République Algérienne Démocratique et Populaire
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique
Université Saad Dahlab – Blida 1

Faculté des Sciences
Département Informatique

Mémoire de fin d'études

Pour l'obtention du diplôme MASTER en Informatique
Option : **Ingénierie du Logiciel**

Thème

Analyse des données en ligne (OLAP) dans un environnement Data Lake

Réalisé par :
- GHAZI Lyna Kaouther
- MOKHTARI Fatima Zohra

Encadré par :
- Mr. M. BALA

Soutenu le 27/09/2022, devant le jury composé de :

Mme L. OUKID	Maître de Conférences B	Dépt. d'Informatique U. Blida1	**Présidente**
Mme I. CHERFA	Maître de Conférences B	Dépt. d'Informatique U. Blida1	**Examinatrice**
Mr. M. BALA	Maître de Conférences B	Dépt. d'Informatique U. Blida1	**Encadreur**

Promotion : 2021/2022
Session : Septembre 2022

Remerciements

Nous tenons à remercier en premier lieu Allah le Tout-Puissant pour sa grâce infinie.

Notre profonde gratitude va au Professeur Mahfoud BALA,

Qui nous a guidés de manière experte à travers notre projet de fin d'études et a partagé sa disponibilité, ses orientations, ses précieuses conseils et ses encouragements. Sa passion pour le monde du Big Data, nous a gardés constamment engagés dans nos recherches et nous a en effet motivés à poursuivre notre future carrière dans cette voie. Au-dessus du sol, personne n'a été plus important pour nous dans la poursuite de ce projet que nos familles. Nous leur sommes redevables, car leur valeur pour nous ne fait que croître avec l'âge. Nous voudrions remercier nos parents, dont l'amour et les conseils nous accompagnent dans tout ce que nous entreprenons. Leur soutien n'était pas seulement financier mais aussi émotionnel.

Résumé

Les lacs de données (Data Lake) sont devenus une tendance et ont trouvé leur popularité chez les entreprises à cause des avantages qu'ils offrent à savoir l'ingestion des données basée sur le principe AS-IS et le processus ELT (Extract-Load-Transform) au lieu du processus classique ETL (Extract-Transform-Load), qui rend les données dans les Data Lake caractérisés par la préservation de leur format natif et la diversité des structures (des données structurées, semi-structurées et non structurées). Nous présentons, dans ce mémoire, la mise en œuvre d'un système d'analyse de données en ligne (OLAP) dans un environnement big data où les données sont issues à partir d'un lac de données (Data Lake).

L'objectif de ce travail est de construire des cubes de données OLAP à partir d'un Data Lake. Pour ce faire, nous avons mis en œuvre une plate-forme d'analyse OLAP constituée de trois couches principales à savoir (1) transformation des données, (2) stockage des cubes de données dans un modèle NoSQL et (3) algèbre OLAP adaptée au modèle NoSQL. La plate-forme OLAP proposée a été développée avec le langage python, la plate-forme de distribution et de parallélisation d'Apache en l'occurrence Hadoop et Spark et enfin le SGBD MongoDB pour le stockage des cubes OLAP dans un modèle NoSQL orienté document.

Mots clés : Système décisionnel, Big Data, Data Lake, OLAP, NoSQL, ELT.

Abstract

Data lakes have become a trend and have found popularity among companies because of the advantages they offer, namely data ingestion based on the AS IS principle and the ELT (Extract-Load-Transform) process instead of the classical ETL (Extract-Transform-Load) process, which renders the data in the data lake characterised by the preservation of its native format and the diversity of structures (structured, semi-structured and unstructured data). In this thesis, we present the implementation of Online Analytical Processing (OLAP) in a big data environment where the data is taken from a data lake.

This work aims to build OLAP data cubes from a data lake. For this purpose, we have implemented an OLAP analysis platform consisting of three main layers namely (1) data transformation, (2) storage of data cubes in a NoSQL model, and (3) OLAP algebra adapted to the NoSQL model. The proposed OLAP platform has been developed using the Python language, the Apache Hadoop and Spark distribution and parallelization platform, and finally the MongoDB DBMS for storing OLAP cubes in a document-oriented NoSQL model.

Keywords: Decision support system, Big Data, Data Lake, OLAP, NoSQL, ELT.

الملخص

أصبحت بحيرات البيانات اتجاهًا واكتسبت شعبية بين الشركات بسبب المزايا التي تقدمها، وهي استيعاب البيانات استنادًا إلى مبدأ AS IS وعملية (Extract-Load-Transform) ELT بدلاً من ETL الكلاسيكية

(Extract-Transform-Load) عملية تجعل البيانات في بحيرة البيانات تتميز بالحفاظ على تنسيقها الأصلي وتنوع الهياكل (بيانات منظمة وشبه منظمة وغير منظمة). في هذه الأطروحة، نقدم تنفيذ المعالجة التحليلية (OLAP) في بيئة البيانات الضخمة حيث يتم أخذ البيانات من بحيرة البيانات.

يهدف هذا العمل إلى بناء مكعبات بيانات OLAP من بحيرة بيانات. لهذا الغرض، قمنا بتنفيذ منصة تحليل OLAP تتكون من ثلاث طبقات رئيسية هي (1) تحويل البيانات، (2) تخزين مكعبات البيانات في نموذج NoSQL ، و (3) تكييف الجبر OLAP مع نموذج .NoSQL تم تطوير منصة OLAP المقترحة باستخدام لغة Python، ومنصة التوزيع Apache Hadoop و Spark ، وأخيراً نظام MongoDB DBMS لتخزين مكعبات OLAP في نموذج NoSQL موجه للمستندات.

الكلمات الرئيسية: نظام دعم القرار ، البيانات الضخمة ،بحيرة البيانات، OLAP، NoSQL، ELT.

Table des matières

Table des figures

Liste des tableaux

I. Introduction générale

1. Contexte et problématique

« L'information est le pétrole du XXIe siècle, et l'analytique en est le moteur à combustion ». disait déjà en 1965, Peter Sondergaard, Vice-président senior et responsable mondial de la recherche chez Gartner Inc, une entreprise américaine de conseil et de recherche dans le domaine des techniques avancées. En effet, avec l'avènement de la numérisation, une énorme quantité de données est générée chaque seconde. Par ailleurs, les données ne sont pas seulement le résultat des processus des organisations mais proviennent d'une variété de sources issues d'écosystèmes différents et d'une multitude de véhicules de collecte. Ces quantités de données, qui couvrent un large éventail de domaines, est inestimable pour les applications d'analyse et d'aide à la décision. Avec l'essor du Big Data, les organisations ont commencé à stocker un flux infini de données collectées dans des structures appelées "lacs de données". Les lacs de données (Data Lake) se basent sur des schémas en lecture (schema-on-read) contrairement aux entrepôts de données (Data Warehouse) qui eux, sont basés sur un schéma plutôt en écriture (schema-on-write). Ceci donne une capacité au Data Lake à stocker des données collectées à partir de différentes sources hétérogènes en préservant leur format natif. Cependant cette puissance de stockage rend plus complexe la réalisation d'analyses du contenu hétérogène du Data Lake. Afin de permettre aux applications d'exploiter ses données, le Data Lake adopte un processus d'intégration de données de type ELT (Extract-Load-Transform) au lieu d'ETL (Extract-Transform-Load) et ce afin que les applications puissent adapter elles-mêmes les données à leurs objectifs de traitements.

« Si la collecte des données était un jeu d'enfant, tirer quelque chose d'utile de cette mer de connaissances était le vrai défi. » [1]. Dans les systèmes décisionnels classiques, le cube OLAP est une méthode de stockage de données sous forme multidimensionnelle, adaptée à des fins d'analyse et d'aide à la décision. L'OLAP est principalement, destiné aux bases des données relationnelles. En revanche, les systèmes traditionnels ne peuvent plus faire face à l'hétérogénéité de données devenue accrue ni à leur volumétrie devenue excessive. La mise en œuvre d'une plate-forme OLAP dans un environnement Data Lake

(DL) fait face aujourd'hui à des défis à savoir le format natif des données (structurées, semi-structurées et non structurées) dans les lacs de données, la phase de transformation (la construction de Data Lake fait par le processus ELT) et aussi OLAP est principalement conçu pour le modèle relationnel.

2. Objectifs et contributions

Les lacs de données et les systèmes OLAP jouent un rôle important en aidant les décideurs à obtenir le maximum d'avantages des données en question. Dans le cadre de ce travail, Nous proposons une approche de traitement des cubes OLAP dans un environnement Data Lake. Étant donné un Data Lake préalablement alimenté, notre système devra permettre la transformation des données issues du Data Lake en vue de les transformer en cubes de données exploitables pour des fins d'analyse. En ce sens, notre travail ne couvre pas les phases E et L de 'ELT puisque nous ne considérons pas d'autres sources de données à part le Data Lake. Différents modules seront intégrés dans la solution que nous proposons : (1) module d'extraction des données à partir du Data Lake, (2) module de transformation des données (Phase T de l'ELT) afin d'intégrer les données hétérogènes dans une structure homogène sous forme d'un cube OLAP, (3) module de stockage des cubes OLAP NoSQL et enfin (4) module des opérations OLAP NoSQL et visualisation des résultats.

3. Organisation du mémoire

Dans un souci de bien structurer ce travail, notre rapport de mémoire se subdivise en trois parties comme suites :

Partie 1 : se compose de trois chapitres consacrés aux fondamentaux. Nous définissons, dans le premier chapitre de cette partie, les concepts de base sur les systèmes décisionnels et analyse OLAP. Le deuxième chapitre est dédié pour les généralités sur le Big Data et les quatre modèles de données NoSQL (Clé/Valeur, Orienté colonne, Orienté document, Orienté graphe). Le troisième chapitre, quant à lui, est consacré aux environnements Data Lake.

Partie 2 : Nous présentons, dans cette partie, les travaux connexes selon les objectifs de ce projet. Cette partie est scindée en deux chapitres. Le chapitre 5 est dédié pour les

approches ayant trait à l'analyse des données en ligne que nous concluons par une synthèse et une discussion. Quant au chapitre 6, celui-ci présente l'approche ainsi que l'architecture proposées.

Partie 3 : Cette partie est, elle aussi, organisée en deux chapitres, Nous présenterons, dans le chapitre 7, les outils ainsi que l'environnement d'implémentation de notre système. Le chapitre 8 est consacré pour la description du système.

Première partie

Les fondamentaux

II. Système décisionnel et Analyse OLAP

Dans ce chapitre, nous décrivons le contexte général sur le Système d'information décisionnel (SID), nous présentons l'architecture de ce système et les concepts clés qui y sont associés.

---

1. Système d'information décisionnel (SID)

1.1. Introduction

La généralisation des bases de données et des progiciels intégrés (*ERP : Enterprise Ressources Planning*) a multiplié la quantité d'informations à traiter au sein des entreprises. Au-delà des traitements classiques pour des fins de gestion quotidienne, ces données doivent être, en plus, analysées avec soin pour améliorer la compétitivité de l'entreprise, comprendre le marché et faire face à une concurrence devenue aujourd'hui accrue. Cependant, l'émergence de volumes inhabituels de données avec des formats plus variés et des sources plus diverses notamment le web, les réseaux sociaux, la téléphonie mobile et les capteurs digitaux, rend les processus de traitement et d'analyse des données plus complexes puisque ceux-ci s'exécutent aujourd'hui dans des environnements big data. L'informatique décisionnelle étudie des données provenant de différentes sources pour en restituer un résultat clair et concis permettant à l'entreprise d'évaluer ses performances et de comprendre le marché et la concurrence.

1.2. Définition

Systèmes d'information décisionnels A la différence des systèmes opérationnels (ou transactionnels) qui sont dédiés aux métiers de l'entreprise pour les assister dans leurs tâches de gestion quotidiennes, les systèmes d'information décisionnels facilitent la définition et la mise en œuvre de stratégies. Mais il ne s'agit pas de définir une stratégie une fois pour toute, mais d'être à même de continuellement s'adapter à son environnement, et de le faire plus que ses concurrents. [...] Les systèmes décisionnels traditionnels permettent de faire l'analyse des activités déjà réalisées et d'en tirer des enseignements

pour les activités futures, pour cela ils utilisent des données plus ou moins récentes (au mieux mises à jour quotidiennement) . Les systèmes décisionnels plus avancés gèrent des données plus fraîches (certaines sont mises à jour en quasi temps réel), automatisent des décisions et supportent en temps réel des opérations (centre d'appels, web par exemple) [2].

Un SID est généralement défini comme étant « un regroupement de données orientées vers certains sujets, intégrées, dépendantes du temps, non volatiles, ayant pour but d'aider les gestionnaires dans leurs prises de décision » [3].

Un système décisionnel désigne un ensemble de moyens, d'outils et de méthodes qui permettent de collecter, consolider, modéliser et restituer les données, matérielles ou immatérielles, d'une entreprise en vue d'offrir une l'aide à la décision et de permettre aux responsables de la stratégie d'entreprise d'avoir une vue d'ensemble de l'activité traitée [4].

1.3. SI Opérationnels vs SI Décisionnels

Le système d'information (SI) joue le rôle de canal d'information et assure donc le lien entre le système pilote et le système opérant (Figure II-1). Le système opérant produit l'information, stockée dans le SI et la diffuse au système pilote pour que celui-ci soit informé du moindre détail sur l'activité opérationnelle. A son tour, le système pilote prend des décisions qu'il transmet via le SI au système opérant afin que celui-ci soit orienté pour améliorer l'activité quotidienne [5].

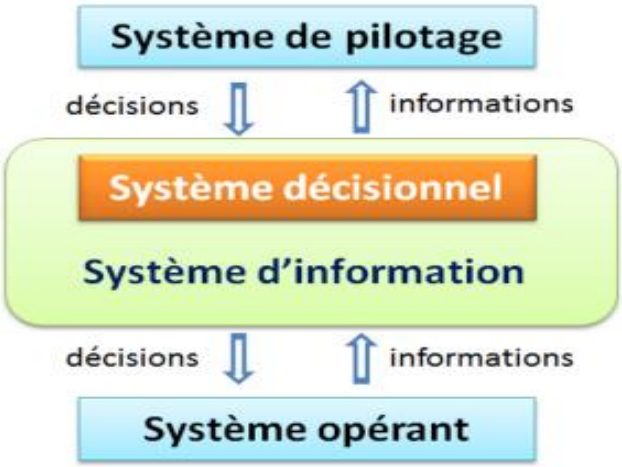


Figure II-1: SI opérationnel vs SI décisionnel [5].

Système d'information opérationnel (SIO) a pour objectif premier de servir le support à la réalisation des activités d'un ensemble de processus métier. Par exemple, un SIO dédié au processus de vente.

Système d'information décisionnel (SID) par opposition à un SIO dont l'objectif est l'exécution d'un processus métier, un SID a pour but l'évaluation de la performance des processus. Il a pour vocation de faciliter la prise de décision [6].

1.4. Architecture Décisionnelle

Les SID sont des outils du système d'information de gestion orientés vers la production de tableaux de bords et d'outils de pilotage. Ces systèmes visent à faciliter la prise de décision au sein d'une organisation [7].

Nous présentons dans la figure II-2 une architecture d'un système décisionnelle, Il regroupe l'ensemble des outils informatiques permettant d'extraire, de transformer et de charger (E.T.L.), de stocker, d'analyser et de restituer les données décisionnelles d'une organisation.

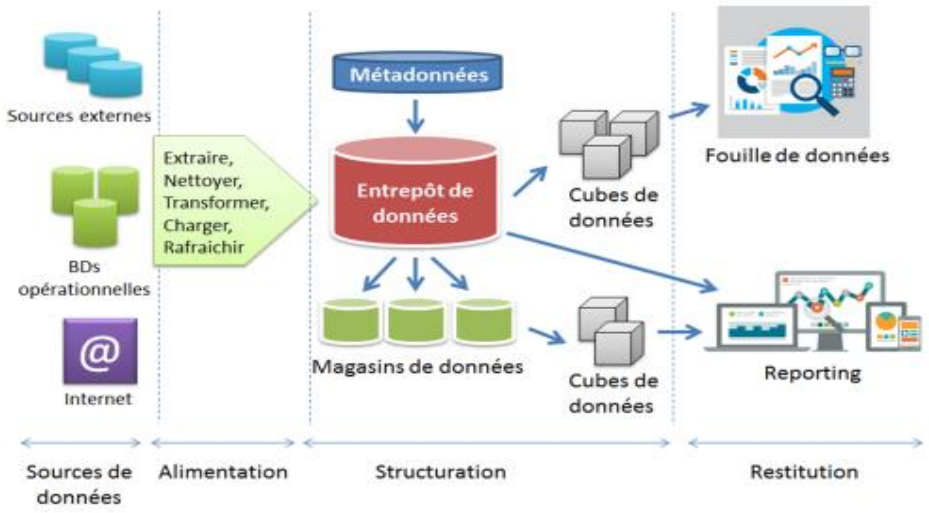


Figure II-2: Architecture typique d'un système décisionnel [5].

.

Sources de données sont distribuées, variées, et hétérogènes. Elles peuvent être internes ou externes à l'organisme.

Processus ETL (Extract, Transform, Load) permet le nettoyage, l'intégration et le chargement périodiques de toutes les données au sein de l'entrepôt de données (ED) nécessaires pour l'analyse.

Métadonnées sont des informations sur les données indispensables à une exploitation efficace d'un ED. Les métadonnées décrivent le schéma d'un ED et les données individuelles. Dans l'entreposage de données, les métadonnées sont classées d'après leur objet et le public auquel elles s'adressent.

L'entrepôt de données (DW : Data Warehouse) est le lieu de stockage centralisé, il contient les données orientées métier. Une fois ces données stockées dans le Data Warehouse, on va pouvoir créer des magasins de données appelés Datamarts [7].

Magasins de données (Datamarts) le magasin de données est un extrait de l'entrepôt de données relatif à un domaine fonctionnel particulier qui décrit la vision verticale au sein de l'organisation. La figure II-3 représente la vision verticale et la vision transversale de l'activité d'une organisation [5].

Figure II-3: Vision transversale et verticale de l'activité d'une organisation [5].

Cube OLAP (Online Analytical Processing) permet d'accéder rapidement et interactivement à des données stockées via une large variété de vues possible d'informations.

Outils de visualisation de données permettent de visualiser les données selon des axes d'analyses. L'information est visualisée via des interfaces interactives et fonctionnelles dédiées à des décideurs souvent non informaticiens (directeurs, chefs de services, . . .) [7].

1.5. Entrepôt de données

Un entrepôt de données (DW : Data Warehouse) est une base de données relationnelle pensée et conçue pour les requêtes et les analyses de données, la prise de décision et les activités de type Business Intelligence sont d'avantage que pour le traitement de transactions ou autres usages traditionnels des bases de données [9].

1.5.1. Les caractéristiques de l'entrepôt de données

Inmon dans le livre de '*The data warehouse and data mining*' définit l'entrepôt de données comme étant : « une collection de données thématiques, intégrées, historiées, et non-volatiles pour supporter le processus de prise de décision d'une organisation ».

- **Thématiques (orientées sujet) :** les données sont organisées par sujet ou fait.
- **Intégrées :** les données provenant de diverses sources hétérogènes, doivent être uniformes et intégrées dans l'ED.

- **Historiques :** l'évolution des données est essentielle pour la prise de décision, par exemple, en s'appuyant sur les évolutions passées pour prévoir les évolutions futures.
- **Non-volatiles :** les données insérées dans l'ED ne sont jamais modifiées ou effacées c'est-à-dire permanentes et ne peuvent pas être modifiées, elles sont conservées pour des analyses futures [3].

1.5.2. Modélisation d'un entrepôt de données

La modélisation des entrepôts de données s'appuie sur deux concepts fondamentaux : le concept de fait et le concept de dimension. Un fait représente un sujet d'analyse, caractérisé par une ou plusieurs mesures, qui ne sont autres que des indicateurs décrivant le sujet d'analyse. Ce fait est analysé selon des axes d'observation qui constituent également ses descripteurs.

Un entrepôt de données présente alors une modélisation dite "multidimensionnelle" puisqu'elle répond à l'objectif d'analyser des faits en fonction de dimensions qui constituent les différents axes d'observation des mesures. Ces dimensions peuvent présenter des hiérarchies qui offrent la possibilité de réaliser des analyses à différents niveaux de granularité (niveaux de détail). Ces concepts de base ont permis de définir trois schémas classiques reconnus comme relevant d'un niveau logique de conception [10].

A partir des faits et des dimensions, il est possible d'établir une structure de données simple qui correspond au besoin de la modélisation multidimensionnelle.

1.5.2.1. Le modèle de données en étoile (Schéma en étoile)

Est constitué d'une table de faits centrale et plusieurs tables de dimensions dé-normalisées. Chaque table de fait correspond à un fait conceptuel et inclut une clé primaire, des clés étrangères liées à des dimensions et une colonne pour chaque mesure du fait (Figure II -4).

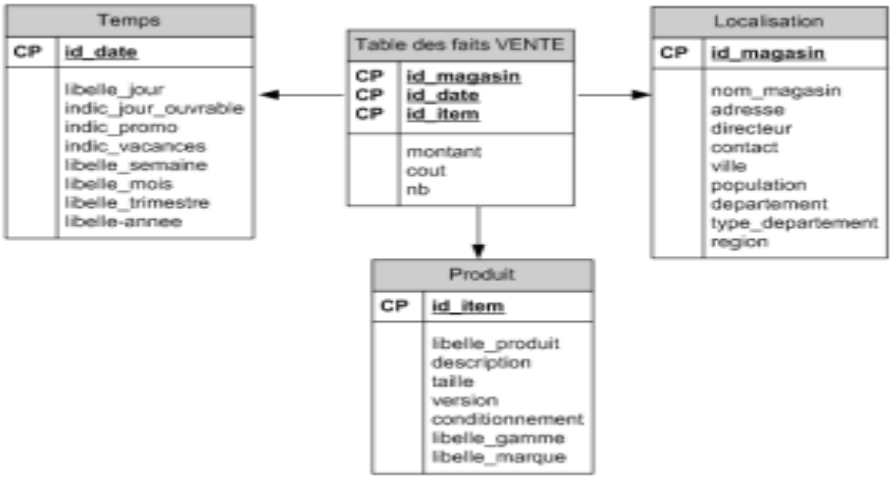


Figure II-4: Modèle en étoile [11].

1.5.2.2. Le modèle de données en flocons (Schéma en flocons)

Le modèle flocon en neige est une version normalisée du schéma en étoile. Il est composé d'une table de fait entourée par les différentes dimensions, qui sont décomposées en sous hiérarchies (chaque niveau est représenté dans une table différente) (Figure II-5).

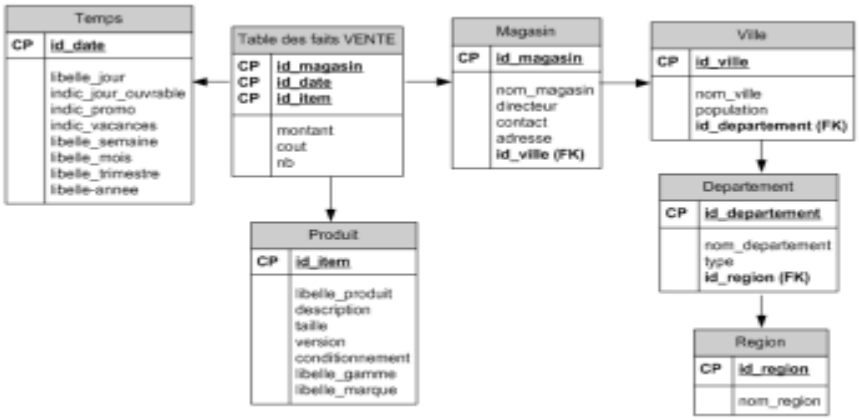


Figure II-5: Modèle flocon en neige [11].

Cette modélisation est plus facile à maintenir permettant d'éviter certaines redondances de données, par contre, elle est plus lente lors de l'interrogation suite à l'augmentation du nombre de jointure [7].

1.5.2.3. Le modèle de données en constellation (Schéma en constellation)

Est une généralisation du schéma en étoile. Une constellation regroupe plusieurs sujets d'analyse (faits) étudiés selon différents axes (dimensions) éventuellement partagés [12]. Dans la figure II-6, les tables de dimension (Temps et Produit) sont les plus communes entre les deux schémas en étoile.

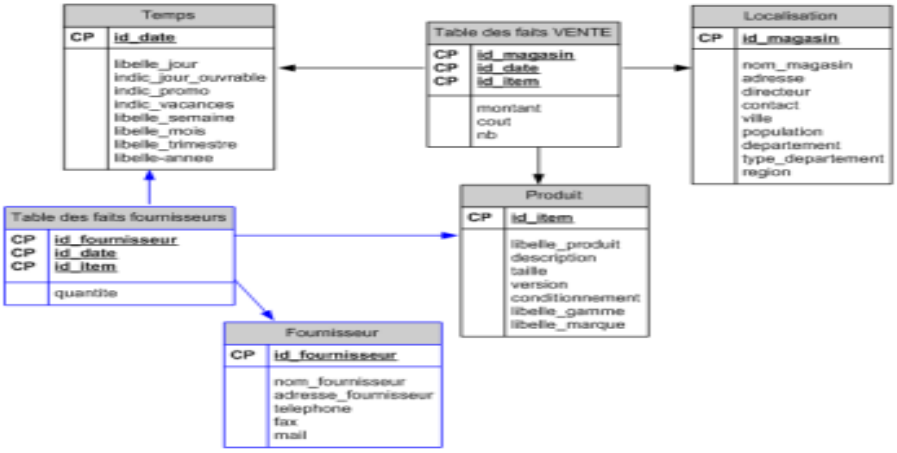


Figure II-6: Modèle en constellation [11].

1.6. L'alimentation de l'entrepôt de données

1.6.1. Processus ETL (Extract, Transform, Load)

L'alimentation d'ED se déroule par le processus d'alimentation de données (ETL). En effet, l'outil ETL (Extract, Transform and Load) est un progiciel de type médiateur (middleware) qui permet de réaliser un passage en masse d'information d'une base de données vers une autre.

Processus ETL consiste à rassembler les données des différentes sources, harmoniser et standardiser leurs formats, et finalement les acheminer vers la destination finale, le Data warehouse, qui possède un processus d'historisation et de versioning des données.

L'ETL est un processus dont le but est d'intégrer des données provenant de bases opérationnelles dans un entrepôt et/ou des datamarts. Ce processus se divise en trois grandes phases :

- **L'extraction** qui consiste à récupérer les données dans une ou plusieurs bases opérationnelles et à les stocker temporairement.
- **La transformation** dont le but est de convertir les données ainsi stockées vers une forme respectant les contraintes appliquées sur l'entrepôt (le nettoyage des données est parfois distingué de la transformation en tant qu'étape à part entière).
- **Le chargement** qui est l'action de transférer les données ainsi formatées vers l'entité de stockage [13].

2. Analyse des données en ligne (OLAP : On-line Analytical Processing)

2.1. Définition de l'OLAP

Un système d'analyse des données en ligne (OLAP : On-line Analytical Processing) est défini comme « une catégorie de logiciels axés sur l'exploration et l'analyse rapides des données selon une approche multidimensionnelle à plusieurs niveaux d'agrégation » [14] . Et comme un système décisionnel dans lequel les magasins de données suivent une organisation multidimensionnelle des données afin d'assurer un support efficace pour les analyses OLAP [15].

Le but d'OLAP est de distribuer les données aux utilisateurs sans les obliger à apprendre des complexes formules de programmation, d'interrogation ou même à ce qu'ils aient à programmer leurs tableurs.

2.2. OLTP (On-Line Transaction Processing) vs OLAP

La conception ou la modélisation d'un entrepôt est très différente de celle des bases de données des systèmes de type OLTP, Les bases de données de type OLTP (On-Line Transaction Processing) permettent de gérer des données variées et de réaliser des transactions sur ces données (lectures, mises à jour, etc.). La fouille de données à une forte relation avec OLAP, dont les objectifs sont similaires, mais il faut cependant souligner une différence essentielle [16].

2.3. La vue multidimensionnelle de OLAP

La modélisation multidimensionnelle et les technologies OLAP (On-Line Analytical Processing) permettent de réaliser une analyse rapide, intuitive et facile dans de grands volume de données. Ces données sont modélisées sous la forme d'hypercubes dans lesquels les dimensions constituent des axes d'analyse indépendants, et les sujets d'analyse, ou faits, sont caractérisés par des mesures qui sont pré-calculées à l'aide de

fonctions d'agrégations selon les différentes granularités définies par le schéma hiérarchique de chaque dimension [17].

La figure II-7 représente le modèle multidimensionnel d'un cube de données OLAP relatif aux ventes de livres. Notons que, dans cet exemple, les faits sont analysés par rapport à trois dimensions seulement. Théoriquement, la notion de cube de données se généralise en hyper-cube où le nombre d'axes d'analyse peut atteindre plusieurs dizaines.

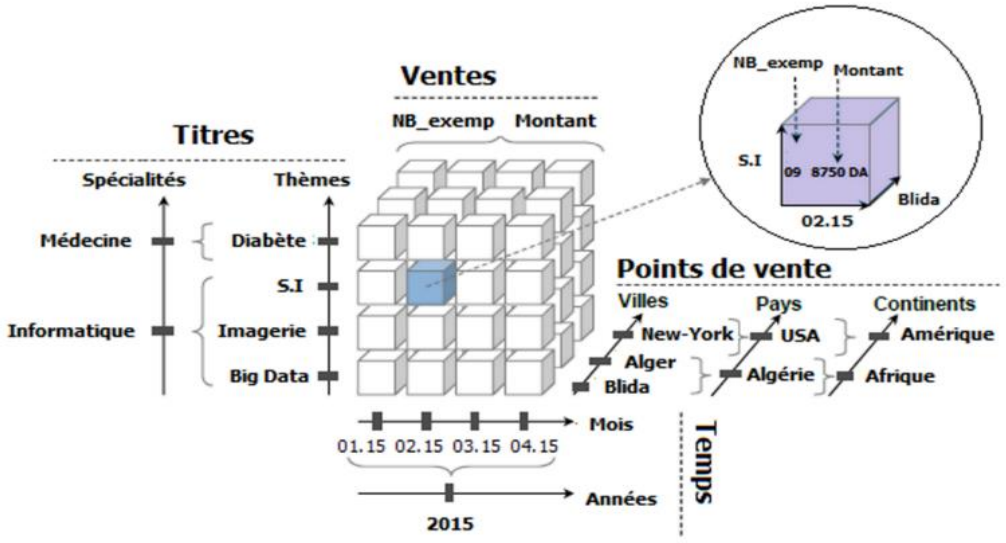


Figure II-7: Modèle multidimensionnel d'un cube de données OLAP relatif aux ventes de livres [5].

2.4. Différentes implémentations de systèmes OLAP

Dans la littérature des systèmes OLAP, il existe différentes implémentations, nous citons, dans les sections suivantes, les trois implémentations les plus connues.

2.4.1. Architecture d'analyse multidimensionnelle (MOLAP : (Multidimensional-OLAP)

Le Multidimensionnel OLAP consiste à utiliser un système multidimensionnel pur, qui gère des structures multidimensionnelles natives. Elles utilisent des tableaux à n

dimensions. L'accès aux données se fait directement dans le cube. Cela permet une rapidité d'accès à l'information mais augmente le temps de mise à jour (Figure II-8).

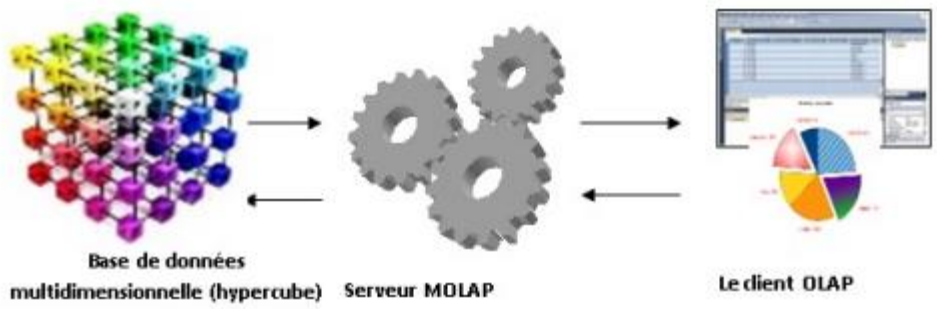


Figure II-8: Architecture d'un système MOLAP [7].

2.4.2. Architecture d'analyse relationnelle (ROLAP : Relational-OLAP)

Dans le Relationnel OLAP, les données sont stockées dans une base de données relationnelle. Un moteur OLAP permet de simuler le fonctionnement d'un hyper cube. Cela permet une facilité dans la mise à jour des données (Figure II-9).

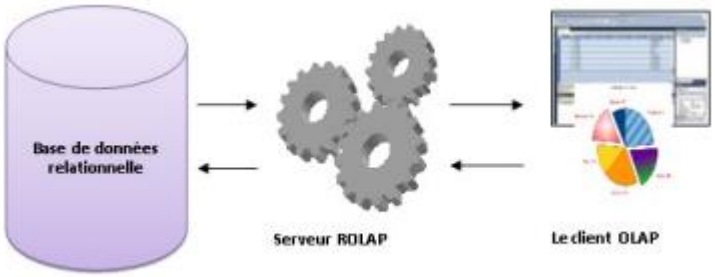


Figure II-9: Architecture d'un système ROLAP [7].

2.4.3. Architecture d'analyse hybride (HOLAP: (Hybrid-OLAP)

Est une hybridation entre ROLAP et MOLAP. Les tables de faits et dimensions sont stockées dans une base relationnelle standard tandis que le reste des données (les calculs) sont stockées dans une base multidimensionnelle (Figure II-10) [18].

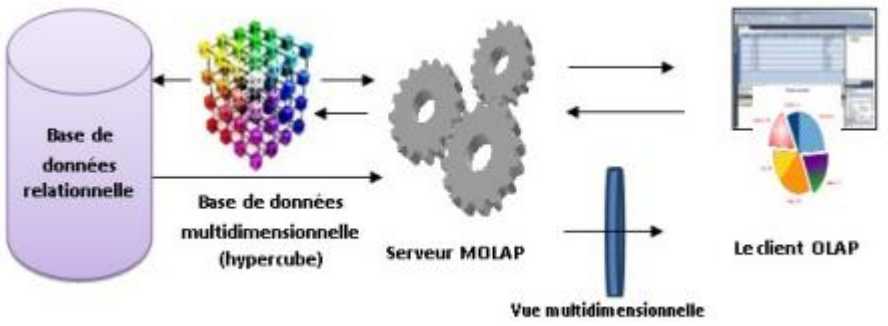


Figure II-10: Architecture d'un système HOLAP[7].

2.5. Les opérations OLAP

Les opérateurs OLAP (Roll-up, Drill-down, Slice, Rotate etc.) permettent de visualiser les mesures pour des ensembles de membres et des niveaux de granularité sélectionnés par l'utilisateur » [17].

Le serveur OLAP doit assurer un ensemble d'opérateurs permettant la navigation à travers les données entreposées. Voici les opérateurs typiques :

2.5.1. Opérations de forage.

Ces opérations permettent la navigation au moyen de la structure hiérarchique des axes d'analyses, afin de permettre l'analyse d'un indicateur avec plus ou moins de précisions. Le forage vers le haut (roll-up) consiste à analyser les données en fonction d'un niveau de granularité moins détaillé. L'inverse, le forage vers le bas (drill-down) permet d'analyser les données avec un niveau plus fin.

2.5.2. Opérations de sélection.

Ces opérations permettent à un utilisateur de restreindre l'ensemble des données analysées. La spécification d'une « tranche de cube » (slice) consiste à exprimer une restriction sur une des données de l'un des axes d'analyse. La spécification d'un « sous-cube » (dice) consiste à exprimer une restriction sur les données d'un indicateur d'analyse.

2.5.3. Opérations de rotation.

Ces opérations permettent la réorientation d'une analyse. Elles permettent de changer l'un des axes d'analyse en cours d'utilisation (rotation de dimension), de changer le sujet de

l’analyse (rotation de fait ou drill-across) et de changer de perspective d’analyse (rotation de hiérarchie) [19].

La figure II-11 illustre les opérateurs d’OLAP.

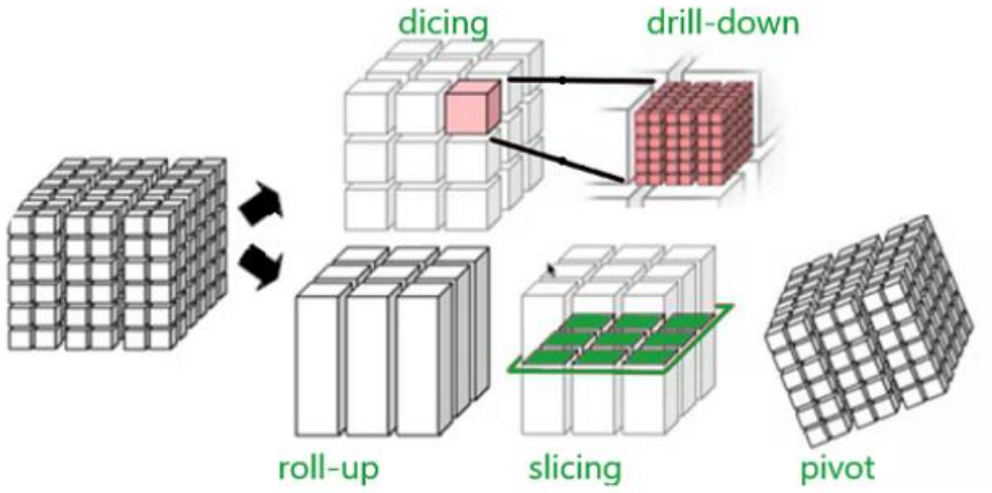


Figure II-11: Les opérations OLAP [8].

2.6. Conclusion

Dans ce chapitre nous avons tenté de présenter les principaux concepts sur les systèmes décisionnels. Il permet au décideur de définir et de mettre en œuvre des stratégies efficaces. Nous avons également exposé l’Analyse des données en ligne et ses différentes architectures et opérations.

III. Big Data et modèles NoSQL

Dans ce chapitre, nous allons présenter les notions de base sur les Big Data et les modèles de base de données NoSQL (clé-valeur, orienté colonnes, orienté document et orienté graphe).

---

1. Big Data

1.1. Introduction

Avec l'avènement d'Internet, du Wi-Fi, des smartphones et réseaux sociaux, de plus en plus des données sous différentes formes sont générées quotidiennement en quantités inhabituelles. Le big data est un terme qui décrit le grand volume de données - structurées et non structurées - qui engloutit une entreprise au quotidien. Mais ce n'est pas la quantité de données qui est vitale. C'est ce que les organisations font de ces données qui compte. Les big data peuvent être analysées pour en tirer des connaissances qui conduisent à de meilleures décisions et à des mouvements commerciaux stratégiques.

1.2. Définition de Big Data

Le Big Data est un ensemble de données de formats très variés, immense issues de diverses sources, qu'aucun des systèmes informatiques traditionnels n'est capable de stocker, de gérer, d'analyser et de traiter [20].

Cela a donné naissance à une nouvelle génération des technologies de l'information pour faciliter les traitements nécessaires pour l'analyse et l'extraction de la valeur des ensembles de données, à l'aide bien sur des matériels et des logiciels spécialisés. Le phénomène Big Data ne désigne pas seulement l'explosion de la masse de données produites, rendue possible par le développement des capacités de stockage et de diffusion de l'information sur tous les supports. Mais il fait référence également à un second phénomène qui est la capacité nouvelle à traiter les données.

1.3. Les différents formats de Big Data

Le terme Big Data est utilisé pour décrire un univers de très grands ensembles de données qui détiennent une variété de types de données[21]

Ces données peuvent prendre la forme de mail, de logs (journaux de connexion), des postes sur les réseaux sociaux, de images, de vidéos, de documents, de page web, de transaction, de coordonnées de géo localisation... [20].

Pour résumé on a trouvé 3 type de données :

1.3.1. Données structurées

Se trouvent, par exemple, dans les bases de données ou encore dans les langages informatiques. Ainsi, on les reconnaît au fait qu'elles sont disposées de façon à être traitées automatiquement et efficacement par un logiciel, mais non nécessairement par un humain. D'après Alain Garnier, I ‘auteur du livre L'information non structurée dans l'entreprise, « une information est structurée lorsqu'elle est répétable, systématique et calculable ». Il peut s'agir de formulaires, de factures, de fiches de paie, de libellés.

1.3.2. Données non-structurées

Par opposition à la catégorie précédente, les informations non structurées représentent l'ensemble des informations pour lesquelles il est impossible de retrouver une structure prédéfinie. Elles sont toujours destinées à des humains et il s'agit donc essentiellement de documents textes et multimédias, comme des lettres, des livres, des rapports, des collections d'images ou de vidéos, des brevets, des images satellites, des offres de services, des CV, des appels d'offres.

1.3.3. Données semi-structurées

Il est à noter que la frontière entre informations structurées et informations non structurées demeure assez floue et qu'il n'est pas toujours aisé de classer un document dans l'une ou l'autre des catégories. Dans ce cas précis, vous avez sans doute affaire a de l’information semi-structur*ée* [21].

1.4. Les caractéristiques de Big Data

IL existe trois composantes principales définissant une série de données comme étant du Big Data, dans la littérature on les appelle les « 3 V's » (voir la figure III-1) : le Volume, la Vitesse et la Variété.

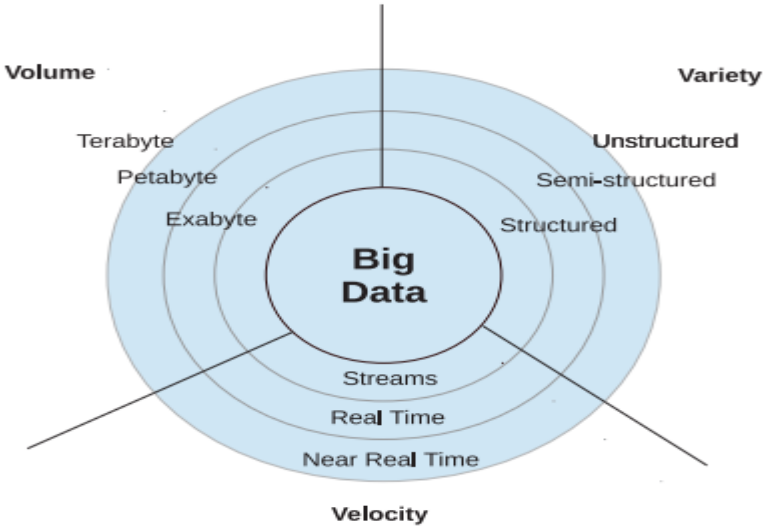


Figure III-1: les 3Vs de Big Data [22].

1.4.1. Le Volume

Due au grand nombre des dispositifs technologiques dont nous faisons usage aujourd'hui, la taille des informations que nous produisons en ce moment a crû énormément jusqu'à atteindre les magnitudes des péta ou exabytes. Si l'on rassemble tous l'information créée par l'humanité jusqu'à 2003, on obtient approximativement 5 exabytes de données, celle-ci est la même quantité d'information que nous produisons maintenant en deux jours.

1.4.2. La Vitesse (Vélocité)

Grâce à « l'Internet of Thing » (Internet des Objets) et la facilité pour avoir accès à l'internet, la vitesse avec laquelle humanité crée des informations croît exponentiellement il y a quelques années. Dans un jour normal, par exemple, le service de Google met sous monitorage plus de 7,2 milliards des pages web et ses serveurs doivent traiter plus de 20 pétabytes de donnés.

1.4.3. La Variété

Les données caractéristiques du Big Data provient d'une énorme quantité de sources : smartphones, ordinateurs, capteurs, codes de barres, etc. Cette diversité de sources fait qu'on ait une grande variété de données à exploiter, rendant impossible de les analyser avec des outils habituels. On peut classer toute cette information en trois énormes types de données : Données structurées, semi-structurées et non-structurées.

Certains auteurs définissent le concept de Big Data avec des caractéristiques supplémentaires telles que la valeur et la validité.

1.4.4. La Valeur

Il ne suffit pas d'avoir une énorme quantité donnée procédant de différentes sources à une grande vitesse de génération, il est nécessaire que cet ensemble d'information ait de la vraie valeur pour l'utilisateur (entreprise ou chercheur) après l'avoir exploitée. On dit donc que le Big Data prend une réelle importance selon la valeur qu'il peut avoir pour son exploiteur. En autres termes, Big Data égal à Big Valeur.

1.4.5. La Véracité (Validité)

La plupart des informations récoltées pour l'analyse Big Data est transmisses par l'internet, n'est pas un secret que toujours une partie des informations transmisses par cette voie se sont perdues ou endommages pendant les processus de décryptage. Il sert à rien le fait d'avoir une grande masse de donnés complexes à analyser si l'on n'est pas sûr de l’intégralité de ces données [23].

1.5. Enjeux du Big Data

Le Big Data représente aujourd'hui la nouvelle réalité de l'économie numérique. Il offre des possibilités énormes de développement et d'accroissement de productivité des entreprises. Une fois recueillies, stockées et utilisées de façon efficiente, ces grosses données" mesurées en trillions d'octets, générées chaque jour permettent aux entreprises d'avoir une véritable mine d'or :

- **La qualité des données :** pour réussir l'aventure Big Data, il faut travailler sur le « nettoyage » régulier de la donnée recueillie.

- **Le traitement des données :** l'enjeu d'un projet Big Data est de mettre en place des outils capables d'analyser et de décrire ces gros flux d'information qui se présentent souvent de manières différentes et sous divers formats.

- **La protection des données :** la majorité des données collectées par les entreprises proviennent des comptes des utilisateurs et sont donc des données privées. Le défi est de mettre en place des procédés d'anonymisation et de protéger ces données comme l'indique le règlement européen sur la protection des données en Europe (RGPD).

- **L'image de la donnée :** le défi est de faire du visuel un moyen de répondre aux différents besoins et attentes des utilisateurs.

- **L'humanité des données :** l'enjeu du Big Data est aussi de respect derrière toutes ces données [24].

1.6. Applications du Big Data

Voici quelques exemples d'applications Big Data :

- **E-santé :** l'analyse avancée des ensembles de données médicales a de nombreuses applications bénéfiques. Il permet de personnaliser les services de santé (par exemple, les médecins peuvent surveiller les symptômes des patients en ligne afin d'ajuster la prescription) ; adapter les plans de santé publique en fonction des symptômes de la population, de l'évolution de la maladie et d'autres paramètres ; il est également utile d'optimiser les opérations hospitalières et de diminuer les dépenses de santé.

- **Internet des objets (IoT) :** représente l'un des principaux marchés des applications Big Data. En raison de la grande variété d'objets, les applications de

l'IoT sont en constante évolution. De nos jours, il existe diverses applications Big Data pour les entreprises logistiques. En fait, il est possible de suivre la position des véhicules avec des capteurs, des adaptateurs sans fil et un GPS. Ainsi, ces applications basées sur les données permettent aux entreprises non seulement de superviser et de gérer les employés, mais aussi d'optimiser les itinéraires de livraison. C'est en exploitant et en combinant diverses informations, y compris l'expérience de conduite passée. La ville intelligente est également un domaine de recherche à chaud basé sur l'application de données IoT.

- **Services publics :** les services publics tels que les organisations d'approvisionnement en eau placent des capteurs dans les canalisations pour surveiller le débit d'eau dans les réseaux complexes d'approvisionnement en eau. Il est rapporté dans la presse que le Bangalore Water Supply and Sewage Board met en œuvre un système de surveillance en temps réel pour détecter les fuites, les connexions illégales et contrôler à distance les vannes pour assurer un approvisionnement équitable en eau dans différentes zones de la ville. Cela permet de réduire le besoin d'opérateurs de vannes et d'identifier et de réparer en temps opportun les conduites d'eau qui fuient [23].

2. Modèles de données NoSQL (Not Only SQL)

2.1. Définition

Le NOSQL est un modèle spécifique de bases de données, permettant de stocker et de récupérer les données après restructuration, en utilisant une extension des différentes techniques de celles connues dans les bases de données relationnelles. Les développeurs de nos jours ont tendance à utiliser ce type de bases de données pour la simplicité de leur implémentation et leur évolutivité sans limites (horizontalement, à travers de nouvelles colonnes) [25].

Les systèmes de gestion de bases de données relationnelles classiques (SGBDR) sont utilisés pour gérer des données d'entreprise qualifiées, mais ne sont pas habilités à

stocker des données à grande échelle avec un traitement rapide. Les bases de données NOSQL (not only SQL) offrent une nouvelle approche du stockage de données plus flexible, plus évolutive et moins vulnérable aux défaillances du système. Les bases de données NOSQL sont spécialisées dans le stockage de données non structurées et fournissent des performances rapides. Les bases de données NOSQL les plus populaires incluent MongoDB, Redis, Cassandra, Hbase et bien d'autres [26].

2.2. Comparaison entre base de données NoSQL et base de données SQL

SQL et NoSQL ont chacun son lot d'avantages et de désavantages, aucune des deux solutions n'est donc meilleure que l'autre. En fonction des types de besoins que peut avoir une entreprise par exemple, une solution SQL pourrait être plus avantageuse qu'une solution NoSQL, et vice-versa.

Dans cette partie on va citer quelques points forts de NoSQL qui n'existent pas dans SQL :

- **Plus évolutif :** NoSQL est plus évolutif. C'est en effet l'élasticité de ses bases de données NoSQL qui le rend si bien adapté au traitement de gros volumes de données. Au contraire, les base de données relationnelles ont souvent tendance à utiliser la scalabilité verticale, qui consiste à augmenter les capacités du serveur (plus d'espace, ajout de RAM, augmentation de la puissance du processeur) quand celui-ci atteint ses limites.
- **Plus flexible :** N'étant pas enfermée dans un seul et unique modèle de données, une base de sonnées NoSQL est beaucoup moins restreinte qu'une base SQL.
- **Plus économique :** Les serveurs destinés aux base de données NoSQL sont généralement bon marché et de faible qualité, contrairement à ceux qui sont utilisés par les bases relationnelles.
- **Plus simple :** Les bases de données NoSQL ne sont pas forcément moins complexes que les bases relationnelles, mais elles sont beaucoup plus simples à

déployer. La façon dont elles ont été conçues (réparation automatique, données distribuées, simplicité des modèles de données) [27].

2.3. Modèles de données NoSQL

Les bases de données NoSQL sont donc une catégorie de base de données qui n'est plus fondée sur l'architecture classique des bases relationnelles, et non pas un type à part entière [28].

Il existe différentes manières de structurer les données mais l'ensemble des solutions développées peut être divisé en quatre modèles : le modèle orienté clé-valeur, modèle orienté colonnes, le modèle orienté documents et le modèle orienté graphes.

2.3.1. Les bases de données clé-valeur

Les bases de données de type clé-valeur sont basées sur un principe simple : chaque valeur stockée est associée à une clé unique. C'est uniquement par cette clé qu'il sera possible d'exécuter des requêtes sur la valeur. Ce type de base de données, permettant un stockage de données sans schéma. De plus, l'implémentation est très simple :

- La clé est de type string,
- La valeur peut être : String, List, hash, set, sortedSet Exemple de compréhension [28].

La figure III-2 représente le modèle clé valeur.

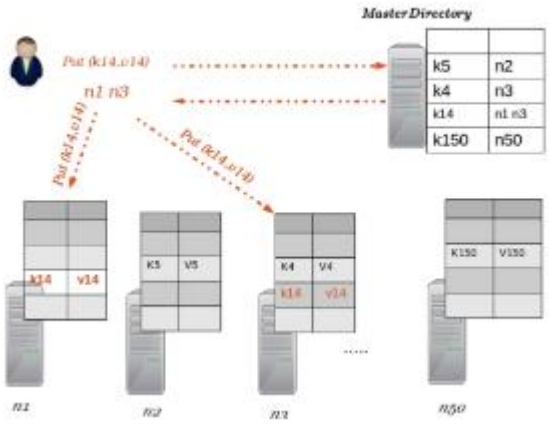


Figure III-2: Base de données clé-valeur [28].

2.3.2. Modèle orientées colonnes

C'est le modèle le plus proche du relationnel. Les bases de données orientées colonnes vont stocker les données de façon à ce que toutes les données d'une même colonne soient stockées ensemble. Ces bases peuvent évoluer avec le temps, que ce soit en nombre de lignes ou en nombre de colonnes. Autrement dit, et contrairement à une base de données relationnelle où les colonnes sont statiques et présentes pour chaque ligne, celles des bases de données orientées colonnes sont dite dynamiques et présentes donc uniquement en cas de nécessité De plus le stockage d'un « nul » est 0.

Important : La valeur NULL coûtera la place en mémoire.

Les bases de données colonnes ont été pensées pour pouvoir stocker plusieurs millions de colonnes et se relèvent donc parfaites pour gérer le stockage dit « one-to-many » [28].

La figure III-3 représente le modèle orientée colonnes.

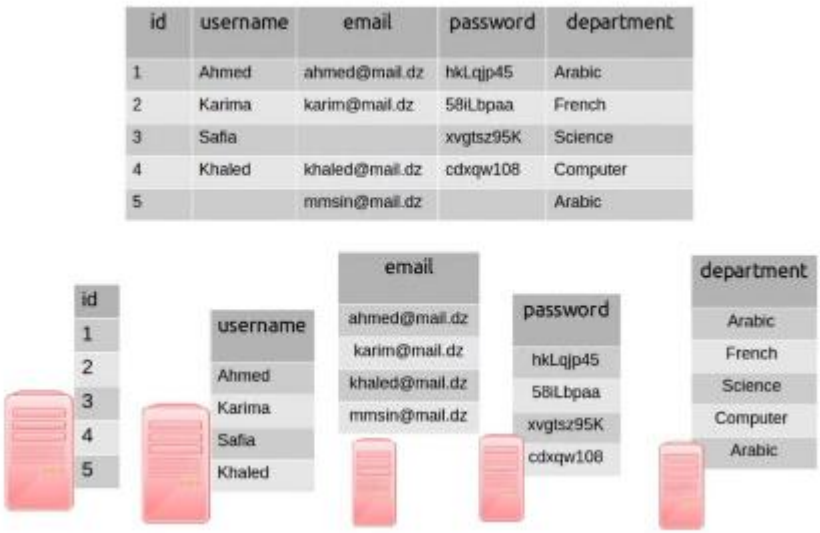


Figure III-3: Base de données orienté colonne [28].

Les principaux concepts associés :

- **Colonne :**
  - Entité de base représentant un champ de donnée
  - Chaque colonne est définie par un couple clé / valeur

- Une colonne contenant d'autres colonnes est nommée super colonne.

➢ **Famille de colonnes :**

- Permettent de regrouper plusieurs colonnes (ou super colonnes)
- Les colonnes sont regroupées par ligne
- Chaque ligne est identifiée par un identifiant unique (assimilées aux tables dans le modèle relationnel) et sont identifiées par un nom unique.

➢ **Super colonnes :**

- Situées dans les familles de colonnes sont souvent utilisées comme les lignes d'une table de jointure dans le modèle relationnel.

2.3.3. Modèle orientées documents

Ce modèle se base sur le paradigme clé-valeur précédent, cependant dans ce nouveau modèle la valeur est un document de type JSON ou XML. Ainsi, dans de modèle, les données sont stockées à l'intérieur de document. Un document peut être vu comme un n-uplet d'une table dans le monde relationnel, à la différence toutefois que les documents peuvent avoir une structure complètement différente les uns des autres. L'avantage est de pouvoir récupérer, via une seule clé, un ensemble des informations structurées de manière hiérarchique. La même opération dans le monde relationnel impliquerait plusieurs jointures [27].

La figure III-4 représente le modèle orientée document.

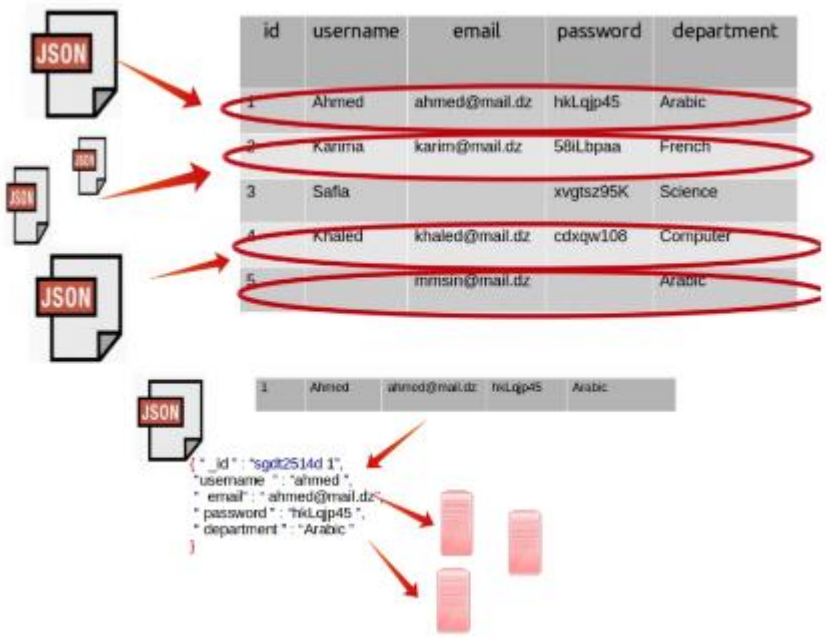


Figure III-4: Base de données orienté document [28].

2.3.4. Modèles orienté graphes

Ce modèle qui repose sur la théorie des graphes, permet de représenter les données sous la forme de graphes. Le modèle s'appuie sur la notion de nœuds, de relations et de propriétés qui leur sont rattachées. Les entités sont alors les nœuds du graphe et les relations que partagent les entités sont alors des arcs qui relient ces entités. Ce modèle est notamment adapté au traitement des données des réseaux sociaux. Notons que les systèmes NoSQL orienté graphe trouvent un certain intérêt pour des applications dans le domaine du Web sémantique, dans la gestion de bases de données de triplets RDF (triple-stores), permettant de stocker des connaissances ou ontologies, un triplet étant une arête d'un graphe [27]. La figure III-5 représente le modèle orientée graphe.

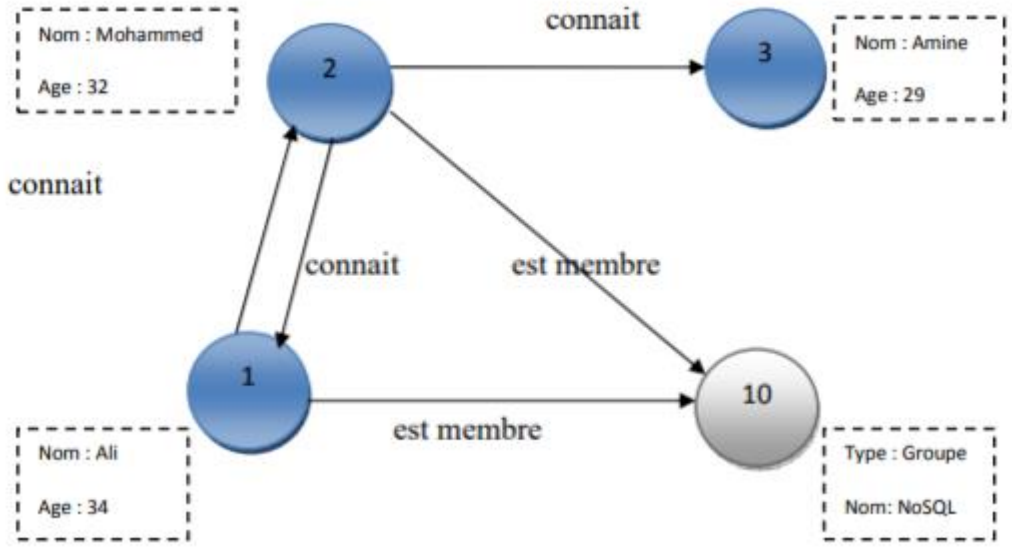


Figure III-5: Base de données orienté graphe [27].

2.4. Conclusion

Le big data est plus complexe en volume, variété et vélocité des données, Un problème se pose quant au stockage et à l'analyse des données. NoSQL a été utilisée ces dernières années pour résoudre ce problème, et c'est ce que nous avons vu dans ce chapitre. Les bases de données NoSQL peuvent prendre en charge des données structurées ou non structurées de la même manière. Ces bases de données sont idéales pour stocker et analyser des données volumineuses.

IV. Data Lake

Dans ce chapitre, nous présentons les concepts de base des Data Lakes, plus particulièrement, leur définition, leurs fonctionnalités, leur architecture, leurs caractéristiques et enfin leurs avantages.

---

1. Introduction

Le volume des données numériques se multiplie chaque année, la plupart de ces données (jusqu'à 90%) sont non structurées ou semi-structurées, ce qui présente un double défi pour trouver une solution efficace de stocker toutes ces données et disposer en permanence des capacités nécessaires à leur traitement rapide. Le Data Lake est capable de relever ces deux défis.

2. Définition

Un lac de données est une solution d'analyse de données volumineuses qui ingère des données brutes structurées de manière hétérogène provenant de diverses sources (locales ou externes à l'organisation) et stocke ces données brutes dans leur format natif, permet de traiter les données selon différentes besoins et fournit des accès aux données disponibles à différents utilisateurs (data scientistes, analystes de données, professionnels de la BI, etc.) pour l'analyse statistique, la Business Intelligence (BI), l'apprentissage automatique (ML), etc., et gouverne les données pour assurer la qualité, la sécurité et le cycle de vie des données [29].

Une définition plus consensuelle consiste à voir un lac de données comme un dépôt central où des données de tous formats sont stockées sans schéma strict. Cette définition est basée sur deux caractéristiques clés du lac de données la variété des données et l'approche schema-on-read. La variété des données désigne le fait d'intégrer des données de tous types et hétérogènes. La propriété schema-on-read consiste à définir le schéma des données seulement lors de leur analyse [30].

3. Fonctionnalités des Data Lake

Le lac de données à un grand potentiel. Il peut être utilisé pour effectuer des traitements analytiques qui n'ont jamais été réalisés auparavant. Des gouvernements aux petites entreprises, il peut également être utilisé pour identifier, analyser et même prédire des modèles importants qui sont passés inaperçus jusqu'à présent [1].

Voici quelques-unes des fonctionnalités du lac de données

3.1. L'acquisition

Cette fonctionnalité est celle par laquelle les sources de données, internes, externes, structurées, non structurées vont être "ingérées" par le lac de données. On peut faire l'analogie avec le système décisionnel, et son composant ODS (Operation Data Store).

3.2. Catalogue de données

Chaque fois que des données provenant de différentes sources, contextes et avec divers schémas sont rassemblées, des métadonnées sont nécessaires pour suivre ces données. Cela s'applique également aux lacs de données, où la gestion des métadonnées est un élément crucial. Les métadonnées capturent des informations sur les données réelles, par exemple des informations de schéma, une sémantique ou une lignée. Ils garantissent que les données peuvent être trouvées, fiables et utilisées. Il existe un grand nombre d'approches différentes pour la gestion des métadonnées. Selon la littérature Data Lake, les catalogues de données sont utilisés pour stocker les métadonnées.

3.3. Le stockage

Ce composant est celui où les données acquises et cataloguées vont pouvoir être stockées avant, pendant et après (potentiellement) la phase d'exploitation (ou d'exploration) de ces données. C'est à ce composant qu'est souvent réduit un lac de données associé à la technologie Apache Hadoop. Russom emploie d'ailleurs le mot "Hadoop Data Lake". Mais le lac de données ne doit pas être réduit à cette fonction (ou composant) mais bien être considéré comme un concept d'architecture.

3.4. L’exploitation ou l’exploration

Ce composant est l’objectif des lacs de données : permettre l’exploitation et l’exploration des données de l’organisation.

3.5. La gouvernance

La gouvernance du lac de données a en son cœur la gestion du catalogue des métadonnées mais ce n’est pas le seul élément nécessaire pour gouverner un lac de données. La gestion de la sécurité, du cycle de vie et de sa qualité sont les fondamentaux[23].

4. Architecture de data lake

Le concept de Data Lake reste relativement floue à l’heure actuelle, mais est tout de même caractérisé par certains aspects qui font l’unanimité.

Data Lakes stocker en un endroit unifié toutes les données utiles, peu importe le format de ces données. Un Data Lake est donc un endroit où mettre toutes les données que les entreprises (peuvent) vouloir rassembler, stocker, analyser et transformer en idées et en actions, y compris des données structurées, semi-structurées et non structurées [31].(voir la figure IV-1)

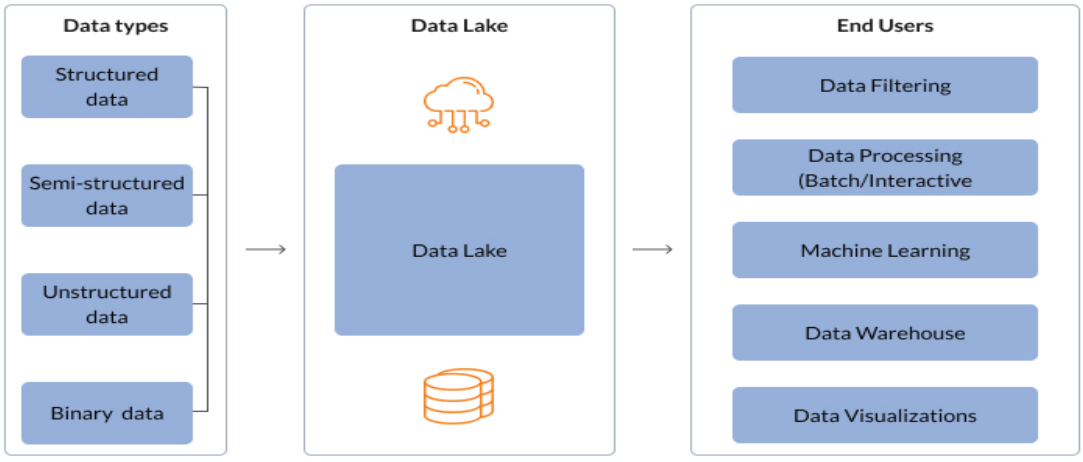


Figure IV-1: Exemple d’architecture du data lake.

Contrairement aux entrepôts de données (*DW : Data Warehouse*) basés sur un schéma en écriture (*schema on write*), les data lakes sont basés, quant à eux, sur des schémas en lecture (*schema on read*) puisque les données collectées à partir de diverses sources hétérogènes sont ingérées en préservant leur format natif et laissant le soin aux applications d'opérer les transformations nécessaires selon leurs objectifs de traitements. Comme la figure IV-1 illustre les utilisateurs finaux du Data Lake peuvent être Machine Learning , Data Warehouse , Data Processing ...etc.

5. Les caractéristiques d'un data lake

Les caractéristiques d'un lac de données incluent :

- Un catalogue de métadonnées qui assure la qualité des données
- Une politique et des outils de gouvernance des données
- L'ouverture à tous types d'utilisateurs
- L'intégration de données de tous types
- Une organisation logique et physique
- Le passage à l'échelle [30].

6. Approche ELT

Le concept d'ELT est assez lié à la philosophie des Data Lakes. Dans l'approche ELT, une fois les données extraites, elles sont immédiatement chargées dans un référentiel de données unique et centralisé. Ce référentiel peut bien sûr être un Data Lake. D'ailleurs, la stratégie ELT est aussi née grâce à l'évolution des technologies d'infrastructure, pouvant désormais prendre en charge de grands volumes de stockage et des calculs évolutifs. Par conséquent, un pool de données étendu et un traitement rapide sont virtuellement illimités pour la maintenance de toutes les données brutes extraites. De cette manière, l'approche ELT offre une alternative moderne à ETL. Cependant, le monde des Data Lakes, avec toutes les avancées périphériques, continue d'évoluer et rapidement. A l'heure actuelle, les outils pour soutenir le processus ELT ne sont pas toujours entièrement développés pour faciliter la charge et le traitement d'une grande quantité de données, mais plutôt dans une optique de preuve de concept. L'avantage de la stratégie ELT est de

permettre un accès illimité à toutes les données, à tout moment, et d'économiser les efforts et le temps des développeurs pour les utilisateurs et les analystes BI [31]. (Voir la figure IV-2).

Figure IV-2: ETL vs ELT [32].

7. Les avantages des data lakes

Les Data Lakes permettent de stocker dans un écosystème unique toutes les données que l'entreprise juge utile, et ainsi d'effacer les cloisons entre les données. Tous les types de données doivent pouvoir être ingérés dans les Data Lakes.

- **Analyses plus riches et performantes**

Etant donné que les données sont stockées au même endroit, les possibilités de croisement de données, et donc d'analyses, sont grandement améliorées. En effet, cela permet de s'affranchir de beaucoup de barrières techniques et de format, qui auraient été nécessaires avec des données distribuées dans des systèmes différents. Par ailleurs, les Data Lakes étant dotés d'une grande capacité de calcul, permettent aussi de produire des analyses beaucoup plus poussées et efficaces. En utilisant certaines technologies, il est même possible de faire aisément du calcul temps réel, sur les données elles-mêmes ingérées en temps depuis des sources opérationnelles vers le Data Lake.

- **Flexibilité**

Par ailleurs, les Data Lake offrent une grande flexibilité au niveau de l'infrastructure. Ils sont hautement évolutifs et flexibles, et cela de manière relativement peu complexe, en utilisant les outils adaptés et les bonnes compétences. Par exemple, pour les Data Lakes basés sur la solution Hadoop, il est très aisé de rajouter des nœuds avec du stockage et de la puissance de calcul en plus. Il est aussi aisé d'ajouter de nouveaux outils et services, permettant toujours d'exploiter de manière encore plus diversifiée les données du Data Lake [31].

8. Conclusion

Dans ce chapitre, nous avons expliqué le concept de Data Lake, ses fonctions, son architecture et ses propriétés, Ainsi que l'approche ELT. Donc un Data Lake est un élément fondamental dans une architecture orientée Data mais surtout dans la gouvernance de vos données de référence. Ce n'est pas uniquement un espace de stockage de données brutes, c'est aussi la source pour le traitement de ces données par des applications tierces.

Deuxième partie

Etat de l’art

V. Travaux Connexes

Ce chapitre présente une synthèse sur la littérature ayant trait au l'analyse des données en ligne OLAP. La présentation de ce chapitre est comme suit : Introduction, présentation des approches et nous concluons par une discussion qui apporte sur la comparaison sur les approches en basant sur certains critères.

---

1. Introduction

Dans les systèmes décisionnels le cube OLAP est une méthode de stockage de données sous forme multidimensionnelle, dédiée, généralement, pour des fins d'analyse et de reporting. L'OLAP est dédié, principalement, aux Bases de données relationnelles, en revanche la naissance de Big Data au début de 21eme siècle à apporter des nouvelles bases des données : NoSQL DB. Par conséquent le centre d'intérêt des chercheurs est orienté vers l'adaptation des cubes OLAP au Bases des données NoSQL. Dans ce contexte, nous citons des travaux tel que : Khalil, A., & Belaissaoui, M. [33], Davardoost, F., and all [34], Chevalier, M., and all [35].

Pour discuter différents aspects de l'analyse des données en ligne structurées dans un modèle NoSQL il est essentiel de parler de certains critères comme :

- Le schéma NoSQL
- Le Schéma (en étoile, en flocon, en constellation)
- Mapping : passage d'un modèle en étoile vers un modèle NoSQL
- Techniques d'agrégation

D'autres critères peuvent être ajoutés s'ils satisfont les critères de complexité et de performance.

2. Approche de Khalil, A., & Belaissaoui, M. [33]

L'objectif de cette contribution est de proposer une nouvelle façon de modéliser le cube OLAP et de créer des moteurs OLAP basés sur une base de données de clé valeur,

et de répondre à la question de savoir comment étendre les opérations OLAP de base sur un modèle clé valeur de NoSQL !

Cette contribution représente le cube de données sous la base de données NoSQL en deux modèles de données, chacun d'eux diffère en termes de structure et de complexité.

2.1. Le modèle aplati et hiérarchie

En utilisant des hiérarchies d'enregistrements et des tables imbriquées pour stocker des mesures et des dimensions agrégées. (Voir la figure V-1)

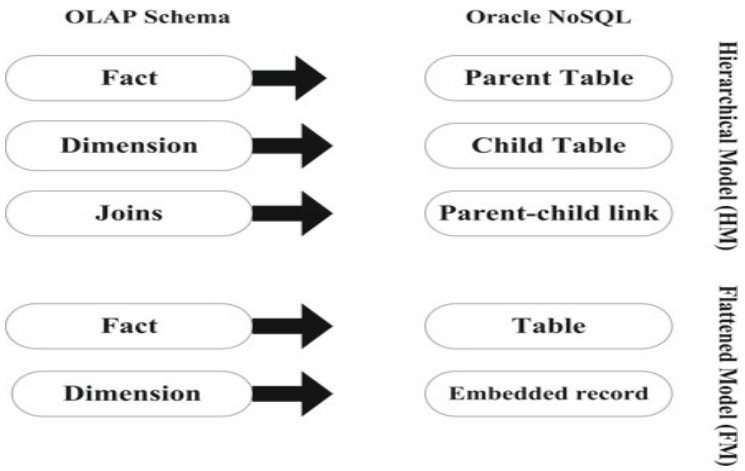


Figure V-1: Transformation rules from OLAP schema to Oracle NoSQL.

2.2. L'opérateur KV

En utilisant la structure simple clé-valeur Comme il n'y a pas d'hypothèses concernant les données stockées dans la clé et la valeur, en profitant de cette souplesse de schéma pour représenter les cellules de cube dans la valeur et les dimensions dans la clé.

a) De l'entrepôt de données relationnel au cube clé-valeur

Le schéma en étoile comporte deux composantes principales : la valeur agrégée et les axes d'analyse (dimensions).

Un attribut de dimension peut être modélisé dans une structure clé-valeur comme suit :

$$\langle Name^D \rangle : \langle Key^D \rangle : \langle Param^D \rangle \Rightarrow Value^{Param}$$

Où :

- NameD est le nom de la dimension.

- KeyD est la clé de la dimension.
- ParamD est un attribut de la dimension appelé paramètre.
- ValueParam est la valeur du paramètre.

Un cuboïde OLAP peut être représenté par le schéma suivant :

$$\langle Name^{C} \rangle : \langle Key^{D_i} : \ldots : Key^{D_n} \rangle : \langle Name^{M} \rangle \Rightarrow Value^{M}$$

Où :

- NomC est le nom du cuboïde.
- Clé Di : ... : Clé Dn est un ensemble de clés de dimension.
- NomM est le nom de la valeur agrégée appelée mesure.
- ValueM est la valeur de la mesure.

b) KV-Opérateur pour le stockage Clé Valeur

Cet opérateur permet de calculer des cubes OLAP à partir d'entrepôts de données implémentés sous key value store en deux étapes.

Etape 1 : Vue sur les attributs (dimensions, Mesures) nécessaires au calcul de l'OLAP Le résultat obtenu est une relation R, cette dernière est un résultat intermédiaire pour constituer toutes les parties du cube OLAP.

Etape 2 : Chaque dimension clé de la relation R est hachée avec les valeurs qui la composent pour obtenir la liste des positions de ces valeurs, ces listes sont binaires, et peuvent correspondre à "1" ou "0" ; le "1" indique que la valeur de hachage existe à cette position et "0" sinon. La valeur de hachage existe à cette position et "0" sinon. Ces listes permettent d'avoir les agrégats de chaque dimension séparément (Figure V- 2**).**

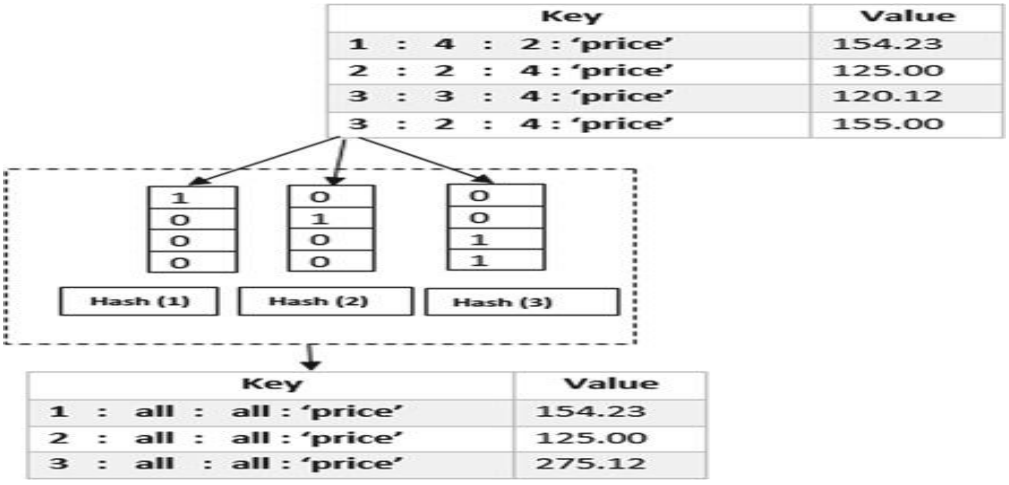


Figure V-2: Hashing positions and performing aggregations.

Pour la réalisation de cette approche, les machines physiques utilisés possèdent un processeur Intel Core i7 avec 16Go de RAM fonctionnant avec la distribution Linux Debian. L'architecture logiciel est décrite dans la figure V-3.

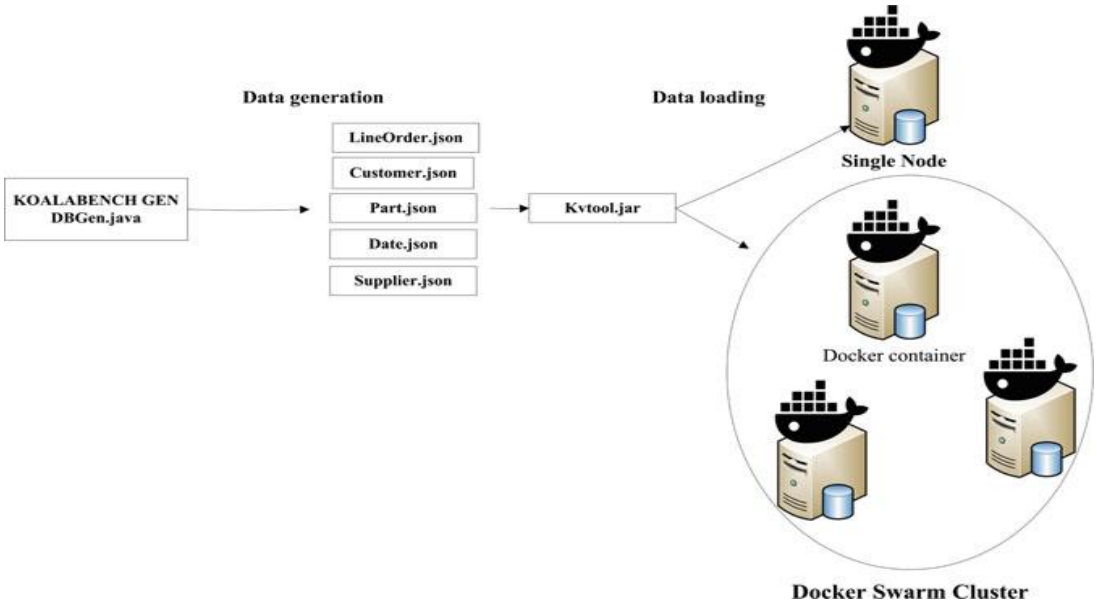


Figure V-3: L'architecture logicielle.

2.3. Synthèse

Pour les deux modèles logiques proposé dans la première approche appelant modèle et modèle hiérarchique. Les expériences montrent que le modèle aplati offre de meilleures performances de requête mais nécessite plus d'espace disque. Ils basent dans cette approche sur Oracle NoSQL, une solution qu'elle n'est pas standard.

La deuxième approche, définit des règles de transformation de l'entrepôt de données relationnelles vers un cube OLAP NoSQL et implémentons un algorithme pour effectuer des opérations OLAP de base sur le cube. Il s'agit aussi d'un opérateur d'agrégation pour calculer le cube OLAP à partir de l'entrepôt de données implémenté sous stockage clé-valeur.

L'objectif n'est pas de mesurer les performances du produit de base de données utilisé dans cette expérience, mais d'évaluer les performances de l'approche de modélisation sous le stockage clé-valeur.

3. Approche de Davardoost, F., & al.[34]

Il s'agit dans cette approche, de proposer un modèle d'extraction de cubes OLAP à partir d'une base de données NoSQL orientée-documents.

Les données sont stockées dans une collection de documents avec différents attributs dans le modèle de base de données NoSQL orienté documents. Une méthode évolutive de classificateur Naive Bayes a été utilisée à cette fin.

Le principe de base est que les attributs sont conditionnellement indépendants les uns des autres. Le stockage et le traitement peuvent être complexes lorsque l'ensemble de données est important ; pour surmonter ces problèmes, un système de traitement parallèle MapReduce avec Naive Bayesian est utilisé. NBMR est évolutif dans les grands ensembles de données. Le champ d'application de cette étude est décrit dans la figure V-4.

Figure V-4: Le champ d'application de l'étude.

Les trois phases préparation, Naive Bayes, et l'algorithme NBMR sont inclus dans la solution proposée.

3.1. Première phase (phase de préparation)

La base de données orientée document NoSQL est d'abord utilisée comme donnée d'entrée dans le système, et elle sera convertie en matrice. Une matrice D avec des éléments 0 et 1 est créée. Les lignes de la matrice D représentent les documents, les colonnes représentent les attributs. Si l'attribut est nul, la valeur est fixée à 0 ; sinon, la valeur est fixée à 1. La matrice D est créée sur la base de ces éléments.

3.2. Deuxième phase (Naive Bayes)

La matrice D est soumise à l'algorithme Naive Bayes. Les parties de Naive Bayes sont présentées en détail dans la figure V-5.

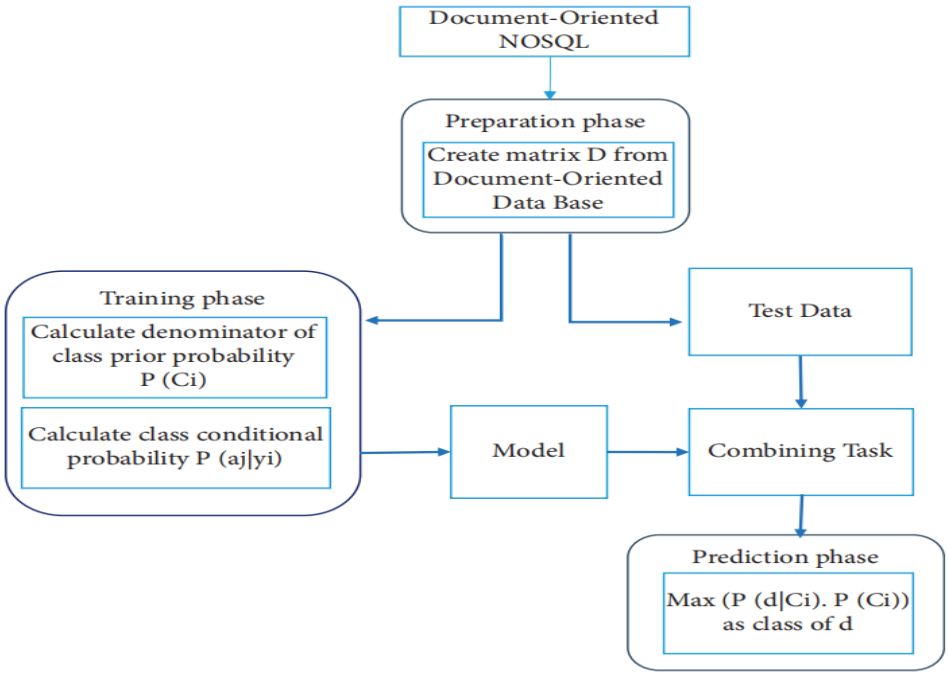


Figure V-5: Deuxième phase du modèle proposé.

3.3. Troisième Phase (NMBR)

Cette section explique comment mettre le modèle proposé. Comme le calcul des probabilités conditionnelles dans la troisième phase prend beaucoup de temps, le modèle de programmation MapReduce est utilisé pour paralléliser et accélérer le processus. L'algorithme proposé, NBMR, est un hybride de la classification de Naıve Bayes (NBC) et du modèle de programmation distribué MapReduce. Le classificateur Naıve Bayes attribue un document à la classe ayant la probabilité la plus élevée, telle que déterminée par la règle de Bayes, et MapReduce permet le calcul parallèle, ce qui est idéal pour les grandes quantités de données. Le maître, l'entraînement et la prédiction sont les trois tâches que nous avons utilisées pour mettre en œuvre.

La figure V-6 illustre le processus de mise en œuvre du modèle proposé dans cette approche.

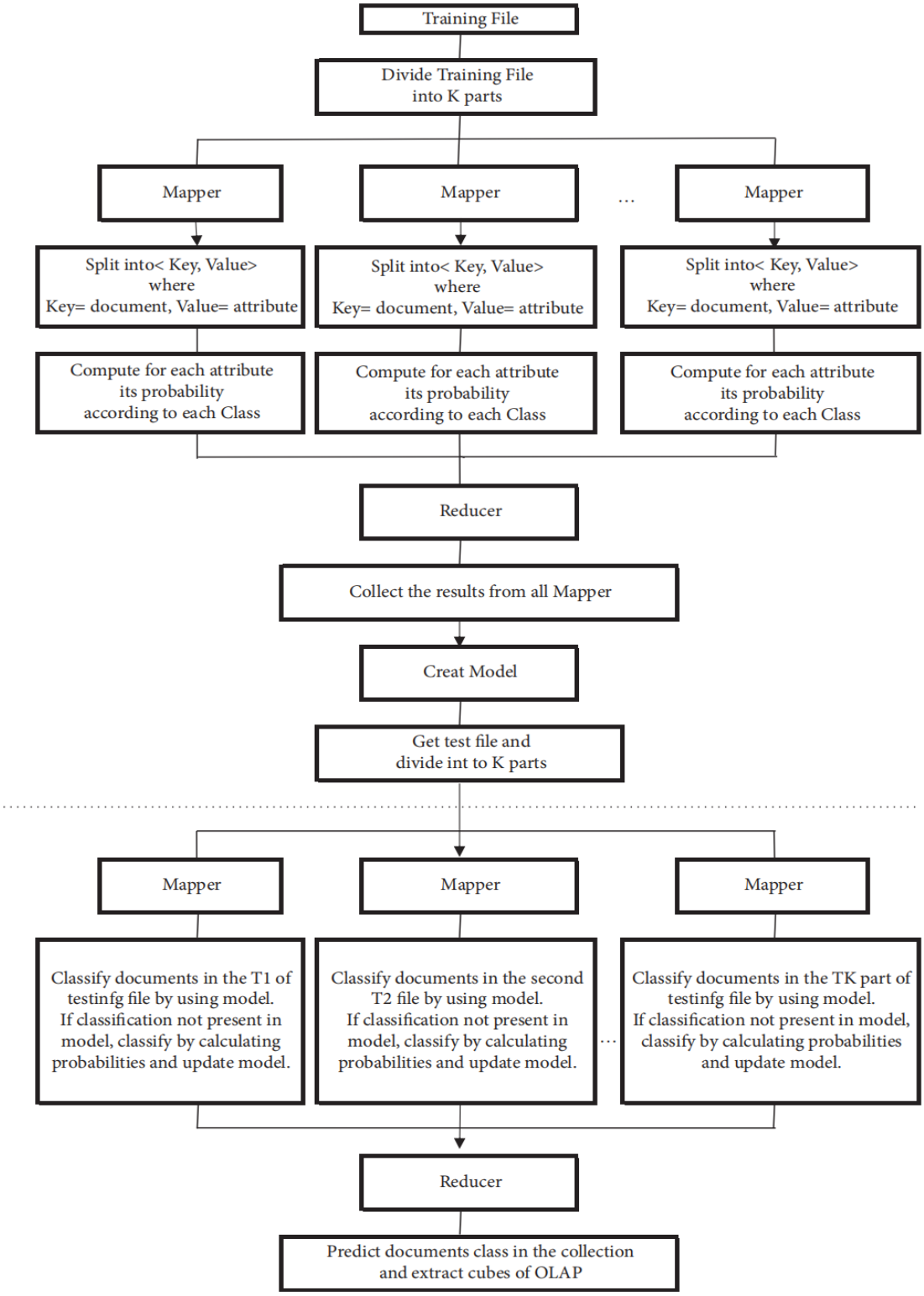


Figure V-6: La solution proposée (NBMR).

3.4. Synthèse

Une nouvelle méthode d'extraction de cubes OLAP à partir de bases de données NoSQL orientées documents est proposée où :

Nous voyons l'introduction de la méthode de classificateur Naïve Bayes évolutive.

L'utilisation de MapReduce renforce la performance de traitement.

Cette étude s'est concentrée sur les bases de données NoSQL orientées documents, cependant, elle peut être appliquer les mêmes techniques à d'autres types de bases de données NoSQL, comme celles orientées graphes, colonnes, etc.

4. Approche de Chevalier, M., & al. [35]

Les auteurs dans cette approche ont proposé une approche des Entrepôts de données multidimensionnelles NoSQL. Ils définissent une règle qui transforme le schéma en étoile en deux modèles logiques NoSQL avec son optimisation (treillis agrégé pré-calculé). Orienté colonne ou orienté document. Utilisez ces règles pour implémenter et analyser deux systèmes de décision, un pour chaque modèle, avec MongoDB. HBase. Comparez-les dans la phase de chargement des données (Généré par le benchmark TPC-DS), calculez et interrogez le treillis.

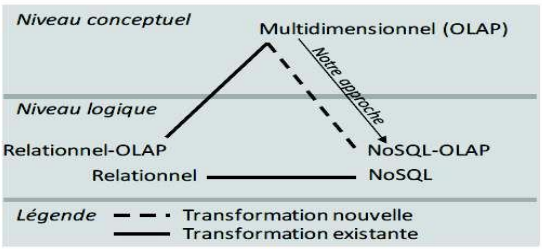


Figure V-7: Règles de transformations d'un modèle conceptuel en un modèle logique NoSQL.

Comme le montre la figure V-7, cette approche propose un niveau d'abstraction du système d'information qui remplace le modèle logique R-OLAP traditionnel par deux modèles NoSQL.Cette contribution est multiple :

– Définit des règles qui peuvent transformer un modèle conceptuel multidimensionnel en un modèle logique. Orienté colonne et orienté document.

– Définir des règles incluant une grille pré-agrégée dans ces modèles.

– Implémenter une base de données multidimensionnelle en MongoDB et HBase

Appliquer les règles pour étudier la phase de chargement des données et exécution de requêtes treillis et OLAP en NoSQL.

4.1. Les règles de passage

4.1.1. Orienté colonnes

Les éléments (faits, dimensions) du modèle conceptuel multidimensionnel doivent être transformés en éléments du modèle NoSQL orienté colonnes (Figure V-8).

– Chaque schéma en étoile conceptuel (chaque fait Fi de F E, et ses dimensions Star(Fi)) est transformé en une table T E.

– Le fait Fi est transformé en une famille de colonnes CF M de T E dans laquelle chaque mesure mi est une colonne Ci ∈ CF M.

– Chaque dimension Di ∈ StarE(Fi) est transformée en une famille de colonnes CF Di où chaque attribut de dimension Ai ∈ ADi (paramètres et attributs faibles) de la dimension Di est transformé en une colonne Ci de la famille de colonnes CF Di (Ci ∈ CF Di), à l'exception du paramètre AllD. (Figure V-8)

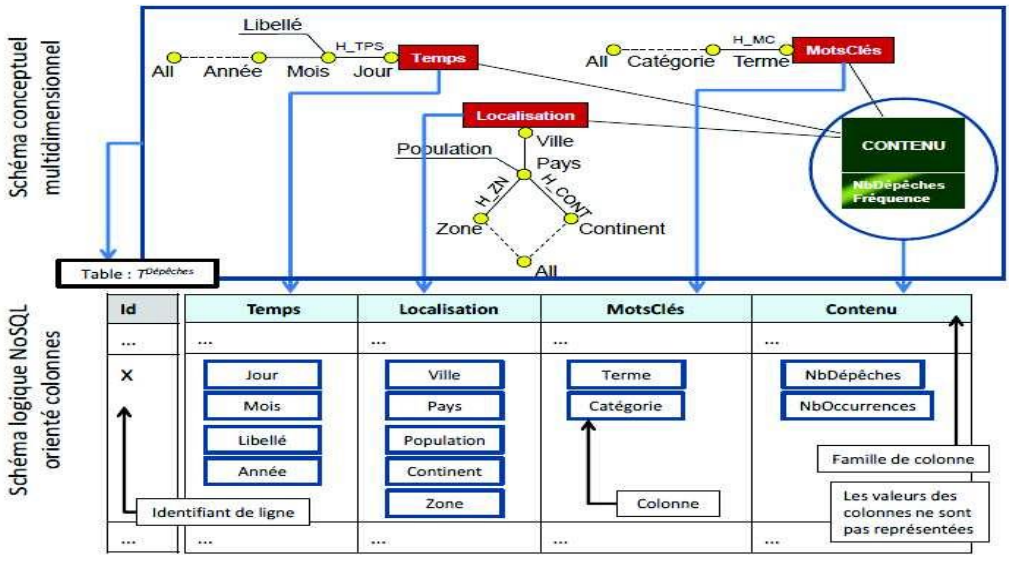


Figure V-8: Transformation du MCM au modèle logique NoSQL orienté colonnes.

4.1.2. Orienté Documents

Formellement, une base de données orientée documents peut être définie comme une collection CD = {D1, . . . , Dn} constituée d'ensemble de documents Di . Chaque document Di est défini par un ensemble de paires {(Att1i , Vi1),. . ., (Attmi i , Vi mi)} où Attji est un attribut avec j ∈ [1, mi] (similaire à une clef) et Vij est une valeur qui peut être de deux formes :

- Soit la valeur est atomique.
- Soit la valeur est elle-même composée d'un document imbriqué défini comme un nouvel ensemble de paires (attribut, valeur).

Nous distinguons les attributs simples dont les valeurs sont atomiques des attributs composés dont les valeurs sont des documents imbriqués (Figure V-9).

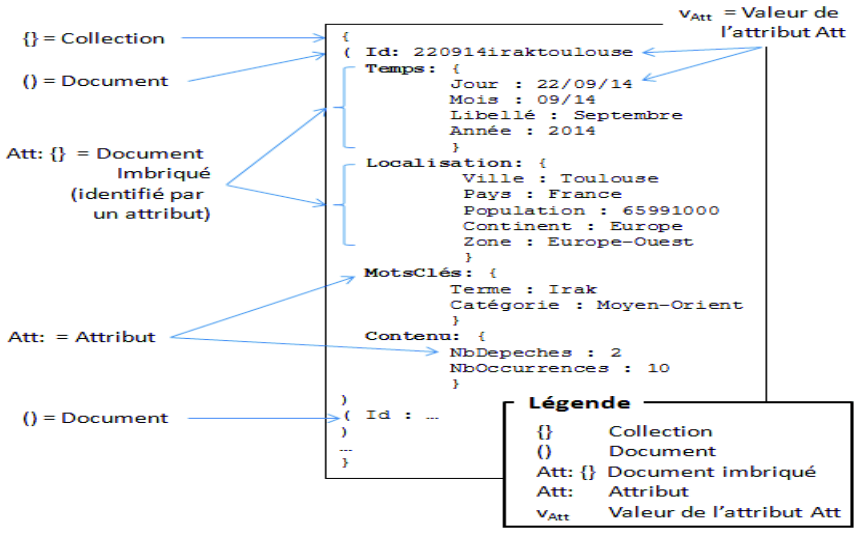


Figure V-9: Représentation graphique de la collection C Dépêches.

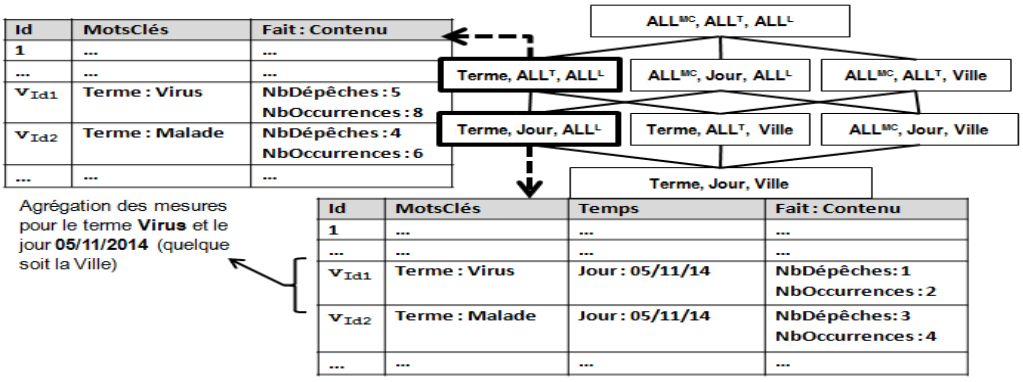
4.2. Treillis d'agrégats dans le NoSQL

Nous utiliserons les notations suivantes pour définir les règles de correspondance du treillis.

Un treillis d'agrégats pré-calculés, noté L, est un ensemble de nœuds AL reliés par des arcs EL (chemins envisageables pour calculer les agrégats les uns à partir des autres). Un nœud (dit d'agrégat) A ∈ AL est composé d'un ensemble de paramètres pi (un par

dimension) et d'un ensemble de mesures mi agrégées fi(mi). A =< p1, ..., pk, f1(m1), ..., fv(mv) >, k ≤ m (m étant le nombre de dimensions, v étant le nombre de mesures du fait). Pour implanter le treillis dans une base de données NoSQL orientée colonnes, les règles suivantes sont appliquées :

- Chaque nœud d'agrégat A ∈ AL est stocké dans une table T A, T A 6 = T E.
- Pour chaque dimension Di correspondant au nœud A, une famille de colonne CFDi est créée. Chaque attribut ai de la dimension est stocké dans une colonne Cai de CFDi,
- L'ensemble des mesures agrégées est aussi stocké dans une famille de colonnes CFF où chaque mesure agrégée est stockée dans une colonne C. (Figure V-10 et figure V-11)



Id	MotsClés	Fait : Contenu
1	...	...
...	...	...
v_{Id1}	Terme : Virus	NbDépêches : 5 NbOccurrences : 8
v_{Id2}	Terme : Malade	NbDépêches : 4 NbOccurrences : 6
...	...	...

Id	MotsClés	Temps	Fait : Contenu
1	...	...	...
...	...	...	...
v_{Id1}	Terme : Virus	Jour : 05/11/14	NbDépêches: 1 NbOccurrences : 2
v_{Id2}	Terme : Malade	Jour : 05/11/14	NbDépêches: 3 NbOccurrences : 4
...	...	...	...

Figure V-10:Treillis simplifié en modèle orienté colonnes.

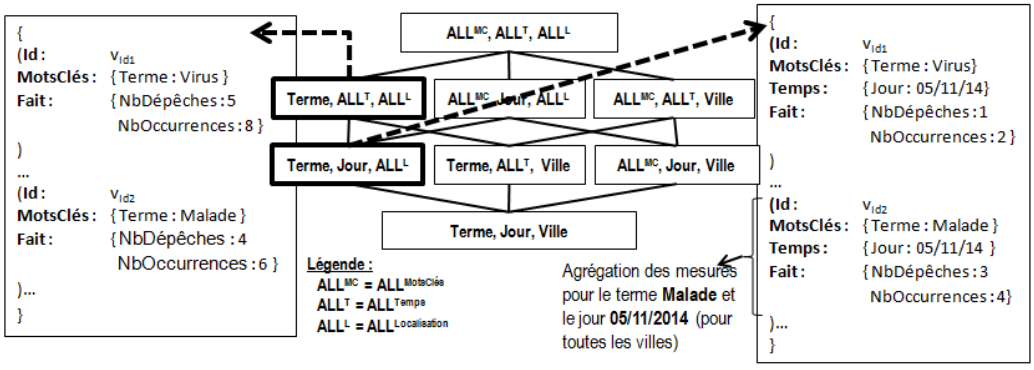


Figure V-11: Treillis simplifié en modèle orienté documents.

4.3. Synthèse

Les auteurs de cette approche ont réussi à faire un passage du modèle conceptuel au model logique NoSQL.

Ils ont proposé un processus de transformations pour deux modèles de NoSQL : Orienté Colonnes et Documents. Agrégation dans les transformations ont été faite pour les deux modèle.

Cette approche base sur les schémas en étoiles.

5. Discussion

Les trois approches présentées ci-dessus sont considérées parmi les approches les plus récentes ayant traité sur l'OLAP dans un modèle NoSQL. Le tableau V-1 récapitule et compare les trois approches étudiées.En terme de performances du produit de base de données utilisé, l'approche de Khalil, A., & Belaissaoui, M. n'est pas encore mature, l'objectif était d'évaluer les performances de l'approche de modélisation sous le stockage clé-valeur.Dans l'approche de Davardoost, F., Babazadeh Sangar, A., & Majidzadeh, K., les auteurs présentent une approche cube OLAP reposant sur la méthode de classificateur Naïve Bayes évolutive avec L'utilisation de MapReduce pour renforcer la performance de traitement. L'approche de Chevalier, M., El Malki, M., Kopliku, A., Teste, O., & Tournier, R. ont converti le schéma à la fois en mode orienté colonnes (HBase) et orienté documents (MongoDB). Ils ont utilisé un schéma en étoile pour les règles de mappage qui se composent uniquement de faits et de dimensions. Chevalier et les autres auteurs ont transposé les règles directement du modèle de données multidimensionnel au modèle NoSQL orienté document (au lieu du modèle de données multidimensionnelles vers le modèle relationnel NoSQL orienté document).

Ces contributions proposent d'utiliser un schéma en étoile et de créer ensuite les cubes OLAP NoSQL où les données sont déjà nettoyées et transformées. Par ailleurs, l'écosystème des lacs des données n'est pas pris en charge lors de la construction du cube OLAP. C'est là que notre contribution intervient, pour déplacer l'image de l'OLAP vers les technologies tendances, où le Data Lake commence à trouver sa popularité en raison de ses divers avantages, y compris le déplacement de la phase de transformation, où celle-

ci est effectuée pour chaque application séparément, de sorte que les données sont enregistrées dans leurs formats natifs garce au processus ELT du Data Lake.

Critères \ Approches		Davardoost, F., Babazadeh Sangar, A., & Majidzadeh, K.,	Khalil, A., & Belaissaoui, M.	Chevalier, M., El Malki, M., Kopliku, A., Teste, O., & Tournier, R.
Modèle logique (NoSQL)		Orienté Document	Clé-valeur	Orienté Colonne - Orienté Document
Schéma conceptuel		En étoile	En étoile	En étoile
Système de stockage ou SGBD		HDFS (Hadoop 2.7.3) et MongoDB	Oracle NoSQL	HBase , MongoDB
Mapping (passage)		Naive Bays	KV Operator	Règle de passage vers NoSQL orienté colonnes/ Documents
Techniques agrégation		Algorithme NMBR	Technique de Hashage	Treillis d'agrégats OCL/OD
DataSet	Source du dataset	DPLB, "DBLP XML records,"2022,http://dblp.uni-trier.de/xml/.	TCP benchmark	benchmark TPC-DS
	Formats de données	Documents XML	Données relationnelles	Données relationnelles
	Taille du dataset	5000 Documents	15 GB	100 GB

Tableau V-1: Comparaison des travaux connexes.

6. Conclusion

La recherche dans les travaux connexes dans ce domaine fournit un ensemble d'information et d'idées qui facilité le travail que nous sommes en train d'accomplir. Dans le cadre de ce mémoire, les règle de passages suggérés dans l'approche de Chevalier, M., El Malki, M., Kopliku, A., Teste, O., & Tournier, R. Sont plus intéressantes car elle offre une solution à une phase de notre plate-forme, à savoir, stockage des cubes sous le modèle NoSQL. Nous avons défini, dans le présent chapitre, quelques contributions récentes sur le cube OLAP adaptée aux bases de données NoSQL. Pour conclure, une discussion sur les approches vienne par la fin où on a introduit un tableau récapitulatif.

VI. Proposition d'une nouvelle approche d'analyse OLAP dans un environnement Data Lake

Ce chapitre présente notre contribution dans le cadre de ce travail à savoir OLAP en environnement Data Lake. La présentation de ce chapitre est comme suit : introduction, présentation de l'approche proposé et ses différents composants et la conclusion par la fin.

---

1. Introduction

Dans les précédents chapitres nous avons exposé les concepts fondamentaux le baie de mieux comprendre notre plate-forme. Dans ce présent chapitre nous proposons une approche de traitement des cubes OLAP dans un environnement Data Lake. Étant donné un Data Lake préalablement alimenté, notre système permettre la transformation des données issues du Data Lake en vue de les transformer en cubes de données exploitables pour des fins d'analyse. En ce sens, notre travail ne couvre pas les phases E et L de 'ELT' puisque nous ne considérons pas d'autres sources de données à part le Data Lake.Lors de la construction des cubes OLAP plusieurs aspects seront pris en charge par notre plate-forme tels que la structure des données et leur transformation. Les cubes générés doivent être stockées dans un SGBD NoSQL orienté documents, à savoir, MongoDB. À cette fin, nous avons utilisé des règles de transformations permettant de passer depuis un modèle conceptuel multidimensionnel (modèle en étoile) vers un modèle logique NoSQL (orienté documents). La plate-forme englobe trois composants importants, à savoir, la transformation, stockage des cubes et traitement de requêtes.

2. Description de la plate-forme

La plate-forme proposée est une solution OLAP et ses opérations appliquées dans un environnement Data Lake. En effet, notre plate-forme permet de transmettre un objectif d'analyse à un cube OLAP NoSQL à partir des lacs de données. La transformation, le stockage des cubes OLAP NoSQL et le traitement des requêtes sont les principaux modules qui composent cette plate-forme où l'aspect de la diversité des formats de données est supporté.

La figure VI-1 représente l'architecture de notre système.

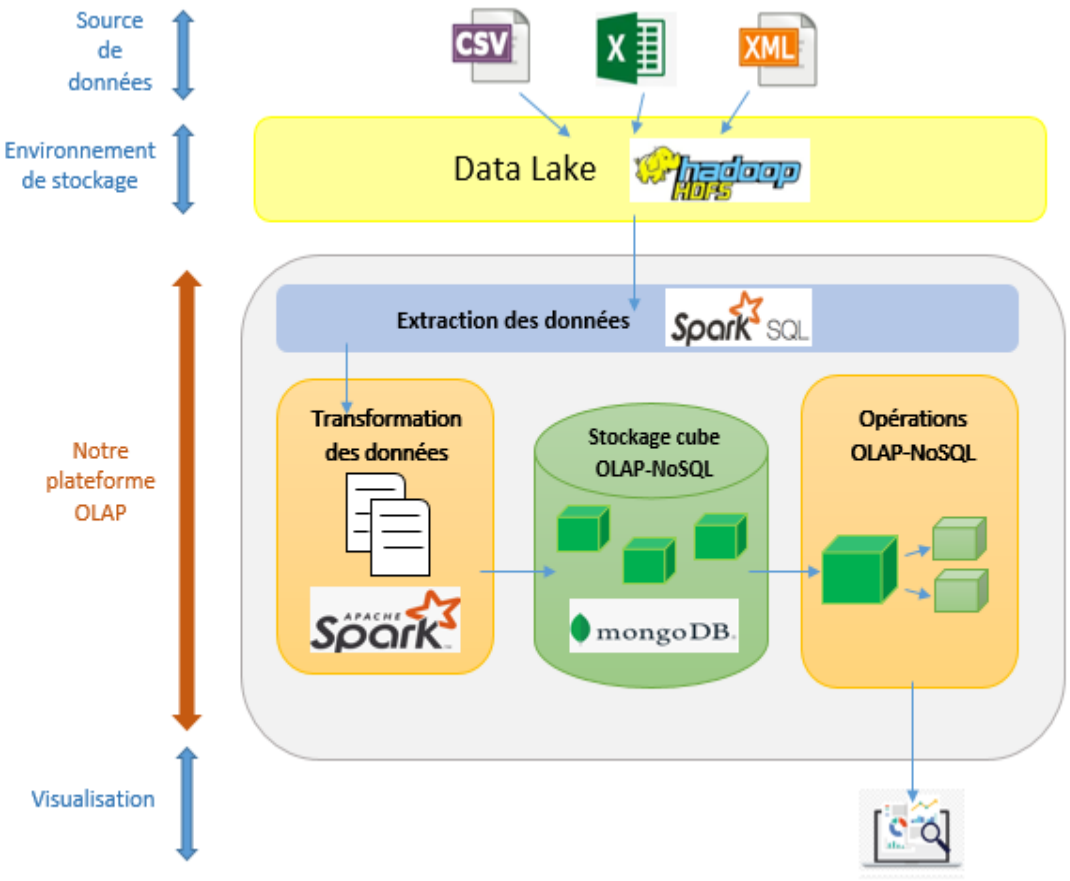


Figure VI-1: Architecture de plateforme OLAP en DataLake.

Nous décrivons l'architecture du système de la manière suivante :

2.1. Extraction à partir de Data Lake

L'extraction des données est le processus d'obtention des données à partir du Data Lake de données qu'elles puissent être répliquées vers une destination conçue pour prendre en charge le traitement analyse en ligne (OLAP). Nos travaux se déroulent dans un environnement caractérisé par les lacs de données (Data Lake) où l'on trouve une diversité des structures de données, à savoir CSV, XML, XLSX, TXT. etc. Ces données se trouve dans leur format brut.

Un environnement Hadoop est choisi pour la représentation du Data Lake dû à ses avantages tel que la gestion automatique des pannes matérielles, le bon fonctionnement avec les autres logiciels ce qu'il nous aide à le relier avec les autres composants de la plate-forme, ainsi que Hadoop traite les gros volumes de données et porte les applications Big Data sans problème.

Dans notre plate-forme proposée le Data Lake est préalablement alimenté. Le composant suivant, à savoir transformation établis une connexion Hadoop HDFS afin de récupérer les fichiers cibles qui répondent au besoin d'analyse.

.

2.2. Transformation

Les données extraites du Data Lake sont brutes et non utilisables dans leur forme originale. Elles doivent donc être nettoyées, cartographiées et transformées. En fait, il s'agit de l'étape clé où nous ajoutons de la valeur et modifions les données de sorte qu'elle répond au besoin d'analyse.

La phase de transformation est un des concepts importants où nous appliquons un ensemble de fonctions sur les données extraites. Les données qui n'ont pas besoin d'être transformées sont appelées "données de passage" ou "données directes". Dans l'étape de transformation, nous pouvons effectuer des opérations personnalisées sur les données. Par exemple, si l'utilisateur veut la date sous forme 'Mois/Année' qui n'est pas dans les fichiers extraits. Il est possible de les concaténer les deux.

En effet, les traitements appliqués sur les fichiers extraits dans cette phase se résume en :

- Unifier le format des fichiers.
- Reformater les données converties dans un format standard pour la comptabilisation.
- Nettoyer les données non pertinentes des ensembles de données.
- Trier et filtrer les données.

- Suppression des informations en double.

Dans notre plate-forme proposée nous utilisons le Framework Apache Spark pour la transformation des fichiers extraits et les unifiés. Les transformations préparent les données pour répondre aux objectifs d'analyse. Ce dernier est traduit en requête SQL avec des jointures entre plusieurs fichiers extraits ainsi que l'agrégation des mesures. L'exécution des requêtes se fait dans Spark.

En résultat, nous obtenons des cubes OLAP bien structurés et prêts pour le stockage.

2.3. Stockage des cubes OLAP-NoSQL

Le résultat de la phase précédente est un cube de données OLAP classique. Pour notre solution, nous allons stocker ce cube dans une base de données NoSQL. Les bases de données NoSQL sont conçues pour minimiser les problèmes des SGBDR, et la compatibilité avec le big data a été la principale raison de ce choix. MongoDB est une base de données NoSQL orienté documents dont voici quelques avantages :

- Le support proposé par MongoDB est de très bonne qualité. Ils sont toujours pertinents et répondent de manière très rapide.
- La fonctionnalité la plus appréciable est que pratiquement toutes les langues sont prises en charge par MongoDB pour le développement d'applications.
- Avec l'intégration de Spark, MongoDB propose de nouveaux horizons pour des analyses avancées

Pour cela, MongoDB était un idéal choix afin de stocker les cubes NoSQL obtenus après l'application du processus de traduction basé sur les règles de passage proposé par [39] sur le cube obtenu par la requête après la connexion avec MongoDB.

2.3.1. Les règles de passage d'un modèle conceptuel vers un modèle logique NoSQL

Comme l'illustre la figure V-7, nous avons utilisé des règles de transformation permettant de passer directement depuis un modèle conceptuel multidimensionnel (modèle en étoile) vers un modèle logique NoSQL orienté documents.

Dans le modèle orienté documents, les données sont stockées en collections de documents. La structure des documents est définie par l'intermédiaire d'attributs (pouvant s'apparenter à des couples clé / valeur où l'attribut est la clé). Nous distinguons les attributs simples dont les valeurs sont atomiques et des attributs composés dont les valeurs sont-elles - mêmes des documents, appelées documents imbriqués.

Les règles de transformation que nous avons proposées sont :

- Règle 1 : Une table de faits TF du modèle en étoile, devient une collection C, dans le modèle logique NoSQL orienté documents, dont les documents prennent leurs attributs à partir du schéma de la TF.
- Règle 2 : Une ligne de la TF du modèle en étoile, devient un document D inclus dans la collection dans le modèle logique orienté documents.
- Règle 3 : La clé primaire de la TF du modèle en étoile, devient un attribut simple dans les documents D de la collection C dans le modèle logique orienté documents.
- Règle 4 : Une mesure dans une TF du modèle en étoile, devient un attribut simple dans les documents D de la collection C dans le modèle logique orienté documents.
- Règle 5 : La clé étrangère de la TF du modèle en étoile, devient un attribut simple dans les documents D de la collection C dans le modèle logique orienté documents. Cet attribut permettra de relier un document de type 'Fait' avec un document de type 'Dimension' correspondant dans la même collection, afin d'assurer le lien implicite entre ces documents.
- Règle 6 : Une table de dimension TD du modèle en étoile, devient une collection C '', dans le modèle logique orienté documents, dont les documents prennent leurs attributs à partir du schéma de la TD.

Le tableau VI-1 synthétise les règles énoncées ci-dessus.

Modèle OLAP / Modèle NoSQL	Fait	Mesure	Dimension	Paramètre	Attribut faible
Orienté Document	Document imbriqué (Attribut composé)	Attribut simple	Document imbriqué (Attribut composé)	Attribut simple	Attribut simple

Tableau VI-1: Règles de transformations d'un modèle en étoile (conceptuel) vers un modèle logique NoSQL orienté document.

2.4. Proposition d'une algèbre OLAP adaptée au modèle NoSQL

L'algèbre OLAP dépend étroitement de la structure et du modèle dans lequel sont stockées les données du cube. L'algèbre usuelle utilisée largement dans les systèmes OLAP est basée sur le modèle relationnel. Cette algèbre ne peut continuer à être utilisée lorsque les données changent de modèle et de structure. C'est pour cela que l'algèbre que nous proposons dans le cadre de ce travail dépend des notions NoSQL orienté documents.

- **Rappel sur le schéma logique d'un cube de données NoSQL orienté document**

Une base de données orientée documents contient des documents représentant des enregistrements dont chacun décrit les données d'un document. Formellement, une base de données orientée documents, est définie comme une collection CD où :

CD = {D1, D2, …, Dn} et Di (i=1, n) est un document .

Dans le cube des données, chaque document de la collection CD est un document imbriqué qui représente un fait et dans chaque fait il existe une collection des documents

qui représente les dimensions. Les attributs de chaque dimension ainsi les mesures sont des attributs simples.

L'algèbre OLAP proposée est classée en deux catégories à savoir (1) les opérateurs OLAP et (2) les opérations OLAP. Deux opérateurs sont inclus dans la première catégorie, il s'agit des opérateurs de selection et d'agrégation. D'autre part, cinq types d'opérations sont inclus dans la deuxième catégorie et sont définies grâce aux deux opérateurs de sélection et d'agrégation.

- **Opérateurs OLAP proposés**

Les représentations formelles de deux opérateurs OLAP sont proposées ci-après.

(a) Opérateur de sélection (π) : Cet opérateur extrait la dimension D et sa hiérarchie HFD de la famille de dimensions FD en fonction d'un certain prédicat *p*. Il peut s'agir d'un prédicat atomique noté p ou d'un prédicat composite noté $p1 < op > p2 < op >.... . . < op > pn$ où le prédicat composite est un ensemble de propositions reliées par un connecteur logique tel que AND, OR, etc. La notation algébrique de l'opérateur est la suivante :

$$\pi_p (\text{FD}) = \text{FD}_o$$

Ici FD est la famille de dimensions originale sur mesure et FDo est la famille de dimensions de sortie sur mesure après la restriction. L'opérateur prédicat nul renvoie la FD originale. Par conséquent,

$$\pi_\emptyset (\text{FD}) = \text{FD}$$

(b) Opérateur d'agrégation (α) : L'opérateur d'agrégation effectue la fonction de regroupement GF sur l'attribut de mesure MAT de l'ensemble spécifié de FD dans un cube C. La GF est une fonction d'agrégation , qui va opérer sur les MAT uniquement. La notation algébrique de l'opérateur d'agrégation est la suivante :

$$\alpha_{GF(MAT)\ \{\text{FD1}\ V\ \text{FD2}\ V\ \text{FD3}\ \ldots\ V\ \text{FD}n\}}\ (C)$$

- **Opérations OLAP proposées**

Dans cette section, cinq opérations OLAP sont formellement spécifiées.

(a) Opération de SLICE : L'opération SLICE sélectionne une valeur spécifique d'une dimension d'un cube d'entrée pour extraire une tranche de ce cube. La notation algébrique de l'opération SLICE est la suivante

$$\text{SLICE}(C) = \alpha_{GF(MAT)\{\text{FD}\},\, CON}(C)$$

Ici, CON est la condition définie comme suit,

$$CON = \pi_p\,(\text{FD})$$

(b) Opération de DICE : L'opération DICE sélectionne deux ou plusieurs valeurs de certaines dimensions d'un cube d'entrée pour extraire un sous-cube. La notation algébrique de l'opération DICE est la suivante :

$$\text{DICE}(C) = \alpha_{GF(MAT)\{\text{FD}\},\, CON}(C)$$

Ici, CON est la condition définie comme,

$$CON = \pi_{p1}\,(\text{FD1}) < op > \pi_{p2}\,(\text{FD2}) \ldots < op > \pi_{pn}(\text{FD}n)$$

(c) Opération ROLL-UP : L'opération Roll-up effectue une agrégation sur un cube de données en remontant d'une dimension dans la hiérarchie dimensionnelle ou en retirant une ou plusieurs dimensions. La notation algébrique de l'opération ROLL-UP est la suivante :

$$\text{ROLL-UP}\,(\text{FD}_{ij})\,(C) = \alpha_{GF(MAT)\{\text{FD}i(j+1)\}}\,(C)$$

Ici, l'opération de ROLL-UP va de la granularité supérieure à la granularité inférieure en augmentant la valeur de j de 1 pour un niveau de ROLL-UP. Si l'opération de ROLL-UP est de 2 ou plus de 2 niveaux, alors l'opération ROLL-UP est calculée à chaque niveau. Ici, j et i sont utilisés uniquement à des fins d'indexation et sont des entiers numériques positifs. i est défini pour FD et j est défini pour la hiérarchie dimensionnelle.

(d) Opération DRILL-DOWN : L'opération de DRILL-DOWN est l'opération inverse du roll-up. L'opération de forage vers le bas est effectuée en descendant dans la hiérarchie des dimensions ou en ajoutant une dimension. Ainsi, elle passe d'une granularité inférieure à une granularité supérieure. La notation algébrique de l'opération de DRILL-DOWN est la suivante :

$$\text{DRILL-Down } (FD_{ij})\ (C) = \alpha_{GF(MAT)\{FDi(j\text{-}1)\}}\ (C)$$

(e) Opération PIVOTE : L'opération pivot fournit une présentation alternative d'une donnée en faisant pivoter les axes de données dans une vue. Ainsi, cette opération est également appelée rotation. Il s'agit d'analyser la combinaison de paires de dimensions sélectionnées. La notation algébrique de l'opération PIVOTE est la suivante :

$$\text{PIVOTE}(C) = \alpha_{GF(MAT)\ \{FD1,\ FD2\}^{\ T}}\ (C)$$

$$= \alpha_{GF(MAT)\ \{FD2,\ FD1\}}\ (C)$$

3. Conclusion

Tout au long de ce chapitre, notre objectif est d'exposer notre approche de conception du nouveau système et les bases théoriques de celle-ci à savoir la modélisation d'un cube NoSQL et la manipulation de celui-ci avec des opérateurs OLAP qui lui sont adaptés. Nous avons exposé notre contribution, à savoir l'analyse des données en ligne OLAP dans un environnement Data Lake. Nous avons détaillé notre architecture en soulignant ses caractéristiques et les principes importants de notre plateforme, commençant par l'extraction des données, passant par la transformation, ainsi le stockage des cubes OLAP-NoSQL et le traitement de requêtes par la fin.

Notre travail traite sur un écosystème en tendance, en l'occurrence les Data Lakes, et la sensibilité de la tâche, à savoir l'aide à la. L'utilisation des outils évolutifs qui traitent sur les données massives donne une flexibilité à notre plateforme.

Troisième partie

Implémentation

VII. Environnement de développement

Dans ce chapitre, nous présentons tous les outils logiciels et matériels ainsi que l'environnement de développent utilisés pour mettre en œuvre notre plateforme d'analyse OLAP dans un environnement Data Lake.

---

1. Introduction

Il existe une variété d'outils développés et intégrés dans de nombreux langages de programmation, ce qui facilite la tâche aux programmeurs dans le développement des projets. Le choix des outils de développement impacte énormément le coût de développement, ainsi que sur la flexibilité du produit à réaliser.

2. Outils de développement

Dans le cadre de notre projet, nous avons adopté un environnement de développement intégrant plateformes, langages, librairies et les outils suivants :

2.1. Python

Notre application a été développée avec le langage de programmation Python en utilisant l'environnement de développement Pycharm IDE. Python est un langage de programmation facile à apprendre car il possède une syntaxe très simple et une documentation très fournie [36].

La figure VII-1 représente la version python utilisée.

```
C:\Users\Mokhtari>python --version
Python 3.10.4
```

Figure VII-1:la version de Python.

2.1. Apache Hadoop

Apache Hadoop est un cadre basé sur Java, destiné au traitement distribué par lots et au stockage distribué de très grandes quantités de données [37]. Hadoop a pour noyau le système de fichiers distribués Hadoop ou HDFS comme système de stockage de données

distribuées, et les API Map Reduce qui rendent possible le traitement parallèle de données distribuées sur HDFS.

HDFS est essentiellement un système de fichiers distribué conçu pour contenir de très grandes quantités de données (téraoctets ou pétaoctets) et fournir un accès à haut débit à ces informations. Les fichiers sont divisés en blocs et stockés de manière redondante sur plusieurs ordinateurs. Cela garantit leur durabilité en cas de défaillance et leur haute disponibilité pour les applications parallèles. La taille de bloc par défaut pour HDFS est de 128 Mo mais elle est également configurable. HDFS stocke un bloc entier sur une seule machine. HDFS se trouve au sommet du système de fichiers d'un système d'exploitation, donc le système de fichiers du système d'exploitation stocke les fichiers HDFS en plus petits morceaux qui correspondent à la taille de bloc dans le système de fichiers natif [38]. La figure VII-2 présente la taille de bloc par défaut pour un fichier de 300MB.

Figure VII-2: la taille de bloc par défaut pour HDFS [38].

Hadoop est basé sur un ensemble de machines formant un cluster Hadoop. Chaque machine est appelée nœud, le système HDFS repose dans son fonctionnement sur deux types de nœuds :

- Le NameNode : le nœud maître est l'orchestrateur du système HDFS, au moyen de métadonnées. Il maintient l'arborescence de tous les fichiers du système et gère l'espace de nommage, autrement dit, il gère la correspondance entre un fichier et les blocs qui le constitue. Il prend en charge également la localisation des blocs des données dans le cluster. Il n'y a qu'un NameNode par cluster HDFS. Le secondary NameNode : il sert à effectuer à intervalle réguliers des sauvegardes du NameNode. Il est utilisé pour prendre la relève en cas de panne du NameNode.
- Les DataNode : ils servent d'espace de stockage et de calcul des blocs de données [39]. (Figure VII-3)

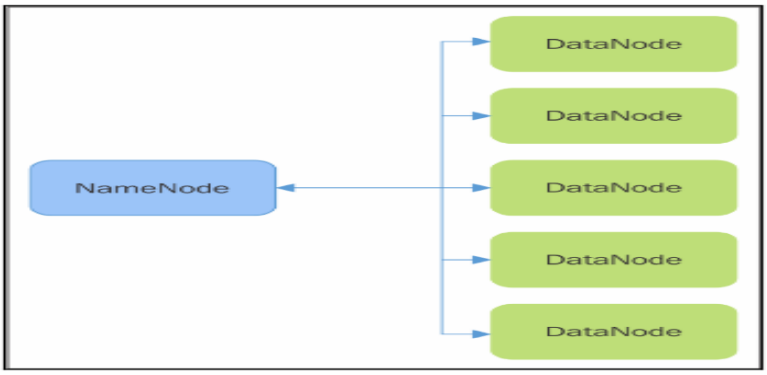


Figure VII-3: NameNode et DataNode [38].

La plateforme Hadoop représente, dans notre architecture, l'environnement Data Lake où les données initiales sont stockées. Les étapes d'installation de Hadoop sont présentés ci-dessous.

Étape 1 : Télécharger le paquet Java 8. Java est le principal logiciel prérequis pour exécuter Hadoop.

Étape 2 : Extraire le fichier Tar de Java.

Étape 3 : Téléchargez le paquetage Hadoop 3.2.2.

Étape 4 : Extraire le fichier tar Hadoop.

Étape 5 : Ajouter les chemins accès à Hadoop et Java dans les variables d'environnement.

Étape 6 : Éditer les fichiers de configuration Hadoop.

Étape 7 : Ouvrir le site core.

Par la suite ouvrir l'invite de commande en mode administrateur et lancer la commande - start-all. la figure (VII-4) illustre l'étape.

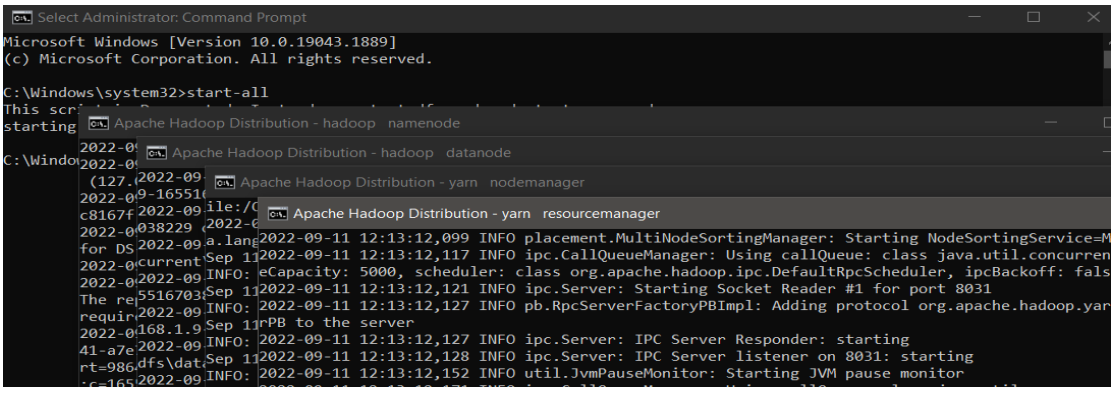


Figure VII-4 :Start-all command - Apache Hadoop-.

l'interface web du Resource Manager ; par défaut, elle est disponible à l'adresse suivante : http://localhost:8088/ . (Figure VII-5)

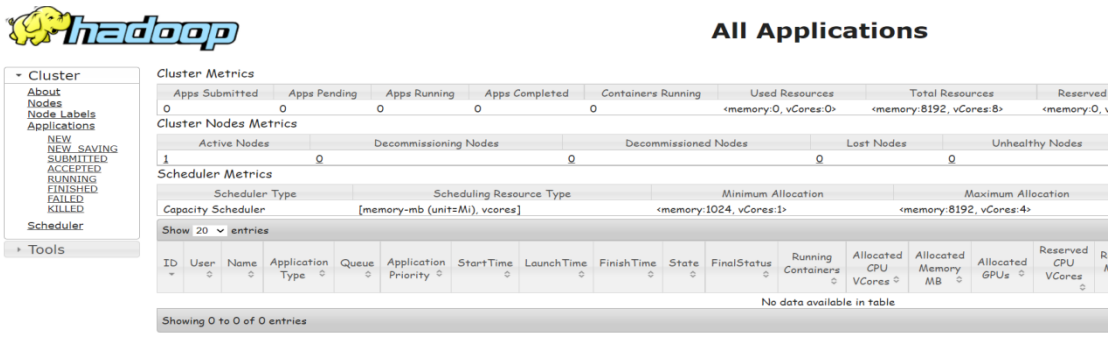


Figure VII-5:Interface web du Resource Manage -Apache Hadoop-.

2.2. Apache Spark

Apache Spark est un moteur de traitement distribué capable de traiter des pétaoctets (PB) de données. Comme il s'agit d'un moteur à usage général, il convient à une grande variété de cas d'utilisation à l'échelle, notamment la conception et l'exécution de pipelines d'extraction, de transformation et de chargement (ETL) à l'aide de la bibliothèque Spark SQL, l'analyse interactive, le traitement en continu à l'aide de la bibliothèque Spark Streaming [40].

Spark n’est pas une base de données et ne stocke rien, il se base sur des systèmes de stockage comme Hadoop. Et nous utilisons la version 3.1.2 comme montre la figure VII-6.

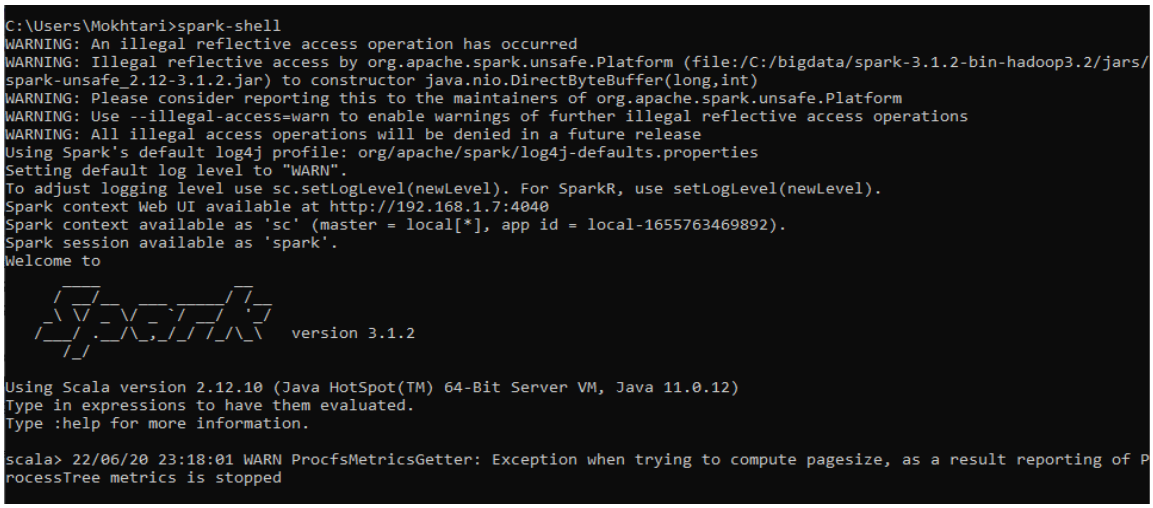


Figure VII-6: la version d'Apache Spark.

Le rôle de Spark se concentre dans la phase de transformation ou il va extraire les données à partir HDFS et les transformer. Les étapes d'installation de Spark sont presque similaires à Hadoop. Java est prérequis pour exécuter Spark.

Pour lancer Spark il suffit de lancer la commande : - spark-shell

l'interface web du spark-shell ; par défaut, elle est disponible dans l'adresse suivante : http://localhost:4040/. (Figure VII-7)

Figure VII-7: Interface web du spark-shell -Apache Spark-.

2.3. MongoDB

MongoDB est une base de données documentaire. Au niveau le plus élevé de l'organisation. Dans son modèle de stockage, les données sont stockées dans des documents. Les documents sont regroupés dans une collection où chaque document est constitué d'un ensemble de paires clé-valeur pouvant être organisées de manière

hiérarchique. Un document est identifié par une clé unique (de taille maximale de 96 octets) gérée automatiquement par le système (comme elle peut être gérée par le client). Chaque document est stocké dans une représentation comprise et optimisée par MongoDB, BJSON (Binay JSON) et peut posséder sa propre structure, on parle de schema-less. [39] Et on a utilisé la version 5.0.8. (Figure VII-8)

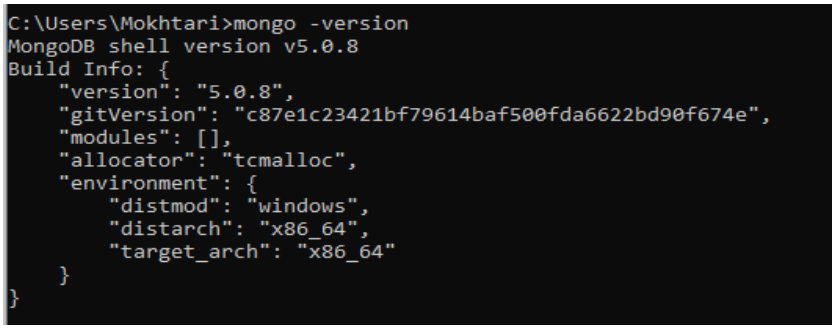


Figure VII-8: Version de MongoDB.

Les cubes OLAP -NoSQL construit seront stockés dans les MongoDB. L'installation de MongoDB est simple après le téléchargement le package à partir du site officiel il suffit de suivre les étapes dans l'interface.

2.4. Matériel utilisé

Pour la réalisation de notre système, nous avons utilisé un PC avec les capacités et les caractéristiques suivantes :

Unité	Caractéristiques
Processeur	Intel® Core™ i5-10210U CPU @1.60 GHz 2.11 GHz
Mémoire RAM	16.0 GB (15.8 GB usable)
Type de système d'exploitation	64-bit operating system, x64-based processor
Edition	Windows 10 Pro
Version	21H1

Tableau VII-1: Caractéristiques du matériel utilisé

3. Conclusion

Dans ce chapitre nous avons présenté l'environnement de développement logiciel et matériel de notre plateforme et aussi les outils utilisés.

VIII. Implémentation

Ce chapitre est dédié aux détails d'implémentation et de tests. Après une brève description du jeu de données, nous décrivons les détails de chaque module et nous terminons ce chapitre par une conclusion.

---

1. Introduction

Après avoir exposé l'architecture cible ainsi que l'environnement de développement, nous arrivons à ce niveau à la phase de concrétisation de notre projet à savoir la mise en œuvre du système d'analyse OLAP dans un environnement Data Lake.

2. Jeu de données

2.1. Description de la base de données

Pour nos besoins de tests et expérimentations, nous avons utilisé un programme permettant de générer des données synthétiques relatives à une chaîne hôtelière que nous résumons de la manière suivante :

Une chaîne hôtelière est présente sur plusieurs pays et continents. Chaque établissement (hôtel) du groupe dispose d'infrastructures de séjour (*chambres, suite.*), de restauration (*restaurants, cafétérias, pizzeria.*), de sport (*piscines, salles Fitness.*), de distraction (*salle de projection, salle de jeux.*), de détente (*Sauna, Jacuzzi, Soin.*) et de regroupements professionnels (*salle de conférence, salles de réunions.*). En plus des infrastructures d'accueil, un établissement organise aussi des sorties touristiques (*mer, montagne.*), culturelles (*visite de musée, théâtre.*), etc. Les clients sont de plusieurs catégories (*particuliers, entreprises, associations.*). Tous les établissements utilisent le même progiciel et le même schéma de la base de données (ci-dessous).

Etablissement (**IDEtab**, NomEtab, SuperficieEtab, NomResponsable, Ville, Pays, Continent)

Séjour (**IDsjr**, Nom, Superficie, TypeSjr, Etat)

TypeSjr = 'Chambre simple', 'Chambre double', 'Suite',

Restauration (**IDresto**, Nom, Superficie, TypeResto, Etat)

TypeResto = 'Resto', 'Pizzeria', 'Cafétéria',

Sport (**IDsport**, Nom, Superficie, TypeSport, Etat) *TypeSport ='Piscine', 'Salle Fitness'.*

Distract (**IDdistract**, Nom, Superficie, TypeDistract, Etat)

TypeDistract = 'Cinéma', 'Salle jeux', etc.

Detente (**IDdetente**, Nom, Superficie, TypeDetente, Etat)

TypeDetente = 'Sauna', 'Jacuzzi', 'Soins'.

Professionnel (**IDprof**, Nom, Superficie, TypeProf, Etat)

TypeProf ='Conférence', 'Réunion'.

Etat : décrit l'état de l'infrastructure ('Excellent', 'Très bon', 'Bon', 'En réparation' 'Fermé')

Sorties (**IDsortie**, Date, HDépart, HRetour, TypeSortie, Lieu)

TypeSortie = 'Touristique', 'Culturelle'.

Lieu dépend des valeurs de TypeSortie. Lieu = 'Mer', 'Montagne', 'Musée', 'Théâtre'.

Clients (**IDclient**, Nom, Adresse, tél, Catégorie, ModePaiement)

Catégorie = 'Particulier', 'Entreprise', Association

ModePayement='Especes', 'Virement', 'CIB', 'Chèque'

Facture (**IDclient, IDservice, DateDébut**, DateFin, TypeService, Montant)

La facture d'un client au sein de l'hôtel est détaillée par période (Date début –Date Fin) et par type de service.

Le paiement de la facture se fait globalement, au niveau de la réception de l'hôtel, selon le mode de paiement du client.

TypeService='Séjour', 'Restauration', 'Sport', 'Distraction', 'Detente', 'Professionnel', 'Sortie', etc.

IDservice : Selon les valeurs de TypeService, il correspond à l'un des ID des tables Séjour, Restauration, Sorties, etc.

Le schéma de la base de données montre des tables décrivant les différentes infrastructures d'accueil (*Séjour*, *Restauration*, *Sport*, *Distract*, *Detente* et *Professionnel)*. La table *Sorties* stocke les informations sur les sorties organisées par l'établissement. Par ailleurs, la table *Facture* contient les détails de consommation des clients des différents services offerts par l'établissement. Enfin, les tables *Etablissement* et *Clients* décrivent, respectivement, l'établissement où est installé le progiciel et les clients ayant accédé et consommé au moins une fois à l'établissement. Il faut noter que :

- Le groupe hôtelier peut posséder plusieurs hôtels dans un même pays.
- Le même client peut séjourner dans différents établissements du groupe hôtelier et peut consommer, au sein d'un même établissement, différents services.
- Tous les services offerts, quel que soit leur nature, sont identifiés de façon unique (PK en gras et soulignées) à l'échelle d'un établissement mais aussi à l'échelle du groupe : deux chambres ou deux salles de conférences, par exemple, appartenant à des établissements différents ne peuvent, en aucun cas, posséder les mêmes valeurs d'ID.

2.2. Génération de données

Pour la génération des données nous avons utilisé un générateur 'Mockaroo'. Mockaroo aide à créer des données aléatoires pour les tests. En utilisant Mockaroo, on peut télécharger des données en déférents formats.

Caractéristiques :

- Il prend en charge plus de 100 types de données.
- Il permet de générer des données aux formats CSV, JSON, SQL, XML et Excel.
- Il fournit une API fantaisie.
- Crée des données réalistes.

Le tableau VIII-1 montre le format et la taille de chaque fichier généré.

Fichier	Format	Taille (par lignes)
Etablissement	CSV	8
Client	CSV	10,000
Facture	XML	10,000
Séjour	CSV	1000
Sport	EXCEL	1000
Restauration	CSV	1000
Professionnel	CSV	1000
Sortie	CSV	1000
Distract	EXCEL	1000
Detente	CSV	1000

Tableau VIII-1: Détail de Data Set.

Les figures VIII -1 et VIII -2 représentent respectivement d'un extrait fichier Facture.xml et Client.csv.

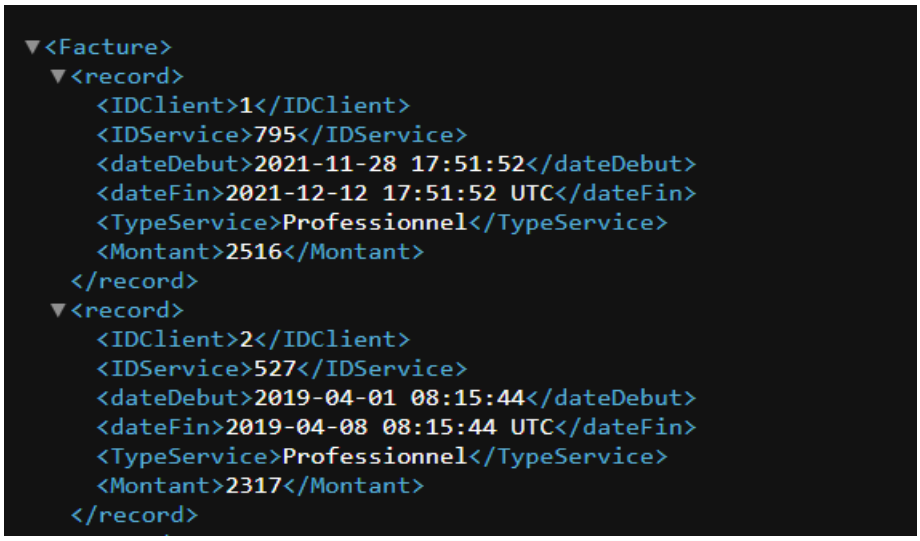


Figure VIII-1: Fichier facture.xml.

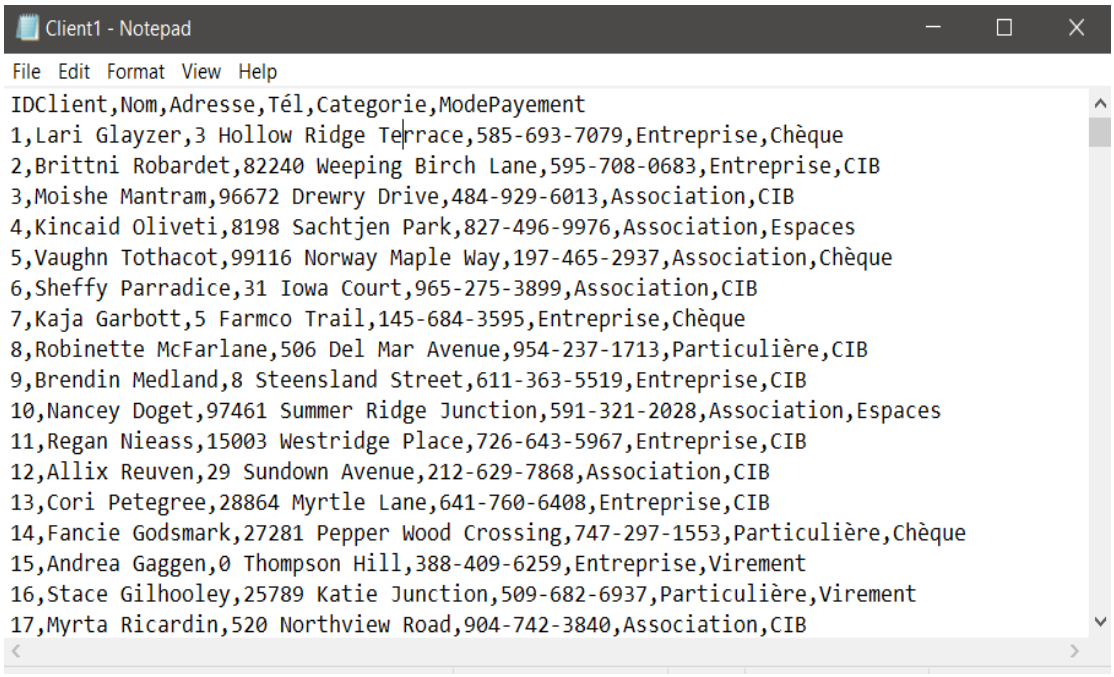


Figure VIII-2: Fichier client.csv.

2.3. Stockage des données dans le Data Lake HDFS

Hadoop HDFS est l'environnement Data Lake adopté pour notre système d'analyse OLAP. Afin d'ingérer notre jeu de données dans HDFS, il faut tout d'abord lancer l'invite de commande puis entrer les commandes HDFS suivantes :

* hdfs dfs –mkdir /datasets : Pour créer un fichier dans HDFS

* hdfs dfs –put C:\Users\windows\Downloads\Facture.xml /datasets/ : Pour l'ajout d'un document dans le fichier

* hdfs dfs –ls –R / : Pour consulter la liste des fichiers dans HDFS

* hdfs dfs –rm –R /datasets/Facture.xml : Pour la suppression d'un document

La figure VIII -3 représente le stockage des données dans HDFS (Data Lake).

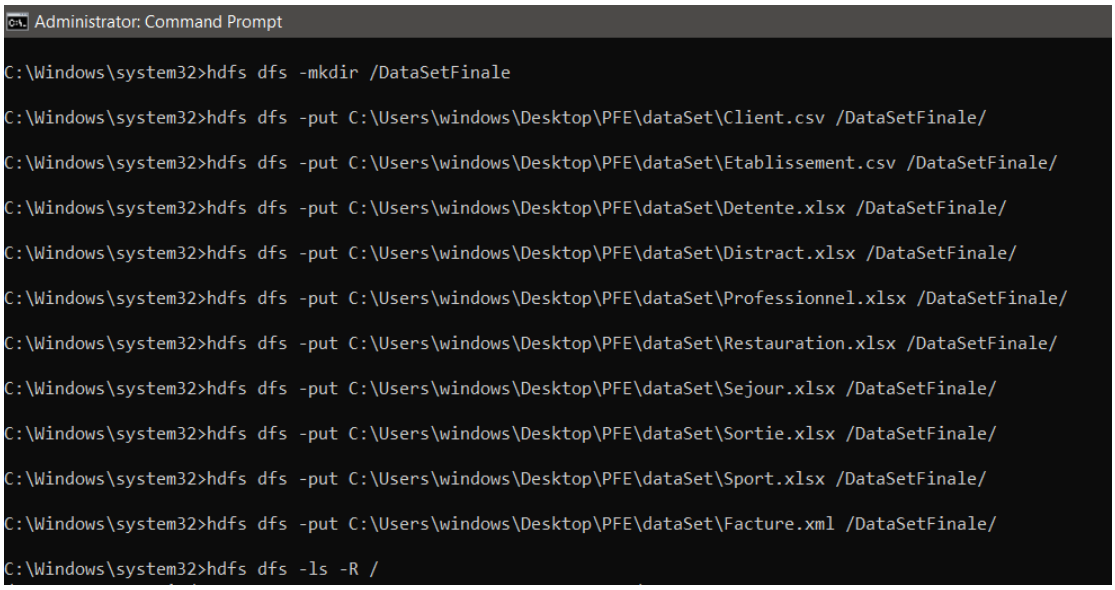


Figure VIII-3:Chargement de données dans HDFS.

3. Transformation de données

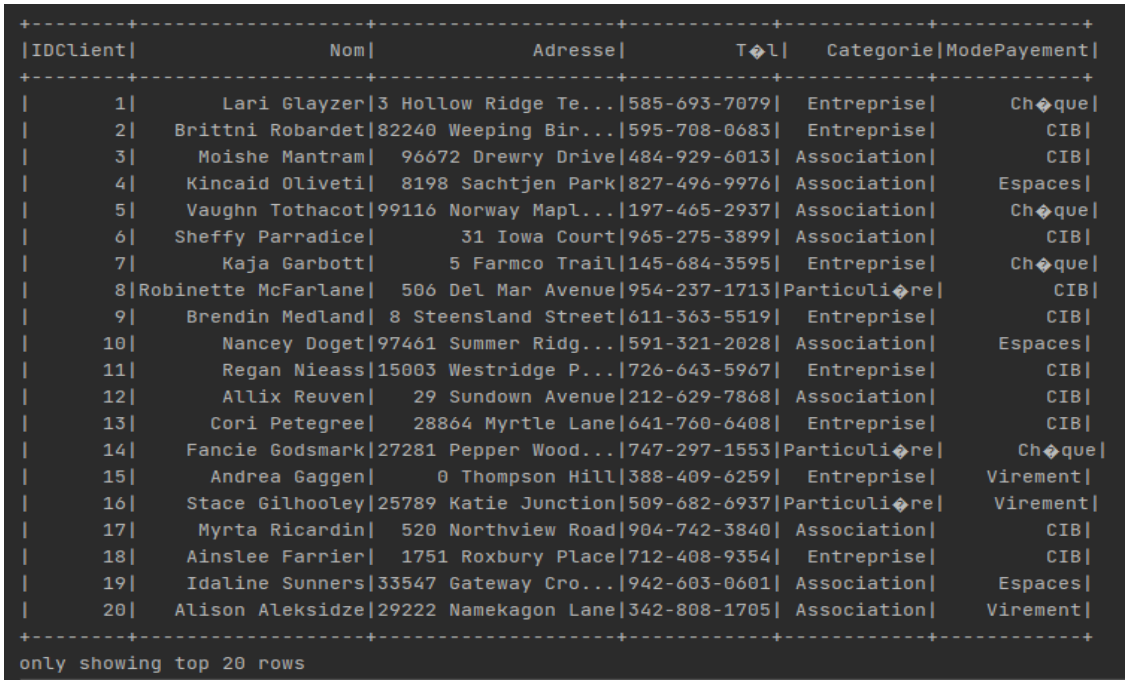
La première phase dans notre plate-forme est la transformation des données en vue d'adapter ces dernières aux objectifs d'analyse. Elle se compose de deux étapes, à savoir, extraction et lecture de données ainsi que le traitement de requête du cube.

3.1. Extraction et lecture de données

Le traitement commence par l'extraction et lecture des données à partir du Data Lake i.e. HDFS. Pour ce faire, nous utilisons Apache Spark et différents packages ainsi que la spécification du chemin des fichiers dans HDFS à savoir :

«/hdfs ://localhost:9000/NomFichier/NomDoc.csv »

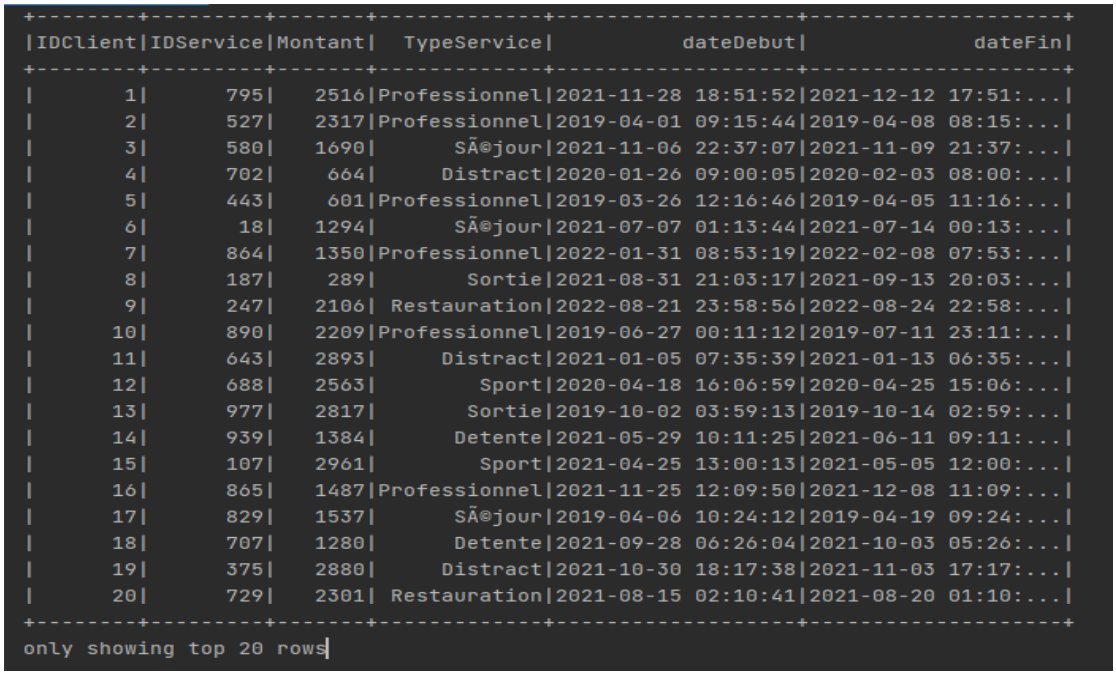
Les figures VIII-4 et VIII-5 montrent le résultat de la lecture du fichier client et du fichier facture.



IDClient	Nom	Adresse	T◆l	Categorie	ModePayement
1	Lari Glayzer	3 Hollow Ridge Te...	585-693-7079	Entreprise	Ch◆que
2	Brittni Robardet	82240 Weeping Bir...	595-708-0683	Entreprise	CIB
3	Moishe Mantram	96672 Drewry Drive	484-929-6013	Association	CIB
4	Kincaid Oliveti	8198 Sachtjen Park	827-496-9976	Association	Espaces
5	Vaughn Tothacot	99116 Norway Mapl...	197-465-2937	Association	Ch◆que
6	Sheffy Parradice	31 Iowa Court	965-275-3899	Association	CIB
7	Kaja Garbott	5 Farmco Trail	145-684-3595	Entreprise	Ch◆que
8	Robinette McFarlane	506 Del Mar Avenue	954-237-1713	Particuli◆re	CIB
9	Brendin Medland	8 Steensland Street	611-363-5519	Entreprise	CIB
10	Nancey Doget	97461 Summer Ridg...	591-321-2028	Association	Espaces
11	Regan Nieass	15003 Westridge P...	726-643-5967	Entreprise	CIB
12	Allix Reuven	29 Sundown Avenue	212-629-7868	Association	CIB
13	Cori Petegree	28864 Myrtle Lane	641-760-6408	Entreprise	CIB
14	Fancie Godsmark	27281 Pepper Wood...	747-297-1553	Particuli◆re	Ch◆que
15	Andrea Gaggen	0 Thompson Hill	388-409-6259	Entreprise	Virement
16	Stace Gilhooley	25789 Katie Junction	509-682-6937	Particuli◆re	Virement
17	Myrta Ricardin	520 Northview Road	904-742-3840	Association	CIB
18	Ainslee Farrier	1751 Roxbury Place	712-408-9354	Entreprise	CIB
19	Idaline Sunners	33547 Gateway Cro...	942-603-0601	Association	Espaces
20	Alison Aleksidze	29222 Namekagon Lane	342-808-1705	Association	Virement

only showing top 20 rows

Figure VIII-4: Dataframe de fichier client.



IDClient	IDService	Montant	TypeService	dateDebut	dateFin
1	795	2516	Professionnel	2021-11-28 18:51:52	2021-12-12 17:51:...
2	527	2317	Professionnel	2019-04-01 09:15:44	2019-04-08 08:15:...
3	580	1690	SÃ©jour	2021-11-06 22:37:07	2021-11-09 21:37:...
4	702	664	Distract	2020-01-26 09:00:05	2020-02-03 08:00:...
5	443	601	Professionnel	2019-03-26 12:16:46	2019-04-05 11:16:...
6	18	1294	SÃ©jour	2021-07-07 01:13:44	2021-07-14 00:13:...
7	864	1350	Professionnel	2022-01-31 08:53:19	2022-02-08 07:53:...
8	187	289	Sortie	2021-08-31 21:03:17	2021-09-13 20:03:...
9	247	2106	Restauration	2022-08-21 23:58:56	2022-08-24 22:58:...
10	890	2209	Professionnel	2019-06-27 00:11:12	2019-07-11 23:11:...
11	643	2893	Distract	2021-01-05 07:35:39	2021-01-13 06:35:...
12	688	2563	Sport	2019-04-18 16:06:59	2020-04-25 15:06:...
13	977	2817	Sortie	2019-10-02 03:59:13	2019-10-14 02:59:...
14	939	1384	Detente	2021-05-29 10:11:25	2021-06-11 09:11:...
15	107	2961	Sport	2021-04-25 13:00:13	2021-05-05 12:00:...
16	865	1487	Professionnel	2021-11-25 12:09:50	2021-12-08 11:09:...
17	829	1537	SÃ©jour	2019-04-06 10:24:12	2019-04-19 09:24:...
18	707	1280	Detente	2021-09-28 06:26:04	2021-10-03 05:26:...
19	375	2880	Distract	2021-10-30 18:17:38	2021-11-03 17:17:...
20	729	2301	Restauration	2021-08-15 02:10:41	2021-08-20 01:10:...

only showing top 20 rows

Figure VIII-5: Dataframe de fichier facture.

3.2. Traitement de requêtes du cube

Les cubes OLAP se construisent selon les besoins et les objectifs d'analyse des décideurs. Par exemple :

> Analyse des **chiffres d'affaire** (∑*montants*) par Mode de paiement, Type de services, Date et Catégorie du client.
>
> - *L'aspect temporel pour cette analyse correspond aux dates de début des périodes de consommation. Les mois doivent être concaténés avec l'année correspondante : la date '03/2014' correspond à Mars 2014.*

Le schéma en étoile pour l'analyse en question est modélisé dans la figure VIII-6, où :

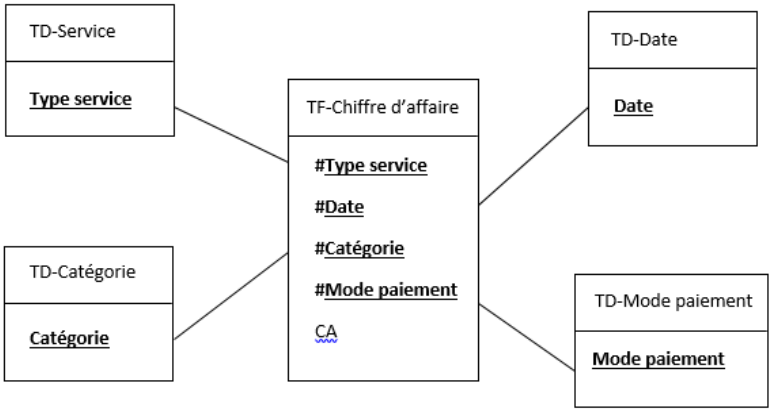


Figure VIII-6:Schéma en étoile.

La requête SQL correspondante à la construction du cube est la suivante :

" SELECT F.TypeService , C.Categorie , C.ModePaiement ,
CONCAT(CAST(MONTH(F.DateDebut) AS CHAR (2)) ,'/', CAST(YEAR(F.DateDebut)AS CHAR(4))) AS Date , SUM (F.Montant) AS CA FROM Facture AS F

JOIN Client AS C ON F.IDClient = C.IDClient

GROUP BY F.TypeService ,C.Categorie , C.ModePaiement ,
CONCAT(CAST(MONTH(F.DateDebut) AS CHAR (2)) ,'/', CAST(YEAR(F.DateDebut)AS CHAR(4))) "

L’exécution de cette requête se fait par Spark SQL après avoir unifier le format de chaque fichier en une vue temporaire des dataframes.

```
reqCube1 = spark.sql("select F.TypeService , "
                     "CONCAT(CAST(MONTH(F.DateDebut ) AS CHAR (2)) ,'/', CAST(YEAR(F.DateDebut)AS CHAR(4))) AS Date ,"
                     " C.Categorie , C.ModePayement , sum(F.Montant) as CA "
                     "from Facture as F join Client as C on F.IDClient = C.IDClient "
                     "group by F.TypeService , C.ModePayement , C.Categorie , "
                     "CONCAT(CAST(MONTH(F.DateDebut ) AS CHAR (2)) ,'/', CAST(YEAR(F.DateDebut)AS CHAR(4)))")
reqCube1.show()
```

Figure VIII-7:Traitement requête par Spark SQL.

La figure VIII -7 représente un extrait du code qui traite la requête du cube de l’exemple abordé, en revanche, la figure VIII -8 et le résultat affiché i.e. Le Cube en dataframe.

TypeService	Date	Categorie	ModePayement	CA
Professionnel	11/2021	Entreprise	Ch\u00fffque	2516
Distract	10/2021	Association	Espaces	2880
Professionnel	4/2019	Entreprise	CIB	2317
Professionnel	8/2019	Association	Virement	2020
Sortie	8/2021	Particuli\u00fffre	CIB	289
Detente	7/2019	Particuli\u00fffre	Virement	229
Sport	4/2020	Association	CIB	2563
Detente	9/2021	Entreprise	CIB	1280
Detente	6/2020	Association	Virement	468
Detente	5/2021	Particuli\u00fffre	Ch\u00fffque	1384
Distract	1/2021	Entreprise	CIB	2893
Distract	1/2020	Association	Espaces	664
Restauration	8/2022	Entreprise	CIB	2106
Detente	10/2020	Entreprise	Ch\u00fffque	2764
Detente	1/2019	Particuli\u00fffre	Ch\u00fffque	2086
Professionnel	6/2019	Association	Espaces	2209
Sortie	10/2019	Entreprise	CIB	2817
Professionnel	11/2021	Particuli\u00fffre	Virement	1487
Detente	8/2021	Association	CIB	281
SÃ©jour	3/2021	Entreprise	Ch\u00fffque	2007

only showing top 20 rows

Figure VIII-8:Dataframe cube 1.

La figure VIII -9 représente l'interface indiquant la fin de la transformation avec succès.

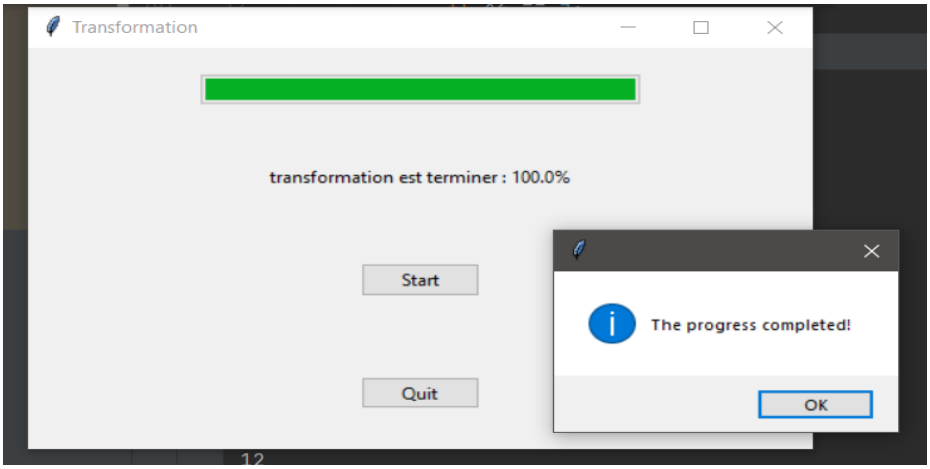


Figure VIII-9: Interface progrès de transformation.

4. Stockage des cube OLAP -NoSQL

Les cubes OLAP construit par la première phase vont être traduit en cube OLAP -NoSQL orienté documents, pour ce fait, des règles de passage de [33] ont été appliqué (Voir chapitre approche pour les détails).

L'algorithme ci-dessus décrit la technique utilisée afin de charger les cubes OLAP-NoSQL dans la base de données NoSQL orienté documents à savoir MongoDB.

Algorithme VIII-1 : Chargement cube OLAP-NoSQL

```
Entrées :
1  ReqCube : dataframe du cube
2  i : entier  // i étant le rang du reqCube
Sortie :
3  Collection de documents MongoDB
4  début
5      Pour i ← 1 à ( nbLigne (reqCube))
6          Doc = {'Dim1' : [{'Att1 : reqCube.collect()[i][0]}],
7          'Dim2' :  [{'Att2 : reqCube.collect()[i][1]}],
8          'Dim3' :  [{'Att3 : reqCube.collect()[i][2]}],
9           'Mesure' : reqCube.collect()[i][3] }
10     Finpour
11 Fin
```

Un exemple de cube stocké dans MongoDB est illustré dans la figure VIII-10, où chaque dimension est un document imbriqué (Attribut composé) et chaque mesure est présenté par un attribut simple.

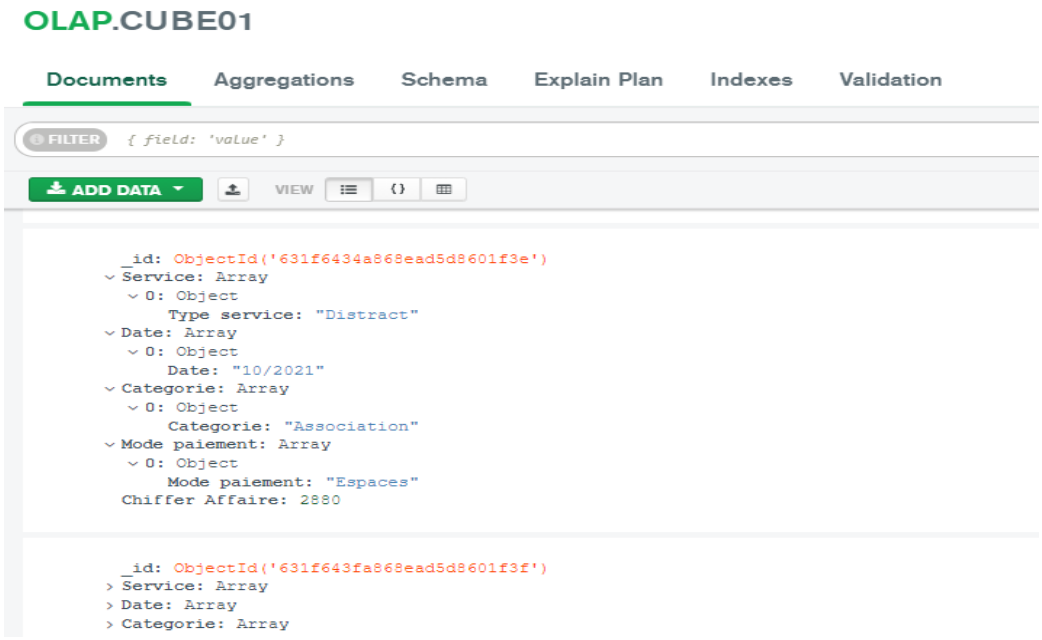


Figure VIII-10: Cube NoSQL dans MongoDB.

5. Opérations OLAP-NoSQL

Après avoir stocké les cubes OLAP-NoSQL dans le SGBD MongoDB, notre plate-forme établit une connexion avec la base des cubes MongoDB. Afin de consulter les cubes, l'utilisateur entre le nom du cube souhaité dans l'interface d'analyse.

La figure VIII-11 représente l'affichage des cubes.

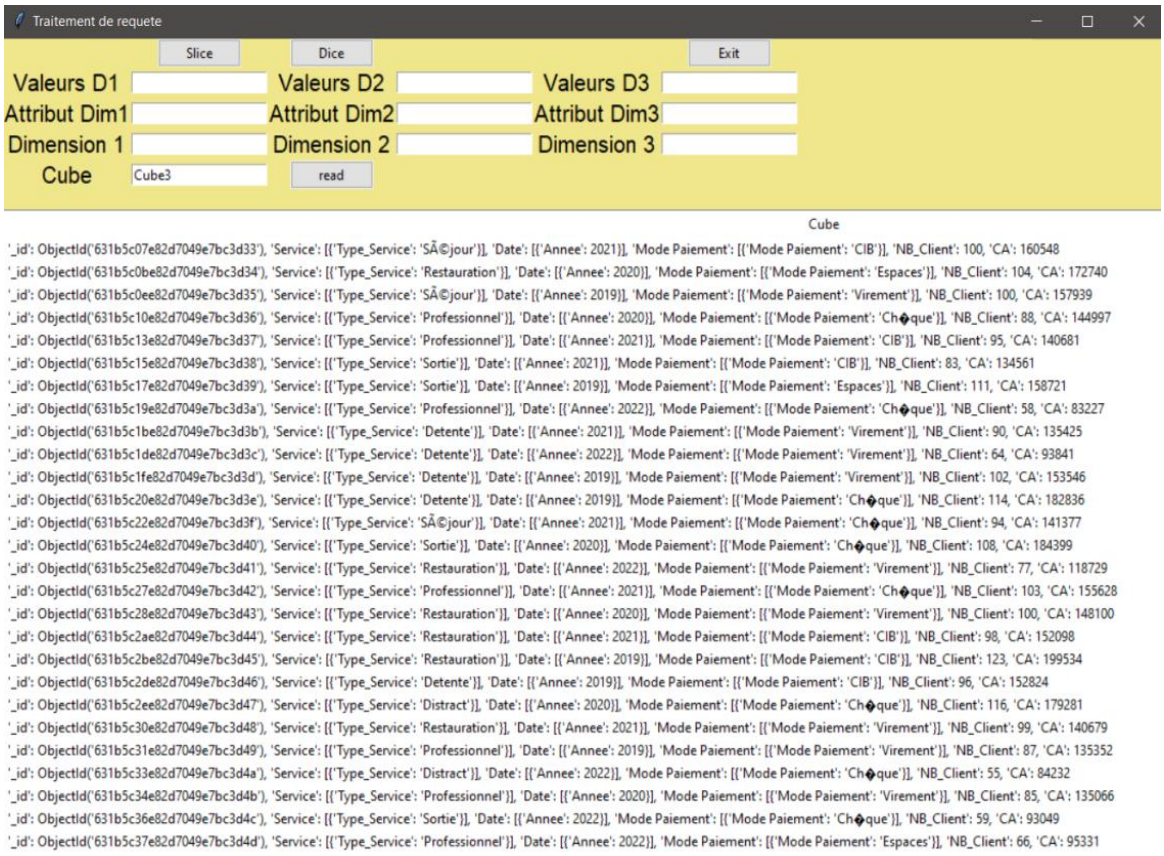


Figure VIII-11:Interface affichage des cubes.

Les opérateurs et opérations OLAP-NoSQL proposés sont implémentés dans cette partie. L'utilisateur peut effectuer des opérations sur le cube OLAP-NoSQL telles que Slice, Dice en spécifiant les dimensions ainsi que les filtres associés.

La figure VIII -12 montre un exemple opération Dice sur le cube 1où le critère est d'afficher juste les lignes où type service='Sortie' ou 'Sport' dans la dimension 'Service' et Mode de paiement ='CIB' dans la dimension 'Mode Paiement'.

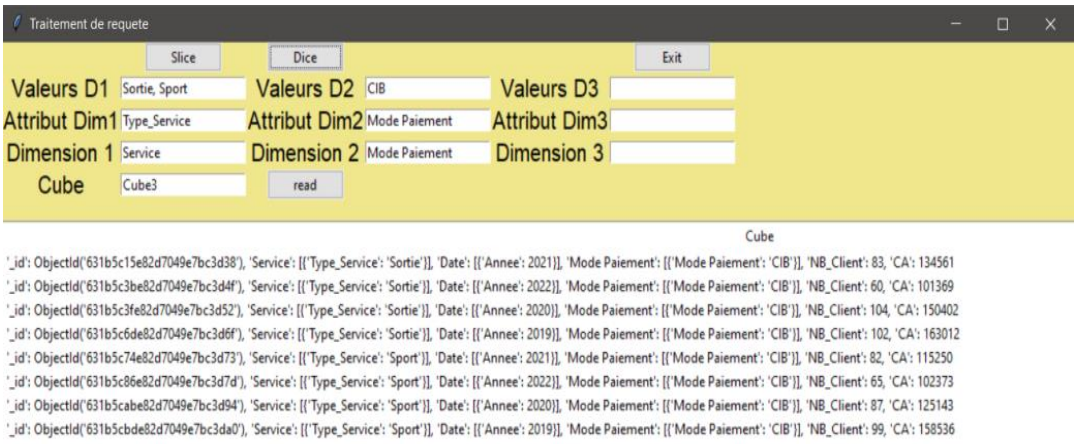


Figure VIII-12:Interface opération Dice.

6. Conclusion

Ce chapitre est dédié aux aspects d'implémentation et de test de notre travail. Nous avons présenté le prototype développé dans le cadre de ce mémoire, à savoir OLAP dans environnement Data Lake en exposant les principaux composants de la plate-forme ainsi que le jeu de données utilisé dans le test de notre plate-forme.

Conclusion générale

1. Conclusion

Dans le cadre de notre projet de PFE, nous avons traité un sujet qui s'intitule « Analyse des données en ligne (OLAP) dans un environnement Data Lake » et dont l'objectif premier est de mettre en œuvre un environnement OLAP où les données sources sont issues d'un Data Lake et les cubes de données sont stockés dans un modèle NoSQL. Après une revue de la littérature dans les domaines de l'informatique décisionnelle, les méga-données, les data lakes et les modèles NoSQL, nous avons étudié des travaux connexes afin de faire toute la lumière sur les avancées dans le domaine que nous traitons dans le cadre de ce mémoire. Une étude comparative suivi d'un bilan a été établis afin de positionner notre travail.

Nous avons par la suite exposé notre approche qui traite sur les analyses de données en ligne OLAP dans un environnement Data Lake. Notre objectif principal était de permettre la transformation des données issues du Data Lake en vue de les transformer en cubes de données exploitables pour des fins d'analyse.

Lors de la construction des cubes OLAP plusieurs aspects seront pris en charge par notre plate-forme tel que le format des données et leur transformation. Les cubes générés doivent être stockées dans un SGBD NoSQL orienté documents, à savoir MongoDB. Pour ce faire, nous avons utilisé des règles de transformation permettant de passer depuis un modèle conceptuel multidimensionnel (modèle en étoile) vers un modèle logique NoSQL (orienté documents). La plate-forme englobe trois composants importants à savoir la phase de transformation, stockage des cubes OLAP-NoSQL et traitement de requêtes.

La troisième partie de ce mémoire a été consacrée à l'implémentation, où le chapitre 7 est réservé à l'environnement de développement .Nous avons présenté dans ce chapitre la bibliothèque logicielle Apache Hadoop, Apache Spark, MongoDB et le langage de programmation utilisé .Le chapitre 8 concerne le développement de la plate-forme , nous avons réalisé un programme qui content quatre modules principaux à savoir (1) module d'extraction des données à partir du Data Lake, (2) module de transformation des données (Phase T de l'ELT) afin d'intégrer les données hétérogènes dans une structure homogène

sous forme d'un cube OLAP, (3) module de stockage des cubes OLAP NoSQL et enfin (4) module de visualisation des résultats.

La réalisation de ce projet de fin de cycle de Master nous a permis d'acquérir une bonne expérience dans un domaine qui est à la croisée des chemins entre l'informatique décisionnelle et les big data et plus précisément l'analyse des données en ligne (OLAP), les data lakes et les modèles NoSQL. L'apport du projet est considérable sur le plan pratique, car il nous a permis d'acquérir davantage de connaissances sur les techniques de configuration et de programmation.

Toutes les tâches de notre projet nous ont permis d'enrichir nos expériences dans l'organisation, la démarche et la cohérence entre les déférentes parties du projet et aussi, de nous familiariser avec plusieurs outils tels que Spark et MongoDB.

2. Travaux futurs

Comme tout projet, notre PFE pourra faire l'objet d'une continuité en vue d'apporter des améliorations et des extensions. Notre travail a consisté à aborder un aspect de l'analyse des données OLAP dans un environnement Data Lake. Nous pensons que notre travail peut encore être amélioré, et la liste ci-dessous présente les travaux futurs que nous prévoyons d'intégrer.

Format des données :

Comme notre travail se fait dans un écosystème des lacs de données donc nous visons à intégrer plus de format de données dans la phase de transformation.

Passage à l'échelle :

Passer à l'échelle en menant une analyse expérimentale sur des Benchmarks connus. Nous visons, particulièrement, des volumes de données plus importants et par conséquent tester en mode clusters de grande capacité que ce soit en nombre de nœuds, en capacité de calcul et de mémoire.

Algèbre d'OLAP :

Afin d'étendre les capacités d'analyse offertes aux décideurs, nous comptons à intégrer plusieurs d'autres opérations OLAP-NoSQL, tel que, Roll-up, Drill Down, Pivot, Drill-in, Drill-out.

Bibliographie

[1] Inmon, B. (2016). *Data Lake Architecture: Designing the Data Lake and avoiding the garbage dump.* Technics publications. pp.16, 24

[2] Elsa Negre, *Système de recommandation : Introduction*, ISTE Groupe, 2015, 82pages, p.10

[3] Inmon, W. H., *The data warehouse and data mining.* Communications of the ACM, 1996, 39(11), pp.49-51

[4] Dernoncourt, F. &. Métais, E., *La Logique Floue : le raisonnement humain au cœur du système décisionnel.* Memory NFE211 engineering decision systems, Paris, 2011, p.3

[5] Bala, M., *Parallélisation et distribution du processus d'intégration ETL pour le traitement de données massives.* Thèse de doctorat, FACULTE d'Electronique et Informatique, UNIVERSITE HOUARI BOUMADIENE, 2017, pp.28, 31

[6] LARBI, A., *Conception des entrepôts de données : Evaluation des besoins flexibles.* Thèse de doctorat, FACULTE des sciences exactes, UNIVERSITE DJILLALI LIABES, Sidi belabbas, 2018, p. 10.

[7] Younsi, F. Z., *Mise en place d'un Système d'Information Décisionnel pour le Suivi et la Prévention des Epidémies.* Thèse de doctorat, ECOLE de doctorale Informatique et Mathèmatiques, UNIVERSITE of Lyon. 2016, pp. 61-62, 65-66.

[8] [Kimball, 1996] Kimball, R. (1996). *The data warehouse toolkit : practical techniques for building dimensional data warehouse*. NY : John Willey & Sons, 248:4.

[9] Jiofack, M. R., *Intégration d'un système décisionnel d'aide à la prise de décisions aux campagnes marketing et commerciales dans les entreprises de vente en ligne : Data Warehouse et Business Intelligence*, GRIN Verlag, thèse Master, 2020, p. 12

[10] Bouillot, F., *Acquisition, modélisation et mise en œuvre d'un entrepôt de données pour l'analyse d'informations issues de twitter,* Mémoire d'Ingénieur CNAM, Montpellier, 2011, p. 21-22

[11] Wehrle, P. *Modèle multidimensionnel et OLAP sur architecture de grille. Informatique,* Lyon, École doctorale Informatique et Information pour la Société, Institut National des Sciences Appliquées de Lyon. 2009, pp.21-22

[12] Ghozzi, F. ,Ravat, F., Teste, O. & Zurfluh, G., *Contraintes pour modèle et langage multidimensionnels,* UNIVERSITE PAUL SABATIER, 2010, p.2.

[13] KHALIS, M. & Merzak, A., *Business Intelligence ETL (Extract, Transform & Load), Quelles limites ? REVUE DE L'ENTREPRENEURIAT ET DE L'INNOVATION,* FACULTE des sciences BEN M'SIK, 2020, pp. 106-107.

[14] RIVEST., S. *Investigation des modes d'intégration physique entre un serveur de base de données multidimensionnelle et un SIG.* FACULTE de foresterie de géomatique, UNIVERSITE LAVAL, Canada, 2000, p. 5.

[15] Teste, O. *Modélisation et manipulation des systèmes OLAP : de l'intégration des documents à l'usager.* Thèse de doctorat, UNIVERSITE PAUL SABATIER -Toulouse III, 2009, p. 7.

[16] ZIYATI, E. *Optimisation de requêtes OLAP en Entrepôts de Données : Approche basée sur la fragmentation génétique.* Thèse de doctorat, FACULTE de science, UNIVERSITE MOHAMMED, 2010, p.20.

[17] Tchounikine, A., Miquel, M., Laurini, R., Ahmed, T. O., Bimonte, S., & Baillot, V., *Panorama de travaux autour de l'intégration de données spatio-temporelles dans les hypercubes.* France, 2005, p.21.

[18] Meguireche, A., *Conception et implémentation d'un système décisionnel TAOUAB-Bousaâda.* Doctoral dissertation, Mémoire de Master Académique FACULTE Mathématique et Informatique, UNIVERSITE MOHAMED BOUDIAF - M'SILA, 2020, pp. 6, 18-19.

[19] Tournier, R., *Analyse en ligne (OLAP) de documents.* Thèse de doctorat, UNIVERSITE PAUL SABATIER -Toulouse III, 2007, p.48

[20] Numerica. *Le Big Data : Que fait-on de nos données numériques ?* BoD - Books on Demand, 2020, p. 9.

[21] Jean-Louis Monino, S. S., *Volume 4 Big Data, Open Data et valorisation des données,* 2016, pp.11-13.

[22] Mehta, B. B., Rao, U. P., Kumar, N., & Gadekula, S. K. Towards privacy preserving big data analytics. In *Proceedings of the 2016 Sixth Int. Conf. Advanced Computing and Communication Technologies, Ser. ACCT-2016, Rohtak, India: Research Publishing* (pp. 28-35)2016 .p.28

[23] Ouahab, R. & Boukraa, D. E., *Développement d'un système de gestion des méta-données dans les data lakes à base de sources NoSQL.* Thèse de doctorat, UNIVERSITE de jijel, 2021, pp.11, 13

[24] Abderrazak, M., *L'archives à l'ère du numérique*, ISTE Groupe, 2021, p.121

[25] Hashem, H. *Le traitement BigData : Informatique.* Publishroom, 2021, p.10

[26] Soraya Sedkaoui, M. K. *L'economiede partage et le Big Data analytics*. ISTE Group,2019, p. 148.

[27] Ziani Khouloud, S. M. *Outil Graphique de génération de Banchmarks NoSQL (OGGB-NoSQL).* Mémoire Masetr Ingénierie Logiciel, Département Informatique.Université de SAAD DAHLEB, 2019, p. 23, 25, 26.

[28] Hamdane, M. E. *Bases de données avancées : Du relationnel au NoSQL: Cours et travaux pratiques*. Éditions Al-Shafi'i-.2018, pp..139, 142.

[29] Ravat, F. et Y. Zhao (2019a). Data lakes : Trends and perspectives. In Database and Expert Systems Applications, pp. 304–313.

[30] Sautot, L. *Business Intelligence & Big Data. Montpellier France 2019: 15ème Edition de conference EDA*. BoD - Books on Demand, 2019, p. 79-80.

[31] DJEDAINI, M., MALKI, J., & BOUJU, A. *Architecture Big Data open source pour l'Informatique Décisionnelle.* UNIVERSITE LA ROCHELLE, 2018, pp. 11-12, 15.

[32] Hadfi A., Amouboudi D., Implementation of heterogeneous data ingestion framework in a data lake environment, 2021, p.39.

[33] Khalil, A., & Belaissaoui, M., *Implementing big OLAP Cubes using a NoSQL-Based approach: Cube models and aggregation operators. In Enabling Machine Learning Applications in Data Science,* 2021, pp. 179-191.

[34] Davardoost, F., Babazadeh Sangar, A., & Majidzadeh, K., *An Innovative Model for Extracting OLAP Cubes from NOSQL Database Based on Scalable Naïve Bayes Classifier*. Mathematical Problems in Engineering, Iran, 2022.

[35] Chevalier, M., El Malki, M., Kopliku, A., Teste, O., & Tournier, R., *Entrepôts de données multidimensionnelles NoSQL,* UNIVERSITE de Toulouse, France. 2015.

[36] Rey, P., *Mation Initiale Python avec Jupyter et PyCharm*. BoD - Books on Demand, 2022, p. 19.

[37] Nair, P., *Beginning Apache Hadoop Administration: The First Step towards Hadoop Administration and Management.*Notion Press, 2017, p.6

[38] Shrivastava, A., & Deshpande, T., *Hadoop blueprints*. Packt Publishing Ltd, 2016, p. 15-19.

[39] El Malki, M. *Modélisation NoSQL des entrepôts de données multidimensionnelles massives*, Thèse de doctorat, UNIVERSITE Toulouse le Mirail-Toulouse II, 2016, p.28.

[40] Quddus, J., *Machine Learning with Apache Spark Quick Start Guide: Uncover patterns, derive actionable insights, and learn from big data using MLlib*. Packt Publishing L, 2018, p. 30.