PROGRAMME FEDER-FSE 2014-2020

Rapport de recherche

Laboratoire L3i, La Rochelle Université

**********

**Discipline : BIG DATA AND CLOUD COMPUTING**

# Architecture Big Data open source pour l'Informatique Décisionnelle

Par : **Mahfoud DJEDAINI - Ingénieur de recherche au L3i**

**Sous la direction de**

Jamal MALKI, Maître de Conférences, Université de La Rochelle, Lab. L3i

Alain BOUJU, Maître de Conférences HDR, Université de La Rochelle, Lab. L3i

,

**Date du rapport :** JUILLET 2018

# Clauses de confidentialité

Le rapport de recherche fourni reste la propriété du projet FEDER PLAIBDE.

Les informations, les données, les figures, les modèles, etc. contenues dans ce rapport sont strictement couvertes par le secret professionnel (article 226-13 du code pénal). Conformément à l'article 34 de la loi « Informatique et Libertés », le destinataire de ce rapport s'engage à prendre toutes précautions utiles afin de préserver la sécurité des informations et notamment d'empêcher qu'elles ne soient déformées, endommagées ou communiquées à des personnes non autorisées.

Le destinataire de ce rapport s'engage donc à respecter les obligations suivantes et à les faire respecter par son personnel :

— ne faire aucune copie du rapport, ni aucune publication publique (web et autre) sans l'accord préalable du consortium du projet PLAIBDE ;

— ne pas divulguer ce rapport à d'autres personnes, qu'il s'agisse de personnes privées ou publiques, physiques ou morales ;

— prendre toutes mesures permettant d'éviter toute utilisation détournée ou frauduleuse de ce rapport car il porte sur des contrats en cours d'exécution ;

— prendre toutes mesures de sécurité, notamment matérielles, pour assurer la conservation et l'intégrité de ce rapport.

Le consortium du projet PLAIBDE se réserve le droit de procéder à toute vérification qui lui paraîtrait utile pour constater le respect des obligations précitées.

En cas de non-respect des dispositions précitées, la responsabilité du titulaire peut également être engagée sur la base des dispositions de l'article 226-17 du code pénal.

La CNIL a publié un guide téléchargeable sur son site Internet : `www.cnil.fr` relatif à la sécurité des données personnelles et qui présente un ensemble de préconisations sur la sécurisation des systèmes d'informations.

# Table des matières

Chapitre 1

# Introduction

## 1 L'entreprise aYaline

aYaline est une société proposant des solutions informatiques basées sur des solutions open source. La société a des clients allant des entreprises aux organisations gouvernementales. Par exemple, elle propose des outils et services autour du commerce électronique, du e-tourisme, de la gestion de communautés (intranet, extranet, dématérialisation...), ... La spécificité de aYaline est de baser toutes ses solutions sur de l'open source, lui permettant alors de bénéficier de tous les avantages de telles solutions, telle que la gratuité des outils et l'importance et la disponibilité du support communautaire, pour ne citer que ces deux aspects.

## 2 Problèmes & Limitations

Depuis quelques années, avec l'utilisation de plus en plus massive des données provenant des réseaux sociaux et l'open data, aYaline se retrouve confrontée à des problématiques de volumétrie des données. En effet, les données deviennent tellement volumineuses pour certains clients, que les outils traditionnels utilisés pour exploiter ces données ne sont plus tout à fait adaptés. Dans certains cas, les outils ne sont tout simplement pas utilisables. Dans d'autre cas, les outils fonctionnent mais sont tellement lents qu'au final, leur utilisation n'est pas agréable ni même profitable.

Afin d'illustrer ce propos, nous pouvons citer le cas des clients d'aYaline pour lesquels aYaline fournit des solutions décisionnelles. Pour ces clients, le service proposé par aYaline est constitué de deux étapes importantes

— **ETL (Extract - Transform - Load)** Cette étape consiste à étudier l'ensemble des sources de données liées à l'activité du client, puis de proposer et créer un entrepôt de ces données qui soit centralisé, tout en assurant la cohérence ainsi que la qualité des données avant de les injecter dans l'entrepôt en question.

— **Analyse** Cette étape consiste à fournir des outils d'analyse, idéalement relativement simples d'utilisation et adaptés aux utilisateurs finaux, leur permettant d'analyser aisément les données de l'entrepôt, avec la possibilité de produire des rapports sous formes de tableaux, graphiques, ...

La première étape d'ETL, qui était déjà une étape complexe à l'ère des données "traditionnelles", devient encore plus complexe à l'ère du Big Data. En effet, la multiplication des sources de données disponibles rend les schémas de données encore plus complexes, et complexifie donc le processus d'intégration de ces données. De plus, au delà de la complexité des schémas, l'important volume des données ainsi que leur vélocité a un impact important sur les performances des processus.

En ce qui concerne la partie Analytics, qui consiste donc à analyser les données une fois intégrées dans l'entrepôt, la limitation principale rencontrée est relative à la performance des outils. Par exemple, certaines requêtes analytiques (pourtant pas si complexes) qui s'éxecutaient en quelques secondes avec un volume de données raisonnable, s'éxecutent maintenant en un temps dépassant plusieurs minutes !

## 3 Périmètre de nos objectifs

C'est dans le contexte décrit dans les lignes ci-dessus que s'inscrit notre mission. Notre objectif est d'intervenir en amont (ETL) et en aval (Analytics) de la chaîne décisionnelle, afin de proposer des solutions aux problématiques et limitations décrites ci-dessus.

Au final, les solutions proposées doivent être principalement basées sur des outils open source, respectant ainsi la philosophie et les contraintes de la société aYaline.

Améliorer la stratégie ETL Concernant la partie ETL, nous devrions proposer une solution qui permette d'effectuer un ETL plus facilement, et si possible plus visuel et convivial.

Aussi, ne figurant pas dans le cahier des charges initial mais tendant vers l'idéal, il serait intéressant d'étudier la mise en place d'un ETL temps réel, permettant l'intégration des données en continu.

## 4 Contexte experimental

Tout d'abord, avant de détailler nos propositions, nous décrivons de manière succincte le contexte expérimental dans lequel nous évoluons. Nous avons les bases de données opérationnelles mises en place par aYaline, mais en plus de cela nous avons aussi les entrepôts qui ont été déjà conçus par aYaline en utilisant l'approche traditionnelle ETL. Le fait d'avoir les données brutes ainsi que les données transformées et directement requêtables nous permet d'avoir une baseline pour être bien sûrs que les résultats obtenus avec nos propositions d'architecture soient jsutes et cohérents.

Nous utiliserons les données dont nous disposons dans un cadre de validation de nos approches. Il ne s'agit pas ici de restreindre la portée de nos approches, qui elles se doivent d'être généralisables et applicables à n'importe quel jeu de données convenant.

## 4.1 L'application VTiger pour la relation client

Nous avons à disposition les bases de données utilisées en production par l'office de tourisme pour gérer sa relation client. L'application utilisée est VTiger, qui est un outil CRM connu et répandu. Les données avec lesquelles nous travaillons sont donc issues de la base de données sous-jacente à VTiger, avec néanmoins des modifications apportées par la société.

Vtiger est une application intégrée de gestion de la relation client (CRM) qui peut être utilisée sur l'intranet ou sur Internet à l'aide d'un navigateur. Il est distribué sous licence libre. Cet ouil aide les employés de l'organisation à gérer leur entreprise dans les domaines suivants :

— gestion de l'activité commerciale (comptes et prospects, contacts, devis, factures, ...)
— assistance ou assistance aux clients et service après-vente
— gammes de produits ou de services
— analyse d'activité à travers des rapports et des tableaux de bord
— agenda partagé, mémos et suivi des activités ...

## 4.2 Entrepôts disponibles

A partir des données de VTiger, ainsi que d'autres sources de données, aYaline a conçu différents entrepôts de données pour répondre aux besoins en analyse. Ici, nos détaillons les différents cubes à notre disposition.

### L'entrepôt BI-CMT-Demandes

L'entrepôt BI-CMT-Demandes contient les données issues des personnes ayant effectué une demande pour du tourisme auprès de l'office de tourisme. Le schéma du cube *Demandes* est présenté à la figure 1.1.

— **Table de faits**
Les faits suivis sont les demandes faites par les clients. La table des faits contient donc essentiellement un enregistrement par demande faite. Chaque enregistrement est ensuite lié à des données le concernant à travers les tables de dimensions, décrites ci-dessous.

— **Dimension thème**
Les thèmes sont extraits du SIT et renseignent sur la nature de la demande (Hébergement,Camping, etc.). Les thèmes ombrelles sont la base de la structure, puis viennent ensuite les thèmes génériques qui précisent le second niveau des thèmes ombrelles. Ce modèle inclut aussi une structure hiérarchique de tous les thèmes en définissant des relations d'héritage parent-fils

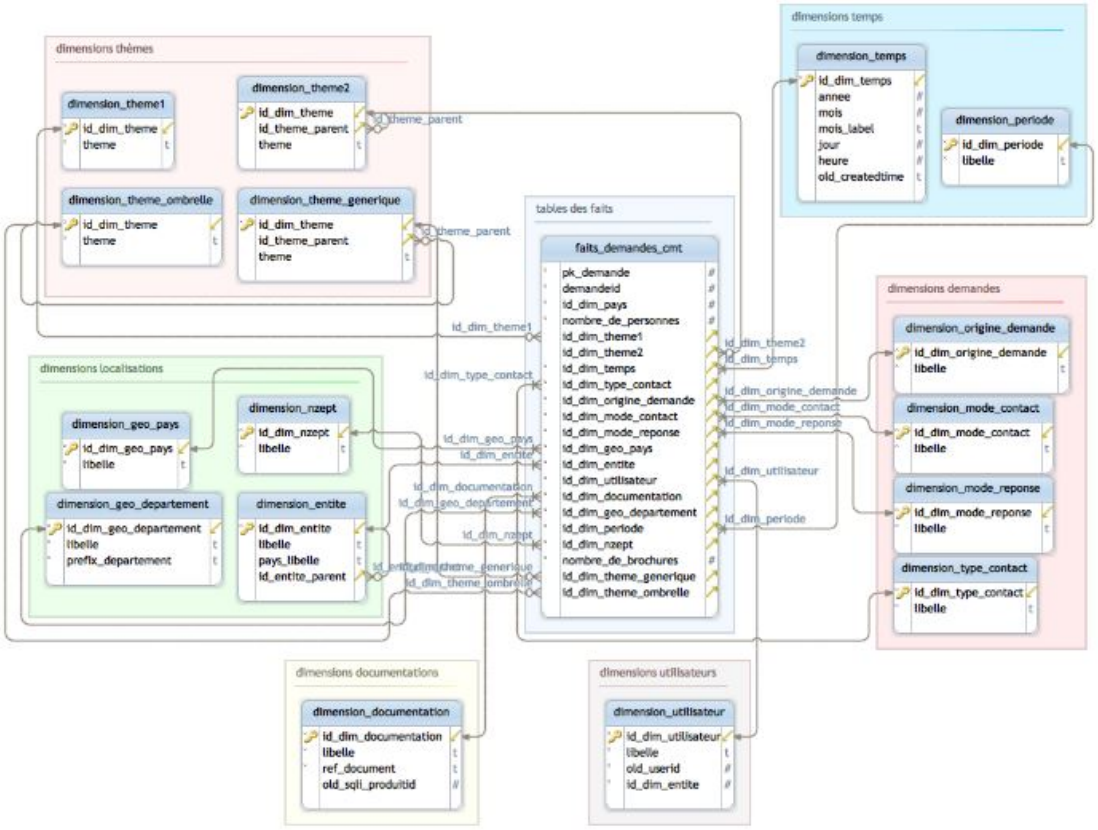


FIGURE 1.1 – Entrepôt BI-CMT-Demandes

— **Dimension localisation**
Chaque demande est localisée par rapport à un ensemble de données territoriales. On trouve l'entité, le département, le pays (au sens touristique), etc.

— **Dimension temps**
Chaque demande est datée à sa création dans le système

— **Dimension demande**
Chaque demande possède des méta-données concernant son origine, le mode de contact et de réponse associée, le type de contact, etc.

— **Dimension documentation**
Chaque demande peut être associée à une documentation, extraite du SIT, et fournie au demandeur

— **Dimension utilisateur**
Chaque demande peut être associée à l'utilisateur du système qui a saisi la demande

#### L'entrepôt BI-CMT-SIT

Le projet BI-CMT-SIT porte sur l'analyse des données concernant les produits touristiques. Ces derniers sont de différents types : restauration, hébergement, hôtellerie, événement, ... Tous les produits touristiques disposent d'attributs en commun, amis chaque type de produit touristique a aussi des attributs spécifiques à ce type. Deux niveaux de modélisation ont été conçus : 1) un cube global utilisant contenant tous les produits touristiques, en ne considérant que les attributs qu'ils ont en commun 2) un cube spécifique pour chaque type de produit touristique, considérant en plus les attributs spécifiques des types, et permettant ainsi une analyse plus poussée

## 5 Organisation du travail

Dans ce document, nous présentons plus en détails les problématiques abordées. Tout d'abord, dans un premier chapitre 2, nous présentons un état de l'art des solutions liées à notre problématique, tout en nous concentrant sur les outils des écosystèmes Big Data. Dans le chapitre suivant, nous décrivons nos propositions d'outils et d'architecture pour répondre aux problématiques dans notre contexte de tavail. Enfin, dans un dernier chapitre avant de conclure, nous présentons la progression de notre travail sur chacune de nos propositions, ainsi que sur la mise en place des infrastructures techniques, qui ont représenté une part non négligeable du capital temps.

Chapitre 2

# L'Informatique Décisionnelle à l'ère du Big Data

Depuis les début de l'informatique dans le monde de l'entreprise, l'Informatique Décisionnelle a toujours été une composante primordiale. L'Informatique Décisionnelle peut être décrite comme un processus axé sur la technologie qui permet d'analyser les données et de présenter des informations exploitables pour aider les cadres, les responsables et les autres utilisateurs finaux à prendre des décisions d'affaires éclairées. Cela englobe une grande variété d'outils, d'applications et de méthodologies qui permettent aux organisations de collecter des données à partir de systèmes internes et de sources externes. Ensuite, il s'agit de préparer ces données pour l'analyse en les "nettoyant" selon des procédés définis par des règles métiers. Au final, les analystes sont en charge d'éxécuter des requêtes sur ces données pour créer des rapports, des tableaux de bord et des visualisations de données mettant ainsi une information résumée et visuelle à disposition des décideurs.

Depuis plusieurs années et l'apparition de ce qui est communément appelé le Big Data (ou Méga Données), de nouvelles difficultés et complexités sont apparus dans les procédés habituels. L'utilisation massive et répandue des réseaux sociaux, des smartphones, mais aussi les objets connectés sont autant de facteurs favorisant une production de données toujours plus volumineuses. De plus, ces données sont souvent produites par des sources hétérogènes, proposant des formats divers et variés, ajoutant ainsi de la complexité dans l'intégration de données. Cette tendance touche aussi les entreprises gérant des volumes de données relativement faibles en interne, car ces dernières peuvent être tentées d'augmenter leur capital données en utilisant des données ouvertes [HB11] telles que les données météo, de circulation, données INSEE, ...

De nombreux outils ont été développés en parallèle pour faire face aux problématiques du Big Data mentionnées. L'outil le plus connu et le plus utilisé est certainement l'écosystème Hadoop [Bor07], basé sur un système de fichiers distribué sur un cluster de machines, permettant de distribuer aussi bien les données elles mêmes que les calculs effectués sur ces dernières. Hadoop est devenu, depuis plusieurs années, la référence en matière de technologie Big Data open source. De nombreux outils propriétaires ont d'ailleurs été développés par des entreprises autour de l'écosytème Hadoop.

Finalement, avec ces évolutions technologiques, de nouveaux paradigmes sont apparus, mettant à profit les nouvelles capacités offertes par les nouveaux outils.

Dans ce chapitre, nous passons en revue certaines des grandes tendances actuelles dans les pratiques actuelles en terme de gestion des méga données au sens large, mais aussi dans le contexte des entrepôts de données, à différentes étapes de la chaîne décisionnelle. Nous traitons notamment les avancées autour des datalakes (ou lac de données), ainsi que les nouvelles approches ETL, plutôt appelée ELT.

# 1 Du datawarehouse au datalake

Traditionnellement, les entrepôts de données ont été la référence en matière de stockage centralisé des données en vue de leur analyse. Ces dernières années, en réponse au volume et la variété des données dans un contexte Big Data, le concept de data lake a vu le jour, et se propose en tant que remplaçant plus complet des entrepôts traditionnels. Nous discutons ici l'état de l'art actuel au sujet des data lakes, autour des motivation, des avantages, mais aussi de leurs limites et challenges.

## 1.1 Généralités sur les data lakes

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.

### Le stockage des données dans le data lake

Dans les systèmes traditionnels, les données brutes sont stockées dans des systèmes opérationnels, très souvent cloisonnés. Puis, des schémas de transformation rigides permettent de construire des entrepôts de données dédiés au croisement et à l'analyse de données provenant de différentes sources. Ces entrepôts sont sensés contenir des données uniformisées, résultant d'une suite de processus de "nettoyage".

Avec les data lakes, la tendance consiste à *stocker toutes les données.* Il n'est plus question dans les data lake de stocker des données "finales", résultant par exemple d'un croisement de données ou d'une analyse. Il s'agit plutôt de stocker toutes les données brutes, avec l'hypothèse que n'importe lesquelles de ces données pourront nous être utile dans un avenir plus ou moins proche. Ainsi, les data lakes assurent qu'aucune possibilité d'analyse ne sera rendue impossible dans le futur dûe à l'absence de données. Les capacités de stockage quasi illimitées à un coût toujours plus bas, ainsi que l'apparition des architectures distribuées, tout cela a joué grandement en la faveur de ces nouvelles tendances.

### Vers un stockage unifié

Cependant, cette vision en silos de données ne convient plus à l'heure actuelle, ou le volume des données croit rapidement, et où les sources de données sont toujours plus nom-

breuses, mais aussi très variables. Les données proviennent de formats souvent différents tant dans la structure (fichier, sgbdr, ...) que dans le modèle des données.

L'idée des data lakes est d'effacer ce cloisonnement, et de 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.

## 1.2 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 calul, 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 des 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 noeuds 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.

## 1.3 Les challenges liés aux data lakes

Comme nous l'avons vu, les avantages des datalakes sont nombreux et couvrent des aspects importants. Cependant, les datalakes à l'heure actuelle ne sont pas encore parfaits, et font encore et toujours l'objet de recherche par rapport à de nombreux challenges à relever. Nous discutons ici quelques uns des challenges les plus importants.

Un des grands challenges liés aux datalakes concerne la **gouvernance des données** [Soa12]. En effet, si les datalakes permettent de stocker des grandes quantités de données, sous des formats différents, provenant de sources différents, alors il y a de forts risques que le datalake devienne un marécage de données [WA15]. Dans [Ott11], les auteurs mettent le doigt sur les aspects importants de la gouvernance des données, notamment l'organisation de l'information, la réglementation de l'accès aux données, la facilité d'accès aux données, ... Les auteurs proposent donc des directions de recherche pour inciter les différentes communautés à creuser ce sujet, qui selon eux est délaissé à tort au moment d'écrire leur papier.

La problématique de gouvernance a aussi été adressée dans le monde de l'entreprise. Par exemple, l'entreprise Teradata a cédé son datalake Kylo [Ter] en open source. Kylo est présenté comme un outil mettant en oeuvre les meilleures pratiques en terme de gouvernance de données. Teradata a d'ailleurs ouvert une branche dédiée à Kylo, nommée ThinkBig Analytics, qui lui permet de définir des bonnes pratiques de gouvernance par son expérience d'environ 150 clients parmi les plus grosses entreprises mondiales.

Kylo propose en effet des solutions pour s'attaquer aux principaux problèmes de gouvernance. Par exemple, pour la classification des données, Kylo donne une importance particulière aux metadata, reconnues comme des éléments importants dans les écosystèmes Big Data [Vem13]. Pour gérer l'organisation des données, Kylo permet de données des metadata aux données directement, mais aussi aux processus important les données, afin de pouvoir traquer les données. Il est possible d'assigner des données à des domaines particuliers, comme par exemple la comptabilité, les ressources humaines, etc... ou alors à des catégories, qui donnent une possibilité supplémentaire de rangement. En ce qui concerne l'accès aux données, Kylo indexe permet d'indexer les données que l'on souhaite, ainsi que les métadonnées associées, en utilisant le puissant moteur de recherche ElasticSearch [GT15]. Ainsi, il est possible d'effectuer des recherches par mots clés sur l'interface de Kylo pour se voir afficher une liste des (méta)données correspondantes. Par rapport à la sécurité d'accès, Kylo est capable de gérer des utilisateurs avec des profils et des droits d'accès différents. Il est aussi possible pour chaque donnée importée de spécifier le propriétaire des données. Ainsi, avec ces différentes possibilits, Kylo se positionne donc comme un datalake open source, et avec des fonctionnalités gérant permettant de favoriser la gouvernance des données au sein d'une organisation.

## 2 Le ETL traditionnel vs ELT du Big Data

Jusqu'à il y a peu, la stratégie ETL (Extract - Transform - Load) a été l'approche par excellence pour la construction d'entrepôts de données centralisés et "propres". Rappelons brièvement la signification de ces 3 étapes Extraction, Transformation et Chargement.

— L'étape d'**extraction** consiste en la récupération des données brutes d'un pool de

données non structuré et sa migration vers un référentiel de données temporaire

— L'étape de **transformation** consiste en la structuration, l'enrichissement et la conversion des données brutes pour correspondre au stockage cible

— L'étape de **chargement** consiste à charger des données structurées dans un entrepôt de données à analyser, utilisé au final par les outils d'analyse de l'Informatique Décisionnelle

Cependant, suite à l'apparition des problématique Big Data, et avec ces dernières les outils pour y répondre, de nouvelles approches ont vu le jour. L'approche qui retient aujourd hui beaucoup l'attention, et qui semble prometteuse et bien adaptée aux nouvelles possibilités, est appelée ELT [Ran09] pour (Extract - Load - Transform). Il s'agit d'une alternative mettant à profit les nouvelles capacités de stockage et de calcul des technologies Big Data.

Dans cette section, nous mettons en évidence les concepts importants et les avancées en terme d'ETL au sens large. Nous passons aussi en revue certains outils ayant vu le jour récemment et permettant justement de mettre en oeuvre ces dernières avancées.

## 2.1 L'approche ETL

L'approche ETL a longtemps été considérée comme l'approche traditionnelle pour la chaîne décisionnelle, depuis les données opérationnelles jusqu'au bases de données analytiques.

La philosophie de l'ETL est de **penser a priori les besoins** en analyse de l'entreprise, puis de figer ces besoins en les traduisant dans un modèle de données. Typiquement, un modèle est un constitué d'un ou plusieurs schémas multidimensionnels traduisant entre autre toutes les possibilités d'analyse, de croisement de données, que permettra l'entrepôt créé.

Une fois le modèle d'analyse défini à l'avance, toute la chaîne ETL est construite de telle sorte à récupérer les données sources, puis de les modifier jusqu'à convenir parfaitement au modèle d'analyse. Des compétences IT ainsi que des compétences métier sont nécéssaire tout au long de la chaîne ETL pour localiser et comprendre les données, puis effectuer les bonnes trasnformations.

En utilisant ETL, les analystes et les autres utilisateurs de BI se sont habitués à attendre, car le simple accès à l'information n'est pas disponible tant que tout le processus ETL n'est pas terminé. Un graphique décrivant le fonctionnement d'ETL : Extraire, Transformer, Charger

## 2.2 L'approche ELT

Le concept d'ELT est assez lié à la philosophie des datalakes. Dans l'approche ELT, une fois les données extraites, elles sont immédiatement chargées dans un référentiel de données unique et centralisé. Ce référentiel peut biensur être un datalake. 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 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 datalakes, 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.

## 2.3 Deux paradigmes pour un objectif commun

Il convient de souligner que ces 2 stratégies, ETL et ELT, ont pour objectif commun de répondre à un même besoin au sens large. D'importants volumes de données doivent être collectés, traités et analysés par les entreprises. Cependant, les données produites à la source sont sujettes à beaucoup de défauts, dûs par exemple à des valeurs manquantes, des valeurs erronées, ou autres. Pour obtenir des analyses correctes, les données doivent auparavant passer par une étape de nettoyage. Ensuite, afin d'obtenir des analyses plus riches et intéressantes, il est aussi possible d'effectuer une étape d'enrichissement, où par exemple les données sources pourraient être croisées avec des données ouvertes. Le "comment" est ce qui diffère entre les deux paradigmes. Les nouveaux outils Big Data ont ouvert la porte à de nouvelles possibilités techniques, et automatiquement de nouvelles stratégie de gestion des données, que ça soit au niveau de la gestion des données brutes, du moment auquel les données sont transformées, et aussi sur la manière dont les données sont analysées.

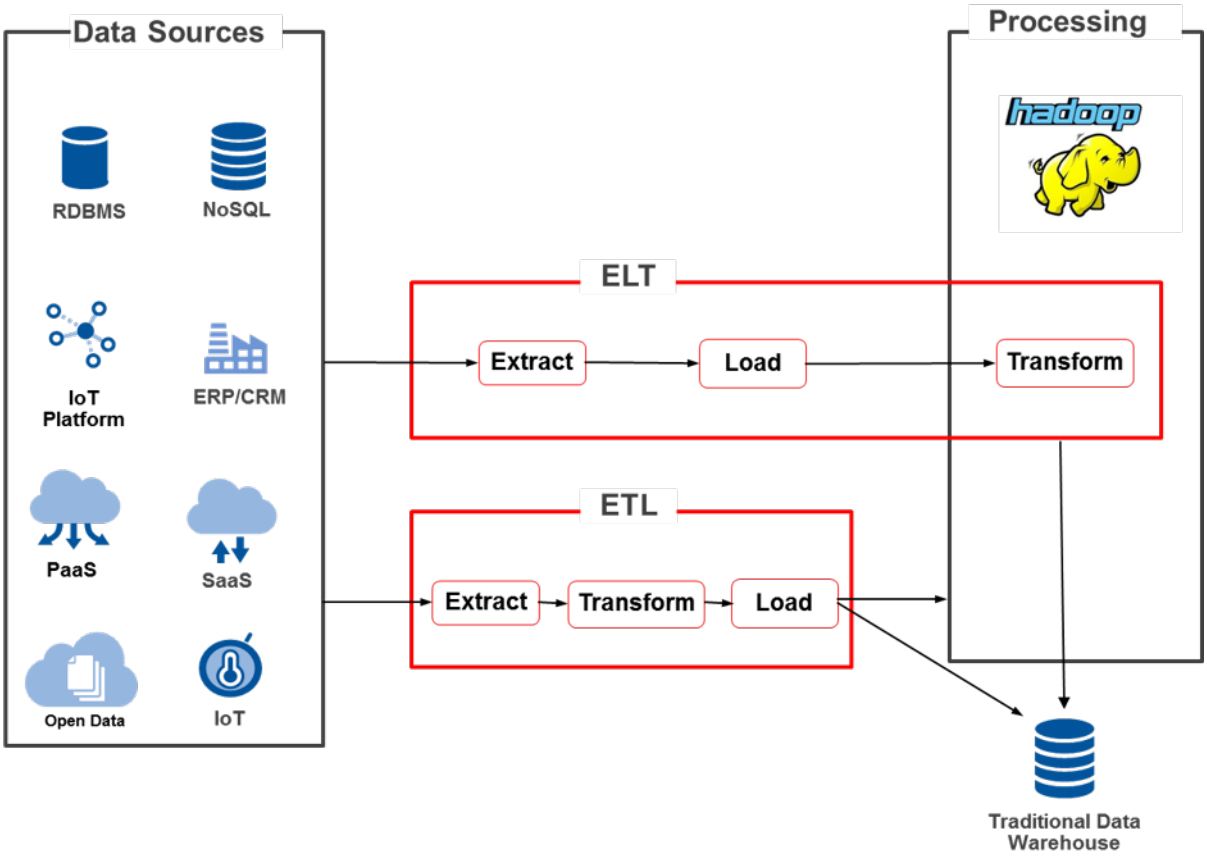


FIGURE 2.1 – Credit https ://www.gartner.com/document/3337317

Chapitre 3

# Nos propositons

Dans ce chapitre, nous présentons nos propositions d'architectures innovantes, basées exclusivement sur des technologies Big Data open source, pour effectuer la chaîne de transformation des données, depuis les sources opérationnelles, jusqu'aux systèmes analytiques. Notre proposition consiste en une architecture globale utilisant les nouveaux outils et paradigmes Big Data, notamment ceux qui sont liés aux data lakes. Ensuite, à partir de cette architecture globale, nous proposons plusieurs déclinaisons permettant de compléter cette architecture, notamment en ce qui concerne la partie utilisateur final, i.e. tout ce qui touche à l'analyse et la visualisation.

La suite de ce chapitre est organisée selon la logique décrite ci-dessus. Nous décrivons tout d'abord notre architecture globale, puis nous décrivons deux possibilités de déclinaisons. Notre proposition d'architecture étant relativement flexible, les déclinaisons que nous décrivons ne sont absolument pas exhaustives. Il serait tout à fait envisageable de construire d'autres déclinaisons.

## 1 Architecture globale

L'architecture globale que nous proposons permet de réaliser ce que l'on appelle communément ETL en Informatique Décisionnelle. Elle permet donc de prendre en charge les données à la source, puis de les faire passer à travers une suite de traitements, pour finalement aboutir à l'utilisation de données "propres". L'utilisation finale des données peut aller de la visualisation simple des données jusqu'à l'application d'algorithmes d'Intelligence Artificielle.

Comme le montre la figure 3.1, notre architecture comporte 3 grandes étapes, qui sont l'**ingestion des données**, la **transformation** des données, et l'**utilisation des données en self service**. Dans la suite de cette section, nous décrivons plus en détails les différentes étapes susmentionnées.

### 1.1 Ingestion des données

La première étape de notre architecture consiste à ingérer les données opérationnelles. Conformément aux principes émergents aves les datalakes, nous proposons d'ingérer toutes les données opérationnelles dont on a besoin, ou dont on pourrait avoir besoin. Pour

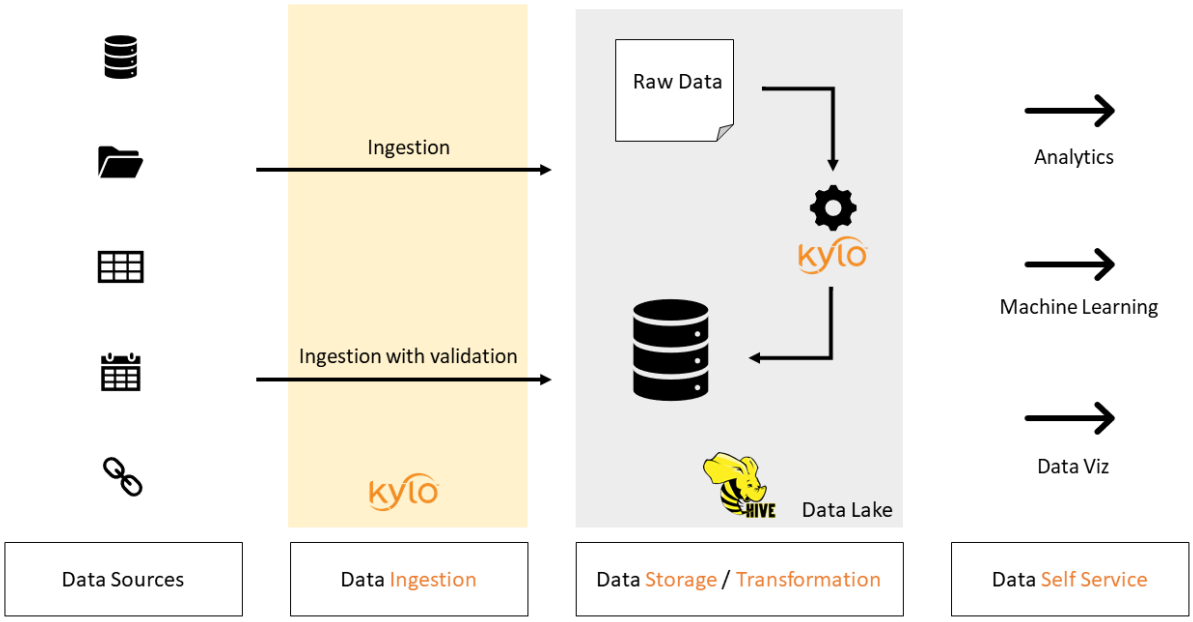


FIGURE 3.1 – Architecture globale de notre approche

l'ingestion des données, nous proposons d'utiliser l'outil Kylo décrit dans la section 2, lui même connecté à un cluster Hadoop, qui sera en charge du stockage et du traitement des données à l'étape suivante. Nous proposons de stocker toutes les données ingérées dans des tables Hive, qui est considéré comme l'entrepôt de données référence de l'écosystème Hadoop.

Kylo permet la validation des données à la volée, au moment de l'ingestion. De nombreuses possibilités sont disponibles dans Kylo pour spécifier, lors de l'ingestion, des typages de données ou même de la sémantique dans les données. Par exemple, il est possible de spécifier qu'un champ contiendra des adresses email, ainsi toutes les données dans ce champ seront scannées pour vérifier qu'elles répondent bien à un format d'adresse email. Il existe aussi des validateurs pour les dates, les numéros de carte de crédit. Les données détectées comme invalides sont isolées par Kylo, afin qu'il soit possible d'agir dessus par la suite, par exemple en les nettoyant manuellement.

Kylo permet aussi, grâce à Apache NiFi, de traquer les données afin de savoir de quelles sources elles proviennent, ainsi que la suite des transformations qu'elles ont subies pour arriver à une étape donnée du processus. Ainsi, dès l'ingestion, Kylo est capable de lier n'importe quelle donnée du datalake à la source dont elle provient.

## 1.2 Stockage et Transformation

Kylo est capable d'effectuer des transformations au moment de l'ingestion. Cependant, l'architectre que l'on propose n'utilise pas cette fonctionnalité, car l'on souhaite

une architecture utilisant toute la flexibilité des datalakes. En effet, une transformation des données dès l'ingestion impliquerait qu'il ne soit plus possible d'avoir accès, dans le datalake, à la donnée de base.

Une fois les données stockées dans le datalake, dans des tables Hive, il est possible de leur appliquer toute sorte de transformation. Le gros avantage d'appliquer les transformations dans le datalake est de bénéficier de la puissance de calcul disponible dans ce dernier. En effet, dans le cas d'un datalake basé sur un cluster Hadoop, toute la puissance du cluster Hadoop sera disponible pour les transformations. Non seulement les transformations sont plus performantes dans le datalake, mais le champs des transformations possibles est bien plus vaste, avec la possibilité d'utiliser n'importe quel autre outil de l'écosystème Hadoop.

Dans notre architecture, la transformation des données ne doit pas être faite de manière traditionnelle. Nous proposons plutôt un modèle flexible, que nous baptisons **data-on-demand**, dans lequel les transformations effectivement réalisées sont demandées par

Par exemple, pour un outil de visualisation donne, il faudra faire des trasnformations données pour correspondre à l'outil de visualisation...

Ensuite, l'utilisation de ces données, par exemple en les extrayant et retransformant, est possible afin par exemple d'ahérer précisément à un format correspondant à un outil d'analyse comme Kylin, Druid, ou encore correspondant à des algorithmes de Machine Learning, etc...

Le stockage est donc effectué dans Hivee. Les transformations nécéssaires sont ensuite effectuées dans le Data Lake, c'est à dire là où sont situées les données, et là où la puissance de calcul est dispoible.

## 1.3 Services de données

Les deux étapes précédentes sont des parties figées de l'architecture.

L'idée ici est que chaque application nécéssitant des données fasse la demande de transformations au niveau du Data Lake. Le data lake pourrait ainsi effectuer les transformations demandées et exposer les données ainsi calculées pour qu'elles puissent être accessibles. Techniquement, la manière d'exposer les données peut être diverse et variée, mais il semble que l'accès via des API rest soit une bonne solution. Il serait aussi possible de donner l'accès directement à certaines tables etc... Les règles d'accès ici sont définies au niveau de la gouvernance du data lake...

Le gros avantage des data lakes est de pouvoir stocker toutes les données dans un seul et même environnement. En plus de cela, la capacité de calculs des data lakes leur permet d'effectuer des traitements sur des masses de données importantes, et ce de manière efficace. Ainsi, une des bonnes manières d'utiliser les data lakes est le self service. Les données sont stockées, et ensuite nous pouvons effectuer les transformations immédiate-

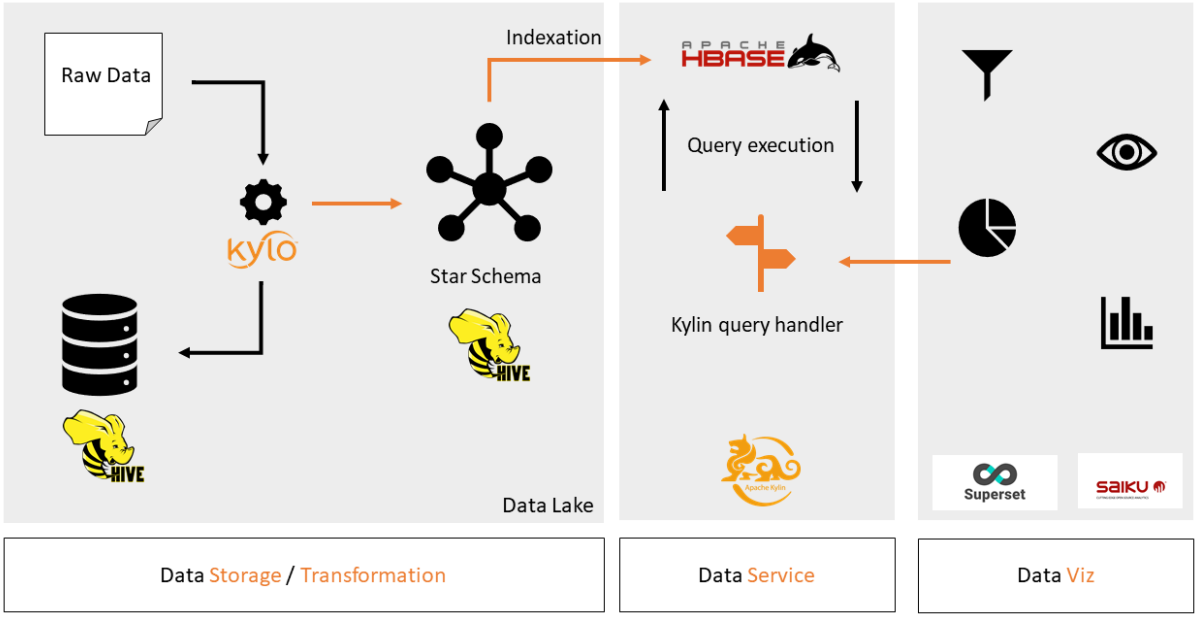


FIGURE 3.2 – Implémentation du service Kylin dans notre architecture

ment dans le data lake, pour par exemple produire un schéma de données cibles. Notre proposition est ainsi : une fois les données ingérées, nous proposons plusieurs chemins de transformations permettant d'aboutir à des représentations de données différentes, qui par la suite permettent l'utilisation d'outils différents à présenter à l'utilisateur.

## 2 Service Kylin

Dans cette section, nous présentons une proposition d'implémentation de service en utilisant l'outil Kylin, décrit à la section 2. Kylin [Yan+14] est un outil né dans la période Big Data, permettant initialement de faire de l'analyse OLAP dans l'écosystème Hadoop. L'approche de Kylin est une approche MOLAP, réalisée grâce aux nouvelles possibilités des outils Big Data. Concrètement, à partir d'un schéma en étoile déjà conçu, Kylin est capable de matérialiser le cube, de telle sorte à stocker directement les mesures résultant de croisements plutôt que de refaire la calcul pour chaque croisement au moment de la requête. Cette approche est typiquement "Big Data", du fait qu'elle privilégie la rapidité d'éxécution des requêtes par rapport à la quantité de stockage requise.

Notre idée ici est donc, à partir de nos données ingérées et nettoyées dans le data lake, de produire un schéma en étoile qui sera obtenu à partir de transformations des données existantes dans le datalake. Il est biensûr possible de créer plusieurs schéms en étoile indépendants, qui correspondront chacun à un besoin d'analyse. La figure 2 représente visuellement les différentes étapes du service Kylin, depuis le datalake jusqu'à la visualisation.

## 2.1 Définition des modèles d'analyse

La première étape consiste à définir les modèles d'analyse que nous souhaitons. Ce modèle doit nous permettre de définir clairement les croisements de données qui devront être réalisables. Le modèle, conformément aux attentes de Kylin, doit être composé d'un schéma en étoile à la manière des entrepôts ROLAP traditionnels.

Une fois le modèle d'analyse bien conçu, il s'agit de créer le modèle dans Kylin. Kylin offre une interface graphique permettant de spécifier des modèles multidimensionnels visuellement, en spécifiant les dimensions, hiérarchies, niveaux, mesures, ...

La prochaine étape consiste donc à préparer, au sein du datalake, les données selon le schéma en étoiles pour l'indexation par Druid.

## 2.2 Création des schémas en étoile

Une fois le modèle créé dans Kylin, l'idée est de préparer les données dans le datalake pour permettre une ingestion efficace par Kylin. La préparation des données consiste principalement en un croisement de plusieurs tables existantes, par le biais de jointures, pour au final obtenir le schéma en étoile désiré, peuplé des données correspondantes. Il est donc raisonnable d'effectuer cette préparation dans le datalake, qui lui dispose de la puissance de calcul.

Pour effectuer le croisement, nous utilisons les **Visual Query** de l'outil Kylo, qui est un outil puissant permettant de faire des requêtes complexes visuellement. Il est aussi possible, si besoin, de faire des réajustements comme par exemple pour le typage des données, le filtrage, la conversion de format, ... Kylo offre la possibilité de stocker le résultat de la visual query directement dans le datalake, ou alors de l'exporter dans d'autres formats ou bases de données. Nous décidons de stocker ce résultat dans le datalake.

## 2.3 Indexation dans Kylin

Pour l'indexation des données dans Kylin, nous choisissons de faire l'indexation des données à partir du schéma en étoiles stocké dans le datalake. Cette étape intermédiaire nous sert de zone tampon entre les données du datalake et Kylin. Cela permet notamment d'avoir toujours une version du schéma en étoile dans le datalake pour par exemple réagir en cas de corruption des données dans Druid, ou même pour d'autres cas d'utilisation.

L'idée est donc d'utiliser l'interface graphique fournie par Kylin pour éxécuter l'indexation des données.

## 2.4 Visualisation des données

Plusieurs outils sont disponibles pour la visualisation des données Kylin, donc certains sont open source. Pour le moment, nous avons identifié l'outil Tableau (référence vers

tableau) comme un des meilleurs candidats. Cependant, il s'agit d'un outil propriétaire et ne rentre donc pas dans nos critères de sélection open source.

L'outil Superset (référence) semble être une très bonne alternative. Il s'agit d'une application complète de visualisation de données massives développée initialement au sein de Airbnb, et transférée à la fondation Apache il y a peu. L'avantage de cette application est qu'elle est très complète en terme de possibilités de visualisation, et permet de travailler avec Kylin grâce notamment à la publication récente de l'outil kylinpy (reférence), qui est un client kylin écrit en python, permettant de faire le lien entre Superset et Kylin.

Enfin, il est aussi possible d'utiliser Mondrian avec des données stockées dans Kylin, en utilisant le driver Mondrian pour Kylin. L'idée est d'écrire des requêtes en MDX gérées par Mondrian, qui est capable de les soumettre directement à Kylin grâce à son driver dédié. Cette possibilité d'intégration avec Mondrian ouvre donc la possibilité d'utiliser des applications telles que Saiku pour exécuter des requêtes sur Mondrian.

# 3 Service Druid

Dans cette section, nous présentons une proposition d'implémentation de service en utilisant l'outil Druid, décrit à la section 2. Druid [Yan+14] est un outil puissant permettant de faire de l'analyse OLAP sur des données massives, avec une grande capacité à gérer les données temps réel. Contrairement à l'approche relationnelle, Druid travaille essentiellement avec des tables dénormalisées, sans donner d'importance à la redondance des données, qui est considérée comme non problématique avec les capacités de stockage actuelles. Le modèle de données géré par Druid est essentiellement composé d'un ensemble de colonnes correspondant à des caractéristiques d'un fait donné. Chaque fait correspond donc à une ligne de la table. Il est possible, pour chaque colonne, de définir si il s'agit d'une mesure ou d'une dimension. En se basant sur cela, Druid permet d'écrire aisément des requêtes permettant de calculer des valeurs de mesures au croisement de plusieurs dimensions, à la manière d'OLAP.

Notre idée ici est donc, à partir de nos données ingérées et nettoyées dans le data lake, de produire des tables dénormalisées contenant chaque information que l'on souhaite pouvoir utiliser dans notre analyse. Il est biensûr possible de créer plusieurs modèles, qui seront chacun instancié par une table dénormalisée indépendante. La figure 3 représente visuellement les différentes étapes du service Druid, depuis le datalake jusqu'à la visualisation.

## 3.1 Définition des modèles d'analyse

La première étape consiste à définir les modèles d'analyse que nous souhaitons. Ce modèle doit nous permettre de définir clairement les croisements de données qui devront

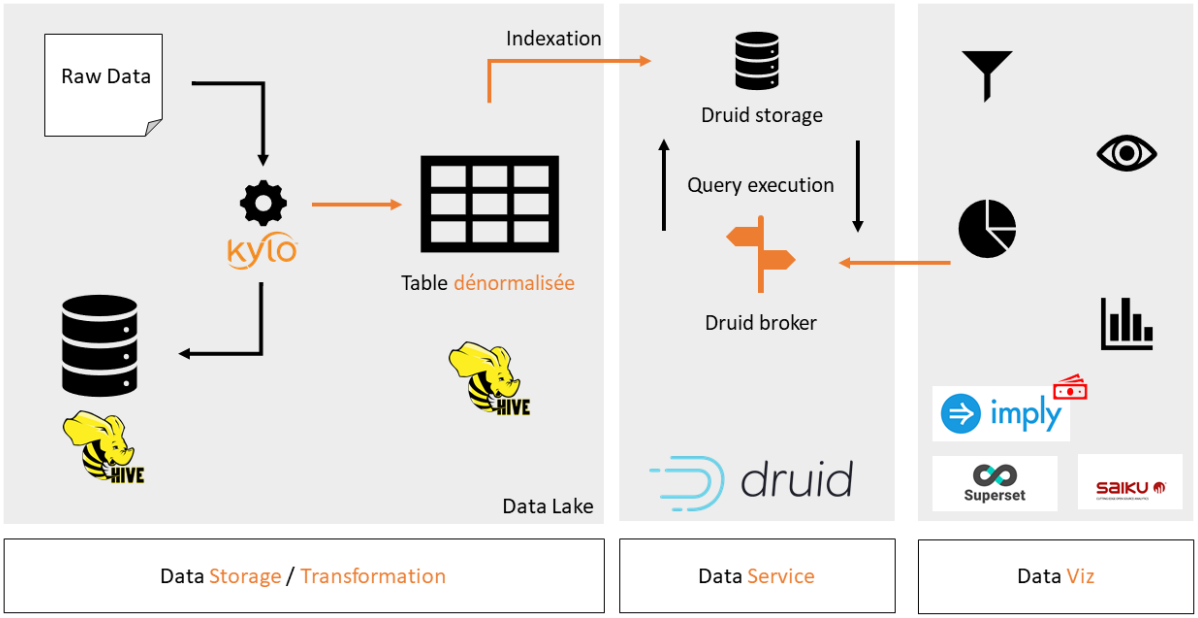


FIGURE 3.3 – Implémentation du service Druid dans notre architecture

être réalisables. Le modèle, conformément aux attentes de Druid, doit être composé d'un ensemble de colonnes, chacune ayant un type de données et étant définie comme mesure ou dimension.

Une fois le modèle d'analyse bien conçu, il s'agit de créer le modèle dans Druid. Druid donne effet la possibilité de créer des modèles en utilisant le format json. Il y a aussi la possibilité de créer des modèles visuellement en utilisant l'outil Imply Pivot, qui est une interface graphique spécialement conçue pour Druid.

La prochaine étape consiste à préparer les données pour l'ingestion dans Druid.

## 3.2 Création des tables dénormalisées

Une fois le modèle d'analyse créer dans Druid, l'idée est de préparer les données dans le datalake pour permettre une ingestion efficace par Druid. La préparation des données consiste principalement en un croisement de plusieurs tables existantes, par le biais de jointures, pour au final obtenir une table dénormalisée correspondant au modèle précédemment construit dans Druid. Il est donc raisonnable d'effectuer cette préparation dans le datalake, qui lui dispose de la puissance de calcul.

Pour effectuer le croisement, nous utilisons les **Visual Query** de l'outil Kylo, qui est un outil puissant permettant de faire des requêtes complexes visuellement. Il est aussi possible, si besoin, de faire des réajustements comme par exemple pour le typage des données, le filtrage, la conversion de format, ... Kylo offre la possibilité de stocker le résultat de la visual query directement dans le datalake, ou alors de l'exporter dans d'autres formats ou bases de données. Nous décidons de stocker ce résultat dans le datalake.

## 3.3 Indexation dans Druid

Pour l'indexation des données dans Druid, nous choisissons de faire l'indexation des données à partir de la table hive dénormalisée créée à l'étape précédente dans le datalake. Cette table intermédiaire nous sert de zone tampon entre les données du datalake et Druid.

L'idée est donc d'utiliser l'API REST de Druid pour effectuer l'indexation. Pour cela, il est nécéssaire d'écrire un programme personnalisé (par exemple un script), qui permette de lire les données dans Hive, puis de faire appel à l'API REST de Druid en demandant l'ingestion des données en cours de lecture. Il est important que le modèle dans Druid et la table dénormalisée construite dans le datalake coincident parfaitement. Cela permet d'écrire un programme générique qui fonctionnera dans tous les cas d'utilisation, qui fera tout simplement la bijection entre les colonnes de la table dénormalisée et celles du modèle Druid.

Il est à souligner qu'il existe d'autres possibilités, apparues récemment, pour l'intégration entre Druid et Hive, que nous étudierons dans un avenir proche.

## 3.4 Visualisation des données

Plusieurs outils sont disponibles pour la visualisation des données Druid, donc certains sont open source. Pour le moment, nous avons identifié l'outil Imply Pivot comme le meilleur outil pour la visualisation pour Druid, étant donné qu'il a été créé spécifiquement pour Druid, et par les concepteurs de Druid. Imply Pivot propose une interface ergonomique et très simple d'utilisation. Cependant, il s'agit d'un outil propriétaire et ne rentre donc pas dans nos critères de sélection open source.

Superset semble être une très bonne alternative. Il s'agit d'une application complète de visualisation de données massives développée initialement au sein de Airbnb, et transférée à la fondation Apache il y a peu. L'avantage de cette application est qu'elle est très complète en terme de possibilités de visualisation, et possède un **connecteur natif** pour Druid !

Enfin, il est aussi possible d'utiliser Mondrian avec des données stockées dans Druid, en utilisant l'outil Apache Calcite comme intermédiaire. L'idée est d'écrire des requêtes en MDX gérées par Mondrian, qui lui mêmes les transmet à Apache Calcite, qui lui même exécute les requêtes sur Druid. Cette possibilité d'intégration avec Mondrian ouvre donc la possibilité d'utiliser des applications telles que Saiku pour exécuter des requêtes sur Mondrian.

Chapitre 4

# Progression du travail

Dans cette section, nous décrivons la progression de notre travail. Comme nos objectifs reposent sur des considérations techniques élevées, nous avons passé beaucoup de temps à étudier et à mettre en place l'infrastructure dont nous avions besoin. Ici, nous décrivons d'abord toutes les infrastructures que nous avons mises en place jusqu'à présent, puis nous consacrons une section à la progression de la mise en œuvre de notre proposition ETL en temps réel.

En ce qui concerne le matériel sur lequel nous travaillons, nous utilisons un rack Dell avec 4 nœuds physiques, avec les caractéristiques suivantes.

## 1 Mise en place de l'infrastructure

Afin de pouvoir atteindre nos objectifs, nous avions besoin de mettre en place une plateforme complète, contenant notamment des clusters Hadoop. Fondamentalement, notre infrastructure est composée de trois grandes couches :

— 1) la couche OS

— 2) la couche cluster Hadoop

— 3) la couche application contenant toutes la partie applicative

Pour chacune de ces couches, nous avons dû faire les bons choix d'outils en fonction de nos besoins. Une fois les choix faits, nous avons effectivement installé et paramétrer ces outils. Dans cette section, nous allons décrire chacune de ces différentes couches. Un schéma général de l'infrastructure mise en place est présenté dans la figure 4.1.

### 1.1 OS et gestion du stockage

Pour la gestion des machines, nous avons utilisé le système d'exploitation Proxmox. Concernant la gestion du stockage, nous avons utilisé l'outil CEPH.

### 1.2 Installation de l'écosystème Hadoop

Pour installer l'écosystème Hadoop, nous avons utilisé Cloudera Manager Express (CDM). Sur la base de machines que l'on met à sa disposition, CDM est capable d'installer

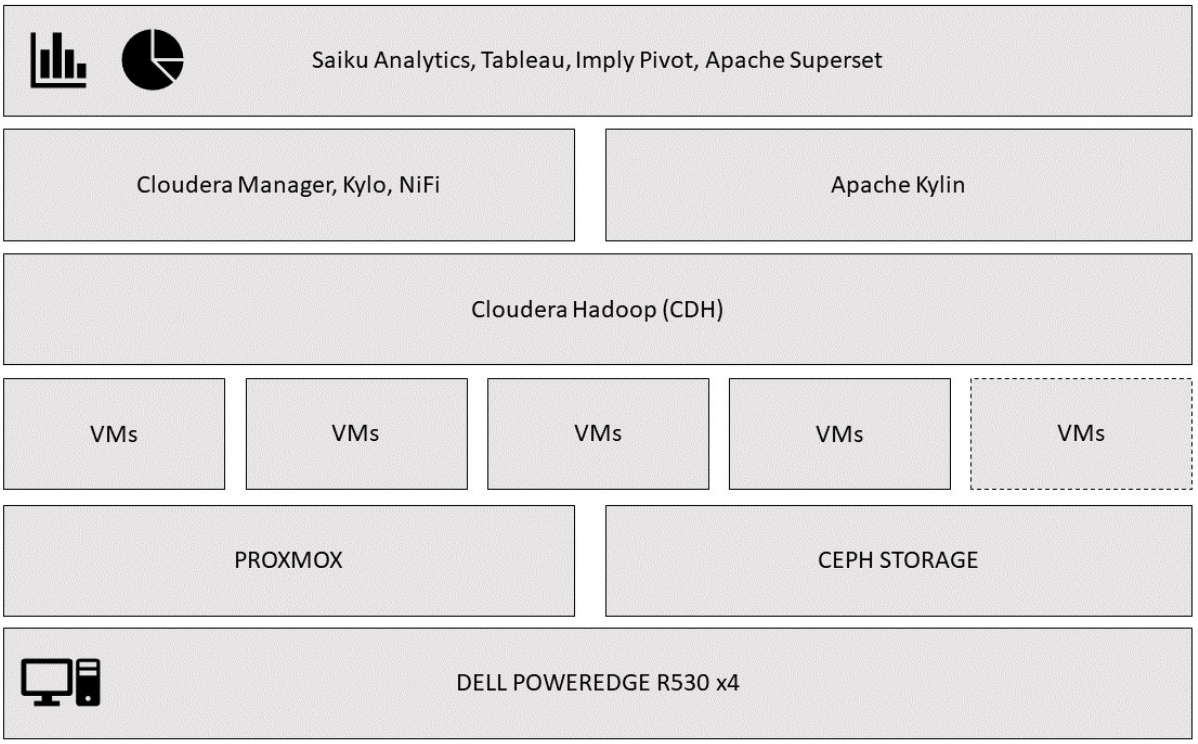


FIGURE 4.1 –

puis de gérer un ou plusieurs clusters Hadoop, avec une liste de services disponibles associés. CDM est bien entendu basé sur la distribution Hadoop de Cloudera, appelée CDH pour Cloudera Hadoop.

En utilisant CDM, l'installation complète d'un cluster Hadoop se fait entièrement via une interface graphique ergonomique. CDM propose aussi l'ajout de service à la volée, comme par exemples les outils Hive, Hue, ... Par ailleurs, CDM fournir aussi des rapports sur le fonctionnement du cluster ainsi que des alertes, sous forme de tableaux de bords.

## 1.3 Installation des applications

Une fois les clusters Hadoop installés, paramétrés et fonctionnels, nous avons procédé à l'installation des outils sélectionnés. Nous pouvons catégoriser les outils utilisés en 2 catégories. Certains outils sont applicatifs, dans le sens ou ils contiennent des fonctionnalités de traitement, d'analyse, ou plus généralement de la donnée. Ensuite, afin de tester et valider le rendu final, nous avons aussi installé des outils de visualisation, dont le rôle principal est de de gérer le rendu.

### Installation des outils applicatifs

Une fois l'infrastrcture prête, ainsi que le cluster Hadoop, nous avons installé d'autres outils open source.

Comme décrit dans le chapire 2, Kylo est un DataLake open source dont l'objectif est de faciliter l'acquisition de données, puis d'y appliquer des transformations afin d'obtenir des données en self-service. Kylo s'appuie sur Hadoop, il nécessite en effet : HDFS, Hive, Spark pour la transformation des données. Nous avons installé Kylo sur un edge node, c'est à dire une machine externe au cluster Hadoop, mais capable d'interagir avec dernier. Apache NiFi pour la gestion des pipelines de données, est installé sur la même machine que Kylo, pour faciliter l'interaction intensive entre ces deux outils.

Kylin et Druid étant deux composants importants, nous avons aussi installé chacun de ces outils sur edge node distinct. Ainsi, chacun de ces outils peut bénéficier de ses propres ressources sur sa machine, tout en pouvant attaquer le cluster Hadoop.

#### Installation des outils de visualisation

Au niveau de la visualisation, nous avons installé les outils suivants : Saiku, Tableau, Imply Pivot, et Apache Superset.

# 2 Implémentation de l'architecture

Notre architecture est basée sur deux grandes parties, qui sont 1) une partie d'ingestion permettant d'ingérer les données, les préparer, et les mettre à disposition de services de données et 2) une partie services où des services de données peuvent faire appel au datalake pour obtenir et transformer des données selon leur besoin. Dans cette section, nous présentons notre avancement sur chacune de ces deux parties. Nous présentons tout d'abord l'avancement sur la partie ingestion, puis ensuite sur chacun des deux services, Kylin et Druid, présentés au chapitre 3.

## 2.1 Définition des besoins en analyse

Avant d'entamer l'implémentation de l'architecture, nous définissons les besoins en analyse, qui sont notre fil directeur et nous permettent par la même occasion de produire notre implémentation sur un cas concret. Notre objectif est donc de permettre une analyse multivariée des demandes/renseignements demandés par les clients concernant le tourisme dans la région Poitou-Charentes. Pour cela, nous utiliserons comme source principale la base de données du CRM VTiger (voir section 4) qui lui tourne en production et permet de centraliser la gestion de toutes les demandes faites. Notre idée est de permettre aux analystes de pouvoir analyser les demandes selon des axes similaires aux axes de l'entrepôt existant (voir 4.2), mais qui lui est en ROLAP.

Au final, notre objectif est donc de pouvoir déboucher sur une analyse du nombre de demandes selon les dimensions suivantes :

— date de la demande (jour, mois, année)

— thème de la demande
— type de demandeur (couple, famille, personne seule, ...)
— origine du demandeur (ville, pays)

## 2.2 Ingestion et Transformation des données

Afin de pouvoir réaliser nos objectifs, la première étape consiste à ingérer les données dans le datalake. Pour l'ingestion, l'idée est d'ingérer intégralement la base de données de l'outil VTiger. Au moment de l'ingestion, nous procédons à la validatio à la volée des données grâce aux fonctionnalités de Kylo. Il s'agit dans cette étape d'uniformiser les données ingérées, mais aussi et surtout de les typer, de les étiqueter, etc... de telle sorte à ce qu'elles soient traquées et donc facilement retrouvables. D'ailleurs, l'ingestion est aussi suivie d'une étape d'indexation, où le moteur de recherche ElasticSearch [GT15] indexe les données et leurs métadonnées permettant ainsi de les retrouver aisément.

## 2.3 Implémentation du service Kylin

Pour fonctionner en natif, Kylin a besoin de travailler sur la base d'un schéma en étoile. Or, le schéma en étoile ici n'est pas présent a priori dans le datalake, qui contient surtout les données brutes. Nous devons donc définir notre modèle de schéma en étoile, de telle sorte à répondre aux besoins en analyse énoncés.

### Définition du schéma en étoile

Le schéma en étoile que l'on propose est proche de celui utilisé dans l'entrepôt ROLAP existant, car il s'agit en effet des mêmes paradigmes de modélisation. Pour l'expérimentation, nous utilisons un schéma allégé, qui est comme suit :

— une table de dimension date contenant jours, mois et année.
— une table de dimension thème
— une table de dimension type pour le type de demandeur
— une table de dimension origine pour l'origine géographique du demandeur (ville, pays)
— Une table de fait enregistrant pour chaque demande un identifiant, ainsi que des liens vers les différentes tables de dimension

### Transformations

La prochaine étape, après la définition du schéma en étoile, est de créer effectivement les différentes tables mentionnées dans le datalake. Nous avons réalisé cette étape en utilisant la fonctionnalité Visual Query de Kylo. La principale difficulté dans cette étape relève

de la localisation des données, c'est à dire de savoir quelles tables/colonnes correspondent à la donné recherchée. Cette étape, ainsi que l'étape suivante, sont à l'heure actuelle en cours de réalisation.

#### Indexation dans Kylin

Une fois les tables du schéma en étoile créées dans le datalake, l'indexation par Kylin est aisée. Il s'agit simplement d'utiliser l'interface fournie par Kylin, puis de lui indiquer où se trouvent la table de faits, les tables de dimensions, et les colonnes impliquées dans les dimensions. Il ne s'agit pas ici d'un verrou scientifique, mais d'une simple étape dans notre architecture pour le moment.

## 2.4 Implémentation du service Druid

Pour fonctionner en natif, Druid a besoin de travailler sur la base d'un modèle de table dénormalisée. Nous devons donc définir notre modèle de table conforme à Druid, de telle sorte à répondre aux besoins en analyse énoncés.

#### Définition de la table dénormalisée

Etant donné que Druid fonctionne principalement sur les séries temporelles, il requiert que chaque table ait un champ date. Nous avons donc défini notre modèle de table dénormalisée pour représenter les demandes comme suit, chaque item représentant une colonne de notre table :

- id de la demande
- date de la demande (timestamp)
- jour de la demande (1, . . . , 31)
- mois de la demande
- année de la demande
- thème de la demande
- type du demandeur
- ville de la demande
- pays de la demande

#### Transformations

La prochaine étape, après la définition du modèle précédent, est de créer effectivement cette table dans le datalake. Nous avons réalisé cette étape en utilisant la fonctionnalité Visual Query de Kylo. La principale difficulté dans cette étape relève de la localisation

des données, c'est à dire de savoir quelles tables/colonnes correspondent à la donné recherchée. Pour une utilisation optimale des transformations, la personne en charge des transformations à la demande devrait avoir des compétences métier lui permettant ainsi de localiser aisément les données.

#### Indexation des données

Pour le moment, la procédure d'indexation des données est en phase d'expérimentation. En effet, comme nous l'avons vu à la section 3, il existe plusieurs possibilités d'intégration entre Druid et Hive. Celle qui semble adaptée à nos besoins pour le moment consiste à utiliser directement dans Hive le gestionnaire de stockage Druid. Ainsi, toutes les données ajoutées dans une telle table Hive seront en fait directement stockées et gérées par Druid.

## 3 Documentation

documentation de l'installation, l'utiliation et la maintenance des différents outils installés est disponile sous forme de wiki

Chapitre 5

# Conclusion & Travaux à venir

Dans ce document, nous avons traité des différents problèmes auxquels l'Informatique Décisionnelle faisait face à l'ère du Big Data. Nous avons aussi passé en revue les différentes avancées technologiques permettant justement d'accompagner cette tendance. Ces nouvelles technologiques, dont la plus connue est Hadoop, ont radicalement changé la manière de gérer les données. En effet, dans ce nouveau monde, la masse des données n'est plus forcément une problématique, mais il convient maintenant de savamment gérer ces données, puis en extraire le jus.

Afin de pouvoir commencer à travailler sur les problématiques de recherche, nous avons du passer un certain temps à étudier, puis installer, l'infrastructure nécéssaire. Jusqu'à présent, nous avons installé les différents systèmes d'exploitation permettant la virtualisation des machines, une solution CEPH de stockage résilient, puis un cluster Hadoop de travail. Nous avons ensuite installé tous les outils nécéssaires (Kylo, Kylin, Druid, ...) en suivant les bonnes pratiques des SI.

Concernant les problématiques de recherche, nous avons étudié les nouvelles avancées dans le monde du Big Data, et étudié comment elles pouvaient être mises en oeuvre pour améliorer les processus de l'Informatique Décisionnelle, et notamment répondre aux problématiques actuelles. Nous avons vu aussi que les datalakes, bien que prometteurs, ne sont pas encore forcément matures. La solution de Teradata, Kylo, qui est d'ailleurs open source depuis 2017, semble être la solution la plus adaptée à nos besoins. Nous avons proposé, en nous basant sur Kylo, une architecture globale orientée services de gestion des données depuis la source jusqu'à leur analyse. L'architecture que l'on propose à l'avantage d'être flexible, en permettant d'ajouter des nouvelles possibilités d'analyse sans avoir à modifier en amont les processus d'ingestion et de transformation des données.

Les différentes étapes à venir seront les suivantes. Dans un avenir proche, nous terminerons l'implémentation des deux services proposés (Kylin et Druid), puis nous publierons une étude comparative de ce deux outils sur un jeu de données connu, comme par exemple celui du benchmark SSB [OOC07]. A moyen terme, nous avons l'intention de proposer une extension de notre architecture pour gérer l'analyse de données temps réel. En effet, pour le moment, les données sont ingérées à des intervals réguliers mais distants dans le temps (en batch). Les outils actuels sont assez puissants pour réaliser un "ETL" temps réel depuis les sources jusqu'à l'outil d'analyse. Même avant le Big Dat, l'ETL temps réel a été une préoccupation au niveau de la recherche [VS09], dû à l'avantage certain que cela

procurerait aux organisations, de pouvoir analyser en temps réel et donc réagir en temps réel.

# Bibliographie


[Bar13] Mike Barlow, *Real-time big data analytics : Emerging architecture*, " O'Reilly Media, Inc.", 2013.

[Bor07] Dhruba Borthakur, « The hadoop distributed file system : Architecture and design », *in* : *Hadoop Project Website* 11.*2007* (2007), p. 21.

[GT15] Clinton Gormley et Zachary Tong, *Elasticsearch : The Definitive Guide : A Distributed Real-Time Search and Analytics Engine*, " O'Reilly Media, Inc.", 2015.

[HB11] Noor Huijboom et Van den Broek, « Open data : an international comparison of strategies », *in* : *European journal of ePractice* 12.*1* (2011), p. 4–16.

[NRD18] Iuri Nogueira, Maram Romdhane et Jérôme Darmont, « Modeling Data Lake Metadata with a Data Vault », *in* : *22nd International Database Engineering & Applications Symposium (IDEAS 2018)*, ACM, 2018.

[OOC07] Patrick E O'Neil, Elizabeth J O'Neil et Xuedong Chen, « The star schema benchmark (SSB) », *in* : *Pat* 200 (2007), p. 50.

[Ott11] Boris Otto, « A morphology of the organisation of data governance. », *in* : *ECIS*, t. 20, 1, 2011, p. 1.

[Ran09] Vikas Ranjan, *A comparative study between ETL (Extract, Transform, Load) and ELT (Extract, Load and Transform) approach for loading data into data warehouse*, rapp. tech., viewed 2010-03-05, http ://www. ecst. csuchico. edu/˜ juliano/csci693/Presentations/2009w/Materials/Ranjan/Ranjan. pdf, 2009.

[Soa12] Sunil Soares, *Big data governance : an emerging imperative*, Mc Press, 2012.

[Ter] Teradata, *Kylo datalake*, `https://kylo.io/`, Accessed : 2018/07/16.

[Vas09] Panos Vassiliadis, « A survey of extract–transform–load technology », *in* : *International Journal of Data Warehousing and Mining (IJDWM)* 5.*3* (2009), p. 1–27.

[Vem13] Gautham Vemuganti, « Metadata Management in Big Data », *in* : *Big Data : Countering Tomorrow's Challenges* 3 (2013).

[VS09] Panos Vassiliadis et Alkis Simitsis, « Near real time ETL », *in* : *New Trends in Data Warehousing and Data Analysis*, Springer, 2009, p. 1–31.

[WA15] Coral Walker et Hassan Alrehamy, « Personal data lake with data gravity pull », *in* : *Big Data and Cloud Computing (BDCloud), 2015 IEEE Fifth International Conference on*, IEEE, 2015, p. 160–167.

[Yan+14] Fangjin Yang et al., « Druid : A real-time analytical data store », *in* : *Proceedings of the 2014 ACM SIGMOD international conference on Management of data*, ACM, 2014, p. 157–168.