République Algérienne Démocratique et Populaire
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique
Université de Larbi Tébessi –Tébessa-
Faculté des Sciences Exactes et des Sciences de la Nature et de la Vie
Département : Mathématiques et Informatique

MEMOIRE DE MASTER
Domaine : Mathématiques et informatique
Filière : Informatique
Option : Systèmes d'informations

Thème :

Construction des Cubes OLAP dans L'environnement Cloud Computing avec MapReduce

Présenté par :
CHABBI Ridha
CHAIB Hamza

Devant le jury :

Abdel Malek Matrouh	MAA	Université de Tébessa	Président
BOUAKEZ Fatima	MAA	Université de Tébessa	Examinatrice
BRADJI Louardi	MCB	Université de Tébessa	Encadreur

Date de soutenance : 29/05/2016

Note : ……. Mention : ……

Résumé

Toutes les informations techniques, économiques, pratiques et culturelles sont accessibles en temps réel, pour promouvoir l'activité des entreprises, Ce mémoire traite la problématique de l'augmentation des données et la difficulté d'analyser ces données. De ce fait, l'objectif principal de ce travail est d'utiliser de nouvelles technologies Hadoop et MapReduce et à son utilisation dans un Cloud Computing. Notre thème s'articule autour de plusieurs axes (OLAP, DW, Hadoop, MapReduce, Cloud Computing). Nous avons exploité un entrepôt de données sous Hive pour effectuer des requêtes décisionnelles afin de construire un Cube en utilisant plusieurs dimensions.

Nous avons aussi comparé les résultats des temps de construction de cube OLAP en différent environnements de différentes tailles d'entrepôts de données selon différent requêtes de construction de cube OLAP.

Mots-clés: Entrepôt, OLAP, Cube, Hadoop, MapReduce, Cloud, Hive, SQL Server, SQL, HQL.

Abstract

All technical, economic, cultural and practical information is accessible in real time, to promote business activity.

This manuscript addresses the issue of data growth and difficulty of analyzing the data. Therefore, the main objective of this work is to use Hadoop and MapReduce technology and exploit its strength on the cloud. Our theme revolves around several axes (OLAP, DW, Hadoop, MapReduce and Cloud Computing). We operated under a Hive data warehouse to perform decision queries to build a Cube using several dimensions.

We also compared the results of OLAP cube build time in different environments and with different sizes of data warehouses using different queries for building OLAP cube.

Keywords: Warehouse, OLAP Cube Hadoop, MapReduce, Cloud, Hive, SQL Server, SQL, HQL.

Remerciements

Tout d'abord, nous tenons à remercier Allah tout puissant, de nous avoir permis de mener à bien ce Mémoire, et de nous avoir orienté au chemin du savoir.

Ensuite, je voudrais exprimer mes vifs remerciements :

- *À Mr.* ***BRADJI Louardi*** *pour la confiance qu'il nous a témoignée en acceptant de nous encadrer dans ce travail, et qui n'a cessé de nous prodiguer ses conseils, ses orientations et son aide à plusieurs reprises.*
- *A tous les membres du jury pour avoir accepté de juger ce travail.*

Au terme de ce travail, nous tenons à exprimer toutes nos reconnaissances et remerciements à toute personne ayant contribué de près ou de loin à l'élaboration de ce mémoire.

بسم الله الرحمن الرحيم

A mon père

Que Dieu bénisse son âme

A ma mère

Auxquels je dois ce que je suis. Que dieu la protège.

A mes sœurs et A mes frères.

Qu'ils trouvent dans ce mémoire l'expression de mes remerciements les plus sincères.

A tous mes amis proches.

A toutes les personnes qui ont une place spéciale dans ma vie.

A toute la famille.

Chaib Hamza

Dédicace

A notre cher et dynamique encadreur
Bradji Louardi
Un remerciement particulier et sincère pour tous vos efforts fournis. Vous avez toujours été présent.
Que ce travail soit un témoignage de ma gratitude et mon profond respect.
A mon chère Père,
A ma mère et ma sœur
A mes deux Frères
En témoignage de l'attachement, de l'amour et de
L'affection que je porte pour vous.
Je vous remercie pour
Votre affection si sincère.
A mes chers oncles
A mes chères tantes
et leur famille
Vous avez toujours été présents pour les bons conseils.
Votre affection et votre soutien m'ont été d'un grand secours au long de ma vie professionnelle et personnelle.
Veuillez trouver dans ce modeste travail ma reconnaissance pour tous vos efforts.
A mes chères amies
Je ne peux trouver les mots justes et sincères pour vous exprimer mon affection et mes pensées, vous êtes pour moi des frères, des amis sur qui je peux compter.
En témoignage de l'amitié qui nous uni et des souvenirs de tous les moments que nous avons passé ensemble, je vous dédie ce travail et je vous souhaite une vie pleine de santé et de bonheur.

Ridha Chabbi

Liste des tableaux

Liste des figures

Table des Matières

Introduction Générale

Pendant de nombreuses années, les entreprises stockent les données dans une base de données et les traites via requêtes SQL. L'accroissement de ces données a apporté des difficultés au niveau des ressources de stockage et l'efficience de production des rapports. Pour faire face à ces problèmes plusieurs technologies ont été mises en place au sein des entreprises. Ces technologies permettent un traitement sophistiqué pour le support d'un processus d'aide à la décision qui repose sur les concepts d'entrepôt de données et d'OLAP. Dans l'objectif de homogénéiser et consolider les données de manière uniforme la technologie des Data Warehouse a vu le jour comme une base multidimensionnelle stockant des données pertinente organisé selon des thèmes principaux et OLAP transforme les données de l'entrepôt en informations stratégiques. Afin d'assurer un support efficace pour les analyses OLAP et l'exploration de CUBE, plusieurs outils et requêtes OLAP permettent la construction de ces cubes à l'aide d'opérateurs spécifiques.

En même temps, l'entreposage de gros volume de données exige un environnement haut niveau tel que Cloud Computing. Ce dernier fournit des services permettant la manipulation des Cube OLAP avec le paradigme MapReduce. Ces données sont importées dans le système Hadoop Distributed File (HDFS). Ensuite, une certaine forme de traitement a lieu en utilisant MapReduce où l'un des plusieurs langages construits sur MapReduce (Hive, Pig,).

Le but de notre recherche vise à exploiter un entrepôt de données AdventureWorksDW pour construire un cube OLAP en utilisant deux langage d'interrogation le premier est SQL qui est adaptée aux donnée structurée l'autre est habituée à des donnée non structurée et structurée. Il arrive à traiter des données de taille massive, et nous montrons qu'à travers nos expérimentations les performances de différentes requêtes de construction de cube OLAP dans différentes environnement.

Ayant présenté les outils et la méthode adoptée, nous allons maintenant exposer le plan du mémoire qui se subdivisera en quatre principaux chapitres :

Dans le premier chapitre intitulé « Données multidimensionnelle et cubes OLAP », nous commençons tout d'abord, par présenter les concepts d'entrepôts de données et des systèmes OLAP.

Puis, au sein de « Cloud Computing et L'écosystème Hadoop », deuxième chapitre de ce travail, nous définissons quelques concepts jugés nécessaire sur Cloud Computing, l'écosystème Hadoop et essentiellement un état de l'art dans ce domaine.

Au niveau de troisième chapitre intitulé «L'entreposage de données dans le Cloud», nous présentons notre problématique et décrivons notre principale contribution qui sera détaillé dans le chapitre suivant.

Finalement dans le dernier chapitre qu'on a nommé « Validation Et Expérimentation », nous présentons la configuration et les outils qui nous ont servi pour la construction de cube OLAP et les résultats obtenus sous forme des graphes.

Chapitre 1 :

Données multidimensionnelle et Cubes OLAP

1. Introduction

Dans ce chapitre, nous portons notre regard sur l'aide à la décision du point de vue des systèmes décisionnels au sens des entrepôts de données et de l'analyse en ligne OLAP. L'entrepôt permet de stocker les données d'une façon uniforme et agrégées afin d'apporter des réponses rapides et cube OLAP est exploré à l'aide de nombreuses opérations qui permettent sa manipulation.

2. Contexte de l'aide à la décision

Les systèmes d'information décisionnels sont utilisés pour faciliter l'accès, l'interrogation et l'analyse de l'information d'une organisation pour ses décideurs. La dernière évolution notable des SID repose sur les concepts d'entrepôt de données et d'OLAP. [1] [2]

Définition : Le *système décisionnel* est un système dédie au support de la prise de décision (pilotage). Il regroupe l'ensemble des outils informatiques permettant d'extraire et de transformer (E.T.L.), de stocker (S.G.B.D.), d'analyser et de restituer les données décisionnelles d'une organisation.

3. Entrepôts de Données

3.1. Définition

Le DW est généralement un élément essentiel du processus de l'ECD. Il est en amant du processus de l'ECD dans les systèmes d'aide à la décision. Selon la Définition de W H Inmon :

« Le Data WareHouse est une collection de données orientées sujet, intégrées, agrégées, non volatiles et historisées, organisées pour le support d'un processus d'aide à la décision ». [3]

3.2. Caractéristiques des données d'un DW

Orientées sujet : un DW rassemble et organise des données pertinentes pour des sujet ou des thèmes principaux (production, achat, vents….) afin de fournir une vue

simple et concise à propos de chaque sujet ou thème pour l'aide à la décision et aux besoins d'analyse.

Intégrées : les données résultent de l'intégration de données provenant des bases des données transactionnelles et qui sont hétérogènes de point de vue donnée et schémas des données. Ainsi ces données seront homogénéisées et leurs donner un schéma et sens uniques.

Historisées : les données d'un DW représentent l'activité d'une entreprise durant une certaine période (plusieurs années) permettant d'analyser les variations d'une donnée dans le temps.

Non-volatiles : les données d'un DW sont essentiellement utilisées en interrogation (consultation) et ne peuvent pas être modifiées et supprimées (sauf certain cas de rafraîchissement).

Agrégées : L'agrégation est le problème majeur lié à la construction des DWS car elle permet de résumer les données opérationnelles élémentaires à un niveau de détail plus élevé. Ce niveau plus élevé devient le niveau élémentaire dans le DW. L'agrégation est utilisée afin de définir la granularité des données stockées. Le choix de la granularité influence de manière significative la pertinence des données contenues dans le DW, et dans le même temps, affecte le type des requêtes possibles. [3]

3.3. Le rôle du DW

Le rôle primordial d'un data Warehouse apparaît ainsi évident dans une stratégie descensionnelle. L'alimentation du data Warehouse en est la phase la plus critique, En effet, importer des données inutiles apportera de nombreux problèmes. Cela consommera des ressources système et temps. De plus, cela rendra les services d'analyses plus lents. Comme nous l'avons indiqué, le Data Warehouse est le centre de la chaîne décisionnelle.

3.4. Architecture des entrepôts de données :

L'entrepôt de données joue un rôle stratégique dans la vie d'une entreprise. Il stocke des données pertinentes aux besoins de prise de décision en provenance des

systèmes opérationnels de l'entreprise et d'autres sources externes. A la différence d'une base de données classique supportant des requêtes transactionnelles de type OLTP (On-Line Transaction Processing), un entrepôt de données est conçu pour supporter des requêtes de type OLAP (On-Line Analytical Processing). L'interrogation est l'opération la plus utilisée dans le contexte d'entrepôt de données où la mise à jour consiste seulement à alimenter l'entrepôt.

Le processus de construction d'un entrepôt de données est composé de trois principales phases :

1- extraction des données à partir des différentes sources.
2- organisation et intégration des données dans l'entrepôt.
3- accès aux données intégrées et analyse de ces dernières dans une forme efficace et flexible. Dans la première et la deuxième phase, les données issues de différentes sources de données sont extraites, nettoyées et intégrées dans l'entrepôt de données. Les métadonnées contiennent des informations utiles sur la création, l'utilisation et la gestion de l'entrepôt. Durant la troisième phase, un serveur OLAP se charge de présenter les informations demandées par les utilisateurs sous plusieurs formes : tableaux, rapports, statistiques. [6]

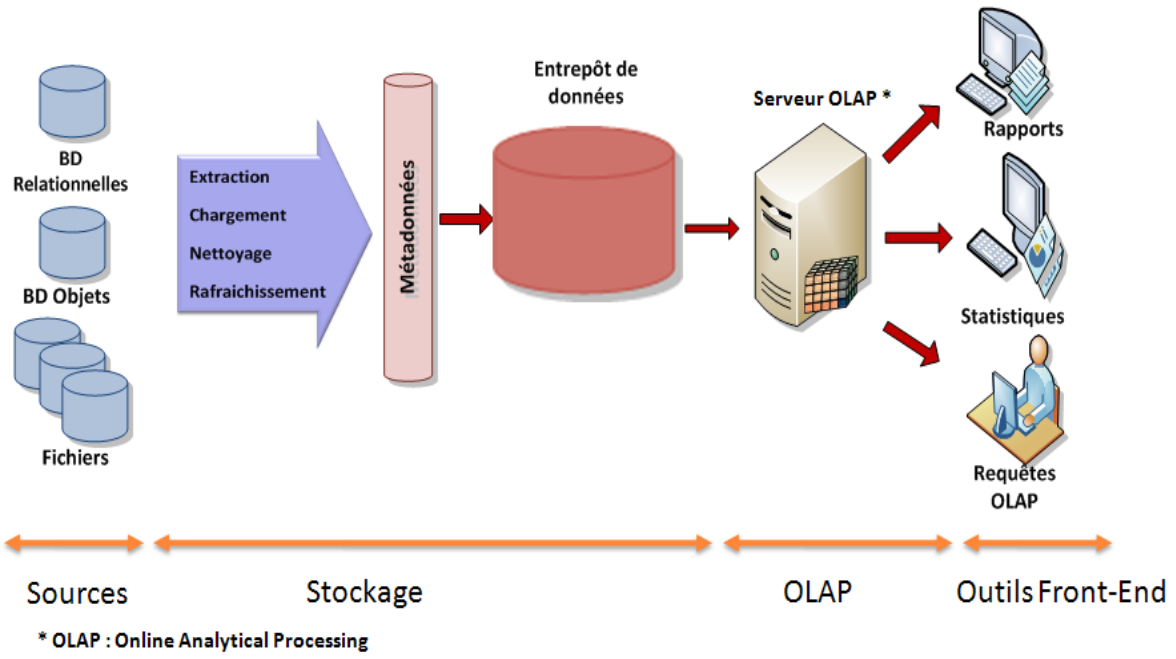

Figure 1.1 : Architecture d'un Entrepôt de données.

3.5. Différence entre SGBD et entrepôt de données

Les SGBD et les entrepôts ont des objectifs différents et réalisent des traitements différents. Ils stockent des données différentes. Ils font des requêtes différentes. Ils s'appuient des organisations de données différentes.

Les SGBD sont des systèmes dont le mode de travail est transactionnel (OLTP). Ils permettent d'insérer, modifier, interroger des données rapidement, efficacement, en sécurité, ils sont conçus pour optimiser l'espace de stockage.

Les entrepôts de données (DW) sont des systèmes conçus pour l'aide à la décision (OLAP). Ils permettent d'analyser des données provenant de sources diverses, ils sont conçus pour optimiser le temps de réponse aux différentes requêtes.

Les principales différences entre les SGBD et les entrepôts de données sont présentées dans le tableau suivant. [4]

Caractéristiques	SGBD	Entrepôt de données
Fonction	Gestion courante, production	Analyse, aide à la décision
Modèle de donnée	Entité-association	Etoile, flocon, constellation
Normalisation	Fréquente	Plus rare
Mise à jour	Actuelles, brutes	Historisées, parfois agrégées
Niveau de consolidation	Faible	Elevé
Perception	Bidimensionnelle	Multidimensionnelle
Priorité	Sécurité et intégrité	Analyse et exploration
Optimisation	Espace de stockage	Temps de réponse
Opérations	Lectures, mises à jour, suppressions	Lectures, analyses croisées
Taille	En giga-octets	En téraoctets

Tableau 1.1 : Comparaison entre SGBD et DW. [4]

3.6. Différence entre OLTP et OLAP

3.6.1. OLTP

Les systèmes transactionnels sont centrés sur la mise à jour des BDs opérationnelles. Les applications OLTP automatisent les opérations quotidiennes des entreprises et sont caractérisées par des simples et courtes transactions simultanées contre une grande base de données partagées. Les modèles OLTP créent une vue relationnelle pour l'enregistrement et le suivi des activités de l'entreprise.

3.6.2. OLAP

Les systèmes OLAP mettent en œuvre des technologies permettant de rassembler, gérer, traiter et présenter des données multidimensionnelles à des fins d'analyse et de décision. [5]

Le Tableau décrit une comparaison entre ces deux modèles.

Items	OLTP	OLAP
Nature	Transactionnel	Décisionnel
But	Enregistrement et suivi	Analyse et décision
Utilisation	Quotidienne	Aléatoire
Modèle de donnés	Relationnel (Entités / Relations)	Multidimensionnel (Fait / Dimensions)
Données	Détaillées	Détaillés, agrégées
Requêtes	Simples interrogeant un petit volume de données.	Complexes interrogeant un grand volume de données.

Tableau 1.2 : Comparaison entre OLTP et OLAP. [5]

3.7. Modélisation des entrepôts de données :

3.7.1. Modélisation multidimensionnelle

La modélisation multidimensionnelle permet d'observer un sujet analysé comme un point dans un espace à plusieurs dimensions. Les données sont organisées d'une manière qui met en évidence le sujet en cours d'analyse et ses différentes perspectives d'analyse. Ainsi, ce modèle a donné naissance aux concepts de fait et de dimension. [7]

Un schéma multidimensionnel noté E est défini par (F^E, D^E, $Star^E$) où :

$F^E = \{F_1 \ldots. F_n\}$ un ensemble fini de faits.

$D^E = \{D_1 \ldots. D_m\}$ un ensemble fini de dimensions.

$Star^E : F^E \rightarrow 2^{D^E}$ est une fonction qui associe les faits de F^E à des ensembles de dimensions, selon lesquelles ils peuvent être analysés (2^{D^E} étant l'ensemble des parties de l'ensemble D^E).

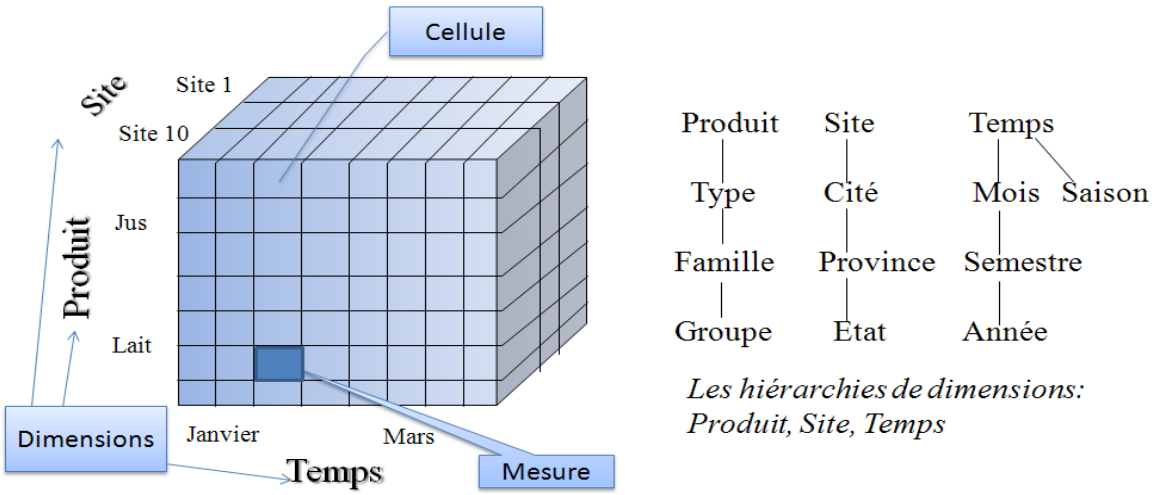

Figure 1.2 : Vue multidimensionnelle des données. [2]

3.7.1.1. Fait, Dimension et hiérarchie

Un fait : représente le sujet analysé. Il est formé de mesures qui correspondent aux informations liées au thème analysé. Les mesures sont stockées dans des tables de faits qui contiennent les valeurs des mesures et les clés vers les tables de dimensions.

Un fait F_j est défini par $(N^{Fj}, M^{Fj}, I^{Fj}, IStar^{Fj})$ où N^{Fj} est le nom du fait.

$M^{Fj} = \{m_1, \ldots, m_w\}$ est un ensemble de mesures.

$I^{Fj} = \{I_1^{Fj}, I_2^{Fj}, \ldots\}$ est l'ensemble des instances de F.

$IStar^{Fj}$ est une fonction associant chaque instance de I^{Fj} à une instance de chaque dimension liée au fait.

Une dimension :est un axe d'analyse c'est-a-dire une base sur laquelle seront analysées les données.

Une dimension D_i est définie par $(N^{Di}, A^{Di}, H^{Di}, I^{Di})$ où N^{Di} est son nom.

$A^{Di} = \{a_1^{Di} \ldots, a_u^{Di}\}$ représente les attributs.

$H^{Di} = \{h_1^{Di}, \ldots, h_y^{DI}\}$ représente les hiérarchies.

$I^{Di} = \{I_1^{Di}, I_2^{Di}, \ldots\}$ est l'ensemble des instances de Di.

Une hiérarchie : organise les données (membres) dans une structure hiérarchique permettant aux décideurs d'analyser les mesures à différentes granularités.

Une hiérarchie est un chemin élémentaire acyclique débutant par l'attribut de plus forte granularité (All) et se terminant par celui de plus faible granularité (Id).

Une hiérarchie est définie par $(N_x^{Di}, Param_x^{Di}, suppl_x^{Di})$ où N_x^{Di} est son nom.

$Param_x^{Di} = < a_{k0}^{Di}, \ldots, a_{kz}^{Di} >$ est un ensemble ordonné décrivant la hiérarchie des attributs (chaque attribut, appelé paramètre, correspond à un niveau de granularité d'analyse), $a_{k0}^{Di} = All\ et\ a_{kz}^{Di} = Id, et\ suppl_x^{Di} \rightarrow 2^{A^{Di} - Param_x^{Di}}$ est une application spécifiant les attributs faibles qui complètent la sémantique des paramètres (chaque paramètre est associé à un ensemble d'attributs faibles). [8] [9] [10] [11]

3.7.1.2. Modèles multidimensionnelle

Un ED peut être représenté par différents modèles conceptuels. Ces modèles ont été créés et développés par Kimball et al : [12]

3.7.1.3. Modèle en étoile (star schéma) :

Ce modèle est composé d'une seule grande table de faits relie par des tables de dimensions.

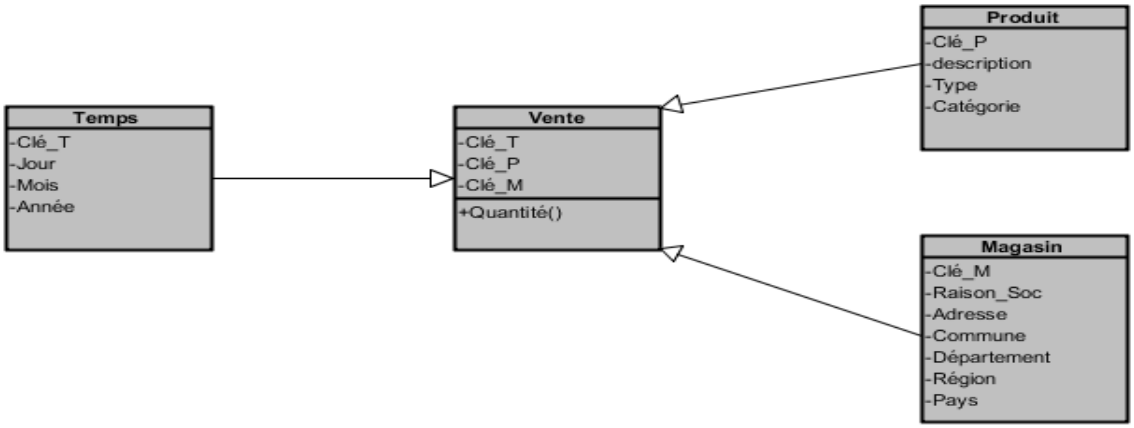

Figure 1.3 : Exemple de modélisation en étoile.

3.7.1.4. Modèle en flocon de neige (snow flake schema) :

Ce modèle est une première variante du modèle en étoile. Il consiste à décomposer les dimensions d'un modèle en étoile en des hiérarchies explicites. Cette modélisation permet de réduire le volume de stockage et autorise des analyses par paliers sur la dimension hiérarchisée. [13]

Les hiérarchies pour le schéma en flocon de neige sont :

Dimension temps : jour ➔mois ➔ année

Dimension magasin : commune➔département➔région➔pays

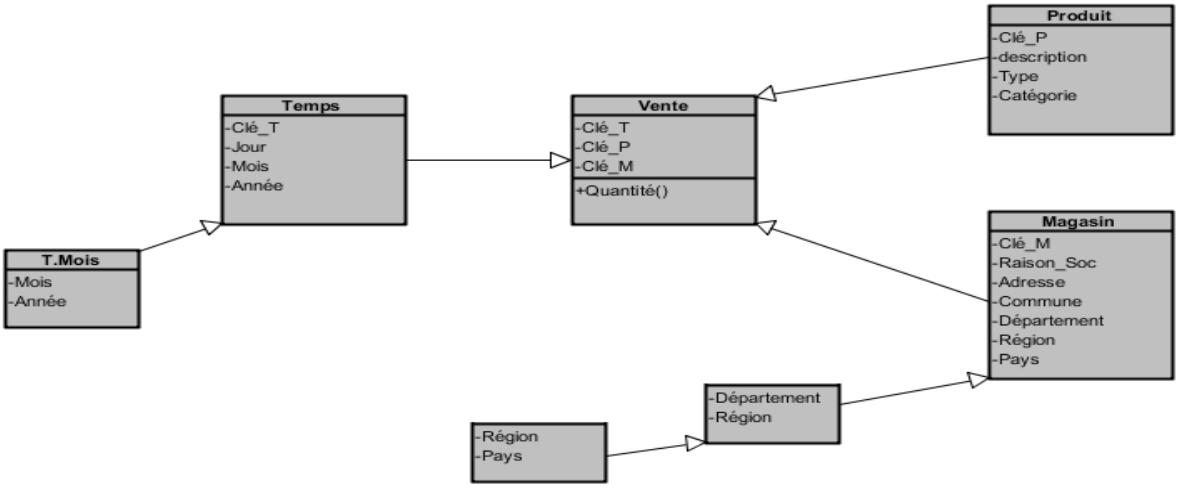

Figure 1.4 : Exemple de modélisation en flocon.

3.7.1.5. Modèle en constellation :

Consiste à fusionner plusieurs modèles en flocons, permettant le partage de certaines dimensions par plusieurs ensembles de faits. [13]

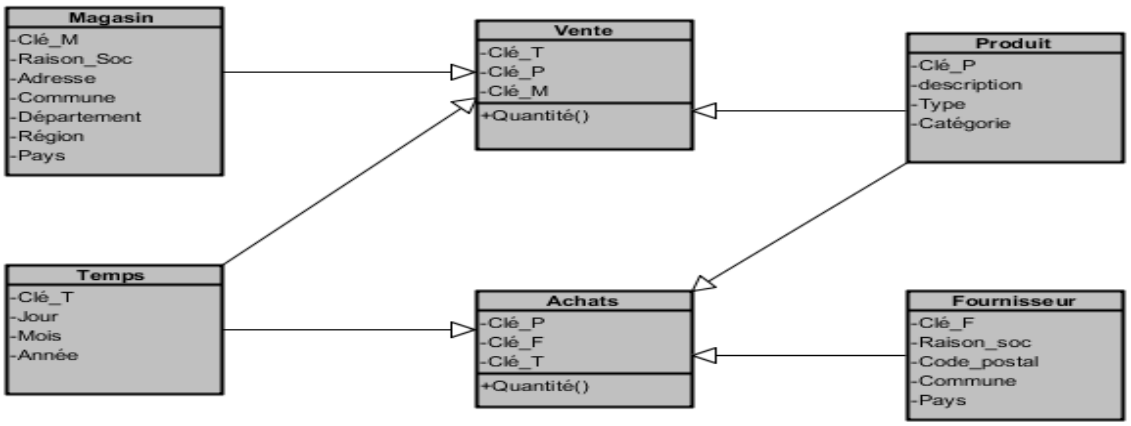

Figure 1.5 : Exemple de modélisation en constellation.

Les trois modèles précédents créent une vue multidimensionnelle afin de faciliter aux gestionnaires la consultation et l'analyse des données de façon transversale. Ils sont utilisés dans les applications de traitement analytique en ligne (OLAP).

4. Approche OLAP

4.1. OLAP (On-line Analytical Processing)

L'analyse en ligne constitue un autre aspect du processus d'entreposage des données. Codd (1993) a défini l'OLAP comme "*l'analyse dynamique d'une entreprise qui est requise pour créer, manipuler, animer et synthétiser l'information des modèles d'analyse de données. Cela inclut la capacité à discerner des relations nouvelles ou non anticipées entre les variables, la capacité à identifier les paramètres nécessaires pour traiter des grosses quantités de données, la création d'un nombre illimité de dimensions*".

L'acronyme FASMI (Fast Analysis of Shared Multidimensional Information, Ce terme traduit par : analyse rapide d'information multidimensionnelle partagée) permet de résumer la définition des produits OLAP. Cette définition fut utilisée pour la première fois en 1995 et depuis aucune autre définition n'est plus proche pour résumer le terme OLAP. [Selon nigel pendse et richard creeth]. [15]

1. Fast : temps de réponse aux demandes des utilisateurs entre 1 et 20 secondes : utilisation dans les produits OLAP de pré-calculs pour réduire les durées des requêtes.
2. Analysis : faire face à toutes les logiques d'affaire et de statistiques, ainsi que fournir la possibilité aux utilisateurs de construire leurs calculs et leurs analyses sans avoir à programmer : outils fournis avec les produits OLAP.
3. Shared : le système doit créer un contexte où la confidentialité est préservée et doit gérer les cas où plusieurs utilisateurs ont des droits en écritures (plutôt une faiblesse des produits OLAP actuels).
4. Multidimensional : caractéristique majeure, les produits OLAP doivent fournir des vues conceptuelles multidimensionnelles des données et supporter des hiérarchies de dimensions.
5. Informations : ensemble des données et les informations nécessaires pour un produit OLAP.

Définition : Un *système OLAP* est défini comme un système décisionnel dans lesquelles magasins de données suivent une organisation multidimensionnelle des données afin d'assurer un support efficace pour les analyses OLAP.

4.2. Cube OLAP

4.2.1. Définition

Cube OLAP : est une structure de données supérieure aux bases de données relationnelles grâce à une analyse rapide des données. Les cubes peuvent afficher et additionner de grandes quantités de données, tout en permettant de parcourir le contenu des points de données. De cette manière, les données peuvent être cumulées et découpées selon vos besoins. [16]

4.2.2. Caractéristiques des cubes OLAP

Les cubes OLAP ont les caractéristiques suivantes :

1- obtenir des informations déjà agrégées selon les besoins de l'utilisateur.

2- simplicité et rapidité d'accès

3- capacité à manipuler les données agrégées selon différentes dimensions

4- un cube utilise les fonctions classiques d'agrégation : min, max, count, sum, avg, mais peut utiliser des fonctions d'agrégations spécifiques. [17]

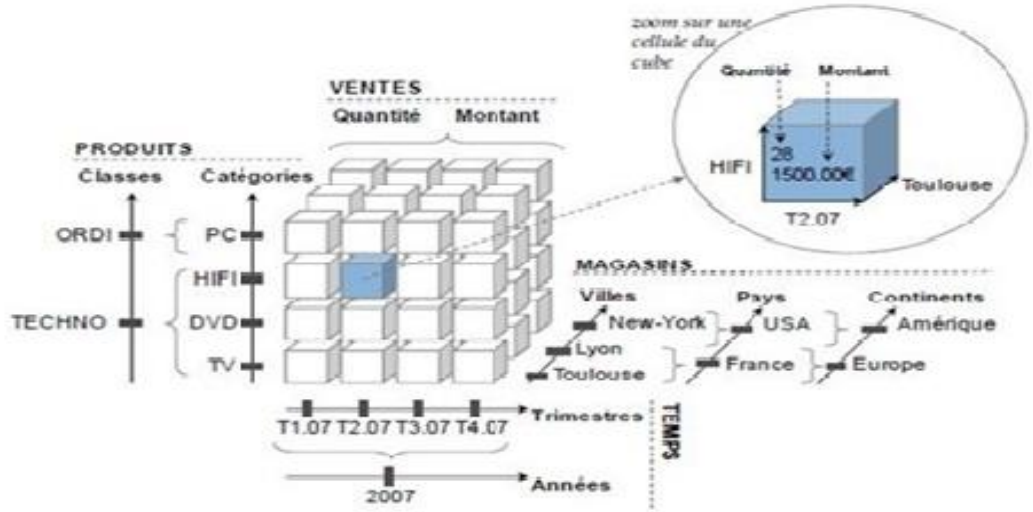

Figure 1.6 : Cube de données. [2]

4.3. Modèles logiques pour les données

L'implémentation du modèle multidimensionnel sur un SGBD réel peut se faire selon deux modèles : MOLAP (Multidimensional On-Line Analytical Processing) et ROLAP (Relational On-Line Analytical Processing). [18]

4.3.1. Les systèmes ROLAP :

Les systèmes de type ROLAP ("*Rolational On-Line Analytical Processing*") utilisent un SGBD relationnel pour stocker les données de l'entrepôt. Le moteur OLAP est un élément supplémentaire qui fournit une vision multidimensionnelle de l'entrepôt, des calculs de données dérivées et des agrégations à différents niveaux. Il est aussi responsable de la génération des requêtes SQL mieux adaptées au schéma relationnel, qui bénéficient des structures d'optimisation existantes pour exécuter efficacement ces requêtes. Ces systèmes peuvent stocker de grands volumes de données, mais ils peuvent présenter un temps de réponse élevé. Leurs principaux avantages sont la facilité d'intégration dans les SGBDs relationnels existants et une bonne efficacité pour stocker les données multidimensionnelles.

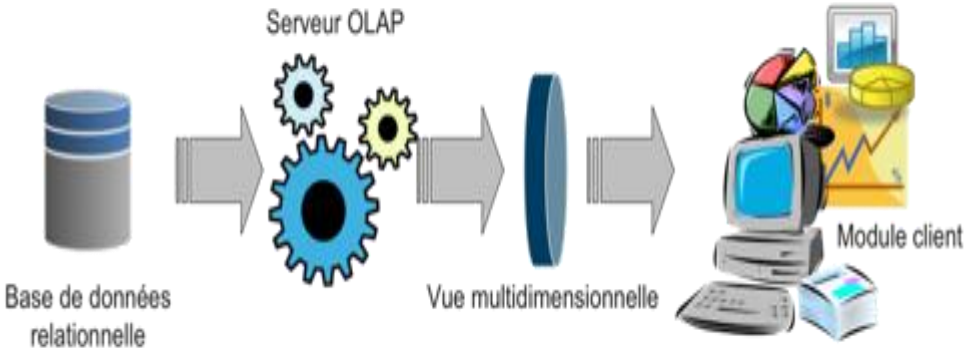

Figure 1.7 : Les systèmes de type ROLAP. [17]

4.3.2. Les systèmes MOLAP :

Les systèmes multidimensionnels OLAP utilisent une base de données multidimensionnelle pour stocker les données de l'entrepôt de et les applications analytiques sont construites directement sur elle. Dans cette architecture, le système de base de données multidimensionnelle sert tant au niveau de stockage qu'au niveau de gestion des données. Les données des sources sont conformes au modèle multidimensionnel, et dans toutes les dimensions, les différentes agrégations sont pré-calculées pour des raisons de performance.

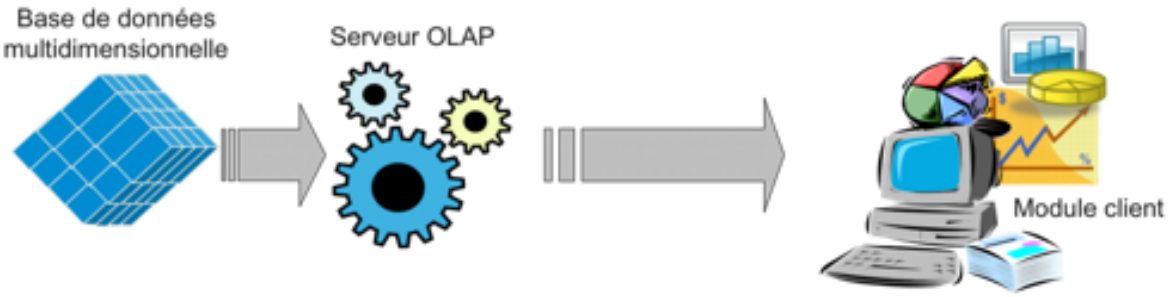

Figure 1.8 : Les systèmes de type MOLAP. [17]

4.3.3. Les systèmes HOLAP :

Un système HOLAP est un système qui supporte et intègre un stockage des données multidimensionnel et relationnel d'une manière équivalente pour profiter des caractéristiques de correspondance et des techniques d'optimisation. Cette architecture est un croisement des architectures ROLAP et MOLAP. Les données détaillées sont stockées dans une base de données relationnelle tandis que les données agrégées sont mémorisées dans une base de données multidimensionnelle. Le serveur accède aux deux bases de données et présente les données, sous forme d'une vue multidimensionnelle, au module client.

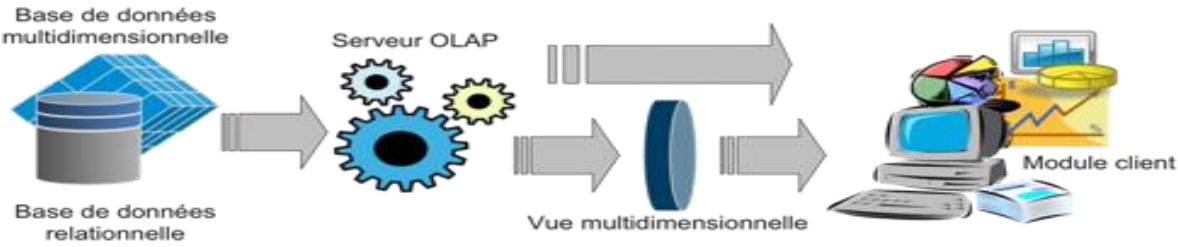

Figure 1.9 : Les systèmes de type HOLAP. [17]

4.3.4. Différence entre ROLAP, MOLAP, HOLAP

	ROLAP	MOALP	HOLAP
tockage des données de base	BD relationnelle	BD multidimensionnelle (hybercube)	BD relationnelle
tockage des agrégations	BD relationnelle	BD multidimensionnelle (hybercube)	3D multidimensionnelle (hybercube)
Structure de la BD	odèle particulier (étoile, flocon, etc)	Structure propriétaire au logiciel utilisé	Croisement des architectures ROLAP et MOLAP

Fonctionnement	Le serveur extrait les nnées par des requêtes SQL et interprète les données selon une vue multidimensionnelle vant de les présenter au module client	Le serveur MOLAP extrait les données de l'hypercube et les présentes directement au module client.	Accédé aux deux BD et les présentes au module client selon leur méthode respective.
Performance des requêtes	Le moins performent	Le plus performant	Performance moyenne

Tableau 1.3 : Différence entre ROLAP, MOLAP, HOLAP. [11]

5. Manipulation des cubes de données

Afin d'explorer un cube, l'utilisateur a besoin d'opérations et d'un langage de manipulation de données. Dans le domaine des bases de données relationnelles, il existe des opérations relationnelles et le langage standard est le langage SQL. Nous présentons dans cette section les opérations propres au requêtage multidimensionnel ainsi qu'un langage approprié.

5.1. Langages Implémentant des Opérations OLAP

Il existe principalement deux approches :

Extension du langage SQL (SQL-OLAP), pour un cube stocké dans un SGBD relationnel, sous la forme de schéma en étoile ou en flocon. On parle de modèle ROLAP ou Relational-OLAP.

Expression multidimensionnelle (ex : MDX de Microsoft) s'appliquant directement à un SGBD multidimensionnel. On parle de modèle MOLAP ou Multidimensional-OLAP.

5.2. Les requête OLAP

Les requêtes OLAP permettent l'analyse interactive d'un gros volume de données multidimensionnelles à l'aide d'opérateurs spécifiques de consolidation et d'agrégation pour résumer l'information et de présentation et structuration pour

naviguer et explorer l'information. Ces données sont organisées sous forme de cubes de données et proviennent d'un entrepôt de données.
Un utilisateur accède aux données du cube à l'aide d'une requête OLAP dont la réponse est composée d'un ensemble des tuples de données détaillées ou agrégées, présentées sous forme de tableau croisé. Pour cela, on peut utiliser des langages spécifiques tels que MDX, HQL.

6. Conclusion

Ce premier chapitre a présenté les deux concepts Data WareHouse et OLAP ainsi que l'ensemble des outils informatiques permettant de traité, analysé et de restituer les données décisionnelles d'une organisation. Le chapitre suivant va présenter le rôle de l'environnement Cloud Computing et l'écosystème Hadoop au sein de l'entreprise et sa nécessité à nos jours.

Chapitre 2 :

Cloud Computing

Et

L'écosystème Hadoop

1. Introduction

Les technologies sont des phases essentielles dans le cycle de vie d'une entreprise. Parmi ces technologies une infrastructure (Cloud) composée d'un grand nombre de ressources virtualisées (par exemple : réseaux, serveurs, stockage, applications ou services), Cette philosophie n'est pas nouvelle car John McCarthy ont proposé en 1961. L'arrivé de Hadoop, basé sur le paradigme MapReduce et Google File System permettant de structurer un projet et de manipuler de manière plus simple et effectuer traitement parallèle de données évolutif du travail que l'on souhaite réaliser.

2. Cloud Computing

2.1. Définition

Le Cloud Computing s'appuie sur une infrastructure (le Cloud) composée d'un grand nombre de ressources virtualisées (par exemple : réseaux, serveurs, stockage, applications ou services), distribuées dans le monde entier. Ces ressources peuvent être allouées, puis relâchées rapidement, avec des efforts de gestion minimaux et avec peu d'interactions entre le client et le fournisseur. Aussi, cette infrastructure peut être dynamiquement reconfigurée pour s'ajuster à une charge de travail variable (passage à l'échelle). Finalement, les garanties de prestation offertes par l'informatique dans le cloud prennent typiquement la forme de contrats de niveau de service.

« Le Cloud Computing est un modèle d'accès à travers le réseau internet à un ensemble de ressources numériques, pouvant être allouées et libérées à la demande et pour lesquelles le fournisseur du service assure l'ensemble des activités de maintenance, de support et d'exploitation » [19]

2.2. Modèles de services de Cloud

Les offres de Cloud Computing se décomposent en trois familles de services : IaaS, PaaS et SaaS.

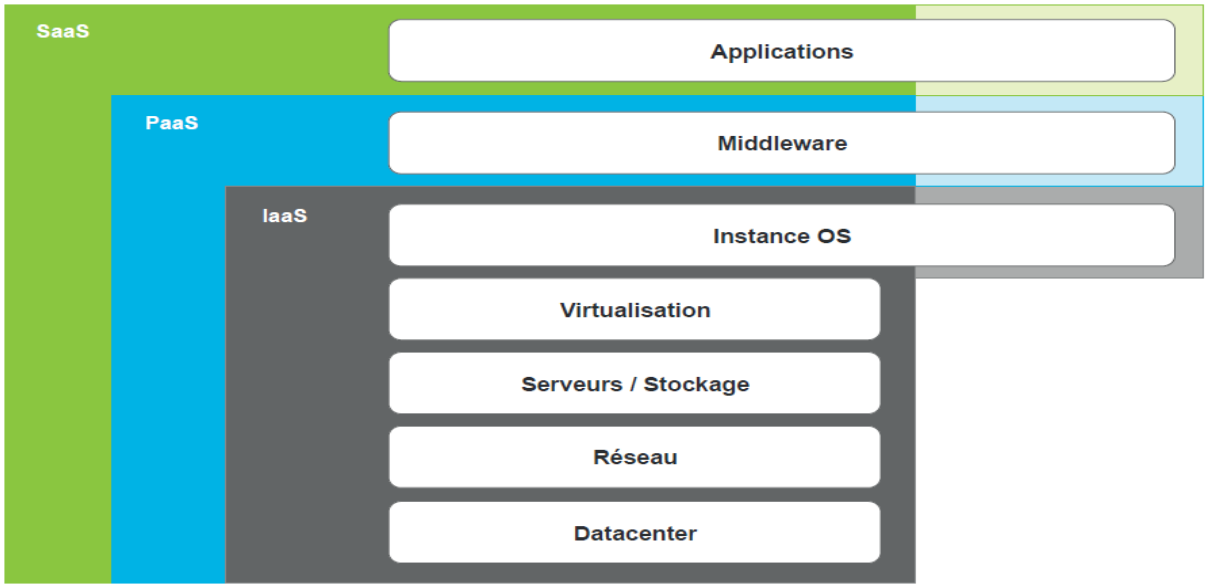

Figure 2.1 : Modèles de services de Cloud. [21]

2.2.1. IaaS (Infrastructure as a service)

L'IaaS ou Infrastructure as a Service consiste à fournir de l'infrastructure informatique (typiquement via une plateforme virtualisée) comme un service. Plutôt que d'acheter des serveurs, des logiciels, de l'espace pour le centre de données ou de l'équipement réseau, les clients acquièrent ces ressources informatiques comme un service complètement externalisé. Ce service est facturé en fonction du niveau de consommation de la ressource. C'est une évolution de l'hébergement web. [20]

2.2.2. PaaS (Platform as a service)

Le PaaS ou Plateforme as a Service consiste à fournir une plateforme informatique ainsi qu'une pile de solutions de développement, de test et d'exécution ; le tout étant consommé comme un service. Cela facilite le déploiement d'applications, en réduisant le coût et la complexité généralement associés à l'achat et à la gestion des couches de base du matériel informatique et des logiciels. Toutes les fonctionnalités requises pour soutenir le cycle de vie complet des applications sont fournies. [20]

2.2.3. SaaS (Software as a service)

Le SaaS ou Software as a Service est un modèle de déploiement de logiciels par lequel un éditeur offre à ses clients la licence d'utilisation d'une application à la

demande. Le logiciel est consommé comme un service. Les fournisseurs de logiciels en régime SaaS assurent soit l'hébergement de l'application sur leurs propres serveurs web, soit le téléchargement du logiciel sur l'environnement client (ils peuvent le désactiver après utilisation ou expiration du contrat). Les fonctionnalités à la demande peuvent être gérées en interne ou par un tiers fournisseur de services d'applications. [20]

2.3. Modèles de déploiement

Le concept de Cloud Computing est encore en évolution. On peut toutefois dénombrer trois types de Cloud Computing. [21]

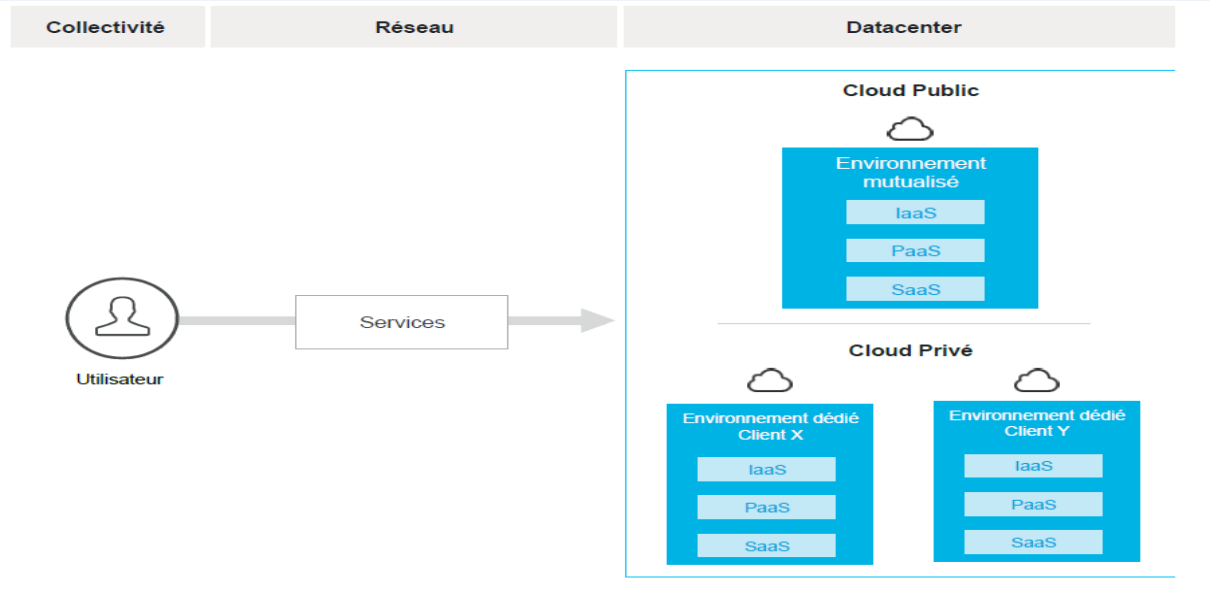

Figure 2.2 : Cloud Computing et modes de déploiement. [21]

2.3.1. Le Cloud privé

Le Cloud privé se distingue par son aspect dédié. Son usage est réservé pour une seule entreprise - toutes entités géographiques confondues - pour répondre à un besoin personnalisé de ressources informatiques bénéficiant en plus d'une tarification à l'usage et reposant sur des infrastructures techniques virtualisées, redimensionnables à chaud et disposant d'une gestion/administration automatisée.

2.3.2. Le Cloud public

Le Cloud public propose des ressources informatiques hébergées distantes et mutualisées. Il est externe à l'organisation, accessible via Internet, géré par un

prestataire externe propriétaire des infrastructures, avec des ressources partagées entre plusieurs sociétés. C'est le grand public qui a été mis au goût le Cloud public : Google, Hotmail, Facebook sont parmi les utilisations du Cloud les plus communes.

2.3.3. Le Cloud hybride

Cloud hybride (ou encore Cloud mixte) l'utilisation d'un Cloud privé et d'un Cloud public. Par exemple, un Cloud uniquement pour les données, et un autre pour les applications. Ce Cloud hybride permet d'allier les avantages du Cloud. C'est à dire d'une part, l'élasticité des infrastructures, les économies réalisées et la facilité de déploiement d'une application, avec d'autre part, la sécurité et dans certains cas la performance du système d'information interne.

2.4. Concepts lies au Cloud Computing

2.4.1. Clusters

Le Cloud Computing permet de construire un nuage de clusters, il permet la connexion entre un ensemble de machines sur un réseau bien défini. Par conséquent, les utilisateurs peuvent déployer des machines virtuelles dans ce nuage, ce qui leur permet d'utiliser un certain nombre de ressources. (Par exemple de l'espace disque, de la mémoire vive, ou encore du CPU). [22]

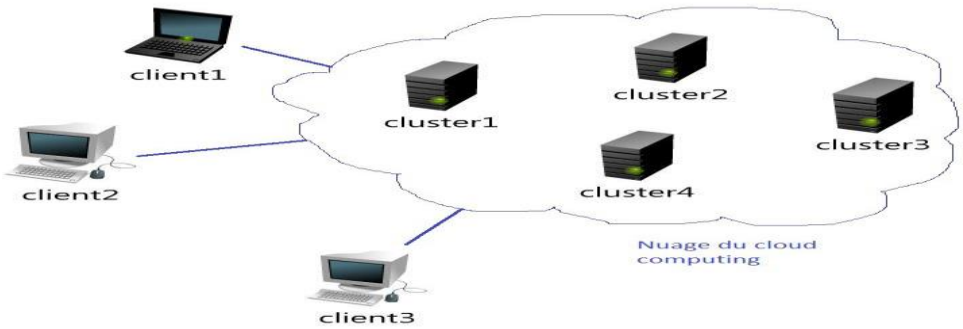

Figure 2.3 : Clusters Cloud. [24]

2.4.2. Nœuds

Un nœud au sens du Cloud peut être une machine ou une machine virtuelle. Il est caractérisé par une adresse propre.
Le nœud du Cloud est constitué par un système de virtualisation (VMM = Virtual Machine Monitor), adapté à l'infrastructure matérielle et qui gère le cycle de vie de plusieurs images logicielles. Une application logicielle bas niveau « node » gère également les interactions entre le gestionnaire du Cloud et l'hyperviseur (Scheduler) pour le lancement et l'arrêt d'images. [23]

2.4.3. La virtualisation

La virtualisation est un ensemble de techniques matérielles et/ou logicielles qui autorisent l'exécution de plusieurs applications indépendantes sur une même machine hôte. Grâce à la virtualisation, il est possible d'exécuter plusieurs systèmes d'exploitation (OS invité) sur un même serveur. Ainsi, il n'est plus nécessaire d'utiliser un serveur par application. On parle souvent d'environnement virtuel (*Virtual Environment*– VE) ou de serveur privé virtuel (*Virtual Private Server* – VPS) lorsqu'une machine exploite la virtualisation. Pour bénéficier de cette technologie, il suffit d'équiper une machine d'un logiciel de virtualisation permettant d'ajouter une couche de virtualisation, appelée hyperviseur. Ce hyperviseur masque les véritables ressources physiques de la machine afin de proposer des ressources différentes et spécifiques en fonction des applications qui tournent. Il y a donc une totale indépendance entre le matériel et les applications. Le logiciel de virtualisation simule autant de machines virtuelles que de systèmes d'exploitation souhaité. Chaque OS croit alors qu'il est installé seul sur une machine alors qu'en réalité, plusieurs OS peuvent fonctionner en parallèle en partageant les mêmes ressources. [24]

3. Hadoop

3.1. Le Framework Apache Hadoop

Le Framework Apache Hadoop a été créé par Doug Cutting et Michael J. Cafarella. Doug, qui travaillait avec Yahoo à l'époque, l'a nommé ainsi en référence

à l'éléphant avec lequel jouet son fils. Il a été développé à l'origine pour soutenir la distribution de Nutch, un moteur de recherche. [27]

Hadoop est un projet Open Source écrit en java basé sur le paradigme *MapReduce* et *Google File System*, ainsi que d'un certain nombre de projets connexes y compris Apache Hive, Apache Hbase, et d'autres. Il peut être considéré comme un système de traitement de données évolutif pour le stockage et le traitement par lot de très grandes quantités de données. [25]

La spécialité d'Hadoop, ce serait plutôt le traitement à très grande échelle de grands volumes de données non structurées tels que des documents textuels, des images, des fichiers audio… même s'il est aussi possible de traiter des données semi-structurées ou structurées avec Hadoop. [26]

Le schéma ci-dessous décrire ces quatre composants disponibles dans le Framework Hadoop.

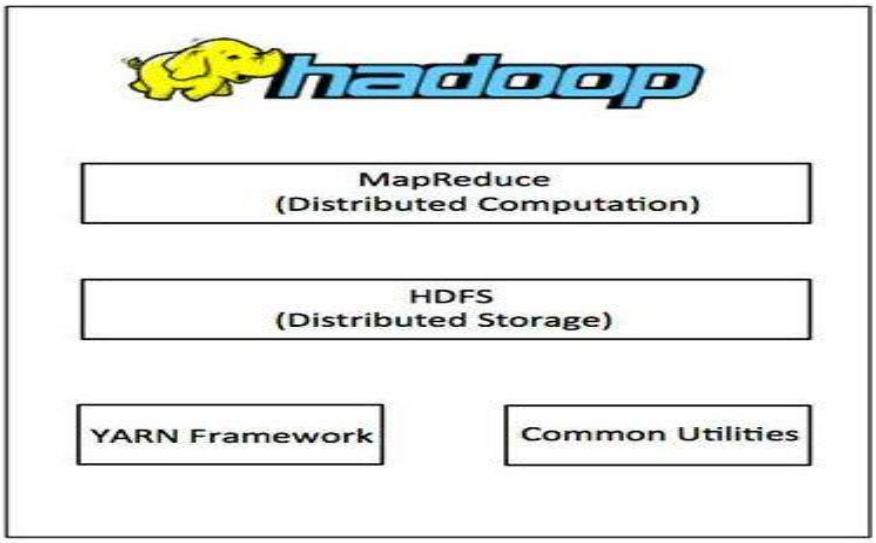

Figure 2.4: Framework Hadoop.

3.2. Architecture de Hadoop:

La plateforme Hadoop se compose des modules suivants:

3.2.1. Hadoop Common *:* contient les bibliothèques et les services publics nécessaires par modules Hadoop. [36]

3.2.2. Hadoop YARN *:* est un cadre pour la planification des tâches et la gestion des ressources de cluster. [36]

3.2.3. Hadoop Distributed File System (HDFS)

HDFS a été conçu pour stocker de très gros volumes de données sur un grand nombre de machines équipées de disques durs banalisés. Le file system HDFS est conçu pour assurer la sécurité des données en répliquant de multiples fois l'ensemble des données écrites sur le cluster.

HDFS fournit un système de réplication des blocs dont le nombre de réplications est configurable, Par défaut, chaque donnée est écrite sur trois nœuds différents.

Un cluster HDFS a le bénéfice d'offrir une solution de stockage très économique par rapport à celui des baies de stockage traditionnelles. En l'état, HDFS est optimisé pour maximiser les débits de données et non pas pour les opérations transactionnelles aléatoires. La taille d'un bloc de données est ainsi de 64 Mo dans HDFS contre 512 octets à 4 Ko dans la plupart des systèmes de fichiers traditionnels.

Pendant la phase d'écriture, chaque bloc correspondant au fichier est répliqué sur plusieurs nœuds. Pour la phase de lecture, si un bloc est indisponible sur un nœud, des copies de ce bloc seront disponibles sur d'autres nœuds. [26], [28]

3.2.3.1. Clusters Hadoop

Un cluster aura un serveur JobTracker, un serveur de NameNode, et un serveur de NameNode secondaire, et les nœuds de données (plusieurs DataNode) et les TaskTrackers. Le JobTracker coordonne les activités de la TaskTracker et le NameNode gère les DataNodes. [31]

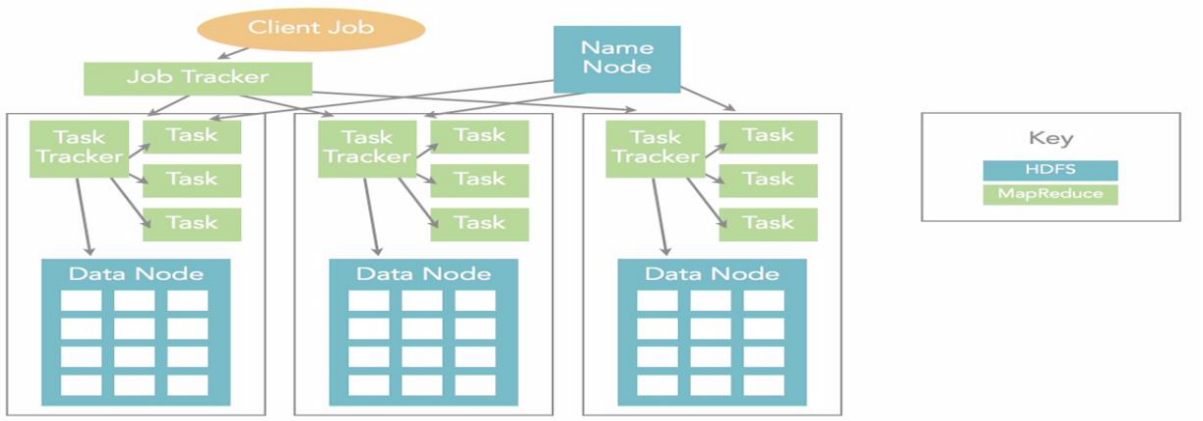

Figure 2.5 : cluster Hadoop.

3.2.3.2. NameNode

Un NameNode est un service central (généralement appelé aussi maître) unique sur le *cluster* qui s'occupe de gérer l'état du système de fichiers. Il maintient l'arborescence du système de fichiers et les métadonnées de l'ensemble des fichiers et répertoires d'un système Hadoop. Le NameNode a une connaissance des DataNodes dans lesquels les blocs sont stockés. [28]

3.2.3.3. Secondary NameNode

Secondary NameNode a été mis en place dans l'architecture Hadoop. Son fonctionnement est relativement simple puisque le NameNode secondaire vérifie périodiquement l'état du NameNode principal et copie les métadonnées via les fichiers edits_xxx et fsimage_xxx. Si le NameNode principal est indisponible, le NameNode secondaire prend sa place. [28]

3.2.3.4. DataNode

Le DataNode fournit des services de stockage de données pour le système de fichiers partagé. Les DataNodes sont sous les ordres du NameNode et sont surnommés les Workers. [31]

3.2.3.5. JobTracker

Le JobTracker, unique sur le *cluster*. Reçoit les tâches Map/Reduce à exécuter (sous la forme d'une archive Java.jar), organise leur exécution sur le cluster.

3.2.3.6. TaskTracker

Le TaskTracker, plusieurs sur le cluster. Exécute le travail Map/Reduce lui-même (sous la forme de tâches Map et Reduce ponctuelles avec les données d'entrée associées).

3.2.4. Hadoop MapReduce

MapReduce est un paradigme de programmation qui permet une scalabilité massive à travers une distribution de données et un traitement parallèle. Ce modèle a été introduit par Google en 2004, Après avoir prouvé son efficacité, il est utilisé par d'autres sociétés telles que Yahoo et Facebook. [29]

Dans ce modèle, le programmeur spécifie le traitement en deux étapes en utilisant une fonction Map() et une fonction Reduce(). Le système MapReduce, fonctionnant sur une plateforme de type cluster, parallélise alors automatiquement le traitement en découpant le processus en sous processus où chacun sera confié à un nœud. [30]

Fonction Map : La fonction Map s'écrit de la manière suivante : Map (clé1, valeur1) → List (clé2, valeur2). À partir d'un couple clé/valeur, la fonction Map retourne un ensemble de nouveaux couples clé/valeur.
Avant de présenter la fonction Reduce, deux opérations intermédiaires doivent être exécutées pour préparer la valeur de son paramètre d'entrée. La première opération appelée *shuffle* permet de grouper les valeurs dont la clé est commune. La seconde opération appelée *sort* permet de trier par clé.

Fonction Reduce : elle s'exécute après la fonction Map, le nœud maitre collationne les réponses emontant des nœuds de traitement et les combine afin de fournir la réponse à la question posée à l'origine.
La fonction Reduce s'écrit de la manière suivante : Reduce (clé2, List(valeur2)) → List(valeur2). [28]

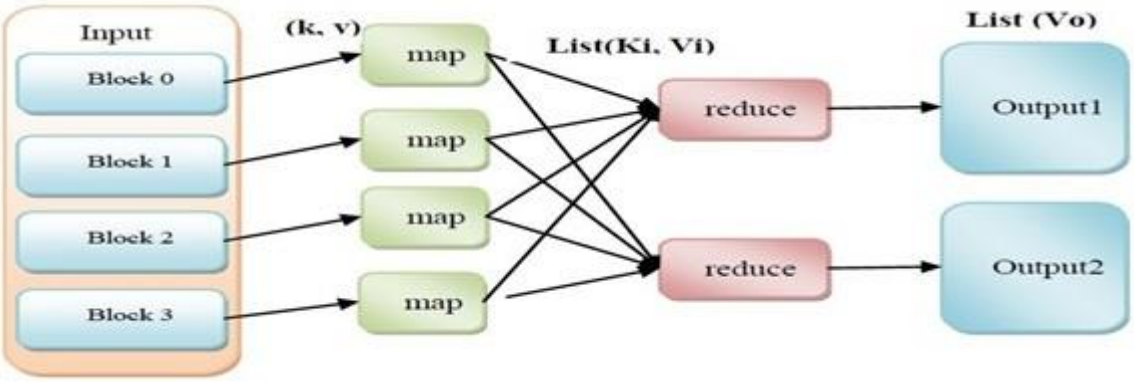

Figure 2.6 : Fonctionnement du modèle Map/Reduce. [32]

Fonction Combiner (Facultative)

Après l'exécution de la fonction de Map, s'il y a beaucoup paires clé-valeur avec la même clé, Hadoop doit se déplacer toutes les valeurs de la fonction Reduce. Cela peut encourir une surcharge importante. Pour optimiser ces scénarios, Hadoop prend en charge une fonction spéciale appelée combiner. Si fourni, Hadoop va appeler le combiner du même nœud que le nœud de Map avant d'invoquer le Reducer et après l'exécution du mapper. Cela peut réduire considérablement la quantité de données transférées à l'étape Reduce. [32]

3.2.5. L'écosystème Hadoop

La majorité des publications Hadoop aujourd'hui se concentrer soit sur la description des différents composants de cet écosystème, ou sur l'approche pour l'utilisation des outils d'analyse de l'entreprise(comme Pig and Hive) dans Hadoop.

Hive projet open-source conçu chez Facebook. Hive fournit un langage de requête haut niveau semblable à SQL nommé HiveQL, pour interagir avec un cluster Hadoop, dans le but réalisé des analyses sur une masse importante de données, permettant sélections, jointures, agrégats. Hive compile les requêtes en jobs MapReduce qui les exécute sur Hadoop. [37]

HiveQL est un langage d'interrogation semblable au SQL est utilisé pour organiser et questionner les données stockées dans Hive. Dans la version actuelle, HiveQL permet de créer et supprimer des tables et partitions, une table est subdivisée en parties multiples sur une clé de partition spécifique aussi bien que la requête avec

des déclarations de sélection ; UPDATE et DELETE ne sont pas encore supporté. Les fonctionnalités les plus importantes qui sont supporté à travers SELECT dans HiveQL sont : la possibilité de jointé des tables sur une clé commune pour filtrer des données utilisant des techniques de sélection des lignes et projeter des colonnes.

Pig a été conçu pour simplifier l'utilisation des API MapReduce exigeant de bonnes connaissances Java. Yahoo a créé la plateforme de traitement de données Pig et son langage Pig Latin (projet Apache). Efficace et simple, le langage Pïg Latin (proche du scripting) permet d'écrire des applications exécutées sur Hadoop (runtimePig) sans passer par MapReduce. Le développeur charge les données, y compris d'une source externe vers le cluster Hadoop, et les manipule. [37]

Sqoop est un outil de connectivité pour déplacer des données entre les bases de données relationnelles et des entrepôts de données et Hadoop. Sqoop exploite la base de données pour décrire le schéma pour les données importées/exportées et MapReduce pour le fonctionnement de parallélisassions et de tolérance aux pannes. [33], [34]

Impala est l'une des premières implémentations open source de systèmes temps réel. Impala exige actuellement une installation de Hadoop spécifique (CDH4.2 pour la dernière version d'Impala). Impala fonctionne sur les tables de Hive et utilise HiveQL comme un langage de requête. Les requêtes Impala est annulés si une défaillance matérielle est rencontrée lors d'une requête de longue durée. [33]

Spark est un autre modèle de calcul distribué qui peut fonctionné à la fois sur Hadoop. Spark a été développé à l'Université de Berkeley AMPLab et devient un projet Apache haut niveau en Février 2014. Le gros avantage de Spark sur les autres modèles parallèles est la performance et la facilité d'utilisation. Spark est annoncé comme fonctionnant jusqu'à 30 fois plus rapide que la charge de travail à base de MapReduce. Spark peut accéder à des ensembles de données stockées dans HDFS et Amazon S3. Spark est écrit principalement dans Scala et soutient Scala, Java et Python API. Scala, une autre langue qui fonctionne sur la machine virtuelle Java (JVM). [34]

Shark, une autre open-source alternative à Hive, est construit au-dessus de Spark Apache. Shark bénéficie autant que 100x améliorations de la performance sur Hive. [34]

3.2.6. Différents fournisseurs Hadoop

Cloudera CDH, Manager, and Enterprise:

Basé sur Hadoop2, CDH comprend HDFS, YARN, HBase, MapReduce, Hive, Pig, Zookeeper, Oozie, Mahout, Hue, et d'autres outils open source (y compris le moteur de recherche en temps réel-Impala). Cloudera Manager Edition gratuit comprend tous CDH, plus un gestionnaire de base prenant en charge jusqu'à 50 nœuds de cluster. Cloudera Enterprise combine CDH avec un gestionnaire plus sophistiqués ou tenant un nombre illimité de nœuds de cluster, une surveillance proactive, et des outils d'analyse de données supplémentaires.

Hortonworks Data Platform

Basé sur Hadoop2, cette distribution inclut HDFS, YARN, HBase, MapReduce, Hive, Pig, HCatalog, Zookeeper, Oozie, Mahout, Hue, Ambari, Tez, et une version en temps réel de Hive(Stinger) et d'autres outils open source. Fournit Hortonworks haute disponibilité support, une haute performance pilote ODBC Hive, et Talend Open Studio forBig Data.

MapR

Basé sur Hadoop 1, cette distribution inclut HDFS, HBase, MapReduce, Hive, Mahout, Oozie, Porc, ZooKeeper, Hue, et d'autres outils open source. Il comprend également un accès direct NFS, implémentation propriétaire HBase qui est entièrement compatible avec les API Apache, et une console de gestion de MapR.

4. Travaux connexes

Dans [Edgar Codd et al.], A la fin 1993, Edgar F. Codd publie un article « *Providing OLAP to User Analysts*» commandité par Arbor Software, dans lequel il énumérait douze règles qui permettaient à une technologie de faire de l'analyse sur les données.

1. Conception et vue multidimensionnelle : un outil OLAP doit se baser sur un modèle multidimensionnel pour faire de l'analyse.
2. Transparence : la technologie utilisée, la conception ainsi que toutes les spécifications techniques doivent être invisibles à l'utilisateur final.
3. Accessibilité : les outils OLAP doivent permettre d'accéder les données de façon à produire de la connaissance rapide. Une information pertinente et on-time doit être fournie en tout temps.
4. Rapidité : les montées de charges ne doivent pas freiner l'analyste. L'outil doit pouvoir supporter de grosses requêtes (c'est la caractéristique la plus difficile à satisfaire).
5. Architecture Client Serveur : pour un accès uniforme et des traitements plus rapides et plus sophistiqués.
6. Dimensions génériques : essayer, autant que possible d'avoir une unicité dans la définition des dimensions. Ne pas avoir deux dimensions client.
7. Gestion des matrices creuses : en mathématiques, les matrices creuses sont des matrices qui contiennent beaucoup de zéros. En informatique, il existe des algorithmes qui utilisent cette spécificité pour optimiser le stockage de ce type de matrices. Les performances sont, en général, au rendez-vous. Les outils OLAP doivent avoir cette capacité d'optimisation d'espace de stockage par la gestion des matrices creuses.
8. Multi-utilisateurs : les outils OLAP sont par définition destinés à un accès concurrent.
9. Croisement inter dimensions illimité : l'utilisateur ne doit avoir aucune restriction quant au nombre de croisements qu'il fait entre les dimensions.

10. Intuitifs : les utilisateurs d'outils OLAP ne sont pas forcément informaticiens. Il est donc nécessaire d'offrir des solutions adaptées à leur style cognitif.
11. Affichage flexible : l'utilisateur doit pouvoir aisément " arranger " son résultat au format désiré.
12. Nombre illimité de dimensions et de niveaux d'agrégation. [14]

Dans [Fadila Bentayeb et al.], ils ont présenté le domaine de l'aide à la décision au travers du prisme des entrepôts de données et de l'analyse en ligne OLAP. Ainsi, l'aide à la décision apparaît ainsi dans ce domaine comme la proposition de méthodes et d'outils permettant aux décideurs de naviguer dans les données consolidées dédiées à l'analyse. Dans cet article, ils proposent d'aborder les problématiques de recherche qui leur sont liées selon quatre points de vue : les données, les environnements de stockage, les utilisateurs et la sécurité. [13]

Dans [Hana Gara Kort et al.], La sécurité dans le Cloud Computing était le sujet de plusieurs travaux de recherche, ce domaine a connu un essor incomparable pendant les dernières années et s'est intéressé au développement des solutions de sécurité pour le Cloud. Il en est de même pour l'entrepôt de données. Mais peux de travaux pour la sécurité de l'entrepôt de données dans le cloud !

Ils proposent quatre scénarios de l'entrepôt de données à faire immigrer sur le Cloud Computing. [40]

Scénario 1: One out two in

Figure 2.7: *One out two in.* [40]

Scénario 2: Two out one in

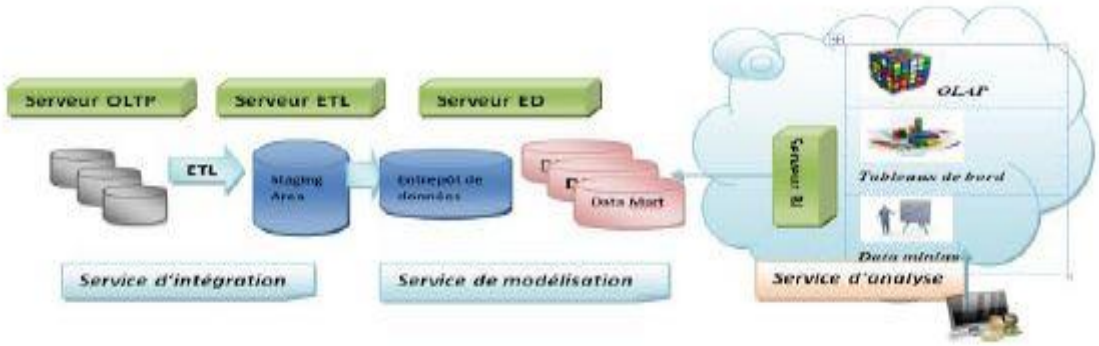

Figure 2.8: ***Two out one in.*** **[40]**

Scénario 3: One out one in one out

Figure 2.9: ***One out one in one out.*** **[40]**

Scénario 4: All in

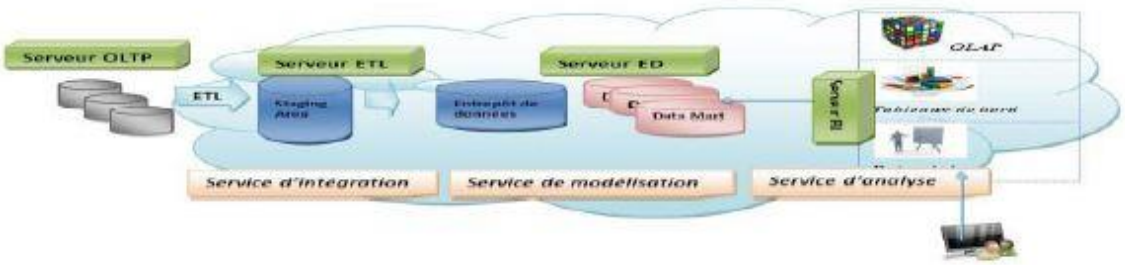

Figure 2.10: *All in*. [40]

Dans [Ashish Thusoo et al.], ils présentent Hive comme une solution d'entreposage de données open-source construit au-dessus de Hadoop. Hive fournit un langage déclarative HQL de plus haut niveau pour faciliter le traitement des données à grande échelle. Un langage proche du SQL, interprété en job MapReduce et exécuté sur la plateforme Hadoop. Cet article décrit le modèle de données Hive, la langue HiveQL, l'architecture du système Hive et Workflow d'une requête HQL. [38]

Dans [Alberto Abelló et al.], ils examinent la possibilité d'avoir des données dans un nuage en utilisant BigTable pour stocker les données historiques de l'entreprise et MapReduce comme un mécanisme agile pour déployer des cubes en ad-hoc des données Marts. Cette contribution est la comparaison des trois approches différentes *(Full Source Scan (FSS), Indexed Random Access (IRA), Index Filtered Scan (IFS))* pour récupérer des cubes de données de BigTable au moyen de MapReduce. Ils sont implémentée les trois algorithmes on hadoop et comparée les résultats obtenus. [39]

Dans [Billel ARRES et al.], L'arrivée d'*Hadoop*, basé sur le paradigme *MapReduce*, pour le traitement parallèle des grandes quantités de données, a permis de faciliter l'accès à un tel environnement. Ils proposent d'abord, de construire un entrepôt de donnée dans le *Cloud (Cloud Computing privé)*, et ensuite d'évaluer les performances de différentes variantes d'architectures de plateformes mise en place avec *Hadoop*. Ils ont opté d'utiliser un échantillon de données SSB, qui est un banc d'essai d'aide à la décision, conçu pour mesurer les performances

d'un entrepôt de données en étoile. Ils proposent de réaliser un tel environnement avec l'implémentation d'un entrepôt de donnés sous *Hive*. D'exploiter les fonctions *Map* et *Reduce* de cet environnement afin de comparer les résultats des temps de chargement de l'entrepôt de données et de construction d'un cube OLAP. Puis présenter les principaux résultats de l'étude qui permettent de comparer les performances de *Hive* entre un environnement virtuel et un environnement physique, en fixant le nombre de nœuds (1 nœud) et la taille de l'entrepôt de données. Puis le passage à l'échelle dans un environnement physique, avec la variation du nombre de nœuds (de 4 à 6 nœuds) ainsi que la taille de l'entrepôt de données (1Go à 1To). [25]

5. Conclusion

Nous avons présenté l'ensemble de technologies efficaces pour le stockage et le traitement par lot de très grandes quantités de données. Ainsi que les différents outils spécifiques de notre système. Le chapitre suivant présente notre problématique et notre principale contribution.

Chapitre 3 : L'entreposage de données dans le Cloud

1. Introduction

L'entreprise manipule et stocke un gros volume de données au sein de plusieurs différentes sources hétérogènes. Ces données sont utilisées quotidiennement et sont souvent inappropriées pour des besoins de prise de décision. Il devient alors impératif de collecter, intégrées, agrégées, historisées et d'homogénéiser ces données On a formé le soi-disant entrepôt de données qui s'adapter à l'analyse OLAP et de prise de décision. Apres l'émergence du DW, des systèmes OLAP sont peu à peu mis en place afin de permettre l'analyse des indicateurs pertinents.

L'arrivée d'Hadoop, basé sur le paradigme MapReduce, pour le traitement parallèle des grandes quantités de données, a permis de faciliter l'utilisation de ces systèmes, le traitement des données à haute échelle, a donné naissance à Cloud Computing qui permet le traitements haut performance.

2. Problématique

L'augmentation de volume de données pose un problème majeur au niveau de l'interrogation et l'analyse de ces données en termes de temps de réponse. Les bases de données relationnelles sont des systèmes dont le mode de travail est transactionnel (OLTP). Ils sont conçus pour optimiser l'espace de stockage, cependant les bases de données multidimensionnelles sont conçus pour optimiser le temps de réponse aux différentes requêtes, de nouvelles technologies ont été mis en place pour traiter cette obstacle comme Cloud, MapReduce, HQL ....

Il faut donc trouver un équilibre entre l'architecture de stockage et l'interrogation en parallèle pour cela nous considérons le système OLAP et MapReduce comme solution adéquate pour minimiser et réduire le temps de réponse, alors notre question se pose de la façon suivante comment pourrions-nous réduire le temps de construction d'un cube OLAP ?

Nous résumons le problème ainsi que notre objectif comme suit :

La plupart des entreprises dotées de bases de données multidimensionnelles utilisent le SQL pour l'interrogation de ces bases.

Cependant, le temps d'exécution accroit proportionnellement avec le volume de données. Pour réduire le temps d'exécution de requête.

Les entreprises sont entre deux solutions :

1. Utilisation des langages plus adapté ce qui obligent les entreprises à changer beaucoup de chose ceci rendre l'opération couteuse en termes du temps et de monnaie.
2. Rester avec les mêmes technologies notamment le SQL, et pensez à réduire le temps d'exécution.

Afin de répondre à cette question nous avons proposé le schéma suivant (Voir Figure 3.1).

3. Constructions d'un cube OLAP

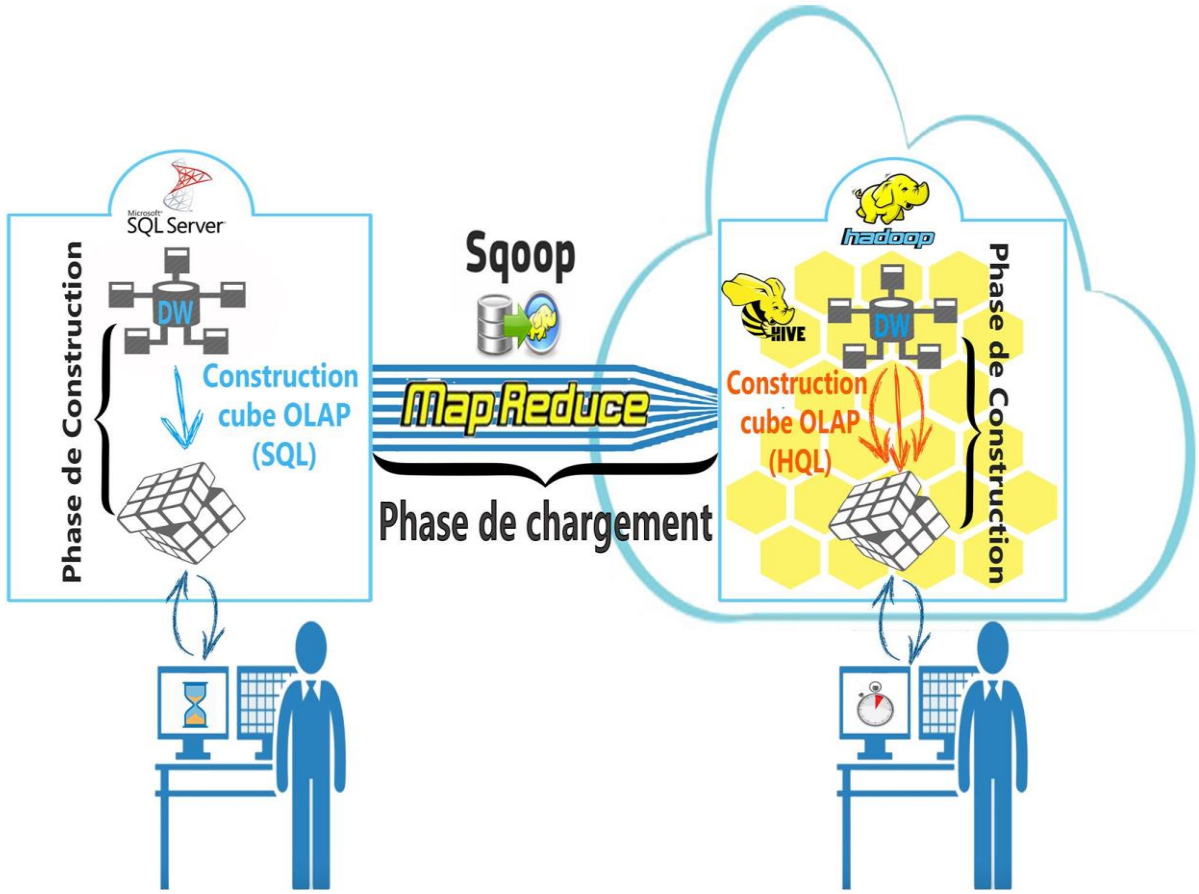

Figure 3.1 : Construction un cube OLAP Dans différentes environnements.

Les entrepôts de données et les systèmes d'analyse en ligne OLAP (On-Line Analytical Processing) fournissent des méthodes, des outils et des technologies d'aide à la décision qui permet l'analyse en ligne de gros volumes de données issues des systèmes d'information des entreprises.

Notre principale contribution consiste donc, dans un premier temps à construire un cube OLAP a l'aide de SQL SERVER à partir d'un entrepôt de données (*AdventureWorksDW*), Ensuite dans l'objectif de minimiser, optimiser et réduire le temps de construction de cube, à travers multiples nœuds distribués nous proposons de construire un cube OLAP avec MapReduce dans un environnement Cloud Computing en utilisant HiveQL comme langage d'interrogation plus performant que SQL, et enfin comparer les différentes résultats obtenus.

L'architecture suivante (Voir Figure 2) est une variante détaillée de la partie droite de la première architecture (Figure 1). Cette architecture permet d'importer l'entrepôt de données à travers Sqoop et le partitionnement de ces données sur des différents blocks. L'exploitation de l'*HDFS* pour gérer, stocker et répliquer de multiples fois l'ensemble des données résidant au niveau du cluster, ainsi nous avons utilisé HiveQl pour effectuer l'ensemble de requêtes de construction de cube OLAP, Notre requête décisionnelle sous Hive exécutée et compilé dans Job MapReduce et les fonctions MAP et REDUCE pour le traitement parallèle.

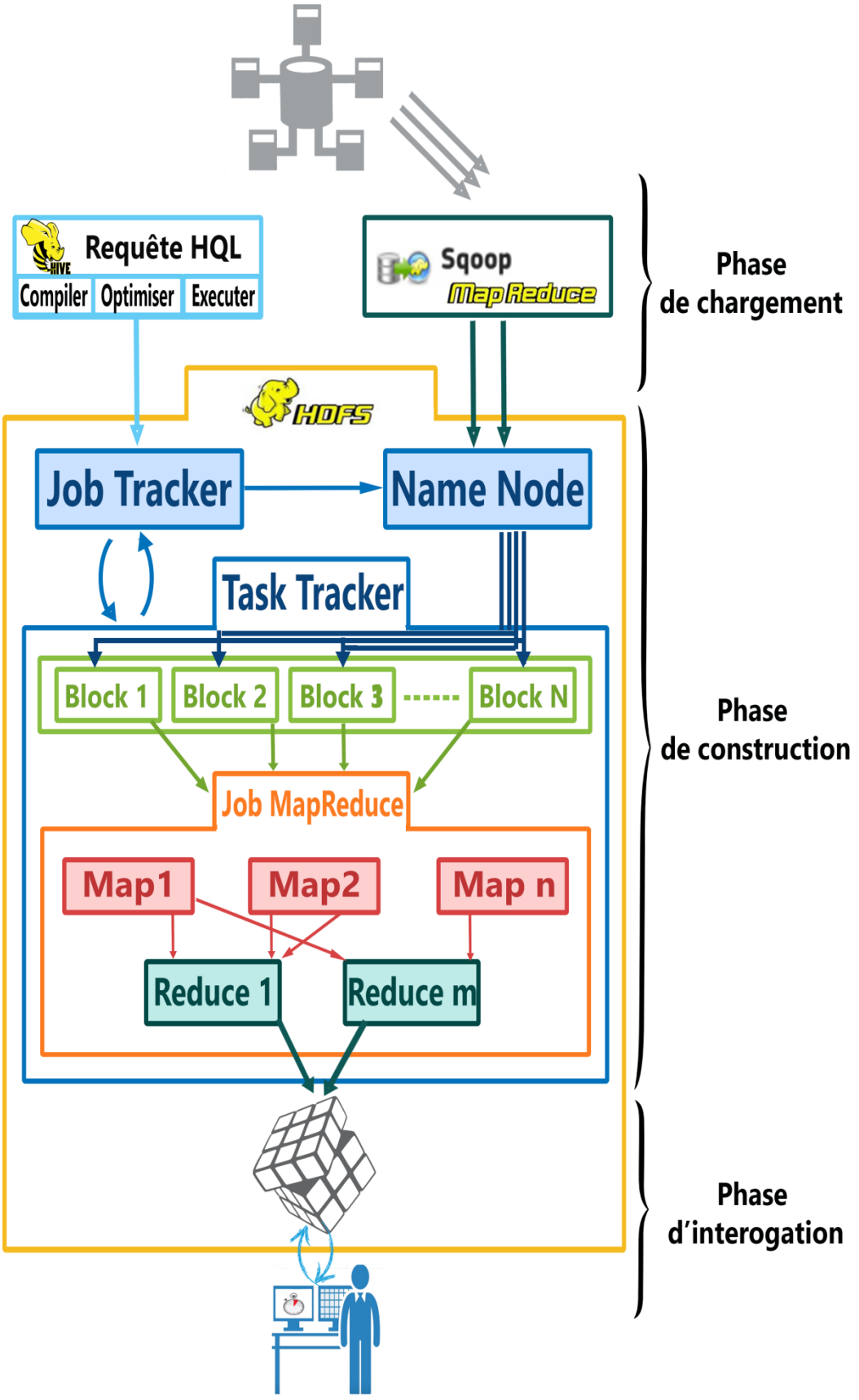

Figure 3.2 : Construction un cube OLAP sous Hive avec MapReduce.

3.1. Phase de construction d'entrepôt de données

Dans le but de garantir une meilleure accessibilité, interrogation et analyse de l'information d'une organisation, plusieurs architectures modernes ont privilégié d'agréger et centraliser les données de manière uniforme dans un entrepôt de données pour le support d'un processus d'aide à la décision.

AdventureWorksDW : est un entrepôt de données l'échantillon utilisé dans SQL Server pour démontrer les fonctionnalités de business intelligence qui sont disponibles dans SQL Server. Les données de l'entrepôt ont suivi une tendance à soutenir à la fois l'exploration de données et le traitement analytique en ligne (OLAP).

Notre entrepôt de données (AdventureWorksDW) est constitué d'une table de FAIT appelée FactInternetSales qui contient vingt-sept attributs (mesures) avec les clés vers les tables de dimensions. Et aussi six tables de dimension (DimCurrency, DimDate, DimProduct, DimPromotion, DimSalesTerritory et DimCustomer) (Voir Figure 3).

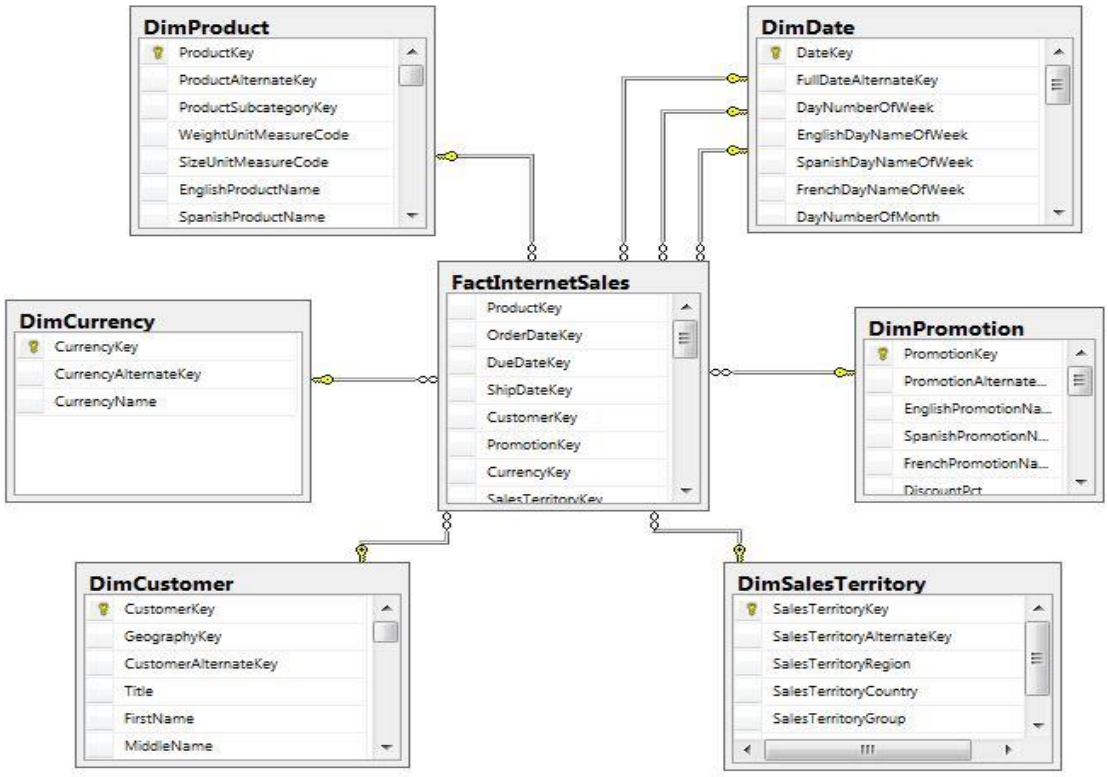

Figure 3.3 : Schéma de l'entrepôt de données.

3.2. Phase de chargement de l'entrepôt de données

On peut charger un entrepôt stocké dans un système de gestion de bases de données relationnelles sous Hive, en utilisant Sqoop comme un outil de connectivité pour déplacer des données ou des bases de données relationnelles et des entrepôts de données SQL Server vers Hadoop pour élaborer en HQL des requêtes de construction de cube OLAP.

3.3. Phase de construction de cube OLAP

Le Cube OLAP est constitué d'un ensemble de mesures organisées sous forme de dimensions. Chaque dimension représente un axe d'analyse sur lequel les données seront analysées.

OLAP et les entrepôts de données sont complémentaires, un entrepôt de données stocke et gère les données, OLAP transforme les données de l'entrepôt en informations stratégiques.

Nous avons choisi de construire un cube de données répondant à la requête décisionnelle suivante : *« Quelle est la somme des revenues des ventes par année et par marque pour la région Europe depuis 2006 ? »*. On cherche ici la somme des revenus (REVENUE dans *FactInternetSales*) comme mesure selon les trois dimensions *Produit* et *Date* (Période) et *SalesTerritory* (région).

Les trois requêtes de construction du cube OLAP en HiveQL :

Première requête:

```
select * from (

select sum(l.UnitPrice) as revenu, d.CalendarYear as ANNEE, p.EnglishProductName as
MARQUE

from FactInternetSales l JOIN DimDate d ON (l.ShipDateKey = d.DateKey) JOIN DimProduct p
On

(l.Productkey = p.Productkey)JOIN DimSalesTerritory s ON (l.SalesTerritoryKey =
s.SalesTerritoryKey)

where d.CalendarYear >= 2006 and s.SalesTerritoryGroup = 'Europe'

group by d.CalendarYear, p.EnglishProductName
```

```
UNION ALL

select sum(l.UnitPrice) as revenu, d.CalendarYear as ANNEE, 'NULL' as MARQUE

from FactInternetSales l JOIN DimDate d ON (l.ShipDateKey = d.DateKey) JOIN DimProduct p
On

(l.Productkey = p.Productkey)JOIN DimSalesTerritory s ON (l.SalesTerritoryKey =
s.SalesTerritoryKey)

where d.CalendarYear >= 2006 and s.SalesTerritoryGroup = 'Europe'

group by d.CalendarYear

UNION ALL

select sum(l.UnitPrice) as revenu, NULL as ANNEE, p.EnglishProductName as MARQUE

from FactInternetSales l JOIN DimDate d ON (l.ShipDateKey = d.DateKey) JOIN DimProduct p
On

(l.Productkey = p.Productkey)JOIN DimSalesTerritory s ON (l.SalesTerritoryKey =
s.SalesTerritoryKey)

where d.CalendarYear >= 2006 and s.SalesTerritoryGroup = 'Europe'

group by p.EnglishProductName

UNION ALL

select sum(l.UnitPrice) as revenu, NULL as ANNEE, 'NULL' as MARQUE

from FactInternetSales l JOIN DimDate d ON (l.ShipDateKey = d.DateKey) JOIN DimProduct p
On

(l.Productkey = p.Productkey)JOIN DimSalesTerritory s ON (l.SalesTerritoryKey =
s.SalesTerritoryKey)

where d.CalendarYear >= 2006 and s.SalesTerritoryGroup = 'Europe'

) FactInternetSales;
```

Deuxième requête :

```
select * from (

select sum(l.UnitPrice) as revenu, d.CalendarYear as ANNEE, p.EnglishProductName as
MARQUE

from FactInternetSales l JOIN DimDate d ON (l.ShipDateKey = d.DateKey) JOIN DimProduct p
On

(l.Productkey = p.Productkey)JOIN DimSalesTerritory s ON (l.SalesTerritoryKey =
s.SalesTerritoryKey)

where d.CalendarYear >= 2006 and s.SalesTerritoryGroup = 'Europe'

group by ROLLUP (d.CalendarYear, p.EnglishProductName)
```

```
UNION ALL

select sum(l.UnitPrice) as revenu, NULL as ANNEE, p.EnglishProductName as MARQUE

from FactInternetSales l JOIN DimDate d ON (l.ShipDateKey = d.DateKey) JOIN DimProduct p
On

(l.Productkey = p.Productkey)JOIN DimSalesTerritory s ON (l.SalesTerritoryKey =
s.SalesTerritoryKey)

where d.CalendarYear >= 2006 and s.SalesTerritoryGroup = 'Europe'

group by ROLLUP (p.EnglishProductName)

) FactInternetSales;
```

Troisième requête :

```
select * from (

select sum(l.UnitPrice) as revenu, d.CalendarYear as ANNEE, p.EnglishProductName as
MARQUE

from FactInternetSales l JOIN DimDate d ON (l.ShipDateKey = d.DateKey) JOIN DimProduct p
On

(l.Productkey = p.Productkey)JOIN DimSalesTerritory s ON (l.SalesTerritoryKey =
s.SalesTerritoryKey)

where d.CalendarYear >= 2006 and s.SalesTerritoryGroup = 'Europe'

group by CUBE (d.CalendarYear, p.EnglishProductName)) FactInternetSales;
```

3.4. Performance de requête

Les analystes utilisent les requêtes pour extraire des informations à partir de différentes dimensions existant dans la base de données dans un laps de temps.

Le temps de réponse doit être court. Pour cela, il est nécessaire d'agréger les données afin d'apporter des réponses rapides, aussi il est nécessaire d'optimiser les requêtes les plus fréquemment utilisées afin d'améliorer les temps de réponse.

Les deux premières requêtes (voir Figure 4 et 5) combinent les ensembles de résultats de plusieurs requêtes (plusieurs instructions SELECT) par UNION ALL, dans la requête spécifique, une jointure entre la table de fait FactInternetSales et les tables de dimension DimDate, DimProduct et DimSalesTerritory

La première requête (Figure 4) est composée de quatre instructions SELECT avec des changements rapportés à quelques paramètres des instructions. Dans la deuxième instruction nous remplaçons l'attribut « p.EnglishProductName » par « NULL ». Dans la troisième « d.CalendarYear » par « NULL ». Dans la quatrième nous remplaçons les deux attributs de sélection par « NULL ».

La deuxième requête (Figure 5) est composée de deux instructions SELECT (le regroupement par ROLLUP permet de résumer quatre instructions en deux instructions). Dans La dernière requête (Figure 6), nous ne remarquons qu'une seule instruction SELECT et aucune combinaison à cause de l'utilisation de l'opérateur CUBE.

3.5. Système OLAP dans le nuage

Le calcul distribué peut être défini comme l'utilisation d'un système distribué pour résoudre un grand problème en le décomposant dans plusieurs tâches où chacune est calculée dans les ordinateurs individuels du système distribué. Un système distribué comprend plus d'un ordinateur par réseau. Tous les ordinateurs connectés dans un réseau communiquent entre eux pour atteindre un but commun en exploitant leur propre mémoire local. D'autre part les utilisateurs différents d'un ordinateur pourraient probablement avoir des exigences différentes et les systèmes distribués aborderont la coordination des ressources partagées en les aidants à communiquer avec d'autres nœuds pour réaliser leurs tâches individuelles. Cloud Computing se réfère d'habitude à la fourniture d'un service via internet. Ce service peut comprend logiciel d'affaires qui est en accès via le web du stockage hors-site ou des ressources de calcul, tandis que des moyens distribué de calcul résout un grand problème pour permettre ensemble d'ordinateurs de marcher en même temps.

L'entreposage de gros volume de données exige des moyens de traitements haut performance pour cela il y de nouveaux travaux et des technologies afin de traiter la difficulté liés aux Data Warehouse et les systèmes OLAP dans l'environnement Cloud Computing.

L'environnement d'exécution MapReduce est L'exemple probablement le plus connu dans l'informatique dans les nuages pour le traitement et la génération de grands volumes de données et son implémentation de sources libres Hadoop.

L'informatique dans le nuage offre des langages d'interrogation à haut niveau, tels que Hive de Facebook, Scope de Microsoft, qui sont basés sur des modèles de données particuliers comme le modèle orienté colonne ou des modèles étendant le modèle relationnel.

L'implémentation d'une architecture OLAP dans les nuages afin de réduire les coûts de stockage des données. Cependant, les charges de travail des entrepôts de données sont particulièrement difficiles dans le cloud. En effet, la plupart des environnements de cloud ne sont pas configurés pour les charges de travail typiques de l'entrepôt de données. Nous avons opté l'environnement privé de Cloud (*Cloud privé*), afin d'effectuer ces travaux, nous avons choisi de stocker toutes nos données sur le service de stockage repose sur HDFS et utilise MapReduce comme un environnement d'exécution et enfin Hive comme langage d'interrogation.

3.6. Scénario générale

- ✓ Chargement de DW en utilisant Sqoop.
- ✓ Le DW doit être stocké sous block sur HDFS.
- ✓ Le NameNode indique qu'il souhaite écrire les blocks.
- ✓ Les blocks ont été envoyés au DataNode.
- ✓ Le NameNode contient les métadonnées.
- ✓ Effectuer une requête sous Hive afin de construire un Cube OLAP.
- ✓ Notre requête Hive exécuté dans Job MapReduce.
- ✓ Le serveur JobTracker est en communication avec DataNode, il sait où sont les données d'entrée du programme Map/Reduce.
- ✓ Soumettre le programme au JobTracker.
- ✓ JobTracker découpe et affecte chaque tache aux TaskTracke.
- ✓ TaskTracker responsable de l'exécution de la tâche.
- ✓ MAP : traitement en parallèle.
- ✓ REDUCE : synthèse et agrégation des traitements.

- ✓ Apres l'exécution de la requête sous les étapes précédentes, le Cube OLAP a été construite.

4. Conclusion

Comme nous l'avons vu dans la section précédente notre architecture peut aider à faire une prise de décision. Ici, nous avons essayé de présenter chaque composant d'une manière simple pour vous permettre de tout comprendre le Principe de cette architecture. Le chapitre suivant est un coup d'œil sur la construction du cube OLAP et les différents outils utilisés.

Chapitre 4 :

Validation

Et

Expérimentation

1. Introduction

La construction d'un cube OLAP nécessite à un entrepôt de données (schéma en étoile ou en flocons ou en constellation), et aussi des requêtes de construction d'une part et des outils performants d'autre part.

Dans ce chapitre, on va montrer les étapes de configuration de la plateforme Cloudera, une configuration qui nous a permis d'importer l'entrepôt de données offert par Microsoft intitulé « AdventureWorks ». Ensuite nous entamerons nos expérimentations qui ont été effectuée sur Hive en exploitant Sqoop. Enfin on termine avec une comparaison entre la création d'un cube OLAP en SQL Server et Hive.

2. Configurations

2.1. Les caractéristiques du PC utilisé :

PC portable, ASUS

Processeur : Intel Core i7-3610QM CPU 3.4 GHz x8

RAM: 16G

Disque Dur: 256 Go SSD + 750 HDD

Système d'exploitation : windows 8

2.2. Les caractéristiques de la machine virtuelle

RAM: 12 Go

Disque Dur: 120 Go

Processeur:

- Nombre de processeur : 3
- Nombre de cœur pour un processeur : 2
- Totale cœurs de processeur : 6

Système d'exploitation : Linux CentOS 6.4

3. Construction de cube OLAP

L'évaluation des performances de construction de cube OLAP va se porter sur deux parties (Voir Figure 4.1) :

Dans la première partie on trouve tout d'abord les étapes de construction de notre cube OLAP en utilisant SQL Server, ensuite on présente les résultats des expérimentations effectuées sur l'entrepôt.

Dans la deuxième partie on montre les étapes d'installation de la machine virtuelle, Cloudera CDH5, la Configuration de CDH en Virtual machine en utilisant Cloudera Quickstart VM qui contient l'écosystème Hadoop.

Et se termine par la mesure du temps d'exécution obtenue à travers Hive.

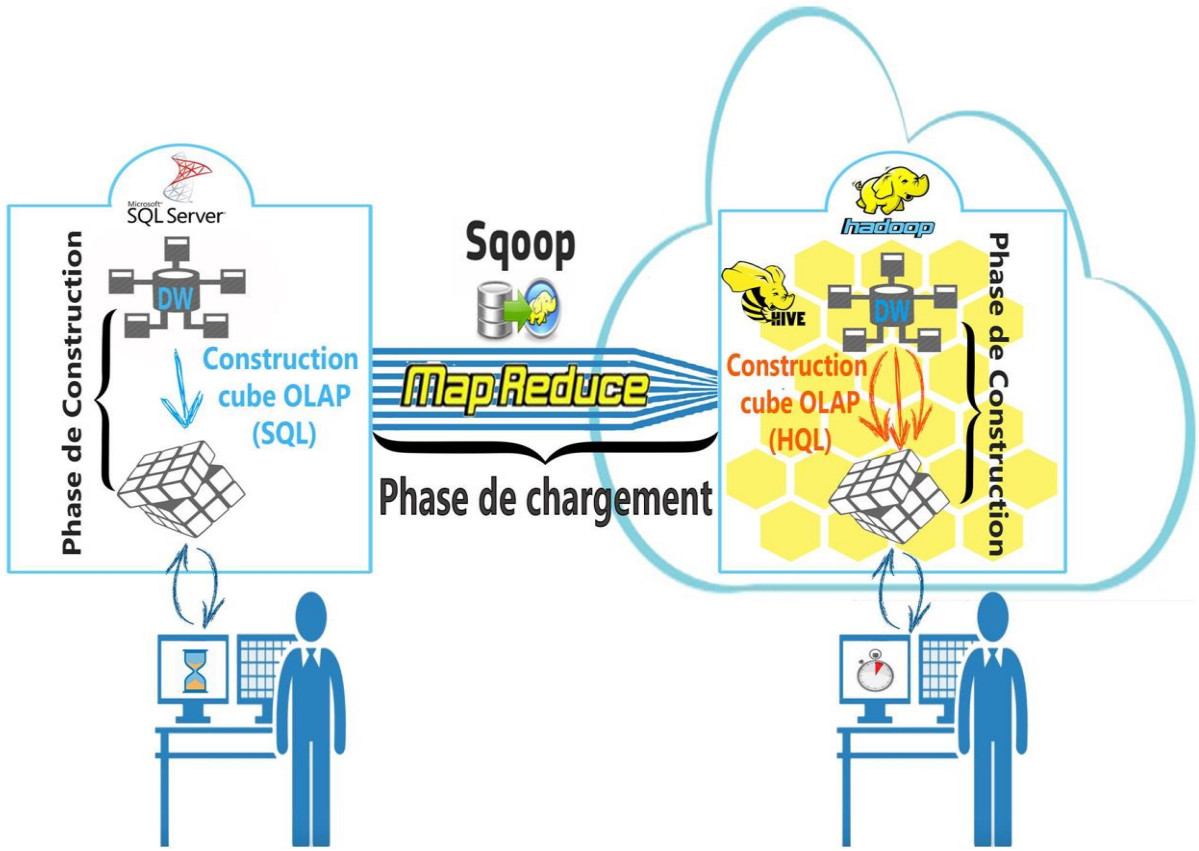

Figure 4.1 : Architecture comparative.

3.1. Première partie

Dans cette partie on a installé SQL SERVER 2014 pour montrer les expérimentations.

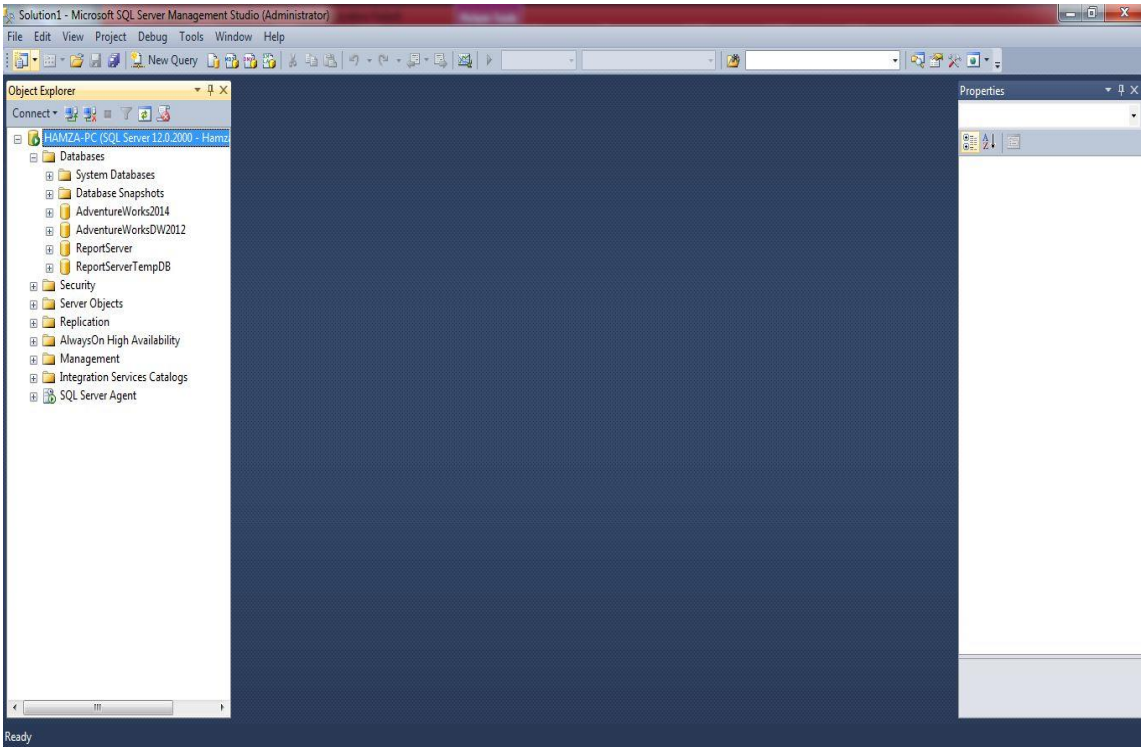

Figure 4.2 : L'environnement SQL SERVER 2014.

3.1.1. Cube OLAP en SQL Server

Cette expérimentation permet de tester la performance de trois requêtes décisionnelles (en utilisant l'opérateur CUBE, ROLLUP, UNION ALL) de construction cube OLAP avec des différentes tailles de l'entrepôt de données (passage à l'échelle).

Les graphiques suivants illustrent de manière plus visuelle les résultats de temps de construction du cube OLAP à partir d'une table de Fait (FactInternetSales) d'une taille allant de 2 millions de lignes jusqu'à 120 millions de lignes.

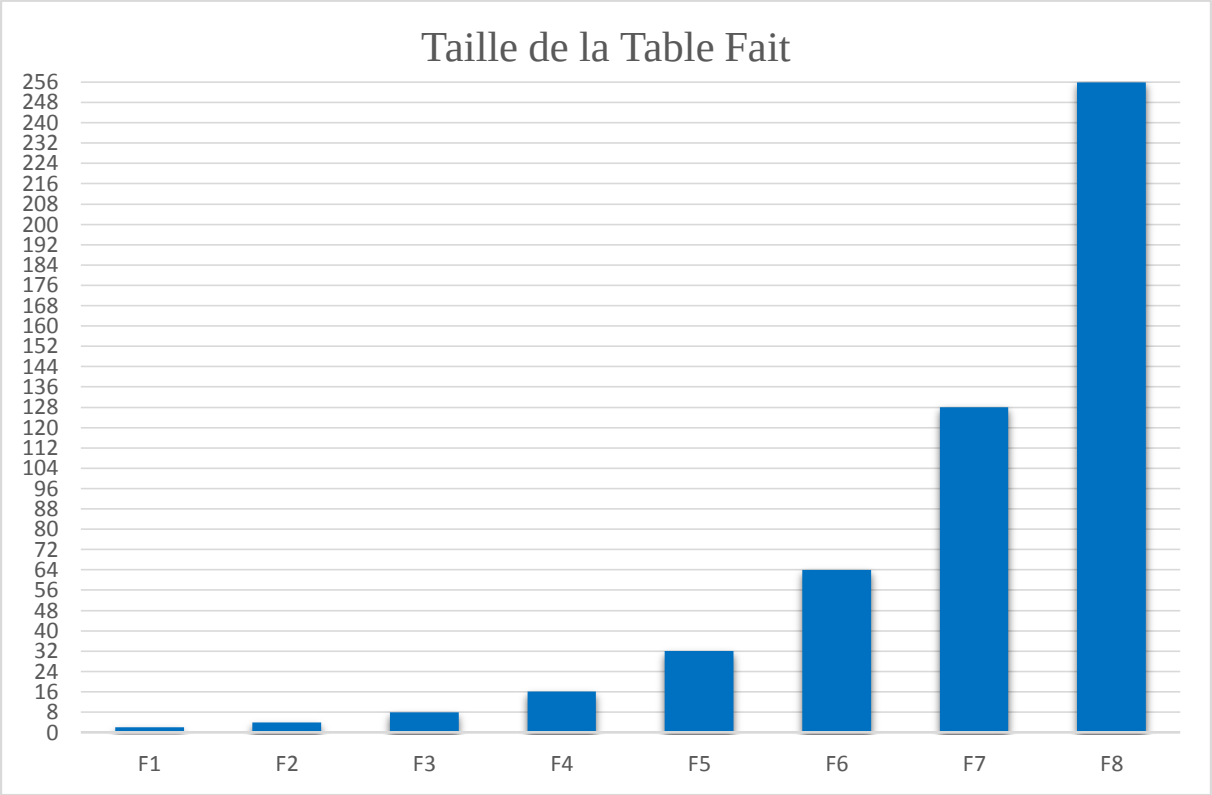

Figure 4.3 : passage à l'échelle du DW.

Cette figure (4.3) montre le passage à l'échelle de données de notre table de Fait on a commencé avec une table de fait contenant 2 millions lignes en augmentant sa taille en la multipliant par 2 à chaque reprise jusqu'à ce que nous ayons arrivé à une taille de 256 millions de lignes.

Dans le tableau suivant on présente le temps de construction du cube OLAP de chacune des requêtes suivantes (UNION ALL, CUBE, ROLLUP), l'unité de temps est SECONDE et les requêtes sont exécutées sous SQL Server.

	Construction de Cube OLAP dans SQL Server (S)		
	Requête 1 (UNION ALL)	Requête 2 (CUBE)	Requête 3 (ROLLUP)
2 Millions lignes	49	17	40
4 Millions lignes	76	33	68
8 Millions lignes	154	71	138
16 Millions lignes	298	117	228
32 Millions lignes	528	241	469
64 Millions lignes	1051	475	917
128 Millions lignes	2128	985	1921
256 Millions lignes	4152	1943	3789

Tableau 4.1 : Construction de Cube OLAP dans SQL Server.

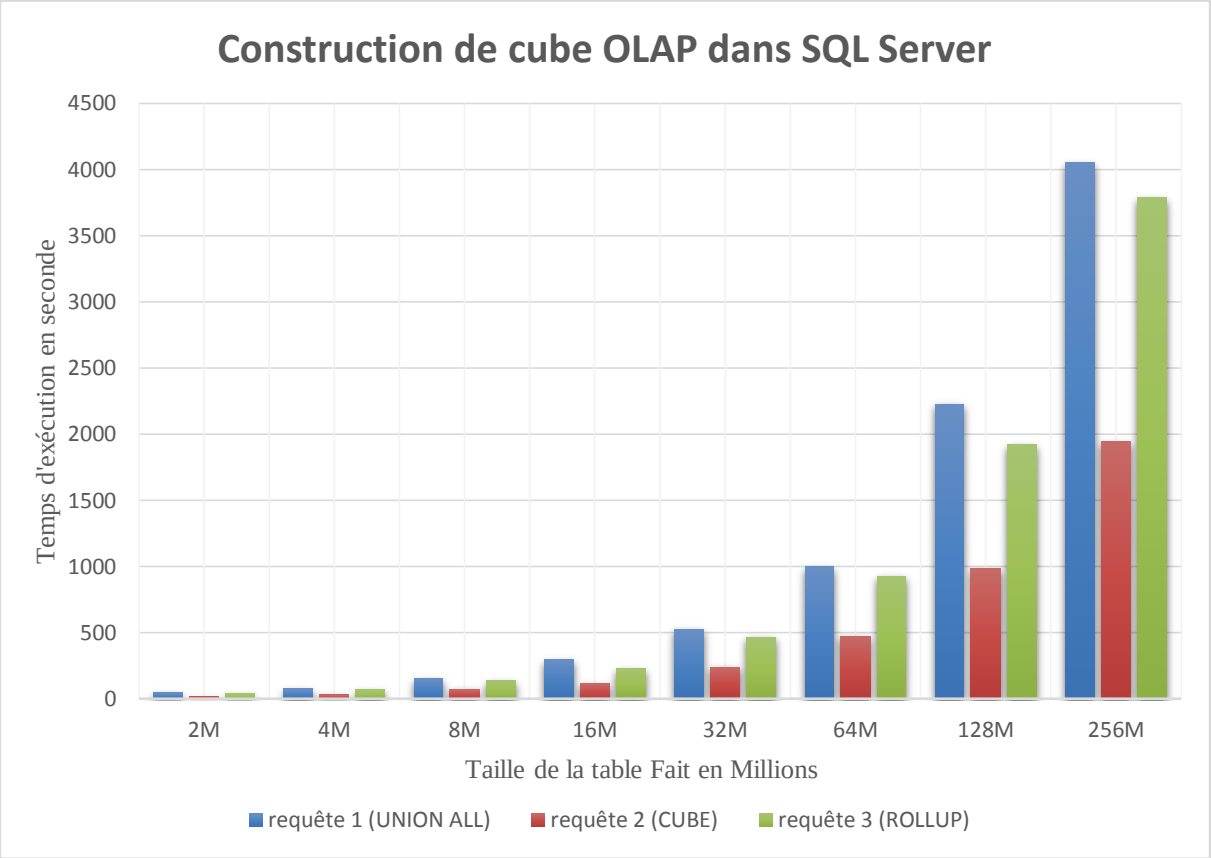

Figure 4.4 : Temps de construction de cube OLAP dans SQL Server.

Ce graphe montre les performances de trois requêtes (UNION ALL, CUBE, ROLLUP) de construction de cube OLAP dans SQL Server :

On remarque que le temps de réponse des requêtes s'augmentent progressivement à chaque fois que la taille de la table de fait incrémente, on constate que la requête CUBE présente la meilleure performance du temps d'exécution de la construction du cube OLAP par contre les requêtes ROLLUP et UNION ALL ont une performance médiocre.

3.2. Deuxième partie

3.2.1. Cube OLAP sous Hive

3.2.1.1. Méthodes d'installer CDH5 :

On peut installer CDH5 dans l'une des façons suivantes:

1. Installation à l'aide Cloudera Quickstart VM.
2. Installation de façon automatique en utilisant Cloudera manager et la configuration de sur un cluster.
3. Installation de façon manuelle (Hadoop, Hive, Sqoop...).

Dans Notre travail on a installé CDH en utilisant la première méthode.

Installation et Configuration de CDH on Virtual machine en utilisant Cloudera Quickstart VM :

a. **Installation :**

1. L'installation de machine virtuelle.

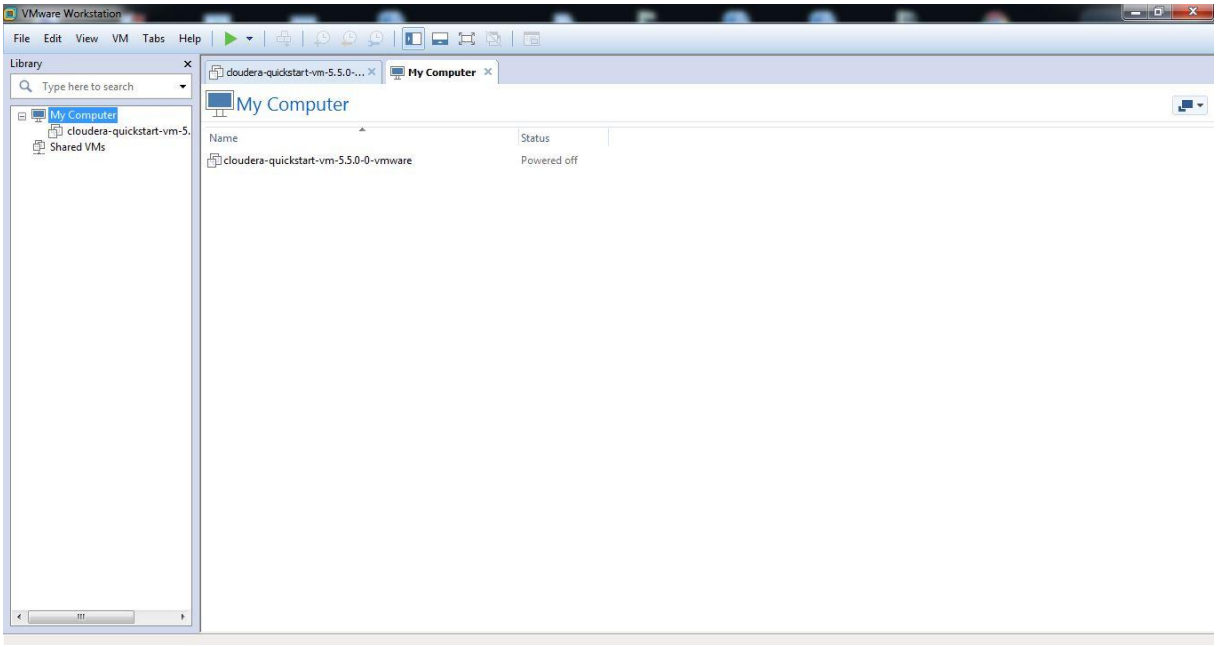

Figure 4.5 : VMware Workstation Pro.

2. Ouvrir machine virtuelle et cliquez sur "Nouveau" pour créer une nouvelle boîte virtuelle.

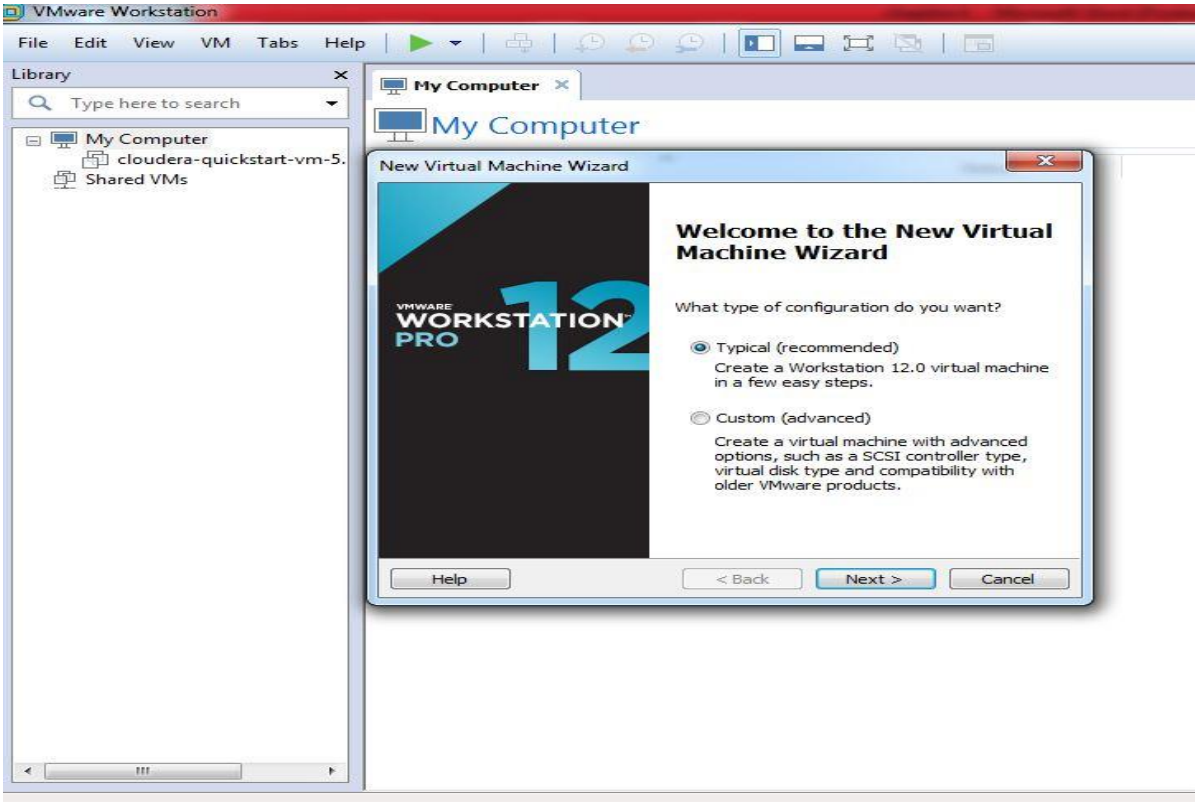

Figure 4.6 : Créer une nouvelle boîte virtuelle.

3. Sélectionner cloudera-quickstart-vm-5.5.0-0-vmware CDH est installé sur la machine virtuelle.

Figure 4.7 : Machine Cloudera-Quickstart.

b. Configuration :

1. Connecter sur route en utilisant « su » pour configurer la machine (Password : cloudera).

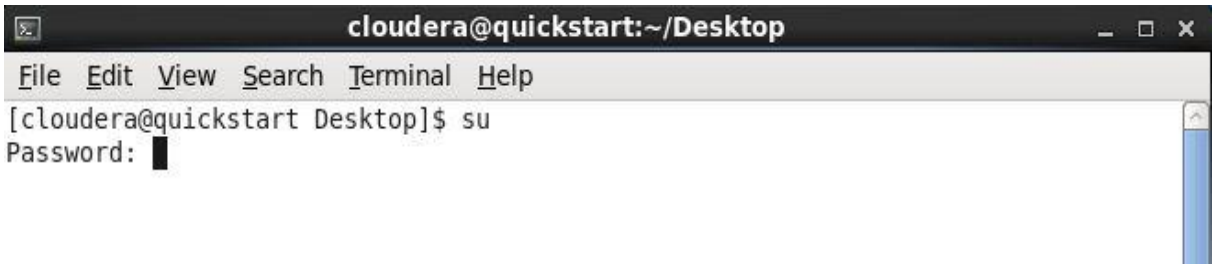

Figure 4.8 : Connecter sur route.

2. Accéder au fichier resolv pour configurer localdomain et nameserver.

Figure 4.9 : Configuration *localdomain et nameserver*.

3. Accéder au fichier network pour configurer hostname.

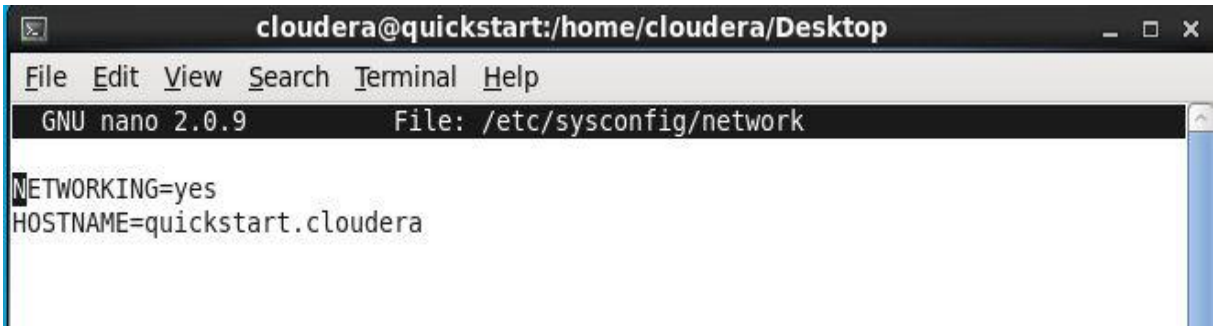

Figure 4.10 : Configuration hostname.

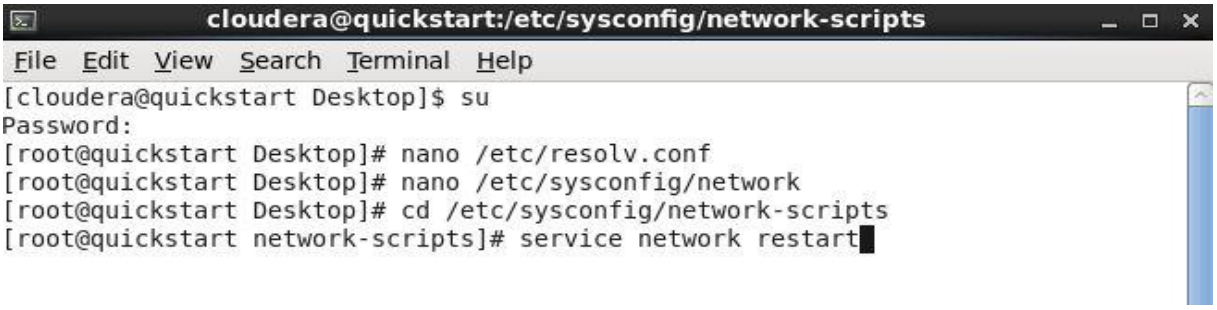

Figure 4.11 : Redémarrer les services.

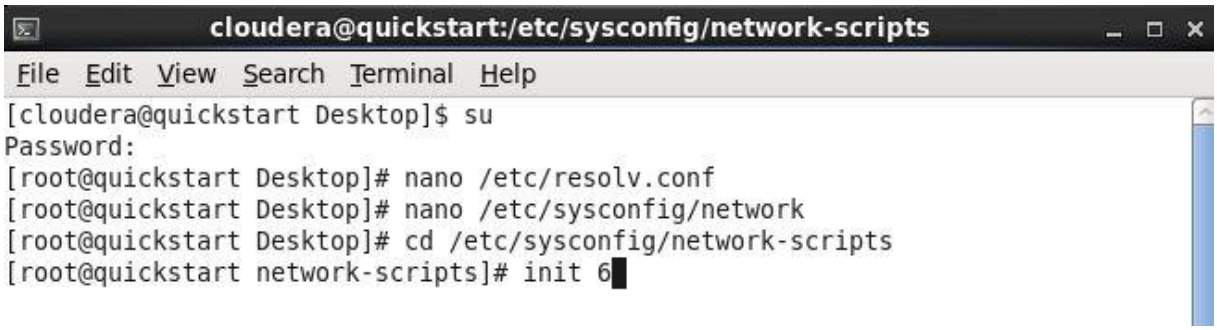

Figure 4.12 : Redémarrer la machine.

4. Utiliser Cloudera Manager express.

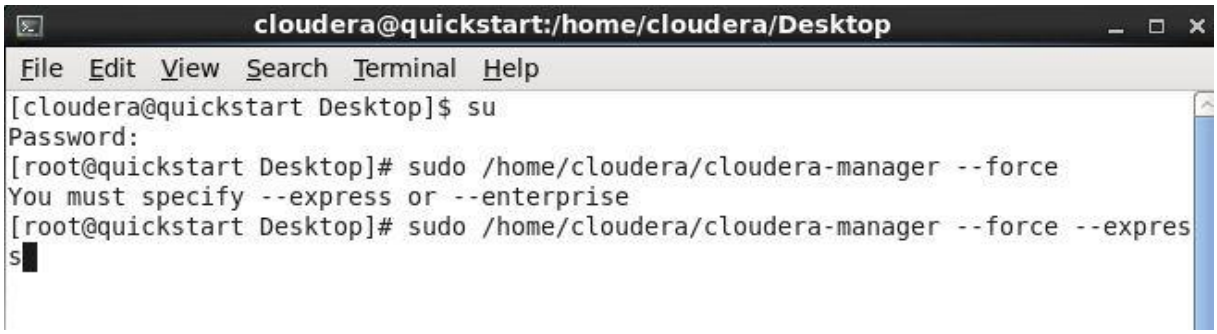

Figure 4.13 : Choisir Cloudera Manager Express.

5. Accéder au Cloudera Manager.

UserName : cloudera

PassWord : cloudera

6. les services sur notre machine.

Figure 4.14 : L'écosystème Hadoop.

3.2.1.2. Connexion entre SQL Server et Sqoop

Dans le but de charger des données de SQL Server dans Cloudera en utilisant Sqoop, (Sqoop fournit un moyen de transfère efficace des données entre Hadoop et bases de données relationnelles). Dans cette partie nous montre comment utiliser Sqoop pour importer des données dans le Cloudera à partir d'une source de données Microsoft SQL Server.

a. Configuration de l'environnement de SQL Server.

Côté SQL Server : il y a quelques paramètres qui doivent être configurés pour assurer une connectivité bienséante.

b. Création d'un utilisateur dans SQL Server.

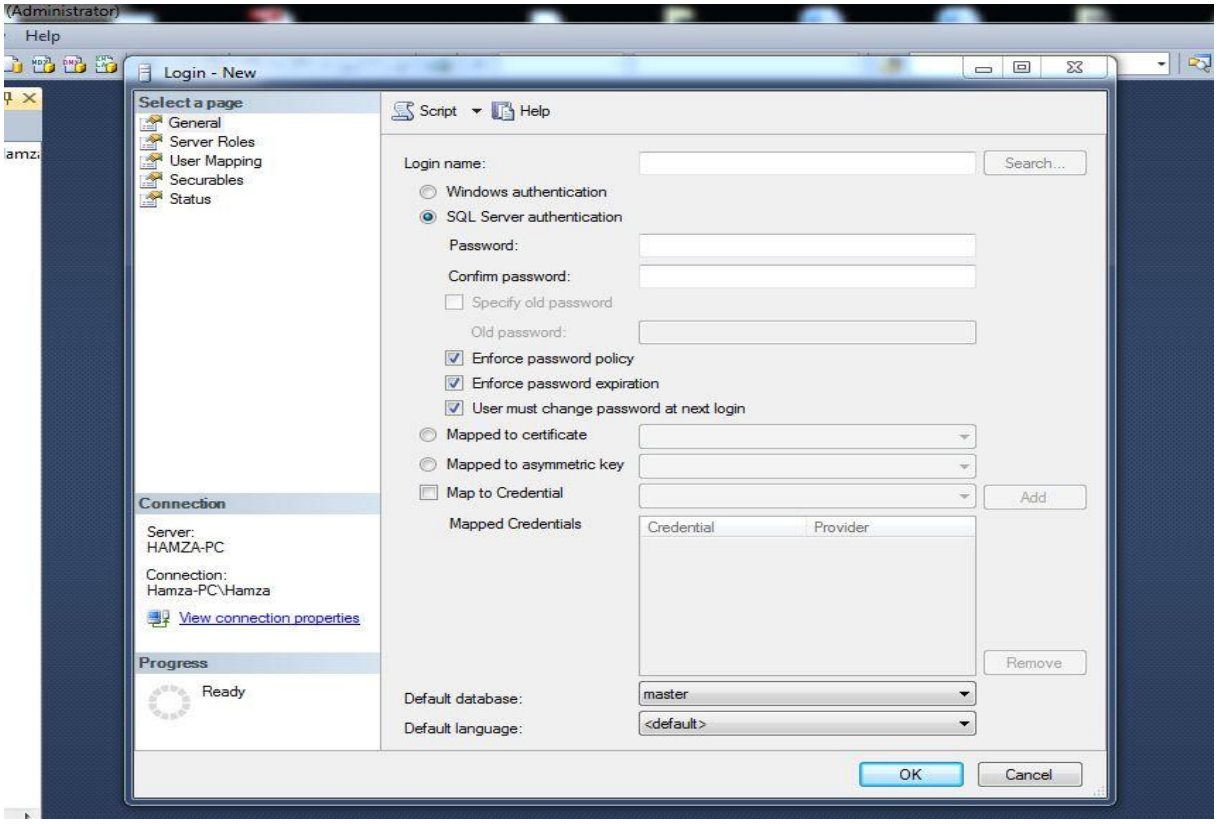

Figure 4.15 : ***Création d'un utilisateur dans SQL Server.***

c. Configurer l'adresse IP et le numéro de port.

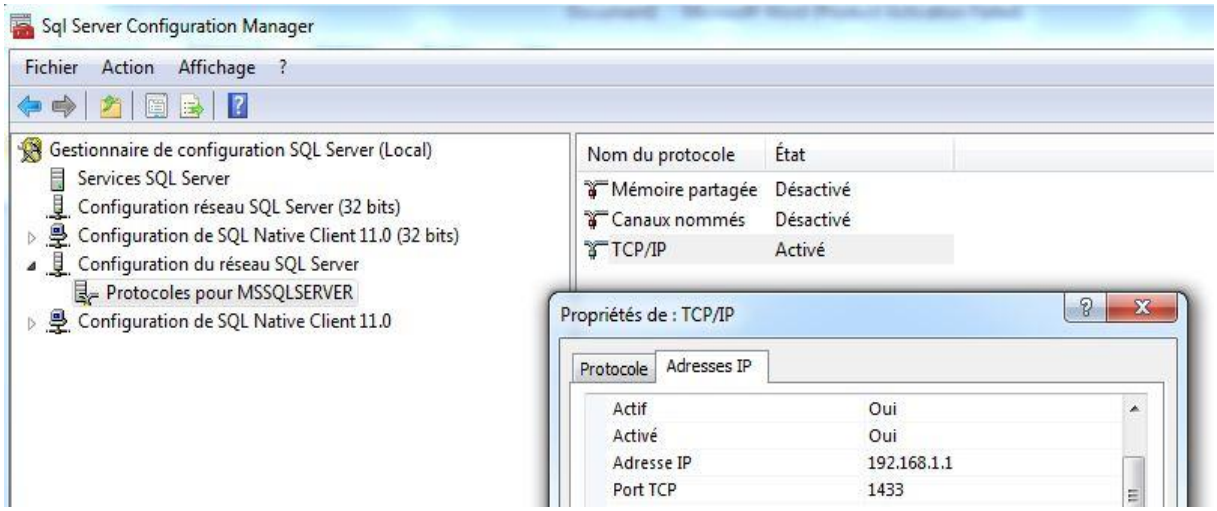

Figure 4.16 : Configuration SQL Server.

d. Téléchargez le pilote JDBC SQL Server.

Le pilote JDBC Microsoft SQL Server n'est pas inclus dans la bibliothèque Sqoop, donc on a besoin de télécharger ce pilote pour faire connecter les systèmes.

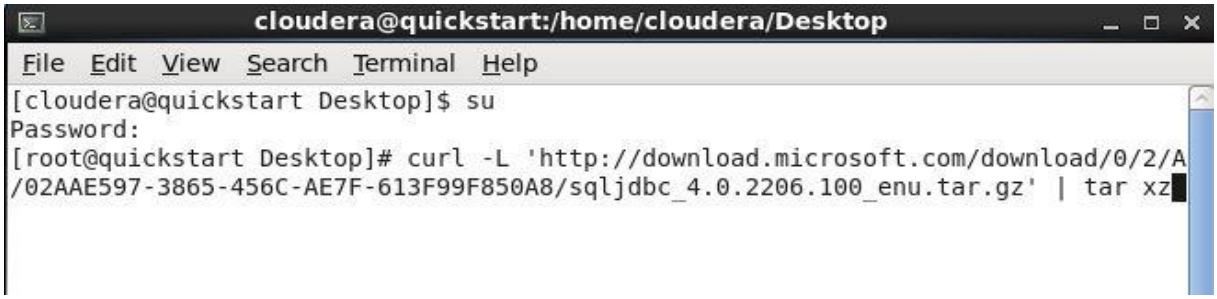

Figure 4.17 : Téléchargement *JDBC SQL Server*.

e. Copiez le pilote à la bibliothèque Sqoop.

Figure 4.18 : Ajouter JDBC *à la bibliothèque Sqoop*.

f. Confirmer si le pilote dans la liste de Sqoop.

Figure 4.19 : Confirmation de sqljdbc4.jar.

g. Etablir la connexion.

Avant de commencer à importer les tables de dimensions, nous Vérifions notre connexion via Sqoop à SQL en utilisant l'adresse IP et le numéro de port avec le nom d'utilisateur et mot de passe configuré.

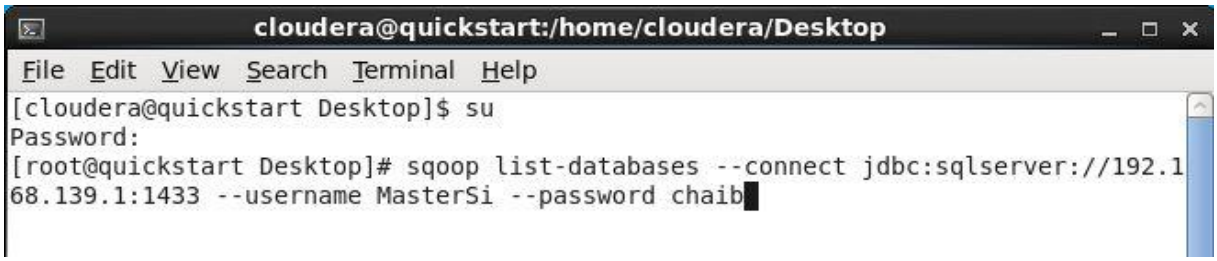

Figure 4.20 : *Etablir la connexion.*

Enfin, on connecte Sqoop avec SQL Server pour importer l'AdventureWorks.

h. chargement les tables sous Hive.

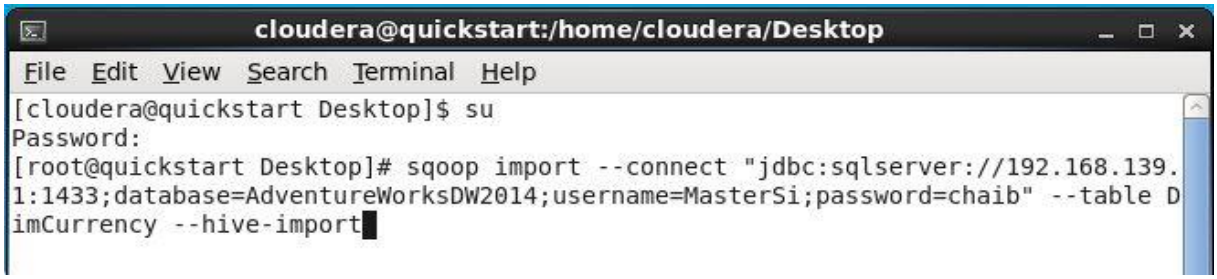

Figure 4.21 : *Chargement la table DimCurrency sous Hive.*

3.2.1.3. Présentation et analyse des résultats

Le tableau suivant représente le temps de chargement des tables par seconde en utilisant Sqoop à partir de SQL Server à Hive.

Tables	Temps de chargement (S)
DimCurrency	2,642
DimCustomer	3,747
DimDate	27,094
DimProduct	18,229
DimPromotion	6,677
DimSalesTerritory	10,369
FactInternetSales (2 Millions)	210,647
FactInternetSales (4 Millions)	411,294
FactInternetSales (8 Millions)	801,588
FactInternetSales (16 Millions)	1107,176
FactInternetSales (32 Millions)	1521,352
FactInternetSales (64 Millions)	2039,411
FactInternetSales (128 Millions)	3119,193
FactInternetSales (256 Millions)	4237,039

Tableau 4.2 : Temps de chargement des tables de dimension.

Apres le chargement de la table de Fait et les table de dimensions sous Hive, les données ont été stockées dans un répertoire par défaut : ***user /Hive / Warehouse / nom de la table.*** Chaque table est divisée en plusieurs blocks, comme on le voit dans la figure suivante.

Figure 4.22 : Présentation des blocks de données.

Accéder au Hive Pour exécuter les requêtes précédentes :

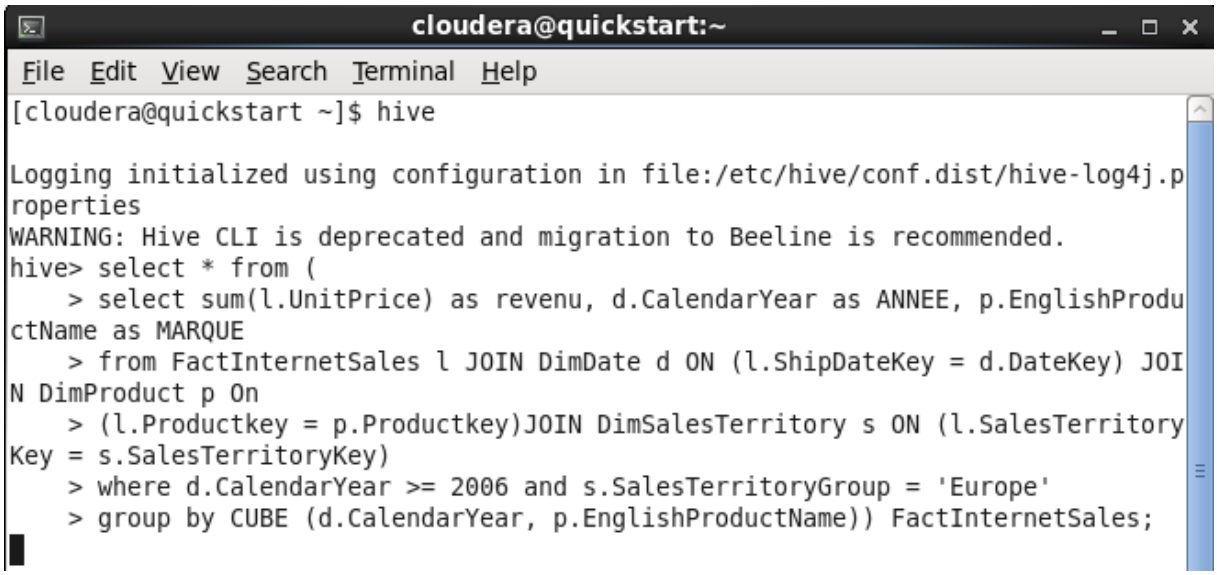

Figure 4.23 : Requête de cube OLAP dans Hive.

La requête Hive est transformée et exécutée en plusieurs Job MapReduce sur la plateforme Hadoop.

On remarque comment le pourcentage de l'opération Map et l'opération Reduce s'augmente progressivement, cela montre que le programme est exécuté avec succès sur CDH en utilisant MapReduce.

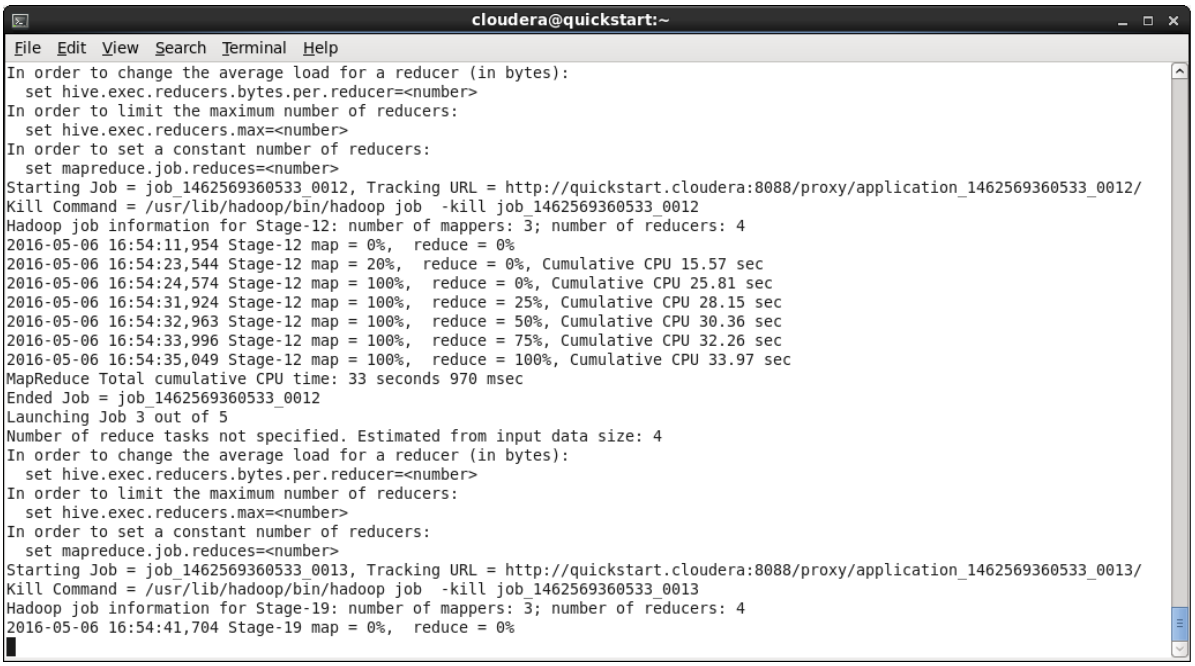

Figure 4.24 : Job MapReduce.

La figure suivante représente l'historique de chaque Job MapReduce

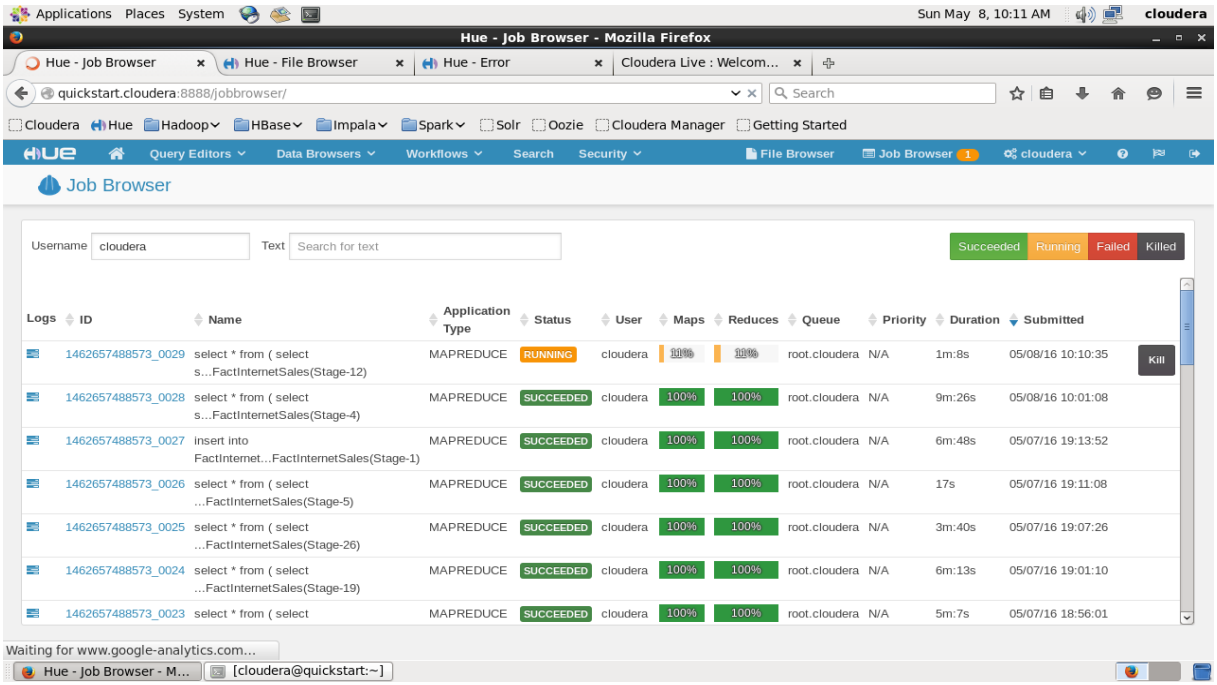

Figure 4.25 : Historique Job MapReduce.

Pour le tableau suivant on a fait la même exécution pour les requêtes précédentes sous Hive :

	Construction de Cube OLAP sous Hive (S)		
	Requête 1 (UNION ALL)	Requête 2 (CUBE)	Requête 3 (ROLLUP)
2 Millions lignes	164,64	32,97	97,32
4 Millions lignes	210,39	55,95	112,72
8 Millions lignes	264,58	84,23	149,32
16 Millions lignes	371,88	149,43	259,91
32 Millions lignes	640,76	269,71	491,64
64 Millions lignes	1200,05	495,11	936,09
128 Millions lignes	2006,93	910,28	1811,98
256 Millions lignes	3521,72	1655,93	3224,57

Tableau 4.3 : Temps de construction de Cube OLAP sous Hive.

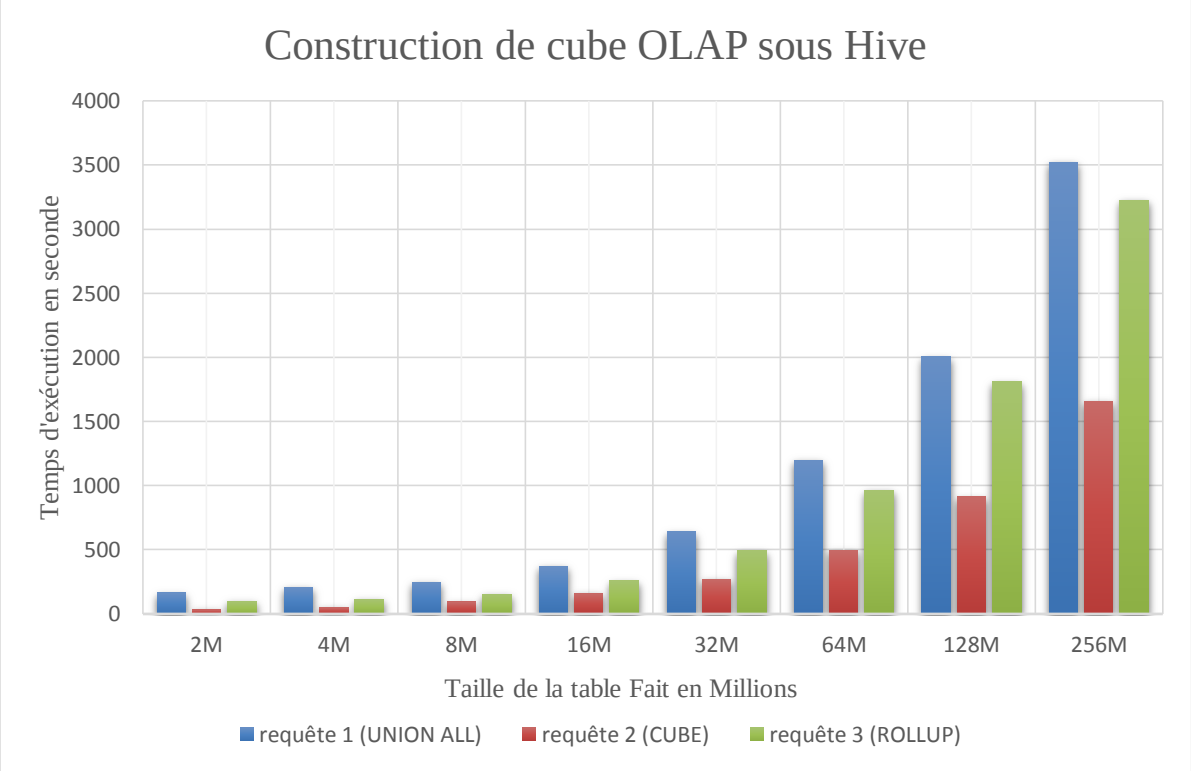

Figure 4.26 : Construction de cube OLAP sous Hive.

Dans le premier graphe, on a testé les trois requêtes (UNION ALL, CUBE, ROLLUP) sur l'entrepôt de données dans l'environnement SQL Server, le même traitement a été effectué sous Hive et les résultats ont montré que la requête cube est la plus adéquate pour la construction d'un cube OLAP.

Les résultats de notre requête :

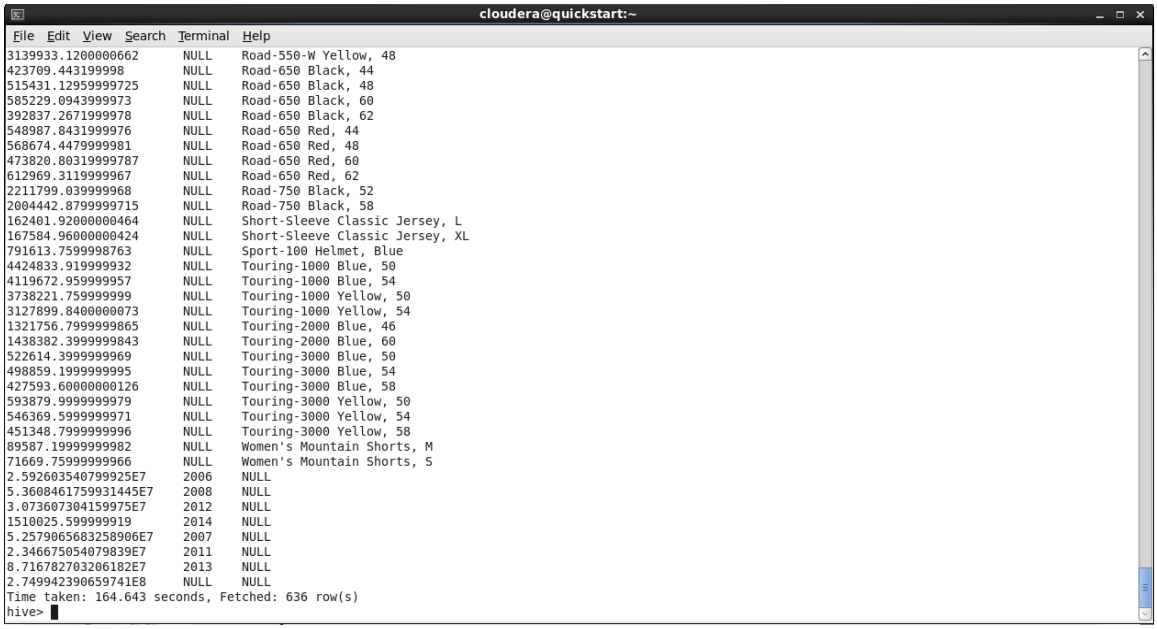

Figure 4.27 : Résultat de requête cube OLAP.

4. Etude comparative

4.1. Exécution sous Hive VS Exécution dans SQL Server

Pour la figure suivante, on va faire une comparaison d'exécution de la requête UNION ALL dans Hive et SQL Server, ensuit on remarque que le temps de construction de cube OLAP en SQL server est minime par rapport à Hive quand on gère une table de fait avec une taille allant de 2 millions à 64 millions mais une fois la taille dépasse 64 millions, on remarque que la performance de SQL Server baisse et celle de Hive s'améliore.

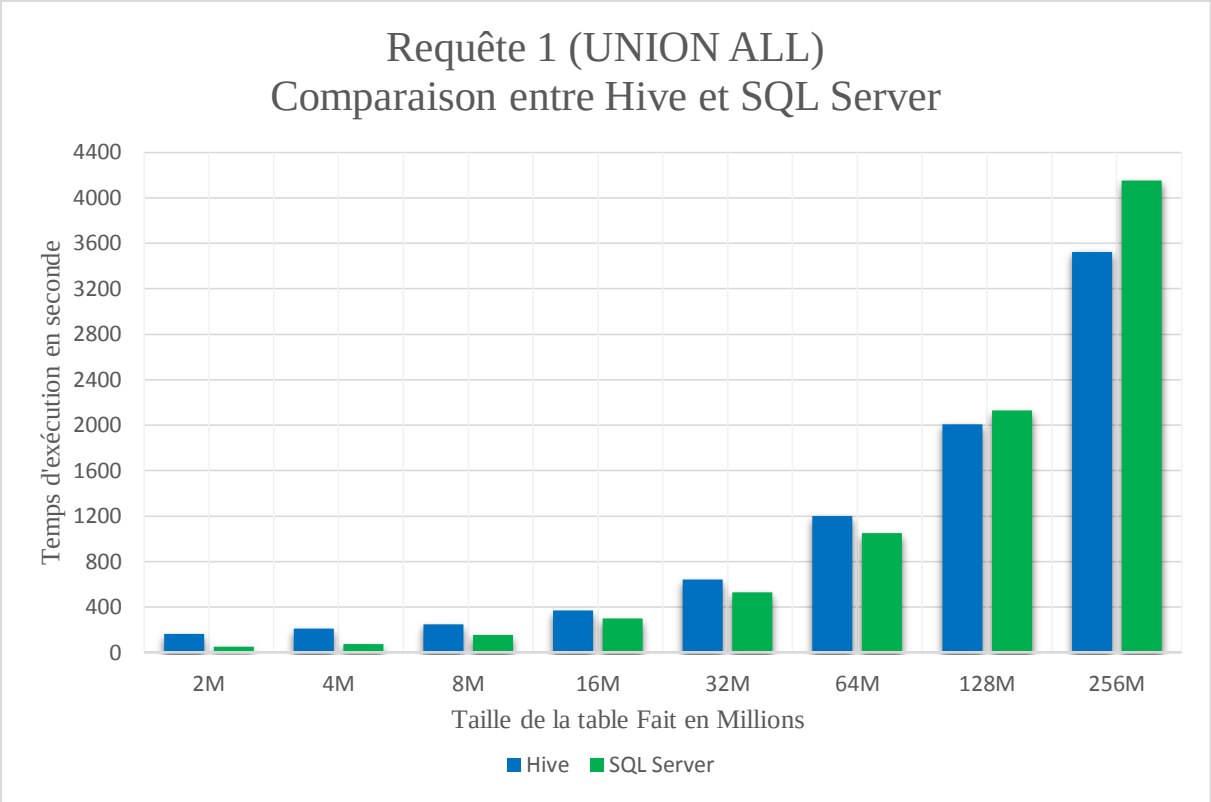

Figure 4.28 : Comparaison entre Hive et SQL Server (Requête 1 (UNION ALL)).

Dans la figure suivante, on va faire une autre comparaison d'exécution mais avec la requête ROLLUP dans Hive et SQL Server, on constate que les résultats d'exécution de la requête sont presque les même avec la figure () mais la requête ROLLUP est plus optimisée que la requête UNION ALL.

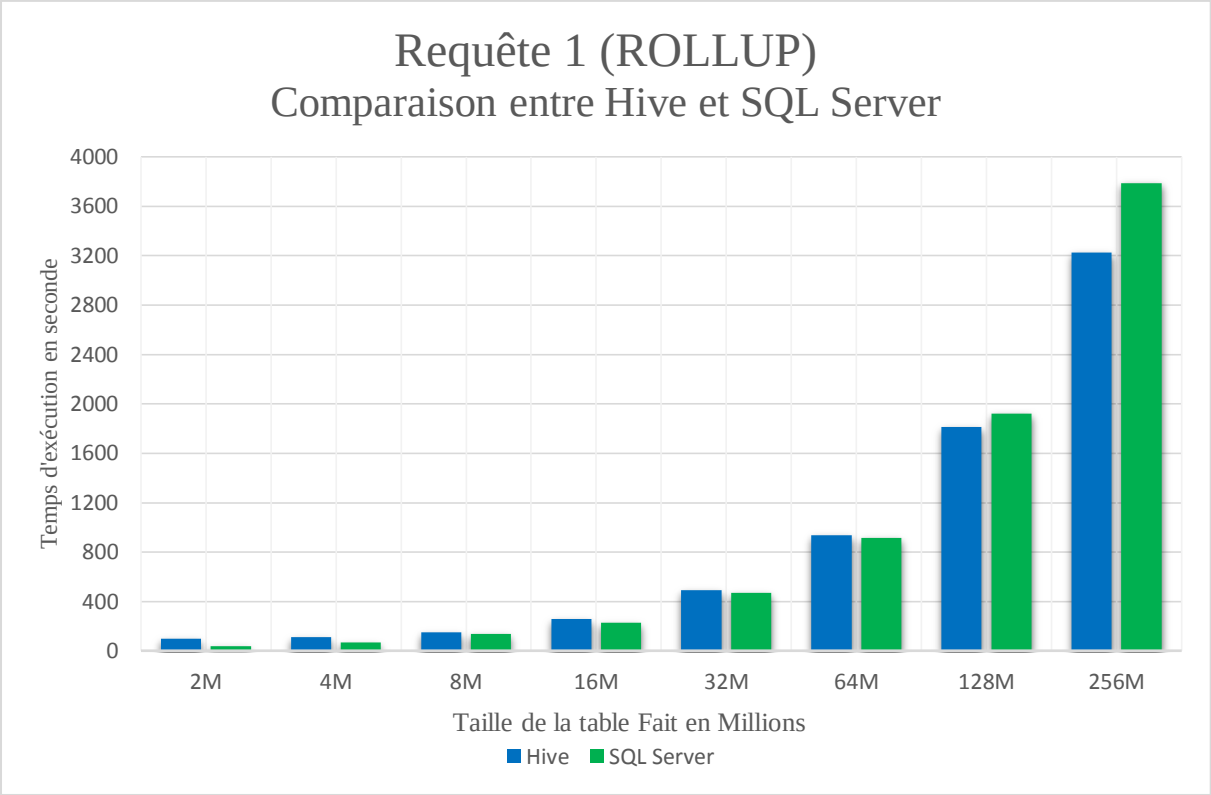

Figure 4.29 : Comparaison entre Hive et SQL Server (Requête 1 (ROLLUP)).

Pour la figure suivante, on a fait une construction avec la requête plus optimisé CUBE entre les trois requêtes, remarque que le temps de la construction est mieux dans Hive pour des tailles grande d'échelle avec un temps d'exécution 15min 10s pour une table de fait de 128 millions lignes et 27min 35s pour une table de fait du taille 256 millions lignes, par contre dans SQL server le temps est plus amendé pour la construction de cube pour les tailles minimes.

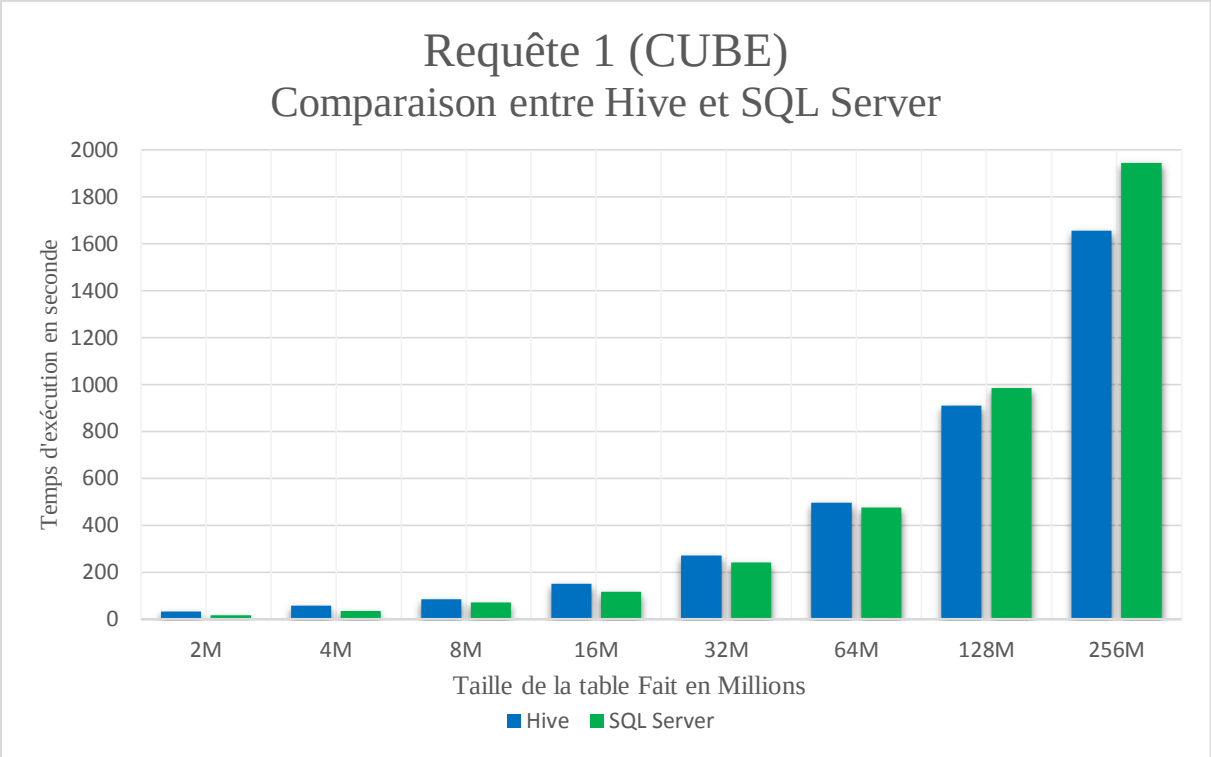

Figure 4.30 : Comparaison entre Hive et SQL Server (Requête 1 (CUBE)).

4.2. Taux d'accélération Hive VS Taux d'accélération SQL Server

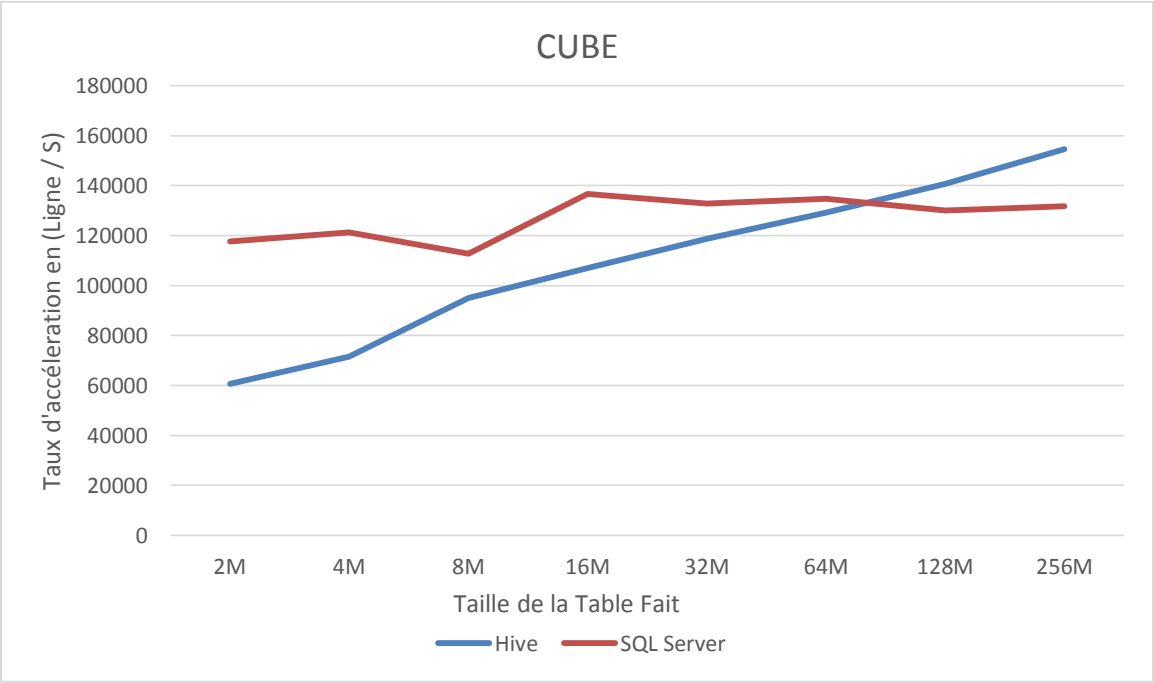

Figure 4.31 : Taux d'accélération entre Hive et SQL Server (Requête 1 (CUBE)).

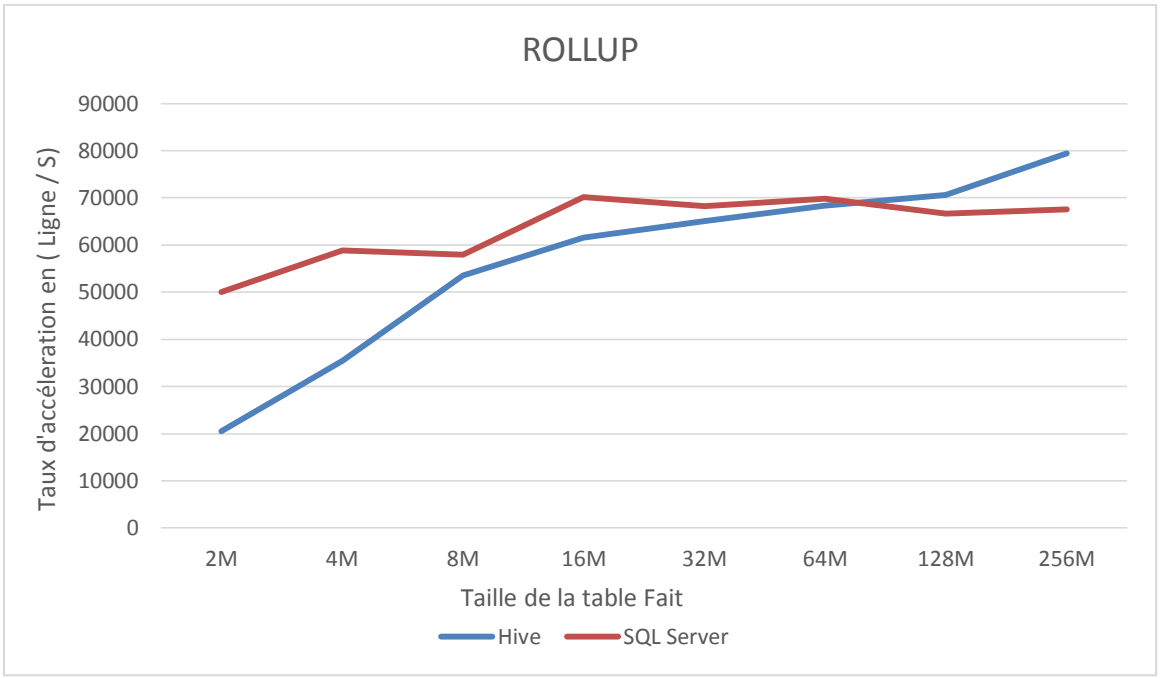

Figure 4.32 : Taux d'accélération entre Hive et SQL Server (Requête 1 (ROLLUP)).

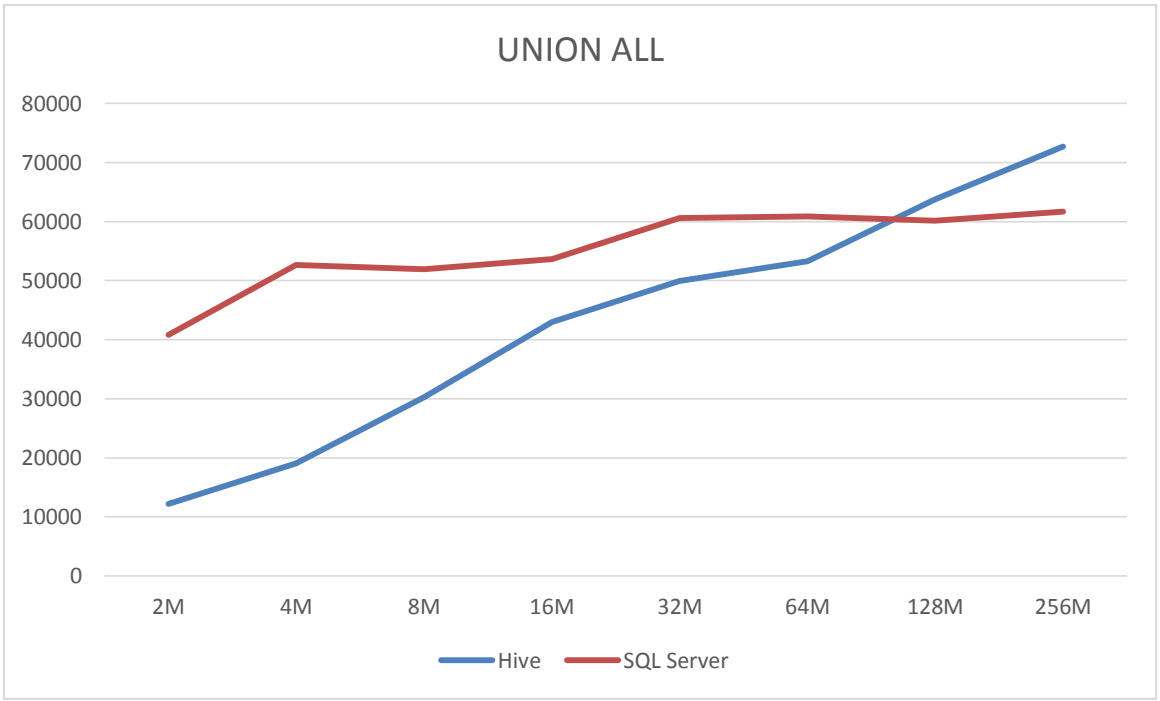

Figure 4.33 : Taux d'accélération entre Hive et SQL Server (Requête 1 (UNION ALL)).

Ces trois graphes a été obtenus grâce à une formule qui est définit comme suit :

Nombre de lignes / temps de réponse

Depuis les graphes présentés précédemment nous pouvons remarquer que l'accélération de SQL Server au début était plus élevé que celle de Hive ceci montre qu'avec des donnés de petite et moyenne taille faut mieux opter pour SQL Server, après que le taille des donnée as dépassée 90 millions de lignes la performance de SQL Server as débutée à baisser alors que celle de Hive as vu une amélioration impressionnante depuis cette remarque on déduit que pour des données de taille massive Hive serait une meilleur solution.

Si on interprète la figure (4.34) qui compare les temps liés au SQL et Hive, on constate que:

Si le temps d'exécution est défini sur un intervalle [0, b], il existe un temps a qu'est graphiquement l'intersection de deux graphes tel que :

$$T_{SQL} <= T_{hive} \text{ sur } [0, a]$$

$$T_{SQL} >= T_{hive} \text{ sur } [a, b]$$

Ainsi notre travail consiste à prouver que si on intègre MapReduce et SQL sur l'intervalle [a, b], on arrive à un temps SQL inférieur que le temps Hive quel que soit la taille des données.

NB : une meilleure accélération implique une meilleure performance.

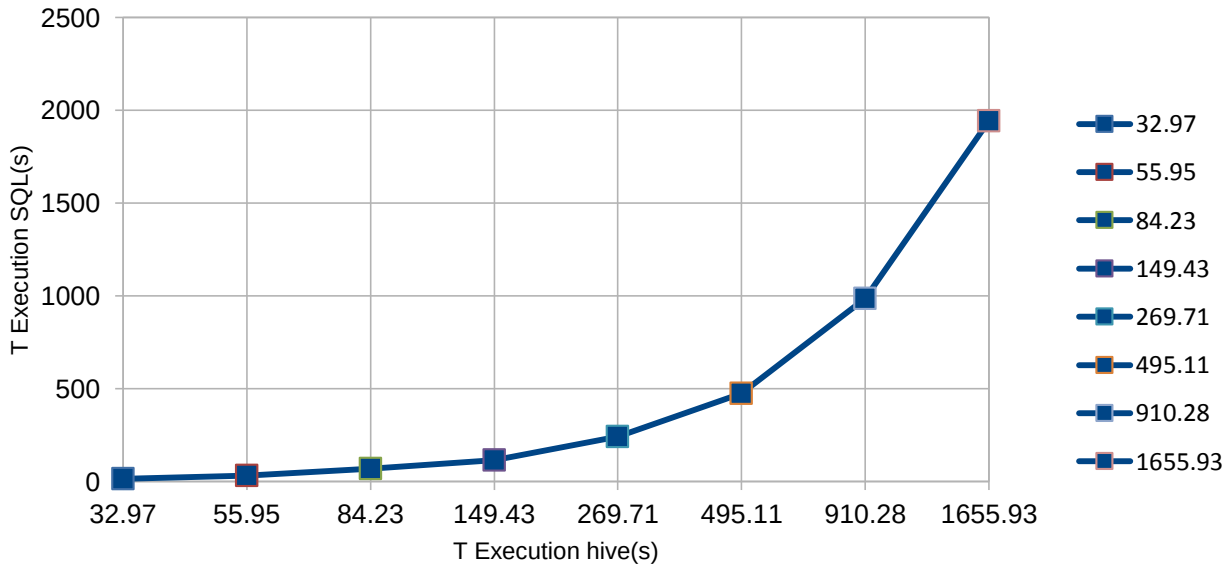

Figure 4.34 : les résultats entre le temps d'exécution d'une requête SQL et de la même requête Hive.

Le graphe de la figure 4.34 représente les résultats entre le temps d'exécution d'une requête SQL et de la même requête Hive.

La courbe est hyperbolique ($ax^2 + bx + c$).

Mathématiquement, ce cube sur un intervalle [a, b] peut être divisée en plusieurs Sous cube sur des intervalles $[a_i, b_i] \subset [a, b]$.

Cette subdivision permet de trouver un segment $[c_i x + d_i]$ / $(C_i) >= C$.

A partir du segment nous pourrions déduire le nombre des nœuds nécessaire pour qu'une requête SQL soit plus performante qu'une requête Hive.

Calcul de a_i, b_i :

$$[a_i, b_i] = (C_i) \cap (C)$$

$T_{SQL} = a_i T_{hive} + b_i$

Sur chaque $[a_i T_{Min\ hive} + b_i]$:

$$T_{SQL\ Min} = a_i T_{Min\ hive} + b_i$$

$$T_{SQL\ Max} = a_i T_{Max\ hive} + b_i$$

$$a_i = \frac{T_{Min\ SQL} - T_{Max\ SQL}}{T_{Min\ hive} - T_{Max\ hive}}$$

$$b_i = T_{Min\ SQL} - a_i T_{Max\ hive}$$

Calcul de nombre des nœuds :

$$T_{SQL} < T_{hive}$$

$$a_i T_{hive} + b_i < T_{hive}$$

L'exécution de l'étude comparative peut être adaptée comme une solution pour permettre l'amélioration du temps d'exécution on intégrant le SQL Server basée MapReduce.

Choix de langage SQL ou Hive dépend étroitement des nœuds de jobs

Si $n_{SQL} > n_{job\ hive}$ alors ou exemple de requête

Discussion :

L'étude graphique que nous avons réalisée permet de déduire qu'il est nécessaire d'utiliser MapReduce avec SQL pour permettre de réduire le temps d'exécution de SQL server. Telle façon qu'il est plus performant que celui de Hive.

Calcule des nombres des nœuds pour chaque intervalle $[a_i, b_i]$

$$n \geq \frac{T_{Max\ SQL}}{T_{Max\ hive}} = \frac{1943}{1655,3} = 2$$

Hive jobs → 3, SQL jobs → 2.

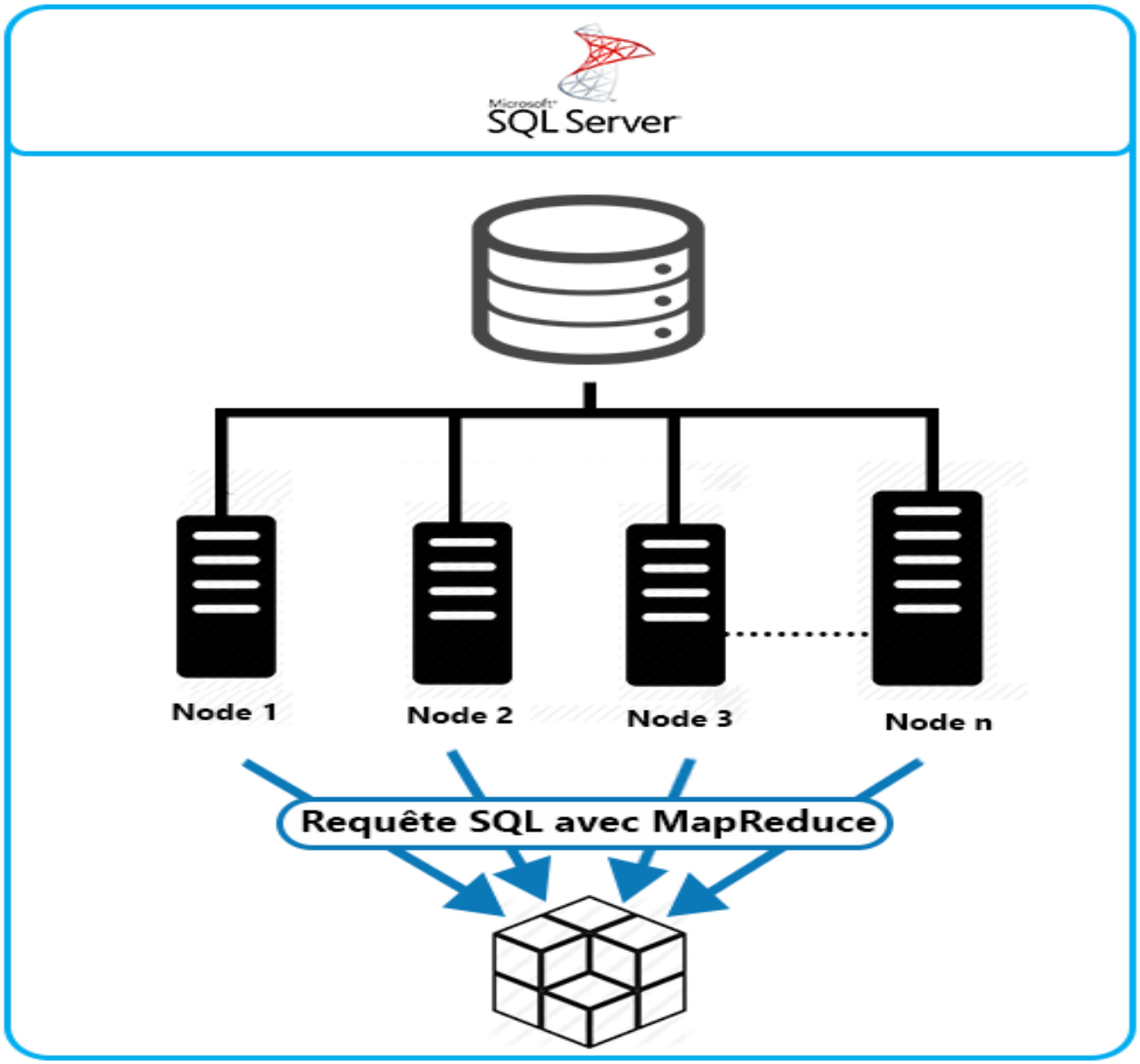

Figure 4.35 : SQL Server avec MapReduce

5. Conclusion

Ce dernier chapitre est composé de deux parties : la première met l'accent sur les différentes technologies pour la construction des cubes OLAP. La seconde réalise en détail notre projet constitué d'un entrepôt et un cube OLAP, nous avons comparé les différents résultats et enfin Choisit le meilleur.

Conclusion générale et perspectives

Tout au long de ce mémoire, nous avons présenté les différentes technologies nécessaires pour le processus d'aide à la décision. Depuis le CUBE OLAP et DW a été identifié il y a eu plein de travaux de recherches établis dans ce domaine, de plus nous nous sommes intéressés à la technologie Hadoop et MapReduce et à son utilisation dans un Cloud Computing.

Après avoir présenté les concepts fondateurs de ce domaine, ce travail présente l'intégration de l'entrepôt de données, l'analyse en ligne OLAP dans le Cloud pour réduire les coûts de stockage. Nous avons conduit un ensemble d'expérimentations pour étudier le processus de construction de cube, nous avons utilisé deux environnements (SQL SERVER et Hive) et des jeux de données (2000000, 4000000, 8000000, 60000000...., 120000000) lignes.

Ce travail étant une œuvre humaine, n'est pas un modèle unique et parfait, c'est pourquoi nous restons ouverts à toutes les critiques et nous sommes prêts à recevoir toutes les suggestions et remarques tendant à améliorer d'avantage cette étude.

Dans nos futurs travaux, nous souhaiterons proposer une nouvelle approche qui tenterais d'adapter SQL server a MapReduce afin d'améliorer les performances de ce dernier, cette adaptation nécessite une augmentation du nombre de nœuds.

Références Bibliographiques

[1] Chaudhuri, S., Dayal, U., Narasayya, V. R., « An overview of business intelligence technology ». Communications of the ACM, Vol.54, n°8, p.88-98, 2011.

[2] Olivier Teste. Modélisation et manipulation des systèmes OLAP : de l'intégration des documents à l'usager. Interface homme-machine [cs.HC],2009.

[3]W.H. Inmon, Building the Data Warehouse. John Wiley and Sons, Third edition, 2002.

[4] http://www.christian.braesch.fr/page/difference-entre-sgbd-et-entrepot-de-donnees.

[5] DJAMEL GARAR, compression dans les entrepôts de données pour l'amélioration des performances, 2013.

[6] Kamel Boukhalfa, De la conception physique aux outils d'administration et de tuning des entrepôts de données.

[7] la construction en ligne des tables individus variable par apprentissage automatique numérique (PMC).

[8] F. Ravat, O. Teste, G. Zurfluh, Algèbre OLAP et langage graphique.

[9] Laurent d'Orazio, Sandro Bimonte, Entreposage et analyse en ligne dans les nuages avec Pig, 2011.

[10] Max Chevalier, Mohammed El Malki, Arlind Kopliku, Olivier Teste, Ronan Tournier, Entrepôts de données multidimensionnelles NoSQL, 2014.

[11] Datawarehouse: Cubes OLAP.

[12] Kimball, R., Ross M., Thornthwaite W., Mundy J. et Becker B. 2008. «The Data WarehouseLifecycleToolkit», p 672.

[13] Cécile Favre, Fadila Bentayeb, Omar Boussaid, Jérôme Darmont, Gérald Gavin, Nouria Harbi, Nadia Kabachi, Sabine Loudcher, Les entrepôts de données pour les nuls. . . ou pas ! 2014.

[14] E.F. Codd, S.B. Codd and C.T. Salley, Providing OLAP to User Analysts, 1993.

[15] Entrepôts de données : Systèmes ROLAP, MOLAP et HOLAP(5) Bernard ESPINASSE Professeur à Aix Marseille Université (AMU) Ecole Polytechnique Universitaire de Marseille Décembre 2015.

[16] https://technet.microsoft.com/fr-fr/library/hh916543.aspx.

[17] http://igm.univ-mlv.fr/~dr/XPOSE2009/informatique_decisionnelle_olap/colap.html.

[18] Wu. M-C and buchmann. A.P. « research issues in data warehousing ». in BTW ferman database conference, 1997.
[19] sur le Cloud Computing et les Datacenters à l'attention des collectivités locales Juillet 2015.

[20] La maîtrise des applications et des prestataires dans une opération d'outsourcing.

[21] Guide sur le Cloud Computing et les Datacenters à l'attention des collectivités locales 2015, Page27, 28, 30.

[22] Administration de systèmes, réseaux et applications à base de logiciels libres IUT Nancy Charlemagne.

[23] Cloud Computing Connectivity (Windows Azure, DRMAA) with ProActive Resource Manager, 2009/2010.

[24] Livre Blanc La sécurité et la virtualisation Mai 2012.

[25] Building OLAP cubes on a Cloud Computing environment with MapReduce, Billel ARRES, Nadia KABBACHI, Omar BOUSSAID.

[26] Tout savoir sur Hadoop : Vulgarisation de la technologie et les stratégies de certains acteurs.

[27] http://www.expert-only.com/business-intelligence/big-data/apache-hadoop.

[28]http://mbaron.developpez.com/tutoriels/bigdata/hadoop/introduction-hdfs-map reduce/.

[29] Un modèle de fouille de données Cloud basée sur le principe Map/Reduce de Google.

[30] Modélisation de processus ETL dans un modèle MapReduce.

[31] Jason Venner, Pro Hadoop, 2009, page 71, 73.

[32] Srinath Perera, Thilina Gunarathne, Hadoop MapReduce Cookbook, 2013, page 12.

[33] Boris Lublinsky, Kevin T. Smith, Alexey Yakubovich, PROFESSIONAL HADOOP SOLUTIONS, 2013, page 9, 320.

[34] Sameer Wadkar, Madhu Siddalingaiah, Pro Apache Hadoop, 2013 page 238, 336.

[35] Jonathan R. Owens, Jon Lentz, Brian Femiano, Hadoop Real-World Solutions Cookbook, 2013, page16.

[36] http://www.tutorialspoint.com/hadoop/hadoop_introduction.htm.

[37] GUIDE DU BIG DATA 2014/2015.

[38] Hive: A Warehousing Solution Over a Map-Reduce Framework, Ashish Thusoo, Joydeep Sen Sarma, Namit Jain, Zheng Shao,Prasad Chakka, Suresh Anthony, Hao Liu, Pete Wyckoff, Raghotham Murthy, Facebook Data Infrastructure Team.

[39] Building Cubes with MapReduce, Alberto Abelló, Jaume Ferrarons, Oscar Romero.

[40] Les problèmes de sécurité liés aux architectures de l'entrepôt de données dans le Cloud, 2000.