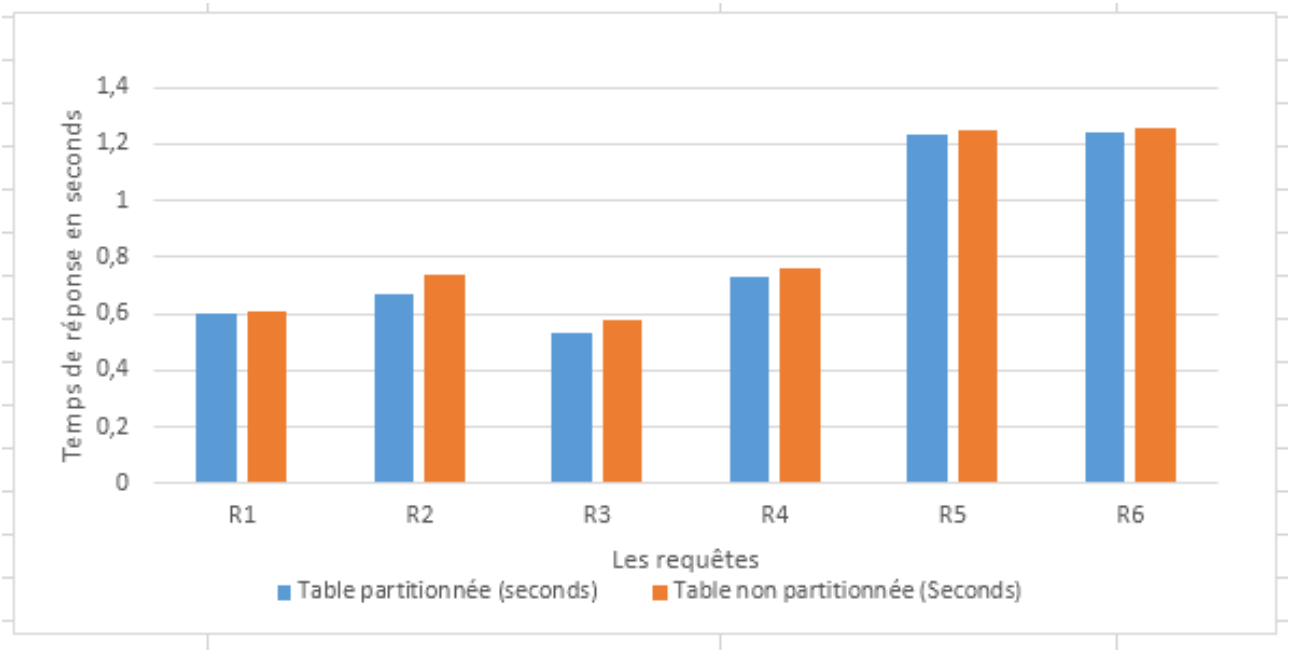

FIG. 4.21 – *Présentation graphique des temps de réponse des requêtes.*

Le temps de réponse des requêtes appliquées sur les tables non partitionnées est plus élevé, à travers ce constat nous pouvons déduire que le partitionnement de table garantie une performance en ce qui concerne le temps de réponse.

4.5 Conclusion

Au cours de ce chapitre, nous avons implémenté un entrepôt de données pour une répartition optimale des unités de santé afin de subvenir aux exigences du système sanitaire. A traves la démarche suivie dans la conception et l'implémentation, nous avons pu donner une vision globale de l'ensemble d'informations composant le système de sanitaire, ainsi qu'un ensemble

de statistiques qui va permettre une prise de décision stratégique.

Conclusion Générale et Perspectives

Ce mémoire a pour objectif l'implémentation d'un entrepôt de données dans le secteur sanitaire, Afin d'aider les décideurs et gestionnaires du système de santé dans leur démarche de prise de décision.

Pour atteindre cet objectif nous avons suivi au cours de la réalisation de notre mémoire les étapes suivantes, D'abord nous avons présenté une vision globale sur les entrepôts de données, leur architecture et leur modélisation, et exposer les limites des entrepôts de données traditionnels à travers le chapitre 1. Ensuite dans le chapitre 2 nous avons introduit dans la première partie une plate-forme performante et évolutive, le Framework Hadoop, qui contient parmi la panoplie de ces outils, un outil d'entreposage de données qui répond aux nouveaux besoins des données volumineux et qui garantit une performance de temps de réponses des requêtes. Enfin pour comprendre réellement l'implémentation de la plate-forme Hadoop et de son outil Hive, nous avons étudié dans le chapitre 3 les projets réalisés dans quelques secteurs : réseaux sociaux, spatial, entreprise et secteur de santé. À travers l'ensemble de concepts acquis, nous avons pu implémenter notre entrepôt de données sous la plate-forme Hadoop avec l'outil Hive ainsi mettre en œuvre une solution de modélisation de données qui comble les lacunes d'Hive et garantie une performance des requêtes.

La réalisation de ce projet de fin de cycle nous a permis d'acquérir une bonne expérience dans le domaine des systèmes décisionnels, plus précisément les systèmes d'entreposage de données. L'apport du projet est considérable sur le plan pratique, car il nous a permis d'acquérir d'avantage de connaissances sur les techniques de modélisation et de réalisation d'entrepôt.

Toutes les taches de notre projet nous ont permis d'enrichir nos expériences dans l'organi-

sation et la cohérence entre les différentes parties du projet et aussi, de nous familiariser avec plusieurs outils tels qu'Hadoop et Hive.

Comme perspectives, nous souhaitons avoir un système d'information opérationnelle au niveau du secteur sanitaire national, pour pouvoir construire un système décisionnel avec des données réelles et de grandes tailles. Nous aimerions aussi mettre en place un cluster Hadoop contenat plus d'une machine pour pouvoir exploiter plus la fonctionnalité de distributivité d'Hadoop. Et offrir une interface plus accueillante aux décideurs.

BIBLIOGRAPHIE

[1] Polraj Ponia, *Data Warehousing Fundamentals for IT Professionals*, The second edition John Wiley and Sons, 2011.

[2] William Inmon, *Building the data warehouse, QED Technical publishing groupe*, Wellesly,Massachusetts,USA, 1992.

[3] J F Desnos, *Entrepôt de données-introdiction, Spécialité Double Compétence : informatique et Sciences Scociales*, Université de Grenoble, 2011.

[4] William Inmon, *Building the data warehouse*, Fourth edition, Wiley edition, USA , 2005.

[5] Lilia Sfaxi, *Business Intelligence : Chapitre 2 - Les entrepôts de données (Data Warehouse)*, Institue National des Sciences Appliquées et Technologie (INSAT) en Tunisie, 2013/2014.

[6] Mathieu Durgé. *Mémoire sur le thème : Conception et réalisation d'un entrepôt de données, intégration à un système existant et étape nécessaire vers le forage de données*, Université du Québec à Trois-Rivières, Mars 2004.

[7] http// :www.durofy.com, visité le 03/05/2016.

[8] Matteo Glofarelli and Stefano Rizzi, *Data Warehouse Design : Modern Principles and Methodologies*, juin 2009.

[9] Elizabieta Malinowski and Esteban Zimanyi, *Advanced Data Warehouse Designe : From Conventional to Spatial and Temporal*, Edition Springer, 2008.

[10] Ralph Kimball, Laura Reeves, Margy Ross et Warren Thornthwaite, *Concevoir et déployer un data warehouse : Guide de conduit de projet*, édition Eyrolles, octobre 2000.

[11] Georges Gardarin, *Internet/intranet et bases de données : Data Web, Data Warehouse, Data mining*, édition Eyrolles, avril 2000.

[12] Sandro Bimonte, *Thèse Intégration de l'information géographique dans les entrepôts de données et l'analyse en ligne : de la modélisation à la visualisation*, Institut National des Sciences Appliquées de Lyon, 2007.

[13] Alejandro Vaisman and Esteban Zimányi, *Data warehouse systems : design and implementation*, Edition Springer, 2014.

[14] Krish Krishnan, *Data Warehousing in the Age of Big Data*, Edition Morgan Kaufmann, 2013.

[15] Tom White, *The Definitive Guide* , Edition Morgan Kaufmann, may 2012.

[16] Laurent Jolia-Ferrier, *Big Data concepts et mise en oeuvre de Hadoop*, Edition ENI,février 2014.

[17] https ://hadoop.apache.org, *site officiel*, visité le 04/04/2016.

[18] http ://www.linusnova.com/hive-le-data-warehouse-de-hadoop, *site officiel*, visité le 04/04/2016.

[19] http ://bradhedlund.com, visité le 04/05/2016.

[20] http ://hbase.apache.org, *site officiel*, visité le 03/05/2016.

[21] https ://www.linusuova.com, visité le 08/05/2016.

[22] http ://sqoop.apache.org, *site officiel*, visité le 08/04/2016.

[23] https ://hive.apache.org, *site officiel*, visité le 08/04/2016.

[24] http ://hortonworks.com/industry/telecom, *site officiel*, visité le 08/04/2016.

[25] Ashish Thusoo, Joydeep Sen Sarma, Namit Jain, Zheng Shao, Prasad Chakka, Ning Zhang, Suresh Antony, Hao Liu et Raghotham Murthy, *Article Hive-A Petabyte Scale Data Warehouse Using Hadoop*, 2010.

[26] https ://hive.apache.org, *site officiel*, visité le 06/05/2016.

[27] Edward Capriolo, Dean Wampler, and Jason Rutherglen, *Programmin Hive*, first edition d'Oreilly,octobre 2012.

[28] https ://cwiki.apache.org/confluence/display/Hive/LanguageManual+DDL, *site officiel*, visité le 07/05/2016.

[29] http ://bigdatawarehouse.blogspot.com , visité le 07/05/2016.

[30] Soumendra Mohanty, Madhu Jagadeesh and Harsha Srivatsa, *Big Data Imperative Entreprise Big Data Warehouse, BI Implementations and*, Edition Apress,2013.

[31] Krish Krishnan, *Data Warehouse in age of Big Data*, Edition Morgan Kaufman,2013.

[32] Ashish Thusoo, Zheng Shao Suresh Anthony, Dhruba Borthakur, Namit Jain, Joydeep Sen Sarma, Raghotham Murthy, et Hao Liu, *Data Warehousing and Analytics Infrastructure at Facebook*, 2010.

[33] Ablimit Aji, Fusheng Wang, Hoang Vo, Rubao Lee, Qiaoling Liu, Xiaodong Zhang, and Joel Saltz, *Hadoop-GIS : A High Performance Spatial Data Warehousing System over MapReduce*, Aoûte 2013.

[34] https ://www.cloudera.com/products/apache-hadoop, *site officiel*, visité le 16/04/2016.

[35] http ://aws.amazon.com, *site officiel*, visité le 17/04/2016.

[36] http ://hortonworks.com, *site officiel*, vésité le 14/04/2016.

[37] https ://azure.microsoft.com, *site officiel*, visité le 14/04/2016.

Résumé

La production massive de données de santé impose aux systèmes décisionnel des nouvelle exigences : de voluminosité, variation de source de données, intégration de données non-structurées et de traitement analytique performant, Or les infrastructures des entrepôts de données traditionnels sont inadaptées à ces nouvelles exigences de volume massive de données. L'émergence de nouvelles technologie scalable et évolutif tel qu'Hadoop a rendu possible le stockage et le traitement d'un grand volume de données.

Hadoop est basé sur d'une architecture de développement dédiée aux stockages distribués et calculs parallèles, il permet la manipulation des données en très grande quantité, distribuées sur le cluster de nœuds de serveurs. C'est ainsi que des données non structurées peuvent faire l'objet d'un traitement analytique et que la décomposition de données en blocs accélère le traitement des requêtes. Plusieurs organismes tels que Facebook, Amazone et Microsoft, ont adopté Hadoop comme plate-forme d'entrepôt de données en utilisant son composant d'entreposage 'Hive'. Une nouvelle génération d'entrepôt de données a vu le jour, une génération d'entrepôts basé sur une approche décentralisé, qui intègre différentes types de données comme les fichiers XML et HTML, issues de diverses sources de données, de quantité volumineuse et qui garantit un traitement analytique en un temps de réponse correct.

Dans cette perspective, l'objectif de ce travail est de développer un système décisionnel capable de garantir une distributivité optimale des unités et éléments du système sanitaire, pour cela nous avons implémenté notre entrepôt sous la plate-forme Hadoop avec l'outil Hive. En suivant une modélisation de données qui se présente par un schéma de constellation sur quoi nous avons intégré deux concepts d'Hive qui sont le partitionnement et le buckets, afin de combler le manque d'implémentation de clé primaire et étrangère, qui garantit aussi une performance d'exécution de requêtes en un temps de réponse optimal.

Mots-clés : Entrepôt de données, Hadoop,Cluster Hadoop, HDFS, MapReduce, HBase, Sqoop, Hive, Pig, API de straming,Zookeeper,Oozie,Partitionnement,Buckets.

Abstract

The mass production of health data requires decision-making systems of new requirements : bulking, data source changes, integration of unstructured data and powerful analytical processing, Gold infrastructure of traditional data warehouses are not adapted to the new requirements this massive volume of data. The emergence of new technology such scalable and scalable Hadoop made possible the storage and processing of large amounts of data.

Hadoop is based on a development architecture dedicated to distributed storage and parallel computing, it allows the manipulation of data in very large quantities, distributed on the server cluster nodes. Thus, unstructured data can be analytical processing and the block data decomposition speeds up the processing of applications. Several organizations such as Facebook, Amazon and Microsoft, have adopted Hadoop as a platform data warehouse using its storage component 'Hive'. A new generation of data warehouse was created a generation of warehouse based on a decentralized approach that integrates different data types such as XML and HTML files, from various sources of data, voluminous and guarantees an analytical processing a correct response.

In this perspective, the objective of this work is to develop a decision-making system that ensure optimal distributive units and elements of the health system, why we implemented our warehouse in the Hadoop platform with Hive tool. Following a data modeling that comes with a constellation diagram on which we built two concepts that are Hive partitioning and buckets, to address the lack of primary and foreign key implementation, which also guarantees execution performance of queries in an optimal response time.

Keywords : Datawarehouse, Hadoop,Hadoop cluster, HDFS, MapReduce, HBase, Sqoop, Hive, Pig, Straming API ,Zookeeper,Oozie,Partitionnement,Buckets.